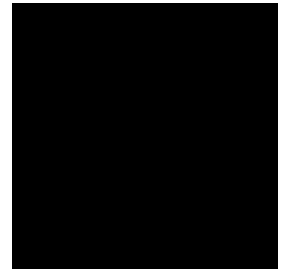


Assembly Reference



Version 2.0



GEOS Software Development Kit Library

Version 2.0

Assembly Reference



Initial Edition, Unrevised and Unexpanded

Geoworks, Inc.
Alameda, CA



Geoworks provides this publication "as is" without warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability or fitness for a particular purpose. Geoworks may revise this publication from time to time without notice. Geoworks does not promise support of this documentation. Some states or jurisdictions do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

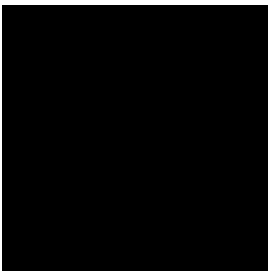
Copyright © 1994 by Geoworks, Incorporated.
All rights reserved. Published 1994
Printed in the United States of America

Geoworks®, Geoworks Ensemble®, Ensemble®, GEOS®, PC/GEOS®, GeoDraw®, GeoManager®, GeoPlanner®, GeoFile®, GeoDex® and GeoComm® are registered trademarks of Geoworks in the United States and Other countries.

Geoworks® Pro, PEN./GEOS, Quick Start, GeoWrite, GeoBanner, GeoCrypt, GeoCalc, GeoDOS, Geoworks® Writer, Geoworks® Desktop, Geoworks® Designer, Geoworks® Font Library, Geoworks® Art Library, Geoworks® Escape, Lights Out, and Simply Better Software are trademarks of Geoworks in the United States and other countries.

Trademarks and service marks not listed here are the property of companies other than Geoworks. Every effort has been made to treat trademarks and service marks in accordance with the United States Trademark Association's guidelines. Any omissions are unintentional and should not be regarded as affecting the validity of any trademark or service mark.

Contents



- 1 **Parameters File Keywords** 7
- 2 **Routines**..... 19
- 3 **Structures**..... 399

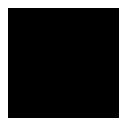


Parameters File Keywords



Keywords used in the **.gp** file of an geode are shown in alphabetical order in this section. These keywords define how the Glue linker will link the geode.

1





■ appobj

appobj <name>

The **appobj** field indicates the name of the application object. All geodes with *appl* set under **type** (see above) must have an **appobj** entry. The *name* argument should be the name of the object of **GenApplicationClass** specified in the application's **.goc** file.

■ class

class <name>

The **class** field specifies the name of the object class to be bound to the geode's process thread. This field has significance only if **process** is specified in the geode's **type** field (see below). This should be the same as the **ProcessClass** object designated in the **.goc** file (see the Hello World sample for an example of this connection). Note that this class binding will only be for the geode's first (primary) thread.

■ driver

driver <name> [noload]

This field specifies another driver that is used by this geode. The *noload* flag indicates that the used driver does not need to be loaded when the geode is first launched. Most applications and libraries will not use exported routines from drivers, so few geodes will use this field. (Notable exceptions are those geodes that access serial and parallel ports—those geodes will include the serial or parallel driver.)

■ entry

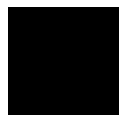
entry <name>

This field is used by library geodes. The *name* argument is the name of the library routine to be called by the kernel when the library is loaded or unloaded and when a program using the library is loaded or unloaded.

■ exempt

exempt <library-name>

If you wish to exempt a certain library from Glue's platform checking, call it out with the exempt keyword. Glue will not complain if you then use parts of the library not normally available with platforms named in your **platform** statement.



■ **export**

export *<name>* [*as <name2>*]

This field identifies routines usable by geodes other than the one being compiled; these routines are “exported” for use by other programs. Both forms create entry point symbols for the routines. The first *name* argument must be the actual name of the routine. If the second, optional, *name2* argument is included, then other programs will call that routine using the second name rather than the original. This allows a routine to have a different global name than that used by its creator geode.

This field is also used to export classes defined in a **.goc** or **.goh** file. See Hello World for an example of this usage.

■ **incminor**

incminor [*<name>*]

The **incminor** directive is used at the end of a library’s **.gp** file before new routines are added (after a release of the library has already been made). After this release, new **export** and **publish** directives will be put after this **incminor** directive. The **incminor** directive causes two things: First, the geode’s minor protocol number gets incremented by one. Second, any geode that uses your library will depend only on the higher minor protocol number if it actually uses one or more of the entry points exported after the **incminor** directive.

Any number of **incminor** directives may be used in a given **.gp** file. The major and the base minor numbers still come from a **.rev** file, if one exists.

The *name* argument is optional; it may be used in conjunction with the **protominor compiler** directive. Glue will know that the structures marked with the **protominor** label should be associated with the revision represented by the **incminor** directive.

■ **library**

library *<name>* [*noload*]

This field specifies another library that is used by this geode. The *noload* flag indicates that the used library does not need to be loaded when the geode is first launched (though symbolic information will be loaded in any case). Note that every geode must have the line

`library geos`

included in the **.gp** file. Most will also have the following line:

`library ui`

Routines

Any number of used libraries may be specified.

■ **load**

```
load      <name> [ "<class>" ] as [ <name2> ] [ <align> ] [ <combine> ] \
           [ "<class2>" ]
```

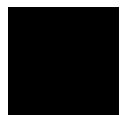
The **load** field is used when you want to alter the way a segment is linked for your geode. This is especially useful, for example, when integrating another company's runtime routines into your application or library; their segments may correspond to specifications other than yours.

Every segment read in has a given name, class, alignment, and combination type. These are described below (the **load** parameters appear after):

name	This is the actual name of the segment being loaded in. Segments with the same name are treated as one continuous segment.
class	Segments with the same class name are always loaded together into memory regardless of their order in the geode's source code. Class names in the load directive must always be enclosed in quotation marks.
align	This specifies the alignment type of the segment—on what type of address the segment can start. Possible alignment settings are byte, word, double word, paragraph, and page.
combine	Segments with the same name may appear in different code modules. The <i>combine</i> parameter specifies how these segments are to be combined when loaded. The combine type may be one of the following (see your assembly reference manual for more information): COMMON, PRIVATE, PUBLIC, STACK, or RESOURCE.

The parameters for load are listed below. Only the first is necessary, to inform Glue which segment is to undergo the alterations. For an example of using the load statement, see below.

name	This represents the actual original name of the segment. It is a necessary parameter so Glue knows which segment's linkage is to be altered.
class	This is the original class name of the segment. It must be enclosed in quotation marks if given. If you do not need to change the class, this parameter is unnecessary.
name2	This is the new name of the segment, if any.
align	This specifies the new align type of the segment, if any.



Goc Keywords

12

- combine** This specifies the new combine type of the segment, if any.
- class2** This specifies a new class name for the segment, if any is required. If you do not need to change the class, this parameter is unnecessary. The new class must be in quotation marks.

Examples:

```
load _NAME_ "CODE" as CODE word public
load _NAME_ "CODE" as DATASEG para common "DATA"
```

■ longname

longname "<*string*>"

The **longname** field designates a 32-character name for the geode. This name will be displayed with the geode's icon by GeoManager; all geodes should be given a long name.

■ name

name <*pname*>.<*ext*>

The **name** field in the parameters file gives the geode a permanent name which will be used by both the Glue linker and the Swat debugger. Every geode must have a permanent name. Note that the *pname* argument must be no more than eight characters, and the *ext* argument must be no more than four. Additionally, the *ext* argument may not be "appl," as that is reserved.

When Glue is linking an error-checking geode, it drops the fourth character of *ext* and adds "ec" to the end of *pname*.

■ nosort

nosort

This keyword should appear before the list of resources. Normally glue will sort the geode's resources to optimize their arrangement. This keyword turns off that sorting. If you will generate .GYM (generic symbol) files for your geode, you should use the nosort option, as it will be important that all versions of your geode order their resources in the same way. If you won't generate .GYM files, you probably don't want to use this option.

■ platform

platform <*name*>

The platform directive specifies that the Geode is compatible with the named system. This gives a sign of how backwards-compatible the application is. If multiple platforms are specified, Glue will make sure that the major protocol

Routines

numbers for each of the libraries it finds within the platforms match. Having done that, it will use the smallest minor protocol number it can find for each library to ensure compatibility across all platforms.

If a reference is ever made to an entry point in a library that would cause the executable to depend upon a later version of the library than specified in the platform file, glue will complain. For example, if the specified platform used GrObj version 534.1 and glue found a reference to an entry point that didn't exist until GrObj 534.3 (i.e., an entry point exported following 3 'incminor's in grobj.gp), glue will spit out an error message like:

```
error: file "somegeode.gp", line 59: Usage of
NewGrObjRoutine requires grobj minor protocol 3, but
platform files only allow minor protocol 1
```

If the new routine happens to be a “published” routine, glue will copy it into the geode in an effort to avoid the error.

■ publish

```
publish <name>
```

Normally, If a geode is required to run (via platform specifications) with a version of a library that doesn't contain one of the entry points required by the geode, glue will notify the user of the inconsistency, and the link will fail. However, if that entry point happens to be a published routine, glue will actually copy the routine into the geode and switch the call over to the newly copied routine to remove the dependency on the library routine. Glue does this by copying any routines marked “publish” in a library's .gp file into the .ldf file, then copying them out into whatever other geodes needs when those geodes are linked. Routines are marked “publish” by replacing the word “export” with the word “publish” in the .gp file, like so:

```
publish PublishedRoutinei
```

The published routines appear in .ldf files in individual segments named after the routine (e.g. _PUBLISHED_PublishedRoutine), each containing a routine, also named after the published routine (e.g., _PUBLISHED__PUBLISHED_PublishedRoutine) You'll know one of these routines has been linked into your geode by examining the resource summary output by glue:

Resource	Size	# Relocs
CoreBlock	0	0
dgroup	240	8
_PUBLISHED_GROBJCALCCORNERS	53	1
_PUBLISHED_GROBJBODYPROCESSALLGR	94	2

Routines



Goc Keywords

14

TEST2_E	478	27
INTERFACE	652	1
CHANGETEXTDIALOG	232	1
APPRESOURCE	416	1

■ resource

resource <name> (read-only|preload|discardable|fixed|conforming|shared|\code|data|lmem|discard-only|swap-only|ui-object|object|\no-swap|no-discard)+

The **resource** field indicates to Glue that the geode uses the named resource. Not all resources used by a geode must be declared here, however. (Resources are described in more detail in "GEOS Programming," Chapter 5.) Resources must be designated with the proper attributes, all of which are listed below:

(none) If no attribute is specified, the resource named becomes a private data resource for the geode.

read-only The resource block may not be modified by the program.

preload The resource block should be loaded when the geode is first launched.

discardable The resource block may be discarded from memory if necessary.

fixed The resource block should reside in fixed memory.

conforming The resource block, if containing code, may be called from a lower privilege level. If containing data, it may be accessed from a lower privilege level. (This applies only in protected mode and is not currently implemented.)

shared The resource block may be used by other geodes. (Note: It is an error to specify *code* and *shared* without *read-only*.)

code The resource block contains executable code.

data The resource block contains data only. If a data resource is designated *read-only* and not fixed, it is assumed to be discardable.

lmem The resource block consists of a local memory heap. This implies the attribute *data* (above), though not the condition pertaining to being discardable.

discard-only The resource block should not be swapped but may be discarded. This is useful for initialization code.

swap-only The resource block should not be discarded but may be swapped.

Routines

- ui-object** The resource block contains objects to be run by the UI. This implies *lmem*, *shared*, and *no-discard*. All blocks for a geode designated *ui-object* will be run in a UI thread created specifically for the geode's UI objects.
- object** The resource block contains objects that are to be run by the application's process thread rather than by the UI. This implies *lmem* and *no-discard*.
- no-swap** The resource block will not be swappable.
- no-discard** The resource block will not be discardable.

Because most resources are code resources, standard code does not have to be declared in the parameters file. Code resources default to *code*, *read-only*, and *shared*. However, if the resource is named in the **.gp** file, the default is overridden in favor of the settings presented. This fact is useful primarily when programming in assembly—in C, code resources are not declared explicitly.

The Hello World sample application uses only standard code resources (undeclared) and UI resources (designated *ui-object*). Some other examples are listed below:

- ◆ **Shared data**

```
resource <name> data shared
```
- ◆ **Initialization code**

```
resource <nm> code shared read-only preload no-swap
```
- ◆ **Common code used by several geodes (this is the default)**

```
resource <name> code shared read-only
```
- ◆ **Self-modifying code (strongly discouraged)**

```
resource <name> code
```

■ **stack**

stack <number>

The **stack** field designates the size of the application's stack in bytes. The default stack size is 2000 bytes. This field is not necessary for geodes unless they require a different size stack (the Hello World sample uses a slightly smaller stack size for example only). The **stack** field is valid only for geodes with a process aspect.



Goc Keywords

16

■ tokenchars

tokenchars "<string>"

This is one of two fields that identifies a unique token in GeoManager's token database file (see **tokenid**, below). The **tokenchars** field must be a string of four characters that identifies the geode's token. Note that these characters also appear in the geode file's extended attributes.

■ tokenid

tokenid <number>

This is the other of two fields that identifies a unique token in GeoManager's token database file (see **tokenchars**, above). It must be a number corresponding to the programmer's manufacturer ID number. Note that this number also appears in the geode file's extended attributes.

■ type

type (process|driver|appl|library)+ [single] [system] [uses-coproc]\
[needs-coproc] [has-gcm] [c-api]

The **type** field in the parameters file designates certain characteristics of the geode being compiled. These attributes correspond to the **GeodeAttrs** type and determine how the Glue linker will put the geode together. The attributes are as follows:

- | | |
|--------------------|---|
| process | This attribute indicates the geode has its own thread. Applications should always have process specified in the type field. |
| driver | This attribute indicates the geode has a driver aspect. |
| appl | This attribute indicates the geode has an application aspect. |
| library | This attribute indicates the geode has a library aspect. |
| single | This geode may only have one copy running at a time. Some applications may allow multiple copies to be running at once; they should not specify single as a type attribute. |
| system | This attribute is set for drivers that must be exited specially and must always be exited. For example, a swap driver has special exit conditions that must always be met and is therefore a system driver. |
| uses-coproc | This attribute is set if the geode will make use of a math coprocessor if one is available. Note that if the geode with this |

Routines

attribute set is a library, all applications that use the library will inherit the property. This attribute is used to indicate that the coprocessor's state must be saved during a context switch.

needs-coproc

This attribute indicates that the geode must have a math coprocessor to run. (This implies *uses-coproc*, above).

has-gcm

This attribute indicates that the application being compiled has a GCM (appliance) version. This information is used by Welcome to locate all GCM applications.

c-api

This attribute indicates the library entry points are written in C so the kernel must call them with C calling conventions.

■ **usernotes**

usernotes "<string>"

This field specifies text to be put in the **.geo** file's usernotes field. The text must be within quotation marks and can be up to 100 characters long. It must contain no line breaks. This can be useful for containing copyright notices in the executable files. The user can read the text in the usernotes by using GeoManager's File/Get Info command.

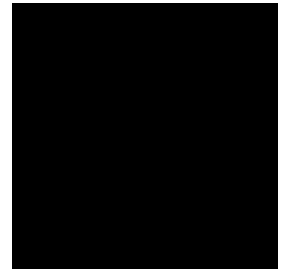


Goc Keywords

18

 **Routines**

Routines



2



■ ArrayQuickSort

Sort the given array using a modified quicksort algorithm.

Pass:	ds:si	Address of the first element in the array.
	ax	Size of each element (all of uniform size).
	cx	Number of elements in the array.
	ss:bp	Address of an inheritable QuickSortParameters structure.
	bx	Value to pass to callback routine specified in ss:bp .
Returns:	Nothing.	
Destroyed:	ax, cx, dx	
Library:	chunkarr.def	

■ CellDirty

Mark a cell as dirty.

Pass:	es	Segment address of block containing the cell.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	cell.def	

■ CellGetDBItem

Get the group and item numbers of the database item associated with the specified cell.

Pass:	ds:si	Address of CellFunctionParameters structure.
	ax	Row number of cell.
	c1	Column number of cell.
Returns:	CF	Set if the item exists, clear otherwise.
	ax	Group number of the cell.
	di	Item number of the cell.
Destroyed:	Nothing.	
Library:	cell.def	

■ CellGetExtent

Get the extent (bounds) of the current sheet of spreadsheet cells.

Pass:	ds:si	Address of CellFunctionParameters structure.
-------	--------------	---



CellLock

■ 22

	ss:bx	Address of a RangeEnumParams structure. The caller does not need to set any values in the structure.
Returns:	ss:bx	Address of the RangeEnumParams structure; the <i>REP_bounds</i> field will be filled in with the extent of the spreadsheet. If there is no current spreadsheet, all bounds will be set to -1.
Destroyed:	Nothing.	
Library:	cell.def	

■ CellLock

Lock a cell's data to examine or change it. The cell should not be locked while also working with other cells; the caller should lock the cell, copy the data, then unlock it with **CellUnlock**.

Pass:	ds:si ax c1	Address of CellFunctionParameters structure. Row number of cell. Column number of cell.
Returns:	CF *es:di	Set if the item exists, clear otherwise. Segment:chunk handle of the cell data if cell exists.
Destroyed:	di , unless it is returned.	
Library:	cell.def	

■ CellReplace

Replace a cell's data with new data.

Pass:	ds:si ax c1 es:di dx	Address of CellFunctionParameters structure. Row number of cell. Column number of cell. Address of new data. Size of data pointed to, or zero to free the cell.
Returns:	Nothing.	
Destroyed:	Possibly es , if it pointed to a database item in the same file.	
Library:	cell.def	

■ CellUnlock

Unlock a cell previously locked with **CellLock**.

Pass:	es	Segment address of block containing cell data.
Returns:	Nothing.	

Assembly Reference

Destroyed: Normally, nothing. If using the error-checking kernel and segment error-checking is active, then if either DS or ES is pointing to a block that has become unlocked, that register will be set to NULL_SEGMENT.

Library: **cell.def**

■ CheckForDamagedES

When using the error-checking version of the ui, this routine checks the ES register to make sure it points to a valid LMem block. This comes in handy in code where *es:xx should point to an object.

In a non-error-checking environment, this routine does nothing.

Pass: **es** Alleged local memory block handle.

Returns: Nothing (flags preserved as well).

Destroyed: Nothing.

Library: **ui.def**

■ ChunkArrayAppend

Append a new element to the end of a chunk array.

Pass: ***ds:si** Segment:chunk handle of the chunk array.
ax Size of new element, if variable-sized.

Returns: **ds:di** Address of new, locked element.

Destroyed: Nothing.

Library: **chunkarr.def**

Warning: This routine may resize or move the LMem block, invalidating all pointers to chunks or elements within it.

■ ChunkArrayCreate

Create a new general chunk array with no elements.

Pass: **ds** Global handle of block for new array.
bx Element size (zero for variable-sized elements).
cx Size for **ChunkArrayHeader**, or zero for default. Extra space is initialized to zeroes.
si Chuck to resize and use for chunk array (zero to allocate a new chunk).
al **ObjChunkFlags** to be passed to **LMemAlloc**.

Returns: ***ds:si** Address of the new array, locked.

Destroyed: Nothing.



ChunkArrayDelete

■ 24

Library: **chunkarr.def**

Warning: This routine may resize or move the passed block, invalidating all segment pointers to it.

■ ChunkArrayDelete

Delete a specified element from the given array.

Pass: ***ds:si** Segment:chunk handle of the chunk array.
 ds:di Address of locked element.

Returns: **ds:di** Address of the same element, if it still exists.

Destroyed: Nothing.

Library: **chunkarr.def**

■ ChunkArrayDeleteRange

Delete a range of elements from the given chunk array.

Pass: ***ds:si** Segment:chunk handle of the chunk array.
 ax First element to delete (inclusive).
 cx Total number of elements to delete (-1 to delete to the end of
 the array).

Returns: Nothing.

Destroyed: Nothing.

Library: **chunkarr.def**

■ ChunkArrayElementResize

Resize an element in a variable-sized chunk array.

Pass: ***ds:si** Segment:chunk handle of the chunk array.
 ax Element number of the element to be resized.
 cx New size of the element.

Returns: Nothing.

Destroyed: Nothing.

Library: **chunkarr.def**

Warning: If you are resizing the element larger, all pointers you've stored to chunks in
the block are invalidated. If you are resizing the element smaller, the array
is guaranteed not to move or cause other chunks to move.

■ ChunkArrayElementToPtr

Return the address of a specified element in a chunk array.

Assembly Reference

Pass:	*ds:si ax	Segment:chunk handle of the chunk array. Element number of element to find.
Returns:	CF cx ds:di	Set if element number out of bounds. Size of the returned element, if variable-sized. Address of element; if ax was out of bounds, ds:di will be returned pointing to the last element in the array.
Destroyed:	Nothing.	
Library:	chunkarr.def	
Warning:	The error-checking version fatal-errors if passed CA_NULL_ELEMENT in ax .	

■ ChunkArrayEnum

Process all elements in a chunk array, calling a callback routine for each.

Pass:	*ds:si bx:di ax cx, dx, bp, es	Segment:chunk handle of the chunk array. Address of the callback routine. Initial data passed to callback <i>only</i> if fixed-size elements. Initial data to pass to callback routine.
Returns:	CF ax, cx, dx, bp, es	Set if enumeration aborted by the callback routine. As set by the last call to the callback routine.
Destroyed:	bx	
Callback Routine Specifications:		
Passed:	*ds:si ds:di ax cx, dx, bp, es	Segment:chunk handle of the chunk array. Address of element being processed. Size of element, if variable-sized; otherwise, inherited from ChunkArrayEnum . Inherited from ChunkArrayEnum .
Return:	CF cx, dx, bp, es	Set to abort processing. Data to pass to next enumeration.
May Destroy:	bx, si, di	
Library:	chunkarr.def	

■ ChunkArrayEnumRange

Process the specified elements in a chunk array, calling a callback routine for each element.

Pass:	*ds:si bx:di ax	Segment:chunk handle of the chunk array. Address of the callback routine. Number of first element to process.
-------	--	---



ChunkArrayGetCount

■ 26

	cx	Number of elements to process (-1 to process all including the last element).
	dx, bp, es	Initial data to pass to the callback routine.
Returns:	CF	Set if the routine was aborted before all specified elements were processed.
	ax, cx, dx, bp, es	As returned by the last call to the callback routine.
Destroyed:	Nothing.	
Callback Routine Specifications:		
	Passed:	*ds:si Segment:chunk handle of the chunk array.
		ds:di Address of element being processed.
		ax Size of element, if variable-sized; otherwise, inherited from ChunkArrayEnumRange .
		cx, dx, bp, es Inherited from ChunkArrayEnumRange .
	Return:	CF Set to abort processing.
		cx, dx, bp, es Data to pass to next enumeration.
	May Destroy:	bx, si, di
Library:	chunkarr.def	

■ ChunkArrayGetCount

Return the number of elements in the given chunk array.

Pass:	*ds:si	Segment:chunk handle of the chunk array.
Returns:	cx	Number of elements in the array.
Destroyed:	Nothing.	
Library:	chunkarr.def	

■ ChunkArrayGetElement

Get an element given its element number.

Pass:	*ds:si	Segment:chunk handle of the chunk array.
	ax	Element number of element to be retrieved.
	cx:dx	Address of a buffer in which the element will be returned.
Returns:	ax	Size of element returned.
	cx:dx	Address of filled buffer
Destroyed:	Nothing.	
Library:	chunkarr.def	
Warning:	The error-checking version will fatal-error if CA_NULL_ELEMENT is passed in ax or if ax is out of bounds.	

Assembly Reference

■ ChunkArrayInsertAt

Insert the specified element at a given position in the array.

Pass:	*ds:si	Segment:chunk handle of the chunk array.
	ds:di	Address of element to insert.
	ax	Size of new element, if variable-sized.
Returns:	ds:di	Address of new element in the array.
Destroyed:	Nothing.	
Library:	chunkarr.def	
Warning:	This routine may resize or move the passed block, invalidating all segment pointers to it.	

■ ChunkArrayPtrToElement

Return the element number of the element pointed to.

Pass:	*ds:si	Segment:chunk handle of the chunk array.
	ds:di	Address of the element to be checked.
Returns:	ax	Zero-based element number.
Destroyed:	Nothing.	
Library:	chunkarr.def	

■ ChunkArraySort

Sort the given chunk array in ascending order.

Pass:	*ds:si	Segment:chunk handle of the chunk array to sort.
	bx	Value to pass to callback routine.
	cx:dx	Address of callback routine.
Returns:	Nothing.	
Destroyed:	cx, dx	
Callback Routine Specifications:		
Passed:	bx	Value inherited from ChunkArraySort .
	ds:si	Address of first element to compare.
	es:di	Address of second element to compare.
Return:	SF, OF, ZF	These flags should be set in the following conditions:
		first element less than second element
		SF set
		OF clear
		ZF unrestricted



ChunkArrayZero

■ 28

first element equal to second element

SF unrestricted

OF unrestricted

ZF set

first element larger than second element

SF clear

OF clear

ZF clear

May Destroy: **ax, bx, cx, dx, si, di**

Library: **chunkarr.def**

■ ChunkArrayZero

Free all the elements of the given chunk array and resize it.

Pass: ***ds:si** Segment:chunk handle of the chunk array.

Returns: Nothing.

Destroyed: Nothing.

Library: **chunkarr.def**

■ ClipboardAbortQuickTransfer

This routine aborts a quick-transfer. This routine is normally used if the quick-transfer source object is about to be destroyed or if an error occurs trying to register the quick-transfer item.

Pass: Nothing.

Returns: Nothing.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardAddToNotificationList

Add the passed OD to the transfer notify list.

Pass: **cx:dx** OD to add. If **cx** is the process handle, then **dx** must be zero.

Returns: Nothing.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardClearQuickTransferNotification

This routine removes the quick-transfer OD notification.

Assembly Reference

Pass: **bx:di** Notification OD to remove.
Returns: Nothing.
Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardDoneWithItem

This routine must be called when you are finished using the requested transfer item.

Pass: **bx:ax** Transfer item header, as returned by **ClipboardQueryItem**.
Returns: Nothing.
Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardEndQuickTransfer

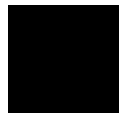
End a quick-transfer. Reset the mouse pointer image, clear the quick-transfer region (if any), and clear the quick-transfer item. Send out notification if necessary.

Pass: **bp** **ClipboardQuickNotifyFlags**. If a quick-transfer move operation was done, then **CQNF_MOVE** should be set. If a quick-transfer copy operation was done, then **CQNF_COPY** should be set. If the item was not accepted, the **CQNF_NO_OPERATION** should be set.
Returns: Nothing.
Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardEnumItemFormats

This routine returns the list of all available formats (**ClipboardItemFormatID** structures).

Pass: **bx:ax** Transfer item header, as returned by **ClipboardQueryItem**.
cx Maximum number of formats to return.
es:di Buffer for formats (should be at least **cx * sizeof(ClipboardItemFormatID)**).
Returns: **cx** Number of formats returned.
es:di (unchanged) Buffer now filled.



ClipboardFreeItem

■ 30

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardFreeItem

Free the passed clipboard item.

Pass: **bx:ax** **VMFileHandle:VMBlockHandle** of item to free.

Returns: Nothing.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardFreeItemsNotInUse

Frees normal or quick transfer item if nobody's using it, nukes references to it, sends proper GCN messages out.

Pass: Nothing.

Returns: Nothing.

Destroyed: Nothing.

Library: **clipbrd.def**

Warning: The user should not have called **ClipboardRegisterItem** with this transfer item, as this routine just frees the data without updating any references in the map block.

■ ClipboardGetClipboardFile

Return the VM file used to hold clipboard items.

Pass: Nothing.

Returns: **bx** Handle of UT's clipboard VM file.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardGetItemInfo

Get more information about the passed transfer item.

Pass: **bx:ax** Transfer item header, as returned by **ClipboardQueryItem**.

Returns: **cx:dx** Handle:chunk of *CIH_sourceID* from clipboard item header.

Destroyed: Nothing.

Assembly Reference

Library: **clipbrd.def**

■ ClipboardGetNormalItemInfo

Return normal clipboard item information.

Pass: Nothing.

Returns: **bx** Clipboard VM file handle.
ax Clipboard VM block handle.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardGetQuickItemInfo

Return quick clipboard item information.

Pass: Nothing.

Returns: **bx** Clipboard VM file handle.
ax Clipboard VM block handle.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardGetQuickTransferStatus

Check to see if a quick-transfer is in progress.

Pass: Nothing.

Returns: **ZF** Clear if quick-transfer is in progress; set otherwise.
ax If a quick transfer is in progress, this will be **ClipboardQuickTransferFlags** indicating what stage the process is in.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardGetUndoItemInfo

Return undo clipboard item information.

Pass: Nothing.

Returns: **bx** Clipboard VM file handle.
ax Clipboard VM block handle.

Destroyed: Nothing.

Library: **clipbrd.def**



ClipboardHandleEndMoveCopy

■ 32

■ ClipboardHandleEndMoveCopy

This routine handles a MSG_META_END_COPY, either preparing to finish the quick-transfer and send a MSG_META_END_MOVE_COPY to the object with the active grab or ending the quick-transfer and sending a MSG_META_END_OTHER to the object with the implied grab.

Pass:	bx	Zero to send a MSG_META_END_OTHER; non-zero to send a MSG_META_END_MOVE_COPY.
	bp	High byte is a UIFunctionsActive structure.
	CF	Should be set clear. Set to check if quick-transfer is in progress (this is needed only for internal input handling).
Returns:	ax	MSG_META_END_OTHER or MSG_META_END_MOVE_COPY (as determined by passed value in bx).
Destroyed:	Nothing.	
Library:	clipbrd.def	

■ ClipboardQueryItem

This routine registers the passed transfer item.

Pass:	bp	ClipboardItemFlags (all but CIF_QUICK will be ignored).
Returns:	bp	Number of formats available (zero if no clipboard item).
	cx:dx	Owner of clipboard item.
	bx:ax	VMFileHandle:VMBlockHandle to clipboard item header (which may then be passed to ClipboardRequestItemFormat).
Destroyed:	Nothing.	
Warning:	After calling this routine, ClipboardDoneWithItem must be called.	
Library:	clipbrd.def	

■ ClipboardRegisterItem

This routine registers the passed transfer item.

Pass:	ax	Handle of VM block containing ClipboardItemHeader structure or zero to null the clipboard item.
	bx	Handle of VM file containing clipboard item.
	bp	ClipboardItemFlags .
Returns:	CF	If registering a quick transfer item, this flag's behavior is undefined. If registering a normal transfer item, this flag will be set on an error, clear otherwise.

Assembly Reference

Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardRemoteReceive

Receive clipboard from remotely connected machine.

Pass: Nothing.
Returns: CF Set on error, clear on success.
Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardRemoteSend

Send clipboard to remotely connected machine.

Pass: Nothing.
Returns: CF Set on failure, clear on success.
Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardRemoveFromNotificationList

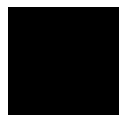
Remove the passed OD from the transfer notify list.

Pass: **cx:dx** OD to remove. If **cx** is the process handle, then **dx** must be zero.
Returns: CF Clear if successfully removed, set if was not found.
Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardRequestItemFormat

This routine requests the given transfer item, as stored in the given format type.

Pass: **bx:ax** Transfer item header, as returned by **ClipboardQueryItem**.
cx:dx Format manufacturer:format type.
Returns: **bx** File handle of transfer item.
ax:bp VM chain (zero if none).
cx First extra data word.
dx Second extra data word.



ClipboardSetQuickTransferFeedback

■ 34

Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardSetQuickTransferFeedback

Set mouse cursor for quick-transfer.

Pass: **ax** **ClipboardQuickTransferFeedback** value.
bp If ax is CQTF_MOVE or CQTF_COPY, then the high byte of this register should hold a UIFunctionsActive value. UIFA_MOVE signals that you wish to force the cursor to be that associated with a quick-move; UIFA_COPY that you wish to force the quick-copy cursor.

Returns: Nothing.
Destroyed: Nothing.
Library: **clipbrd.def**

■ ClipboardStartQuickTransfer

Initiate a quick transfer (normally called from MSG_META_START_MOVE_COPY).

Pass: **si** **ClipboardQuickTransferFlags**. The CQTF_COPY_ONLY flag should be set if the source only supports copying. The CQTF_USE_REGION flag should be set if you will be passing a region to use as the quick-transfer cursor. The CQTF_NOTIFICATION flag should be set if the source wants notification when the quick-transfer item is accepted by the destination.

ax Initial cursor to use (CQTF_MOVE or CQTF_COPY). This should be -1 if you wish to use the default cursor (i.e. object is a quick-transfer source, but not a quick transfer destination).

cx, dx If CQTF_USE_REGION set, these registers hold the mouse position in screen coordinates. Otherwise they are ignored.

bx:di If CQTF_NOTIFICATION set, these registers hold the OD to receive MSG_NOTIFY_QUICK_TRANSFER_MOVE, MSG_NOTIFY_QUICK_TRANSFER_COPY, and MSG_META_CLIPBOARD_NOTIFY_QUICK_TRANSFER_FEEDBACK.

ss:bp If CQTF_USE_REGION set, values will be passed on stack.

Pass on stack: The following structure will only be passed if CQTF_USE_REGION is set:
ClipboardQuickTransferRegionInfo struct
CQTRI_paramAX word
CQTRI_paramBX word

Assembly Reference

CQTRI_paramCX word
 CQTRI_paramDX word
 CQTRI_regionPos Point
 CQTRI_strategy dword
 CQTRI_region dword ; pointer to region
 ClipboardQuickTransferRegionInfo ends

CQTRI_region must be in a block that is in memory already.

CQTRI_strategy should be a video driver strategy. To find out the strategy of the video driver associated with your window, send your object a MSG_VIS_VUP_QUERY with VUQ_VIDEO_DRIVER. Pass the handle thus gained to **GeodeInfoDriver**, which will return the strategy.

Returns: CF Clear if UI part of new quick-transfer successfully begun; set if a quick-transfer was already in progress.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardTestItemFormat

This routine determines whether the clipboard item supports the specified format.

Pass: **bx:ax** Transfer item header, as returned by **ClipboardQueryItem**.
cx:dx Format manufacturer:format type.

Returns: CF Clear if format supported; set otherwise.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ClipboardUnregisterItem

This routine unregisters the passed clipboard item, restoring any transfer which may have been disturbed by the last normal clipboard item.

Pass: **cx:dx** Owner output descriptor used when registering previous item.

Returns: Nothing.

Destroyed: Nothing.

Library: **clipbrd.def**

■ ConfigBuildTitledMoniker

Global routine to build a titled moniker based on the passed moniker list



ConfigBuildTitledMonikerUsingToken

■ 36

Pass: `*ds:si` visual moniker list.
Returns: Nothing; visMoniker list replaced with visMoniker.
Destroyed: Nothing.
Library: **config.def**

■ ConfigBuildTitledMonikerUsingToken

Combine 2 vis monikers—placing the text moniker centered below the picture moniker.

Pass: `ds` Local memory block in which to create the new moniker.
`ax:bx:si` Token characters.
Returns: `CF` Clear if found; set otherwise.
`*ds:dx` New moniker.
Destroyed: `ax, bx, cx, di`
Library: **config.def**

■ DBAlloc

Allocate a new database item in a specified group.

Pass: `bx` File handle of the database file.
`ax` Group identifier of the new item (VM block handle).
For an ungrouped item, pass DB_UNGROUPED.
`cx` Size of the new item.
`ds` Optional segment address of an item-block.
`es` Optional segment address of an item-block.
Returns: `di` Item number of newly allocated item.
`ax` Group number of new item.
`ds` (If passed) fixed up.
`es` (If passed) fixed up.
Destroyed: Nothing.
Library: **dbase.def**

■ DBCopyDBItem

Copy an existing database item into a newly-allocated item.

Pass: `bx` File handle of the source database file.
`ax` Group identifier of the source database item.
`di` Item number of the source database item.
`bp` File handle of the destination database file.

Assembly Reference

	cx	Group identifier of the destination item's group. For and ungrouped item, pass DB_UNGROUPED.
Returns:	di	Item number of the new item.
	ax	Group identifier of the new item.
Destroyed:	Nothing.	
Library:	dbase.def	
Warning:	Because a new chunk is allocated, chunks or blocks may be moved. Thus, all pointers are invalidated by this routine.	

■ DBDeleteAt

Delete a given number of bytes from within the specified database item.

Pass:	bx	File handle of the database file.
	ax	Group identifier of the item.
	di	Item number of the item.
	dx	Offset of first byte to be deleted.
	cx	Total number of bytes to be deleted.

Returns: Nothing.

Destroyed: Nothing.

Library: **dbase.def**

■ DBDirty

Mark a database item as dirty so it will be written to the database file with its changes.

Pass:	es	Segment of locked block containing the database item.
--------------	-----------	---

Returns: Nothing.

Destroyed: Nothing.

Library: **dbase.def**

■ DBFree

Remove the specified item from the database.

Pass:	bx	File handle of the database file.
	ax	Group identifier of the item.
	di	Item number of the item to be freed.

Returns: Nothing.

Destroyed: Nothing.



DBGetMap

■ 38

Library: **dbase.def**

■ DBGetMap

Return the item that is set to be the database's map.

Pass:	bx	File handle of the database file.
Returns:	ax	Group identifier of the map item's group.
	di	Item number of the map item.
Destroyed:	Nothing.	
Library:	dbase.def	

■ DBGroupAlloc

Create a new database group.

Pass:	bx	File handle of the database file.
Returns:	ax	Group identifier of the new group.
Destroyed:	Nothing.	
Library:	dbase.def	

■ DBGroupFree

Remove all items in the specified group and delete the group.

Pass:	bx	File handle of the database file.
	ax	Group identifier of the group to be deleted.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	dbase.def	

■ DBInsertAt

Insert a specified number of bytes within a given database item. The bytes may be inserted at any offset, and the new bytes will be zeroed.

Pass:	bx	File handle of the database file.
	ax	Group identifier of the item.
	di	Item number of the item.
	cx	Total number of bytes to insert.
	ds	Optional segment address of an item-block.
	es	Optional segment address of an item-block.
Returns:	ds	(If passed) fixed up.
	es	(If passed) fixed up.

Assembly Reference

	si	Old segment address of changed item block.
	ax	New segment address of changed item block.
Destroyed:	Nothing.	
Library:	dbase.def	
Warning:	Because the chunk is resized larger, chunks or blocks may be moved. Thus, all pointers are invalidated by this routine.	

■ DBLock

Lock a database item for exclusive access. When you're done with the item, unlock it with **DBUnlock**.

Pass:	bx	File handle of the database file.
	ax	Item's group number.
	di	Item's item number.
Returns:	*es:di	Segment:chunk handle of database item.
Destroyed:	Nothing.	
Library:	dbase.def	

■ DBLockMap

Lock the map item for the database file. This is a utility that is slightly quicker than calling **DBGetMap** followed by **DBLock**. When finished with the map item, you must call **DBUnlock** on it.

Pass:	bx	File handle of the database file.
Returns:	*es:di	Segment:chunk handle of the locked map item.
	di	Zero if there is no map item. In this case, es is not returned.
Destroyed:	Nothing.	
Library:	dbase.def	

■ DBReAlloc

Change the size of an existing database item.

Pass:	bx	File handle of the database file.
	ax	Group identifier of the item (VM block handle).
	di	Item number of the item to be reallocated.
	cx	New size of the item.
	ds	Optional segment address of an item-block.
	es	Optional segment address of an item-block.
Returns:	ds	(If passed) fixed up.
	es	(If passed) fixed up.



DBSetMap

■ 40

Destroyed: Nothing.
Library: **dbase.def**

■ DBSetMap

Mark a database item as being the map item for the database file.

Pass: **bx** File handle of the database file.
ax Group identifier of the item's group.
di Item number of the item to be made the map.
Returns: Nothing.
Destroyed: Nothing.
Library: **dbase.def**

■ DBUnlock

Unlocks a database item that had previously been locked with **DBLock**.

Pass: **es** Segment address of the item's item-block.
Returns: Nothing.
Destroyed: Nothing. (Error-checking code may destroy ds or es by writing NULL_SEGMENT to it, if it pointed to a block that had become unlocked.)
Library: **dbase.def**

■ DiskCheckInUse

Determine if the passed disk is actively being used, either by an open file or by a thread having a directory on the disk in its directory stack.

Pass: **bx** Disk handle of disk to be checked.
Returns: **CF** Set if disk is in use, clear if it is not.
Destroyed: Nothing.
Library: **disk.def**

■ DiskCheckUnnamed

Check if the passed disk handle refers to an unnamed disk (i.e. a disk that has no user-supplied volume name).

Pass: **bx** Disk handle of disk to be checked.
Returns: **CF** Set if disk is unnamed, clear if it is named.
Destroyed: Nothing.

Assembly Reference

Library: **disk.def**

■ DiskCheckWritable

See if the passed volume is writable.

Pass: **bx** Disk handle of volume to be checked.
 Returns: **CF** Set if the volume is writable, clear if it is not.
 Destroyed: Nothing.
 Library: **disk.def**

■ DiskCopy

Copies the contents of the source disk to the destination disk, prompting for them as necessary.

Pass: **dh** source drive number
d1 destination drive number
al **DiskCopyFlags**
cx:bp callback routine
 Returns: **ax** **DiskCopyError**/FormatError; Zero if successful.

Callback Routine Specifications:

Passed: **ax** DiskCopyCallback value signalling what to do next.
bx, dx Additional information based on value of ax:
 DCC_GET_SOURCE_DISK
d1 = Zero-based drive number.
 DCC_REPORT_NUM_SWAPS
dx = Number of swaps required.
 DCC_GET_DEST_DISK
d1 = Zero-based drive number.
 DCC_VERIFY_DEST_DESTRUCTION
bx = Disk handle of destination disk
d1 = Zero-based drive number
 DCC_REPORT_FORMAT_PCT
dx = Percentage of disk formatted.
 DCC_REPORT_READ_PCT
dx = Percentage of disk read.
 DCC_REPORT_WRITE_PCT
dx = Percentage of disk written.

Return: Zero to continue, non-zero to abort.
 Destroyed: Nothing.
 Library: **disk.def**



DiskFind

■ 42

■ DiskFind

Search the list of registered disks and return the handle of that having the passed volume name. An additional search is also made to ensure the match is unique.

Pass:	ds:si	Address of null-terminated volume name to search for.
Returns:	CF ax	Set if error. DiskFindResult: DFR_UNIQUE if found and unique match. DFR_NOT_UNIQUE if found but not unique. DFR_NOT_FOUND if no match found.
	bx	If successful, disk handle of first disk found; otherwise, zero.
Destroyed:	Nothing.	
Library:	disk.def	

■ DiskForEach

Call a callback routine for each disk registered with the system, allowing the callback to cancel the operation.

Pass:	ax, cx, dx, bp di:si	Initial data to pass to the callback routine. Address of callback routine.
Returns:	ax, cx, dx, bp CF bx	As returned by last callback execution. Set if callback forced early termination. If CF set, the disk handle of the last disk processed; otherwise will be returned zero.

Destroyed: **di, si**

Callback Routine Specifications:

Passed:	bx ax, cx, dx, bp	Disk handle of current disk. Data set by caller and callback.
Return:	ax, cx, dx, bp CF	May be modified or preserved. Set if processing should be aborted.
May Destroy:	ax, cx, dx, bp	

Library: **disk.def**

■ DiskFormat

Format the disk in the specified drive.

Pass:	al ah	Drive number. GEOS media descriptor (MediaType).
-------	------------------------	--

Assembly Reference

	bx	Handle of the disk to be formatted, or Zero if disk is known to be unformatted, or -1 if state of drive is not known.
	bp	DiskFormatFlags.
	ds:si	New null-terminated ASCII volume name for the disk.
	cx:dx	Address of callback routine, initialized only if DFF_CALLBACK_PCT_DONE or DFF_CALLBACK_CYL_HEAD passed in bp .
Returns:	CF	Set on error.
	ax	Error code if error (FormatError), or FMT_DONE if successful.
	si:di	If successful, returns number of bytes in good clusters.
	dx:cx	If successful, returns number of bytes in bad clusters.
Destroyed:	ax, bx	
Callback Routine Specifications:		
	Passed: ax	Percentage done or number of cylinder heads finished, appropriate to bp parameter.
	Return: CF	Return set to cancel format.
	May Destroy: Nothing.	
Library:	disk.def	
Warning:	All data on the destination disk is lost.	

■ DiskGetDrive

Return the drive number of the drive in which the passed disk was registered.

Pass:	bx	Disk handle of registered disk.
Returns:	al	Zero-based drive number.
Destroyed:	ah	
Library:	disk.def	

■ DiskGetVolumeFreeSpace

Return the number of bytes free on a volume.

Pass:	bx	Disk handle of registered volume.
Returns:	CF	Set if error, clear if successful.
	ax	If error, error code: ERROR_INVALID_VOLUME.
	dx:ax	If successful, number of bytes free on volume.
Destroyed:	Nothing.	



DiskGetVolumeInfo

■ 44

Library: **disk.def**

■ DiskGetVolumeInfo

Return information about a registered disk.

Pass:	bx	Disk handle of registered disk.
	es:di	Address of DiskInfoStruct to fill in.
Returns:	CF	Set if error.
	ax	Zero if successful, otherwise ERROR_INVALID_VOLUME.
	es:di	If successful, the address of the returned DiskInfoStruct structure, filled in.
Destroyed:	ax	
Library:	disk.def	

■ DiskGetVolumeName

Return the volume name of the disk specified by the passed handle.

Pass:	bx	Disk handle of registered disk.
	es:di	Pointer to locked or fixed buffer at least VOLUME_NAME_LENGTH_ZT bytes long.
Returns:	es:di	Pointer to null-terminated volume name (with no trailing spaces).
Destroyed:	Nothing.	
Library:	disk.def	

■ DiskRegisterDisk

Register a disk with the system.

Pass:	al	Drive number containing disk to be registered.
Returns:	CF	Clear if successful, set if error.
	bx	Disk handle of registered disk if successful, zero if error.
Destroyed:	Nothing.	
Library:	disk.def	

■ DiskRegisterDiskSilently

Register a disk with the system without informing the user.

Pass:	al	Drive number containing disk to be registered.
Returns:	CF	Clear if successful, set if error.
	bx	Disk handle of registered disk if successful, zero if error.

Assembly Reference

Destroyed: Nothing.

Library: **disk.def**

■ DiskRestore

Restore a disk handle that had been saved to the state file with **DiskSave**.

Pass: **ds:si** Address of locked or fixed buffer originally passed to **DiskSave**.
cx:dx Address of callback routine to call if the user must be prompted for the disk. If **cx** is zero, no callback will be attempted and the routine will fail if the disk is unavailable (i.e. the drive no longer exists or the disk is not in the drive).

Returns: **CF** Set if disk handle could not be restored.
ax If **CF** clear, the disk handle of the registered disk.
If **CF** set, a **DiskRestoreError** indicating the failure.

Destroyed: Nothing.

Callback Routine Specifications:

Passed: **ds:dx** Address of null-terminated drive name, with “:” (colon) as the last character.
ds:di Address of null-terminated disk name.
ds:si Address of buffer originally passed to **DiskSave**.
ax **DiskRestoreError** to be returned if the callback routine was not being called.
bx, bp As passed to **DiskRestore**.
Return: **CF** Clear if disk should be in the drive, set if user cancelled the restoration.
ds:si If **CF** clear, address of buffer originally passed to **DiskSave**.
ax If **CF** set, error code returned (typically **DRE_USER_CANCELED_RESTORE**).

May Destroy: Nothing.

Library: **disk.def**

■ DiskSave

Save information that will allow a disk handle to be restored when the caller is restoring itself from a state file after a shutdown.

Pass: **bx** Disk handle to save.
es:di Address of locked or fixed buffer for opaque data.
cx Size of buffer in **es:di**.



DiskSetVolumeName

■ 46

Returns:	CF CX	Clear if successful, set if error. If CF clear, actual number of bytes used in the buffer. If CF set, the number of bytes needed to save the disk; zero if the disk can not be saved at all (e.g. it is a network drive that no longer exists).
Destroyed:	Nothing.	
Library:	disk.def	

■ DiskSetVolumeName

Set the name of a registered disk.

Pass:	bx ds:si	Disk handle of registered volume. Address of null-terminated ASCII name.
Returns:	CF ax	Set if error. Error code, or zero if no error ERROR_INVALID_VOLUME ERROR_ACCESS_DENIED.
Destroyed:	Nothing.	
Library:	disk.def	

■ DosExec

Begin execution of a DOS application, shutting down GEOS to state files or creating a new task in the task-switcher for the DOS application.

Pass:	bx ds:si es:di ax dx:bp CX	Optional disk handle for the disk on which the DOS program resides. (Pass zero for the disk containing GEOS.) Address of null-terminated path of the DOS application. If this is just a null character, the system's command interpreter will be run with the given command-line arguments. If DEF_MEM_REQ is passed, this will contain the DosExecArgAndMemReqsStruct containing the memory requirements of the program; Otherwise, this is the address of a buffer containing the command-line arguments to pass to the application. Optional disk handle for the disk that contains the path or directory in which the application should be executed. Address of a buffer containing the null-terminated path or directory name in which the application should be executed. A record of DosExecFlags .
Returns:	CF	Set if the program could not be run.

Assembly Reference

ax Error code if CF is set:
 ERROR_FILE_NOT_FOUND
 ERROR_DOS_EXEC_IN_PROGRESS
 ERROR_INSUFFICIENT_MEMORY
 ERROR_ARGS_TOO_LONG

Destroyed: **ax, bx, cx, dx, di, si, bp, ds, es**

Library: **system.def**

■ DriveGetDefaultMedia

Return the GEOS media descriptor of the highest density format supported by the specified drive.

Pass: **al** Zero-based drive number.

Returns: **CF** Set if drive does not exist.
ah GEOS media descriptor (**MediaType**).

Destroyed: Nothing.

Library: **drive.def**

■ DriveGetExtStatus

Return the extended status word for the specified drive.

Pass: **al** Zero-based drive number.

Returns: **CF** Set if drive does not exist.
ax **DriveExtendedStatus** record if successful.

Destroyed: Nothing.

Library: **drive.def**

■ DriveGetName

Return the name of the specified drive.

Pass: **al** Zero-based drive number.
es:di Address of locked or fixed buffer into which the null-terminated name will be written.
cx Number of bytes in the buffer.

Returns: **CF** clear:
cx Number of bytes written to the buffer including the terminating null.
es:di Address of the terminating null, *not* the first character.



DriveGetStatus

■ 48

CF set:

CX

Zero if the drive does not exist.

Total number of bytes needed if the buffer is too small.

Destroyed: Nothing.

Library: **drive.def**

■ DriveGetStatus

Returns status information on the specified drive.

Pass: **al** Zero-based drive number.

Returns: **CF** Set if drive does not exist.
ah **DriveStatus** record if successful.

Destroyed: Nothing.

Library: **drive.def**

■ DriveTestMediaSupport

Test if the specified drive supports the given media type.

Pass: **al** Zero-based drive number.

ah **MediaType** media descriptor.

Returns: **CF** Clear if media type supported by the drive; set otherwise.

Destroyed: Nothing.

Library: **drive.def**

Assembly Reference

■ ECCheckBounds

Verifies that a pointer is in bounds (**ds** is a valid segment and **si** is a valid offset).

Pass: **ds:si** Pointer to be checked.
Returns: Nothing. Calls **FatalError** if pointer is out of bounds.
Destroyed: Nothing.
Library: **ec.def**

■ ECCheckClass

Checks that the pointer actually points at a class definition.

Pass: **es:di** Class pointer.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECCheckDriverHandle

Checks that the passed handle actually references a valid driver.

Pass: **bx** Driver handle.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECCheckEventHandle

Checks that the passed handle references a valid **EventHandle**.

Pass: **bx** Event handle.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECCheckGeodeHandle

Checks that the passed handle actually references a valid geode.



ECCheckGStateHandle

■ 50

Pass: **bx** Geode handle.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECKheckGStateHandle

Makes sure the passed handle actually references a valid GState.

Pass: **di** GState handle.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECKheckLibraryHandle

Makes sure the passed handle actually references a valid library.

Pass: **bx** Library handle.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECKheckLMemHandle

Ensures that the passed handle references a valid, sharable local memory block.

Pass: **bx** Handle of local memory block.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing.
Library: **ec.def**

■ ECKheckLMemHandleNS

Ensures that the passed handle references a valid local memory block, ignoring issues of sharing.

Pass: **bx** Handle of local memory block.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing.

Assembly Reference

Library: **ec.def**

■ ECCheckLMemObject

Makes sure that the given pointer points to an object within an object block. Will not allow the pointer to point to a Process object. (If it does, this routine will call **FatalError**.)

Pass: ***ds:si** Segment:chunk handle to check.

Returns: Nothing. Calls **FatalError** if assertions fail.

Destroyed: Nothing. Flags are preserved.

Library: **ec.def**

■ ECCheckLMemOD

Makes sure that the given optr points to an object within an object block. Will not allow the pointer to point to a Process object.

Pass: **bx:si** The optr of the object to check.

Returns: Nothing. Calls **FatalError** if assertions fail.

Destroyed: Nothing. Flags are preserved.

Library: **ec.def**

■ ECCheckLMemODCXDX

Checks that the passed **cx:dx** is a valid optr to an LMem-based object (not a Process).

Pass: **cx:dx** The optr to check.

Returns: Nothing. Calls **FatalError** if assertions fail.

Destroyed: Nothing.

Library: **ui.def**

■ ECCheckMemHandle

Checks the validity of a passed global memory handle.

Pass: **bx** Memory handle to check.

Returns: Nothing. Calls **FatalError** if assertions fail.

Destroyed: Nothing. Flags are preserved.

Library: **ec.def**



■ ECheckMemHandleNS

Checks the validity of the passed memory handle, ignoring sharing violation errors (when a block should be sharable but is not).

Pass: **bx** Memory handle to check.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved. Interrupts left in the same state as when the routine was called.
Library: **ec.def**

■ ECheckObject

Ensures that the locked object is valid. This routine can check both local memory objects and Process objects.

Pass: **ds** If the “object” is a Process object, **ds** points to dgroup.
***ds:si** Segment:chunk handle of object to validate.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECheckOD

Ensures that the optr passed references an object. This routine considers Process objects valid, unlike **ECheckLMemOD**.

Pass: **bx:si** The optr of the object to check.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing. Flags are preserved.
Library: **ec.def**

■ ECheckODCXDX

Checks to see if the passed **cx:dx** is a valid optr.

Pass: **cx:dx** The optr to check.
Returns: Nothing. Calls **FatalError** if assertions fail.
Destroyed: Nothing.
Library: **ui.def**

■ ECCheckProcessHandle

Checks that the passed handle actually references a valid Process.

Pass: **bx** Process handle.
 Returns: Nothing. Calls **FatalError** if assertions fail.
 Destroyed: Nothing. Flags are preserved.
 Library: **ec.def**

■ ECCheckQueueHandle

Checks that the passed handle actually references a valid event queue.

Pass: **bx** Queue handle.
 Returns: Nothing. Calls **FatalError** if assertions fail.
 Destroyed: Nothing. Flags are preserved.
 Library: **ec.def**

■ ECCheckResourceHandle

Checks that the passed handle actually references a valid resource (device).

Pass: **bx** Resource handle.
 Returns: Nothing. Calls **FatalError** if assertions fail.
 Destroyed: Nothing. Flags are preserved.
 Library: **ec.def**

■ ECCheckSegment

Checks that the passed segment value actually points to a locked block.

Pass: **ax** Segment address to check.
 Returns: Nothing. Calls **FatalError** if assertions fail.
 Destroyed: Nothing. Flags are preserved.
 Library: **ec.def**

■ ECCheckThreadHandle

Checks that the passed handle actually references a valid thread.

Pass: **bx** Thread handle.
 Returns: Nothing. Calls **FatalError** if assertions fail.



ECCheckUILMemOD

■ 54

Destroyed: Nothing. Flags are preserved.

Library: **ec.def**

■ ECTestUILMemOD

Checks that the passed optr references a valid UI-run object (not a Process).

Pass: **bx:si** The optr to be checked.

Returns: Nothing. Calls **FatalError** if assertions fail.

Destroyed: Nothing.

Library: **ui.def**

■ ECTestUILMemODCXDX

Checks that the passed **cx:dx** is a valid optr pointing to a UI-run object in an object block (not a Process).

Pass: **cx:dx** The optr to check.

Returns: Nothing. Calls **FatalError** if assertions fail.

Destroyed: Nothing.

Library: **ui.def**

■ ECTestWindowHandle

Checks that the passed handle actually references a valid window.

Pass: **bx** Window handle.

Returns: Nothing. Calls **FatalError** if assertions fail.

Destroyed: Nothing. Flags are preserved.

Library: **ec.def**

■ ElementArrayAddElement

Add an element to a given element array. If the element already exists, its reference count will be incremented.

Pass: ***ds:si** Segment:chunk handle of the element array's chunk.

cx:dx Address of element to add.

ax Size of the element, if variable-sized element array.

bx:di Address of a callback routine; pass zero in both registers to invoke a straight binary-value comparison.

bp Value to pass to the callback routine.

Assembly Reference

Returns: **CF** Set if the element was newly added, clear if the reference count was incremented for an existing element.
ax Element number of the newly added element.

Destroyed: Nothing.

Callback Routine Specifications:

Passed:	es:di	Address of element to be added.
	ds:si	Address of comparison element.
	cx	Size of the elements; sizes will be identical, or the callback will not be called.
	ax	Value passed initially in bp to ElementArrayAddElement .
Return:	CF	Set if the elements are equal.
May Destroy:	ax, bx, cx, dx	

Library: **chunkarr.def**

Warning: This routine may resize or move chunks or blocks; therefore, you must dereference all stored pointers after a call to this routine.

■ ElementArrayAddReference

Increments the reference count for an element in an element array.

Pass: ***ds:si** Address of the locked element array.
ax Element number of the subject element.

Returns: Nothing.

Destroyed: Nothing.

Library: **chunkarr.def**

■ ElementArrayCreate

Creates a new element array in the specified chunk. The new array will have no elements.

Pass: **ds** Global handle of the block that will contain the array.
bx Size of each element; pass zero for variable-sized elements.
cx Size of the header; pass zero to get the default size.
si Chunk handle of the chunk in which the array will be created.
al A record of **ObjChunkFlags** to pass to **LMemAlloc**.

Returns: ***ds:si** Address of the new, locked element array.

Destroyed: Nothing.

Library: **chunkarr.def**



ElementArrayDelete

■ 56

Warning: This routine may resize or move chunks or blocks on the heap; therefore, all stored pointers must be dereferenced.

■ ElementArrayDelete

Deletes an element regardless of its reference count.

Pass: ***ds:si** Segment:chunk handle of the element array.
 ax Element number of element to be deleted.

Returns: Nothing.

Destroyed: Nothing.

Library: **chunkarr.def**

■ ElementArrayElementChanged

Checks to see if a recently changed element is now equal to another element. If the element is now a duplicate, it will be combined with the other element, and the other element's reference count will be incremented.

Pass: ***ds:si** Segment:chunk handle of the element array.
 ax Element number of the changed element.
 bx:di Address of a callback comparison routine. Pass zero in both registers to invoke straight binary comparison.
 bp Value to pass to the callback routine (in **ax**).

Returns: **ax** The new element number of the changed element.

Destroyed: Nothing.

Callback Routine Specifications:

Passed:	es:di	Address of the changed element.
	ds:si	Address of a comparison element.
	cx	Size of elements; both will be identical size, or the callback will not be called.
	ax	Value for callback routine passed in bp to ElementArrayElementChanged .
Return:	CF	Set if elements are equal, clear otherwise.
May Destroy:	ax, bx, cx, dx	

Library: **chunkarr.def**

■ ElementArrayGetUsedCount

Returns the number of elements in the array that actually hold data.

Pass: ***ds:si** Segment:chunk handle of the element array.

Assembly Reference

bx:di Address of a callback routine to make qualification more explicit. Pass zero in **bx** to use no callback routine.

cx, dx Data passed through to callback routine.

Returns: **ax** Number of “used” elements as determined by the callback.

Destroyed: Nothing.

Callback Routine Specifications:

Passed: ***ds:si** Segment:chunk of the element array.
ds:di Address of element being processed.
cx, dx Data passed through, as modified by the last calling of the callback routine.

Return: **CF** Set if the element qualifies as “used.” Clear otherwise.

May Destroy: **ax, bx, cx, dx, si, di**

Library: **chunkarr.def**

■ ElementArrayRemoveReference

Removes a reference to the specified element, removing the element itself if the reference count drops to zero.

Pass: ***ds:si** Segment:chunk handle of the element array.
ax Element number of subject element.
bx:di Address of routine to call if the reference count drops to zero.
cx Value to pass to callback routine (in **ax**).

Returns: **CF** Set if element removed, clear otherwise.

Destroyed: Nothing.

Callback Routine Specifications:

Passed: **ax** Value passed in **cx** to **ElementArrayRemoveReference**.
ds:di Address of the element to be removed.

Return: Nothing.

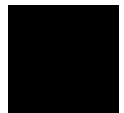
May Destroy: **ax, bx, cx, dx**

Library: **chunkarr.def**

■ ElementArrayTokenToUsedIndex

Returns the index of an element with respect to used elements in the array, given its token.

Pass: ***ds:si** Segment:chunk handle of the element array.
ax Token.



ElementArrayUsedIndexToToken

■ 58

	bx:di	Address of a callback routine to make qualification more explicit. Pass zero in bx to use no callback routine.
	cx, dx	Data passed through to callback routine.
Returns:	ax	Index of the element with respect to <i>used</i> elements in the array.
Destroyed:	Nothing.	
Callback Routine Specifications:		
Passed:	*ds:si	Segment:chunk of the element array.
	ds:di	Address of element being processed.
	cx, dx	Data passed through, as modified by the last calling of the callback routine.
Return:	CF	Set if the element qualifies as “used.”
May Destroy:	ax, bx, cx, dx, si, di	
Library:	chunkarr.def	

■ ElementArrayUsedIndexToToken

Returns the token of an element given its index with respect to used elements in the array.

Pass:	*ds:si	Segment:chunk handle of the element array.
	ax	Index into used list of element array.
	bx:di	Address of a callback routine to make qualification more explicit. Pass zero in bx to use no callback routine.
	cx, dx	Data passed through to callback routine.
Returns:	CF	Set if a valid token was found. Clear if not found.
	ax	Token of the element, if found. If not found, ax will be returned <code>CA_NULL_ELEMENT</code> .
Destroyed:	Nothing.	
Callback Routine Specifications:		
Passed:	*ds:si	Segment:chunk of the element array.
	ds:di	Address of element being processed.
	cx, dx	Data passed through, as modified by the last calling of the callback routine.
Return:	CF	Set if the element passed qualifies as “used.”
May Destroy:	ax, bx, cx, dx, si, di	
Library:	chunkarr.def	

Assembly Reference

■ FatalError

Indicates that a fatal error has been encountered within an application. Note that it is impossible to return from a fatal error. This routine is meant to identify what precipitated the fatal error.

Pass: **ax** Error code (this code is application-specific and must be custom-defined).

Returns: Not applicable

Destroyed: Not applicable

Library: **ec.def**

■ FileAddStandardPathDirectory

Adds the specified directory to the standard path table.

Pass: **ds:dx** Pointer to a null-terminated path string.
ax **StandardPath** to add
bx **FileAddStandardPathFlags**

Returns: **CF** Set if an error was encountered
ax **FileError** (if CF is set) ERROR_PATH_NOT_FOUND

Destroyed: Nothing.

Library: **file.def**

■ FileClose

Close an open file.

Pass: **al** **FileAccessFlags** record. Only FILE_NO_ERRORS is used by this routine; other flags in the record *must* be cleared.
bx File handle of the file to be closed.

Returns: **CF** Set if an error occurred.
ax Error code (**FileError**) if error occurred.

Destroyed: Nothing.

Library: **file.def**

■ FileCommit

Commits a file to the disk by forcing all changes to be written out.

Pass: **al** **FileAccessFlags** record. Only FILE_NO_ERRORS is used by this routine; other flags in the record *must* be cleared.
bx File handle of the file to be closed.



FileComparePaths

■ 60

Returns: **CF** Set if an error occurred.
ax Error code (**FileError**) if error occurred.

Destroyed: Nothing.

Library: **file.def**

■ FileComparePaths

Compares two paths, returning their relationship.

Pass: **cx** Disk handle of disk containing the first path.
ds:si Address of the first null-terminated path name.
Pass **ds** = 0 for a null path.
dx Disk handle of disk containing the second path.
es:di Address of the second null-terminated path name.
Pass **es** = 0 for a null path.

Returns: **al** Value of **PathCompareType**:
PCT_EQUAL
PCT_SUBDIR
PCT_UNRELATED
PCT_ERROR

Destroyed: Nothing.

Library: **file.def**

Warning: Neither path may contain a trailing backslash. Also, this routine does not deal with links; call **FileConstructActualPath** on each path if you suspect either involves links.

■ FileConstructActualPath

Similar to **FileConstructFullPath**, this routine also replaces links with their actual targets. It creates a full path string from the passed information.

Pass: **dx** Non-zero value to prepend drive name and colon to the path.
bx Disk handle or path identifier:
bx = 0
Prepend the current path and use the current disk handle.
bx = **StandardPath** value
Prepend the logical path for the standard path, returning the top-level disk handle.
bx = disk handle
Passed path is an absolute path; the disk handle will be used only if **dx** is non-zero.

Assembly Reference

	ds:si	Address of the <i>tail</i> of the path string being constructed. If bx is non-zero and not a StandardPath value, this path must be absolute.
	es:di	Address of buffer into which the constructed path will be written.
	cx	Size of the buffer pointed to by es:di .
Returns:	CF	Set if error, clear otherwise.
	ax	A FileError error code if CF set.
	es:di	Address of the terminating null of the constructed path.
	bx	Disk handle of the disk for the path.
Destroyed:	Nothing.	
Library:	file.def	

■ FileConstructFullPath

Constructs a full path given a standard path constant and a path relative to the standard path. This routine does not resolve links; instead, use **FileConstructActualPath**.

Pass:	dx	Non-zero value to prepend drive name and colon to the path.
	bx	Disk handle or path identifier: bx = 0 Prepend the current path and use the current disk handle. bx = StandardPath value Prepend the logical path for the standard path, returning the top-level disk handle. bx = disk handle Passed path is an absolute path; the disk handle will be used only if dx is non-zero.
	ds:si	Address of the <i>tail</i> of the path string being constructed. If bx is non-zero and not a StandardPath value, this path must be absolute.
	es:di	Address of buffer into which the constructed path will be written.
	cx	Size of the buffer pointed to by es:di .
Returns:	CF	Set if error, clear otherwise.
	ax	A FileError error code if CF set.
	es:di	Address of the terminating null of the constructed path.
	bx	Disk handle of the disk for the path.
Destroyed:	Nothing.	



FileCopy

■ 62

Library: **file.def**

■ FileCopy

Copies a source file into a destination file. If the destination file does not already exist, it will be created. Any existing destination file with the same name will be truncated and overwritten.

Pass:	ds:si	Address of the null-terminated source file name. Or, pass zero in ds and the file handle of an open source file in si .
	cx	Source file's disk handle. If zero, the disk handle of the thread's current path will be used. If a disk handle is provided, the path <i>must</i> be absolute.
	es:di	Address of null-terminated destination file name.
	dx	Destination file's disk handle. If zero, the disk handle of the thread's current path will be used. If a disk handle is provided, the path <i>must</i> be absolute.
Returns:	CF	Set if error, clear otherwise.
	ax	Zero if successful, error code if error: ERROR_FILE_NOT_FOUND ERROR_PATH_NOT_FOUND ERROR_TOO_MANY_OPEN_FILES ERROR_ACCESS_DENIED ERROR_SHORT_READ_WRITE

Destroyed: Nothing.

Library: **file.def**

■ FileCopyExtAttributes

Copies all the extended file attributes from an open file into another named file.

Pass:	bx	File handle of open source file.
	ds:dx	Address of null-terminated name of destination file.
Returns:	CF	Set on error, clear otherwise.
	ax	Error code if error; destroyed otherwise.

Destroyed: **ax**, if not returned.

Library: **file.def**

■ FileCopyLocal

Copies the source file to the destination file. If the destination file does not exist, this routine will create it; if the destination file already exists, it will be truncated to accommodate the new source.

Assembly Reference

This routine copies a file to a local standard path directory even if a file of the same name exists in the remote directory.

Pass:	ds:si	Address of the null-terminated source file name. Or, pass zero in ds and the file handle of an open source file in si .
	cx	Source file's disk handle. If zero, the disk handle of the thread's current path will be used. If a disk handle is provided, the path <i>must</i> be absolute.
	es:di	Address of null-terminated destination file name.
	dx	Destination file's disk handle. If zero, the disk handle of the thread's current path will be used. If a disk handle is provided, the path <i>must</i> be absolute.
Returns:	CF	Set if error, clear otherwise.
	ax	Zero if successful, error code if error: ERROR_FILE_NOT_FOUND ERROR_PATH_NOT_FOUND ERROR_TOO_MANY_OPEN_FILES ERROR_ACCESS_DENIED ERROR_SHORT_READ_WRITE
Destroyed:	Nothing.	
Library:	file.def	

■ FileCopyPathExtAttributes

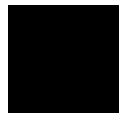
Copies the extended attributes of a file to another file without opening either file.

Pass:	cx	Source file's disk handle.
	ds:si	Address of the null-terminated source file name.
	dx	Destination file's disk handle.
	es:di	Address of null-terminated destination file name.
Returns:	CF	Set if error, clear otherwise.
	ax	FileError code if CF is set.
Destroyed:	Nothing.	
Library:	file.def	

■ FileCreate

Creates a new file or truncates an existing file.

Pass:	ah	Record of FileCreateFlags : FCF_NATIVE Set to force the file to use the native format of the system on which it's created. There is little reason for a GEOS
-------	-----------	---



FileCreateDir

■ 64

		application to set this flag. If this flag is used, the file already exists in a different format, and FCF_MODE (below) is not set to FILE_CREATE_ONLY, then an error will occur (ERROR_FILE_FORMAT_MISMATCH).
		FCF_MODE
		This bitfield may contain one of three values:
		FILE_CREATE_TRUNCATE to truncate an existing file.
		FILE_CREATE_NO_TRUNCATE to create a new file or to abort.
		FILE_CREATE_ONLY to create a new file or fail if a file of the same name already exists.
	al	Record of FileAccessFlags requesting at least write access.
	cx	Record of FileAttrs for the new file if it is created anew.
	ds:dx	Address of the null-terminated file name.
Returns:	CF	Set if error, clear otherwise.
	ax	File handle if successful. If CF set, ax will contain an error code.
Destroyed:	Nothing.	
Library:	file.def	

■ FileCreateDir

Creates a new directory.

Pass:	ds:dx	Address of null-terminated path name to create.
Returns:	CF	Set if error, clear otherwise.
	ax	If CF set, error code: ERROR_PATH_NOT_FOUND ERROR_ACCESS_DENIED
Destroyed:	Nothing.	
Library:	file.def	

■ FileCreateTempFile

Creates a temporary file with a unique name.

Pass:	ah	FileCreateFlags ; only FCF_NATIVE is used here.
	al	FileAccessFlags for the temporary file.
	cx	FileAttrs for the temporary file.
	ds:dx	Address of the null-terminated directory in which to create the file. Add 14 (fourteen) extra null bytes at the end of the path name to be replaced by the file name.
Returns:	CF	Set if error, clear otherwise.

Assembly Reference

	ax	File handle of the temporary file, if successful. Error code (FileError) if CF set.
	ds:dx	Address of the file's path, if successful.
Destroyed:	Nothing.	
Library:	file.def	

■ FileDelete

Deletes the specified file.

Pass:	ds:dx	Address of null-terminated file name.
Returns:	CF ax	Set if error, clear otherwise. Error code if CF set: ERROR_FILE_NOT_FOUND ERROR_ACCESS_DENIED ERROR_FILE_IN_USE
Destroyed:	Nothing.	
Library:	file.def	

■ FileDeleteDir

Deletes a directory and all the files and subdirectories within it.

Pass:	ds:dx	Address of null-terminated path to be deleted.
Returns:	CF ax	Set if error, clear otherwise. FileError code if CF set: ERROR_PATH_NOT_FOUND ERROR_IS_CURRENT_DIRECTORY ERROR_ACCESS_DENIED ERROR_DIRECTORY_NOT_EMPTY
Destroyed:	Nothing.	
Library:	file.def	

■ FileDeleteStandardPathDirectory

Deletes the specified directory from the standard path table.

Pass:	ds:dx ax	Address of null-terminated path to be deleted. The StandardPath that this particular path was added as.
Returns:	CF ax	Set if error, clear otherwise. FileError code if CF set: ERROR_PATH_NOT_FOUND



FileDuplicateHandle

■ 66

Destroyed: Nothing.
Library: **file.def**

■ FileDuplicateHandle

Duplicates the passed handle, returning a new handle referring to the same file.

Pass: **bx** File handle to duplicate.

Returns: **CF** Set if error, clear otherwise.
ax New handle, if successful. If **CF** set, an error code: **ERROR_TOO_MANY_OPEN_FILES**

Destroyed: Nothing.
Library: **file.def**

■ FileEnum

Enumerates all files in a directory, calling a callback routine for each. This routine receives its parameters on the stack; the parameter structure is not returned on the stack but is instead popped during the routine's execution.

Pass: **ss:sp** On the stack, a **FileEnumParams** structure.

FEP_searchFlags A **FileEnumSearchFlags** record indicating the types of files and directories to match.

FEP_returnAttrs A far pointer to an array of attributes to be returned.

FEP_returnSize Size of each entry in the *FEP_returnAttrs* buffer.

FEP_matchAttrs A far pointer to an array of **FileExtAttrDesc** attributes to be matched by **FileEnum**.

FEP_bufSize Number of structures the return buffer may hold. Defaults to **FE_BUFSIZE_UNLIMITED**.

FEP_skipCount Number of matches to skip before adding entries to the return buffer. Allows consecutive enumerations of all files in a directory.

FEP_callback Address of a callback routine to be called for each file that passes the other filters.

FEP_callbackAttrs A far pointer to a list of supplemental

		attributes (FileExtAttrDesc structures) that don't or can't appear in <i>FEP_matchAttrs</i> .
	<i>FEP_cbData1</i>	
	<i>FEP_cbData2</i>	Two separate dwords of data that is passed along to the callback routine.
	<i>FEP_headerSize</i>	Amount of space to reserve at the start of the returned buffer if FESF_LEAVE_HEADER set.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code (FileError) if CF set on return.
	bx	Memory handle of buffer created, if any. If no files were found, or if an error occurred, no buffer is returned and bx is returned a null handle (zero).
	cx	Number of matching files returned in the buffer in bx .
	dx	Number of matching files that would not fit in the passed buffer in bx . The maximum buffer size is passed in the <i>FEP_bufSize</i> parameter in the FileEnumParams structure. If <i>FEP_bufSize</i> is zero, dx will contain the number of matching files in the directory upon return.
	buffer	The buffer returned in bx will contain structures of the type requested in <i>FEP_returnAttrs</i> for all the matching files.
	di	The updated real skip count if FESF_REAL_SKIP passed; Register preserved if FESF_REAL_SKIP not passed.
Destroyed:	Nothing.	
Callback Routine Specifications:		
	Passed:	ds Segment of FileEnumCallbackData structure, an array of FileExtAttrDesc structures indicating the attributes to be retrieved for each matching file.
		bp Inherited stack frame, which must be passed to any FileEnum helper routines called by the callback routine.
Library:	fileEnum.def	

■ FileEnumLocateAttr

Locates an extended attribute within an array of **FileExtAttrDesc** structures. This routine usually acts as a “helper” of **FileEnum** callback routines.

Pass:	ax	FileExtendedAttribute value indicating the attribute to be found (FEA_MULTIPLE not allowed here).
	ds:si	Address of array to search.

FileEnumPtr

■ 68

	es:di	Address of attribute's name, if FEA_CUSTOM passed in ax.
Returns:	CF	Set if the attribute was not found, clear if it was.
	es:di	Address of the attribute, if CF clear on return. If the file does not have the given attribute, es:di.FEAD_value.segment will be zero.
Destroyed:	es, di	if attribute searched for was not found.
Library:	fileEnum.def	

■ FileEnumPtr

This routine performs an enumeration identical to **FileEnum** except that it accepts a pointer to a **FileEnumParams** structure rather than needing all of the parameters passed on the stack. It is, therefore, somewhat simpler to call.

Pass:	ds:si	Address of FileEnumParams structure. See FileEnum for a description of these elements.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code (FileError) if CF set on return.
	bx	Memory handle of buffer created, if any. If no files were found, or if an error occurred, no buffer is returned and bx is returned a null handle (zero).
	buffer	The buffer returned in bx will contain structures of the type requested in <i>FEP_returnAttrs</i> for all the matching files.
	cx	Number of matching files returned in the buffer in bx .
	dx	Number of matching files that would not fit in the passed buffer in bx . The maximum buffer size is passed in the <i>FEP_bufSize</i> parameter in the FileEnumParams structure. If <i>FEP_bufSize</i> is zero, dx will contain the number of matching files in the directory upon return.
	di	The updated real skip count if FESF_REAL_SKIP passed; Register preserved if FESF_REAL_SKIP not passed.
Destroyed:	Nothing.	
Callback Routine Specifications:		
	Passed:	ds Segment of FileEnumCallbackData structure, an array of FileExtAttrDesc structures indicating the attributes to be retrieved for each matching file.
		bp Inherited stack frame, which must be passed to any FileEnum helper routines called by the callback routine.
Library:	fileEnum.def	

Assembly Reference

■ FileEnumWildcard

Checks if the virtual name of the current file matches the pattern passed to **FileEnum** in *FEP_cbData1*. In this case, *FEP_cbData1* is cast to a far pointer to the name string. This routine acts as a **FileEnum** “helper” routine.

Pass:	ds	Segment of FileEnumCallbackData structure.
	ss:bp	Inherited stack frame including: <i>FEP_cbData1</i> Address of name pattern to match. <i>FEP_cbData2.low</i> Non-zero if the matching should be case-insensitive.
Returns:	CF	Clear if the FEA_NAME attribute of the file matches the name pointed to by <i>FEP_cbData1</i> . Set if they don't match.
Destroyed:	Nothing.	
Library:	fileEnum.def	

■ FileGetAttributes

Retrieves a file's attributes.

Pass:	ds:dx	Address of the null-terminated file name.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code if CF set: ERROR_FILE_NOT_FOUND ERROR_PATH_NOT_FOUND
	cx	FileAttrs record of the file if successful.
Destroyed:	Nothing.	
Library:	file.def	

■ FileGetCurrentPath

Returns the thread's current directory. If the directory is a standard path, the returned disk handle (**bx**) will actually be a **StandardPath** constant and the buffer will contain a relative path. To retrieve a full path, use **FileConstructFullPath** instead, passing **dx** a non-zero value and **bp** the value of -1.

Pass:	ds:si	Address of a locked or fixed buffer into which the path will be written.
	cx	Size of the buffer, or zero if only the disk handle should be returned.

FileGetCurrentPathIDs

■ 70

Returns:	bx	Disk handle of the disk on which the current path resides. If the current path is a standard path, this will be the StandardPath constant rather than the disk handle.
	ds:si	Buffer stores the address of the null-terminated path string. If a standard path is returned in bx , this will be relative to that standard path. Otherwise, it is an absolute path.
Destroyed:	Nothing.	
Library:	file.def	

■ FileGetCurrentPathIDs

Returns an array of **FilePathID** structures for the current path. These IDs may be used in handling file change notification messages.

Pass:	ds	Segment of LMem block in which to allocate the array.
Pass:	CF	Set if error; clear otherwise.
	ax	FileError if CF set.
	ax	Chunk handle of array (if CF clear).
	ds	Fixed up.
Destroyed:	Nothing.	
Library:	file.def	

■ FileGetDateAndTime

Returns the modification date and time of an open file.

Pass:	bx	File handle of open file.
Returns:	cx	FileTime record indicating last modification time.
	dx	FileDate record indicating last modification date.
Destroyed:	ax	
Library:	file.def	

■ FileGetDiskHandle

Retrieves the disk handle of an open file.

Pass:	bx	File handle of an open file.
Returns:	bx	Disk handle of the file's disk (or zero if file is open on a device).
Destroyed:	Nothing.	
Library:	file.def	

Assembly Reference

■ FileGetHandleExtAttributes

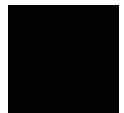
Retrieves one or more extended attributes of the specified open file. This routine is similar to **FileGetPathExtAttributes** except that the file must be open.

Pass:	bx	File handle of the open file.
	ax	FileExtendedAttribute specifying the attribute to get.
	es:di	Address of a locked or fixed buffer into which the attribute will be fetched. Alternatively, an array of FileExtAttrDesc structures, if ax is <code>FEA_MULTIPLE</code> . To retrieve custom attributes, you must use <code>FEA_MULTIPLE</code> and an appropriate FileExtAttrDesc structure.
	cx	Size of the buffer pointed to by es:di , or the number of entries if es:di points to an array of FileExtAttrDesc structures.
Returns:	CF	Set if one or more attributes could not be retrieved, either because the file system does not support them or because the file did not have them. Even if CF is set on return, those attributes that could be retrieved are retrieved.
	ax	Error code if CF is set: ERROR_ATTR_NOT_SUPPORTED ERROR_ATTR_SIZE_MISMATCH ERROR_ATTR_NOT_FOUND ERROR_ACCESS_DENIED
Destroyed:	Nothing. (ax destroyed if CF clear).	
Library:	file.def	

■ FileGetPathExtAttributes

Retrieves one or more extended attributes from the file whose path is specified. This routine is similar to **FileGetHandleExtAttributes** but specifies the file by its path rather than its handle.

Pass:	ds:dx	Address of the null-terminated name of the file or directory to have its attributes returned.
	ax	FileExtendedAttribute value indicating the attribute to retrieve.
	es:di	Address of a locked or fixed buffer into which the attributes will be returned. If ax is passed as <code>FEA_MULTIPLE</code> , this will point to an array of FileExtAttrDesc structures. To retrieve custom attributes, you must use <code>FEA_MULTIPLE</code> and an appropriate FileExtAttrDesc structure.



FileLockRecord

■ 72

	cx	Size of the buffer in es:di , or the number of FileExtAttrDesc structures in the array there.
Returns:	CF	Set if one or more attributes could not be retrieved, either because the file system does not support them or because the file did not have them. Even if CF is set on return, those attributes that could be retrieved are retrieved.
	ax	Error code if CF is set: ERROR_FILE_NOT_FOUND ERROR_ATTR_NOT_SUPPORTED ERROR_ATTR_SIZE_MISMATCH ERROR_ATTR_NOT_FOUND
Destroyed:	Nothing.	
Library:	file.def	

■ FileLockRecord

Locks a region of a file. The region may later be unlocked with **FileUnlockRecord**. This routine does not keep other threads from using or writing to the file; it only keeps others from locking the same region.

Pass:	bx	File handle of open file.
	cx:dx	32-bit start offset of region.
	si:di	32-bit ending offset of region.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code if CF set: ERROR_ALREADY_LOCKED.
Destroyed:	Nothing. (ax destroyed if CF clear).	
Library:	file.def	

■ FileMove

Moves a file or subdirectory from one place in the file system to another. Some file systems will allow directories to be moved across volumes, but other file systems will return an error in this case. Files, however, will always be moved (assuming the file name does not already exist and the destination directory is writable).

Pass:	ds:si	Address of the null-terminated source file name.
	cx	Disk handle of the source disk. If a null handle is passed, the thread's current path is used and ds:si is assumed to point to a path extension relative to the current directory. If a disk handle is passed, the path in ds:si <i>must</i> be absolute.
	es:di	Address of the null-terminated destination file name.

Assembly Reference

	dx	Disk handle of the destination disk. If a null handle is passed, the thread's current path is used and es:di is assumed to point to a path extension relative to the current directory. If a disk handle is passed, the path in es:di <i>must</i> be absolute.
Returns:	CF ax	Set if error, clear otherwise. Error code if CF set on return: ERROR_FILE_NOT_FOUND ERROR_PATH_NOT_FOUND ERROR_TOO_MANY_OPEN_FILES ERROR_ACCESS_DENIED ERROR_SHORT_READ_WRITE (not enough space on dest.) ERROR_DIFFERENT_DEVICE ERROR_INSUFFICIENT_MEMORY
Destroyed:	Nothing.	
Library:	file.def	
Warning:	This routine does not do an optimal move if links are involved in either the source or destination. To fix this, call FileConstructActualPath on either or both paths before calling FileMove .	

■ FileOpen

Opens an existing file.

Pass:	al ds:dx	FileAccessFlags record indicating the opening mode. Address of the null-terminated file name.
Returns:	CF ax	Set if error, clear if the file was opened successfully. If CF is clear, the file handle of the opened file. The <i>HF_otherInfo</i> field of the handle contains either the youngest handle to the same file or zero if no other handles are open to the file. If CF set, error code: ERROR_FILE_NOT_FOUND ERROR_PATH_NOT_FOUND ERROR_TOO_MANY_OPEN_FILES ERROR_SHARING_VIOLATION ERROR_WRITE_PROTECTED
Destroyed:	Nothing.	
Library:	file.def	

■ FileOpenAndRead

Opens a file and reads its contents into a memory block.



FileParseStandardPath

■ 74

Pass:	ax ds:dx	FileOpenAndReadFlags. Address of null-terminated file name.
Returns:	CF ax bx cx	Set if error, clear otherwise. If CF is clear, memory handle of filled buffer. If CF is set, a FileError is returned in ax . (If CF is clear) file handle. (If CF is clear) buffer size of buffer pointed at by ax .
Destroyed:	Nothing.	
Library:	file.def	

■ FileParseStandardPath

Constructs the best combination of a **StandardPath** constant and a relative path. If the file system on which GEOS resides is case-insensitive, then the passed path must be in all upper case for it to be properly recognized.

Pass:	es:di bx	Address of null-terminated path string to parse. Disk handle of disk on which the path resides. Passing a null handle in bx indicates the drive name is specified in the path.
Returns:	ax es:di	StandardPath constant. SP_NOT_STANDARD_PATH indicates no standard path is applicable. Address of the null-terminated string representing the remainder of the path (relative to standard path in ax). No leading slash is returned.
Destroyed:	Nothing.	
Library:	file.def	

■ FilePos

Sets an open file's read/write position.

Pass:	a1 bx cx:dx	FilePosMode indicating how to set the position: FILE_POS_START for start of file FILE_POS_RELATIVE for current read/write position FILE_POS_END for end of file File handle of open file. 32-bit offset at which to put the read/write position. This offset will be added to the appropriate file position as determined by the FilePosMode passed in a1 .
Returns:	dx:ax	New 32-bit read/write position. This number is absolute with respect to the start of the file.
Destroyed:	Nothing.	

Assembly Reference

Library: **file.def**■ **FileRead**

Reads a number of bytes from an open file.

Pass: **al** **FileAccessFlags**. Only FILE_NO_ERRORS is valid for this routine; all others must be clear.
bx File handle of the open file.
cx Number of bytes to read from the file.
ds:dx Address of a locked or fixed buffer into which to read the data.

Returns: **CF** Set if error, clear otherwise.
ax If CF set, an error code (otherwise destroyed):
ERROR_SHORT_READ_WRITE if hit end of file
ERROR_ACCESS_DENIED if file could not be opened
cx Total number of bytes read into the buffer.
ds:dx Address of the filled buffer containing the data.

Destroyed: Nothing. (**ax** is destroyed if **CF** is set.)

Library: **file.def**

■ **FileRename**Renames the specified file. This routine may not be used for moving a file to a new directory; use **FileMove** for that purpose.

Pass: **ds:dx** Address of the current null-terminated file name.
es:di Address of the new null-terminated file name.

Returns: **CF** Set if error, clear otherwise.
ax Error code (**FileError**) if **CF** set:
ERROR_FILE_NOT_FOUND
ERROR_PATH_NOT_FOUND
ERROR_ACCESS_DENIED
ERROR_INVALID_NAME

Destroyed: Nothing.

Library: **file.def**

■ **FileResolveStandardPath**Given a path and the current directory (set to a **StandardPath**), searches the subdirectories of the standard path and returns both the full path of the desired file and its disk handle.

Pass: **ds:dx** Address of the null-terminated path to find.

FileSetAttributes

■ 76

	es:di	Address of a locked or fixed buffer into which the resulting full path will be written.
	cx	Size of the buffer in es:di .
	ax	FRSPFlags record: FRSPF_ADD_DRIVE_NAME Set if the drive name should be prepended to the returned path name. FRSPF_RETURN_FIRST_DIR Set if the routine should assume the desired path exists in the first existing directory along the standard path.
Returns:	CF	Set if file not found, clear otherwise.
	bx	Disk handle if CF is clear; destroyed if CF is set.
	al	FileAttrs record of found file or directory.
	es:di	Address of the null at the <i>end</i> of the absolute path.
Destroyed:	ah, cx (bx if CF is set)	
Library:	file.def	

■ FileSetAttributes

		Sets a file's FileAttrs record.
Pass:	cx	New attributes record (FileAttrs): FILE_ATTR_NORMAL for normal file FILE_ATTR_READ_ONLY for read-only file FILE_ATTR_HIDDEN for hidden file FILE_ATTR_SYSTEM for a system file
	ds:dx	Address of null-terminated file name.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code if CF set: ERROR_FILE_NOT_FOUND ERROR_PATH_NOT_FOUND ERROR_ACCESS_DENIED
Destroyed:	Nothing.	
Library:	file.def	

■ FileSetCurrentPath

		Sets the current directory of the calling thread.
Pass:	bx	Disk handle of the path, or a StandardPath constant. Pass zero (null handle) for the current disk handle. If a StandardPath constant is passed, the path specified in ds:dx is taken relative to that standard path.

Assembly Reference

	ds:dx	Address of a null-terminated path. The path may or may not contain the drive name, and it may be relative or absolute. If the path contains the drive name, bx should be passed as zero. The drive name in the path will be ignored.
Returns:	CF ax bx	Set if error, clear otherwise. FileError code if CF set on return: ERROR_PATH_NOT_FOUND Disk handle of new current path if bx passed as zero.
Destroyed:	Nothing.	
Library:	file.def	

■ FileSetDateAndTime

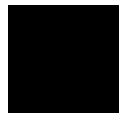
Sets an open file's modification date and time attributes.

Pass:	bx cx dx	File handle of open file. FileTime record indicating new modification time. FileDate record indicating new modification date.
Returns:	CF ax	Set if error, otherwise clear. Error code if CF returned set: ERROR_ACCESS_DENIED
Destroyed:	Nothing.	
Library:	file.def	

■ FileSetHandleExtAttributes

Sets one or more of an open file's extended attributes, given the file's handle. This is similar to **FileSetPathExtAttributes** except it specifies the file by its handle rather than its name.

Pass:	bx ax es:di cx	Handle of the open file. FileExtendedAttribute indicating the attribute(s) to set. Address of a buffer containing either the value of the extended attribute specified in ax , or an array of FileExtAttrDesc structures (if ax is FEA_MULTIPLE). Size of the buffer in es:di , or the number of entries if it is an array of structures.
Returns:	CF ax	Set if one or more attribute could not be set, either because the file system does not support it or the file can not have it. CF will be clear on a successful operation. Error code if CF set on return (destroyed if CF is clear): ERROR_ATTR_NOT_SUPPORTED



FileSetPathExtAttributes

■ 78

ERROR_ATTR_SIZE_MISMATCH
ERROR_ACCESS_DENIED

Destroyed: Nothing. (**ax** is destroyed if **CF** is clear.)

Library: **file.def**

■ FileSetPathExtAttributes

Sets one or more of a file's extended attributes, given the file's path. This is similar to **FileSetHandleExtAttributes** except it specifies the file by its name rather than its handle.

Pass:	ds:dx	Address of the null-terminated file or directory name.
	ax	FileExtendedAttribute indicating the attribute(s) to set.
	es:di	Address of a buffer containing the value of the attribute indicated in ax , or an array of FileExtAttrDesc structures if ax is FEA_MULTIPLE .
	cx	Size of the buffer in es:di , or the number of structures in the array if ax is FEA_MULTIPLE .
Returns:	CF	Set if one or more attributes could not be set, either because the file system does not support it or the file can not have it. If successful, CF will be returned clear.
	ax	FileError if CF returned set. ERROR_FILE_NOT_FOUND ERROR_ATTR_NOT_SUPPORTED ERROR_ATTR_SIZE_MISMATCH ERROR_ACCESS_DENIED

Destroyed: Nothing. (**ax** is destroyed if **CF** is returned clear.)

Library: **file.def**

■ FileSetStandardPath

Changes the thread's current directory to one of the standard system paths.

Pass: **ax** **StandardPath** value indicating the new directory.

Returns: Nothing.

Destroyed: Nothing.

Library: **file.def**

■ FileSize

Returns the size of an open file, in bytes.

Pass: **bx** File handle of open file.

Assembly Reference

Returns: **dx:ax** 32-bit size of file, in bytes.
 Destroyed: Nothing.
 Library: **file.def**

■ FileStdPathCheckIfSubDir

Checks if the given **StandardPath** constant is actually a subdirectory of another **StandardPath**.

Pass: **bp** First **StandardPath** value; checks if this is a potential parent directory.
 bx Second **StandardPath** value; checks if **bx** is a subdirectory of **bp**.
 Returns: **ax** Zero if **bx** is a subdirectory of **bp**, non-zero otherwise.
 Destroyed: Nothing.
 Library: **file.def**

■ FileTruncate

Truncates the given file to the passed length.

Pass: **al** **FileAccessFlags**; only FILE_NO_ERRORS is accepted. The other flags *must* be cleared (zero).
 bx File handle of open file to truncate.
 cx:dx 32-bit desired length of the file, in bytes.
 Returns: **CF** Set if error, clear otherwise.
 ax Error code (**FileError**) if **CF** set on return.
 File read/write position will be at the passed **cx:dx** upon return.
 Destroyed: **cx, dx**
 Library: **file.def**

■ FileUnlockRecord

Unlocks a region of an open file previously locked with **FileLockRecord**.

Pass: **bx** File handle of open file.
 cx:dx 32-bit position of start of region to unlock.
 si:di 32-bit region length.
 Returns: **CF** Set if error, clear otherwise.
 ax Error code (**FileError**) if **CF** set on return.
 Destroyed: Nothing. (**ax** is destroyed if **CF** is clear.)
 Library: **file.def**



FileWrite

■ 80

■ FileWrite

Writes a string of bytes from a buffer to an open file.

Pass:	al	FileAccessFlags ; only FILE_NO_ERRORS is accepted. The other flags <i>must</i> be clear (zero).
	bx	File handle of the open file to be written to.
	ds:dx	Address of the locked or fixed buffer containing the bytes to be written to the file.
	cx	Number of bytes in the buffer to be written.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code (FileError) if CF returned set: ERROR_SHORT_READ_WRITE (possibly disk full) ERROR_ACCESS_DENIED (file not writable)
	cx	Number of bytes successfully written to the file.
Destroyed:	Nothing. (ax is destroyed if CF is clear.)	
Library:	file.def	
Warning:	FileWrite will not truncate the file; it will only overwrite the bytes already there or append bytes to the end of the file.	

■ FloatAsciiToFloat

Converts a number represented in an ASCII text format into a GEOS FP number. The routine recognizes two flags:

Pass:	al	FloatAsciiToFloatFlags . FAF_PUSH_RESULT pushes the result onto the FP stack. FAF_STORE_NUMBER stores the result in the address passed in es:di .
	cx	Number of characters in the string that the routine should concern itself with.
	ds:di	String in this format: "[+-] dddd.dddd [Ee] [+ -] dddd"
Returns:	CF	Set if error, clear otherwise.
	es:di	Buffer filled in if FAF_STORE_NUMBER was passed in al .
Destroyed:	Nothing.	
Warning:	There can be at most a single decimal point in the passed string. Any spaces or thousands separators are ignored. The string is assumed to be legal; no error-checking on the string is performed.	
Library:	math.def	

Assembly Reference

82

Destroyed: ax

■ FloatDateNumberGetMonthAndDay

Library: **math.def**Library: **math.def**Library: **math.def**

Returns: Nothing.

Destroyed: Nothing.
Library: **math.def**

■ FloatFloatToAscii

Converts a GEOS floating point number into an ASCII string. The FP number on top of the FP stack is operated on unless *FFA_stackFrame.FFA_FROM_ADDR* is passed a value of 1.

Pass: **ss:bp** an *FFA_stackFrame* stack frame.
es:di Destination address for the string to be written. This buffer must be either *FLOAT_TO_ASCII_NORMAL_BUF_LEN* or *FLOAT_TO_ASCII_HUGE_BUFFER_LEN*.
(If *FFA_stackFrame.FFA_FROM_ADDR* is equal to 1:
ds:si Location of FP number to convert.
Returns: **cx** Number of characters in the string (excluding the null-terminator). If *cx* is equal to zero, then the string produced a NAN, either “underflow,” “overflow,” or “error.”
Destroyed: Nothing.
Library: **math.def**

■ FloatFloatToAscii_StdFormat

Converts an FP number into an ASCII string using the format passed in *a1*. This routine provides a means of converting a GEOS FP number into an ASCII string without having to set up the *FFA_stackFrame* required in **FloatFloatToAscii**.

Numbers are rounded away from 0.
E.g. if the number of fractional digits desired is equal to 1:
0.56 is rounded to 1
-0.56 is rounded to -1

Commas only apply to the integer portion of fixed and percentage format numbers. I.e. in scientific formats, the fractional and exponent portions of numbers will contain no commas even if *FFAF_USE_COMMAS* is passed.

Pass: **ax** **FloatFloatToAsciiFormatFlags**
Flags permitted:
FFAF_FROM_ADDR=1 Use the FP number at the address specified by **ds:si**.
FFAF_FROM_ADDR=0 Use the FP number on top of the FP stack. The number will be popped.
FFAF_SCIENTIFIC=1

FloatGeos80ToIEEE32

■ 84

		Returns number in the form x.xxxE+xxx in accordance with information passed in bh and b1 . FFAF_SCIENTIFIC=0 Returns number in the form xxx.xxx in accordance with information passed in bh and b1 . FFAF_PERCENT set Returns number in the form xxx.xxx% in accordance with information passed in bh and b1 . FFAF_USE_COMMAS FFAF_NO_TRAIL_ZEROS
	bh	Number of significant digits desired (must be greater than or equal to one). Fixed format numbers that require more digits than available will be forced to use scientific notation.
	b1	Number of fractional digits (number of digits following the decimal point) desired.
	ds:si	(If FFAF_FROM_ADDR is equal to 1 in ax): Address of the FP number to convert.
	es:di	Destination address for the string.
Returns:	cx	Number of characters in the string. If cx = 0 the string produced was NAN, either "underflow," "overflow," or "error."
Destroyed:	Nothing.	
Library:	math.def	

■ FloatGeos80ToIEEE32

	Converts a GEOS 80 bit FP number into a 32 bit IEEE-standard FP number.	
Pass:	Nothing. The number on top of the FP stack is converted.	
Returns:	dx:ax	32 bit number.
Destroyed:	Nothing.	
Library:	math.def	

■ FloatGeos80ToIEEE64

	Converts a GEOS FP number into a 64 bit IEEE-standard FP number and pushes it onto the FP stack.	
Pass:	es:di	Location to store the 64 bit IEEE FP number.
Returns:	CF	Set if an error occurred; clear otherwise.
Destroyed:	ax	
Library:	math.def	

Assembly Reference

■ FloatGetDateNumber

Creates a GEOS date number for the given date.

Pass:	ax	year (1900 through 2099 are valid)
	b1	month (1 through 12)
	bh	day (1 through 31)
Returns:	CF	Set if error; clear otherwise. The date number is placed on the FP stack if CF is clear.
	a1	Error code (FLOAT_GEN_ERR) if CF set.
Destroyed:	ax	
Library:	math.def	

■ FloatGetDaysInMonth

This utility routine calculates the number of days in a month when also given its year.

Pass:	ax	year (1900 through 2099)
	b1	month (1 through 12)
Returns:	bh	Number of days in the month.
Destroyed:	Nothing.	
Library:	math.def	

■ FloatGetStackDepth

Returns the current depth (in number of elements) of the floating point stack.

Pass:	Nothing.	
Returns:	ax	Stack depth.
Destroyed:	Nothing.	
Library:	math.def	

■ FloatGetTimeNumber

Calculates a GEOS time number given integral time data. GEOS time numbers are consecutive decimal values that correspond to times from midnight (0.000000) through 11:59:59 PM (0.999988).

Pass:	ch	hours (0 through 23)
	d1	minutes (0 through 59)
	dh	seconds (0 through 59)



FloatGt0

■ 86

Returns:	CF	Set if error; clear otherwise. The time number is placed on the FP stack if CF is clear.
	al	Error code (if CF is set) (FLOAT_GEN_ERR).
Destroyed:	ax	
Library:	math.def	

■ FloatGt0

Checks whether the number on top of the FP stack is greater than zero.

Pass: Nothing. The number checked is on top of the FP stack for the current thread.

Returns: CF Set if true; clear otherwise. The number on the FP stack is popped off.

Destroyed: Nothing.

Library: **math.def**

■ FloatIEEE32ToGeos80

Converts a 32 bit IEEE-standard FP number into a GEOS 80 bit FP number.

Pass: dx:ax 32 bit FP number.

Returns: Nothing. The converted number is placed on the FP stack.

Destroyed: ax, dx

Library: **math.def**

■ FloatIEEE64ToGeos80

Converts a 64 bit IEEE-standard FP number into a GEOS 80 bit FP number.

Pass: es Floating point stack segment
ds:si IEEE 64 bit FP number.

Returns: Nothing. The converted number is placed on the FP stack

Destroyed: ax

Library: **math.def**

■ FloatInit

Initializes a floating point stack for the current thread. This routine allocates a block of memory for this purpose and makes note of it in **ThreadPrivateData**.

Pass: ax Floating point stack size (in number of elements).

Assembly Reference

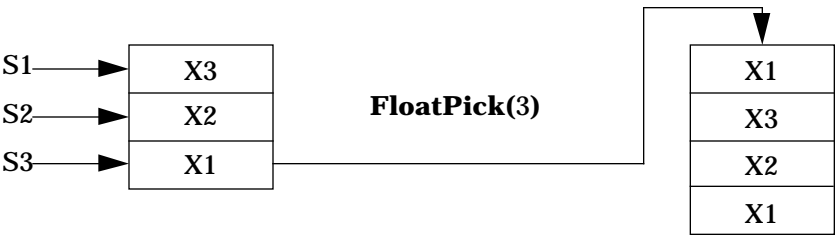
	b1	FloatStackType enumerated type indicating how to handle the exhaustion of the floating stack space: FLOAT_STACK_GROW FLOAT_STACK_WRAP FLOAT_STACK_ERROR
Returns:	bx	Handle of the floating point stack. (Normal applications will not need this handle; this is used by co-processor libraries.)
Destroyed:	Nothing.	
Library:	math.def	

FloatLt0

	Checks whether the number on top of the FP stack is less than zero.	
Pass:	Nothing. The number checked is on top of the FP stack for the current thread.	
Returns:	CF	Set if true; clear otherwise. The number on the FP stack is popped off.
Destroyed:	Nothing.	
Library:	math.def	

FloatPick

Selects an FP number on the floating point stack, copies it, and pushes it on top of the FP stack. The entire stack is pushed in the process. For example, **FloatPick** passed with a value of 3 would copy the contents of the third number on the FP stack onto the top of the stack.



Pass:	bx	Integer stack location of the FP number to copy (1 being the top FP number on the stack).
	ds	FP stack segment.
Returns:	Nothing.	
Destroyed:	ax	
Library:	math.def	

FloatPopNumber

■ 88

■ FloatPopNumber

Pops a floating point number off the FP stack into a passed location.

Pass: **es:di** Address of location to store the FP number (5 words).
Returns: CF Set if error; clear otherwise.
Destroyed: Nothing.
Library: **math.def**

■ FloatPushNumber

Pushes an FP number onto the top of the FP stack for the current thread from a passed buffer. The number must be already set up in 80 bit, FP format.

Pass: **ds:si** Address of GEOS 80 bit FP number.
Returns: CF Set if error; clear otherwise.
Destroyed: Nothing.
Library: **math.def**

■ FloatRandomize

Primes the random number generator, in preparation for a call to **FloatRandom** or **FloatRandomN**. If **FloatRandomize** is passed the flag RGIF_USE_SEED, the routine must also pass a developer supplied seed.

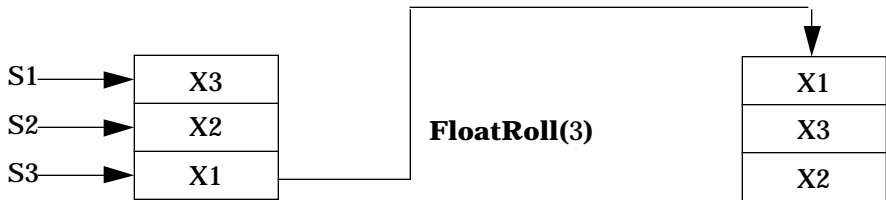
Pass: **al** **RandomGenInitFlags:**
RGIF_USE_SEED
RGIF_GENERATE_SEED
cx:dx Seed (if RGIF_USE_SEED is passed in **al**).
ds Floating point stack segment.
Returns: Nothing.
Destroyed: **ax, dx**
Library: **math.def**

■ FloatRoll

Pushes a selected FP number onto the top of the stack, removing it from its previous location in the process. **FloatRoll** passed with a value of 3 would move the FP number in the third stack position onto the top of the stack,

Assembly Reference

pushing the stack in the process. All FP numbers below the extracted number remain unaffected by this routine.



Pass: bx Position of number on FP stack to “roll.”
ds Floating point stack segment.

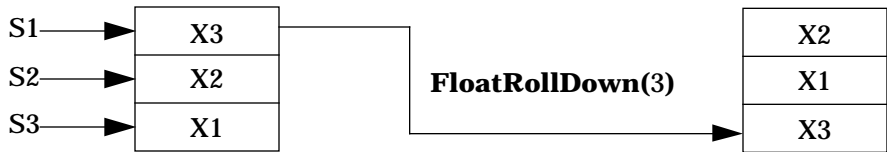
Returns: Nothing.

Destroyed: ax

Library: **math.def**

FloatRollDown

Performs the inverse operation of **FloatRoll**, popping the top stack value into a specified location on the stack. **FloatRollDown** passed with a value of 3 would move the FP number on top of the stack into the third stack location, shifting the stack in the process.



Pass: Nothing.

Returns: CF Set if error: clear otherwise.
al **FloatErrorType** if CF set.

Destroyed: ax

Library: **math.def**

FloatRound

■ 90

■ FloatRound

Rounds the top FP stack number to a given number of decimal places.

FloatRound passed with zero as an argument rounds the top FP number to the nearest integer, rounding up if greater than or equal to .5, rounding down if less than .5.

Pass: Nothing.

Returns: CF Set if error; clear otherwise.
 ax **FloatErrorType** if CF set.

Destroyed: ax

Library: **math.def**

■ FloatSetStackSize

Sets the depth of the FP stack.

Pass: ax Stack depth.

Returns: Nothing.

Destroyed: Nothing.

Library: **math.def**

■ FloatSetStackPointer

Sets the floating point stack pointer to a previous position saved with **FloatGetStackPointer**. This routine must be passed a value that is greater than or equal to the current value of the stack pointer. (I.e. you must be throwing something, or nothing, away.)

Pass: ax Desired value of the stack pointer.

Returns: CF Clear if successful; dies in EC code if unsuccessful.

Destroyed: Nothing.

Library: **math.def**

■ FloatSetStackSize

Sets the size of the FP stack.

Pass: Nothing.

Returns: Nothing.

Destroyed: Nothing.

Assembly Reference

Library: **math.def**

■ FloatStringGetDateNumber

Parses a string containing a date and returns its date number.

Pass: **es:di** String to parse.

Returns: **CF** Set if error; clear otherwise.
al (If **CF** is set) error code (**FLOAT_GEN_ERR**).
ax **DateTimeFormat** used.

Destroyed:

Library: **math.def**

■ FloatStringGetTimeNumber

Parses a string containing a time and returns its time number.

Pass: **es:di** String to parse.

Returns: **CF** Set if error; clear otherwise.
al (If **CF** is set) error code (**FLOAT_GEN_ERR**).

Destroyed:

Library: **math.def**

■ FloatWordToFloat

Converts a signed integer (word value) into a GEOS 80 bit floating point number on the FP stack.

Pass: **ax** Signed integer.

Returns: Nothing.

Destroyed: **ax, dx**

Library: **math.def**

■ FlowCheckKbdShortcut

Determines whether the key-press event maps to a shortcut.

Pass: **ds:si** Pointer to a shortcut table.
ax Number of entries in the shortcut table.
cl, ch **Character, CharacterSet** (as passed by **MSG_META_KBD_CHAR**).
dl, dh **CharFlags, ShiftState** (as passed by **MSG_META_KBD_CHAR**).



FlowDispatchSendOnOrDestroyClassedEvent

■ 92

	bp	Scan code: ToggleState (as passed by MSG_META_KBD_CHAR).
Returns:	CF si	Set if a keyboard shortcut match was found. Offset into table where shortcut was found.
Destroyed:	Nothing.	
Library:	uiInputC.def	

■ FlowDispatchSendOnOrDestroyClassedEvent

This utility routine relays a classed routine.

Pass:	*ds:si	Object instance data.
	ax	Message to send.
	cx	Handle of classed event. If Class is null, event should be sent directly to optr passed in bx:bp .
	dx	Other data to send on.
	bx:bp	Optr to relay message to if the current object isn't of the proper class to handle the message. If this optr is null and event can't be handled by the current object, then the event will be destroyed.
	di	MessageFlags for data to send on (MF_CALL also passed on to ObjDispatchMessage , if used, in order to allow for return data).
Returns:	CF	Clear if no destination, otherwise returned as per ObjMessage .
	ax, cx, dx, bp	If MF_CALL was passed and the call was completed, then these will hold the message's return values.
	ds	Update to point at segment of same block as on entry if MF_FIXUP_DS was set and destination found.
Destroyed:	Nothing.	
Library:	uiInputC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

■ FlowGetTargetAtTargetLevel

This routine retrieves the target within the current level of the target hierarchy.

Pass:	*ds:si	Object instance data.
	ax	TargetLevel of object in *ds:si .
	bx	Offset to Master part.

Assembly Reference

	di	Offset to targetExcl field of type HierarchicalGrab in instance data of object in *ds:si .
	cx	TargetLevel searching for.
Returns:	cx:dx	If the passed object is the object being searched for, then this will be the object instance data. If this object is not the one being searched for and there is no target below this node, then this will be zero. Otherwise, this will be the instance data of the object below this node that contains the target.
	ax:bp	If the passed object is the object being searched for, then this will be the class pointer for the object. If this is not the object being searched for and there is no target below this node, then this will be zero. Otherwise, this will be the class pointer of the target below this node.
	ds	Updated to point to the segment of the same block as on entry.
Destroyed:	Nothing.	
Library:	uiInputC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

■ FlowGetUIButtonFlags

	Returns the current UIButtonFlags .	
Pass:	Nothing.	
Returns:	al	UIButtonFlags structure.
Destroyed:	Nothing.	
Library:	uiInputC.def	

■ FlowReleaseGrab

	Releases the grab of the current OD if it matches that passed. The object is sent a MSG_META_LOST_..._EXCL (if specified) and the OD and data word are zeroed out to indicate that there is no current grab.	
Pass:	*ds:si	Object instance data.
	ax	Number of "gained grab" message to send (e.g. MSG_META_GAINED_MOUSE_EXCL). This message will be sent to the object gaining the grab, and the next higher message number will be sent to the object losing the grab. Because of the way these messages are set up, this will be the corresponding "lost grab" message (e.g. MSG_META_LOST_MOUSE_EXCL)



FlowRequestGrab

■ 94

	bx	Offset to Master part holding BasicGrab structure (zero if no master parts).
	di	Offset to BasicGrab structure.
	cx:dx	Number of bytes in the buffer to be written.
Returns:	CF	Set if grab OD changed and messages were sent.
	ds	Updated to point at segment of same block as on entry.
Destroyed:	Nothing.	
Library:	uiInputC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

■ FlowRequestGrab

This routine grants the grab to the OD passed if there is no active grab. If the OD passed matches that in existence then the data word is updated and no message is sent.

Pass:	*ds:si	Object instance data.
	ax	Number of “gained grab” message to send (e.g. MSG_META_GAINED_MOUSE_EXCL). This message will be sent to the object gaining the grab, and the next higher message number will be sent to the object losing the grab. Because of the way these messages are set up, this will be the corresponding “lost grab” message (e.g. MSG_META_LOST_MOUSE_EXCL)
	bx	Offset to Master part holding BasicGrab structure (zero if no master parts).
	di	Offset to BasicGrab structure.
	cx:dx	Number of bytes in the buffer to be written.
	bp	Data to be placed in BG_data field.
Returns:	CF	Set if grab OD changed and messages were sent.
	ds	Updated to point at segment of same block as on entry.
Destroyed:	Nothing.	
Library:	uiInputC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

■ FlowTranslatePassiveButton

This routine translates a MSG_META_PRE_PASSIVE_BUTTON or MSG_META_POST_PASSIVE_BUTTON to a generic message.

Assembly Reference

Pass:	ax	Message, either MSG_META_PRE_PASSIVE_BUTTON or MSG_META_POST_PASSIVE_BUTTON.
	cx, dx	Mouse position (not used here, but left intact through call).
	bp	UIFunctionsActive:ButtonInfo as passed in bp in above messages.
	cx	Number of bytes in the buffer to be written.
Returns:	ax, cx, dx, bp	Set up to send translated message (e.g. ax will be changed to the appropriate message number).
Destroyed:	Nothing.	
Library:	uiInputC.def	

FlowUpdateHierarchicalGrab

		Update exclusive based on passed message.
Pass:	*ds:si	Object instance data.
	ax	Number of “gained grab” message to implement (e.g. MSG_META_GAINED_SYS_FOCUS_EXCL).
	bx	Offset to Master part.
	di	Offset to HierarchicalGrab structure.
	bp	Base message for level exclusive (e.g. MSG_META_GAINED_...EXCL). This message will be sent to the requesting object if it gains the exclusive. The next higher message (MSG_META_LOST_...EXCL, thanks to the way these messages are ordered) will be sent when the object eventually loses the grab. The message number plus two (MSG_META_GAINED_SYS_...EXCL) is sent out after the “gained” message to the object that gains the system exclusive. If the HGF_SYS_EXCL but is set in this node, then the message plus three (MSG_META_LOST_SYS_...EXCL) is sent out before the “lost” message to any object losing the system exclusive.
Returns:	ds	Updated to point at segment of same block as on entry.
Destroyed:	Nothing.	
Library:	uiInputC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

■ GCNListAdd

Add an object or process to a GCN list. The object will be notified the next time the list's notification is sent out.

Pass:	cx:dx	The optr of the object or Process to add.
	bx:ax	The GCNListType of the list: bx <i>GCNLT_manuf</i> ax <i>GCNLT_type</i> , with GCNLTF_SAVE_TO_STATE flag set appropriately.
Returns:	CF	Set if the optr was added to the list. Clear if the optr was already there and was not added.
Destroyed:	Nothing.	
Library:	gcnlist.def	
Warning:	This routine may resize or move the LMem block, invalidating segment pointers.	

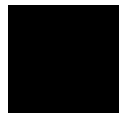
■ GCNListAddToBlock

Add a new GCN list to the block containing the GCN list of lists.

Pass:	cx:dx	The optr of the new GCN list's list chunk; this optr will be added to the list of lists.
	bx:ax	The GCNListType of the new list: bx <i>GCNLT_manuf</i> ax <i>GCNLT_type</i> , with GCNLTF_SAVE_TO_STATE flag set appropriately.
	ds	Segment address of block containing the GCN lists.
	di	Chunk handle of the list of lists chunk.
Returns:	CF	Set if optr added to GCN list of lists. Clear if optr already there and not added.
Destroyed:	Nothing.	
Library:	gcnlist.def	
Warning:	This routine may resize or move the LMem block, invalidating segment pointers.	

■ GCNListAddToList

Called by **GCNListAdd**, this routine adds an object to a specified GCN list. Most programs will call **GCNListAdd**, not this routine.



GCNListCreateBlock

■ 98

Pass:	cx:dx *ds:si	The optr of the object to add to the list. Segment:chunk handle of the GCN list to add the object to.
Returns:	CF ds	Set if the object was added to the list. Clear if the object was already there and was not added. Updated segment address of the GCN list block, if moved.
Destroyed:	Nothing.	
Library:	gcnlist.def	
Warning:	This routine may resize or move the LMem block, moving it on the heap and invalidating pointers into it.	

■ GCNListCreateBlock

Create a new GCN list of lists within a locked LMem block.

Pass:	ds	Segment address of locked LMem block.
Returns:	*ds:si	Segment:chunk handle of new list of lists chunk. The ds register will be updated if the block is moved.
Destroyed:	Nothing.	
Library:	gcnlist.def	

■ GCNListCreateList

Create an empty GCN list within a locked LMem block.

Pass:	ds bx cx	Segment address of locked LMem block. Size of one element in the list, typically the size of the GCNListElement structure. Size of the list header, typically size of GCNListHeader .
Returns:	*ds:si	Segment:chunk handle of the new GCN list chunk.
Destroyed:	Nothing.	
Library:	gcnlist.def	

■ GCNListDestroyBlock

Cleanly destroy a GCN list of lists and all the lists in it.

Pass:	ds:di	Chunk containing the list of lists and the GCN lists.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	gcnlist.def	

Assembly Reference

■ GCNListDestroyList

Cleanly destroy a GCN list.

Pass: ***ds:si** Segment:chunk handle of the GCN list chunk to destroy.

Returns: Nothing.

Destroyed: Nothing.

Library: **gcnlist.def**

■ GCNListFindItemInList

Return the address of the GCN list entry corresponding to the optr passed.

Pass: **cx:dx** The optr to search for. Pass **dx = 0** to match any chunk handle in the specified object block.

***ds:si** Segment:chunk handle of the GCN list to search.

Returns: **CF** Set of the optr was found in the list, clear if not found.

ds:di Address of list entry in the GCN list corresponding to the passed optr, if found in the list.

ds Updated to keep pointing at the GCN list block, if moved.

Destroyed: Nothing.

Library: **gcnlist.def**

■ GCNListFindListInBlock

Find the appropriate GCN list entry in the GCN list of lists. If necessary and desired, this routine creates a new GCN list and adds it to the list of lists.

Pass: ***ds:si** Segment:chunk handle of the GCN list of lists chunk.

bx:ax **GCNListType** indicating the GCN list:
bx *GCNLT_manuf*
ax *GCNLT_type*

CF Set to create list if it does not already exist.
Clear to suppress list creation.

Returns: **CF** Set if the list was found or created, clear if the list was not found and was not created.

***ds:si** Segment:chunk handle of the new or found GCN list chunk, if **CF** returned set.

Destroyed: Nothing.

Library: **gcnlist.def**

■ GCNListRecordAndSend

Broadcast the given message to all the objects on the specified GCN list. This is a utility message used to make broadcasting such messages easier.

Pass:	ax	Message to broadcast.
	bx:si	GCNListType of the GCN list to broadcast to:
		bx <i>GCNLT_manuf</i>
		si <i>GCNLT_type</i>
	cx, dx, bp	Data to pass with the message.
	di	GCNListSendFlags to set options.
Returns:	cx	Total number of messages dispatched to objects on the list.
Destroyed:	ax, dx, bp, di, si	
Library:	gcnlist.def	

■ GCNListRelocateBlock

Relocate all the GCN lists within the given locked block, updating optrs and other handles.

Pass:	ds	Segment address of the locked block.
	di	Chunk handle within the block of the GCN list of lists.
	dx	Handle of the block containing the relocation information.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	gcnlist.def	
Warning:	This routine may resize or move the LMem block, invalidating all pointers into it.	

■ GCNListRelocateList

Relocate a particular GCN list. This routine is called by **GCNListRelocateBlock** and will not generally be called by applications.

Pass:	*ds:si	Segment:chunk handle of GCN list to relocate.
	dx	Handle of the block containing the relocation information.
Returns:	ds	Updated to keep pointing to the GCN list block, if moved.
Destroyed:	Nothing.	
Library:	gcnlist.def	
Warning:	This routine may resize or move the LMem block, invalidating all pointers into it.	

■ GCNListRemove

Remove an optr from a specified GCN list.

Pass:	cx:dx	The optr to be removed from the list.
	bx:ax	GCNListType: bx <i>GCNLT_manuf</i> ax <i>GCNLT_type</i>
Returns:	CF	Set if optr found and removed, clear if optr not found in list.
Destroyed:	Nothing.	
Library:	gcnlist.def	

■ GCNListRemoveFromBlock

Remove the given optr from the specified GCN list type in the passed block.

Pass:	cx:dx	The optr to be removed from the lists.
	bx:ax	GCNListType: bx <i>GCNLT_manuf</i> ax <i>GCNLT_type</i>
	ds	Segment of locked block containing the GCN lists.
	di	Chunk handle within the locked block of the GCN list of lists chunk.
Returns:	CF	Set of the optr was found and removed, clear if not found.
Destroyed:	Nothing.	
Library:	gcnlist.def	

■ GCNListRemoveFromList

Remove an optr from a GCN list. This routine is called by GCNListRemove; in general, applications should call GCNListRemove instead of this routine.

Pass:	cx:dx	The optr to remove from the list.
	*ds:si	Segment:chunk handle of the GCN list to remove the optr from.
Returns:	CF	Set if the optr was found and removed from the list, clear if the optr was not found.
	ds	Updated to point to the GCN list block, if moved.
Destroyed:	Nothing.	
Library:	gcnlist.def	

■ GCNListSend

Send a specified message to each element of a GCN list. If a data block with a reference count is passed to this routine, the reference count should be incremented before the call; otherwise, this routine will decrement the count and free the block if the count reaches zero.

Pass:	bx:ax	GCNListType:
		bx <i>GCNLT_manuf</i>
		ax <i>GCNLT_type</i>
	cx	Handle of a recorded classed event to be sent out. The destination class of this event is ignored.
	dx	Zero unless sending extra data block with the message. Otherwise, this is the handle of the extra data block; this handle may also be stored in the classed event according to the message parameters in the event.
	bp	GCNListSendFlags:
		GCNLSF_SET_STATUS
		Saves the message as the GCN list's current "status." The "status" message is automatically sent to any object adding itself to the list later.
		GCNLSF_IGNORE_IF_STATUS_TRANSITIONING
		Has no effect in this routine.
Returns:	cx	Total number of message dispatched, if GCNLSF_SET_STATUS was not passed. Event handle of the status event, if GCNLSF_SET_STATUS was passed.
Destroyed:	Nothing.	
Library:	gcnlist.def	
Warning:	Data blocks with reference counts will have their reference counts decremented by this routine. To ensure the block does not get freed accidentally, increment the reference count before calling this routine.	

■ GCNListSendToBlock

Send a specified message to each element in a GCN list. This routine differs from **GCNListSend** only in that you pass a pointer to the list of lists to indicate a particular GCN list block (usually a custom block) to use. Other parameters, warnings, etc., are identical.

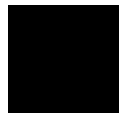
Pass:	bx:ax	GCNListType:
		bx <i>GCNLT_manuf</i>
		ax <i>GCNLT_type</i>

	cx	Handle of a recorded classed event to be sent out. The destination class of this event is ignored.
	dx	Zero unless sending extra data block with the message. Handle of the extra data block; this handle may also be stored in the classed event according to the message parameters in the event.
	bp	GCNListSendFlags: GCNLSF_SET_STATUS Saves the message as the GCN list's current "status." The "status" message is automatically sent to any object adding itself to the list later. GCNLSF_IGNORE_IF_STATUS_TRANSITIONING Has no effect in this routine.
	ds	Segment address of locked GCN list block.
	di	Chunk handle of GCN list of lists within the locked block.
Returns:	cx	Total number of message dispatched, if GCNLSF_SET_STATUS was not passed. Event handle of the status event, if GCNLSF_SET_STATUS was passed.
Destroyed:	Nothing.	
Library:	gcnlist.def	
Warning:	Data blocks with reference counts will have their reference counts decremented by this routine. To ensure the block does not get freed accidentally, increment the reference count before calling this routine.	

■ GCNListSendToList

Dispatches a classed event to all the elements on a particular GCN list. This routine is called by both **GCNListSend** and **GCNListSendToBlock**. In general, you should use one of those two routines rather than this routine.

Pass:	*ds:si	Segment:chunk handle of the GCN list to use.
	cx	Handle of a recorded classed event to be sent out. The destination class of this event is ignored.
	dx	Zero unless sending extra data block with the message. Handle of the extra data block; this handle may also be stored in the classed event according to the message parameters in the event.
	bp	GCNListSendFlags: GCNLSF_SET_STATUS Saves the message as the GCN list's current "status." The "status" message is



GCNListUnRelocateBlock

■ 104

automatically sent to any object adding itself to the list later.

GCNLSF_IGNORE_IF_STATUS_TRANSITIONING
Has no effect in this routine.

Returns: **cx** Total number of message dispatched, if GCNLSF_SET_STATUS was not passed.
Event handle of the status event, if GCNLSF_SET_STATUS was passed.

Destroyed: Nothing.

Library: **gcnlist.def**

Warning: Data blocks with reference counts will have their reference counts decremented by this routine. To ensure the block does not get freed accidentally, increment the reference count before calling this routine.

■ GCNListUnRelocateBlock

Unrelocate all the GCN lists within the specified, locked GCN list block. This routine will be called only in very rare situations by applications.

Pass: **ds** Segment address of locked GCN list block.
di Chunk handle of list of lists within the list block.
dx Handle of the block containing relocation information.

Returns: **CF** Clear if list of lists saves lists to state.
Set if list of lists is destroyed because it saves no lists to state.

Destroyed: Nothing.

Library: **gcnlist.def**

Warning: This routine may resize or move the GCN list block, invalidating any pointers to that block.

■ GCNListUnRelocateList

Unrelocate a particular GCN list. This routine is called by **GCNListUnRelocateBlock**; applications should call that routine instead.

Pass: ***ds:si** Segment:chunk handle of the GCN list to unrelocate.
dx Handle of the block containing relocation information.

Returns: **ds** Updated to keep pointing to the GCN list block, if moved.

Destroyed: Nothing.

Library: **gcnlist.def**

Warning: This routine may resize or move the GCN list block, invalidating any pointers to that block.

Assembly Reference

■ GenControlOutputActionRegs

This utility routine calls MSG_GEN_OUTPUT_ACTION. This is used when a controller needs to send out an action. This handles GenAttrs such as GA_SIGNAL_INTERACTION_COMPLETE, GA_INITIATES_BUSY_STATE, and GA_INITIATES_INPUT_HOLD_UP.

This routine is normally used only within the context of a controller method.

Pass:	*ds:si	Object pointer of controller.
	bx:di	Class of controller.
	ax	Message to send. This message will be recorded and passed as one of the arguments to MSG_GEN_OUTPUT_ACTION.
	cx, dx, bp	Registers to pass with recorded message.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	gCtrlC.def	

■ GenControlOutputActionStack

Utility routine to call MSG_GEN_OUTPUT_ACTION. This is used when a controller needs to send out an action. This handles GenAttrs such as GA_SIGNAL_INTERACTION_COMPLETE, GA_INITIATES_BUSY_STATE and GA_INITIATES_INPUT_HOLD_UP.

Pass:	*ds:si	object
	bx:di	class
	ax	message
	ss:bp	data
	dx	data size—How many bytes pushed on stack
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	gCtrlC.def	

■ GenControlSendToOutputRegs

Utility routine to send a message to the output of a controller. This message may take arguments passed in via registers.

Pass:	*ds:si	Optr of controller object.
	bx:di	Class of controller object.
	ax	Message to send.
	cx, dx, bp	Data for message.
Returns:	Nothing.	



GenControlSendToOutputStack

■ 106

Destroyed: Nothing.

Library: **ui**

■ GenControlSendToOutputStack

Utility routine to send a message to the output of a controller. This message may take arguments passed via the stack.

Pass: ***ds:si** Optr of controller object.
 bx:di Class of controller object.
 ax Message to send.
 ss:bp Data for message, pushed on stack.
 dx data size—How many bytes pushed on stack.

Returns: Nothing.

Destroyed: Nothing.

Library: **gCtrlC.def**

■ GenItemSendMsg

This utility routine sends a message to a GenItemGroup's destination, with the usual arguments.

Pass: ***ds:si** Segment:chunk handle of the GenItemGroup.
 ax Message to send.
 c1 State flags to pass.
 di Non-zero to close window if we should check to close the window.

Returns: Nothing.

Destroyed: Nothing.

Library: **gItemGC.def**

■ GenPathConstructFullPath

This utility routine constructs a full path from the object's path stored under the passed vardata type.

Because the kernel returns **StandardPath** constants and relative path tails (and this is how paths are stored in a **GenFilePath**),

GenPathGetObjectPath will not always get you what you need: a user-readable representation of the full path bound to the object. This routine gives you that.

Pass: ***ds:si** Segment:chunk handle of the GenItemGroup.
 es:di Buffer in which to store the constructed path.

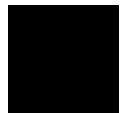
Assembly Reference

	cx	Number of bytes in the buffer (ignored if es is zero).
	ax	Vardata type under which the path is stored.
	dx	Vardata type under which the disk handle should be saved for shutdown.
	bp	Non-zero to place drive specifier before the returned absolute path.
Returns:	CF	Set if passed buffer is too small.
	bx	If the path fit in the buffer, this holds the disk handle for the path.
	es:di	If the whole path fits in the buffer, then this points at the constructed string's null terminator.
Destroyed:	ax, dx, bp.	
Library:	genC.def	

■ GenPathGetObjectPath

This utility scans the requested variable data field and fetches the path stored in that field.

Pass:	*ds:si	Segment:chunk handle of the GenItemGroup.
	es:di	Buffer in which to store the path. If es is zero, a block will be allocated for the path and the handle returned.
	cx	Number of bytes in the buffer (ignored if es is zero).
	ax	Vardata type under which the path is stored.
	dx	Vardata type under which the disk handle should be saved for shutdown.
Returns:	CF	Set if passed buffer is too small.
	ax	If there was an error, this register holds the number of bytes required to hold the path, or zero if the path is invalid.
		If there was no error and if the passed es was zero, then this register holds the handle of the block containing the block containing the path.
	cx	Disk handle for the path.
	es:di	If there was no error and the passed es was not zero, then this is a filled, null-terminated buffer.
Destroyed:	bx, dx.	
Library:	genC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	



GenPathSetCurrentPathFromObjectPath

■ 108

■ GenPathSetCurrentPathFromObjectPath

This utility routine sets the thread's current path to the value stored in a GenFilePath in the object's vardata.

Pass:	*ds:si	Segment:chunk handle of the GenItemGroup.
	ax	Vardata type under which the path is stored.
	dx	Vardata type under which the disk handle is saved.
Returns:	CF	Set if current path couldn't be set.
	ax	If there was an error, this register holds the error code, of type FileError .
Destroyed:	ax .	
Library:	genC.def	

■ GenPathSetObjectPath

This utility routine changes the path stored in the indicated vardata entry to match that passed.

Pass:	*ds:si	Segment:chunk handle of the GenItemGroup.
	es:di	Path to set (may not be in same block as object).
	ax	Vardata type under which the path is stored.
	dx	Vardata type under which the disk handle should be saved for shutdown.
	bp	Disk handle (or StandardPath).
Returns:	CF	Set if passed path is invalid.
	ax	If there was an error, this register holds the error code, of type FileError .
	ds	Updated to point to same block as passed ds .
Destroyed:	bx, cx, dx .	
Library:	genC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

■ GenPathUnrelocObjectPath

This utility routine changes the path stored in the indicated vardata entry to match that passed.

This routine must be called when object receives MSG_META_UNRELOCATE—this implies any class using these routines must have a handler bound to the “reloc” keyword.

Pass:	*ds:si	Segment:chunk handle of the GenItemGroup.
-------	---------------	---

Assembly Reference

	ax	Vardata type under which the path is stored.
	dx	Vardata type under which the disk handle should be saved for shutdown.
Returns:	CF	Cleared.
Destroyed:	Nothing.	
Library:	genC.def	

■ GenRelocMonikerList

This utility routine sends a message to a GenItemGroup's destination, with the usual arguments. You can think of this routine as the equivalent of a MSG_GEN_RELOC_MONIKER_LIST.

Pass:	*ds:cx dx	Visual moniker—may be a moniker list. Block in which moniker list came from and will be stored back to. This is the block whose owner has the correct relocation handles to use. In most cases ds:[LMBH_handle] will work fine. this option is offered for cases where an unrelocated moniker list is copied out of one library's resource and into a block owned by another geode. In this latter case, the block handle in which the moniker list came from should be passed.
	bp	Zero to relocate the list, one to unrelocate it.
Returns:	CF	Set if ObjDoRelocation or ObjDoUnrelocation (as appropriate) returned carry set. Note that carry set is a sign of an error for both cases.
Destroyed:	Nothing.	
Library:	genC.def	

■ GenReturnTrackingArgs

This utility routine sends the track scrolling information structure back to the object which signalled the scroll event.

Pass:	ss:bp cx ds	TrackScrollingParams. Caller's chunk handle. Segment of LMem block or block in which ds:[LMBH_handle] is the block handle.
Returns:	ds	Updated to point at segment of same block as on entry.
Destroyed:	Nothing.	
Library:	gViewC.def	



GenSetupTrackingArgs

■ 110

Warning: This routine may resize LMem and/or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ GenSetupTrackingArgs

This utility routine fills in extra data for track scrolling. Normally this routine is only called by a view.

Pass: **cx:dx** TrackScrollingParams.
***ds:si** Pointer to view's instance data.

Returns: **ss:bp** (unchanged) updated **TrackScrollingParams**.

Destroyed: Nothing.

Library: **gViewC.def**

Warning: This routine may resize LMem and/or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ GenValueSendMsg

This utility routine sends a message to a GenValue's destination.

Pass: ***ds:si** Optr of object.
ax Message to send.
cl State flags to pass. A **GenValueStateFlags** value.

Returns: Nothing.

Destroyed: **ax, cx, dx, bp, di, si**

Library: **gValueC.def**

■ GenViewSendToLinksIfNeeded

This utility routine encapsulates the current message and sends it to the GenView's linked views, if there are any.

Pass: **ds:si** Object pointer of GenView.
ax Message to send.
cx, dx, bp Arguments to message.
di **MessageFlags**. This should be MF_STACK if a stack message, zero otherwise.

Returns: **CF** Set if message successfully sent via MSG_GEN_VIEW_SEND_TO_LINKS with **ax, cx, dx, bp** destroyed. Clear if message was not sent to links and should be handled normally.

Destroyed: **ax, cx, dx, bp** if message successfully sent.

Library: **gViewC.def**

Assembly Reference

■ GenViewSetSimpleBounds

This utility routine fills in extra data for track scrolling.

Pass: **bx:si** View handle.
 cx 16 bit right bound to set (width of your document).
 dx 16 bit bottom bound to set (height of your document).

Returns: Nothing.

Destroyed: **ax, cx, dx, bp**.

Library: **gViewC.def**

■ GeodeAddReference

Artificially increase the reference count of the specified geode. This is useful for geodes which want to make sure they determine when they exit, by keeping the reference count artificially above zero.

Pass: **bx** Geode handle of the geode to be referenced.

Returns: Nothing.

Destroyed: Nothing.

Library: **geode.def**

■ GeodeAllocQueue

Allocate an event queue.

Pass: Nothing.

Returns: **bx** Handle of created queue.

Destroyed: Nothing.

Library: **geode.def**

■ GeodeDuplicateResource

Load a resource from a geode's file into a new block of memory.

Pass: **bx** Handle of resource to duplicate.

Returns: **bx** Handle of newly-allocated duplicate block.

Destroyed: Nothing.

Library: **resource.def**

■ GeodeFind

Find a geode, given its name, and return its handle.



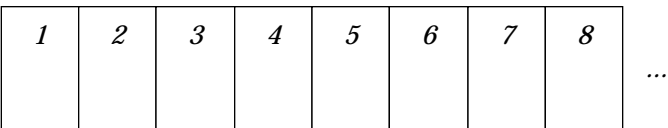
GeodeFindResource

■ 112

Pass:	es:di	Address of null-terminated permanent name of the geode to be searched for.
	ax	Number of characters in the name in es:di . Pass 8 to match the name only, 12 to match name plus extension characters.
	cx	GeodeAttrs that must be set in the geode for positive match.
	dx	GeodeAttrs that must be clear for positive match.
Returns:	CF	Set if the geode was found, clear otherwise.
	bx	Geode handle of the geode.
	cx, dx	Preserved.
Destroyed:	Nothing.	
Library:	geode.def	

■ GeodeFindResource

Locate a particular resource in the geode's **.geo** file. Every geode is laid out as shown in the following diagram:



- 1 GeodeFileHeader
- 2 Imported library table
- 3 Exported routine table
- 4 Resource size table
- 5 Resource position table
- 6 Relocation table size table
- 7 Allocation flags table
- 8... Other resources

Pass:	bx	File handle of open .geo file.
	cx	Resource number to find.
	dx	Offset within resource at which to set the file's read/write position.
Returns:	cx:dx	32-bit base position of the resource in the file.

Assembly Reference

ax Size of the resource in bytes.
 Destroyed: Nothing.
 Library: **geode.def**

■ GeodeFlushQueue

Flush all events in one queue to another queue, synchronously.

Pass: **bx** handle of queue to flush
 si handle of destination queue
 cx:dx OD to send any event to which was previously destined for
 the method table of the thread reading the source queue (To
 specify the method table of the thread reading the
 destination queue pass the destination queue handle in cx)
 di **MessageFlags** field. Only the following request applies:
 MF_INSERT_AT_FRONT
 Set to flush source queue's events to the front of the destination
 queue. If clear, events are placed at the back.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **geode.def**

■ GeodeFreeDriver

Free a driver, given its geode handle. The driver should have been loaded with **GeodeUseDriver**.

Pass: **bx** Geode handle of the driver.
 Returns: CF Set if library was exited.
 Destroyed: **bx**
 Library: **driver.def**

■ GeodeFreeLibrary

Free a library, given its geode handle. The library should have been loaded with **GeodeUseLibrary**.

Pass: **bx** Geode handle of the library.
 Returns: CF Set if library was exited.
 Destroyed: **bx**
 Library: **library.def**



■ GeodeFreeQueue

Free an event queue.

Pass: **bx** Handle of queue to free (or thread handle to free queue for).

Returns: Nothing.

Destroyed: **bx**

Library: **geode.def**

■ GeodeGetAppObject

Return the optr of the **GenApplicationClass** object for the specified process.

Pass: **bx** Geode handle of the process to query;
Zero for the current-running process.

Returns: **bx:si** The optr of the **GenApplicationClass** object.

Destroyed: Nothing.

Library: **geode.def**

■ GeodeGetDefaultDriver

Return the handle of the default driver for the indicated type.

Pass: **ax** **GeodeDefaultDriverType** value indicating the type of driver to check the default for.

Returns: **ax** Geode handle of the default driver for the given type.

Destroyed: Nothing.

Library: **driver.def**

■ GeodeGetDGroupDS

Return the address of the dgroup segment of the geode owning the memory handle passed. The segment is put into **ds**.

Pass: **bx** Memory handle owned by the subject geode. Often this will be the handle portion of an object's optr.

Returns: **ds** Segment address of the dgroup segment of the geode.

Destroyed: Nothing.

Library: **resource.def**

■ **GeodeGetDGroupES**

Return the address of the dgroup segment of the geode owning the memory handle passed. The segment is put into **es**.

Pass: **bx** Memory handle owned by the subject geode. Often this will be the handle portion of an object's optr.

Returns: **es** Segment address of the dgroup segment of the geode.

Destroyed: Nothing.

Library: **resource.def**

■ **GeodeGetGeodeResourceHandle**

Return the resource handle of the specified resource in the given geode.

Pass: **ax** Resource number to find.

bx Geode handle of the geode owning the resource.

Returns: **bx** Handle of the resource.

Destroyed: Nothing.

Library: **resource.def**

■ **GeodeGetInfo**

Return information about a particular geode.

Pass: **ax** **GeodeGetInfoType** indicating the type of information returned by the routine:

 GGIT_ATTRIBUTES Returns geode's **GeodeAttrs** in **ax**.

 GGIT_TYPE Returns geode's type in **ax**.

 GGIT_GEODE_RELEASE Returns geode's **ReleaseNumber** structure in the passed buffer.

 GGIT_GEODE_PROTOCOL Returns geode's **ProtocolNumber** structure in the passed buffer.

 GGIT_TOKEN_ID Returns geode's **GeodeToken** structure in the passed buffer.

 GGIT_PERM_NAME_AND_EXT Returns geode's permanent name and extension characters in the passed buffer. The buffer must be at least



GeodeGetProcessHandle

■ 116

GEODE_NAME_SIZE +
GEODE_NAME_EXT_SIZE characters.
GGIT_PERM_NAME_ONLY
Returns the geode's permanent name
without extension characters in the passed
buffer. The buffer must be at least
GEODE_NAME_SIZE characters.

bx Geode handle of geode to return information about. (Zero to
get information about the current buffer.)

es:di Address of a locked or fixed buffer into which the results for
the following types are placed:
GGIT_GEODE_RELEASE
GGIT_GEODE_PROTOCOL
GGIT_TOKEN_ID
GGIT_PERM_NAME_AND_EXT
GGIT_PERM_NAME_ONLY

Returns: **ax** Value dependent on **GeodeGetInfoType** passed in **ax**, or
zero if the return value is in the passed buffer.

es:di Depending on **GeodeGetInfoType**, this buffer may have been
filled.

Destroyed: Nothing.

Library: **geode.def**

■ GeodeGetProcessHandle

Return the geode handle of the current process.

Pass: Nothing.

Returns: **bx** Geode handle of the current process.

Destroyed: Nothing.

Library: **geode.def**

■ GeodeGetResourceHandle

Return the resource handle of the specified resource identifier owned by the
current geode.

Pass: **bx** Resource number to find the handle of.

Returns: **bx** Resource handle of the found resource.

Destroyed: Nothing.

Library: **resource.def**

Assembly Reference

■ GeodeGetUIData

Return the UI data (a reserved word which should not be used by applications) for the specified process.

Pass: **bx** Geode handle of the process to check.
Zero for the current process.

Returns: **bx** The UI data.

Destroyed: Nothing.

Library: **geode.def**

■ GeodeInfoDriver

Return a pointer to the specified driver's information block.

Pass: **bx** Geode handle of the driver to be checked.

Returns: **ds:si** Pointer to a locked block containing the driver's **DriverInfoStruct** structure.

Destroyed: Nothing.

Library: **driver.def**

■ GeodeInfoQueue

Return information about a driver.

Pass: **bx** Handle of driver.

Returns: **ds:si** Driver's info block (type **DriverInfoStruct**).

Destroyed: Nothing.

Library: **geode.def**

■ GeodeLoad

Load a geode from the given file and execute it based on its type.

Pass: **al** Priority for new geode, if it's an application.
ah Zero in all cases.
cx **GeodeAttrs** to set for the newly-loaded geode. (See below).
dx **GeodeAttrs** to clear for the newly-loaded geode. (See below).
di:bp Two words of information for the new geode. For libraries and drivers, **di:bp** should be a far pointer to a null-terminated string of parameters. For processes, **di** should be passed in **cx** and **bp** should be passed in **dx**.
ds:si Address of a string containing the geode's file name.



GeodePrivAlloc

■ 118

Returns:	CF	Set if error, clear otherwise.
	ax	Error code (GeodeLoadError) if CF returned set: GLE_FILE_NOT_FOUND GLE_FILE_READ_ERROR GLE_NOT_GEOS_FILE GLE_NOT_GEOS_EXECUTABLE_FILE GLE_FILE_TYPE_MISMATCH GLE_ATTRIBUTE_MISMATCH GLE_MEMORY_ALLOCATION_ERROR GLE_PROTOCOL_ERROR_IMPORTER_TOO_RECENT GLE_PROTOCOL_ERROR_IMPORTER_TOO_OLD GLE_NOT_MULTI_LAUNCHABLE GLE_LIBRARY_PROTOCOL_ERROR GLE_LIBRARY_LOAD_ERROR GLE_DRIVER_INIT_ERROR GLE_LIBRARY_INIT_ERROR
	bx	Geode handle of the newly-loaded geode, if successful.
Destroyed:	Nothing.	
Library:	geode.def	

■ GeodePrivAlloc

Allocates a string of contiguous words in the geode's private data area.

Pass:	bx	Geode handle of the geode that will "own" the space.
	cx	Number of words to allocate.
Returns:	bx	Offset to (token of) range of words, or zero if unsuccessful.
Destroyed:	Nothing.	
Library:	geode.def	

■ GeodePrivFree

Frees a group of contiguous words from the geode's private data area.

Pass:	bx	Offset to (token of) range of words, as returned by GeodePrivAlloc .
	cx	Number of words to free.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	geode.def	

Assembly Reference

■ GeodePrivRead

Reads a number of words from the geode's private data area, copying them into a passed buffer.

Pass:	bx	Geode handle of the geode that "owns" the private data area. Zero for the current process' geode handle.
	di	Offset to range of words, as returned by GeodePrivAlloc .
	cx	Number of words to read into the buffer.
	ds:si	Address of a locked or fixed buffer to receive the data.
Returns:	ds:si	The buffer will contain all the words read. If no data was previously written to the private data area, the buffer will contain all zeroes.
Destroyed:	Nothing.	
Library:	geode.def	

■ GeodePrivWrite

Writes a number of words into the specified geode's private data area.

Pass:	bx	Geode handle of the geode that "owns" the private data area.
	di	Offset to range of words, as returned by GeodePrivAlloc .
	cx	Number of words to write.
	ds:si	Address of a locked or fixed buffer containing the data to write out.
Returns:	CF	Set if the data could not be written (likely out of memory).
Destroyed:	Nothing.	
Library:	geode.def	

■ GeodeRemoveReference

Remove an extra reference to the specified geode, decrementing its reference count. If the reference count drops to zero, the geode will be removed from memory. This is the counterpart to **GeodeAddReference**, which artificially increases the geode's reference count.

Pass:	bx	Geode handle of the subject geode.
Returns:	CF	Set if reference count dropped to zero as a result of this call, clear otherwise.
Destroyed:	Nothing.	
Library:	geode.def	



GeodeSetDefaultDriver

■ 120

■ GeodeSetDefaultDriver

Sets the specified driver to be the default driver for the passed type.

Pass:	ax	GeodeDefaultDriverType.
	bx	Geode handle of the driver to be set as the default for the type passed in ax .
Returns:	Nothing.	
Destroyed:	ax	
Library:	driver.def	

■ GeodeSetUIData

Set the word of UI data for the specified process. This data is opaque to applications and should not be used nor set by them.

Pass:	ax	Word to set as UI data.
	bx	Geode handle of the process, or zero for the current process.
Returns:	bx	Geode handle of the process.
Destroyed:	Nothing.	
Library:	geode.def	

■ GeodeUseDriver

Dynamically loads a driver given its file name.

Pass:	ds:si	Address of the null-terminated file name of the driver to load.
	ax	Expected major protocol number (pass zero to ignore the major protocol).
	bx	Expected minor protocol number.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code (GeodeLoadError) if CF returned set.
	bx	Geode handle of the loaded driver if successful.
Destroyed:	ax	
Library:	driver.def	

■ GeodeUseLibrary

Dynamically use the given library. If it is not already loaded, find it on the disk and load it. If it is loaded, simply increment its reference count and create an additional core block for it.

Assembly Reference

Whenever possible, you should include the libraries you need in your .def files; use **GeodeUseLibrary** only when it's absolutely necessary.

Pass:	ds:si	Address of the null-terminated file name of the library.
	ax	Expected major protocol number (pass zero to ignore it).
	bx	Expected minor protocol number.
Returns:	CF	Set if error, clear otherwise.
	ax	Error code (GeodeLoadError) if CF returned set.
	bx	Geode handle of the library if successful.
Destroyed:	Nothing	
Library:	library.def	

■ GrApplyRotation

Apply a rotation to the transformation matrix for a GState, then mark the current transformation as invalid. The effects are cumulative to previous transformations.

$$\begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix} = \begin{bmatrix} scaleX & 0 & 0 \\ 0 & scaleY & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix}$$

$$GS_TMatrix = \quad scaling \quad * GS_TMatrix$$

Pass:	di	Handle of the GState.
	dx:cx	32-bit signed integer representing the angle of rotation multiplied by 65536 (angle * 65536).
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	



■ GrApplyScale

Apply a scale factor to a GState's transformation matrix, then mark the current transformation as invalid. The effects are cumulative to previous transformations.

$$\begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix}$$

$$GS_TMatrix = rotation * GS_TMatrix$$

Pass: **di** Handle of the GState.
dx:cx 32-bit horizontal scale factor to apply.
bx:ax 32-bit vertical scale factor to apply.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrApplyTransform

Apply a full transformation to the GState's transformation matrix, then mark the current transformation as invalid. The effects are cumulative to previous transformations. This routine requires advanced matrix manipulation; most applications will not use this routine directly.

$$\begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix} = \begin{bmatrix} e11 & e12 & 0 \\ e21 & e22 & 0 \\ e31 & e32 & 1 \end{bmatrix} \times \begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix}$$

$$GS_TMatrix = matrix * GS_TMatrix$$

Pass: **di** Handle of the GState.
ds:si Address of a buffer containing the new TMatrix elements to use. The elements should be four 32-bit fixed point numbers (**WWFixed** format) and two 48-bit fixed point numbers

(**DWFixed** format) arranged in row order. The last two elements are the 48-bit numbers.

$$\begin{bmatrix} e11 & e12 & 0 \\ e21 & e22 & 0 \\ e31 & e32 & 1 \end{bmatrix} \Rightarrow e11, e12, e21, e22, e31, e32$$

Transformation Matrix *Array Order*

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrApplyTranslation

Apply a translation to the passed GState's transformation matrix, then mark the current transformation as invalid if necessary. The effects are cumulative to previous transformations.

$$\begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ transX & transY & 1 \end{bmatrix} \times \begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix}$$

$$GS_TMatrix = translation * GS_TMatrix$$

Pass: **di** Handle of the GState.
 dx:cx 32-bit **WWFixed** value representing X translation.
 bx:ax 32-bit **WWFixed** value representing Y translation.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**



■ GrApplyTranslationDWord

Apply a 32-bit extended translation to the GState's transformation matrix, then mark the current transformation as invalid, if necessary. The effects are cumulative to previous transformations.

$$\begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ transX & transY & 1 \end{bmatrix} \times \begin{bmatrix} gs11 & gs12 & 0 \\ gs21 & gs22 & 0 \\ gs31 & gs32 & 1 \end{bmatrix}$$

$$GS_TMatrix = translation * GS_TMatrix$$

Pass: **di** Handle of the GState.
dx:cx Signed dword representing X translation.
bx:ax Signed dword representing Y translation.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrBeginPath

Start a new graphics path definition, or alter the existing current path. All graphics operations operated after this routine until **GrEndPath** become part of the path.

Pass: **di** Handle of the GState for which the path will be effective.
cx **PathCombineType** value:
PCT_NULL, PCT_REPLACE, PCT_UNION, or
PCT_INTERSECTION

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrBeginUpdate

Begin an update of an exposed, visible region. This routine is called by applications upon receipt of MSG_META_EXPOSED. Drawing to the GState passed to this routine will be clipped to the exposed region. After drawing is complete, the application must call **GrEndUpdate**.

Pass: **di** Handle of the GState used for drawing to the exposed area of the window. Usually created with **GrCreateState** and

destroyed with **GrDestroyState** in a MSG_META_EXPOSED handler.

Returns: Nothing.
Destroyed: Nothing.
Library: **win.def**

■ GrBitBlt

Transfer a bit-boundary block of pixels between two locations in the video memory. This can be used to shove a block of pixels quickly to give the impression of motion on the screen.

Pass: **ax** Source horizontal document coordinate of the area.
 bx Source vertical document coordinate of the area.
 cx Destination horizontal document coordinate.
 dx Destination vertical document coordinate.
 si Width of the area to be moved, in document units.
 di Handle of the GState used for drawing.

Pass on stack: The following arguments are pushed before the call:
 word Height of the area in document units.
 word **BLTMode** value:
 BLTM_COPY leaves source alone.
 BLTM_CLEAR clears source area.
 BLTM_MOVE clears and invalidates source.

Returns: Nothing. (Arguments popped off stack.)
Destroyed: Nothing.
Library: **graphics.def**

■ GrBrushPolyline

Brushes a connected polyline with the passed brush characteristics.

Pass: **cx** Number of points in the polyline array.
 ds:si Array of Points defining the connected lines.
 di Handle of the GState used for drawing.
 al Rectangular brush width, in pixels.
 ah Rectangular brush height, in pixels.

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**



■ GrCharMetrics

Return the metrics for a single character, given its value and the information to return.

Pass:	di	Handle of the applied GState.
	si	Value of type GCM_info indicating the type of information to return.
	ax	Chars value of character.
Returns:	CF	Set if font or driver is not available.
	dx	If GCM_..._ROUNDED is set, the rounded information. If CF set on return, dx will be zero.
	dx:ah	WBFixed value if non-rounded data requested. If CF set on return, dx and ah will both be zero.
Destroyed:	Nothing.	
Library:	font.def	

■ GrCharWidth

Return the width of a single character, given a GState and the character.

Pass:	di	Handle of the applied GState.
	ax	Chars value of character to check.
Returns:	dx:ah	WBFixed value giving the width of the character.
Destroyed:	Nothing.	
Library:	graphics.def	
Warning:	This routine does not take into account any kerning or space padding or other attributes—it simply returns the character's width.	

■ GrCheckFontAvail

Check if the named font exists and return its ID if it does.

Pass:	d1	FontEnumFlags giving the font type to match:
		FEF_FAMILY Pass FontFamily in dh .
		FEF_STRING Pass a pointer to the null-terminated font name in ds:si .
	dh	Otherwise Pass FontID of font in cx .
Returns:	cx	ID of font if it exists, FID_INVALID otherwise.
Destroyed:	Nothing.	
Library:	font.def	

Assembly Reference

Warning: If you pass FEF_STRING, the string pointed to by `ds:si` must not be null. Otherwise a fatal error will result.

■ GrClearBitmap

Clear out the contents of the bitmap associated with the given GState. The parts of the bitmap actually cleared (set to white) will depend on the bitmap's mode. For the normal mode, the data part of the bitmap is cleared while the mask is left untouched. For the BM_EDIT_MASK mode, the mask is cleared and the bitmap is left untouched. The actual value written to the mask or the bitmap will vary depending on the representation of white in the bitmap's format.

Pass: `di` Handle of the GState to have its bitmap cleared.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrCloseSubPath

Geometrically closes the currently open path. This does not end the path definition—you still must call **GrEndPath**.

Pass: `di` Handle of the GState in which the path is being defined.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrComment

Write a comment into a graphics string.

Pass: `di` Handle of the GString.
`ds:si` Address of the comment string.
`cx` Size of the comment string.
Returns: Nothing.
Destroyed: Nothing.
Library: **gstring.def**

■ GrCompactBitmap

Compact a bitmap stored in a huge array.

Pass: `bx` VM file handle of the huge array containing the bitmap.



GrCopyGString

■ 128

	ax	VM block handle of the huge array containing the bitmap.
	dx	VM file handle of the destination file for the compacted bitmap.
Returns:	dx	Preserved VM file handle.
	cx	New VM block handle of new, compacted bitmap.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrCopyGString

Copy a graphics string from one GString to another.

Pass:	si	Handle of source GString.
	di	Handle of destination GString.
	dx	GSControl flags record indicating how much of the GString should be copied.
Returns:	dx	GSRetType value indicating the type of extra data returned in cx , if any (see below).
	cx	Extra information appropriate to the type in dx :
		GSRT_COMPLETE
		Zero.
		GSRT_NEW_PAGE
		Zero.
		GSRT_FAULT
		Zero.
		GSRT_LABEL
		Label value.
		GSRT_ESCAPE
		Escape number.
		GSRT_ONE
		next opcode (in c1).
		GSRT_MISC
		next opcode (in c1).
		GSRT_XFORM
		next opcode (in c1).
		GSRT_OUTPUT
		next opcode (in c1).
		GSRT_ATTR
		next opcode (in c1).
		GSRT_PATH
		next opcode (in c1).
Destroyed:	Nothing.	
Library:	gstring.def	

■ GrCreateBitmap

Allocate memory for a bitmap and associate the memory with a window.

Pass:	al	BMTYPE record indicating the type of bitmap to create.
	bx	VM file handle of file in which to create the bitmap.
	cx	Width of the new bitmap.
	dx	Height of the new bitmap.

Assembly Reference

	di:si	The optr of the object that will receive MSG_META_EXPOSED for the bitmap (for a process or thread, pass its handle in di and a null chunk handle).
Returns:	bx ax di	Preserved VM file handle of the bitmap. VM block handle of the newly-allocated bitmap. GState handle of the bitmap in the window.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrCreateGString

Open a graphics string and begin redirecting graphics commands to the GString.

Pass:	c1	GStringType indicating the type of handle passed in bx . GST_MEMORY Memory handle in bx . GST_STREAM Stream handle in bx . GST_VMEM VM file handle in bx .
	bx	Handle of the entity which will act as the GString.
Returns:	di si	Handle of the new Graphics String. Newly-allocated chunk (for GST_MEMORY) or VM block handle (for GST_VMEM), as appropriate. For GST_STREAM, nothing is returned in si .
Destroyed:	Nothing.	
Library:	gstring.def	

■ GrCreatePalette

Create a color mapping table and associate it with the current window (the window associated with the passed GState). Initialize the table to the default values for the device's palette.

Pass:	di	GState to apply.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrCreateState

Create graphics state block containing a default GState associated with the passed window. This routine is typically used to create a GState for drawing within a MSG_META_EXPOSED handler. When you are done with the GState,



GrDeleteGStringElement

■ 130

be sure to destroy it with **GrDestroyState** so the GState does not continue to use memory and a handle.

Pass: **di** Handle of the window for which the GState will be created. Zero to create a GState without associating it with a window.

Returns: **di** Handle of new GState.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDeleteGStringElement

Delete a range of GString elements from the specified GString.

Pass: **di** GState handle; the GState contains a handle to the GString, which will be locked automatically.

cx Number of elements to delete from the GString.

Returns: Nothing.

Destroyed: Nothing.

Library: **gstring.def**

■ GrDestroyBitmap

Free the bitmap associated with the given GState and disassociate it from the window.

Pass: **di** GState handle as returned by **GrCreateBitmap**.

al **BMDestroy** value:

BMD_KILL_DATA Frees the bitmap's HugeArray.

BMD_LEAVE_DATA Leave the bitmap data and just disassociate it from the window.

Returns: Nothing.

Destroyed: **di**

Library: **graphics.def**

■ GrDestroyGString

Destroy the specified GString, either removing the GState from the data, or freeing both GState and GString data. You may ask that an additional GState be destroyed as well.

Pass: **si** Handle of Graphics String.

di Handle of "extra" GState to destroy. Zero for none.

Assembly Reference

d1 **GStringKillType** value:
 GSKT_KILL_DATA Frees the GString data along with the handle.
 GSKT_LEAVE_DATA Leaves the GString data intact but frees the handle and associated overhead.

Returns: Nothing.
Destroyed: Nothing.
Library: **gstring.def**

■ GrDestroyPalette

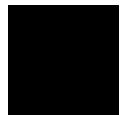
Free the current window's custom palette, if any.

Pass: **di** GState handle; the GState specifies the current window.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrDestroyState

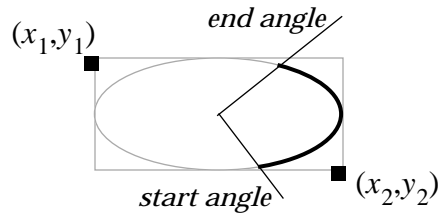
Destroy a GState block and the GState handle. Typically called in MSG_META_EXPOSED after drawing is finished. The GState is normally created with **GrCreateState**.

Pass: **di** GState handle to be destroyed.
Returns: Nothing.
Destroyed: **di**
Library: **graphics.def**



■ GrDrawArc

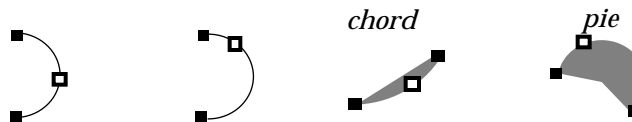
Draws an arc along the ellipse specified by a bounding box, a starting angle, and an ending angle.



Pass:	di	Handle of GState used for drawing.
	ds:si	Address of an ArcParams structure.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrDrawArc3Point

Draw a circular arc specified by three points along the arc: both endpoints and any other point on the arc.



Pass:	di	Handle of the GState used for drawing.
	ds:si	Address of a ThreePointArcParams structure.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrDrawArc3PointTo

Draw a circular arc, given two points along the arc and using the current pen position as the first endpoint. The other two points are the other endpoint and any other point on the arc.

Pass:	di	Handle of the GState used for drawing.
	ds:si	ThreePointArcToParams structure.

Returns: Nothing.
 Destroyed: Nothing.
 Library: **graphics.def**

■ GrDrawBitmap

Draw a bitmap at the coordinates passed. This routine will call a callback routine that is expected to take a pointer to a slice of bitmap and return the next slice. In most cases, the callback will be supplied by the kernel.

Pass: **di** Handle of the GState used for drawing.
ax, bx X, Y coordinates to begin drawing at.
ds:si Address of the bitmap.
dx:cx Address of the callback routine. If you are not supplying a callback, pass zero in **dx**.

Returns: Nothing.

Destroyed: Nothing.

Callback Routine Specifications:

Passed: **ds:si** Address of the bitmap slice just drawn.
 Return: **ds:si** Address of next slice to be drawn.
CF Set if bitmap is finished being drawn.
 May Destroy: Nothing.

Library: **graphics.def**

■ GrDrawBitmapAtCP

Draw a bitmap at the current pen position. This routine will call a callback routine that is expected to take a pointer to a slice of bitmap and return the next slice. In most cases, the callback will be supplied by the kernel.

Pass: **di** Handle of the GState used for drawing.
ds:si Address of the bitmap.
dx:cx Address of the callback routine. If you are not supplying a callback, pass zero in **dx**.

Returns: **ds:si** If a callback routine is supplied, **ds:si** will retain the value supplied in the last call to the callback function.

Destroyed: Nothing.

Callback Routine Specifications:

Passed: **ds:si** Address of the bitmap slice just drawn.
 Return: **ds:si** Address of next slice to be drawn.
CF Set if bitmap is finished being drawn.



GrDrawChar

■ 134

May Destroy: Nothing.

Library: **graphics.def**

■ GrDrawChar

Draw a given character at the specified position with the current text drawing state.

Pass: **ax** X position.
 bx Y position.
 dx Character to draw.
 di Handle of the GState to draw to; the text characteristics (font, color, etc.) are taken from the GState.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawCharAtCP

Draw a given character at the current pen position using the current text drawing state.

Pass: **dx** Character to draw.
 di Handle of the GState to draw to; the text characteristics (font, color, etc.) are taken from the GState.

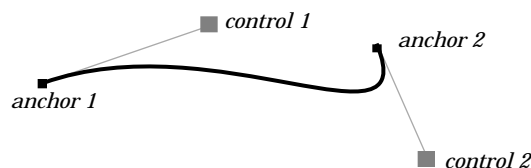
Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawCurve

Draw a bezier curve specified by four points.



Pass: **ds:si** Address of the four points making up the curve's description (all are of structure **Point**):
 Anchor point one
 Control point one

Assembly Reference

Control point two
Anchor point two
Handle of the GState used for drawing.

di

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawCurveTo

Draw a bezier curve beginning with the current pen position.

Pass: **ds:si** Address of the three additional points making up the curve's description (all are of structure **Point**):
Control point one
Control point two
Anchor point two
di Handle of the GState used for drawing.

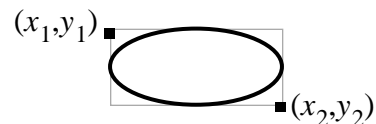
Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawEllipse

Draw a framed ellipse bounded by the passed rectangle.



Pass: **di** Handle of the GState used for drawing.
ax, bx First x, y coordinates of bounding rectangle (x_l, y_l in the diagram).
cx, dx Second x, y coordinates (x_2, y_2).

Returns: Nothing.

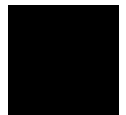
Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawGString

Draw the passed GString at the given coordinates.

Pass: **di** Handle of the GState used for drawing.



GrDrawGStringAtCP

■ 136

	si	Handle of the GString to be drawn (as returned by GrLoadString).
	ax, bx	The <i>x</i> and <i>y</i> coordinates at which to draw.
	dx	GSControl flags indicating how much of the GString should be drawn (see below).
Returns:	dx	GSRetType giving the type of information returned in cx : GSRT_COMPLETE Zero. GSRT_NEW_PAGE Zero. GSRT_FAULT Zero. GSRT_LABEL Label value. GSRT_ESCAPE Escape number. GSRT_ONE next opcode (in c1). GSRT_MISC next opcode (in c1). GSRT_XFORM next opcode (in c1). GSRT_OUTPUT next opcode (in c1). GSRT_ATTR next opcode (in c1). GSRT_PATH next opcode (in c1). cx Information returned, if any, based on value in dx .
Destroyed:	Nothing.	
Library:	gstring.def	

■ GrDrawGStringAtCP

Draw the passed GString at the current pen position.

Pass:	di	Handle of the GState used for drawing.
	si	Handle of the GString to be drawn (as returned by GrLoadString).
	dx	GSControl flags indicating how much of the GString should be drawn (see below).
Returns:	dx	GSRetType giving the type of information returned in cx : GSRT_COMPLETE Zero. GSRT_NEW_PAGE Zero. GSRT_FAULT Zero. GSRT_LABEL Label value. GSRT_ESCAPE Escape number. GSRT_ONE next opcode (in c1). GSRT_MISC next opcode (in c1). GSRT_XFORM next opcode (in c1).

Assembly Reference

GSRT_OUTPUT next opcode (in **c1**).
 GSRT_ATTR next opcode (in **c1**).
 GSRT_PATH next opcode (in **c1**).
cx Information returned, if any, based on value in **dx**.
 Destroyed: Nothing.
 Library: **gstring.def**

■ GrDrawHLine

Draw a horizontal line.
 Pass: **di** Handle of the GState used for drawing.
 ax First *x* coordinate of the line.
 bx The *y* coordinate of the line.
 cx Second *x* coordinate of the line.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **graphics.def**

■ GrDrawHLineTo

Draw a horizontal line using the current pen position as the starting point.
 Pass: **di** Handle of the GState used for drawing.
 cx Second *x* coordinate of the line.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **graphics.def**

■ GrDrawHugeBitmap

Draw a bitmap residing in a HugeArray at the coordinates passed.
 Pass: **dx** VM file handle of the huge array containing the bitmap.
 cx VM block handle of the huge array containing the bitmap.
 ax, bx *x, y* coordinates to begin drawing at.
 di Handle of the GState used for drawing.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **graphics.def**



■ GrDrawHugeBitmapAtCP

Draw a bitmap residing in a HugeArray at the current pen position.

Pass:	dx	VM file handle of the huge array containing the bitmap.
	cx	VM block handle of the huge array containing the bitmap.
	di	Handle of the GState used for drawing.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawHugeImage

Draw a bitmap at the passed position. The bitmap must reside in a HugeArray, and the image may be drawn with a square block of video pixels representing each bitmap pixel (i.e. a “magnified view”).

Pass:	di	Handle of the GState used for drawing.
	ax	<i>x</i> position at which to begin drawing (document coordinates).
	bx	<i>y</i> position at which to begin drawing (document coordinates).
	c1	ImageFlags record:
		IF_BORDER Set if a border should be drawn around the image. The border is drawn using the passed GState's current line color.
	IF_BITSIZE ImageBitSize value indicating how many pixels represent a bit in the image:	
		IBS_1 (one-to-one)
		IBS_2 (two by two pixels for each bit)
		IBS_4 (four by four pixels for each bit)
		IBS_8 (eight by eight pixels for each bit)
		IBS_16 (16 by 16 pixels for each bit)
	dx	VM file handle of the image's HugeArray.
	si	VM block handle of the beginning of the HugeArray.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawImage

Draw a bitmap at the passed position. The bitmap may be drawn with a square block of video pixels representing each bitmap pixel (i.e. a “magnified view”). If the bitmap is in a HugeArray, you should use **GrDrawHugeImage**.

Pass:	di	Handle of the GState used for drawing.
	ax	<i>x</i> position at which to begin drawing (document coordinates).
	bx	<i>y</i> position at which to begin drawing (document coordinates).
	cl	ImageFlags record:
		IF_BORDER Set if a border should be drawn around the image. The border is drawn using the passed GState's current line color.
		IF_BITSIZE ImageBitSize value indicating how many pixels represent a bit in the image:
		IBS_1 (one-to-one)
		IBS_2 (two by two pixels for each bit)
		IBS_4 (four by four pixels for each bit)
		IBS_8 (eight by eight pixels for each bit)
		IBS_16 (16 by 16 pixels for each bit)
	dx:si	Address of locked or fixed bitmap image to be drawn. This should not be a HugeArray.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrDrawLine

Draw a line, given the two endpoints.

Pass:	di	Handle of the GState used for drawing.
	ax, bx	<i>x, y</i> coordinates of first endpoint (document coordinates).
	cx, dx	<i>x, y</i> coordinates of second endpoint (document coordinates).
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrDrawLineTo

Draw a straight line from the pen position to the passed endpoint.

Pass:	di	Handle of the GState used for drawing.
	cx, dx	<i>x, y</i> coordinates of second endpoint (document coordinates).
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	



GrDrawPath

■ 140

■ GrDrawPath

Draws the passed GState's path with the current line attributes.

Pass: **di** GState containing the path and line attributes. The path will be drawn to this GState.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawPoint

Draw a single document pixel at the passed coordinates.

Pass: **di** Handle of the GState used for drawing.
ax, bx *x, y* coordinates of point.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawPointAtCP

Draws a single document pixel at the current pen position.

Pass: **di** Handle of the GState used for drawing.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawPolygon

Draws the passed connected polygon.

Pass: **di** Handle of the GState used for drawing.
ds: si Address of the array of Point structures defining the polygon.
cx Number of points in the polygon's array.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

Assembly Reference

■ GrDrawPolyline

Draws a polyline using the passed GState's line attributes. To use special "brush" attributes, call **GrBrushPolyline**.

Pass:	di	Handle of the GState used for drawing.
	ds:si	Address of an array of Point structures defining the polyline.
	cx	Number of points in the passed array.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrDrawRect

Draw a rectangle defined by the two corner points. To fill the rectangle, call **GrFillRect**.

Pass:	di	Handle of the GState used for drawing.
	ax, bx	<i>x, y</i> coordinates of first corner of the rectangle.
	cx, dx	<i>x, y</i> coordinates of opposite corner.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrDrawRectTo

Draw a rectangle defined by one corner point and the current pen position.

Pass:	di	Handle of the GState used for drawing.
	cx, dx	<i>x, y</i> coordinates of opposite corner.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrDrawRegion

Draw a region.

Pass:	di	Handle of GState used for drawing.
	ax, bx	<i>x, y</i> coordinates to begin drawing the region.
	ds:si	Address of locked or fixed region definition (RectRegion).
	cx, dx	Parameters for region, if required. These correspond to PARAM_2 and PARAM_3.



GrDrawRegionAtCP

■ 142

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**.

■ GrDrawRegionAtCP

Draw a region at the current pen position.

Pass: **di** Handle of GState used for drawing.
ds:si Address of locked or fixed region definition (**RectRegion**).
cx, dx Parameters for region, if required. These correspond to
PARAM_2 and PARAM_3.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**.

■ GrDrawRelArc3PointTo

Draw a circular arc relative to the current pen position, given two additional points: the other endpoint, and any other point on the arc. Both of the additional points are given in offsets from (relative to) the pen position.

Pass: **di** Handle of the GState used for drawing.
ax, bx *x, y* offsets from the current position to any point on the arc.
cx, dx *x, y* offsets from the current position to the other endpoint.
si **ArcCloseType** value: OPEN, CHORD, or PIE.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrDrawRelCurveTo

Draw a bezier curve using the current pen position as the first endpoint. All other points in the curve are defined by offsets from (relative to) the current pen position. See **GrDrawCurve** for a diagram of a bezier curve.

Pass: **di** Handle of the GState used for drawing.
ds:si Address of the three points making up the rest of the curve's description (all are of structure **Point**, and all are relative to the current pen position):
Control point one
Control point two
Anchor point two

Assembly Reference

Returns: Nothing.
 Destroyed: Nothing.
 Library: **graphics.def**

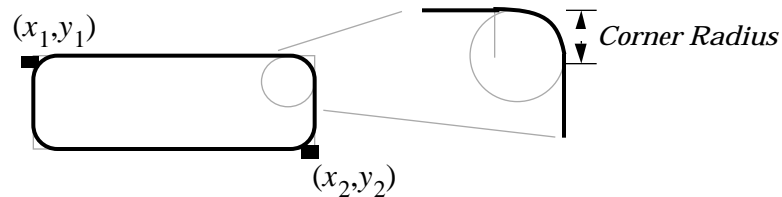
■ GrDrawRelLineTo

Draw a line from the current pen position to the point defined by the passed displacements.

Pass: **di** Handle of the GState used for drawing.
 dx.cx **WWFixed** x displacement, in document coordinates.
 bx.ax **WWFixed** y displacement, in document coordinates.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **graphics.def**

■ GrDrawRoundRect

Draw a rectangle with rounded corners.

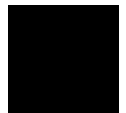


Pass: **di** Handle of the GState used for drawing.
 si Radius of rounded corners, in points.
 ax, bx x, y of first corner (x_1, y_1) in diagram).
 cx, dx x, y of opposite corner (x_2, y_2) in diagram).
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **graphics.def**

■ GrDrawRoundRectTo

Draw a rounded rectangle at the current pen position.

Pass: **di** Handle of the GState used for drawing.
 si Radius of rounded corners, in points.
 cx, dx x, y of opposite corner.



GrDrawSpline

■ 144

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrDrawSpline

Draw a collection of bezier curves defined by the passed array of Points.

Pass: **di** Handle of the GState used for drawing.
ds:si Address of the array of **Point** structures defining the spline.
(These comprise a series of bezier curves.)
cx Number of points in the array.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrDrawSplineTo

Draw a collection of bezier curves defined by the passed array of Points and having its first anchor point at the current pen position.

Pass: **di** Handle of the GState used for drawing.
ds:si Address of the array of **Point** structures defining the spline.
The first point of the spline is the current pen position.
(These comprise a series of bezier curves.)
cx Number of points in the array.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrDrawText

Draw a string at the given position with the current text drawing characteristics (defined by the GState).

Pass: **di** Handle of the GState used for drawing.
ds:si Address of text string to draw.
cx Maximum number of characters to draw from the string.
Pass zero if the string is null-terminated.
ax, bx *x, y* coordinates of the string.
Returns: Nothing.
Destroyed: Nothing.

Assembly Reference

Library: **graphics.def**

■ GrDrawTextAtCP

Draw a string at the current pen position with the current text drawing characteristics (defined by the GState).

Pass: **di** Handle of the GState used for drawing.
ds:si Address of the text string to draw.
cx Maximum number of characters to draw from the string.
 Pass zero if the string is null-terminated.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrDrawTextField

This routine draws a text field.

Pass: **di** Handle of the GState to use for drawing.
ss:bp **GDF_vars** structure.

Returns: **ss:[bp].GDFV_saved.GDFS_drawPos.PWBF_x**
 Set to end position of drawn text.

Destroyed: Nothing.

Callback Routine Specifications:

Passed:	ss:bp	ptr to GDF_vars structure on stack.
	si	offset to current position in text.
	bx:di	fptr to buffer, sizeof TextAttr structure
Return:	bx:di	Buffer is filled.
	cx	Number of characters in this run.
	ds:si	Pointer to text at offset

Library: **graphics.def**

■ GrDrawVLine

Draw a vertical line.

Pass: **di** Handle of the GState used for drawing.
ax, bx *x, y* coordinates of first endpoint.
dx *y* coordinate of second endpoint.

Returns: Nothing.

Destroyed: Nothing.



GrDrawVLineTo

■ 146

Library: **graphics.def**

■ GrDrawVLineTo

Draw a vertical line beginning at the current pen position.

Pass: **di** Handle of the GState used for drawing.
dx y coordinate of second endpoint.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrEditBitmap

Associate a previously created bitmap with a window and a GState to allow the caller to edit the bitmap.

Pass: **bx** VM file handle of file containing bitmap.
ax VM block handle of the first block in the bitmap's HugeArray.
di:si The optr of the object that will handle MSG_META_EXPOSED for the bitmap. If a process, pass the process handle in **di**.

Returns: **di** Handle of the GState to use when drawing the bitmap.

Destroyed: Nothing.

Library: **graphics.def**

■ GrEditGString

Set a specified graphics string into editing mode.

Pass: **bx** VM file handle of the file containing the GString. You may only edit GStrings of type GST_VMEM.
si VM block handle of the first block of the GString's HugeArray.

Returns: **di** Handle of the GState newly created and associated with GString.

Destroyed: Nothing.

Library: **gstring.def**

■ GrEndGString

End a definition of a GString. This is the complement to **GrBeginString**.

Pass: **di** Handle of the GString to finish.

Assembly Reference

Returns: **ax** **GStringErrorType** value:
 GSET_NO_ERROR if no error.
 GSET_DISK_FULL if the file will be truncated for disk space.

Destroyed: Nothing.

Library: **gstring.def**

■ GrEndPath

Ends the definition of the current path. This is the complement to **GrBeginPath**.

Pass: **di** GState owning the path being defined.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrEndUpdate

Unlock a window from a visual update begun with **GrBeginUpdate**.

Pass: **di** Handle of the GState originally passed to **GrBeginUpdate**.

Returns: Nothing.

Destroyed: Nothing.

Library: **win.def**

■ GrEnumFonts

Generate a list of font names of the available fonts. Return the number of matching fonts; the names are returned in a passed buffer.

Pass: **cx** Number of **FontEnumStruct** structures the buffer can hold.
es:di Address of the locked or fixed buffer for returned values. If **cx** is passed zero, no buffer will be used.

d1 **FontEnumFlags** record indicating the type of fonts to find:

- FEF_ALPHABETIZE Alphabetize the returned list.
- FEF_USEFUL Find “useful” fonts only.
- FEF_FIXED_WIDTH Find fixed-width fonts only.
- FEF_FAMILY Match the **FontFamily** in **dh**.
- FEF_STRING Match the font name string.
- FEF_DOWNCASE Downcase all returned strings.
- FEF_BITMAPS Find fonts with bitmap representations.



GrEscape

■ 148

		FEF_OUTLINES	Find fonts with outline representations.
	dh	FontFamily	value; used only if FEF_FAMILY set in d1 .
Returns:	cx		Number of matching fonts found.
	es:di		Buffer filled with one FontEnumStruct for each matching font returned.
Destroyed:	Nothing.		
Library:	font.def		

■ GrEscape

Write an “escape” element to a graphics string. This call is meaningful only between calls to **GrBeginGString** and **GrEndGString**. The element, in full, will have the following structure:

		<u>Byte</u>	<u>Contents</u>	<u>Description</u>
		0	0xFF	GString escape indicator
		1, 2	ax	Application's escape code
		2, 3	cx	Size of escape data
		4–n	ds:si	Data from passed buffer.
Pass:	di	Handle of GString to write to.		
	ax	Escape code of element (defined by caller).		
	cx	Size of element to write.		
	ds:si	Address of locked or fixed data to write to the escape element.		
Returns:	Nothing.			
Destroyed:	Nothing.			
Library:	gstring.def			

■ GrFillArc

Draws an arc as in **GrDrawArc** and fills it like a pie wedge. The arc is defined by an ellipse in a bounding box and two angles that intersect the ellipse (see **GrDrawArc**).

Pass:	di	Handle of the GState used for drawing.
	ds:si	Address of an ArcParams structure:
		<i>AP_close</i> Unused in this context.
		<i>AP_left, AP_top, AP_right, AP_bottom</i>
		Bounds of the ellipse's bounding box (signed words).
		<i>AP_angle1</i> Start angle of arc segment (signed word).
		<i>AP_angle2</i> End angle of arc segment (signed word).
Returns:	Nothing.	

Assembly Reference

Destroyed: Nothing.

Library: **graphics.def**

■ GrFillArc3Point

Fill a circular arc defined by three points: two endpoints and any other point along the arc. The filled portion will be the pie wedge defined by the arc.

Pass: **di** Handle of the GState used for drawing.
ds:si Address of a **ThreePointArcParams** structure.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrFillArc3PointTo

Fill a circular arc defined by the current pen position (as the first endpoint) and two other points: the opposite endpoint and any other point along the arc. The filled portion will be the pie wedge defined by the arc.

Pass: **di** Handle of the GState used for drawing.
ds:si **ThreePointArcToParams** structure.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrFillBitmap

Draw the given bitmap as if it were a mask, filling it with the current area color.

Pass: **di** Handle of the GState used for drawing.
ax, bx *x, y* coordinates at which to draw the bitmap.
ds:si Address of the bitmap to be drawn.
dx:cx Address of the callback routine. If you are not supplying a callback, pass zero in **dx**. It is unusual to use your own callback routine.

Returns: Nothing.

Destroyed: Nothing.

Callback Routine Specifications:

Passed: **ds:si** Address of the bitmap slice just drawn.
Return: **ds:si** Address of next slice to be drawn.



GrFillBitmapAtCP

■ 150

May Destroy: **Nothing.**
Library: **graphics.def**
CF Set if bitmap is finished being drawn.

■ GrFillBitmapAtCP

Draw the given bitmap as if it were a mask, filling it with the current area color. The bitmap will be drawn at the current pen position.

Pass: **di** Handle of the GState used for drawing.
ds:si Address of the bitmap to be drawn.
dx:cx Address of the callback routine. If you are not supplying a callback, pass zero in **dx**. It is unusual to use your own callback routine.

Returns: **Nothing.**

Destroyed: **Nothing.**

Callback Routine Specifications:

Passed: **ds:si** Address of the bitmap slice just drawn.
Return: **ds:si** Address of next slice to be drawn.
CF Set if bitmap is finished being drawn.

May Destroy: **Nothing.**

Library: **graphics.def**

■ GrFillEllipse

Draws a filled ellipse bounded by the passed rectangle.

Pass: **di** Handle of the GState used for drawing.
ax, bx *x, y* coordinates of first corner of bounding rectangle.
cx, dx *x, y* coordinates of opposite corner of bounding rectangle.

Returns: **Nothing.**

Destroyed: **Nothing.**

Library: **graphics.def**

■ GrFillHugeBitmap

Treat a monochrome bitmap as a mask, filling it with the current area color. The bitmap should be stored in a huge array.

Pass: **di** GStateHandle
ax, bx *x, y* coordinate to draw at.
dx VM file handle (or zero if *TPD_file* is set)

Assembly Reference

cx VM block handle
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrFillHugeBitmapAtCP

Treat a monochrome bitmap as a mask, filling it with the current area color. The bitmap should be stored in a huge array.

Pass: **di** GStateHandle
ax,bx *x,y* coordinate to draw at.
dx VM file handle (or zero if *TPD_file* is set)
cx VM block handle
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrFillPath

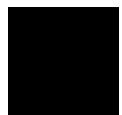
Draws a filled representation of the passed GState's current path, using the supplied fill rule and current GState area attributes.

Pass: **di** Handle of the GState used for drawing.
c1 **RegionFillRule** value: RFR_ODD_EVEN or RFR_WINDING.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrFillPolygon

Draw a filled polygon.

Pass: **di** Handle of the GState used for drawing.
ds:si Array of **Point** structures defining the polygon.
cx Number of points in the array.
al **RegionFillRule** value: RFR_ODD_EVEN or RFR_WINDING.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**



GrFillRect

■ 152

■ GrFillRect

Fill the rectangle defined by the two passed points.

Pass:	di	Handle of the GState used for drawing.
	ax, bx	X, Y coordinates of the first corner.
	cx, dx	X, Y coordinates of the opposite corner.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrFillRectTo

Fill the rectangle defined by the current pen position and the passed point.

Pass:	di	Handle of the GState used for drawing.
	cx, dx	X, Y coordinates of the opposite corner.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrFillRoundRect

Fill the rounded rectangle defined by the parameters.

Pass:	di	Handle of the GState used for drawing.
	si	Radius of the corner roundings, in points.
	ax, bx	X, Y coordinates of the first corner of the rectangle.
	cx, dx	X, Y coordinates of the opposite corner of the rectangle.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrFillRoundRectTo

Fill the rounded rectangle defined by the current pen position and the parameters.

Pass:	di	Handle of the GState used for drawing.
	si	Radius of the corner roundings, in points.
	cx, dx	<i>x, y</i> coordinates of the opposite corner of the rectangle.

Returns: Nothing.

Destroyed: Nothing.

Assembly Reference

Library: **graphics.def**

■ GrFontMetrics

Return metrics information about the given GState's current font.

Pass:	di si	Handle of the GState whose font is to be checked. GFM_info value indicating the type of return value requested.
Returns:	dx.ah dx	WBFixed value giving the result requested. If GFM_ROUNDED is passed in si , dx will contain the entire rounded value (ah is ignored).
Destroyed:	Nothing.	
Library:	font.def	

■ GrGetAreaColor

Return the area color set for the given GState.

Pass:	di	Handle of the GState.
Returns:	a1 b1 bh	Red value of color. Green value of color. Blue value of color.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetAreaColorMap

Return the area color map of the given GState.

Pass:	di	Handle of the GState.
Returns:	a1	ColorMapMode record indicating the color mapping mode effective in the GState.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetAreaMask

Return the area mask type of the passed GState.

Pass:	di a1	Handle of the GState. GetMaskType value GMT_ENUM Returns SysDrawMask record. GMT_BUFFER Returns entire mask buffer and its size.
-------	------------------------	---



GrGetAreaPattern

■ 154

	ds:si	Address of a locked or fixed buffer to receive the mask buffer, if GMT_BUFFER passed in a1 . Otherwise ignored.
Returns:	a1 ds:si	SysDrawMask record. Address of the returned mask if GMT_BUFFER passed in a1 .
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetAreaPattern

Return the fill pattern for the passed GState.

Pass:	di	Handle of the GState.
Returns:	a1 ah bx cx	PatternType value. SystemHatch or SystemBitmap value, depending on a1 . If neither is applicable, ah returned destroyed. Handle of HatchPattern or Bitmap value, if applicable. Size of HatchPattern or Bitmap in bx , if applicable.
Destroyed:	Depending on return values, ah .	
Library:	graphics.def	

■ GrGetBitmap

Copy a bitmap from the screen to a memory block.

Pass:	di ax, bx cx dx	Handle of the GState owning the bitmap. <i>x, y</i> coordinates of upper-left of bitmap area to be copied. Bitmap width (document coordinates). Bitmap height (document coordinates).
Returns:	bx cx dx	Handle of a newly-allocated memory block containing the bitmap. NullHandle (0) if memory allocation error. Width of bitmap copied (pixels). Height of bitmap copied (pixels).
Destroyed:	Nothing.	
Library:	graphics.def	
Warning:	This routine does <i>not</i> check for clipping of the rectangle; it copies directly from the screen to memory. It is therefore useful for screen dumps but not necessarily useful for applications.	

■ GrGetBitmapMode

Return the mode information for the editable bitmap owned by the passed GState.

Assembly Reference

Pass:	di	Handle of the GState, as returned by GrCreateBitmap or GrEditBitmap .
Returns:	CF ax	Set if GState not pointing at a bitmap; clear otherwise. BitmapMode record with the following flags: BM_EDIT_MASK Set if editing mask rather than bitmap. BM_CLUSTERED_DITHER Set if bitmap uses a clustered rather than dispersed dither (only for BMF_MONO bitmaps).
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetBitmapRes

Return the resolution of the bitmap owned by the passed GState.

Pass:	di	Handle of the GState.
Returns:	ax bx	X resolution (horizontal), in dots per inch. Y resolution (vertical), in dots per inch.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetBitmapSize

Return the size of the bitmap owned by the passed GState.

Pass:	ds:si	Handle of the GState.
Returns:	ax bx	<i>x</i> size in points (width). <i>y</i> size in points (height).
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetClipRegion

Return the region corresponding to the clip paths of the passed GState. The region is in device coordinates; the first four words are its bounds.

Pass:	di c1	Handle of the GState. RegionFillRule : RFR_ODD_EVEN or RFR_WINDING.
Returns:	CF bx	Set if error, clear otherwise. Handle of block containing Region structure if successful. Null handle if CF returned set (error).



GrGetCurPos

■ 156

Destroyed: Nothing.
Library: **graphics.def**

■ GrGetCurPos

Return the current drawing position for the passed GState.

Pass: **di** Handle of the GState.
Returns: **ax** Current *x* drawing position.
bx Current *y* drawing position.
Destroyed: Nothing.
Library: **graphics.def**

■ GrGetCurPosWWFixed

Return the current drawing position for the passed GState. The answer returned is rather precise, with one word of fraction information for both the *x* and *y* coordinates.

Pass: **di** Handle of the GState.
dx.cx **WWFixed** *x* coordinate.
bx.ax **WWFixed** *y* coordinate.
Returns:
Destroyed: Nothing.
Library: **graphics.def**

■ GrGetDefFontID

Return the default fond ID and point size as defined in the GEOS.INI file. Also return the font data block.

Pass: Nothing.
Returns: **cx** Default **FontID**.
dx.ah Default point size (**WBFixed**).
bx Handle to data block for default font.
Destroyed: Nothing.
Library: **graphics.def**

■ GrGetExclusive

Check to see if any GState has exclusive drawing rights to the screen.

Pass: **bx** Handle of the current video driver, or zero for the default.

Assembly Reference

Returns: **bx** Handle of the GState which currently has exclusive drawing rights. A null handle (zero) will be returned if no GState has the exclusive.
Destroyed: Nothing.
Library: **graphics.def**

■ GrGetFont

Return the current font's font ID and point size as set in the passed GState.

Pass: **di** Handle of the GState containing the font information.
Returns: **cx** **FontID** of the current font.
dx:ah **WWFixed** value indicating the point size.
Destroyed: Nothing.
Library: **graphics.def**

■ GrGetFontName

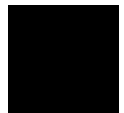
Return the name string of a specified font.

Pass: **cx** **FontID** of font.
ds:si Address of locked or fixed buffer into which the name will be copied. Must be at least FONT_NAME_LEN in size.
Returns: **CF** Set if font found; clear if no match found.
cx If **CF** set, the size of the data in the **ds:si** buffer (not including terminating null character).
If **CF** clear, will be returned zero.
ds:si Address of buffer. If successful, null-terminated file name will be in the buffer.
Destroyed: Nothing.
Library: **font.def**

■ GrGetFontWeight

Return the weight of the current font as set in the passed GState.

Pass: **di** Handle of the GState.
Returns: **al** **FontWeight** value indicating percentage of normal weight.
Destroyed: Nothing.
Library: **graphics.def**



GrGetFontWidth

■ 158

■ GrGetFontWidth

Return the width of the current font as set in the passed GState.

Pass:	di	Handle of the GState.
Returns:	al	FontWidth value indicating percentage of normal weight.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetGStringBounds

Return the coordinate bounds of a graphics string drawn at the current pen position.

Pass:	di	Handle of the GState used for drawing.
	si	Handle of the graphics string.
	dx	GSControl flags record indicating how many graphics string elements to draw.
Returns:	CF	Set on overflow error, clear otherwise. If set, no other return values are valid.
	ax	Left bound of smallest rectangle enclosing the string.
	bx	Top bound.
	cx	Right bound.
	dx	Bottom bound.
Destroyed:	Nothing.	
Library:	gstring.def	

■ GrGetGStringBoundsDWord

Get coordinate bounds of a graphics string.

Pass:	di	Graphics state handle, or zero for no graphics state.
	si	Graphics string handle.
	dx	Enum of type GSControl .
	ds:bx	Far pointer to buffer the size of RectDWord .
Returns:	ds:bx	RectDWord structure filled in with the bounds.
Destroyed:	Nothing.	
Library:	gstring.def	

■ GrGetGStringElement

Extract and return an element from a graphics string.

Pass:	di	Handle of the GState used for drawing.
-------	-----------	--

Assembly Reference

	si	Handle of the graphics string.
	ds:bx	Address of a locked or fixed buffer to hold return data.
	cx	Maximum allowable size of the return data.
Returns:	al	Opcode of the graphics string element.
	cx	Actual size of data returned in the buffer.
	ds:bx	Address of the buffer passed.
Destroyed:	Nothing.	
Library:	gstring.def	

■ GrGetGStringHandle

Return the handle of the graphics string associated with the passed GState.

Pass:	di	Handle of the GState.
Returns:	ax	Handle of the graphics string, or a null handle if none.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetHugeBitmapSize

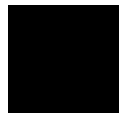
Return the size in points of the bitmap.

Pass:	bx:di	HugeArray vm file/vm block handle
Returns:	ax, bx	<i>x, y</i> size in points.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetInfo

Return information about the GState as requested.

Pass:	di	Handle of the GState to query.
	ax	GrInfoType value indicating return type desired: GIT_PRIVATE_DATA Returns the private data of the GState. GIT_WINDOW Returns the window handle of the GState.
Returns:	ax	If GIT_WINDOW was passed in ax , ax will contain the window handle of the GState. Otherwise:
	ax, bx, cx, dx	If GIT_PRIVATE_DATA was passed in ax , these registers will contain the GState's private data. This is opaque and of limited utility to applications.
Destroyed:	Nothing.	



GrGetLineColor

■ 160

Library: **graphics.def**

■ GrGetLineColor

Return the line drawing color set for the passed GState.

Pass: **di** Handle of the GState.

Returns: **a1** Red value of color.
b1 Green value of color.
bh Blue value of color.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetLineColorMap

Return the line color mapping information for the specified GState.

Pass: **di** Handle of the GState.

Returns: **a1** **ColorMapMode** record indicating the line color mapping mode effective in the GState.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetLineEnd

Return the line end type set for the passed GState.

Pass: **di** Handle of the GState.

Returns: **a1** **LineEnd** value indicating line end type.
LE_BUTTCAP, LE_ROUND CAP, LE_SQUARECAP.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetLineJoin

Return the line join type set for the passed GState.

Pass: **di** Handle of the GState.

Returns: **a1** **LineJoin** value indicating the line join type.
LJ_MITERED, LJ_ROUND, LJ_BEVELED.

Destroyed: Nothing.

Library: **graphics.def**

Assembly Reference

■ GrGetLineMask

Return information about the line mask set for the passed GState.

Pass:	di	Handle of the GState.
	a1	GetMaskType value: GMT_ENUM Returns only a SysDrawMask record. GMT_BUFFER Returns the entire mask in the passed buffer.
	ds:si	Address of a locked or fixed buffer if passing GMT_BUFFER in a1 . This buffer must be at least (size DrawMask) bytes big.
Returns:	a1	SysDrawMask record.
	ds:si	Buffer filled with draw mask, if GMT_BUFFER passed in a1 .
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetLineStyle

Return the line style type (dash pattern) set for the passed GState.

Pass:	di	Handle of the GState.
Returns:	a1	LineStyle value indicating the current line style. LS_SOLID, LS_DASHED, LS_DOTTED, LS_DASHDOT, LSDASHDDOT, LS_CUSTOM.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetLineWidth

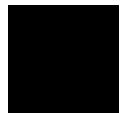
Return the line width set for the passed GState.

Pass:	di	Handle of the GState.
Returns:	dx:ax	WWFixed value indicating the line width, in points.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetMaskBounds

Return the 16-bit bounds of the current clipping rectangle for the passed GState.

Pass:	di	Handle of the GState.
Returns:	CF	Set if the mask is NULL or if the current GState transformation can not be expressed in 16-bit values.



GrGetMaskBoundsDWord

■ 162

	ax	Left bound of the current clip rectangle.
	bx	Top bound.
	cx	Right bound.
	dx	Bottom bound.
	di	Handle of the GState, preserved.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetMaskBoundsDWord

Return the 32-bit bounds of the current clipping rectangle for the passed GState.

Pass:	di	Handle of the GState.
	ds:si	Address of locked or fixed RectDWord for returned bounds.
Returns:	CF	Set if mask is NULL, clear otherwise.
	ds:si	Address of the returned RectDWord containing the clipping rectangle's bounds.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetMiterLimit

Return the miter limit set for the passed GState. The miter limit is the smallest angle that can be drawn with a miter join; smaller angles must be drawn with a beveled join.

Pass:	di	Handle of the GState.
Returns:	bx:ax	WWFixed structure defining the miter limit.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetMixMode

Return the mix mode set for the passed GState.

Pass:	di	Handle of the GState.
Returns:	al	MixMode value: MM_CLEAR, MM_COPY, MM_NOP, MM_AND, MM_INVERT, MM_XOR, MM_SET, MM_OR.

Destroyed: Nothing.

Library: **graphics.def**

Assembly Reference

■ GrGetPalette

Return the palette definition set for the passed GState.

Pass:	di	Handle of the GState.
	al	GetPalType value: GPT_ACTIVE Return the active palette. GPT_DEFAULT Return the default palette.
Returns:	bx	Memory handle to a newly allocated block containing the palette's definition.
Destroyed:	dx.	
Library:	graphics.def	

■ GrGetPath

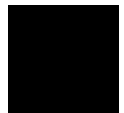
Return the GString data defining the current path of the passed GState.

Pass:	di	Handle of the GState.
	bx	GetPathType value. One of the following: GPT_CURRENT current path GPT_CLIP current clip path GPT_WIN_CLIP win clip path
Returns:	CF bx	Set if no path or if error allocating memory for path. If CF set: Null handle. If CF clear: Memory handle of newly allocated block containing the path GString.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetPathBounds

Return the smallest rectangle that can encompass the path set in the passed GState, as it would be filled.

Pass:	di	Handle of the GState.
	c1	RegionFillRule value: RFR_ODD_EVEN, RFR_WINDING.
Returns:	CF	Set if no path or if bounds can not be expressed in 16-bit format.
	ax	Left bound of rectangle.
	bx	Top bound.
	cx	Right bound.
	dx	Bottom bound.



GrGetPathBoundsDWord

■ 164

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetPathBoundsDWord

Returns the rectangular bounds that encompass the current path (as it would be filled)

Pass: **di** Handle of GState maintaining path.
ax GetPathType value.
ds:bx Far pointer to buffer large enough to hold a **RectDWord** structure.

Returns: **CF** Clear on success, set on failure.
ds:bx RectDWord structure filled with bounds.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetPathRegion

Return the region defined by the path in the passed GState. The region is expressed in terms of device coordinates; the first four words of the returned region are its bounds.

Pass: **di** Handle of the GState.
c1 **RegionFillRule** value: RFR_ODD_EVEN, RFR_WINDING.

Returns: **CF** Set if null path or other error, clear otherwise.
bx If CF set, NULL region.
If CF clear, global handle of a newly-allocated block containing the **Region**.

Destroyed: Nothing.

Library: **graphics.def**

Warning: The returned bounds are in device coordinates, not document coordinates.

■ GrGetPoint

Return the color of a single document pixel.

Pass: **di** Handle of the GState used for drawing.
ax, bx *x, y* coordinates of the point in document units.

Returns: **ah** Raw pixel value, except on 24-bit devices.
a1 Red component of the point's color.
b1 Green component of the point's color.
bh Blue component of the point's color.

Assembly Reference

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetSubscriptAttr

Return the subscript attributes used by the passed GState.

Pass: **di** Handle of the GState.

Returns: **al** **SubscriptPosition** value (down from top as percentage of normal font size).
ah **SubscriptSize** value (percentage of normal font size).

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetSuperscriptAttr

Return the superscript attributes used by the passed GState.

Pass: **di** Handle of the GState.

Returns: **al** **SuperscriptPosition** value (up from bottom as percentage of normal font size).
ah **SuperscriptSize** value (percentage of normal font size).

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetTextBounds

Return the bounds of the smallest rectangle that can enclose the given text string using the passed GState's text attributes.

Pass: **di** Handle of the GState.
ax, bx X, Y coordinates at which text would be drawn.
ds:si Address of the text string.
cx Maximum number of characters in the string to check; if zero, the string is assumed to be null-terminated.

Returns: **cf** Set if font driver not available, clear otherwise.
ax Left bound of rectangle.
bx Top bound.
cx Right bound.
dx Bottom bound.

Destroyed: Nothing.

Library: **graphics.def**



GrGetTextColor

■ 166

■ GrGetTextColor

Return the text color set in the passed GState.

Pass:	di	Handle of the GState.
Returns:	a1	Red component of text color.
	bh	Green component of color.
	b1	Blue component of color.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetTextColorMap

Return the color map used by the given GState when drawing text.

Pass:	di	Handle of the GState.
Returns:	a1	ColorMapMode record indicating the text color mapping mode effective in the GState.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetTextDrawOffset

Return the number of characters to be drawn at the end of a string.

Pass:	di	Handle of the GState.
Returns:	ax	Number of characters; word value stored in GState's <i>GS_textDrawOffset</i> field.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetTextMask

Return information about the given GState's text drawing mask.

Pass:	di	Handle of the GState.
	a1	GetMaskType value:
		GMT_ENUM Returns SysDrawMask record only.
		GMT_BUFFER Returns entire mask in the passed buffer.
	ds:si	Address of locked or fixed buffer to receive mask data if GMT_BUFFER passed in a1 . Buffer must be at least large enough to accommodate a DrawMask structure.
Returns:	a1	SysDrawMask record.

Assembly Reference

ds:si If GMT_BUFFER passed in **a1**, the **DrawMask** structure of the text mask will be returned in the buffer.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetTextMode

Return the text mode used by the passed GState.

Pass: **di** Handle of the GState.

Returns: **a1** **TextMode** record indicating the flags set for the GState.

Destroyed: Nothing.

Library: **graphics.def**

■ GrGetTextPattern

Return the text fill pattern set for the passed GState.

Pass: **di** Handle of the GState.

Returns: **a1** **PatternType** value indicating the type of pattern returned.
ah **SystemHatch** or **SystemBitmap** pattern value, or destroyed depending on the value in **a1**.

bx Handle of a block containing either **HatchPattern** or **Bitmap** structure, depending on the value in **a1**. This return value is optional; if not returned, it will be a null handle.

cx Size of the structure referenced by the handle in **bx**. If not returned, will be zero.

Destroyed: Nothing. Possibly **ah**; see return values.

Library: **graphics.def**

■ GrGetTextSpacePad

Return the given GState's space padding setting. This determines the amount to pad spaces when drawing text.

Pass: **di** Handle of the GState.

Returns: **dx** Pixel spacing, in document coordinates.

b1 Additional fractional spacing, in document coordinates.

Destroyed: Nothing.

Library: **graphics.def**



GrGetTextStyle

■ 168

■ GrGetTextStyle

Return the text style set in the passed GState.

Pass: **di** Handle of the GState.
Returns: **al** **TextStyle** record indicating the text style.
Destroyed: Nothing.
Library: **graphics.def**

■ GrGetTrackKern

Return the degree of track kerning used by the passed GState.

Pass: **di** Handle of the GState.
Returns: **ax** Degree of track kerning (signed word).
Destroyed: Nothing.
Library: **graphics.def**

■ GrGetTransform

Return the transformation matrix set for the passed GState.

Pass: **di** Handle of the GState.
ds:si Address of locked or fixed buffer into which six **WWFixed** values will be written.
Returns: **ds:si** Buffer filled with the six transformation matrix entries as shown in the diagram below:

$$\begin{bmatrix} e11 & e12 & 0 \\ e21 & e22 & 0 \\ e31 & e32 & 1 \end{bmatrix} \Rightarrow e11, e12, e21, e22, e31, e32$$

Transformation Matrix *Array Order*

The above array may be altered directly and then used with **GrApplyTransform**.

Destroyed: Nothing.
Library: **graphics.def**

■ GrGetWinBounds

Return the bounds of the current window's region, in document coordinates. The window is specified by the passed GState.

Assembly Reference

Pass:	di	Handle of the GState.
Returns:	CF	Set if coordinates can not be expressed in 16-bit values, clear otherwise.
	di	Handle of the GState, preserved.
	ax	Left bound of the window's regions.
	bx	Top bound.
	cx	Right bound.
	dx	Bottom bound.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetWinBoundsDWord

Return the bounds of the current window's region, in document coordinates. These coordinates are 32-bit values for extended transformations. If one of the window's transformation matrixes contains a rotation, the returned bounds will be large enough to contain the entire rotated rectangle.

Pass:	di	Handle of the GState.
	ds:si	Address of a fixed or locked buffer containing a RectDWord structure to hold the return values.
Returns:	ds:si	Address of the returned RectDWord structure containing the window's bounds in document coordinates.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGetWinHandle

Return the window handle of the window associated with the passed GState.

Pass:	di	Handle of the GState.
Returns:	ax	Window handle.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrGrabExclusive

Grab the video driver's exclusive access in order to begin drawing directly to the video driver. It is very rare that applications will ever need this.

Pass:	bx	Handle of the video driver to grab.
	di	Handle of the GState to use for drawing.



GrInitDefaultTransform

■ 170

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrInitDefaultTransform

Replace the default transformation matrix with the passed GState's current transformation matrix. This routine should be used only with great care; it will be called by applications only in the absolute rarest of situations.

Pass: **di** Handle of the GState.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**
Warning: This routine should almost never be used by applications.

■ GrInvalRect

Invalidate a portion of the window associated with the passed GState. The rectangle passed will be added to the window's update region. If the rectangle is rotated, the region will be built out before being added to the window's invalid region.

Pass: **di** Handle of the GState.
ax, bx *x, y* coordinates of one corner of invalid rectangle.
cx, dx *x, y* coordinates of opposite corner.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrInvalRectDWord

Invalidate a portion of the window associated with the passed GState. This routine is like GrInvalRect except that it uses 32-bit coordinates rather than 16-bit coordinates.

Pass: **di** Handle of the GState.
ds:si Address of a **RectDWord** structure containing the bounds of the rectangle to be invalidated.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

Assembly Reference

■ GrLabel

Write a label element to a graphics string. This routine writes out three bytes to the graphics string; the first byte is a GR_LABEL opcode, and the second and third bytes are the label passed in ax.

Pass: **di** Handle of the graphics string.
 ax Label value to write (determined by the caller).

Returns: Nothing.

Destroyed: Nothing.

Library: **gstring.def**

■ GrLoadGString

Load a graphics string from a file, a stream, or a locked block. Allocate new memory for the loaded graphics stream, returning the handle of the new block. (This routine does not actually copy the entire string into the new memory; it allocates a GString structure and handle that can then be used with other GString operations.)

Pass: **c1** GStringType value indicating handle type passed in bx:
 GST_STREAM Stream handle passed.
 GST_VMEM VM file handle passed.
 GST_PTR Segment address passed.

bx Handle or segment address of the source of the graphics string.

si If **c1** = GST_VMEM, VM block handle of the GString.
 If **c1** = GST_PTR, offset into the block to start of GString.

Returns: **si** Handle of the new GString.

Destroyed: Nothing.

Library: **gstring.def**

■ GrMapColorIndex

Map a color index to its RGB equivalent.

Pass: **di** Handle of the GState containing the palette to use.
 Pass zero to use default mapping.

ah Color index to map.

Returns: **a1** Red component.
 b1 Green component.
 bh Blue component.

Destroyed: Nothing.



GrMapColorRGB

■ 172

Library: **graphics.def**

■ GrMapColorRGB

Map an RGB color to its palette index.

Pass:	di	Handle of the GState containing the palette to use.
	a1	Pass zero to use default mapping.
	b1	Red component.
	bh	Green component.
	bh	Blue component.
Returns:	ah	Color index the RGB values map to, closest fit.
	a1	Red component of closest fit.
	b1	Green component of closest fit.
	bh	Blue component of closest fit.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrMoveReg

Move a region by a given amount both horizontally and vertically.

Pass:	ds:si	Address of the region's definition.
	cx	<i>x</i> amount to shift (signed).
	dx	<i>y</i> amount to shift (signed).
Returns:	ds:si	Address of the byte just past the end of the region definition.
Destroyed:	ax	
Library:	graphics.def	

■ GrMoveTo

Set the current pen position for the passed GState.

Pass:	di	Handle of the GState.
	ax	New absolute <i>x</i> position of the pen.
	bx	New absolute <i>y</i> position of the pen.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

Assembly Reference

■ GrMulDWFxed

Multiply two 48-bit signed numbers, where each integer is a **DWFixed** structure.

Pass:	di:dx	Integral portion of multiplier.
	cx	Fractional portion of multiplier.
	si:bx	Integral portion of multiplicand.
	ax	Fractional portion of multiplicand.
Returns:	CF	Set on overflow, clear otherwise.
	dx:cx	Integral portion of returned value.
	bx	Fractional portion of returned value.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrMulDWFxedPtr

Multiply two 48-bit signed numbers, where each integer is a **DWFixed** structure. The parameters are passed in two buffers, unlike **GrMulDWFxed**.

Pass:	ds:si	Address of the multiplicand, of type DWFixed .
	es:di	Address of the multiplier, of type DWFixed .
Returns:	CF	Set on overflow, clear otherwise.
	dx:cx	Integral portion of returned value.
	bx	Fractional portion of returned value.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrMulWWFixed

Multiply two 32-bit signed numbers of type **WWFixed**.

Pass:	dx:cx	WWFixed value of multiplier (low 16 bits is fraction).
	bx:ax	WWFixed value of multiplicand (low 16 bits is fraction).
Returns:	dx:cx	WWFixed result of multiplication (low 16 bits is fraction).
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrMulWWFixedPtr

Multiply two 32-bit signed numbers of type **WWFixed**, passed in two buffers.

GrNewPage

■ 174

Pass:	ds:si	Address of WWFixed value of multiplier.
	es:di	Address of WWFixed value of multiplicand.
Returns:	dx:cx	WWFixed result of multiplication (low 16 bits is fraction).
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrNewPage

Execute a form feed for the GState or GString passed. When drawing to a path, this routine is ignored. When writing to a graphics string, it stores a GR_NEW_PAGE code. Otherwise, it invalidates the entire window associated with the given GState.

Pass:	di	Handle of the GState or graphics string to draw to.
	al	PageEndCommand value.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrNullOp

Write a no-operation code to a graphics string.

Pass:	di	Handle of the graphics string to write to.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrParseGString

Parse a graphics string by invoking a callback routine on each desired element.

Pass:	di	Handle of the GState used for drawing.
	si	Handle of the GString to draw.
	dx	GSControl flags indicating which elements are desired.
	bx:cx	Address of a callback routine to be called on each element.
	bp	Data to be passed to the callback routine.
Returns:	Nothing.	
Destroyed:	Nothing.	
Callback Routine Specifications:		

Assembly Reference

Passed:	di bx ds:si	Handle of the GState. Data passed in bp to GrParseGString . Address of the element to be drawn.
Return:	ax	<i>True</i> (i.e., non-zero) if finished drawing. <i>False</i> (i.e., zero) to continue parsing the string.
May Destroy:	Any. May <i>not</i> write into the block pointed to by ds.	
Library:	gstring.def	

■ GrPolarToCartesian

Convert a polar coordinate to its corresponding cartesian equivalents.

Pass:	dx.cx bx.ax	WWFixed value of angle (Θ). This is assumed to be relative to the x axis, increasing counterclockwise. WWFixed value of distance (r).
Returns:	dx.cx bx.ax	WWFixed value of x coordinate, in document coordinates. WWFixed value of y coordinate, in document coordinates.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrQuickArcSine

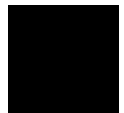
Calculate an inverse sine. This routine returns the largest integral angle with a sine less than the passed value.

Pass:	bx dx.cx	Original delta x value (only the sign matters). WWFixed value of the sine to be inversed.
Returns:	dx.cx	WWFixed value of the angle.
Destroyed:	ax , bx	
Library:	graphics.def	

■ GrQuickCosine

Calculate the cosine of an angle.

Pass:	dx.ax	32-bit integer (dx is high word, ax is low word) representing 65536 times the angle ($\Theta * 65536$).
Returns:	dx.ax	32-bit number representing 65536 times the cosine of the angle ($65536 * \cos(\Theta)$).
Destroyed:	Nothing.	
Library:	graphics.def	



■ GrQuickSine

Calculate the sine of an angle.

Pass: **dx.ax** 32-bit integer (dx is high word, ax is low word) representing 65536 times the angle ($\Theta * 65536$).

Returns: **dx.ax** 32-bit number representing 65536 times the sine of the angle ($65536 * \sin(\Theta)$).

Destroyed: Nothing.

Library: **graphics.def**

■ GrQuickTangent

Calculate the tangent of the passed angle.

Pass: **dx.ax** **WWFixed** value of the angle.

Returns: **dx.ax** **WWFixed** value of the tangent.

Destroyed: Nothing.

Library: **graphics.def**

■ GrReleaseExclusive

Release the exclusive grab made on a video driver with GrGrabExclusive.

Pass: **bx** Handle of the video driver, or zero for the default driver.

di Handle of the GState originally passed to GrGrabExclusive.

Returns: **ax, bx, cx, dx** Bounds of invalidation area required, in device coordinates.

Destroyed: Nothing.

Library: **graphics.def**

■ GrRelMoveTo

Set the current pen position of the passed GState. Sets the new position relative to the current position; to set the position absolutely, use **GrMoveTo**.

Pass: **di** Handle of the GState.

dx.cx **WWFixed** horizontal x displacement.

bx.ax **WWFixed** vertical y displacement.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

Assembly Reference

■ GrRestoreState

Restore the current GState from a GState previously saved with GrSaveState.

Pass: `di` Handle of the GState to be restored.
Returns: `di` Preserved handle of the updated GState.
Destroyed: Nothing.
Library: **graphics.def**

■ GrRestoreTransform

Set the passed GState's transformation matrix to be the one previously saved with GrSaveTransform.

Pass: `di` Handle of the GState to have its transformation matrix restored.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSaveState

Save the current GState characteristics for later restoration with **GrRestoreState**.

Pass: `di` Handle of the GState to save.
Returns: `di` Preserved GState handle.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSaveTransform

Save the transformation matrix of the passed GState for later restoration with GrRestoreTransform.

Pass: `di` Handle of the GState whose matrix should be stored.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**



■ GrSDivDWFbyWWF

Divide the WWFixed value into the DWFixed value. The result is a DWFixed. This routine is optimized for size over speed.

Pass:	dx.cx.bp	DWFixed dividend (signed).
	bx.ax	WWFixed divisor (signed).
Returns:	dx.cx.bp	DWFixed quotient (signed).
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSDivWWFixed

Divide a 32-bit WWFixed value by another 32-bit WWFixed value.

Pass:	dx.cx	WWFixed dividend (signed).
	bx.ax	WWFixed divisor (signed).
Returns:	CF	Set if overflow, clear otherwise.
	dx.cx	WWFixed quotient (signed), if no overflow.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSetAreaAttr

Set all the area attributes of the given GState.

Pass:	di	Handle of the GState.
	ds:si	Address of AreaAttr structure.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSetAreaColor

Set the GState's current area drawing color.

Pass:	di	Handle of the GState.
	ah	ColorFlag value.
	al	Palette index if CF_INDEX in ah.
		Red component if CF_RGB in ah.
	b1	Green component if CF_RGB, ignored otherwise.
	bh	Blue component if CF_RGB, ignored otherwise.
Returns:	Nothing.	

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetAreaColorMap

Set the GState's current area color mapping mode.

Pass: **di** Handle of the GState.
a1 **ColorMapMode** record.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetAreaMask

Set the GState's current area drawing pattern.

Pass: **di** Handle of the GState.
a1 **SysDrawMask** record with the following two parts.
Combine the SDM_INVERSE flag with a value of **SystemDrawMask**.
ds:si If the **SystemDrawMask** passed in **a1** is SDM_CUSTOM, then **ds:si** is the address of the custom pattern to set. Otherwise, **ds:si** is ignored.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetAreaPattern

Set the GState's area fill pattern.

Pass: **di** Handle of the GState.
a1 PatternType value.
ah Optional SystemHatch or SystemBitmap value if not PT_SOLID.
dx:si Address of a buffer containing the system hatch or bitmap pattern, if **ah** passed.
cx Size of buffer in **dx:si** if **ah** passed.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**



GrSetBitmapMode

■ 180

■ GrSetBitmapMode

Set the GState's mode bits for an editable bitmap.

Pass:	di	Handle of the GState as returned by GrEditBitmap .
	ax	BitmapMode record: BM_EDIT_MASK Edit mask rather than bitmap. BM_CLUSTERED_DITHER Use clustered dither rather than dispersed dither (used only for BMF_MONO bitmaps).
	dx	Handle of block containing a ColorTransfer structure (an array of RGBDelta structures. Pass zero if no array is passed).
Returns:		Nothing.
Destroyed:		Nothing.
Library:		graphics.def

■ GrSetBitmapRes

Set the resolution of the GState's bitmap.

Pass:	di	Handle of the GState, as returned by GrCreateBitmap .
	ax	New <i>x</i> resolution (in dots per inch).
	bx	New <i>y</i> resolution (in dots per inch).
Returns:		Carry set if GState not associated with bitmap, clear otherwise.
Destroyed:		Nothing.
Library:		graphics.def

■ GrSetClipPath

Set the GState's current clip path to be the clip path for all future graphics operations on the GState. The path is affected only by the window's transformation matrix.

Pass:	di	Handle of the GState.
	cx	PathCombineType value indicating how the path should be combined with the window's clip path.
	d1	RegionFillRule value: RFR_WINDING or RFR_ODD_EVEN.
Returns:		Nothing.
Destroyed:		Nothing.
Library:		graphics.def

Assembly Reference

■ GrSetClipRect

Modify a window's clipping path by intersecting it with the specified rectangle. The rectangle should be defined in the document coordinate space of the passed GState; the rectangle will be transformed appropriately when used by the graphics and window systems. If the rectangle and the clipping path do not intersect (have no area in common), no part of the window will be updated.

Pass:	di	Handle of the GState associated with the window.
	si	PathCombineType value.
	ax	Left bound of rectangle.
	bx	Top bound.
	cx	Right bound.
	dx	Bottom bound.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSetDefaultTransform

Reset the GState's transformation matrix to be the same as the default transformation matrix. In most cases, the default transformation matrix is simply the identity matrix (no transformations); this is not true in all cases, however, so you should call this routine rather than **GrSetNullTransform**. This routine is the reverse of **GrInitDefaultTransform**.

Pass:	di	Handle of the GState.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSetFont

Set the font used by the GState for all subsequent text drawing. If the desired font is not available, then the default font is set instead.

Pass:	di	Handle of the GState.
	cx	FontID to set as the font. Pass zero not to set the new font.
	dx.ah	WBFixed indicating the new point size to set. Pass zero to set no new point size.
Returns:	Nothing.	
Destroyed:	Nothing.	



GrSetFontWeight

■ 182

Library: **graphics.def**

■ GrSetFontWeight

Set the font weight used by the GState.

Pass: **di** Handle of the GState.
 al **FontWeight** value (percentage of normal weight).

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetFontWidth

Set the font width used by the GState.

Pass: **di** Handle of the GState.
 al **FontWidth** value (percentage of normal width).

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetGStringBounds

Store a GR_SET_GSTRING_BOUNDS opcode to a graphics string, along with the bounds of the string.

Pass: **di** Handle of the GState associated with the graphics string.
 ax Left bound of the string.
 bx Top bound.
 cx Right bound.
 dx Bottom bound.

Returns: Nothing.

Destroyed: Nothing.

Library: **gstring.def**

■ GrSetGStringPos

Set the current drawing position of a graphics string.

Pass: **si** Handle of the graphics string.
 al **GStringSetPosType** value:
 GSSPT_SKIP_1 Advance one element.

Assembly Reference

GSSPT_RELATIVE Advance cx elements.
 GSSPT_BEGINNING Move to beginning element.
 GSSPT_END Move to final element.
cx Number of elements to skip, if GSSPT_RELATIVE. This may be negative for HugeArray-based graphics strings but must be positive otherwise.

Returns: Nothing.
Destroyed: Nothing.
Library: **gstring.def**

■ GrSetLineAttr

Set all the line drawing attributes for the given GState.

Pass: **di** Handle of the GState.
 ds:si Address of a **LineAttr** structure.

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetLineColor

Set the current line drawing color for the GState.

Pass: **di** Handle of the GState.
 ah **ColorFlag** value.
 a1 Palette index if CF_INDEX in ah.
 Red component if CF_RGB in ah.
 b1 Green component if CF_RGB, ignored otherwise.
 bh Blue component if CF_RGB, ignored otherwise.

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetLineColorMap

Set the GState's current line color mapping mode.

Pass: **di** Handle of the GState.
 a1 **ColorMapMode** record.

Returns: Nothing.



GrSetLineEnd

■ 184

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetLineEnd

Set the line end type for the passed GState.

Pass: **di** Handle of the GState.
a1 **LineEnd** value:
LE_BUTTCAP, LE_ROUNDCAp, LE_SQUARECAP.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetLineJoin

Set the line join type for the passed GState.

Pass: **di** Handle of the GState.
a1 **LineJoin** value:
LJ_MITERED, LJ_ROUND, LJ_BEVELED.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetLineMask

Set the current line drawing mask for the passed GState.

Pass: **di** Handle of the GState.
a1 **SysDrawMask** record with the following two parts.
Combine the SDM_INVERSE flag with a value of
SystemDrawMask.
ds:si If the **SystemDrawMask** passed in **a1** is SDM_CUSTOM,
then **ds:si** is the address of the custom pattern to set.
Otherwise, **ds:si** is ignored.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetLineStyle

Set the current line drawing style (dash pattern) of the passed GState.

Assembly Reference

Pass: **di** Handle of the GState.
 a1 **LineStyle** value.
 b1 Skip distance into first dash pair.
 ds:si If **a1** = LS_CUSTOM, the address of a **DashPairArray**
 structure containing up to five dash-dot pairs.
 ah If **a1** = LS_CUSTOM, the number of pairs in the ds:si array.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetLineWidth

Set the line drawing width of the passed GState.

Pass: **di** Handle of the GState.
 dx:ax WWFixed giving the new line width.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetMiterLimit

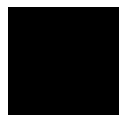
Set the miter limit for the passed GState. The miter limit is the smallest angle that can be drawn with a miter join; smaller angles must be drawn with a beveled join.

Pass: **di** Handle of the GState.
 bx:ax **WWFixed** structure defining the miter limit.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetMixMode

Set the drawing mix mode for the passed GState.

Pass: **di** Handle of the GState.
 a1 New **MixMode** value:
 MM_CLEAR, MM_COPY, MM_NOP, MM_AND, MM_INVERT,
 MM_XOR, MM_SET, or MM_OR.
Returns: Nothing.
Destroyed: Nothing.



GrSetNullTransform

■ 186

Library: **graphics.def**

■ GrSetNullTransform

Replace the GState's transformation matrix with the null (identity) transformation matrix. This routine should not be substituted for **GrSetDefaultTransform**, which reverts a transformation matrix to the default.

Pass: **di** Handle of the GState.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetPalette

Set one or more palette entries in the GState's color map.

Pass: **di** Handle of the GState.

a1 First palette entry (index) to set.

cx Number of palette entries to set.

dx:si Address of a buffer of cx entries, each of type **RGBValue**.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetPaletteEntry

Set a single entry in the GState's palette.

Pass: **di** Handle of the GState.

ah Palette index entry to set.

al Red component of new color.

bl Green component of new color.

bh Blue component of new color.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetPrivateData

Set the private data for the given GState. This may be retrieved with **GrGetInfo**; applications should not use the GState's private data, however.

Assembly Reference

Pass: **di** Handle of the GState.
 ax, bx, cx, dx Private data to set.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetStrokePath

Replace the GState's current path with the one defined as the stroked representation of the current path.

Pass: **di** Handle of the GState.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetSubscriptAttr

Set the GState's subscript attributes.

Pass: **di** Handle of the GState.
 al **SubscriptPosition** value (percentage of font size from top).
 ah **SubscriptSize** value (percentage of font size).
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetSuperscriptAttr

Set the GState's superscript attributes.

Pass: **di** Handle of the GState.
 al **SuperscriptAttr** value (percentage of font size from bottom).
 ah **SuperscriptSize** value (percentage of font size).
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetTextAttr

Set all the GState's text drawing attributes.



GrSetTextColor

■ 188

Pass: **di** Handle of the GState.
 ds:si Address of a **TextAttr** structure.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetTextColor

Set the current text drawing color for the GState.

Pass: **di** Handle of the GState.
 ah **ColorFlag** value.
 al Palette index if CF_INDEX in ah.
 Red component if CF_RGB in ah.
 bl Green component if CF_RGB, ignored otherwise.
 bh Blue component if CF_RGB, ignored otherwise.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetTextColorMap

Set the GState's current text color mapping mode.

Pass: **di** Handle of the GState.
 al **ColorMapMode** record.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetTextDrawOffset

Set the number of characters at the end of a string to draw. This operation is never saved out to a graphics string.

Pass: **di** Handle of the GState used for drawing.
 ax Number of characters to draw, or zero to draw the entire string.

Returns: Nothing.

Destroyed: Nothing.

Library: **text.def**

Assembly Reference

■ GrSetTextMask

Set the current text drawing mask for the passed GState.

Pass:	di	Handle of the GState.
	al	SysDrawMask record with the following two parts. Combine the SDM_INVERSE flag with a value of SystemDrawMask.
	ds:si	If the SystemDrawMask passed in al is SDM_CUSTOM, then ds:si is the address of the custom pattern to set. Otherwise, ds:si is ignored.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSetTextMode

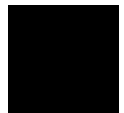
Set the GState's current text drawing mode.

Pass:	di	Handle of the GState.
	al	TextMode flags to set.
	ah	TextMode flags to clear.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSetTextPattern

Set the GState's text fill pattern.

Pass:	di	Handle of the GState.
	al	PatternType value to set.
	ah	Optional SystemHatch or SystemBitmap value if not PT_SOLID.
	dx:si	Address of a buffer containing the system hatch or bitmap pattern, if ah passed.
	cx	Size of buffer in dx:si if ah passed.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	



GrSetTextSpacePad

■ 190

■ GrSetTextSpacePad

Set the amount to pad spaces when drawing text with GrPutString.

Pass:	di	Handle of the GState.
	dx	Pixel spacing of padding.
	bl	Additional fractional spacing.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetTextStyle

Set the GState's current text style.

Pass:	di	Handle of the GState.
	al	TextStyle flags to set.
	ah	TextStyle flags to clear.

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetTrackKern

Set the GState's track kerning.

Pass:	di	Handle of the GState.
	ax	Degree of track kerning (signed word).

Returns: Nothing.

Destroyed: Nothing.

Library: **graphics.def**

■ GrSetTransform

Replace the GState's current transformation matrix with another.

Pass:	di	Handle of the GState.
-------	-----------	-----------------------

Assembly Reference

ds:si Address of an array of six structures. The first four must be **WWFixed** structures, and the last two **DWFixed**. The numbers must be in row order as in the diagram:

$$\begin{bmatrix} e11 & e12 & 0 \\ e21 & e22 & 0 \\ e31 & e32 & 1 \end{bmatrix} \Rightarrow e11, e12, e21, e22, e31, e32$$

Transformation Matrix *Array Order*

In this case, the two **DWFixed** structures will represent numbers *e31* and *e32*.

Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetVMFile

Sets the VM file in the associated Window or GString, if any.

This routine must be called if the VM file handle stored in the Window or GString may have changed (e.g., via **VMSave**)

Pass: **di** Handle of GState
 ax VM file handle.
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**

■ GrSetWinClipPath

Set the GState's current path to be the document clip path for all future graphics operations. This path is affected by both the window and GState's transformation matrixes.

Pass: **di** Handle of the GState.
 cx **PathCombineType** value.
 d1 **RegionFillRule** (unnecessary for PCT_NULL).
Returns: Nothing.
Destroyed: Nothing.
Library: **graphics.def**



■ GrSetWinClipRect

Modify a window's clipping path by intersecting it with the specified rectangle. The rectangle should be defined in the document coordinate space of the passed GState; the rectangle will be transformed appropriately when used by the graphics and window systems. If the rectangle and the clipping path do not intersect (have no area in common), no part of the window will be updated.

Pass:	di	Handle of the GState associated with the window.
	si	PathCombineType value.
	ax	Left bound of rectangle.
	bx	Top bound.
	cx	Right bound.
	dx	Bottom bound.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSqrRootWWFixed

Calculate the square root of a 32-bit number. Numbers less than one return the value one.

Pass:	dx.cx	WWFixed value to get the square root of.
Returns:	dx.cx	WWFixed value representing the square root.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrSqrWWFixed

Calculate the square of a 32-bit number.

Pass:	dx.cx	WWFixed value to square.
Returns:	dx.cx	WWFixed square result.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrTestPath

Checks for the existence of a path.

Pass:	di	Handle of GState
-------	-----------	------------------

	ax	GetPathType value
Returns:	CF	Clear if path exists for GState; set otherwise.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrTestPointInPath

Determine if the passed point is inside the GState's current path.

Pass:	di	Handle of the GState.
	c1	RegionFillRule value: RFR_ODD_EVEN, RFR_WINDING.
	ax, bx	<i>x, y</i> coordinates of the point (document coordinates).
Returns:	CF	Set if point is inside, clear if it is not.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrTestPointInPolygon

Determine if the passed point is inside the given polygon.

Pass:	di	Handle of the GState used for drawing.
	dx, bx	<i>x, y</i> coordinates of the point (document coordinates).
	ds:si	Address of the array of points in the polygon.
	cx	Number of points in the polygon array.
	a1	RegionFillRule value: RFR_ODD_EVEN, RFR_WINDING.
Returns:	ax	Non-zero means the point is inside the polygon. Zero means the point is not inside the polygon.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrTestPointInReg

Determine if the passed point is inside a given region. If so, return the rectangle inside the region including the point.

Pass:	cx, dx	<i>x, y</i> coordinates of the point.
	ds:si	Address of Region definition.
Returns:	CF	Set if point inside region, clear otherwise.
	ax	Top bound of bounding rectangle if CF returned set.
	bx	Bottom bound of bounding rectangle if CF returned set.
	ds:[si-4]	Left bound of bounding rectangle if CF returned set.
	ds:[si-2]	Right bound of bounding rectangle if CF returned set.



GrTestRectInMask

■ 194

Destroyed: Nothing.

Library: **graphics.def**

■ GrTestRectInMask

Determine if a given rectangle is inside the current clip region.

Pass: **di** Handle of the GState used for drawing.
ax, bx *x, y* coordinates of one corner of rectangle to check.
cx, dx *x, y* coordinates of opposite corner of rectangle.

Returns: **a1** **TestRectReturnType** value:
TRRT_OUT The rectangle is entirely outside the region.
TRRT_PARTIAL The rectangle is partially inside the region.
TRRT_IN The rectangle is entirely inside the region.

Destroyed: Nothing.

Library: **graphics.def**

■ GrTestRectInReg

Determine if a given rectangle is inside a specified region.

Pass: **ax, bx** *x, y* coordinates of one corner of rectangle to check.
cx, dx *x, y* coordinates of opposite corner of rectangle.
ds:si Address of the **Region** definition.
CF Set if **ds:si** is a value from *W_clipPtr*; clear if **ds:si** points to the start of a region. (Typically clear on call.)

Returns: **a1** **TestRectReturnType** value:
TRRT_OUT The rectangle is entirely outside the region.
TRRT_PARTIAL The rectangle is partially inside the region.
TRRT_IN The rectangle is entirely inside the region.

Destroyed: Nothing.

Library: **graphics.def**

■ GrTextPosition

Return the nearest offset into a text string when given a pixel position in the string.

Pass: **di** Handle of the GState used for drawing.
cx Number of characters in string (0 if null terminated).
dx Pixel offset into string.

Pass on stack:

Assembly Reference

	ss:bp	<i>GTP_vars</i> structure.
Returns:	cx	Nearest character boundary to the passed cx.
	dx	Nearest valid position in the text string.
	ds	Segment address of last piece of text.
Destroyed:	Nothing.	
Library:	text.def	

■ GrTextWidth

Return the width of a text string, including kerning.

Pass:	di	Handle of the GState used for drawing the text.
	ds:si	Address of the string.
	cx	Number of characters in the string, or zero for null-terminated string.
Returns:	dx	Width of the string, in document points.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrTextWidthWBFixed

Return the width of a text string, including kerning.

Pass:	di	Handle of the GState used for drawing the text.
	ds:si	Address of the string.
	cx	Number of characters in the string, or zero for null-terminated string.
Returns:	dx.ah	WBFixed width of the string, in document points.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrTransform

Transform the given coordinate pair from document units to screen coordinates, including the effects of the GState and window transformation matrixes.

Pass:	di	Handle of the GState used for drawing.
	ax, bx	<i>x, y</i> coordinates in document units.
Returns:	cf	Set if translation overflow, clear if successful.
	ax, bx	Translated <i>x, y</i> coordinates in screen pixels.
Destroyed:	Nothing.	



GrTransformByMatrix

■ 196

Library: **graphics.def**

■ GrTransformByMatrix

Transform the given coordinate pair using the passed transformation matrix.

Pass: **ax, bx** *x, y coordinates of the original point.*
ds:si *Address of a TransMatrix structure.*

Returns: **ax, bx** *Translated x, y coordinates.*

Destroyed: Nothing.

Library: **graphics.def**

■ GrTransformByMatrixDWord

Transform the given coordinate pair using the passed transformation matrix. Coordinates are 32-bit values rather than normal 16-bit values.

Pass: **dx.cx** **DWFixed** value of original *x* coordinate.
bx.ax **DWFixed** value of original *y* coordinate.
ds:si Address of a **TransMatrix** structure.

Returns: **dx.cx** **DWFixed** value of translated *x* coordinate.
bx.ax **DWFixed** value of translated *y* coordinate.

Destroyed: Nothing.

Library: **graphics.def**

■ GrTransformDWordFixed

Transform the given coordinate pair from document units to screen coordinates, including the effects of the GState and window transformation matrixes. Coordinates are in **DWFixed** format.

Pass: **di** Handle of the GState.
es:dx Address of a buffer containing a **PointDWordFixed** structure.

Returns: **es:dx** Address of the returned **PointDWordFixed** structure, translated to screen coordinates.

Destroyed: Nothing.

Library: **graphics.def**

■ GrTransformDWord

Transform the given 32-bit coordinate pair from document units to screen coordinates, including the effects of the GState and window transformation matrixes.

Assembly Reference

Pass:	di	Handle of the GState used for drawing.
	dx.cx	DWFixed <i>x</i> coordinate in document units.
	bx.ax	DWFixed <i>y</i> coordinate in document units.
Returns:	dx.cx	Translated DWFixed <i>x</i> coordinate in screen pixels.
	bx.ax	Translated DWFixed <i>y</i> coordinate in screen pixels.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrTransformWWFixed

Transform the given 32-bit fixed point coordinate pair using the passed GState's transformation matrix. Coordinates are in **WWFixed** format.

Pass:	di	Handle of the GState.
	dx.cx	WWFixed <i>x</i> coordinate, in document units.
	bx.ax	WWFixed <i>y</i> coordinate, in document units.
Returns:	CF	Set if overflow, clear otherwise. If overflow, returned coordinates are invalid.
	dx.cx	WWFixed translated <i>x</i> coordinate.
	bx.ax	WWFixed translated <i>y</i> coordinate.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrUDivWWFixed

Divide two 32-bit unsigned numbers.

Pass:	dx.cx	WWFixed dividend.
	bx.ax	WWFixed divisor.
Returns:	CF	Set if overflow, clear otherwise. If overflow, quotient is invalid.
	dx.cx	WWFixed quotient.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrUncompactBitmap

Uncompact a huge bitmap (a bitmap in a HugeArray).

Pass:	bx	VM file handle of compacted bitmap.
	ax	VM block handle of compacted bitmap's start block.
	dx	VM file handle for destination of the de-compaction; may be the same as bx .



GrUntransform

■ 198

Returns:	dx cx	VM file handle of uncompactd bitmap. VM block handle of uncompactd bitmap's start block.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrUntransform

Transform the given coordinate pair from screen coordinates to document coordinates, including the effects of the GState and window transformation matrixes.

Pass:	di ax, bx	Handle of the GState used for drawing. <i>x, y</i> coordinates in screen pixels.
Returns:	CF ax, bx	Set if overflow, clear otherwise. <i>x, y</i> coordinates in document units.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrUntransformByMatrix

Untransform the given coordinate pair using the passed transformation matrix. This is the inverse operation of transforming the coordinates using the matrix.

Pass:	ax, bx ds:si	<i>x, y</i> coordinates to be untransformed. Address of a TransMatrix structure.
Returns:	ax, bx	<i>x, y</i> coordinates, untransformed.
Destroyed:	Nothing.	
Library:	graphics.def	

■ GrUntransformByMatrixDWord

Untransform the given coordinate pair using the passed transformation matrix. This is the inverse operation of transforming the coordinates using the matrix.

Pass:	dx.cx bx.ax ds:si	DWFixed value of original <i>x</i> coordinate. DWFixed value of original <i>y</i> coordinate. Address of a TransMatrix structure.
Returns:	dx.cx bx.ax	DWFixed value of new <i>x</i> coordinate. DWFixed value of new <i>y</i> coordinate.
Destroyed:	Nothing.	

Assembly Reference

Library: **graphics.def**

■ GrUntransformDWFxed

Untransform the given coordinate pair from screen coordinates to document coordinates using the GState passed.

Pass: **di** Handle of the GState used for drawing.
es:dx Address of a **PointDWFxed** structure holding the screen coordinates pair.

Returns: **es:dx** Address of a **PointDWFxed** structure holding the untransformed coordinate pair, in document coordinates.

Destroyed: Nothing.

Library: **graphics.def**

■ GrUntransformDWord

Untransform the given 32-bit coordinate pair from screen coordinates to document coordinates.

Pass: **di** Handle of the GState.
dx.cx 32-bit *x* integer coordinate (**dx** high word, **cx** low word).
bx.ax 32-bit *y* integer coordinate (**bx** high word, **ax** low word).

Returns: **dx.cx** 32-bit untransformed *x* coordinate in document units.
bx.ax 32-bit untransformed *y* coordinate in document units.

Destroyed: Nothing.

Library: **graphics.def**

■ GrUntransformWWFixed

Untransform the given 32-bit coordinate pair using the GState passed.

Pass: **di** Handle of the GState.
dx.cx **WWFixed** value of *x* coordinate (screen coordinates).
bx.ax **WWFixed** value of *y* coordinate (screen coordinates).

Returns: **CF** Set if overflow, clear otherwise. If set, results are invalid.
dx.cx **WWFixed** value of untransformed *x* (document coordinates).
bx.ax **WWFixed** value of untransformed *y* (document coordinates).

Destroyed: Nothing.

Library: **graphics.def**



■ HandleModifyOwner

Modifies the owner of a block. If passed a process handle, changes the parent process instead of the owner.

Pass:	bx	Handle of block to modify.
	ax	Handle of block's new owner.
Returns:	Nothing.	
Destroyed:	ax	
Library:	heap.def	

■ HandleP

Sets a semaphore on the passed block. This provides the caller with exclusive access to the block if all other processes use the **HandleP/HandleV** mechanism. The *HM_otherInfo* field of the block is used for the semaphore and must not be used for any other purpose.

HM_otherInfo stores the state of the semaphore. If the block is not owned, this value is 1. If this block is owned but no threads are waiting, this value is zero. Otherwise, *HM_otherInfo* stores the handle of the first thread waiting to access the block.

HandleP and **HandleV** can be used both on memory handles and file handles.

Pass:	bx	Handle of block to own.
Returns:	bx	Handle of block owned.
Destroyed:	Nothing.	
Library:	heap.def	

■ HandleV

Releases a semaphore on the given block.

Pass:	bx	Handle of block to release.
Returns:	bx	Handle of block released.
Destroyed:	Nothing. (Flags preserved.)	
Library:	heap.def	



HugeArrayAppend

■ 202

■ HugeArrayAppend

Appends element(s) to the tail end of a Huge Array. If elements are of fixed size, this routine may append several elements. If elements are of variable size, this routine appends one element to the tail end of the Huge Array.

Pass:	bx	VM file handle of the Huge Array.
	di	VM block handle of the Huge Array.
	cx	Number of elements to append (if elements are of fixed size) or size of new element (if elements are variable sized and only one element is being appended).
	bp:si	Fptr to buffer holding element data. If bp = 0 then allocate space but do not initialize the data.
Returns:	dx:ax	New element number. If multiple elements are appended, this is the number of the first element.
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayCompressBlocks

Compress all the free space out of VM blocks containing a HugeArray data.

Pass:	bx:di	VM File and Block handle of the huge array.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayContract

Deletes element(s) in a Huge Array. The elements may be of fixed or variable size but must already be locked down. Elements will be deleted starting at the element location passed.

Pass:	ds:si	Pointer to the locked Huge Array element.
	cx	Number of elements to delete.
Returns:	ds:si	Pointer to same element number. (ds may have changed.)
	ax	Number of elements available through the pointer. If ax = 0 the Huge Array is now empty.
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayCreate

Creates a Huge Array. Allocates a VM block for the directory block, initializes the directory block header and allocates enough VM blocks for any initial elements.

Pass:	bx	VM file handle in which to create the array.
	cx	Number of bytes to allocate per element. Pass zero if elements are of variable size.
	di	Size to allocate for the Huge Array directory block's header. Pass zero if no additional space beyond that of HugeArrayDirectory is needed. If you want to have additional space, make sure the size is at least as large as the size of HugeArrayDirectory plus the size of the additional information.
Returns:	di	Huge Array handle (VM block handle).
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayDelete

Locks down a Huge Array VM block and deletes element(s) starting at the passed element number.

Pass:	bx	VM file handle of Huge Array.
	di	VM block handle of Huge Array.
	cx	Number of elements to delete.
	dx:ax	Element number. New element(s) will be deleted starting at this number.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayDestroy

Destroys a Huge Array. This routine frees all blocks in the Huge Array.

Pass:	bx	VM file handle of Huge Array to destroy.
	di	VM block handle of the Huge Array to destroy.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	hugearr.def	



■ HugeArrayDirty

Marks a VM block containing an element in a Huge Array dirty

Pass: **ds** Pointer to a locked Huge Array element.

Returns: Nothing.

Destroyed: Nothing.

Library: **hugearr.def**

■ HugeArrayEnum

Calls a callback routine for multiple elements within a Huge Array. Pass this routine the element to start at, the number of elements to call the routine on, and the address of the callback routine. The Index number of elements is a zero-based integer.

The callback routine may not do anything which would invalidate any pointers to the Huge Array. For example, it may not allocate, delete, or resize any of the elements. The callback routine should restrict itself to examining elements and altering them without resizing them.

Pass: **ax, cx, dx, bp, es**
Set for callback

Pass on stack: VM file handle of the Huge Array
VM block handle of the Huge Array
Pointer to a Boolean callback routine
Index of first element to start enumerations on.
Number of elements to enumerate (or -1 to continue to the end of the array)

Returns: **CF** Set if callback aborted, clear otherwise.
ax, cx, dx, bp, es Returned from callback.

Destroyed: Nothing.

Callback Routine Specifications:

Passed: **ds:di** Pointer to element.
(For fixed size elements):
ax, cx, dx, bp, es data passed to **HugeArrayEnum** (as changed by previous iterations of callback).
(For variable sized elements):
ax element size.
cx, dx, bp, es data passed to **HugeArrayEnum**.

Return: **CF** Set if callback aborted, clear otherwise (as changed by previous iterations of callback).
ax, cx, dx, bp, es Data for next callback.

May Destroy: Nothing.

Library: **hugearr.def**

■ HugeArrayExpand

Insert element(s) into a locked Huge Array. Elements are inserted starting at the passed element position.

Pass: **ds:si** Pointer to locked Huge Array element.
cx (For fixed size elements):
Number of elements to insert.
(For variable sized elements):
Size of element at **ds:si**.
bp, di Fptr to buffer holding element data. If **bp** = 0 then allocate space but don't initialize data.

Returns: **ds:si** Pointer to first new element added.
ax Number of consecutive elements available starting with returned pointer. (If **ax** = 0, pointer is invalid.)
cx Number of consecutive elements available before (and including) the requested element.

Destroyed: Nothing.

Library: **hugearr.def**

■ HugeArrayGetCount

Retrieves the number of element(s) in a Huge Array.

Pass: **bx** VM file handle of the Huge Array.
di VM block handle of the Huge Array.

Returns: **dx, ax** Number of elements in the Huge Array.

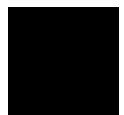
Destroyed: Nothing.

Library: **hugearr.def**

■ HugeArrayInsert

Locks down and insert element(s) into a HugeArray.

Pass: **bx** VM file handle of the HugeArray.
di VM block handle of the HugeArray.
cx (For fixed size elements)



HugeArrayLock

■ 206

		Number of elements to insert (For variable sized elements)
		Size of new element.
	dx:ax	Element number. New element will be inserted before this one.
	bp.si	Fptr to buffer holding element data. If bp = 0 then allocate space but do not initialize data.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayLock

Locks down a HugeArray element. To unlock a locked HugeArray element, use HugeArrayUnlock.

Pass:	bx	VM file handle of HugeArray.
	di	VM block handle of HugeArray.
	dx.ax	Element number to dereference.
Returns:	ds:si	Pointer to requested element.
	ax	Number of consecutive elements available, starting with the returned pointer. If ax = 0, pointer is invalid.
	cx	Number of consecutive elements available before (and including) the requested element.
	dx	Size of the element.
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayLockDir

Locks a HugeArray directory block.

Pass:	bx	VM file handle of HugeArray.
	di	VM block handle of HugeArray.
Returns:	ax	Segment address of directory block.
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayNext

Locks and points to the next HugeArray element.

Pass:	ds:si	Pointer to element in block.
-------	--------------	------------------------------

Assembly Reference

Returns:	ds:si ax	Pointer to next element. This may be in a different block. Number of consecutive elements available with returned pointer. Returns zero if we were at the last element in the array.
	dx	(For variable sized elements): Size of the element. Otherwise dx is undefined.
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayPrev

Locks and points to the previous Huge Array element.

Pass:	ds:si	Pointer to element in block.
Returns:	ds:si ds:di ax	Pointer to previous element. This may be in a different block. Pointer to first element in block. Number of elements available from first element in block to previous element. (For example, if si == di , then ax = 1.)
	dx	Returns zero if we were at the first element in the array. (For variable sized elements): Size of the element. Otherwise dx is undefined.
Destroyed:	Nothing.	
Library:	hugearr.def	

■ HugeArrayReplace

Replace element(s) in a Huge Array.

Pass:	bx di cx	VM file handle of HugeArray. VM block handle of HugeArray. (For fixed size elements) Number of elements to replace. (For variable sized elements) Size of new element.
	dx:ax	Element number. New element will be replaced starting with this one.
	bp:si	Fptr to buffer holding element data. If bp = 0 then replace all bytes with 0.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	hugearr.def	



■ HugeArrayResize

Resizes an array element. If the element is resized to a smaller size then data at the end of the element is truncated (and lost). If it gets larger, the new data is initialized to zero.

Pass:	bx	VM file handle of HugeArray.
	di	VM block handle of HugeArray.
	dx:ax	Element number.
	cx	Size of new element.

Returns: Nothing.

Destroyed: Nothing.

Library: **hugearr.def**

■ HugeArrayUnlock

Unlocks a previously locked Huge Array element.

Pass:	ds	Pointer to element block containing element.
-------	-----------	--

Returns: Nothing.

Destroyed: Nothing. (Flags preserved.)

Library: **hugearr.def**

■ HugeArrayUnlockDir

Unlocks a previously locked block containing the **HugeArrayDirectory**.

Pass:	ds	Pointer to block of HugeArrayDirectory .
-------	-----------	---

Returns: Nothing.

Destroyed: Nothing.

Library: **hugearr.def**

■ IACPCConnect

Establish a connection with one or all of the servers on a particular list.

Pass:	es:di	GeodeToken for the list
	ax	IACPCConnectFlags
	bx	handle of AppLaunchBlock if server is to be launched, should none be registered
	cx:dx	optr of client object, if IACPCF_CLIENT_OD_SPECIFIED set in ax .

Returns: **CF** Set on error; clear on success.

Assembly Reference

	ax	Error code on error (either IACPConnectError or GeodeLoadError). Destroyed on success.
	bp	IACPConnection token.
	cx	Number of servers connected to.
Destroyed:	bx, dx.	
Library:	iacp.def	

■ IACPCreateDefaultLaunchBlock

Utility routine to create an AppLaunchBlock to be given to **IACPConnect**. The block is initialized with the following defaults:

IACP will locate the application, given its token;
 The initial directory will be SP_DOCUMENT;
 There will be no initial data file;
 The application will determine the generic parent for itself;
 No one will be notified in event of an error; and
 No extra data is passed.

Pass:	dx	mode in which server should be launched: MSG_GEN_PROCESS_OPEN_APPLICATION or MSG_GEN_PROCESS_OPEN_ENGINE.
Returns:	CF dx	Clear if block created, set if couldn't allocate memory. Handle of block containing AppLaunchBlock .
Destroyed:	Nothing.	
Library:	iacp.def	

■ IACPFinishConnect

Complete the process of connecting. Called by the server when it's ready to accept messages from the client.

Pass:	cx:dx bp	optr of server object IACPConnection token
Returns:	Nothing	
Destroyed:	Nothing	
Library:	iacp.def	

■ IACPGetDocumentID

Figure the 48-bit ID for a data file, dealing with links.

Pass:	ds:dx bx	directory in which document resides disk on which document resides
-------	---------------------------	---



IACPGetServerNumber

■ 210

	ds:si	name of document
Returns:	CF	Set on error; clear on success.
	ax	FileError on error; disk handle on success.
	cx:dx	FileID on success; destroyed on error.
Destroyed:	Nothing.	
Library:	iacp.def	

■ IACPGetServerNumber

Returns the number a server object is for a particular IACP connection, so the client can use the number to direct a message to a particular server.

Pass:	bp	IACPConnection token
	cx:dx	optr server object
Returns:	ax	Server number (zero if object isn't a server for the connection).
Destroyed:	Nothing.	
Library:	iacp.def	

■ IACPLostConnection

Utility routine for server objects to handle MSG_META_IACP_LOST_CONNECTION

Pass:	*ds:si	Server object
	bp	IACPConnection token
Returns:	Nothing	
Destroyed:	ax, cx, dx, bp, bx, di.	
Library:	iacp.def	

■ IACPProcessMessage

Utility routine to handle a MSG_META_IACP_PROCESS_MESSAGE. Can be bound as the method for this message for any class that might receive it.

Pass:	cx	handle of message to send
	dx	TravelOption or -1 if message should be dispatched via MessageDispatch .
	bp	handle of message to send after cx is processed, or zero if no completion notification needed.
	*ds:si	server object
Returns:	Nothing	

Assembly Reference

Destroyed: **ax, bx, cx, dx, bp, si, di**

Library: **iacp.def**

■ IACPRegisterDocument

Register a document as being open, specifying the server to which to connect to communicate about the document.

Pass: **bx:si** optr of server object
ax disk handle
cx:dx **FileID**

Returns: Nothing

Destroyed: Nothing

Library: **iacp.def**

■ IACPRegisterServer

Register an object as a server for a particular list. Can also be used to change the mode in which the server is registered.

Pass: **es:di** **GeodeToken** for the list
^lcx:dx server object
al **IACPServerMode** structure.
ah **IACPServerFlags** structure.

Returns: Nothing.

Destroyed: Nothing.

Library: **iacp.def**

■ IACPSendMessage

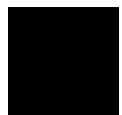
Send a message through an IACP connection to all connected servers, or to the client, depending on which side is doing the sending.

Pass: **bp** **IACPConnection** token
bx recorded message to send
dx **TravelOption**, -1 if recorded message contains the proper destination already
cx Recorded message to send on completion, zero if none
ax **IACPSide** doing the sending.

Returns: **ax** Number of servers to which message was sent.

Destroyed: **bx, cx, dx**, both recorded messages.

Library: **iacp.def**



IACPSendMessageToServer

■ 212

■ IACPSendMessageToServer

Send a message through an IACP connection to a specific connected server.

Pass:	bp	IACPConnection token
	bx	recorded message to send
	dx	TravelOption , -1 if recorded message contains the proper destination already
	cx	Recorded message to send on completion, zero if none
	ax	Server number.
Returns:	ax	Number of servers to which message was sent (One or zero).
Destroyed:	bx, cx, dx	both recorded messages.
Library:	iacp.def	

■ IACPShutdown

Sever an IACP connection. MSG_META_IACP_LOST_CONNECTION is sent to the other side of the connection.

Pass:	bp	IACPConnection to shut down
	cx:dx	optr of server object, or cx == 0 if client is shutting down.
Returns:	Nothing.	
Destroyed:	ax .	
Library:	iacp.def	

■ IACPShutdownAll

Shutdown all connections open to or from an object.

Pass:	cx:dx	optr of client or server object for which all connections are to be shutdown.
Returns:	Nothing.	
Destroyed:	ax	
Library:	iacp.def	

■ IACPShutdownConnection

Utility routine to handle MSG_META_IACP_SHUTDOWN_CONNECTION, as generated by a call to **IACPLostConnection**.

Pass:	*ds:si	server object
	bp	IACPConnection to shut down.
Returns:	Nothing.	

Assembly Reference

Destroyed: **ax, cx, dx, bp.**

Library: **iacp.def**

■ IACPUnregisterDocument

Indicate a document is closed. New-connection messages may still be queued based on the document having been registered, so the caller will need to handle those gracefully.

Pass: **ax** disk handle
cx:dx **FileID**
bx:si optr of server

Returns: Nothing.

Destroyed: Nothing.

Library: **iacp.def**

■ IACPUnregisterServer

Unregister an object as a server for a particular list.

Pass: **es:di** **GeodeToken** for the list
cx:dx server object

Returns: Nothing.

Destroyed: Nothing.

Library: **iacp.def**

■ ImpexCreateTempFile

Creates and opens a unique Metafile in the waste directory.

Pass: **es:di** File name buffer (of size FILE_LONGNAME_BUFFER_SIZE).
ax IMPEX_TEMP_VM_FILE or IMPEX_TEMP_NATIVE_FILE.

Returns: **es:di** File name buffer filled.
bp File handle.
ax **TransError** (or zero if no error).
bx Memory handle of error text if **ax** = TE_CUSTOM.

Destroyed: Nothing.

Library: **impex.def**

■ ImpexDeleteTempFile

Closes and deletes a metafile in the waste directory.

Pass: **ds:dx** File name buffer (of size FILE_LONGNAME_BUFFER_SIZE).



ImpexExportToMetafile

■ 214

	bx	File handle.
	ax	IMPEX_TEMP_VM_FILE or IMPEX_TEMP_NATIVE_FILE.
Returns:	ax	TransError (or zero if no error).
	bx	Memory handle of error text if ax = TE_CUSTOM.
Destroyed:	Nothing.	
Library:	impex.def	

■ ImpexExportToMetafile

Converts a transfer format into a metafile.

Pass:	bx	Handle of the metafile translation library to use.
	ax	Entry point number of library routine to call.
	dx:cx	VM chain containing transfer format.
	di	VM file handle of transfer format.
	bp	Handle of the metafile (open for read/write).
	ds	Additional data for metafile as needed.
	si	Additional data for metafile as needed.
Returns:	ax	TransError (or zero if no error).
	bx	Memory handle of error text if ax = TE_CUSTOM.
Destroyed:	Nothing.	
Library:	impex.def	

■ ImpexImportExportCompleted

Sends a message back to the Import/ExportControl object stating that the application has completed it's import or export operation.

Pass:	ss:bp	ImpexTranslationParams .
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	impex.def	

■ ImpexImportFromMetafile

Converts a metafile into a transfer format.

Pass:	bx	Handle of the metafile (open for reading).
	ax	Entry point number of library routine to call.
	di	VM file handle to hold transfer format.
	bp	Handle of metafile translation library to use.
	ds	Additional data for metafile as needed.
	si	Additional data for metafile as needed.

Assembly Reference

Returns: **dx:cx** VM chain containing transfer format.
ax **TransError** (or zero if no error).
bx Memory handle of error text if **ax** = TE_CUSTOM.
Destroyed: Nothing.
Library: **impex.def**

■ ImpexUpdateImportExportStatus

Apprise the user of the status of an import or export. This routine should only be called by translation libraries, and can be called at any time during the import/export process. If a translation library chooses not to call this function, the default import/export message will be displayed.

Pass: **ds:si** String to display to user (NULL string to not display a new string)
ax Percentage completed so far (this value may range from zero to 100, or may be -1 to signal not to display any percentage)
Returns: **ax** *True* (i.e., non-zero) to continue import/export; *false* to stop import/export.
Destroyed: Nothing.
Library: **impex.def**

■ InitFileDeleteCategory

Deletes an entire category (and therefore all its associated keys) from the GEOS.INI file.

Pass: **ds:si** Category (null-terminated ASCII string) to delete from the GEOS.INI file.
Returns: Nothing.
Destroyed: Nothing.
Library: **initfile.def**

■ InitFileDeleteEntry

Deletes a “key” entry from the GEOS.INI file. Only matching keys within the passed category will be deleted. Keys in other categories are unaffected.

Pass: **ds:si** Category (null-terminated ASCII string) containing the key within the GEOS.INI file.
cx:dx Key (null-terminated ASCII string) to delete from the GEOS.INI file.
Returns: Nothing.



InitFileDeleteStringSection

■ 216

Destroyed: Nothing.

Library: **initfile.def**

■ InitFileDeleteStringSection

Deletes the specific string “blob” starting at the zero-based string number. “Blobs” are usually set off from each other in the GEOS.INI file with CR or LF characters.

Pass:	ds:si	Category (null-terminated ASCII string) of string within the GEOS.INI file.
	cx:dx	Key (null-terminated ASCII string) of string within the GEOS.INI file.
	ax	Null-terminated string number “blob” to remove.
Returns:	CF	Clear if successful, otherwise set.
Destroyed:	Nothing.	
Library:	initfile.def	

■ InitFileEnumStringSection

Calls the passed function on each matching string section within the GEOS.INI file.

Pass:	ds:si	Category (null-terminated ASCII string) of string within the GEOS.INI file.
	cx:dx	Key (null-terminated ASCII string) of string within the GEOS.INI file.
	bp	InitFileReadFlags (IFRF_SIZE is of no importance).
	di:ax	Address (fpnr) of callback routine.
	es	Additional data to pass to callback routine.
	bx	Additional data to pass to callback routine.
Returns:	bx, es	Data from callback routine.
	CF	Clear if enumeration successful, set if failed.

Destroyed: Nothing.

Callback Routine Specifications:(must be declared as far)

Passed:	ds:si	String section (null-terminated).
	dx	Section number.
	cx	Length of section.
	es	Additional data.
	bx	Additional data.
Return:	bx, es	Data returned from callback routine.

Assembly Reference

CF Clear to continue enumeration, set to stop enumeration.

May Destroy: **ax, cx, dx, di, si, bp, es**

Library: **initfile.def**

■ InitFileGetTimeLastModified

Returns the time (from system counter) when the GEOS.INI file was last written to.

Pass: Nothing.

Returns: **cx:dx** System counter time when the GEOS.INI file was last written to.

Destroyed: Nothing.

Library: **initfile.def**

■ InitFileGrab

Grab exclusive access on the initfile routines, and use the passed buffer as a temporary init file.

Pass: **ax** handle of memory block that will be used for init file reads/writes
bx file handle
cx size of file

Returns: CF Set on error; clear on success. Errors can occur when the init file contains non-ASCII characters or is not in a valid format.

Destroyed: Nothing.

Library: **initfile.def**

■ InitFileReadBoolean

Returns the Boolean value specified in the given category and key of the GEOS.INI file.

Pass: **ds:si** Category (null-terminated ASCII string) of data within the GEOS.INI file.

cx:dx Key (null-terminated ASCII string) of data within the GEOS.INI file.

Returns: CF Clear if successful.
ax (If CF = 0) ffffh = TRUE, 0 = FALSE.
 If CF != zero, then **ax** is unchanged.

Destroyed: Nothing.



Library: **initfile.def**

■ InitFileReadData

Locates the contents of the given category and key of the GEOS.INI file and returns a pointer to the associated data.

Pass:	ds:si	Category (null-terminated ASCII string) of data within the GEOS.INI file.
	cx:dx	Key (null-terminated ASCII string) of data within the GEOS.INI file.
	bp	InitFileReadFlags. (If IFRF_SIZE = 0 then a buffer will be allocated for the string, otherwise IFRF_SIZE should contain the size of the buffer and es:di will contain the address of the buffer to fill.)
	es:di	(If IFRF_SIZE is non-zero) Buffer to place string into.
Returns:	CF	Clear if successful.
	cx	Number of bytes retrieved (excluding null-terminator).
	bx	(If IFRF_SIZE = 0 was passed in bp) Memory handle to block containing entry; otherwise not defined.
	es:di	(If IFRF_SIZE was non-zero) Buffer filled.
Destroyed:	Nothing.	
Library:	initfile.def	

■ InitFileReadInteger

Returns the integer value specified in the given category and key of the GEOS.INI file.

Pass:	ds:si	Category (null-terminated ASCII string) of data within the GEOS.INI file.
	cx:dx	Key (null-terminated ASCII string) of data within the GEOS.INI file.
Returns:	CF	Clear if successful.
	ax	(If CF = 0) Integer value, otherwise unchanged.
Destroyed:	Nothing.	
Library:	initfile.def	

■ InitFileReadString

Locates the contents of the given category and key of the GEOS.INI file and returns a pointer to the associated string.

Pass:	ds:si	Category (null-terminated ASCII string) of data within the GEOS.INI file.
	cx:dx	Key (null-terminated ASCII string) of data within the GEOS.INI file.
	bp	InitFileReadFlags. (If IFRF_SIZE = 0 then a buffer will be allocated for the string, otherwise IFRF_SIZE should contain the size of the buffer and es:di will contain the address of the buffer to fill.)
	es:di	(If IFRF_SIZE is non-zero) Buffer to place string into.
Returns:	CF	Clear if successful.
	cx	Number of bytes retrieved (excluding null-terminator).
	bx	(If IFRF_SIZE = 0 was passed in bp) MemHandle to block containing entry; otherwise not defined.
	es:di	(if IFRF_SIZE was non-zero) Buffer filled.
Destroyed:	bx (if not returned).	
Library:	initfile.def	

■ InitFileReadStringSection

Locates the contents of the given category and key of the GEOS.INI file, copies a specified section of the string, and returns a pointer to the copied string section.

Pass:	ds:si	Category (null-terminated ASCII string) of data within the GEOS.INI file.
	cx:dx	Key (null-terminated ASCII string) of data within the GEOS.INI file.
	ax	Zero-based integer specifying the start of the string section "blob" to copy.
	bp	InitFileReadFlags. (If IFRF_SIZE = 0 then a buffer will be allocated for the string, otherwise IFRF_SIZE should contain the size of the buffer and es:di will contain the address of the buffer to fill.)
Returns:	es:di	(If IFRF_SIZE is non-zero) Buffer to place string into.
	CF	Clear if successful.
	cx	Number of bytes retrieved (excluding null-terminator).
	bx	(If IFRF_SIZE = 0 was passed in bp) Memory handle to block containing entry; otherwise not defined.
Destroyed:	es:di	(If IFRF_SIZE was non-zero) Buffer filled.
	Nothing.	
Library:	initfile.def	



InitFileRevert

■ 220

■ InitFileRevert

Restores the GEOS.INI to its backed-up previous state.

Pass: Nothing.

Returns: CF Clear if successful, set otherwise.

Destroyed: Nothing.

Library: **initfile.def**

■ InitFileSave

Saves the GEOS.INI file.

Pass: Nothing.

Returns: CF Clear if successful, set otherwise.

Destroyed: Nothing.

Library: **initfile.def**

■ InitFileWriteBoolean

Writes out a Boolean value to the GEOS.INI file.

Pass: **ds:si** Category (null-terminated ASCII string) to place the Boolean value within the GEOS.INI file.

cx:dx Key (null-terminated ASCII string) to place the Boolean value within the GEOS.INI file.

ax Boolean value. (Non-zero = TRUE, zero = FALSE.)

Returns: Nothing.

Destroyed: Nothing.

Library: **initfile.def**

■ InitFileWriteData

Writes out a string of data (which may represent a null-terminated text string, a Boolean value, or an integer value) to the GEOS.INI file. You may instead use **InitFileWriteString**, **InitFileWriteInteger**, or **InitFileWriteBoolean**.

Pass: **ds:si** Category (null-terminated ASCII string) to place the string of data within the GEOS.INI file.

cx:dx Key (null-terminated ASCII string) to place the string of data within the GEOS.INI file.

es:di Buffer containing the string of data to write out.

Assembly Reference

bp Size of buffer.
Returns: Nothing.
Destroyed: Nothing.
Library: **initfile.def**

■ InitFileWriteInteger

Writes out an integer to the GEOS.INI file.

Pass: **ds:si** Category (null-terminated ASCII string) to place the integer of data within the GEOS.INI file.
 cx:dx Key (null-terminated ASCII string) to place the integer of data within the GEOS.INI file.
 bp Integer value.
Returns: CF Clear if successful. Otherwise, set.
Destroyed: Nothing.
Library: **initfile.def**

■ InitFileWriteString

Writes out a string to the GEOS.INI file.

Pass: **ds:si** Category (null-terminated ASCII string) to place the string of data within the GEOS.INI file.
 cx:dx Key (null-terminated ASCII string) to place the string of data within the GEOS.INI file.
 es:di Body (null-terminated ASCII string) to write out to the category and key of the GEOS.INI file.
Returns: Nothing.
Destroyed: Nothing.
Library: **initfile.def**

■ InitFileWriteStringSection

Appends a string onto the end of a pre-existing GEOS.INI entry.

Pass: **ds:si** Category (null-terminated ASCII string) to place the string of data within the GEOS.INI file.
 cx:dx Key (null-terminated ASCII string) to place the string of data within the GEOS.INI file.
 es:di String (null-terminated ASCII string) to append onto the end of the category and key entries of the GEOS.INI file entry.



InkCompress

■ 222

Returns: Nothing.
Destroyed: Nothing.
Library: **initfile.def**

■ InkCompress

Compress ink data from a VisInk object.

Pass: **cx** Handle of block containing ink data (This will *not* be freed by **InkCompress**):
word numPoints
InkPoint
InkPoint
InkPoint
...
bx file in which to create DB Item
ax:di **DBItem** to hold data (pass 0:0 to create a new DBItem)
Returns: **ax:di** **DBItem** containing compressed ink data.
Destroyed: Nothing.
Library: **pen.def**

■ InkDBGetDisplayInfo

Returns the current folder handle, note ID if any is selected and the page number within the note.

Pass: **bx** File handle (or override).
Returns: **ax:di** Folder handle.
dx:cx Note ID.
bp Page.
Destroyed: Nothing.
Library: **pen.def**

■ InkDBGetHeadFolder

Returns the head (root) folder for the associated Ink DB file.

Pass: **bx** File handle (or override).
Returns: **ax:di** Folder handle.
Destroyed: Nothing.
Library: **pen.def**

Assembly Reference

■ InkDBInit

Creates and initializes a new DB file for use by the Ink object. This routine must be called before calling any other Ink Database functions.

Pass: **bx** Handle of file.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **pen.def**

■ InkDBSetDisplayInfo

Displays the contents of the passed Folder, Note, and Page display information, if applicable.

Pass: **bx** File handle (or override).
ax:di Folder handle.
dx:cx Note ID.
bp Page.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **pen.def**

■ InkDecompress

Uncompresses compressed ink data so that it may be loaded to a VisInk object.

Pass: **bx** File handle.
ax:di **DBItem** containing ink data.
 Returns: **bx** Block containing ink data or zero if out of memory.
 Destroyed: Nothing.
 Library: **pen.def**

■ InkFolderCreateSubFolder

Creates a new folder as a child of the passed folder.

Pass: **ax:di** Folder ID of parent folder (or null:null if no parent).
bx File handle.
 Returns: **ax:di** New child folder.
 Destroyed: Nothing.



InkFolderDelete

■ 224

Library: **pen.def**

■ InkFolderDelete

Deletes a folder. If the folder contains children, it recursively deletes all children.

Pass: **ax.di** Folder to delete.
 bx File handle (or override).

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkFolderDepthFirstTraverse

Performs a depth-first traversal of the folder tree, calling the passed routine with all encountered folders.

Pass: **ax.di** Folder at top of tree.
 bx File handle.
 cx:dx Callback routine (fptr).
 bp extra data to pass to callback routine.

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkFolderDisplayChildInList

Displays a note or folder's name in a GenDynamicList. This routine builds and sends a moniker to the passed list.

Pass: **ax.di** Folder ID.
 bx File handle.
 cx:dx Optr of GenDynamicList.
 bp Entry number of child we want to display in list.
 si Non zero if you want to display folders (if this is zero, then the entry number will be based only on notes).

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

Assembly Reference

■ InkFolderGetChildInfo

Returns information on a folder's child, specifying whether the child is a folder or a note along with the child's ID number.

Pass:	ax.di	Folder ID.
	bx	File handle.
	cx	Child number.
Returns:	CF	Set if folder, clear if note.
	ax.di	Folder or note ID.
Destroyed:	Nothing.	
Library:	pen.def	

■ InkFolderGetChildNumber

Returns the child's number given its folder or note ID.

Pass:	ax.di	Folder ID.
	bx	File handle.
	dx.cx	Note or subfolder.
Returns:	ax	Child number.
Destroyed:	Nothing.	
Library:	pen.def	

■ InkFolderGetContents

Returns a chunk array containing all of the folder's subfolders and a chunk array containing all the folder's child notes.

Pass:	bx	File handle.
	ax.di	Folder ID.
Returns:	di,ax	Item/group of chunk array of subfolders.
	cx,dx	Item/group of chunk array of notes.
Destroyed:	Nothing.	
Library:	pen.def	

■ InkFolderGetNumChildren

Returns the number of children within the passed folder.

Pass:	ax.di	Folder ID.
	bx	File handle.
Returns:	cx	Number of subfolders.



InkFolderMove

■ 226

dx Number of notes.
Destroyed: Nothing.
Library: **pen.def**

■ InkFolderMove

Moves a folder, attaching it below the passed parent folder.

Pass: **bx** VM file handle.
 ax.di Folder to move.
 cx.dx New parent folder.
Returns: Nothing.
Destroyed: Nothing.
Library: **pen.def**

■ InkFolderSetTitle

Sets the title of a folder to use the passed text string.

Pass: **ax.di** Folder ID.
 bx File handle.
 ds:si Null terminated text string.
Returns: Nothing.
Destroyed: Nothing.
Library: **pen.def**

■ InkFolderSetTitleFromTextObject

Sets the title of a folder from the entire text within the passed text object.

Pass: **ax.di** Folder ID.
 bx File handle.
 cx:dx Optr of text object.
Returns: Nothing.
Destroyed: Nothing.
Library: **pen.def**

■ InkGetDocCustomGString

Retrieves the custom GString field from the **InkDataFileMap** structure

Pass: **bx** File handle.
Returns: **ax** GString handle.

Assembly Reference

Destroyed: Nothing.

Library: **pen.def**

■ InkGetDocGString

Retrieves the background picture (in the form of a GString) of the Ink object. If this function returns a token indicating that the Ink object is using a custom GString, use **InkGetDocCustomGString**.

Pass: **bx** File handle.

Returns: **ax** GString handle.

Destroyed: Nothing.

Library: **pen.def**

■ InkGetDocPageInfo

Retrieves the current page information for the Ink database.

Pass: **ds:si** Pointer to hold the structure **PageSizeReport**.
bx File handle.

Returns: **ds:si** **PageSizeReport** structure filled in.

Destroyed: Nothing.

Library: **pen.def**

■ InkGetParentFolder

Returns the parent folder of the passed folder or note.

Pass: **ax.di** Note or folder to retrieve the parent of.
bx File handle.

Returns: **ax.di** Parent folder.

Destroyed: Nothing.

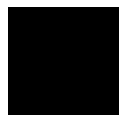
Library: **pen.def**

■ InkGetTitle

Returns the title of the passed folder or note.

Pass: **ax.di** Folder or note ID.
bx File handle (or override).
ds:si Destination buffer to place the title string.

Returns: **ds:si** Buffer filled in.
cx Length of name including null terminator.



InkNoteCopyMoniker

■ 228

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteCopyMoniker

Copies the icon and note name into a visMoniker structure. (May also be used to copy a folder's name into a visMoniker.)

Pass: **di.cx** Title of the note (or folder).
 bx:si Optr of output list.
 ax 1 if a text note; 0 if an ink note. (-1 if a folder.)
 dx Entry index.

Returns: Nothing.

Destroyed: **ax, bx, cx, dx, si, di**

Library: **pen.def**

■ InkNoteCreate

Creates a note below the passed folder.

Pass: **ax.di** Parent folder ID.
 bx File Handle.

Returns: **ax.di** Note ID.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteCreatePage

Creates a new page within a note.

Pass: **ax.di** Note ID.
 bx File handle (or override).
 cx Page number to insert new page (or -1 to append page at end).

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteDelete

Deletes the note and all references to it.

Pass: **ax.di** Note to delete.
 bx File handle.

Assembly Reference

Returns: Nothing.
 Destroyed: Nothing.
 Library: **pen.def**

■ InkNoteFindByKeywords

Retrieves a note by its associated keywords. Returns the first 20,000 or so notes that match the given keywords.

Pass: **ds:si** Pointer to keywords to match.
ax Non-zero if you only want notes that contain all passed keywords (exact match).
bx File handle.
 Returns: **dx** Handle of block containing Note IDs of notes matching or zero if no notes match. Block is set up in the following format:
FindNoteHeader<>
 Note ID
 Note ID
 Note ID etc.

Destroyed: Nothing.
 Library: **pen.def**

■ InkNoteFindByTitle

Retrieves a note by its associated title. Returns the first 20,000 or so notes that match the given title.

Pass: **ds:si** Pointer to string to match.
al **SearchOptions.**
ah Non-zero if we want to search the body.
bx File handle.
 Returns: **dx** Handle of block containing Note IDs of notes matching or zero if no notes match. Block is set up in the following format:
FindNoteHeader<>
 Note ID
 Note ID
 Note ID etc.

Destroyed: Nothing.
 Library: **pen.def**

■ InkNoteGetCreationDate

Returns the creation date of the passed note.



InkNoteGetKeywords

■ 230

Pass:	ax.di	Note.
	bx	File handle.
Returns:	cx	Creation year.
	dl	Creation month.
	dh	Creation day.
Destroyed:	Nothing.	
Library:	pen.def	

■ InkNoteGetKeywords

Copies the keyword string used by the passed note into the passed destination address.

Pass:	ax.di	Note ID.
	bx	File handle.
	ds:si	Destination for copied string.
Returns:	ds:si	Filled in.
Destroyed:	Nothing.	
Library:	pen.def	

■ InkNoteGetModificationDate

Returns the date the passed note was last modified.

Pass:	ax.di	Note ID.
	bx	File handle.
Returns:	cx	Modification year.
	dl	Modification month.
	dh	Modification day.
Destroyed:	Nothing.	
Library:	pen.def	

■ InkNoteGetNoteType

Returns the note type (ink or text) in use by the passed note.

Pass:	ax.di	Note ID.
	bx	File handle.
Returns:	cx	Note type. (0: ink, 2: text)
Destroyed:	Nothing.	
Library:	pen.def	

Assembly Reference

■ InkNoteGetNumPages

Returns the number of pages associated with the passed note.

Pass: **ax.di** Note ID.
 bx VM file handle.
Returns: **cx** Total number of pages in a note.
Destroyed: Nothing.
Library: **pen.def**

■ InkNoteGetPages

Returns the DB item in which the note's information is stored. This DB item contains a chunk array of pages.

Pass: **ax.di** Note ID.
 bx File handle.
Returns: **ax.di** DB item containing chunk array of pages.
Destroyed: Nothing.
Library: **pen.def**

■ InkNoteLoadPage

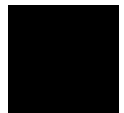
Loads an ink or text object from a page of a note.

Pass: **ax.di** Note ID.
 bx File handle.
 cx Page number.
 dx.bp Optr of ink or text object.
 si Note type (0: ink, 2: text).
Returns: Nothing.
Destroyed: Nothing.
Library: **pen.def**

■ InkNoteMove

Moves a note from one folder to another.

Pass: **ax.di** Note to move.
 dx.cx New parent folder.
 bx File handle.
Returns: Nothing.
Destroyed: Nothing.



Library: **pen.def**

■ InkNoteSavePage

Saves an ink or text object from the page of a note to the instance data of that object.

Pass: **ax.di** Note ID.
 bx File handle (or override).
 cx Page number.
 dx:bp Optr of ink or text object.
 si Note type (0: ink, 2: text)

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteSendKeywordsToTextObject

Replaces a text object's text with the passed note's keywords.

Pass: **ax.di** Note ID.
 bx File handle.
 cx:dx Optr of text object.

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteSetKeywords

Sets the keywords of the passed note using the passed text string.

Pass: **ax.di** Note ID.
 bx File handle.
 ds:si Pointer to text string.

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteSetKeywordsFromTextObject

Sets the keywords of the passed note using the text within a text object.

Pass: **ax.di** Note ID.
 bx File handle.

cx:dx Optr of text object.

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteSetModificationDate

Sets the modification date of the passed note. This allows you to update the note when writing changes.

Pass: **ax.di** Note ID.
 bx File handle.
 cx, dx Modification date.

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteSetNoteType

Sets the note type (ink or text) in use by the passed note.

Pass: **ax.di** Note ID.
 bx File handle.
 c1 Note type (0: ink, 2:text)

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteSetTitle

Sets the title in use by the passed note to the passed text string.

Pass: **ax.di** Note ID.
 bx File handle.
 ds:si Null-terminated text string.

Returns: Nothing.

Destroyed: Nothing.

Library: **pen.def**

■ InkNoteSetTitleFromTextObject

Sets the title in use by the passed note to the text within a text object.



InkSetTitleToTextObject

■ 234

Pass:	ax:di	Note ID.
	bx	File handle.
	cx:dx	Optr of text object.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	pen.def	

■ InkSetTitleToTextObject

Sets the text within a text object to the text within the passed note or folder's title.

Pass:	ax:di	Folder or note ID.
	bx	File handle
	cx:dx	Optr of text object.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	pen.def	

■ InkSetDocCustomGString

Sets the custom GString field (*IDFM_customGstring*) in an Ink object's **InkDataFileMap** field.

Pass:	bx	File handle.
	ax	GString handle.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	pen.def	

■ InkSetDocGString

Sets the GString field (*IDFM_gstring*) in an Ink object's **InkDataFileMap** field.

Pass:	bx	File handle.
	ax	GString handle.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	pen.def	

Assembly Reference

■ InkSetDocPageInfo

Sets the current page information in use by Ink database.

Pass: **ds:si** Pointer to buffer to hold a **PageSizeReport** structure.
 bx File handle.

Returns: **ds:si** **PageSizeReport** structure.

Destroyed: Nothing.

Library: **pen.def**

■ LMemAlloc

Allocates space (a new chunk) on the local-memory heap. This routine may resize the LMem block, moving it on the heap, and invalidating stored segment pointers to it.

Pass: **ds** Segment address of the heap.
 al Object flags (**ObjChunkFlags**) if allocating an object block.
 cx Amount of space to allocate.

Returns: **CF** Set if an error is encountered.
 ax Handle of the new chunk.
 ds Segment address of the same heap block.
 es Unchanged, unless **es** and **ds** were the same upon entry in which case they are the same on return.

Destroyed: Nothing.

Library: **lmem.def**

■ LMemContract

Compacts a local memory block. The local memory manager routines ordinarily take care of heap compaction. This routine compacts the heap manually and frees the unused heap space. The block is guaranteed to remain at the same address after using this routine (if the block is locked).

Pass: **ds** Segment address of block to compact.

Returns: Nothing.

Destroyed: Nothing.

Library: **lmem.def**

■ LMemDeleteAt

Deletes space from within the middle of a chunk on the local memory heap.



LMemFree

■ 236

Pass:	ds	Segment address of the local memory heap.
	ax	Chunk.
	bx	Offset to begin deletion of data within the LMem chunk.
	cx	Number of bytes to delete.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	lmem.def	

■ LMemFree

Frees the space occupied by a local-memory chunk. This routine does not resize the block or shuffle any other chunks.

Pass:	ax	Handle of chunk to free.
	ds	Segment address of local memory heap.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	lmem.def	

■ LMemInitHeap

Creates and initializes a local-memory heap block. This routine may resize the LMem block, moving it on the heap, and invalidating stored segment pointers to it. Where possible, you should try to use the higher-level routines: **MemAllocLMem**, **VMAallocLMem**, or **UserAllocObjBlock**.

Pass:	ds	Segment of memory block to use as the heap.
	bx	Handle of same memory block.
	ax	Type of heap to create (LMemType).
	cx	Number of handles to allocate initially.
	dx	Offset within segment to the start of the heap.
	si	Amount of free space to allocate initially.
	di	LocalMemoryFlags .
Returns:	ds	Segment of block passed (may have changed).
	es	Unchanged unless es and ds were the same upon entry in which case they are the same on return.
Destroyed:	Nothing.	
Library:	lmem.def	

Assembly Reference

■ **LMemInsertAt**

Inserts space within the middle of a chunk on the local memory heap. The new space is initialized to zeroes.

Pass:	ds	Segment address of the local memory heap.
	ax	Chunk.
	bx	Offset to insert space.
	cx	Number of bytes to insert.
Returns:	CF	Set if an error is encountered.
	ds	Segment of block passed (may have changed).
Destroyed:	Nothing.	
Library:	lmem.def	

■ **LMemReAlloc**

Changes the size of a chunk in a local memory heap.

Pass:	ax	Handle of chunk.
	cx	New size to resize the chunk to.
	ds	Segment address of the local memory heap.
Returns:	CF	Set if an error is encountered.
	ds	Segment of block passed (may have changed).
	es	Unchanged unless es and ds were the same upon entry in which case they are the same on return.
	ax	If LMem block has LMF_NO_HANDLES set, then this will be the chunk handle of the resized chunk.
Destroyed:	Nothing.	
Library:	lmem.def	

■ **LocalAsciiToFixed**

This routine converts the ASCII expression of a number to a **WWFixed** number.

Pass:	ds:di	String to evaluate (e.g. "12.345"). This routine does not handle exponents, and handles only four decimal digits.
Returns:	dx.ax	WWFixed value. The dx register holds the integer portion of the number, ax holds the fraction.
	di	Updated to point after last character parsed.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalCalcDaysInMonth

Return the number of days in the passed month/year.

Pass:	ax	Year
	bx	Month
Returns:	cx	Days in the month
Destroyed:	Nothing	
Library:	localize.def	

■ LocalCmpChars

This routine does a lexical comparison of two characters, determining which comes first in alphabetic order.

Pass:	ax	"Source" character.
	cx	"Dest" character.
Returns:	ZF	Set if characters were equal.
	CF	Set if "source" character less (earlier) than "destination".

if source = dest : if (z)
if source != dest : if !(z)
if source > dest : if !(c | z)
if source < dest : if (c)
if source >= dest : if !(c)
if source <= dest : if (c | z)

Destroyed: Nothing.

Library: **localize.def**

Warning: Don't use this routine if it would be more appropriate to use **LocalCmpStrings**. DBCS support requires special parsing of strings—you cannot compare them a character at a time.

■ LocalCmpCharsNoCase

This routine does a lexical comparison of two characters, determining which comes first in alphabetic order. It will ignore case.

Pass:	ax	"Source" character.
	cx	"Dest" character.
Returns:	ZF	Set if characters were equal.
	CF	Set if "source" character less (earlier) than "destination".

if source = dest : if (z)
if source != dest : if !(z)
if source > dest : if !(c | z)

```

if source < dest : if (c)
if source >= dest : if !(c)
if source <= dest : if (c | z)

```

Destroyed: Nothing.

Library: **localize.def**

Warning: Don't use this routine if it would be more appropriate to use **LocalCmpStrings**. DBCS support requires special parsing of strings—you cannot compare them a character at a time.

■ LocalCmpStrings

This routine does a lexical comparison of two text strings, determining which comes sooner in alphabetic order.

Pass: **ds:si** Pointer to string1.
es:di Pointer to string2.
cx Maximum number of characters to compare (0 for NULL terminated).

Returns: ZF Set if strings were equal.
CF Set if string1 less (earlier) than string2.
if string1 = string2 : if (z)
if string1 != string2 : if !(z)
if string1 > string2 : if !(c | z)
if string1 < string2 : if (c)
if string1 >= string2 : if !(c)
if string1 <= string2 : if (c | z)

Destroyed: Nothing.

Library: **localize.def**

■ LocalCmpStringsDosToGeos

Compares strings as does **LocalCmpStrings**, above, but one or both of the strings may be Dos text.

Pass: **ds:si** Pointer to string1.
es:di Pointer to string2.
cx Maximum number of characters to compare (0 for NULL terminated).
ax **LocalCmpStringsDosToGeosFlags** to specify which strings are GEOS, as opposed to Dos, strings.
bx Default character—when there is no GEOS equivalent for a Dos character, this character will be substituted in its place.

Returns: ZF Set if strings were equal.



LocalCmpStringsNoCase

■ 240

CF Set if string1 less (earlier) than string2.
 if string1 = string2 : if (z)
 if string1 != string2 : if !(z)
 if string1 > string2 : if !(c | z)
 if string1 < string2 : if (c)
 if string1 >= string2 : if !(c)
 if string1 <= string2 : if (c | z)

Destroyed: Nothing.

Library: **localize.def**

■ LocalCmpStringsNoCase

Compares strings as does **LocalCmpStrings**, except that case is ignored.

Pass: **ds:si** Pointer to string1.
 es:di Pointer to string2.
 cx Maximum number of characters to compare (0 for NULL terminated).

Returns: ZF Set if strings were equal.
 CF Set if string1 less (earlier) than string2.
 if string1 = string2 : if (z)
 if string1 != string2 : if !(z)
 if string1 > string2 : if !(c | z)
 if string1 < string2 : if (c)
 if string1 >= string2 : if !(c)
 if string1 <= string2 : if (c | z)

Destroyed: Nothing.

Library: **localize.def**

■ LocalCmpStringsNoSpace

Compares two text strings as does **LocalCmpStrings**, except that spaces are ignored.

Pass: **ds:si** Pointer to string1.
 es:di Pointer to string2.
 cx Maximum number of characters to compare (0 for NULL terminated). Note that this count does not include the spaces.

Returns: ZF Set if strings were equal.
 CF Set if string1 less (earlier) than string2.
 if string1 = string2 : if (z)
 if string1 != string2 : if !(z)
 if string1 > string2 : if !(c | z)

Assembly Reference

```

if string1 < string2 : if (c)
if string1 >= string2 : if !(c)
if string1 <= string2 : if (c | z)

```

Destroyed: Nothing.

Library: **localize.def**

■ LocalCmpStringsNoSpaceCase

Compares two text strings as does **LocalCmpStrings**, except that spaces and case are ignored.

Pass: **ds:si** Pointer to string1.
 es:di Pointer to string2.
 cx Maximum number of characters to compare (0 for NULL terminated). Note that this count does not include the spaces.

Returns: ZF Set if strings were equal.
 CF Set if string1 less (earlier) than string2.
 if string1 = string2 : if (z)
 if string1 != string2 : if !(z)
 if string1 > string2 : if !(c | z)
 if string1 < string2 : if (c)
 if string1 >= string2 : if !(c)
 if string1 <= string2 : if (c | z)

Destroyed: Nothing.

Library: **localize.def**

■ LocalCodePageToGeos

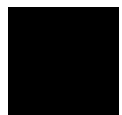
Converts Dos text to GEOS text. The Dos text may use any code page.

Pass: **ds:si** Pointer to text string.
 cx Maximum number of characters to convert (zero for a null-terminated string).
 bx Code page to use.
 ax Default character—when there is no GEOS equivalent for a code page character, this character will substitute.

Returns: CF Set if had to use the default character.
 ds:si Pointer to string converted to GEOS text.

Destroyed: Nothing.

Library: **localize.def**



■ LocalCodePageToGeosChar

Convert a single Dos character to GEOS text. The character may be in any Dos code page.

Pass:	ax	Character to map.
	bx	Default character—when there is no GEOS equivalent for a code page character, this character will be returned.
	cx	Code page to use.
Returns:	ax	Mapped character.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalCustomFormatDateTime

Format a date or time. If you call this directly then you will not be language independent. This routine is intended to be used by applications who do not wish to be language independent or by the higher level date formatting routines.

You only need valid information in the registers which will actually be referenced. I.e., if your format string has no month tokens, then the **bx** register will be ignored.

Pass:	ds:si	Format string.
	es:di	Buffer to save formatted text in.
	ax	Year (0-9999).
	bx	Month (1-12).
	cx	Date (1-31).
	dx	Day(0-6).
	si	Hours(0-23).
	di	Minutes(0-59).
	bp	Seconds(0-59).
Returns:	es:di	(Unchanged) pointer to buffer of formatted text.
	cx	Number of characters in formatted string.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalCustomParseDateTime

Parse a text string using a specific format string.

Any field for which there is not data specified will return containing -1.

Pass:	es:di	Pointer to string to parse.
-------	--------------	-----------------------------

Assembly Reference

	ds:si	Format string to compare against.
Returns:	CF	Set if valid date/time (if we were able to parse).
	ax	Year.
	bl	Month.
	bh	Date (1-31).
	cl	Day (0-6). (If we weren't able to parse, cx will be offset to start of the text that didn't match.)
	ch	Hours. (If we weren't able to parse, cx will be offset to start of the text that didn't match.)
	dl	Minutes.
	dh	Seconds.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalDistanceFromAscii

This routine extracts a distance value from an ASCII text string.

Pass:	ds:di	ASCII string to convert.
	cl	DistanceUnit.
	ch	MeasurementType.
Returns:	dx.ax	Value (zero if illegal).
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalDistanceToAscii

Construct an ASCII string from a distance value.

Pass:	es:di	Buffer to hold ASCII string (must be at least LOCAL_DISTANCE_BUFFER_SIZE).
	dx.ax	Value to convert.
	cl	DistanceUnit.
	ch	MeasurementType.
	bx	LocalDistanceFlags.
Returns:	cx	Length of string, including NULL.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalDosToGeos

This routine converts a Dos text string to GEOS text format.



LocalDosToGeosChar

■ 244

Pass:	ds:si	Pointer to text.
	cx	Maximum number of characters to convert (zero if string is NULL-terminated).
	ax	Default character. When there is no GEOS equivalent for a code page character, this character will substitute.
Returns:	CF	Set if default character was used.
	ds:si	(Unchanged) pointer to converted text.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalDosToGeosChar

This routine converts a single Dos character to a GEOS character.

Pass:	ax	Character to map.
	bx	Default character. (When there is no Geos equivalent for a code page character, this character will substitute.)
Returns:	ax	Mapped character.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalDowncaseChar

This routine returns the lowercase equivalent of any character (if the character has no lower-case equivalent, the character will be returned untouched).

Pass:	ax	Character to downcase.
Returns:	ax	Downcased character.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalDowncaseString

This routine converts a string to all lower case.

Pass:	ds:si	Pointer to string.
	cx	Maximum number of characters to convert (or zero for a null-terminated string).
Returns:	ds:si	(Unchanged) pointer to converted string.
Destroyed:	Nothing.	
Library:	localize.def	

Assembly Reference

■ LocalFixedToAscii

This routine converts a WWFixed number to its ASCII string equivalent.

Pass:	es:di	Buffer to hold result.
	dx:ax	Number to convert.
	cx	Number of digits of fraction.
Returns:	es:di	(Unchanged) pointer to buffer holding string.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalFormatDateTime

This routine takes a date or time and a enumerated type describing how that time should be formatted and returns a text string containing the time/date information formatted nicely.

Pass:	es:di	Pointer to buffer in which to place the formatted text.
	si	DateTimeFormat.
	ax	Year.
	bl	Month.
	bh	Date (1-31).
	cl	Day (0-6).
	ch	Hours.
	dl	Minutes.
	dh	Seconds.
Returns:	es:di	(Unchanged) pointer to buffer containing string.
	cx	Number of characters in the formatted string. This does not include the NULL character at the end of the string.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalFormatFileDateTime

Like LocalFormatDateTime, except it works off a FileDate and a FileTime record

Pass:	ax	FileDate
	bx	FileTime
	si	DateTimeFormat
	es:di	Buffer into which to format
Returns:	cx	Number of characters in formatted string, not including null terminator.

Destroyed: Nothing.

Library: **localize.def**

■ LocalGeosToCodePage

Convert GEOS text string to Dos text using an arbitrary code page.

Pass:	ds:si	Pointer to text string.
	cx	Maximum number of characters to convert (zero for a null-terminated string).
	bx	Code page to use.
	ax	Default character—when there is no code page equivalent for a GEOS character, this character will substitute.
Returns:	CF	Set if had to use the default character.
	ds:si	(Unchanged) pointer to text string converted to Dos text.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalGeosToCodePageChar

Convert one character of GEOS text to Dos text using an arbitrary code page.

Pass:	ax	Character to convert.
	cx	Code page to use.
	bx	Default character—when there is no code page equivalent for a GEOS character, this character will be returned.
Returns:	ax	Converted character.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalGeosToDos

Convert a string of GEOS text to its Dos equivalent.

Pass:	ds:si	Pointer to text.
	cx	Maximum number of characters to convert (zero for null-terminated).
	ax	Default character—when there is no code page equivalent for a GEOS character, this character will be returned.
Returns:	CF	Set if default character was used.
	ds:si	(Unchanged) pointer to converted text.
Destroyed:	Nothing.	

Library: **localize.def**

■ LocalGeosToDosChar

Convert one character of GEOS text to its Dos text equivalent.

Pass: **ax** Character to convert.
cx Code page to use.
bx Default character—when there is no code page equivalent for a GEOS character, this character will be returned.

Returns: **ax** Converted character.

Destroyed: Nothing.

Library: **localize.def**

■ LocalGetCodePage

This routine returns the value of the current code page.

Pass: Nothing.

Returns: **bx** **DosCodePage**.

Destroyed: Nothing.

Library: **localize.def**

■ LocalGetCurrencyFormat

This routine returns the information necessary to format currency text strings in the way preferred by the user.

Pass: **es:di** Pointer to buffer in which to put currency symbol.

Returns: **al** CurrencyFormatFlags.
ah Currency digits
bx Thousands separator (e.g. “,”).
cx Decimal separator (e.g. “.”).
dx List separator (e.g. “;”).
es:di (Unchanged) pointer to buffer filled with currency symbol.

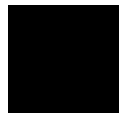
Destroyed: Nothing.

Library: **localize.def**

■ LocalGetDateTimeFormat

Returns the text string associated with a **TimeDateFormat**.

Pass: **es:di** Pointer to buffer to hold format string. Should be prepared to hold string up to DATE_TIME_BUFFER_SIZE.



LocalGetNumericFormat

■ 248

	si	DateTimeFormat in use by the format string.
Returns:	es:di	(Unchanged) pointer to buffer filled with format string.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalGetNumericFormat

This routine returns the information necessary to format currency text strings in the way preferred by the user.

Pass:	Nothing.	
Returns:	al	CurrencyFormatFlags.
	ah	Decimal digits.
	bx	Thousands separator (e.g. “,”).
	cx	Decimal separator (e.g. “.”).
	dx	List separator (e.g. “;”).
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalGetQuotes

This routine returns the localized symbols to use as single or double quotes.

Pass:	Nothing.	
Returns:	ax	Front single quote.
	bx	End single quote.
	cx	Front double quote.
	dx	End double quote.
Destroyed:	Nothing.	
Library:	localize.def	

■ LocalIsAlpha

This routine detects alphabetic characters.

Pass:	ax	Character to check.
Returns:	ZF	Clear if character is alphabetic.
Library:	localize.def	

■ LocalIsAlphaNumeric

This routine detects alphabetic and numeric characters.

Assembly Reference

Pass: **ax** Character to check.
 Returns: ZF Clear if character is alphanumeric.
 Library: **localize.def**

■ LocalIsCodePageSupported

Checks to see if the passed code page is a supported one.

Pass: **ax** Code page to check.
 Returns: ZF Set if supported; clear if not supported.
 Destroyed: Nothing.
 Library: **localize.def**

■ LocalIsControl

This routine detects control characters.

Pass: **ax** Character to check.
 Returns: ZF Clear if character is a control character.
 Library: **localize.def**

■ LocalIsDateChar

This routine detects date characters.

Pass: **ax** Character to check.
 Returns: ZF Clear if character is a digit, or part of the format string associated with DTF_SHORT.
 Library: **localize.def**

■ LocalIsDigit

This routine detects numeric characters.

Pass: **ax** Character to check.
 Returns: ZF Clear if character is numeric.
 Library: **localize.def**

■ LocalIsDosChar

This routine detects characters which are members of the Dos character set.

Pass: **ax** Character to check.
 Returns: ZF Clear if character is in the Dos character set.



Library: **localize.def**

■ LocallsGraphic

This routine detects characters require some sort of drawing. Control characters are not graphic; neither are line-feeds or spaces. Letters, numbers, and punctuation marks are all good examples of graphic symbols.

Pass: **ax** Character to check.

Returns: ZF Clear if character is graphic.

Library: **localize.def**

■ LocallsHexDigit

This routine detects numeric characters, including those letters necessary for expressing hexadecimal numbers.

Pass: **ax** Character to check.

Returns: ZF Clear if character is a hexadecimal digit.

Library: **localize.def**

■ LocallsLower

This routine detects lower-case alphabetic characters.

Pass: **ax** Character to check.

Returns: ZF Clear if character is lower-case.

Library: **localize.def**

■ LocallsNumChar

This routine detects numeric characters, including the decimal separator.

Pass: **ax** Character to check.

Returns: ZF Clear if character is part of the number format.

Library: **localize.def**

■ LocallsPrintable

This routine detects printable characters (this includes all graphic characters and the space character).

Pass: **ax** Character to check.

Returns: ZF Clear if character is printable.

Library: **localize.def**

■ LocallsPunctuation

This routine detects punctuation marks.

Pass: **ax** Character to check.
Returns: ZF Clear if character is a punctuation mark.
Library: **localize.def**

■ LocallsSpace

This routine detects white-space.

Pass: **ax** Character to check.
Returns: ZF Clear if character is a space.
Library: **localize.def**

■ LocallsSymbol

This routine detects symbol characters.

Pass: **ax** Character to check.
Returns: ZF Clear if character is a symbol.
Library: **localize.def**

■ LocallsTimeChar

This routine detects characters which are digits or part of the format string associated with DTF_HMS.

Pass: **ax** Character to check.
Returns: ZF Clear if character is part of a time string.
Library: **localize.def**

■ LocallsUpper

This routine detects upper-case alphabetic characters.

Pass: **ax** Character to check.
Returns: ZF Clear if character is upper-case.
Library: **localize.def**

■ LocalLexicalValue

This routine returns the lexical value of a character, useful when sorting things into alphabetical order. Note that knowing the lexical value of just one



LocalLexicalValueNoCase

■ 252

character is not much use—lexical values are only meaningful when compared to one another.

Pass: **ax** Character.
Returns: **ax** Lexical order.
Destroyed: Nothing.
Library: **localize.def**

■ LocalLexicalValueNoCase

This routine returns the case-insensitive lexical value of a character, useful when sorting things into alphabetical order. Note that knowing the case-insensitive lexical value of just one character is not much use—lexical values are only meaningful when compared to one another.

Pass: **ax** Character.
Returns: **ax** Case-insensitive lexical order.
Destroyed: Nothing.
Library: **localize.def**

■ LocalParseDateTime

This routine parses a text string and extracts time or date information from it. Any field for which there is not data specified in the format string is returned containing -1.

Pass: **es:di** Pointer to the string to parse.
 si DateTimeFormat with which to parse string.
Returns: **CF** Set if string is a valid date/time (i.e. if the string parsed correctly).
 ax Year.
 bl Month.
 bh Date (1-31).
 cl Day(0-6). (If string did not parse correctly, **cx** will be the offset to the start of the text that didn't match.)
 ch Hours. (If string did not parse correctly, **cx** will be the offset to the start of the text that didn't match.)
 dl Minutes.
 dh Seconds.
Destroyed: Nothing.
Library: **localize.def**

Assembly Reference

■ LocalSetCurrencyFormat

This routine sets the currency format information. All items will be added to the .INI file.

Pass: **al** CurrencyFormatFlags.
 ah Currency digits.
 es:di Pointer to string containing currency symbol.

Returns: Nothing.

Destroyed: Nothing.

Library: **localize.def**

■ LocalSetDateTimeFormat

Sets the localization date/time format in use by a **DateTimeFormatString**.

Pass: **es:di** Pointer to format string (of type **DateTimeFormatString**).
 si New **DateTimeFormat**.

Returns: Nothing.

Destroyed: Nothing.

Library: **localize.def**

■ LocalSetMeasurementType

This routine sets the measurement type. The correct value will be written to the .INI file.

Pass: **al** MeasurementType.

Returns: Nothing.

Destroyed: Nothing.

Library: **localize.def**

■ LocalSetNumericFormat

This routine sets the fields used when formatting number strings. The correct values will be written to the .INI file.

Pass: **al** NumberFormatFlags.
 ah Decimal digits.
 bx Thousands separator (e.g. “,”).
 cx Decimal separator (e.g. “.”).
 dx List separator (e.g. “;”)

Returns: Nothing.



LocalSetQuotes

■ 254

Destroyed: Nothing.

Library: **localize.def**

■ LocalSetQuotes

This routine sets the localized single and double quotes.

Pass:	ax	Front single quote.
	bx	End single quote.
	cx	Front double quote.
	dx	End double quote.

Returns: Nothing.

Destroyed: Nothing.

Library: **localize.def**

■ LocalStringLength

This routine computes the number of characters in a null-terminated text string. This routine allows for double byte character support.

Pass: **es:di** Pointer to string.

Returns: **cx** Number of characters in the string, not including the null.

Destroyed: Nothing.

Library: **localize.def**

■ LocalStringSize

This routine determines the number of bytes used to store a null-terminated text string.

Pass: **es:di** Pointer to string.

Returns: **cx** Number of bytes in the string (not counting the NULL).

Destroyed: Nothing.

Library: **localize.def**

■ LocalUppcaseChar

This routine returns a character's upper-case equivalent.

Pass: **ax** Character

Returns: **ax** Upper-case character.

Destroyed: Nothing.

Assembly Reference

Library: **localize.def**

■ LocalUpcaseString

This routine returns the all-caps equivalent of a text string.

Pass: **ds:si** Pointer to string.
 cx Maximum number of characters to convert (zero for NULL terminated).

Returns: **ds:si** (Unchanged) pointer to upcased string.

Destroyed: Nothing.

Library: **localize.def**



LocalUpcaseString

■ 256

■ **Assembly Reference**

■ MemAlloc

Creates a block and assigns a handle to it. This block can be discardable, swappable, fixed or movable. A passed flag determines whether heap compaction or discarding of objects should be used to generate free space.

Pass:	ax	Size (in bytes) to allocate.
	c1	HeapFlags record.
	ch	HeapAllocFlags record.
Returns:	CF	Set if error (not enough memory).
	bx	Handle to block allocated.
	ax	Address of block allocated (if block is fixed or locked).
Destroyed:	cx	
Library:	heap.def	

■ MemAllocLMem

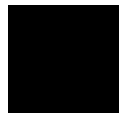
Allocates a block within a local memory heap from scratch. To take an existing block and create a local memory heap built on it, use **LMemInitHeap**.

Pass:	ax	Type of heap (LMemType).
	cx	Size of block header (or 0 for the default).
Returns:	bx	Block handle of the new block. This block will have two handles allocated and 64 bytes heap space.
Destroyed:	Nothing.	
Library:	lmem.def	

■ MemAllocSetOwner

Creates a block, assigns a handle to it, and explicitly sets the owner of the new block by passing the handle of the owning geode. Otherwise, it is identical to **MemAlloc**.

Pass:	ax	Size (in bytes) to allocate.
	c1	HeapFlags record.
	ch	HeapAllocFlags record.
	bx	New owner of block.
Returns:	CF	Set if error (not enough memory).
	bx	Handle to block allocated.
	ax	Address of block allocated (if block is fixed or locked).



MemDecRefCount

■ 258

Destroyed: **cx**

Library: **heap.def**

■ MemDecRefCount

Decrements a reference count in a memory handle and immediately frees it when the count reaches zero.

Pass: **bx** Memory handle. If zero, this routine will do nothing;

Returns: Non-EC: Nothing.
EC: **bx** cleared if reference count drops to zero (and block was freed).

Destroyed: Nothing.

Library: **heap.def**

■ MemDerefDS

Returns the address of the block referenced by its block handle into the DS register. The block must be locked before calling this routine. This routine is useful in allocating a fixed or locked block.

Pass: **bx** Block handle.

Returns: **ds** Segment of handle.

Destroyed: Nothing. Flags preserved.

Library: **heap.def**

■ MemDerefES

Returns the address of a block referenced by its block handle into the ES register. the block must be locked before calling this routine.

Pass: **bx** Block handle.

Returns: **es** Segment of handle.

Destroyed: Nothing. Flags preserved.

Library: **heap.def**

■ MemDiscard

Throws away the contents of a discardable memory block.

Pass: **bx** Handle of memory block.

Assembly Reference

Returns: **CF** Set if block couldn't be discarded. (This may happen if the block being discarded is a VM block, for example, and some other thread is using the file.)

Destroyed: **bx**

Library: **heap.def**

■ MemDowngradeExclLock

Downgrades an exclusive lock to a shared lock, waking any shared lockers blocked on the block.

Pass: **bx** Block handle whose exclusive lock should be downgraded.

Returns: **ax** Segment of block.

Destroyed: Nothing.

Library: **heap.def**

■ MemFree

Frees a memory block. The block may be locked at the time it is freed. Therefore, make sure that no other thread has locked the block before freeing it.

Pass: **bx** Handle of block to free.

Returns: Nothing.

Destroyed: **bx**

Library: **heap.def**

■ MemGetInfo

Returns information about a memory block. Pass this routine the proper **MemGetInfoType**. The return values depend on what information you request.

Pass: **ax** **MemGetInfoType**.
bx Handle of block.

Returns: **ax** Value based on the **MemGetInfoType** passed:
MGIT_SIZE **ax** = size of block in bytes.
MGIT_FLAGS_AND_LOCK_COUNT **al** = **HeapFlags**.
ah = lock count.
MGIT_OWNER_OR_VM_FILE_HANDLE **ax** = handle of owner.



MemIncRefCount

■ 260

MGIT_ADDRESS

ax = segment of block.

MGIT_OTHER_INFO

ax = contents of *HM_otherInfo* field.

MGIT_EXEC_THREAD

ax = handle of thread.

Destroyed: Nothing.

Library: **heap.def**

■ MemIncRefCount

Increments a reference count on a memory handle.

Pass: **bx** Memory handle. If zero, will do nothing.

Returns: Nothing.

Destroyed: Nothing.

Library: **heap.def**

■ MemInitRefCount

Initializes a reference count for a memory handle.

Pass: **bx** Memory handle.
ax Initial reference count (0 is not allowed).

Returns: Nothing.

Destroyed: Nothing.

Library: **heap.def**

■ MemLock

Locks the given block, returning the absolute address of the block of memory pointed to by the given handle. Increments the lock count by one. Locked memory blocks cannot be moved or discarded (though they may be freed). You cannot give a block a lock count greater than 255.

Pass: **bx** Handle to block of memory.

Returns: **CF** Set on error (Block is discarded).
ax Segment address of block of memory.

Destroyed: Nothing.

Library: **heap.def**

Assembly Reference

■ **MemLockExcl**

Locks a memory block for exclusive read/write access. If no one has locked the block, the thread will gain exclusive access; otherwise, the thread will block and the request will go on the queue. Use **MemUnlockExcl** to unlock the block for exclusive access.

Pass: **bx** Block to lock.
 Returns: **ax** Segment address of locked block.
 Destroyed: Nothing.
 Library: **heap.def**

■ **MemLockFixedOrMovable**

Given a virtual segment, locks the corresponding block down if the segment is movable.

Pass: **bx** Virtual segment.
 Returns: **CF** Set if block is discarded and can't be reloaded; clear if block is OK:
 ax Segment if block is OK; zero otherwise.
 Destroyed: Nothing.
 Library: **heap.def**

■ **MemLockShared**

Locks a block down for shared (usually read-only) access.

Pass: **bx** Handle of block to be locked.
 Returns: **ax** Segment of locked block.
 Destroyed: Nothing.
 Library: **heap.def**

■ **MemModifyFlags**

Modify the *HM_flags* associated with a memory block. The following bits may be altered: HF_SHARABLE, HF_DISCARDABLE, HF_SWAPABLE, and HF_LMEM.

Pass: **bx** Handle of block to modify.
 al Flags to set in the *HM_flags* field.
 ah Flags to clear in the *HM_flags* field.
 Returns: Nothing.



MemModifyOtherInfo

■ 262

Destroyed: **ax**

Library: **heap.def**

■ MemModifyOtherInfo

Modifies the *HM_otherInfo* field associated with a memory block.

Pass: **bx** Handle of block to modify.
 ax New *HM_otherInfo* value.

Returns: Nothing.

Destroyed: **ax**

Library: **heap.def**

■ MemOwner

Returns the owner field of a handle.

Pass: **bx** Handle.

Returns: **bx** Owner. if the block is owned by a VM file, the process that owns the VM file is returned.

Destroyed: Nothing.

Library: **heap.def**

■ MemPLock

Calls **HandleP** and **MemLock** in that order.

Pass: **bx** Handle of memory block.

Returns: **ax** Segment address of block.

Destroyed: Nothing.

Library: **heap.def**

■ MemReAlloc

Reallocates space within a given block, changing its size. Also used to reallocate space for a block that has been discarded.

Pass: **ax** Size (in bytes) to allocate. Pass zero to allocate the block with the same size.

bx Handle of block.
 ch **HeapAllocFlags**.

Returns: **CF** Set if error (not enough memory).
 bx Handle of block (unchanged).

Assembly Reference

ax Segment address of block if HF_LOCK was passed in the **HeapAllocFlags**.

Destroyed: **ax, cx**

Library: **heap.def**

■ MemSegmentToHandle

Returns the corresponding handle of a passed segment value.

Pass: **cx** Segment address.

Returns: **CF** Set if a matching handle is found.
cx Handle.

Returns: Nothing.

Library: **heap.def**

■ MemThreadGrab

Locks a block and increments a semaphore on the block; the current thread will be able to perform more **MemThreadGrab** operations but other threads will block on **MemThreadGrab**. If another thread has the semaphore **MemThreadGrab** blocks until it can get the semaphore; it then increments the semaphore, locks the block, and returns the address.

Pass: **bx** Handle of block.

Returns: **CF** Set if block has been discarded; clear if block is resident.
ax If CF is set, **ax** = segment address of block of memory.
If CF is clear, **ax** = 0.

Destroyed: Nothing.

Library: **heap.def**

■ MemThreadGrabNB

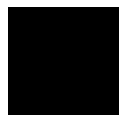
Performs the same function as MemThreadGrab (locks a block and increments a semaphore on the block) except that it doesn't block if another thread has the semaphore.

Pass: **bx** Handle of block.

Returns: **CF** Set if block has been discarded; clear if block is resident.
ax If CF is set, **ax** = segment address of block of memory.
If CF is clear, **ax** = 0.

Destroyed: Nothing.

Library: **heap.def**



MemThreadRelease

■ 264

■ MemThreadRelease

Releases a thread previously grabbed by **MemThreadGrab**.

Pass: **bx** Handle of block.

Returns: Nothing.

Destroyed: Nothing.

Library: **heap.def**

■ MemUnlock

Unlocks a given block of memory by decrementing its lock count. If the lock count reaches zero then the block is subject to moving or discarding.

Pass: **bx** Handle of block.

Returns: Nothing.

Destroyed: Nothing (flags preserved).
(If EC checking):

Possibly **ds** or **es** will be destroyed. (If segment error-checking is on and either **ds** or **es** is pointing to a block that has become unlocked, then that register will be set to **NULL_SEGMENT** upon return from this procedure.)

Library: **heap.def**

■ MemUnlockFixedOrMovable

Given a virtual segment, unlocks the corresponding block if the segment is movable.

Pass: **bx** Virtual segment.

Returns: Nothing.

Destroyed: Nothing (flags preserved).

Library: **heap.def**

■ MemUnlockShared

Unlocks a block that was locked by either **MemLockShared** or **MemLockExcl**.

Pass: **bx** Handle to be unlocked.

Returns: Nothing.

Destroyed: Nothing (flags preserved).

Assembly Reference

Library: **heap.def**

■ MemUnlockV

Unlocks a block and releases a semaphore on that block. (Performs a **MemUnlock** and a **HandleV** in that order.)

Pass: **bx** Handle of memory block.

Returns: **bx** Same handle.

Destroyed: Nothing (flags preserved).

Library: **heap.def**

■ MemUpgradeSharedLock

Upgrades a shared lock to an exclusive lock. If the block is shared by other threads this will block and the memory block may move on the heap before gaining exclusive access.

Pass: **bx** Handle (locked and shared) whose lock should be upgraded. The handle must not be locked more than once by the thread calling this routine

Returns: **ds, es** Fixed up to possibly new block location if they were pointing to the block upon entry.
ax Position of locked block.

Destroyed: Nothing.

Library: **heap.def**

■ MessageDispatch

Dispatches a passed message without destroying the handle of the message (event) to dispatch using **ObjMessage**.

Pass: **bx** Handle of message (event).
di **MessageFlags**. MF_RECORD in this instance means to dispatch but do not destroy the message.

Returns: **ax, cx, dx, bp**
As in **ObjMessage**.

Destroyed: Nothing.

Library: **object.def**

■ MessageProcess

Processes a message (event) dispatching it via a custom callback routine.



MessageSetDestination

■ 266

Pass:	bx di si	Handle of message (event) to dispatch. Data to pass to callback routine, if any. Non-zero to preserve event. Zero to destroy it.
Pass on stack:	fptr	Address of callback routine. (Segment is pushed first).
Callback Routine Specifications:		
Passed:	(Same as in ObjMessage except that di is passed through from caller.)	
	CF	Set if event has stack data.
	ss:[sp+4]	Calling thread. (This address is directly above the return address.)
Return:	Nothing.	
May Destroy:	ax, cx, dx, si, di, bp, ds, es	
Returns:	ax, cx, dx, bp	As returned by callback routine.
Destroyed:	Nothing.	
Library:	object.def	

■ MessageSetDestination

Changes an event's destination optr.

Pass:	bx cx:si	Event handle (message). destination.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	object.def	

■ MetaGrabFocusExclLow

This routine grabs the focus exclusive for the object.

Pass:	*ds:si	Object instance data.
Returns:	ds	Updated to point at segment of same block as on entry.
Destroyed:	Nothing.	
Library:	uiInputC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

Assembly Reference

■ **MetaGrabModelExclLow**

This routine grabs the model exclusive for the object.

Pass: ***ds:si** Object instance data.
Returns: **ds** Updated to point at segment of same block as on entry.
Destroyed: Nothing.
Library: **uiInputC.def**
Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ **MetaGrabTargetExclLow**

This routine grabs the target exclusive for the object.

Pass: ***ds:si** Object instance data.
Returns: **ds** Updated to point at segment of same block as on entry.
Destroyed: Nothing.
Library: **uiInputC.def**
Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ **MetaReleaseFocusExclLow**

This routine releases the focus exclusive for the object.

Pass: ***ds:si** Object instance data.
Returns: **ds** Updated to point at segment of same block as on entry.
Destroyed: Nothing.
Library: **uiInputC.def**
Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ **MetaReleaseFTExclLow**

This routine releases the focus and target exclusives for the object.

Pass: ***ds:si** Object instance data.
Returns: **ds** Updated to point at segment of same block as on entry.
Destroyed: Nothing.
Library: **uiInputC.def**



MetaReleaseModelExclLow

■ 268

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ MetaReleaseModelExclLow

This routine releases the model exclusive for the object.

Pass: ***ds:si** Object instance data.

Returns: **ds** Updated to point at segment of same block as on entry.

Destroyed: Nothing.

Library: **uiInputC.def**

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ MetaReleaseTargetExclLow

This routine releases the target exclusive for the object.

Pass: ***ds:si** Object instance data.

Returns: **ds** Updated to point at segment of same block as on entry.

Destroyed: Nothing.

Library: **uiInputC.def**

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.

■ NameArrayAdd

Creates an element within a name array and copies the data and name into it. If an element with the same name already exists, this routine will not create a duplicate; if NAAF_SET_DATA_ON_REPLACE is passed, the newly added element will replace the duplicate element.

This routine may resize the LMem block, moving it on the heap and invalidating stored segment pointers to it.

Pass: ***ds:si** Name array.
es:di Name to add.
cx Length of name (0 for null-terminated).
bx **NameArrayAddFlags.**
dx:ax Data for element.

Returns: **CF** Set if name added.
ax Name token.

Destroyed: Nothing.

Assembly Reference

Library: **chunkarr.def**■ **NameArrayChangeName**

Changes the element's name within the passed name array.

This routine may resize the LMem block, moving it on the heap and invalidating stored segment pointers to it.

Pass: ***ds:si** Name array.
 ax Name token.
 es:di New name.
 cx Length of name (0 for null-terminated).

Returns: Nothing.

Destroyed: Nothing.

Library: **chunkarr.def**■ **NameArrayCreate**Creates a name array with zero elements. Add elements with **NameArrayAdd**.

This routine may resize the LMem block, moving it on the heap and invalidating stored segment pointers to it.

Pass: **ds** Block for new array.
 bx Data size for each element.
 cx Size for **ChunkArrayHeader**. Default size is zero.
 si ChunkHandle to use (or 0 if you want to allocate one).
 al **ObjChunkFlags** to pass to **LMemAlloc**.

Returns: ***ds:si** Name array.

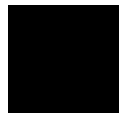
Destroyed: Nothing.

Library: **chunkarr.def**■ **NameArrayFind**

Finds a name element within a name array.

Pass: ***ds:si** Name array.
 es:di Name to find.
 cx Length of name (0 for null-terminated).
 dx:ax Buffer to return data (or zero to not return data).

Returns: **ax** Name token. (CA_NULL_ELEMENT if not found).
 CF Set if name found.



ObjBlockGetOutput

■ 270

Destroyed: Nothing.

Library: **chunkarr.def**

■ ObjBlockGetOutput

Returns the optr of the object set to receive output messages from all objects within the object block. (This output optr is stored in an object block's **ObjLMemBlockHeader**.)

Pass: **ds** Object block.

Returns: **bx:si** Output optr.

Destroyed: Nothing.

Library: **object.def**

■ ObjBlockSetOutput

Sets the optr of the object set to receive output messages from all objects within the object block. (This output optr is stored in an object block's **ObjLMemBlockHeader**.)

Pass: **ds** Object block.

bx:si Output optr.

Returns: Nothing.

Destroyed: Nothing.

Library: **object.def**

■ ObjCallClassNoLock

Sends a message (invokes a method) of the given class for an object. If the message is not resident in the passed class, sends that message on to the class' superclass.

This routine assumes that the object block containing the receiving object is already locked down and is being run by the same process. It is the responsibility of the caller to ensure that this is the case.

Pass: **ax** Message.

cx, dx, bp Other data to pass.

***ds:si** Instance data of object in question; if the receiving object is a process **ds** is the core block of the process. **ds** must be pointing to an object block, another kind of local memory block, or a core block; **ds:0** must be the handle of the block.

es:di Class to call.

Assembly Reference

Returns: **CF** Clear if no message was called; otherwise, this flag is set by the invoked message.

ax, cx, dx, bp Return values set by message.

ds Pointing to same object block. (The address may be different since LMem block may move while locked.)

bx, si, di, es Unchanged.

Message Specifications:

Passed: **es** Segment of class called.

***ds:si** Instance data of object called.

ds:bx Instance data of object called (= *ds:si).

ds:di If class of message handler is in a master part, this is the data for master part of message. Otherwise, ds:di = *ds:si.

cx, dx, bp Other data.

ax Message.

Return: **ax, cx, dx, bp** Return values of message. If message does not return some or all of these registers, those not returned may be destroyed.

May Destroy: **bx, si, di, ds, es** (and unused return registers above).

Destroyed: Nothing.

Library: **object.def**

■ ObjCallInstanceNoLock

Invokes a message of the given instance's class, fixing up ds upon return.

This routine assumes that the object block containing the receiving object is already locked down and is being run by the same process. It is the responsibility of the caller to ensure that this is the case.

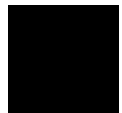
Pass: **ax** Message to invoke.

cx, dx, bp Other data to pass with message.

***ds:si** Instance data of object to call. ds must be pointing to an object block, another local memory block, or a core block. (I.e., ds:0 must be the handle of the block.)

Returns: **CF** Clear if no message was called; otherwise, this flag is set by the invoked message.

ax, cx, dx, bp Return values set by message.



ObjCallInstanceNoLockES

■ 272

ds	Pointing to same object block. (The address may be different since LMem block may move while locked.)
bx, si, di	Unchanged.
es	If es = ds upon entry, then es is destroyed. Otherwise, es is unchanged.

Message Specifications:

Passed:	es	Segment of class called.
	*ds:si	Instance data of object called.
	ds:bx	Instance data of object called (= *ds:si).
	ds:di	If class of message handler is in a master part, this is the data for master part of message. Otherwise, ds:di = *ds:si .
	cx, dx, bp	Other data.
	ax	Message.
Return:	ax, cx, dx, bp	Return values of message. If message does not return some or all of these registers, those not returned may be destroyed.

May Destroy: **bx, si, di, ds, es** (and unused return registers above).

Destroyed: Nothing. (Possibly **es**; see above.)

Library: **object.def**

■ ObjCallInstanceNoLockES

Invokes a message of the given object instance's class, fixing up both **ds** and **es** upon return.

This routine assumes that the object block containing the receiving object is already locked down and is being run by the same process. It is the responsibility of the caller to ensure that this is the case.

Pass:	ax	Message to invoke.
	cx, dx, bp	Other data to pass with message.
	*ds:si	Instance data of object to call. ds must be pointing to an object block, another local memory block, or a core block. (I.e. ds:0 must be the handle of the block.)
	es	Pointing to an object block, a local memory bloc, or a core block. (I.e. es:0 must be the handle of the block.)
Returns:	CF	Clear if no message was called; otherwise, this flag is set by the invoked message.
	ax, cx, dx, bp	Return values set by message (if any).

Assembly Reference

ds	Pointing to same object block as the ds passed. (The address may be different since lmem block may move while locked.)
es	Pointing to same object block as the es passed. (The address may be different since lmem block may move while locked.)
bx, si, di	Unchanged.

Message Specifications:

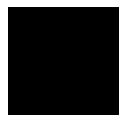
Passed:	es	Segment of class called.
	*ds:si	Instance data of object called.
	ds:bx	Instance data of object called (= *ds:si).
	ds:di	If class of message handler is in a master part, this is the data for master part of message. Otherwise, ds:di = *ds:si .
	cx, dx, bp	Other data.
	ax	Message.
Return:	ax, cx, dx, bp	Return values of message. If message does not return some or all of these registers, those not returned may be destroyed.
May Destroy:	bx, si, di, ds, es	(and unused return registers above).

Destroyed: Nothing.**Library:** **object.def**■ **ObjCallSuperNoLock**

Invokes a message of the given object instance's superclass.

This routine assumes that the object block containing the receiving object is already locked down and is being run by the same process. It is the responsibility of the caller to ensure that this is the case.

Pass:	ax	Message to invoke.
	cx, dx, bp	Other data to pass with message.
	*ds:si	Instance data of object to call. ds must be pointing to an object block, another local memory block, or a core block. (I.e. ds:0 must be the handle of the block.)
	es:di	Class to call superclass of.
Returns:	CF	Clear if no message was called; otherwise, this flag is set by the invoked message.
	ax, cx, dx, bp	Return values set by message (if any).
	ds	Pointing to same object block as the ds passed. (The address may be different since LMem block may move while locked.)



ObjCompAddChild

■ 274

`bx, si, di, es`
Unchanged.

Message Specifications:

Passed:	<code>es</code>	Segment of class called.
	<code>*ds:si</code>	Instance data of object called.
	<code>ds:bx</code>	Instance data of object called (= <code>*ds:si</code>).
	<code>ds:di</code>	If class of message handler is in a master part, this is the data for master part of message. Otherwise, <code>ds:di = *ds:si</code> .
	<code>cx, dx, bp</code>	Other data useful to message.
	<code>ax</code>	Message.
Return:	<code>ax, cx, dx, bp</code>	Return values of message. If message does not return some or all of these registers, those not returned may be destroyed.
May Destroy:	<code>bx, si, di, ds, es</code> (and unused return registers above).	

Destroyed: Nothing.

Library: **object.def**

■ ObjCompAddChild

Adds a child object to the composite field of another, composite object.

Pass:	<code>*ds:si</code>	Instance data of composite object.
	<code>cx:dx</code>	Object to add as child.
	<code>bp</code>	CompChildFlags to specify desired child location. Use the <code>CCF_MARK_DIRTY</code> field to mark chunks as dirty.
	<code>bx</code>	Offset to master instance offset to part containing <code>LinkPart</code> and <code>CompPart</code> fields.
	<code>ax</code>	Offset to field of type “LinkPart” in instance data.
	<code>di</code>	Offset to field of type “CompPart” in instance data.
Returns:	<code>ds</code>	Updated to point at segment of same block as on entry.
Destroyed:	<code>ax, bx, di, bp</code> .	
Library:	metaC.def	

■ ObjCompFindChild

Pass a message to the next sibling of a linkable object.

Pass:	<code>*ds:si</code>	Object.
	<code>cx: dx</code>	Opnr of child object, or to find the n th child let <code>cx</code> be zero and <code>dx</code> the number n of the child to find (first child is zero).
	<code>ax</code>	Offset to field of type “LinkPart” in instance data.

Assembly Reference

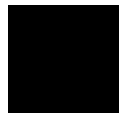
	bx	Offset to master instance offset to part containing LinkPart and CompPart.
	di	Offset to field of type "CompPart" in instance data.
Returns:	CF	Set if not found, clear otherwise.
	bp	If child found, this is the child's position (first child is zero). If child not found, this register holds the total number of children.
	cx:dx	If child found, OD of child. If child not found and you were searching for an OD, then these registers will be unchanged. If child not found and you were searching for a child by position, then cx will be unchanged and dx will be equal to the passed child number minus the total number of children.
Destroyed:	Nothing.	
Library:	metaC.def	

■ ObjCompMoveChild

This routine performs a MSG_MOVE_CHILD for a composite object.

This routine is used to move a child to a different location in the list of children. The child is not physically moved, rather the nextSibling pointers (and possibly the composite's firstChild pointer) are changed. The child to be moved must be a child of the composite (or else a fatal error will be generated in the Error Checking system). The location to add the child is determined by the flags and the reference child passed.

Pass:	*ds:si	Object.
	cx: dx	Optr of child object.
	ax	Offset within master group instance data to "LinkPart" in the child.
	bx	Offset of master group pointer in composite and child's base structure.
	di	Offset within master group instance data to CompPart in the composite.
	bp	CompChildOptions to specify target position.
Returns:	ds	Updated to point at segment of same block as on entry.
Destroyed:	ax, bx, di.	
Library:	metaC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them.	



■ ObjCompProcessChildren

This routine processes the children of a composite object via a callback routine or via several predefined callback routines.

The callback routine is called for each child in order with all passed registers preserved except **bx**. The callback routine returns the carry set to end processing at this point.

Pass: ***ds:si** Object.
 bx Offset of master group pointer in composite and child's base structure.
 di Offset within master group instance data to CompPart in the composite.
 ax, cx, dx, bp Parameters to pass to callback routine. If using an **ObjCompCallType** instead of a custom routine, then **ax** is the message to send to children.

Pass on stack:(Pushed in this order)

optr Object descriptor of initial child to process (or zero to start at composite's *n*th child, where *n* is stored in the chunk half of the **optr**).
 word Offset to field of type LinkPart in instance data.
 fptr Address of callback routine (segment pushed first), or if segment is zero then offset is an **ObjCompCallType**. This field will be popped off of stack on return.

Returns: Callback routine popped off of stack.
 CF Set if call aborted in the middle.
 ax, cx, dx, di, bp Returned with callback routine's changes.
 ds Pointing at same block (could have moved).
 es Untouched (i.e. isn't fixed up, even if it points at a block that might have moved).

Destroyed: **di**.

Callback Routine Specifications:

Passed: ***ds:si** Child object.
 ***es:di** Composite object.
 ax, cx, dx, bp Data.
 Return: **CF** Set to end processing.
 ax, cx, dx, bp Data to send to next child.
 May Destroy: **bx, si, di, ds, es**.

Library: **metaC.def**

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them.

■ ObjCompRemoveChild

This routine performs a MSG_REMOVE_CHILD for a composite object.

This routine is used to remove a child from a composite. The child to be removed must be a child of the composite (The error checking system will generate a fatal error otherwise.). the child is not destroyed and is not sent any notification that it is being removed.

Pass: ***ds:si** Object.
 cx:dx Optr of child object.
 ax Offset within master group instance data to "LinkPart" in the child.
 bx Offset of master group pointer in composite and child's base structure.
 di Offset within master group instance data to CompPart in the composite.
 bp CompChildFlags. Set CCF_MARK_DIRTY to mark chunks as dirty.

Returns: **ds** Updated to point at segment of same block as on entry.

Destroyed: **ax, bx, di.**

Library: **metaC.def**

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them.

■ ObjDecInUseCount

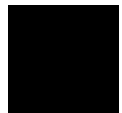
Decrements the in-use count for an object block, stored in the **ObjLMemBlockHeader** structure at the from of an object block. The in-use count ensures that an object block being used is not freed prematurely.

Pass: **ds** Object block.
 si Chunk handle of object in block that is decrementing the count. (In EC code, this is used to keep track of counts on a per-object basis.)

Returns: Nothing.

Destroyed: Nothing.

Library: **object.def**



ObjDecInteractibleCount

■ 278

■ ObjDecInteractibleCount

Decrements the interactible count of the passed object block. This count quantifies the number of objects in the block that are currently interactible with the user.

Pass:	ds	Object block.
	si	Chunk handle of object in block that is decrementing the count. (In EC code, this is used to keep track of counts on a per-object basis.)
Returns:		Nothing.
Destroyed:		Nothing.
Library:		object.def

■ ObjDoRelocation

Relocates a single piece of data (word or double word) within an object block.

Pass:	al	ObjRelocationType.
	bx	Handle of block to perform relocation operation.
	cx	Low word of relocation data.
	dx	High word of relocation data. (This is only used if the ObjRelocationType is RELOC_ENTRY_POINT.)
Returns:	CF	Set if error is encountered.
	cx	Low word relocated.
	dx	High word relocated (unless ObjRelocationType was not RELOC_ENTRY_POINT, in which case dx is unchanged.)
Destroyed:		Nothing.
Library:		object.def

■ ObjDoUnRelocation

Unrelocates a single piece of data (word or double word) within an object block.

Pass:	al	ObjRelocationType.
	bx	Handle of block to perform relocation operation.
	cx	Low word of relocation data.
	dx	High word of relocation data. (This is only used if the ObjRelocationType is RELOC_ENTRY_POINT.)
Returns:	CF	Set if error is encountered.
	cx	Low word unrelocated.

Assembly Reference

dx High word unrelocated (unless `ObjRelocationType` was not `RELOC_ENTRY_POINT`, in which case **dx** is unchanged.)

Destroyed: Nothing.

Library: **object.def**

■ ObjDuplicateMessage

Duplicates an encapsulated message (event), returning a copy of the event.

Pass: **bx** Message to duplicate.

Returns: **ax** Duplicate message (event).

Destroyed: Nothing.

Library: **object.def**

■ ObjDuplicateResource

Duplicates an object resource block (and any objects and chunks within that object block), returning the handle of the newly-created block.

Pass: **bx** Resource block handle to duplicate.
ax Handle of geode to own block.
Zero to have block owned by geode running current thread.
-1 to copy owner from source block.
cx Handle of thread to run new block.
Zero to have block run by current thread.
-1 to copy nature of thread from source block.
(If source is process-driven, duplicated block will be process-driven. If source is ui-driven, duplicated block will be ui-driven. If source is run by anything else, that same thread will run the new thread.)

Returns: **bx** Handle of duplicated block.

Destroyed: Nothing.

Library: **object.def**

■ ObjEnableDetach

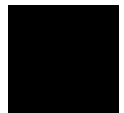
Acknowledge detach for an object.

Pass: ***ds:si** Object.

Returns: **ds** Updated to point at segment of same block as on entry.

Destroyed: Nothing.

Library: **metaC.def**



ObjFreeChunk

■ 280

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them

■ ObjFreeChunk

Frees a chunk within an object block. (Chunks within resources will remain, with their size resized to zero.)

Pass: ***ds:ax** Chunk to free.

Returns: Nothing.

Destroyed: **ax**

Library: **object.def**

■ ObjFreeDuplicate

Frees a block created by **ObjDuplicateResource** or saved with **ObjSaveBlock**.

Pass: **bx** Handle of block to free.

Returns: Nothing.

Destroyed: **bx**

Library: **object.def**

■ ObjFreeMessage

Frees an event handle (and the event attached to that handle).

Pass: **bx** Event handle.

Returns: Nothing.

Destroyed: Nothing.

Library: **object.def**

■ ObjFreeObjBlock

Frees an object block. Object blocks created with **ObjDuplicateBlock** or **ObjSaveBlock** should use **ObjFreeDuplicate** instead.

Pass: **bx** Object block.

Returns: Nothing.

Destroyed: Nothing.

Library: **object.def**

Assembly Reference

■ **ObjGetFlags**

Returns the **ObjChunkFlags** associated with an object block's chunk.

Pass: **ds** Object block.
 ax Chunk.
 Returns: **al** **ObjChunkFlags**.
 ah Zero.
 Destroyed: Nothing.
 Library: **object.def**

■ **ObjGetMessageInfo**

Returns information about an event handle.

Pass: **bx** Event handle.
 Returns: **ax** Message number.
 cx:si Destination optr.
 CF Set if event contains stack data, clear if event contains
 register data.
 Destroyed: Nothing.
 Library: **object.def**

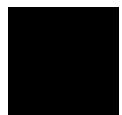
■ **ObjGotoSuperTailRecurse**

This is an optimized version of **ObjCallSuperNoLock** that only works in the case of tail recursion.

Pass: **ax** method number to call
 cx, dx, bp other data to pass
 ds:*si instance data to pass (**si** is lmem handle) or **ds** = core block
 of process. The **ds** register must point to an object block,
 other local memory block, or a core block—**ds:0** must be the
 handle of the block.
 es:di class to call superclass of
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **object.def**

■ **ObjIncDetach**

Increment the acknowledge count for a detaching object.



ObjIncInUseCount

■ 282

Pass: ***ds:si** Object.
Returns: Nothing.
Destroyed: Nothing.
Library: **metaC.def**

■ ObjIncInUseCount

Increments the in-use count for an object block, stored in the **ObjLMemBlockHeader** structure at the front of an object block. The in-use count ensures that an object block being used is not freed prematurely.

Pass: **ds** Object block.
 si Chunk handle of object in block that is incrementing the count. (In EC code, this is used to keep track of counts on a per-object basis.)
Returns: Nothing.
Destroyed: Nothing.
Library: **object.def**

■ ObjIncInteractableCount

Increments the interactible count of the passed object block. This count quantifies the number of objects in the block that are currently interactible with the user.

Pass: **ds** Object block.
 si Chunk handle of object in block that is incrementing the count. (In EC code, this is used to keep track of counts on a per-object basis.)
Returns: Nothing.
Destroyed: Nothing.
Library: **object.def**

■ ObjInitDetach

Prepare to detach an object.

Pass: ***ds:si** Object.
 dx:bp optr to receive MSG_META_SHUTDOWN_ACK.
 ax Message provoking this call (MSG_META_DETACH or MSG_META_APP_SHUTDOWN).
 cx Caller ID; an identifier token for the caller.

Assembly Reference

Returns: **ds** Updated to point at segment of same block as on entry.
Destroyed: Nothing.
Library: **metaC.def**
Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them

■ ObjInitializeMaster

Initializes a master part of an object.

Pass: ***ds:si** Object.
es:di Class of part to initialize.
Returns: **CF** Set.
ds Possibly changed.
Destroyed: Nothing.
Library: **object.def**

■ ObjInitializePart

Ensures that an object is expanded, and initialized if necessary, for all master parts down through the part passed. This routine sends MSG_META_RESOLVE_VARIANT_SUPERCLASS to any master parts above the one passed, if that variable class had not yet been determined.

This routine may resize any LMem and/or object blocks, moving them on the heap and invalidating stored segment pointers to them.

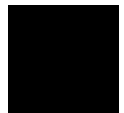
Pass: ***ds:si** Object.
bx Offset to the part to build.
Returns: **ds** Possibly changed.
Destroyed: Nothing.
Library: **object.def**

■ ObjInstantiate

Instantiates (creates) an object of the passed class, allocating a chunk for the object, initializing the chunk to zeroes, filling in the class pointer and passing MSG_PROCESS_INSTANTIATE to the object (if it has no master classes).

If the object block is run by a different process, instantiation is done via a remote call.

Pass: **es:di** Class to instantiate a new object.



ObjIsClassADescendant

■ 284

	bx	Handle of block in which to instantiate the object.
Returns:	si ds	Chunk handle to the new object Updated to point at same segment as on entry.
Destroyed:	Nothing.	
Library:	object.def	

■ ObjIsClassADescendant

	Test whether a given class is a subclass of another specified class.	
Pass:	ds:si es:di	Class pointer to the potential ancestor class. Class pointer to the potential descendant class.
Returns:	CF	Set if class in es:di is a descendant of the class in ds:si . Clear otherwise.
Destroyed:	Nothing.	
Library:	object.def	

■ ObjIsObjectInClass

	Tests whether or not an object is of a given class. If a variant class is encountered, the object will not be grown out past that class in the search. If you wish to do a complete search past any variant classes, send the object MSG_META_DUMMY first.	
Pass:	*ds:si es:di	Object. Class.
Returns:	CF	Set if object is in the given class.
Destroyed:	Nothing.	
Library:	object.def	

■ ObjLinkCallNextSibling

	Pass a message to the next sibling of a linkable object.	
Pass:	*ds:si ax cx, dx, bp bx di	Object. Message to call. Parameters for message. Offset to master class pointer. Offset into master part where LinkPart is.
Returns:	CF ax, cx, dx, bp	Clear if not method routine called; otherwise set by message handler. Return value of message, if any.

Assembly Reference

bx, si, di, es Unchanged.
ds Updated to point at segment of same block as on entry.

Destroyed: Nothing.

Library: **metaC.def**

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them.

■ ObjLinkCallParent

Pass a message to the parent of a linkable object.

Pass: ***ds:si** Object.
ax Message to call.
cx, dx, bp Parameters for message.
bx Offset to master class pointer.
di Offset into master part where LinkPart is.

Returns: **CF** Clear if not method routine called; otherwise set by message handler.
ax, cx, dx, bp Return value of message, if any.
bx, si, di, es Unchanged.
ds Updated to point at segment of same block as on entry. the address could be different since local memory blocks can move while locked.

Destroyed: Nothing.

Library: **metaC.def**

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them.

■ ObjLinkFindParent

Pass a method to the next sibling of a linkable object.

Pass: ***ds:si** Object.

bx Offset to master class pointer.
di Offset into master part where LinkPart is.

Returns: **bx:si** Parent object, or zero if none.
ds Unchanged.

Destroyed: Nothing.



ObjLockObjBlock

■ 286

Library: **metaC.def**

Warning: This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them.

■ ObjLockObjBlock

Locks an object block, loading in the resource if necessary. If the block is an LMem heap but is not an object block, this routine acts like **MemLock**.

Pass: **bx** Handle of block.

Returns: **ax** Segment.

Destroyed: Nothing.

Library: **object.def**

■ ObjMapSavedToState

Maps a saved/duplicated block to its corresponding VM block handle in the process' state file.

Pass: **bx** Process handle (or 0 for current thread's process).

Returns: **CF** Clear if block is found. Set if no such block appears on a process' saved block list.
ax VM block handle (if block found).

Destroyed: Nothing.

Library: **object.def**

■ ObjMapStateToSaved

Maps a VM block ID from a state file to the corresponding memory block handle for a process.

Pass: **ax** VM block handle.
bx Process handle (or 0 for current thread's process).

Returns: **CF** Clear if block is found. Set if no such block appears on a process' saved block list.
bx (If block is found) Memory block handle.

Destroyed: Nothing.

Library: **object.def**

■ ObjMarkDirty

Register-saving routine to mark an object dirty.

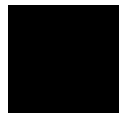
Assembly Reference

Pass: `*ds:si` Chunk in object block to mark dirty.
Returns: Nothing.
Destroyed: Nothing.
Library: **object.def**

■ ObjMessage

Sends a message to an object.

Pass: `bx:si` Destination for message. In most cases, this is the optr of the object to receive the message; this destination may also be a process, in which case `bx` = process ID and `si` contains other data.
 `di` **MessageFlags**. (If `MF_CUSTOM` is passed, a fptr to a callback routine is pushed on the stack. The routine must be locked in memory for the duration of the **ObjMessage**.)
 `ax` Message.
 `cx` Event word 0.
 `dx` Event word 1.
 `bp` Event word 2.
Returns: `di` `MESSAGE_NO_ERROR` if no error is encountered; otherwise **MessageError**.
 (If `MF_CUSTOM` was passed, the custom routine will be popped off the stack.)
 (If `MF_CALL` was passed, `ax`, `cx`, `dx`, `bp`, and `CF` will contain return values.)
Destroyed: Nothing.
Message Specifications:
 Passed: `es` Segment of class called.
 `*ds:si` Instance data of object called. (If class is a subclass of `ProcessClass` `ds` = dgroup of process and `si` = other data passed by caller.)
 `ds:bx` Instance data of object called (= `*ds:si`).
 `ds:di` If the class of the message handler is in a master part, this is the data for master part of message. Otherwise, `ds:di` = `*ds:si`.
 `cx, dx, bp` Other data useful to message.
 `ax` Message.
 Return: `ax, cx, dx, bp` Return values of message. If message does not return some or all of these registers, those not returned may be destroyed.
May Destroy: `bx, si, di, ds, es` (and unused return registers above).



ObjProcBroadcastMessage

■ 288

Library: **object.def**

■ ObjProcBroadcastMessage

Broadcasts an event to all threads with event queues.

Pass: **bx** Encapsulated message (event) to broadcast.

Returns: Nothing.

Destroyed: Nothing.

Library: **object.def**

■ ObjRelocOrUnRelocSuper

Relocate or unrelocate an object's superclass structures and pointers.

Pass: ***ds:si** Segment:Chunk handle of the object's instance chunk.
bp Inherited variables.
es:di Class pointer to the object's class.

Returns: **CF** Set if error, clear otherwise.

Destroyed: **ax, dx, dx**

Library: **object.def**

■ ObjResizeMaster

Resizes a master class part of an object.

Pass: ***ds:si** Object.
bx Offset to master part to expand.
ax New size for master part.

Returns: **ds** Possibly changed.

Destroyed: **ax**

Library: **object.def**

■ ObjSaveBlock

Sets up an LMem block to be saved to its owner's state file.

Pass: **bx** Handle of block to be saved. This block must be an LMem block and have LMF_HAS_FLAGS set in its *LMBH_flags* word (and contain a corresponding flags chunk).

Returns: Nothing.

Destroyed: Nothing.

Library: **object.def**

Assembly Reference

■ **ObjSetFlags**

Sets the object flags (**ObjChunkFlags**) associated with a chunk.

Pass: **ax** Chunk.
 bx ObjChunkFlags to set.
 bh ObjChunkFlags to clear.
 ds Object block.

Returns: Nothing.

Destroyed: Nothing.

Library: **object.def**

■ **ObjSwapLock**

This utility routine locks a new object block and saves the old object's block handle. This is useful when using **ObjCallInstanceNoLock** is much more desirable than using **ObjMessage** for repeated operations; for example, if an object needs to send 5 messages to an object, it might be faster to lock that object and use **ObjCallInstanceNoLock**.

Pass: **ds** Segment of object 1.
 bx Block handle of object 2.

Returns: **ds** Segment of object 2 (now locked, if different from object 1).
 bx Block handle of object 1.

Destroyed: Nothing. Flags preserved.

Library: **object.def**

■ **ObjSwapLockParent**

This utility routine locks the parent of an object and saves the child's block handle. This is useful when using **ObjCallInstanceNoLock** is much more desirable than using **ObjMessage** for repeated operations; for example, if an object needs to send 5 messages to its parent, it might be faster to lock the parent object and use **ObjCallInstanceNoLock**.

Pass: ***ds:si** Object.
 bx Master offset.
 di Offset to linkage part.

Returns: **CF** Set if successful; otherwise, object is not linked to a parent.
 ***ds:si** Instance data of parent object. (**si** = 0 if no parent is encountered.)
 bx Block handle of original object, which is still locked.

Destroyed: Nothing.



ObjSwapUnlock

■ 290

Library: **object.def**

■ ObjSwapUnlock

This utility routine swaps a locked object block, unlocking it in the process, with a new object block.

Pass: **ds** Segment of object 2.
 bx Block handle of object 1 (which must be locked).

Returns: **ds** Segment of object 1.
 bx Block handle of object 2.

Destroyed: Nothing. Flags preserved.

Library: **object.def**

■ ObjTestIfObjBlockRunByCurThread

Determines if the current running thread owns a given object block.

Pass: **bx** Handle of object block. (If **bx** is a VM block, then the thread which runs the VM file is checked.)

Returns: **ZF** Set if current thread is the same thread specified running the object block. The block may be locked, unlocked, and sent messages using any routines available by the current thread. Clear if the block is inaccessible by the current thread. (**ObjMessage** must be called to communicate with any objects in the block.)

Destroyed: Nothing.

Library: **object.def**

■ ObjVarAddData

Adds a new vardata entry or replaces the existing data within an entry (with zeroed data). This routine returns a pointer to the extra data section of the vardata entry in order to allow you to initialize this data immediately.

Pass: ***ds:si** Object to add vardata entry (or replace an entry).
 ax Vardata type (with VDF_SAVE_TO_STATE set correctly by the caller).
 cx Size of extra data. (**cx** = zero if no extra data).

Returns: **ds:bx** (If object has extra data, pointer to the extra data; otherwise, an opaque pointer which may be passed to **ObjVarDeleteDataAt**.)

Object is marked dirty even if vardata entry field previously existed.

Assembly Reference

Destroyed: Nothing.
Library: **object.def**

■ ObjVarCopyDataRange

Copies a group of vardata entries (matching the specified range values) from one object into another object. You can pass an identical start and end vardata type value to copy a single vardata entry from one object another.

Pass: ***ds:si** Source object to copy vardata entries from.
***es:bp** Destination object to copy vardata entries into.
cx Smallest vardata value to copy (inclusive). Any **VarDataFlags** are ignored.
dx Largest vardata value to copy (inclusive). Any **VarDataFlags** are ignored.

Returns: **ds, es** Updated segment addresses of blocks.
Destination object is marked dirty if any entries are copied.

Destroyed: Nothing.
Library: **object.def**

■ ObjVarDeleteData

Deletes a specified vardata entry field and associated extra data (if any) within an object.

Pass: ***ds:si** Object to delete vardata entry from.
ax Vardata type to delete. (**VarDataFlags** ignored.)

Returns: **CF** Set if vardata entry not found, else clear if data deleted.
ds Updated segment of object block.
Object is marked dirty (if data field is found and deleted).

Destroyed: Nothing.
Library: **object.def**

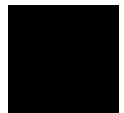
■ ObjVarDeleteDataAt

Deletes a vardata entry field (specified at by the passed opaque pointer).

Pass: ***ds:si** Object to delete vardata entry from.
ds:bx Opaque pointer as returned by **ObjVarAddData**, **ObjVarFindData**, or **ObjVarDerefData**.

Returns: Object is marked dirty.

Destroyed: Nothing.



ObjVarDeleteDataRange

■ 292

Library: **object.def**

■ ObjVarDeleteDataRange

Deletes a group of vardata entries (matching the specified range values). You may also specify that you only wish to delete vardata entries that are not marked VDF_SAVE_TO_STATE.

Pass: ***ds:si** Object to delete vardata entries from.
 cx Smallest vardata value to delete (inclusive). Any **VarDataFlags** are ignored.
 dx Largest vardata value to delete (inclusive). Any **VarDataFlags** are ignored.
 bp Zero to delete all data entries within the passed range values; if non-zero, routine will only delete vardata entries that are both within the passed range and are not marked VDF_SAVE_TO_STATE.

Returns: Object is marked dirty if any vardata entries are deleted.

Destroyed: Nothing.

Library: **object.def**

■ ObjVarDerefData

Returns either a pointer to a vardata entry's extra data section or an opaque pointer to the vardata entry type passed. (If there is no extra data, it returns a pointer to the data entry + offset *VDI_extraData*. This is used as an opaque pointer in such routines as **ObjVarDeleteDataAt**)

Pass: ***ds:si** Object in which to return opaque pointer to vardata data section.
 ax Vardata type. (**VarDataFlags** ignored.)

Returns: **ds:bx** If entry contains extra data, pointer to the extra data section of the vardata entry; otherwise, **ds:bx** is an opaque pointer which may be passed to **ObjVarDeleteDataAt**.

Destroyed: Nothing.

Library: **object.def**

■ ObjVarFindData

Searches an object's vardata section for a given vardata type. If found, this routine returns a pointer to the appropriate vardata section (or an opaque pointer if there is no extra data; this opaque pointer should be used before performing any LMem operations on the block containing the object.)

Pass:	*ds:si ax	Object to search vardata section for the particular data type. Vardata type. (VarDataFlags ignored.)
Returns:	CF ds:bx	Set if vardata type is found; otherwise clear. If vardata entry contains extra data, ds:bx returns containing a pointer to the extra data; otherwise, ds:bx returns containing an opaque pointer which may be passed to ObjVarDeleteDataAt . (This opaque pointer actually points to the location of the vardata entry + <i>VDI_extraOffset</i> .)
Destroyed:	Nothing.	
Library:	object.def	

■ ObjVarScanData

Scans an object's vardata and calls all pertaining routines listed in a "vardata handler" table.

Pass:	*ds:si ax es:di	Object to scan vardata fields. Number of VarDataHandler structures in table. Pointer to a list of VarDataHandler structures. The handler routines must be far routines in the same segment as the handler table.
	cx, dx, bp	Data to pass to vardata handlers.
Returns:	cx, dx, bp ds	Data returned after passing through handlers. Updated segment address of object.
Destroyed:	Nothing.	
VarData Handler Routine Specifications:		
Passed:	*ds:si ds:bx	Object. Extra data of vardata entry in question; otherwise an opaque pointer to the start of vardata entry + size vardata entry.
	ax cx, dx, bp	Vardata type. Data to pass to VarDataHandler .
Return:	ds cx, dx, bp	Updated segment for object. Returned data from VarDataHandler .
May Destroy:	ax, bx, si, di, es	
Library:	object.def	

■ ParserAddDependencies

Adds a set of dependencies from a dependency block.

Pass:	bx	Handle of the dependency block.
-------	-----------	---------------------------------



ParserAddSingleDependency

■ 294

	ss:bp	Pointer to DependencyParameters on the stack.
Returns:	CF al	Set on error; otherwise clear. (If CF is set): Error code.
Destroyed:	ax	
Library:	parse.def	

■ ParserAddSingleDependency

Adds a single dependency to a cell.

Pass:	ds:si ax cx ss:bp	CellFunctionParameters. Row of cell to add dependency to. Column of cell to add dependency to. DependencyParameters.
Returns:	CF al	Set on error; otherwise clear. (If CF is set): Error code.
Destroyed:	ax	
Library:	parse.def	

■ ParserErrorMessage

Returns a text-based error message string, when passed a **ParserScannerEvaluatorError**.

Pass:	ds:si al	Buffer to place the error message string. ParserScannerEvaluatorError
Returns:	cx	Length of the error message (not including the null).
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserEvalExpression

Evaluates a stream of parser tokens.

Pass:	ds:si es:di cx	Pointer to parsed expression. Pointer to the base of a scratch buffer. This buffer consists of two stacks. The argument stack grows down from the top of the buffer and the operator/function stack grows up from the bottom of the buffer. The two stacks should not collide. If they do, an error is reported. Size of scratch buffer. This size must be large enough to avoid collisions between the argument stack and the operator/function stack. (See above.)
-------	---	--

Assembly Reference

	ss:bp	Pointer to EvalParameters structure on the stack.
Returns:	CF	Set if a “serious” error occurs. If a non-serious error occurs, then the evaluator argument stack will contain the error token.
	al	(If CF is set): ParserScannerEvalError code.
	es:bx	Pointer to the evaluated stream.
	ss:[bp].EP_depHandle	If generating dependencies, then this holds the block handle of the locked block containing the list of cells, ranges, names, and functions that the expression depends on.
Destroyed:	ah	
Library:	parse.def	

■ ParserEvalForeachArg

Calls a callback routine for each argument within an argument stack.

Pass:	es:bx	Pointer to argument stack.
	cx	Number of arguments.
	ss:bp	Pointer to EvalParameters structure on the stack.
	ax:si	Pointer to callback routine.
Returns:	CF	Set on error; otherwise clear.
	al	(If CF is set): Error code.
Destroyed:	ax	
Callback Routine Specifications:		
	Passed:	es:bx Pointer to argument on the stack.
		cx Argument number (zero-based).
	Return:	CF Set on error.
		(If CF is set): al = error code.
Library:	parse.def	

■ ParserEvalPopNArgs

Pops a number of arguments off the argument stack. It is important to note that this routine does not perform any sort of destructive acts to the argument stack.

Pass:	es:bx	Pointer to top of argument stack.
	ss:bp	Pointer to EvalParameters structure on the stack.
	cx	Number of arguments to pop off.
Returns:	es:bx	Pointer to new top of argument stack.



ParserEvalPropagateEvalError

■ 296

Destroyed: Nothing.

Library: **parse.def**

■ ParserEvalPropagateEvalError

Propagates an error up the argument stack.

Pass:	es:bx	Pointer to the argument stack.
	es:di	Pointer to the operator stack.
	cx	Number of arguments to pop.
	al	Error code to put on the stack.
Returns:	CF	Set on error; otherwise clear.
	es:bx	New pointer to top of argument stack.
	al	Error code.

Destroyed: **ax**.

Library: **parse.def**

■ ParserEvalPushArgument

Pushes an argument onto the argument stack.

Pass:	es:bx	Pointer to the argument stack.
	es:di	Pointer to the operator stack.
	al	Type of item (EvalArgumentType).
	cx	Additional size to allocate beyond which would normally be assumed for this item.
	ss:bp	Pointer to EvalParameters structure on the stack.
Returns:	CF	Set on error; otherwise clear.
	es:bx	New pointer to top of argument stack. (This points to the allocated item.)
	al	(If CF set): Error code.

Destroyed: Nothing.

Library: **parse.def**

■ ParserEvalPushCellReference

Pushes a cell reference on the argument stack.

Pass:	ds:si	Pointer to ParseTokenCellData .
	es:di	Pointer to top of operator stack.
	es:bx	Pointer to top of argument stack.
	ss:bp	Pointer to EvalParameters structure on the stack.
Returns:	CF	Set on error; otherwise clear.

Assembly Reference

	si	Pointer past the ParserTokenCellData .the allocated item.)
	al	(If CF set): Error code.
Destroyed:	ax	
Library:	parse.def	

■ ParserEvalPushNumericConstant

Pushes a numeric constant on the argument stack.

Pass:	ds:si	Pointer to ParseTokenNumberData .
	es:di	Pointer to top of operator stack.
	es:bx	Pointer to top of argument stack.
	ss:bp	Pointer to EvalParameters structure on the stack.
Returns:	CF	Set on error; otherwise clear.
	si	Pointer past the ParserTokenNumberData .the allocated item.)
	al	(If CF set): Error code.
Destroyed:	ax	
Library:	parse.def	

■ ParserEvalPushNumericConstantWord

Pushes a word-length numeric constant onto the argument stack.

Returns:	es:di	Pointer to top of operator stack.
	es:bx	Pointer to top of argument stack.
	ss:bp	Pointer to EvalParameters structure on the stack.
	cx	Word value to push.
Returns:	es:bx	New argument stack.
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserEvalPushRange

Pushes a range on the argument stack.

Pass:	ds:si	Pointer to EvalRangeData .
	es:di	Pointer to top of operator stack.
	es:bx	Pointer to top of argument stack.
	ss:bp	Pointer to EvalParameters structure on the stack.
Returns:	CF	Set on error; otherwise clear.
	al	(If CF set): Error code.



ParserEvalPushStringConstant

■ 298

Destroyed: **ax**

Library: **parse.def**

■ ParserEvalPushStringConstant

Pushes a string constant on the argument stack.

Pass:	ds:si	Pointer to string data.
	cx	Size of the string.
	es:di	Pointer to top of operator stack.
	es:bx	Pointer to top of argument stack.
	ss:bp	Pointer to EvalParameters structure on the stack.

Returns:	CF	Set on error; otherwise clear.
	al	(If CF set): Error code.

Destroyed: **ax**

Library: **parse.def**

■ ParserEvalRangeIntersection

Implements the range intersection operator on two passed ranges.

Pass:	ds:si	Pointer to first range. This range must contain absolute values (i.e. \$A\$1).
	es:di	Pointer to second range. This range must also contain absolute values (i.e. \$A\$1).

Returns:	es:di	Intersection of the two ranges, in absolute values.
	CF	Set on error.
	al	(If CF is set): Error code. (This is always PSEE_ROW_OUT_OF_RANGE or PSEE_COLUMN_OUT_OF_RANGE .)

Destroyed: **ax**

Library: **parse.def**

■ ParserForeachPrecedent

Calls a routine for each entry within a precedent list. (Precedents are the actual cells that an expression depends on.)

Pass:	bx	Block.
	di:dx	Callback routine.

Returns:	CF	Set on error; otherwise clear.
	al	(If CF is set): Error code.

Destroyed: **ax**

Assembly Reference

Callback Routine Specifications:

Passed:	cx, ds, si, bp	Same as passed in.
	d1	Type of precedent.
	es:di	Pointer to the precedent data.
Return:	CF	Set to abort continuation.
	al	(If CF is set): Error code.
May Destroy:	Nothing.	

Library: **parse.def**

■ ParserForeachReference

Calls a routine for each reference within the given expression. (References are the cells that are referred to in the expression; this may be either a cell reference or a name.)

Pass:	es:di	Pointer to the expression.
	cx:dx	Callback routine.
	ss:bp	Arguments to the callback routine.
	ds:si	Other arguments to the callback routine.

Returns: Nothing.

Destroyed: Nothing.

Callback Routine Specifications:

Passed:	es:di	Pointer to the cell reference.
	ss:bp	Passed parameters.
	ds:si	Other passed parameters.
	al	Type of reference: PARSER_TOKEN_CELL PARSER_TOKEN_NAME
Return:	Nothing.	
May Destroy:	Nothing.	

Library: **parse.def**

■ ParserForeachToken

Calls a routine for each token within the given expression. (A token is a distinct, separate element within an expression; e.g. a number, operator, function, etc.)

Pass:	es:di	Pointer to the expression.
	cx:dx	Callback routine.
	ss:bp	Arguments to the callback routine.



ParserFormatCellReference

■ 300

ds:si Other arguments to the callback routine.

Returns: Nothing.

Destroyed: Nothing.

Callback Routine Specifications:

Passed:	es:di	Pointer to the cell reference.
	ss:bp	Passed parameters.
	ds:si	Other passed parameters.
	al	Type of reference: PARSER_TOKEN_CELL PARSER_TOKEN_NAME
Return:		Nothing.
May Destroy:		Nothing.

Library: **parse.def**

■ ParserFormatCellReference

Formats a single cell reference of the form AB123.

Pass:

ax	Row of cell.
cx	Column of cell.
es:di	Pointer to buffer (MAX_CELL_REF_SIZE or larger).

Returns:

es:di	String (null-terminated).
cx	Length of string (without null-terminator).

Destroyed: Nothing.

Library: **parse.def**

■ ParserFormatColumnReference

Formats a column reference in the form "AB."

Pass:

ax	Column number (zero-based) to format.
es:di	Pointer to buffer to place text.
cx	Size of buffer.

Returns: Nothing.

Destroyed: Nothing.

Library: **parse.def**

■ ParserFormatExpression

Formats a parsed expression into a text string.

Pass:

ds:si	Pointer to the parsed expression.
--------------	-----------------------------------

Assembly Reference

	es:di	Pointer to buffer to place the text.
	ss:bp	Pointer to FormatParameters .
Returns:	cx	Length of the text (not including the null-terminator).
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserFormatRangeReference

Formats a multiple (range) cell reference of the form AB123:CD456.

Pass:	ax	Row of starting cell.
	cx	Column of starting cell.
	dx	Row of ending cell.
	bx	Column of ending cell.
	es:di	Pointer to buffer.
Returns:	es:di	String (null-terminated).
	cx	Length of string (without null-terminator).
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserFormatRowReference

Formats a row reference in the form "123."

Pass:	ax	Row number (zero-based) to format.
	es:di	Pointer to buffer to place string (of MAX_REFERENCE_SIZE or larger).
	cx	Size of buffer.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserFormatWordConstant

Formats a word constant.

Pass:	ax	Number to format.
	es:di	Pointer to buffer to place string (of MAX_REFERENCE_SIZE or larger).
	cx	Size of buffer.
Returns:	Nothing.	
Destroyed:	Nothing.	



ParserGetFunctionArgs

■ 302

Library: **parse.def**

■ ParserGetFunctionArgs

Returns the arguments of a specified parser function. The arguments are stored as a text string within the passed buffer.

Pass:	cx	Item number.
	ax	FunctionType to match.
	es:di	Buffer to place string.
Returns:	cx	Number of characters (not including null terminator).
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserGetFunctionDescription

Returns a description of a specified parser function. The description string is stored in the passed buffer.

Pass:	cx	Item number.
	ax	FunctionType to match.
	es:di	Buffer to place string.
Returns:	cx	Number of characters (not including null terminator).
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserGetFunctionMoniker

Returns the name of the specified parser function.

Pass:	cx	Item number.
	ax	FunctionType to match.
	es:di	Buffer to place string.
Returns:	cx	Number of characters (not including null terminator).
Destroyed:	Nothing.	
Library:	parse.def	

■ ParserGetNumberOfFunctions

Returns the number of parser functions matching a particular function type.

Pass:	ax	FunctionType to match.
Returns:	cx	Number of functions of type FunctionType .

Assembly Reference

Destroyed: Nothing.
Library: **parse.def**

■ ParserLocalizeFormats

Re-initialize localization information.

Pass: Nothing.
Returns: Nothing.
Destroyed: Nothing.
Library: **parse.def**

■ ParserParseString

Parses a string.

Pass: **ds:si** Pointer to text to perform the parse operation.
es:di Buffer to place parsed data.
ss:bp Pointer to **ParserParameters** structure on the stack.
Returns: **CF** Set on error; clear otherwise.
al **ParserScannerEvaluatorError**.
cx, dx Range of text where the error was encountered.
es:di Pointer past the last token written.
Destroyed: **ah**
Library: **parse.def**

■ ParserRemoveDependencies

Removes a set of dependencies from a dependency block.

Pass: **bx** Handle of the dependency block.
ss:bp Pointer to **DependencyParameters** on the stack.
Returns: **CF** Set on error; otherwise clear.
al (If **CF** is set): Error code.
Destroyed: **ax**
Library: **parse.def**

■ PrefTestVideoDevice

Checks if the selected video driver is available on the machine.

Pass: **bx:si** Handle of the **PrefDeviceList**.



ProcCallFixedOrMovable

■ 304

Returns:	CF	Set if device is unavailable, clear if available or if it is indeterminate whether the device is available.
	ax	(If CF clear): DisplayType (in al , 0 in ah). (If CF set): 0 if driver is definitely not present or GeodeLoadError +1 if the driver couldn't be loaded for some reason.
Destroyed:	cx, dx, di	
Library:	config.def	

■ ProcCallFixedOrMovable

Calls the routine specified by the passed virtual far pointer.

Pass:	ax	Offset of routine.
	bx	Virtual segment. (Together bx and ax form a virtual far pointer.)
Returns:		From called routine.
Destroyed:		All registers returned by called routine will be returned to the caller of ProcCallFixedOrMovable .
Library:	resource.def	

■ ProcCallModuleRoutine

Calls a process' routine in another code resource. The stack usage of this routine is guaranteed; the routine called will return two words of the return address on the stack followed by any parameters that the caller may have pushed.

Pass:	ax	Offset to routine.
	bx	Virtual segment of routine. (Together bx and ax form a virtual far pointer.)
Returns:		From called routine.
Destroyed:		All registers returned by called routine will be returned to the caller of ProcCallModuleRoutine .
Library:	resource.def	

■ ProcGetLibraryEntry

Returns the address of a routine within a library. You can pass the result directly to **ProcCallFixedOrMovable** to call the routine.

Pass:	ax	Library entry point number.
	bx	Library handle.

Assembly Reference

Returns: **bx:ax** Library entry point (virtual far pointer).
Destroyed: Nothing.
Library: **resource.def**

■ ProcInfo

Returns the contents of the *HM_otherInfo* field for a given process. This field holds the first thread of this process.

Pass: **bx** Process handle.
Returns: **bx** Contents of *HM_otherInfo* field of process (this field holds the first thread of this process).
Destroyed: Nothing.
Library: **geode.def**

■ QueueGetMessage

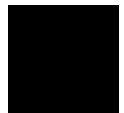
Returns the next event from a given event queue; this routine blocks the queue if it is currently empty until an event is added which can be returned.

Pass: **bx** Handle of event queue.
Returns: **ax** Event handle.
Destroyed: Nothing.
Library: **geode.def**

■ QueuePostMessage

Adds the passed event to the specified event queue.

Pass: **bx** Handle of event queue.
ax Event to post.
si Calling thread.
di **MessageFlags** though only MF_INSERT_AT_FRONT is used (to place the message at the front of the queue).
Returns: Nothing.
Destroyed: Nothing.
Library: **geode.def**



QueuePostMessage

■ 306

■ **Assembly Reference**

■ RangeEnum

Enumerate all cells in a range of cells, calling a callback routine on each.

Pass:	ds:si ss:bp ss:bx d1	Address of a CellFunctionParameters structure. Address (on stack) of local variables for callback. Address (on stack) of RangeEnumParams structure. RangeEnumFlags record with any or all of the following: REF_ALL_CELLS Use callback routine for all cells. REF_NO_LOCK Callback routine will lock and unlock cells. REF_COLUMN_FLAGS Get the ColumnFlags for this cell (passed in the RangeEnumParams structure). REF_MATCH_COLUMN_FLAGS Use callback routine only for cells having matching ColumnFlags as those passed.
-------	---	---

Pass on stack:

NOTE:

These parameters are not to be popped; they are to be referenced by their pointers as specified. The stack frame will be passed to the callback, which also should not pop them.

ss:bp
ss:bx

Local variables for callback routine.

RangeEnumParams structure, with the following fields:

REP_callback Address of the callback routine to process the enumerated cells.

REP_bounds Rectangle structure giving the bounds of the range of cells to enumerate.

REP_columnFlags

ColumnFlags record for the passed cell.

REP_columnFlagsArray

Address of a **ColumnArrayHeader** structure.

REP_cfp

Address of a **CellFunctionParameters** structure.

REP_matchFlags

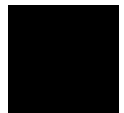
ColumnFlags record to indicate which flags must be set in a cell for it to be processed.

Only valid if REF_MATCH_COLUMN_FLAGS set in **d1**.

Returns: **CF**

Set if callback routine forced an early abortion of the routine.

Destroyed: Nothing.



Callback Routine Specifications:

Passed:	ds:si	Address of CellFunctionParameters as passed to RangeEnum .
	ax	Current cell row.
	cx	Current cell column.
	d1	RangeEnumFlags passed to RangeEnum .
	ss:bp	Address (on stack) of local variables for callback routine. See notes above for values passed on the stack.
	ss:bx	Address (on stack) of RangeEnumParams structure passed to RangeEnum . See notes above for values passed on the stack.
		If REF_COLUMN_FLAGS set in d1 , <i>REP_columnFlagsArray</i> is a pointer to a ColumnArrayHeader structure, and <i>REP_columnFlags</i> is the ColumnFlags record for the cell.
	CF	Set if the cell has data, clear otherwise.
	*es:di	Segment:Chunk handle of cell's data, if any.
Return:	CF	Set to abort RangeEnum , clear to continue.
	es	Updated segment address of the cell.
	d1	Modified RangeEnumFlags to potentially include the following flags: REF_CELL_ALLOCATED Set if the callback routine allocated the cell for which the callback occurred. REF_CELL_FREED Set if the callback routine freed the cell for which the callback occurred. REF_OTHER_ALLOC_OR_FREE Set if the callback routine may have allocated or freed a cell other than the one for which the callback occurred. REF_COLUMN_FLAGS_MODIFIED Set if the callback routine changed the ColumnFlags for the cell.

May Destroy: Nothing.

Library: **cell.def**

Warning: If the caller passes **REF_ALL_CELLS** and **REF_NO_LOCK**, then the callback routine will not know if the cell being called back actually exists or not. In this case, the callback parameters ***es:di** and **CF** are undefined.

If the callback routine allocates the cell which was called back, it should not unlock the cell; **RangeEnum** will take care of this.

If the callback routine allocates a different cell, it must unlock the cell and return `REF_OTHER_ALLOC_OR_FREE`.

If the callback routine frees the cell called back, it must unlock the cell before freeing it. Otherwise, the block containing the freed cell can not be removed (the cell will always be locked). Also, the callback routine must return `REF_CELL_FREED`.

If the callback routine frees a different cell, it must unlock the cell before freeing the cell, as outlined above. Also, the callback must return `REF_OTHER_ALLOC_OR_FREE`.

If the callback routine changes the **ColumnFlags** of the cell, it must also change the data pointed at by *REP_columnFlagsArray* and return the flag `REF_COLUMN_FLAGS_MODIFIED`.

■ RangeExists

Check for the existence of cells in a given range.

Pass:	ds:si	Address of CellFunctionParameters structure.
	ax, cl	Row, Column of first cell in range (inclusive).
	dx, ch	Row, Column of last cell in range (inclusive).
Returns:	CF	Set if one or more cells in the range contain data, clear otherwise.
Destroyed:	Nothing.	
Library:	cell.def	

■ RangeInsert

Insert or delete a range of cells.

Pass:	ds:si	Address of CellFunctionParameters structure.
Pass on stack:	ss:bp	RangeInsertParams structure, with the following fields:
	<i>RIP_bounds</i>	Rectangle structure giving the bounds of the range to insert or delete.
	<i>RIP_delta</i>	Point structure indicating the distance to move the range.
	<i>RIP_cfp</i>	Address of a CellFunctionParameters block; this field should not be initialized by the caller (the others should).
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	cell.def	



RangeSort

■ 310

■ RangeSort

Sort a range of cells either in ascending order or via callback routine.

Pass: **ds:si**

Address of **CellFunctionParameters** structure.

Pass on stack:

ss:bp

RangeSortParams structure, with the following fields:

RSP_range Rectangle structure indicating the range of cells to be sorted.

RSP_active Point structure indicating the cell in the row and column to sort on.

RSP_callback Address of the sort callback routine.

RSP_flags **RangeSortFlags** record with one or more of the following flags set:

RSF_SORT_ROWS

Sort rows in the range.

RSF_SORT_ASCENDING

Sort in ascending order.

RSF_IGNORE_CASE

Ignore case in string comparisons.

RSF_IGNORE_SPACES

Ignore spaces and punctuation in the sort.

This is not supported directly by the cell library but is put here for convenience.

NOTE: The following fields of **RangeSortParams** should not be initialized by the caller as they are used internally by **RangeSort**.

RSP_cfp **CellFunctionParameters**.

RSP_sourceChunk

Chunk for swapping.

RSP_destChunk

Chunk for swapping.

RSP_base Base position for sort.

RSP_lockedEntry

Currently locked entry, or -1.

RSP_cachedFlags

Flags.

Returns: **ax**

RangeSortError value:

RSE_NO_ERROR

No error; the sort succeeded.

RSE_UNABLE_TO_ALLOC

The sorting code was unable to allocate a temporary block necessary for sorting.

Destroyed: Nothing.

Assembly Reference

Library: **cell.def**

■ RowGetFlags

Get flags for specified row.

Pass: **ds:si** Pointer to **CellFunctionParameters** structure.
ax row number.

Returns: **CF** Set if row exists; clear otherwise.
dx Flags for row (zero if row nonexistent).

Destroyed: Nothing.

Library: **cell.def**

■ RowSetFlags

Set the flags for a given row.

Pass: **ds:si** Pointer to **CellFunctionParameters** structure.
ax Row number.
dx Flags for row.

Returns: **CF** Set if row exists; clear otherwise.

Destroyed: Nothing.

Library: **cell.def**

■ RulerScaleDocToWinCoords

Scale a ruler point in document coordinates to window coordinates.

Pass: **ds:si** Segment:Chunk handle of the VisRuler object.
dx.cx.ax **DWFixed** value of point to be scaled.

Returns: **dx.cx.ax** **DWFixed** point, scaled.

Destroyed: Nothing.

Library: **ruler.def**

■ RulerScaleWinToDocCoords

Scale a ruler point in window coordinates to document coordinates.

Pass: **ds:si** Segment:Chunk handle of the VisRuler object.
dx.cx.ax **DWFixed** value of point to be scaled.

Returns: **dx.cx.ax** **DWFixed** point, scaled.

Destroyed: Nothing.

Library: **ruler.def**



■ SoundAllocMusic

Allocate a handle to play FM sounds from fixed memory

Pass:	bx:si	Buffer to play from in fixed memory
	cx	Number of voices used in buffer
Returns:	CF	Set on error; clear on success.
	bx	On success, handle to SoundControl (owned by calling thread); otherwise destroyed.
	ax	On error, this will be a SoundErrors value; otherwise destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundAllocMusicNote

Allocate a note and return its handle.

Pass:	bx	Instrument table seg. (zero for system default)
	si	Instrument number for note
	ax	Frequency
	cx	Volume
	dx	SoundStreamDeltaTimeType .
	di	Duration (in DeltaTimerType units)
Returns:	CF	Clear on success; set on error.
	bx	On success, token for sound; otherwise destroyed.
	ax	On error, SoundErrors value; otherwise destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundAllocMusicStream

Allocate a stream to play FM sounds on.

Pass:	ax	SoundStreamSize
	bx	Starting priority for sound
	cx	Number of voices for sound
	dx	Starting tempo for sound
Returns:	CF	Clear on success; set on error.
	bx	Handle to SoundControl (owned by calling thread); otherwise destroyed.
	ax	On error, SoundErrors value; otherwise destroyed.
Destroyed:	See above.	

Library: **sound.def**

■ SoundAllocNote

Allocate a note structure, define it, and return its handle. A note is a simple sound that has a pre-defined sound buffer. The handle returned may be used with other sound routines to play or free the sound.

Pass: **bx:si** Address of the note's instrument definition (buffer containing sound definition for a given instrument).
ax Frequency of the note, in Hz (cycles per second).
cx Volume of the note (normally a volume constant).
dx **SoundStreamDeltaTimeType** value giving the units of the value passed in **di**.
di Duration of the time between notes, in units of **SoundStreamDeltaTimeType** units as specified in **dx**.
Returns: **bx** Token of the allocated sound.
Destroyed: Nothing.
Library: **sound.def**

■ SoundAllocSampleStream

Allocate a sound handle for the sound.

Pass: Nothing.
Returns: **CF** Clear on success; set on error.
bx (If **CF** clear) Handle to **SoundControl**; (If **CF** set) destroyed.
ax (If **CF** set) **SoundErrors** value; (If **CF** clear) destroyed.
Destroyed: **bx** or **ax** See above.
Library: **sound.def**

■ SoundAllocSimple

Allocate a handle to play FM sounds from fixed memory

Pass: **bx:si** Buffer to play from in fixed memory
cx Number of voices used in buffer
Returns: **CF** Set on error; clear on success.
bx On success, handle to **SoundControl** (owned by calling thread); otherwise destroyed.
ax On error, this will be a **SoundErrors** value; otherwise destroyed.
Destroyed: See above.



SoundAllocSimpleFM

■ 314

Library: **sound.def**

■ SoundAllocSimpleFM

Allocate space on the global heap for a simple, frequency-modulated sound stream for a song. The song should already be created and located in fixed or locked memory. The handle returned references a block containing the sound that may then be played or freed.

Pass:	bx:si	Buffer to play from in fixed memory
	cx	Number of voices used in buffer
Returns:	CF	Set on error; clear on success.
	bx	On success, handle to SoundControl (owned by calling thread); otherwise destroyed.
	ax	On error, this will be a SoundErrors value; otherwise destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundChangeOwner

Change the owner of a sound.

Pass:	bx	Handle of the sound to be changed.
	ax	Geode handle of the sound's new owner.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundChangeOwnerMusic

Change the owner of a sound.

Pass:	bx	Handle of the sound to be changed.
	ax	Geode handle of the sound's new owner.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundChangeOwnerSimple

Change the owner of a sound.

Pass:	bx	Handle of the sound to be changed.
-------	-----------	------------------------------------

Assembly Reference

ax Geode handle of the sound's new owner.
Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

■ SoundChangeOwnerStream

 Change the owner of a sound stream.
Pass: **bx** Handle of the sound to be changed.
 ax Geode handle of the sound's new owner.
Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

■ SoundDisableSampleStream

 Removes the association of DAC and the sound.
Pass: **bx** Handle for **SoundControl**.
Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

■ SoundEnableSampleStream

 Associate a real DAC device to a sound.
Pass: **bx** Handle of **SoundControl**
 ax Priority for DAC (SoundPriority)
 cx Rate for sample
 dx **ManufacturerID** of sample
 si **DACSampleFormat** of sample
Returns: **CF** Clear on success; set on error.
 ax On error, **SoundErrors** value; otherwise destroyed.
Destroyed: See above.
Library: **sound.def**

■ SoundFreeMusic

 Free up a simple FM sound stream.
Pass: **bx** Handle for **SoundControl**.



SoundFreeMusicNote

■ 316

Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

■ SoundFreeMusicNote

Free up an allocated music note.

Pass: **bx** Token of note.
Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

■ SoundFreeMusicStream

Free an FM sound stream.

Pass: **bx** Handle for SoundControl.
Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

■ SoundFreeNote

Free the given note originally allocated with **SoundAllocNote**; the note must *not* be playing when it is freed. If the note may be playing, call **SoundStopNote** before freeing it.

Pass: **bx** Handle of the note to be freed.
Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

■ SoundFreeSampleStream

Frees up the Sound structure of the sound.

Pass: **bx** Handle of **SoundControl**.
Returns: Nothing.
Destroyed: Nothing.
Library: **sound.def**

Assembly Reference

■ SoundFreeSimple

Free up a simple FM sound stream.

Pass: **bx** Handle for **SoundControl**.

Returns: Nothing.

Destroyed: Nothing.

Library: **sound.def**

■ SoundFreeSimpleFM

Free a simple frequency-modulated sound originally allocated with **SoundAllocSimpleFM**. The sound must *not* be playing; if it may be playing, call **SoundStopStream** before freeing it.

Pass: **bx** Handle of the simple sound to be freed.

Returns: Nothing.

Destroyed: Nothing.

Library: **sound.def**

■ SoundGetExclusive

Get exclusive access to the sound driver, blocking if it is currently in use. Generally, applications will call the higher level sound routines rather than access the sound driver's strategy routine directly. If you do call this routine, be sure to call **SoundReleaseExclusive** when done with the sound driver.

Pass: Nothing.

Returns: **cx:dx** Address of the sound library's strategy routine.

bx:di Address of the DAC driver's strategy routine.

ax:si Address of the synthesizer driver's strategy routine.

Destroyed: Nothing.

Library: **sound.def**

■ SoundGetExclusiveNB

Get exclusive access to the sound driver, returning with the carry flag set if it is currently in use. Generally, applications will call the higher level sound routines rather than access the sound driver's strategy routine directly. If you do call this routine and gain exclusive access to the sound driver, be sure to call **SoundReleaseExclusive** when done with the sound driver.

Pass: Nothing.



SoundInitMusic

■ 318

Returns:	CF	Set if another thread has exclusive access, clear if access gained.
	cx:dx	Address of the sound library's strategy routine.
	bx:di	Address of the DAC driver's strategy routine.
	ax:si	Address of the synthesizer driver's strategy routine.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundInitMusic

Initialize a pre-defined simple FM sound structure.

Pass:	bx	Handle to block with empty SoundControl .
	cx	Number of voices for sound.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundInitSimple

Initialize a pre-defined simple FM sound structure.

Pass:	bx	Handle to block with empty SoundControl .
	cx	Number of voices for sound.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundInitSimpleFM

Initialize a pre-defined simple frequency-modulated sound structure. This routine is automatically called when a note is allocated with **SoundAllocNote** or **SoundAllocSimpleFM**.

Pass:	bx	Handle to block with empty SoundControl .
	cx	Number of voices for sound.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundPlayMusic

Play a simple FM sound.

Assembly Reference

Pass:	bx	Handle for SoundControl .
	ax	Starting priority for sound
	cx	Starting tempo setting for sound
	d1	EndOfSongFlags for sound
Returns:	CF	Clear on success; set on error.
	ax	On error, a SoundErrors value; otherwise, destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundPlayMusicNote

Play a note of music.

Pass:	bx	Token of music note.
	ax	Starting priority for sound
	cx	Starting tempo setting for sound
	d1	EndOfSongFlags for sound
Returns:	CF	Clear on success; set on error.
	ax	On error, a SoundErrors value; otherwise, destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundPlayNote

Play the given note according to the other parameters. If the note is currently playing, it will be stopped and immediately restarted. This routine is identical to **SoundPlaySimpleFM**.

Pass:	bx	Handle of the note as returned by SoundAllocNote .
	ax	SoundPriority of the note.
	cx	Tempo of the note (only used if the song requires a tempo).
	d1	EndOfSongFlags record indicating how the note should be handled after being played.
Returns:	CF	Set if the sound library was unavailable, clear otherwise.
	ax	On error, a SoundErrors value; otherwise, destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundPlaySimple

Play a note of music.

Pass:	bx	Token of music note.
-------	-----------	----------------------



SoundPlaySimpleFM

■ 320

	ax	Starting priority for sound
	cx	Starting tempo setting for sound
	dl	EndOfSongFlags for sound
Returns:	CF	Clear on success; set on error.
	ax	On error, a SoundErrors value; otherwise, destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundPlaySimpleFM

Play the given note according to the other parameters. If the sound is currently playing, it will be restarted at the beginning with the new tempo and priority.

Pass:	bx	Handle of the note as returned by SoundAllocSimpleFM .
	ax	SoundPriority of the note.
	cx	Tempo of the note (only used if the song requires a tempo).
	dl	EndOfSongFlags record indicating how the note should be handled after being played.
Returns:	CF	Set if the sound library was unavailable, clear otherwise.
	ax	On error, a SoundErrors value; otherwise, destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundPlayToMusicStream

Play an FM sound to a stream.

Pass:	bx	Handle for SoundControl
	dx:si	Start of event buffer to write to sound stream
	cx	Bytes in buffer (zero if unknown)
Returns:	CF	Clear on success; set on error.
	ax	On error, a SoundErrors value; otherwise destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundPlayToSampleStream

Play a given piece of DAC data to the DAC device.

Pass:	bx	Handle of SoundControl
	dx:si	Buffer of DAC data to put on stream
	cx	Length of buffer (in bytes)

Assembly Reference

	ax:bp	SampleFormatDescription of buffer
Returns:	CF ax	Clear on success; set on error. On error, a SoundErrors value; otherwise destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundReallocMusic

	Change the song setting for a simple stream.	
Pass:	bx ds:si	Handle for SoundControl New sound buffer
Returns:	CF ax	Clear on success; set on error. On error, a SoundErrors value; otherwise destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundReallocMusicNote

	Change the settings of the given note. The note must not be playing; if it may be, call SoundStopNote before reallocating the note.	
Pass:	bx ax cx dx di ds:si	Handle for SoundControl Frequency for note Volume for note Timer type Timer value New instrument setting
Returns:	CF ax	Clear on success; set on error. On error, a SoundErrors value; otherwise destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundReallocNote

	Change the settings of the given note. The note must not be playing; if it may be, call SoundStopNote before reallocating the note.	
Pass:	bx ax cx dx di	Handle of the note as returned by SoundAllocNote . New frequency of the note. New volume of the note. New delay timer type for the note. New delay value for the note, in the units of the new type.



SoundReallocSimple

■ 322

	ds:si	Address of the new instrument for the note.
Returns:	CF	Set if the sound library is unavailable, clear otherwise.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundReallocSimple

		Change the song setting for a simple stream.
Pass:	bx ds:si	Handle for SoundControl New sound buffer
Returns:	CF ax	Clear on success; set on error. On error, a SoundErrors value; otherwise destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundReallocSimpleFM

		Change the settings for a simple note or simple song's stream. This routine restarts the song with the new sound buffer, but it leaves the voices in the state they were in at the end of the last song. This allows playing a very long song by segmenting it into smaller buffers. Each buffer section <i>must</i> end with a GE_END_OF_SONG token, however.
Pass:	bx ds:si	Handle of the simple note or stream, as returned by SoundAllocSimpleFM or SoundAllocStreamFM . Address of the new sound buffer as would be passed to the sound allocation routine.
Returns:	CF ax	Set if the library is unavailable, clear otherwise. On error, a SoundErrors value; otherwise destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundReleaseExclusive

		Release exclusive access to the sound library and sound driver routines. Any thread that calls SoundGrabExclusive or SoundGrabExclusiveNB <i>must</i> call this routine after grabbing the exclusive. In general, a thread should retain exclusive access only as long as it absolutely needs to.
Pass:	Nothing.	
Returns:	Nothing.	

Assembly Reference

Destroyed: Nothing.
Library: **sound.def**

■ SoundSampleDriverInfo

Get information on Sample Driver.

Pass: Nothing.
Returns: **ax** Number of DACs
bx **SoundDriverDACCapability**
Destroyed: Nothing.
Library: **sound.def**

■ SoundStopMusic

Stop a simple stream.

Pass: **bx** Handle of SoundControl.
Returns: **CF** Clear on success; set on error.
ax On error, a **SoundErrors** value; otherwise destroyed.
Destroyed: See above.
Library: **sound.def**

■ SoundStopMusicNote

Stop a music note.

Pass: **bx** Token of music note.
Returns: **CF** Clear on success; set on error.
ax On error, a **SoundErrors** value; otherwise destroyed.
Destroyed: See above.
Library: **sound.def**

■ SoundStopMusicStream

Stop a music stream.

Pass: **bx** Handle of SoundControl.
Returns: **CF** Clear on success; set on error.
ax On error, a **SoundErrors** value; otherwise destroyed.
Destroyed: See above.
Library: **sound.def**



■ SoundStopNote

Stop a note that is playing. This routine should be called before freeing, changing, or reallocating a note if that note could be playing at the time.

Pass:	bx	Handle of the note.
Returns:	CF ax	Set if the library was unavailable, clear otherwise. On error, a SoundErrors value; otherwise destroyed.
Destroyed:	Nothing.	
Library:	sound.def	

■ SoundStopSimple

Stop a music stream.

Pass:	bx	Handle of SoundControl.
Returns:	CF ax	Clear on success; set on error. On error, a SoundErrors value; otherwise destroyed.
Destroyed:	See above.	
Library:	sound.def	

■ SoundStopSimpleFM

Stop a simple frequency-modulated sound from playing. This is similar in usage to **SoundStopNote**.

Pass:	bx	Handle of the simple sound.
Returns:	CF ax	Set if the library was unavailable, clear otherwise. On error, a SoundErrors value; otherwise destroyed.
Library:	sound.def	

■ SoundSynthDriverInfo

Get information on the synthesizer driver.

Pass:	Nothing.	
Returns:	ax bx cx	Number of Voices SupportedInstrumentFormat SoundDriverCapability
Destroyed:	Nothing.	
Library:	sound.def	

■ SpoolAddJob

Add the passed job (a GString file) to the print queue. If the spooler thread has not started, or if there is no queue for the desired device and port, this routine will start them.

Pass:	dx:si	Address of a JobParameters structure. This structure includes information on the document, the paper, the print mode, the number of copies, the port, the device, and the printer.
Returns:	cx	ID of the print job. This ID may be used to track or modify jobs with other spooler routines.
Destroyed:	Nothing.	
Library:	spool.def	

■ SpoolConvertPaperSize

Convert a width and height pair to a paper size index. This is the complement to **SpoolGetPaperSize**.

Pass:	cx	Width, in points.
	dx	Height, in points.
	bp	PageType value: PT_PAPER, PT_ENVELOPE, PT_LABEL.
Returns:	ax	-1 if no page index for the passed size, otherwise Page size number as returned by SpoolCreatePaperSize .
Destroyed:	Nothing.	
Library:	spool.def	

■ SpoolCreatePaperSize

Create a new paper size index and store it in the GEOS.INI file.

Pass:	es:di	Address of a null-terminated text string holding the name of the new size.
	bp	PageType value: PT_PAPER, PT_ENVELOPE, PT_LABEL.
	cx	Width of new paper size, in points.
	dx	Height of new paper size, in points.
	ax	Default PageLayout record for the new size.
Returns:	CF	Set if error, clear otherwise.
	ax	Paper size number of the new size.
Destroyed:	Nothing.	
Library:	spool.def	



SpoolCreatePrinter

■ 326

■ SpoolCreatePrinter

Create a new printer, adding it to the end of the list of installed printers. This routine does not initialize all the information for the printer but only adds it to the installed list.

Pass:	es:di	Address of the null-terminated printer name string. (Must be at most GEODE_MAX_DEVICE_NAME_SIZE bytes.)
	cl	PrinterDriverType of the new printer.
Returns:	CF	Set if error, clear otherwise.
	ax	Index number of the new printer in the installed list.
Destroyed:	Nothing.	
Library:	spool.def	

■ SpoolCreateSpoolFile

Create and open a new, unique spool file in the SP_SPOOL directory.

Pass:	dx:si	Address for the null-terminated 8.3 file name (must be at least 13 bytes long and locked or fixed).
Returns:	dx:si	Address of the new null-terminated filename.
	ax	File handle of the new file. Null handle returned if the file could not be opened.
Destroyed:	Nothing.	
Library:	spool.def	

■ SpoolDelJob

Delete a job from the spooler queue, given a job ID.

Pass:	cx	ID of print job, as returned by SpoolAddJob .
Returns:	ax	SpoolOpStatus value giving the status of the queue.
Destroyed:	Nothing.	
Library:	spool.def	

■ SpoolDelayJob

Move the specified job to the end of the spooler queue.

Pass:	cx	ID of print job, as returned by SpoolAddJob .
Returns:	ax	SpoolOpStatus value giving the status of the queue.
Destroyed:	Nothing.	

Assembly Reference

Library: **spool.def**

■ SpoolDeletePaperSize

Delete a paper size from the paper size list.

Pass: **bp** **PageType** value: PT_PAPER, PT_ENVELOPE, PT_LABEL.
ax Paper size index as returned by **SpoolCreatePaperSize**.

Returns: **CF** Set if error, clear otherwise.

Destroyed: Nothing.

Library: **spool.def**

■ SpoolDeletePrinter

Delete a printer from the list of currently installed printers.

Pass: **ax** Index in list of printer to delete.

Returns: Nothing.

Destroyed: Nothing.

Library: **spool.def**

■ SpoolGetDefaultPageSizeInfo

Return the default page information for the spooler.

Pass: **ds:si** Address of an empty **PageSizeReport** structure.

Returns: **ds:si** Address of the filled **PageSizeReport** structure containing the default page information in the following fields:

PSR_width Width of the page.
PSR_height Height of the page.
PSR_layout **PageLayout** record defining layouts.
PSR_margins **PCMarginParams** structure giving document margins.

Destroyed: Nothing.

Library: **spool.def**

■ SpoolGetNumPaperSizes

Return the number of defined paper sizes for the given type.

Pass: **bp** **PageType** value: PT_PAPER, PT_ENVELOPE, PT_LABEL.

Returns: **cx** Number of paper sizes defined for the type.
dx Index of the default size for the given type.

Destroyed: Nothing.



SpoolGetNumPrinters

■ 328

Library: **spool.def**

■ SpoolGetNumPrinters

Return the number of printers installed for the given driver type.

Pass:	c1	PrinterDriverType: PDT_PRINTER, PDT_PLOTTER, PDT_FACSIMILE, PDT_CAMERA, PDT_OTHER.
	ch	Non-zero requests that only local numbers be counted; zero asks that all printers be counted.
Returns:	ax	Number of installed printers for the driver type.
Destroyed:	Nothing.	
Library:	spool.def	

■ SpoolGetPaperSize

Return the dimensions of the requested paper size. This is the complement to **SpoolConvertPageSize**.

Pass:	ax	Index of the paper size to convert.
	bp	PageType value: PT_PAPER, PT_ENVELOPE, PT_LABEL.
Returns:	cx	Width of paper, in points.
	dx	Height of paper, in points.
	ax	Default PageLayout record for the paper size.
Destroyed:	Nothing.	
Library:	spool.def	

■ SpoolGetPaperSizeOrder

Return the paper size order array for the given page type.

Pass:	bp	PageType of the array to get.
	es:di	Address of a locked or fixed buffer to hold the returned array (must be at least MAX_PAPER_SIZES bytes).
	ds:si	Address of a locked or fixed buffer to hold the array of user-defined sizes (must be at least MAX_PAPER_SIZES bytes).
Returns:	es:di	Address of the filled paper size array. Each entry in the array corresponds to the paper size having that index and is a single byte, the value of which determines its meaning: 0 – 127 Pre-defined paper size. 128 – 255 User-defined paper size.
	ds:si	Address of the filled user-defined size array. Each entry in the array corresponds to a single user-defined size, ordered as in

Assembly Reference

the order array (**es:di**). Each entry in this array is a single byte, whose value signifies the following:

0 Paper size is in-use (displayed to the user).

1 Paper size is not in-use (not displayed).

dx Number of unused (non-displayed) paper sizes.

cx Number of ordered sizes (number of entries in the array pointed to by **es:di**).

Destroyed: Nothing.

Library: **spool.def**

■ SpoolGetPaperString

Return the paper size string for the specified paper size and page type.

Pass: **ax** Index of the paper size.
bp **PageType** value: PT_PAPER, PT_ENVELOPE, PT_LABEL.
es:di Address of a locked or fixed buffer for the returned string.
The buffer must be at least MAX_PAPER_STRING_LENGTH bytes long.

Returns: **CF** Set if error, clear otherwise.
cx Length of returned string, not including the null terminator.
es:di Address of the returned null-terminated name string.

Destroyed: Nothing.

Library: **spool.def**

■ SpoolGetPrinterString

Return the printer name string for the specified printer.

Pass: **ax** Index of the printer on the installed printer list.
es:di Address of a locked or fixed buffer for the returned string.
This must be at least GEODE_MAX_DEVICE_NAME_SIZE bytes long.

Returns: **CF** Set if error, clear otherwise.
cx Length of the returned name string, not including the null terminator.

d1 **PrinterDriverType** of the installed printer.

es:di Address of the returned null-terminated printer name string.

Destroyed: Nothing.

Library: **spool.def**



SpoolHurryJob

■ 330

■ SpoolHurryJob

Move the given print job to the front of the spooler's queue.

Pass: **cx** ID of the print job to be hurried.

Returns: **ax** **SpoolOpStatus** value.

Destroyed: Nothing.

Library: **spool.def**

■ SpoolInfo

Return information about either the spooler's jobs or the spooler's queue.

Pass: **cx** **SpoolInfoType** value:
SIT_JOB_INFO Return information about a job in the
spooler's job queue.
SIT_QUEUE_INFO Return the spool's queue listing.

If SIT_JOB_INFO passed in **cx**:

dx ID of the print job to retrieve.

If SIT_QUEUE_INFO passed in **cx**:

dx:si Address of a **PrintPortInfo** structure
defining which port's queue to return. If **dx** is
passed -1, only a value indicating whether
ports are active will be returned (not
information about the ports or queues).

Returns: If SIT_JOB_INFO passed in **cx**:

ax **SpoolOpStatus** indicating success or failure
of the query.
bx If successful, the handle of a block containing
the **JobStatus** structure for the specified job.
If the job is currently printing and has been
aborted, or if the job can not be found in the
queue, this will be reflected in the returned
ax.

If SIT_QUEUE_INFO passed in **cx**:

ax **SpoolOpStatus** indicating success or failure
of the query. If **dx** was passed -1, only
SPOOL_QUEUE_NOT_EMPTY or
SPOOL_QUEUE_EMPTY will be returned.
bx If successful, the handle of a block containing
an array of print job IDs. This array is
chronological with the active job as the first
element and subsequent jobs following.

Assembly Reference

cx The number of job IDs returned in the block referenced by **bx**.

Destroyed: Nothing.
Library: **spool.def**

SpoolMapToPrinterFont

Map the GEOS font passed to the closest printer font available.

Pass: **cx** **FontID** of the requested font.
dx Point size requested.
bl Pitch value requested.
es Segment address of the **PState** structure defining the printer to be mapped to.
ds Segment address of the device information resource (the locked device resource referenced in the *PS_deviceInfo* field of the **PState** structure pointed to by **es**).

Returns: **cx** **FontID** of the mapped font.
dx New point size (equal or next smaller value, if available; otherwise, average width of the string is computed and the font is treated like a fixed pitch font).
bl New pitch value (equal or next larger pitch value, if available; otherwise, closest smaller value).

Destroyed: Nothing.
Library: **spool.def**

SpoolModifyPriority

Modify the priority of a spool queue's thread.

Pass: **cx** ID of a print job; the thread running this job will have its priority modified.
d1 New thread priority to set.

Returns: **ax** **SpoolOpStatus** value.

Destroyed: Nothing.
Library: **spool.def**

SpoolSetDefaultPageSizeInfo

Set the default page information for the system.

Pass: **ds:si** Address of a **PageSizeReport** structure defining the new default settings for page width, height, layout, and margins.

SpoolSetDefaultPrinter

■ 332

Returns: Nothing.
Destroyed: Nothing.
Library: **spool.def**

■ SpoolSetDefaultPrinter

Set the printer to be used as the system default.

Pass: **ax** Index of the printer to be used as the system default.
Returns: Nothing.
Destroyed: Nothing.
Library: **spool.def**

■ SpoolSetDocSize

Set the document size for a given application.

Pass: **ds:si** Address of a **PageSizeReport** structure defining the page width, height, layout, and margins.
cx If *true* (i.e., non-zero) Document is currently open.
If *false* (i.e., zero) Document is currently closed.
Returns: Nothing.
Destroyed: Nothing.
Library: **spool.def**

■ SpoolSetPaperSizeOrder

Set a new order in which paper sizes should be displayed, for a particular page type.

Pass: **bp** **PageType** value: PT_PAPER, PT_ENVELOPE, PT_LABEL.
ds:si Address of the new array of paper sizes. This array is one byte per element, with each byte containing the corresponding paper size number. These numbers signify the following:
cx 0 – 127 Pre-defined paper size.
128 – 255 User-defined paper size.
Number of entries in the array in **ds:si**.
Returns: Nothing.
Destroyed: Nothing.
Library: **spool.def**

Assembly Reference

■ SpoolUpdateTranslationTable

Initialize the translation table in the passed **PState** structure. This routine is called any time a change in font or country occurs resulting in a change in the ISO substitutions. It is also called once on startup of any print job. This routine is rarely called directly by applications; it is usually called only by the spooler.

Pass: **es** Segment address of the locked **PState** structure.
 dx Handle of the printer driver's extended driver information resource.

Returns: Nothing.

Destroyed: Nothing.

Library: **spool.def**

■ SpoolVerifyPrinterPort

Verify the existence of a printer port.

Pass: **ds:si** Address of a locked **PrintPortInfo** structure giving the printer port type and the port parameters.

Returns: **ax** **SpoolOpStatus** value.

Destroyed: Nothing.

Library: **spool.def**

■ SysGetDosEnvironment

Retrieve the value of a DOS environment variable from the environment buffer.

Pass: **ds:si** Address of the null-terminated name of the variable to get.
 es:di Address of the destination buffer to hold the value.
 cx Maximum number of bytes, including terminating null, to retrieve.

Returns: **CF** Set if environment variable not found, clear otherwise.
 es:di Address of the returned null-terminated value string.

Destroyed: Nothing.

Library: **system.def**



■ SysGetECLevel

Return the current **ErrorCheckingFlags** record set for the system and the block, if ECF_BLOCK_CHECKSUM is set. For full information, see the reference entry for **ErrorCheckingFlags**.

Pass: Nothing.

Returns: **ax** **ErrorCheckingFlags** record.
bx Handle of the error checking block, if ECF_BLOCK_CHECKSUM is set in **ax**.

Destroyed: Nothing.

Library: **ec.def**

■ SysGetInfo

Return general system information dependent on the type passed.

Pass: **ax** **SysGetInfoType** value (one of the following):

- SGIT_TOTAL_HANDLES
Return the total number of handles in the handle table (in **ax**).
- SGIT_HEAP_SIZE
Return the total heap size (in **ax**).
- SGIT_LARGEST_FREE_BLOCK
Return the largest contiguous free block on the heap (in **ax**).
- SGIT_TOTAL_COUNT
Return the total number of ticks GEOS has been running (in **dx:ax**).
- SGIT_NUMBER_OF_VOLUMES
Return the number of registered volumes (in **ax**).
- SGIT_TOTAL_GEODES
Return the total number of geodes loaded (in **ax**).
- SGIT_NUMBER_OF_PROCESSES
Return the total number of process threads running (in **ax**).
- SGIT_NUMBER_OF_LIBRARIES
Return the total number of libraries loaded (in **ax**).
- SGIT_NUMBER_OF_DRIVERS
Return the total number of drivers loaded (in **ax**).

SGIT_CPU_SPEED

Return the CPU speed as a ratio of this CPU relative to a base XT processor times ten (in **ax**).

SGIT_SYSTEM_DISK

Return the handle of the disk on which GEOS resides (in **dx:ax**).

Returns: Depending on the **SysGetInfoType** passed:
ax Return value if the requested value is one word or one byte.
dx:ax Return value if the requested value is size dword.

Destroyed: **dx**, if not returning value.

Library: **sysstats.def**

■ SysLocateFileInDosPath

Search for a file along the path specified in the DOS PATH environment variable.

Pass: **ds:si** Address of the null-terminated file name to search for.
es:di Address of a buffer in which to store the resultant path. The buffer must be locked or fixed and at least DOS_PATH_BUFFER_SIZE bytes long.

Returns: **CF** Set if error, clear if successful.
ax Error code: ERROR_FILE_NOT_FOUND.
es:di Address of the full null-terminated path name, including drive.
bx Disk handle of the disk containing the file.
cx Length of path, including null (in bytes).

Destroyed: Nothing.

Library: **system.def**

■ SysLockBIOS

Gain exclusive access to the BIOS or DOS. Use of this routine is strongly discouraged.

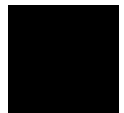
Pass: Nothing.

Returns: Nothing.

Destroyed: Flags are destroyed.

Library: **system.def**

Warning: This is a dangerous routine.



■ SysNotify

Put the **SysNotify** dialog box (the white box with black borders) on the screen. The dialog may have two strings printed in it, which are passed with this routine. This box is typically used for unrecoverable errors, but it may be used for other notifications. Most often an application will use a standard dialog rather than a **SysNotify** for notification messages.

Pass:	ax	SysNotifyFlags record with zero or more of these flags:
		SNF_RETRY Provide "Retry" option to the user.
		SNF_EXIT Provide "Exit Cleanly" option to the user.
		SNF_ABORT Provide "Abort" option to the user.
		SNF_CONTINUE Provide "Continue" option to the user. This implies the notification dialog is not a real error but simply a notification.
		SNF_REBOOT Provide "Reboot" option to the user, executing a dirty shutdown followed by a restart of GEOS. This option will not return.
		SNF_BIZARRE Indicates the notification is unexpected and that the user should be directed to the trouble-shooting guide.
	ds:si	Address of the first string. Must be null-terminated and either locked or fixed. If no string, pass si = zero.
	ds:di	Address of the second string. Must be null-terminated and either locked or fixed. If no string, pass di = zero.
Returns:	ax	Selected option if SNF_RETRY, SNF_ABORT, SNF_CONTINUE, or SNF_EXIT is passed and selected by the user. If SNF_REBOOT passed and selected, this routine does not return.
Destroyed:	Nothing (except SNF_REBOOT).	
Library:	system.def	

■ SysRegisterScreen

Register a new screen with the error mechanism, creating a new window and GState. Use of this routine is strongly discouraged.

Pass:	cx	WindowHandle of the new screen's root window.
	dx	DriverHandle of the video driver for the new screen.
Returns:	Nothing.	
Destroyed:	Nothing.	

Assembly Reference

Library: **system.def**

■ SysSetECLevel

Set the current error-checking flags.

Pass: **ax** **ErrorCheckingFlags** record containing the flags to set. Flags not set will be cleared. See the reference entry for **ErrorCheckingFlags** for full details.

bx Handle of the error-checking block for ECF_BLOCK_CHECKSUM (if any).

Returns: Nothing.

Destroyed: Nothing.

Library: **ec.def**

■ SysSetExitFlags

Set the record of exit flags for use with task-switching drivers. The exit flags are also used for other purposes. Use of this routine is strongly discouraged. For full information on the flags, see the reference entry for **ExitFlags**.

Pass: **bh** **ExitFlags** to clear.

b1 **ExitFlags** to set. Flags in both **bh** and **b1** will be cleared.

Returns: **b1** **ExitFlags** record as set by the routine. The flags are

EF_PANIC	Exit is a panic, so the GEOS.INI file should not be written to disk.
EF_RUN_DOS	Exit is to run a DOS program.
EF_OLD_EXIT	Exit is old-style DOS exit (if GEOS was accidentally run under DOS 1.X).
EF_RESET	Exit should reset the machine.
EF_RESTART	Exit should immediately restart GEOS.

Destroyed: **bh** is destroyed.

Library: **system.def**

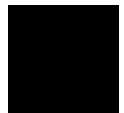
Warning: This is a dangerous routine.

■ SysShutdown

Cause the system to exit based on the **SysShutdownType** passed.

Pass: **ax** **SysShutDownType**. Each shutdown type takes its own parameters, as defined below:

SST_CLEAN Shut down applications cleanly, allowing those that wish to abort the shutdown. This type will cause MSG_META_CONFIRM_SHUTDOWN to be sent out via the



	MANUFACTURER_ID_GEOWORKS: GCNSLT_SHUTDOWN_CONTROL list.
cx:dx	The optr of the object to receive notification once all other objects have acknowledged the shutdown. Pass 0:0 to simply notify the UI in the standard MSG_META_DETACH fashion.
bp	The message to be sent to the cx:dx object. The message will pass cx = 0 if the shutdown request has been denied and cx = non-zero if the shutdown may proceed.
SST_CLEAN_FORCED	Shut down applications cleanly, but do not send MSG_META_CONFIRM_SHUTDOWN (do not allow them to abort the shutdown). Nothing but the shutdown type is passed.
SST_DIRTY	Do not shut down applications, but attempt to exit device drivers and close all open files before shutting down. No notification is sent out.
ds:si	Address of a null-terminated text string giving a reason for the shutdown. This string will be displayed to the user. If no string is passed, pass si = -1.
SST_PANIC	Do not shut down applications, and do not close files; exit device drivers marked GA_SYSTEM. This type of shutdown can be disastrous to the system and should be used only in the most dire of circumstances. Nothing is passed but the shutdown type.
SST_REBOOT	Like SST_DIRTY, this shuts down drivers and closes files; after the shutdown, however, it attempts to warm-boot the machine rather than exit to DOS. Nothing is passed but the shutdown type.
SST_RESTART	Like SST_CLEAN_FORCED in shutdown actions, but reloads the system rather than exiting fully to DOS. Nothing is passed but the shutdown type.
SST_SUSPEND	Suspend system operation in preparation for switching to a new DOS task. This uses the same shutdown confirmation as used by SST_CLEAN (see above).
cx:dx	The optr of the object to receive notification once all other objects have acknowledged the shutdown. Pass 0:0 to simply notify the UI in the standard MSG_META_DETACH fashion.
bp	The message to be sent to the cx:dx object. The message will pass cx = 0 if the shutdown

request has been denied and **cx** = non-zero if the shutdown may proceed.

SST_CONFIRM_START

Called by the recipient of MSG_META_CONFIRM_SHUTDOWN to allow proper ordering of shutdown confirmation dialog boxes. This shutdown type does not actually cause shutdown but grabs “exclusive access” to the user for shutdown confirmation; the caller of this type will block until its turn to confirm comes. If another thread has already aborted the shutdown, the routine will return with **CF** set, indicating the confirmation dialog for the caller should not be put up. Nothing is passed but the shutdown type. After you are done with the confirmation sequence, you must call this routine with SST_CONFIRM_END.

SST_CONFIRM_END

Called after a call to this routine with SST_CONFIRM_START to relinquish “exclusive access” to the user for shutdown confirmation.

cx Zero to deny the shutdown.
Non-zero to allow the shutdown.

Returns: The return value of **SysShutdown** depends on the shutdown type passed:

SST_CLEAN	CF set if another shutdown is in progress.
SST_CLEAN_FORCED	Returns nothing.
SST_DIRTY	Does not return.
SST_PANIC	Does not return.
SST_REBOOT	Does not return.
SST_RESTART	Returns only if could not restart.
SST_SUSPEND	CF set if another shutdown is in progress.
SST_CONFIRM_START	CF set if another caller denied the shutdown.
SST_CONFIRM_END	Returns nothing.

Destroyed: **ax, bx, cx, dx, bp**

Library: **system.def**

■ SysStatistics

Return system performance statistics.

Pass: **es:di** Address of a buffer for the returned **SysStats** structure.

Returns: **es:di** Address of the filled **SysStats** structure. This structure has the following fields:

<i>SS_idleCount</i>	Number of “idle ticks” in the last second.
<i>SS_swapOuts</i>	Outward-bound swap activity.



SS_swapIns Inward-bound swap activity.
SS_contextSwitches Context switches in the last second.
SS_interrupts Interrupts during the last second.
SS_runQueue Number of runnable threads at the end of the last second.

Destroyed: Nothing.

Library: **sysstats.def**

■ SysUnlockBIOS

Relinquish exclusive access to BIOS or DOS, originally gained with **SysLockBIOS**. Use of these routines is strongly discouraged.

Pass: Nothing.

Returns: Nothing.

Destroyed: Nothing (flags preserved).

Library: **system.def**

■ TextAllocClipboardObject

This utility routine allocates a temporary object associated with the clipboard file for purposes of producing a transfer item.

Pass: **al** **VisTextStorageFlags** for object.
 ah Non-zero to create regions for object.
 bx File to associate object with (or zero for clipboard file).

Returns: **bx:si** Handle of created object.

Destroyed: Nothing.

Library: **vTextC.def**

■ TextFindDefaultCharAttr

Given an VisTextCharAttr structure, determine if it is one of the default character attributes.

Pass: **ss:bp** VisTextCharAttr structure.

Returns: **CF** Set if passed character attribute is one of the defaults.
 ax VisTextDefaultCharAttr

Destroyed: Nothing.

Library: **vTextC.def**

■ TextFinishWithClipboardObject

Finish with an object created by TextAllocClipboardObject.

Pass:	^lbx:si	Object
	ax	TextClipboardOption
	cx:dx	owner for clipboard item
	es:di	Name for clipboard item (di = -1 for default)
Returns:	ax	Transfer item handle (if ax passed non-zero)
Destroyed:	Nothing	
Library:	vTextC.def	

■ TextGetSystemCharAttrRun

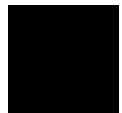
This routine returns the system character attribute run for this object's specific UI.

Pass:	*ds:si	Object to get character attributes for.
	al	Flags to allocate LMem chunk with (if any) (type ObjChunkFlags).
Returns:	CF	Clear if we needed to allocate a chunk, set if a default character attribute run returned.
	ax	New chunk or constant (allocated in passed ds block).
	ds	Updated to point at segment of same block as on entry. Chunk handles in this segment may have moved; be sure to dereference them.
Destroyed:	Nothing.	
Library:	vTextC.def	
Warning:	This routine may resize LMem or object blocks, moving them on the heap and invalidating stored segment pointers to them.	

■ TextSearchInHugeArray

This routine finds an occurrence of a string within another string.

Pass:	ss:bp	Pointer to TextSearchInHugeArrayFrame . TextSearchInHugeArrayFrame struct
		TSIHAF_str1Size dword (?)
		Total length of string to search in (str1).
		TSIHAF_curOffset dword (?)
		Offset (from start of str1) to first char to check
		TSIHAF_endOffset dword (?)
		Offset (from start of str1) to last char to check.



TextSearchInString

■ 342

		Will only match words that start \leq <i>TSIHAF_endOffset</i> . To check to start of string (backward searches only) pass zero To check to end of string (forward searches only) pass <i>TSIHAF_str1.Size-1</i>
	ds:si	<i>TSIHAF_searchFlags</i> SearchOptions
	cx	Pointer to string to search for. This string may contain C_WILDCARD or C_SINGLE_WILDCARD.
		Number of characters in string to search for (or zero if null-terminated).
Returns:	CF	Set if string not found, clear if found.
	dx:ax	Offset to match found.
	bp:cx	Number of characters in match.
Destroyed:	Nothing.	
Library:	vTextC.def	

■ TextSearchInString

This routine finds an occurrence of a string within another string (both strings must be less than 64K in size).

As an example of how to set up the registers for a search, consider the case where you want to search for the string "foo" in "I want some food", but wish to start your search from the "w".

es:bp should point to the "I".

es:di should point to the "w".

es:bx should point to the "d".

Pass:	es:bp	Pointer to first character in string we are searching in.
	es:di	Pointer to character at which to start searching. This is a position within the string to search in, allowing you to find multiple instances.
	es:bx	Pointer to last character to include in search. This is a character within the string to search in. For forward searches, this routine will not match any word that begins after this character, but will match words that start before or at this character and extend beyond it.
	dx	Number of characters pointed to by es:bp (zero if string is null-terminated).
	ds:si	Pointer to string to search for. This string may contain C_WILDCARD or C_SINGLE_WILDCARD.
	cx	Number of characters in string to search for (or zero if null-terminated).
	al	SearchOptions.

Assembly Reference

Returns: **CF** Set if string not found, clear if found.
es:di If found, pointer to start of string found; if not found, pointer to last character checked.
cx Number of characters matched.
Destroyed: Nothing.
Library: **vTextC.def**

■ TextSetSpellLibrary

This routine sets the handle of the spell library to make calls to.

Pass: **bx** Handle of spell library.
Returns: Nothing.
Destroyed: Nothing.
Library: **vTextC.def**

■ ThreadAllocSem

Allocate a semaphore with the initial passed value. The initial value is the number of locks the semaphore can legally have before it causes users to block. This number is nearly always one.

Pass: **bx** Initial value.
Returns: **bx** Handle of the semaphore.
Destroyed: Nothing.
Library: **sem.def**

■ ThreadAllocThreadLock

Allocate a thread lock semaphore; this type of semaphore allows a single thread to lock it multiple times without hitting deadlock.

Pass: Nothing.
Returns: **bx** Handle of the thread lock semaphore.
Destroyed: Nothing.
Library: **sem.def**

■ ThreadAttachToQueue

Attach a thread to an event queue, blocking on the queue until an event is received by it.



ThreadCreate

■ 344

Pass:	bx	Handle of the event queue to attach to. If null handle passed (zero), the caller wants to “re-attach” to the thread’s current queue. This is used when a function will not return but still needs to field events so its application object can detach properly.
	cx:dx	Address of the class that will be bound to the thread. This argument is only used if a null handle is passed in bx .
Returns:	Does not return.	
Destroyed:	Does not return.	
Library:	thread.def	

■ ThreadCreate

Create a new procedural thread. When the thread is started, it begins execution at the routine specified. If you really want to create an event thread (one that runs objects), send `MSG_PROCESS_CREATE_EVENT_THREAD` to your process object instead.

You can call the C stub for this routine, `THREADCREATE`, directly if you like; this allows passing a virtual segment pointer to the thread’s execution routine. The routine may also return an exit code in **ax**. When calling `THREADCREATE`, however, you must pass the arguments on the stack. The kernel will take care of everything else, including calling **ThreadDestroy** with the proper exit code.

Pass:	di	Size of stack for new thread. For most threads, 1024 is a good stack size. Threads that do no file-related work can probably use 512 bytes. If the thread will run objects that use keyboard navigation (e.g. dialog boxes), you may need to make it 3072.
	bp	Geode handle of the owner of the thread. If you’re in the application’s process thread, you can call GeodeGetProcessHandle to get the right owner value. If you’re in a UI thread and have ds or es pointing to a non-shared LMem block owned by the application, you can <code>mov bx, ds:[LMBH_handle]</code> and call MemOwner .
	al	Priority number for the new thread. Usually one of PRIORITY_TIME_CRITICAL PRIORITY_HIGH PRIORITY_UI PRIORITY_FOCUS PRIORITY_STANDARD PRIORITY_LOW PRIORITY_LOWEST

Assembly Reference

	bx	Value to pass to the new thread (the startup routine, defined below) in the cx register.
	cx:dx	Address of the thread's startup routine. This routine will be executed by the thread; it may send messages and call other routines, but when it is done executing, it jumps to ThreadDestroy and kills the thread. The routine's parameters are listed below:
	ds = es	Owning geode's dgroup segment.
	cx	Value passed in bx to ThreadCreate .
	dx, y	Zero.
	si	Handle of the owning geode.
	di	LCT_NEW_CLIENT_THREAD.
	flags, ax, bp	Undefined.
Returns:	CF	Set if error, clear otherwise. The error, quite infrequent, is if the kernel could not allocate enough fixed stack space for the new thread.
	bx	Handle of the new thread.
	cx	Zero.
Destroyed:	ax, dx, si, di, bp	
Library:	thread.def	

■ ThreadDestroy

Exit the current process or thread and destroy it.

Pass:	cx	Exit code indicating the reason for or method of exit. This exit code should be defined by the application and should be meaningful to all other threads of the application.
	dx:bp	The optr of the object to receive MSG_META_ACK after the thread is destroyed.
	si	A word of data to pass with MSG_META_ACK. This message takes dx:bp as the optr of the source of the acknowledgment, but only the dx portion is used in response to ThreadDestroy .
Returns:	Does not return.	
Destroyed:	Does not return.	
Library:	thread.def	

■ ThreadFreeSem

Free a semaphore allocated with **ThreadAllocSem**.

Pass:	bx	Handle of semaphore as returned by ThreadAllocSem .
-------	-----------	--



ThreadFreeThreadLock

■ 346

Returns: Nothing.
Destroyed: Nothing.
Library: **sem.def**

■ ThreadFreeThreadLock

Free a semaphore allocated with **ThreadGrabThreadLock**.

Pass: **bx** Handle of semaphore as returned by **ThreadAllocSem**.
Returns: Nothing.
Destroyed: Nothing.
Library: **sem.def**

■ ThreadGetDGroupDS

Load the ds register with the segment of the current thread's dgroup.

Pass: Nothing.
Returns: **ds** Segment of the caller thread's dgroup.
Destroyed: Nothing.
Library: **resource.def**

■ ThreadGetInfo

Return information about a thread, depending on the type passed.

Pass: **ax** **ThreadGetInfoType** value:
TGIT_PRIORITY_AND_USAGE Return the thread's recent CPU usage in the high byte of the returned word and the thread's priority level in the low byte.
TGIT_THREAD_HANDLE Return the thread's handle.
TGIT_QUEUE_HANDLE Return the handle of the thread's queue.
bx Handle of the thread to get information on, or zero for the caller thread.
Returns: **ax** Value dependent on the **ThreadGetInfoType** passed.
Destroyed: Nothing.
Library: **thread.def**

Assembly Reference

■ ThreadGrabThreadLock

Grab a thread lock (like doing a “P” on a semaphore). A thread that grabs a thread lock it already holds will not deadlock. A thread that grabs a thread lock held by another thread will block until the thread lock is available.

Pass: **bx** Handle of the thread lock as returned by **ThreadAllocThreadLock**.

Returns: Nothing.

Destroyed: Nothing.

Library: **sem.def**

■ ThreadHandleException

Define a handler function so a thread may handle one of the processor exceptions.

Pass: **ax** **ThreadException** to be handled by the routine:
 TE_DIVIDE_BY_ZERO
 TE_OVERFLOW
 TE_BOUND
 TE_FPU_EXCEPTION
 TE_SINGLE_STEP
 TE_BREAKPOINT

bx Handle of the thread that will handle the exception, or zero to specify the current (caller) thread.

cx:dx Address of the fixed-memory handler routine. Pass 0:0 to return to using the kernel's default handler for the exception.

Returns: **bx** Handle of the thread that was modified to run the routine.

Destroyed: Nothing.

Library: **thread.def**

■ ThreadModify

Change a thread's base priority, and/or set the thread's recent CPU usage to zero.

Pass: **bx** Handle of the thread to be modified, or zero to modify the current (caller) thread.

al **ThreadModifyFlags** record, with one or both of the following:
 TMR_BASE_PRIO Modify the thread's base priority.
 TMR_ZERO_USAGE Set the thread's recent CPU usage to zero.



ThreadPrivAlloc

■ 348

	a1	The new base priority, if TMR_BASE_PRIO is set in ah .
Returns:	bx	Handle of the thread modified.
Destroyed:	ax	
Library:	thread.def	

■ ThreadPrivAlloc

Allocate a block of contiguous words in the thread's private data area.

Pass:	cx	Number of words to be allocated.
	bx	Geode handle of the geode that will “own” the block.
Returns:	CF	Set if no block large enough for allocation.
	bx	Offset to the start of the range (token for use with other private data management routines).
Destroyed:	Nothing.	
Library:	thread.def	

■ ThreadPrivFree

Free a range of thread-private space owned by the geode. This space must have been allocated with **ThreadPrivAlloc**.

Pass:	bx	Offset to the words being freed (as returned by ThreadPrivAlloc).
	cx	Number of words to be freed.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	thread.def	

■ ThreadPSem

Grab a semaphore (perform a “P” operation on it). If another thread has the semaphore, the caller will block until the semaphore is available. If the calling thread has the semaphore, the thread will deadlock. This routine provides no deadlock checking.

Pass:	bx	Handle of the semaphore to be grabbed.
Returns:	ax	SemaphoreError value: SE_NO_ERROR SE_PREVIOUS_OWNER_DIED
Destroyed:	Nothing.	
Library:	sem.def	

Assembly Reference

■ ThreadPTimedSem

Grab a semaphore as with **ThreadPSem**, except return an error if the semaphore is not available within a certain time limit. If the timeout is returned, the caller should *not* proceed with the protected action but should take other action.

Pass: **bx** Handle of the semaphore to be grabbed.
 cx Number of ticks before timeout.

Returns: **ax** **SemaphoreError** value:
 SE_TIMEOUT
 SE_NO_ERROR
 SE_PREVIOUS_OWNER_DIED

Destroyed: Nothing.

Library: **sem.def**

■ ThreadReleaseThreadLock

Release a thread lock grabbed with **ThreadGrabThreadLock**. A thread should call this routine once and only once for each time it grabbed the thread lock.

Pass: **bx** Handle of the thread lock semaphore.

Returns: Nothing.

Destroyed: Nothing.

Library: **sem.def**

■ ThreadVSem

Release a semaphore (perform a “V” operation on it). This routine should be called once and only once for each call to **ThreadPSem** on the semaphore by the calling thread.

Pass: **bx** Handle of the semaphore to be released.

Returns: **ax** **SemaphoreError** value:
 SE_NO_ERROR
 SE_PREVIOUS_OWNER_DIED

Destroyed: Nothing.

Library: **sem.def**

■ TimerGetCount

Return the system time counter, reflecting the total number of ticks since GEOS was started.



TimerGetDateAndTime

■ 350

Pass: Nothing.

Returns: **ax** Low word of 32-bit time counter.
bx High word of 32-bit time counter.

Destroyed: Nothing (flags preserved).

Library: **timer.def**

■ TimerGetDateAndTime

Return the current date and time.

Pass: Nothing.

Returns: **ax** Year (zero-based integer where 0 = 1980).
b1 Month (1 through 12).
bh Day (1 through 31).
c1 Day of the week (zero-based integer where 0 = Sunday).
ch Hours (0 through 23).
d1 Minutes (0 through 59).
dh Seconds (0 through 59).

Destroyed: Nothing.

Library: **timedate.def**

■ TimerSetDateAndTime

Set the system's date and time. This routine should not normally be called by any application other than the GEOS Preferences Manager.

Pass: **c1** **SetDateTimeParams** record. One or both of
TIME_SET_DATE Set the year, month, and day.
TIME_SET_TIME Set the hour, minute, and second.

ax Year (zero-based integer where 0 = 1980).
b1 Month (1 through 12).
bh Day (1 through 31).
ch Hours (0 through 23).
d1 Minutes (0 through 59).
dh Seconds (0 through 59).

Returns: Nothing.

Destroyed: **ax, bx, cx, dx**

Library: **timedate.def**

Assembly Reference

■ TimerSleep

Block the calling thread for the given length of time. This routine is not an acceptable substitute for the use of semaphores when synchronizing threads.

Pass: **ax** Number of ticks to sleep (sixty ticks per second).
Returns: Nothing.
Destroyed: Nothing.
Library: **timer.def**

■ TimerStart

Start an event or routine timer, either continual or one-shot. A routine timer calls a specified routine when time is up; an event timer sends a specified message. A one-shot timer counts only once; a continual timer counts until stopped (with **TimerStop**), sending the message or calling the routine each time the specified interval has passed. You can also start a millisecond timer.

For routine timers, the routine called must have the following specifications:

Passed: **ax** The word passed in **dx** to **TimerStart**.
cx:dx The tick count as returned by **TimerGetCount**.

Return: Nothing.

May Destroy: **ax, bx, cx, dx, si, di, bp, ds, es**

For event timers, the message sent will carry the following parameters and may return nothing:

Passed: **ax** Message number sent.
cx:dx The tick count as returned by **TimerGetCount**.
bp The Timer ID for one-shot timers, or
The timer interval, for continual timers.

For timers of type **TIMER_MS_ROUTINE_ONE_SHOT**, the routine must have the following specifications:

Passed: **ax** The word passed in **dx** to **TimerStart**.
Interrupts will be off.

Return: Nothing.

May Destroy: **ax, bx, si, ds**

Pass: **al** **TimerType** value:
TIMER_ROUTINE_ONE_SHOT
Start a one-shot routine timer.



TimerStartSetOwner

■ 352

		TIMER_ROUTINE_CONTINUAL	Start a continual routine timer.
		TIMER_EVENT_ONE_SHOT	Start a one-shot event timer.
		TIMER_EVENT_CONTINUAL	Start a continual event timer.
		TIMER_MS_ROUTINE_ONE_SHOT	Start a one-shot routine timer with millisecond accuracy.
	bx:si	The optr of the object to receive the event message (in the case of TIMER_EVENT...), or The far pointer to the routine to be invoked (in the case of TIMER_ROUTINE...).	
	cx	Number of ticks to count until first timeout, for all timer types except TIMER_MS_ROUTINE_ONE_SHOT. For this type, cx contains the number of milliseconds to count.	
	dx	Message to send, for event timers, or A word of data passed to the routine in ax for routine timers.	
	di	Ticks between timeouts (timer interval), for continual timers.	
Returns:	ax	Timer ID number (needed for TimerStop).	
	bx	Timer handle of the timer.	
Destroyed:	Nothing. Interrupts are in the same state as before.		
Library:	timer.def		

■ TimerStartSetOwner

This routine is exactly the same as **TimerStart**, above, except that it allows the caller to set the timer's owner. All other aspects of the timer are the same. See **TimerStart** for complete details.

Pass:	See TimerStart .
	bp Geode handle of the new owner of the timer.
Returns:	See TimerStart .
Destroyed:	See TimerStart .
Library:	timer.def

■ TimerStop

Stop a timer and remove it. This routine is typically called for continual timers. Note that a continual event timer may have sent one or more events that may be in the recipient's event queue; therefore, you can not assume that all timer notifications have been handled when this routine is called.

Assembly Reference

Pass:	bx ax	Handle of the timer to be removed. Timer ID as returned by the TimerStart routines. Pass zero for continual timers.
Returns:	CF	Set if the timer was not found, clear otherwise.
Destroyed:	ax, bx	
Library:	timer.def	

■ TocAddDisk

Add a disk to the disk array

Pass:	ds:si cx:dx	Full name of disk TocDiskStruct structure
Returns:	bx	Disk token (element number in array).
Destroyed:	Nothing	
Library:	config.def	

■ TocCreateNewFile

create a new TOC file in the current working directory. All subsequent TOC routines will operate on this new file.

Pass:	Nothing.	
Returns:	CF ax	Set on error; clear on success. On error, a FileError value; otherwise, destroyed.
Destroyed:	Nothing.	
Library:	config.def	

■ TocDBlock

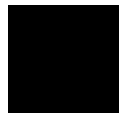
Lock a DB item in the config library's TOC file.

Pass:	ax:di	DBItem to lock.
Returns:	*ds:si	Item.
Destroyed:	Nothing.	
Library:	config.def	

■ TocFindCategory

Find a category in the Toc file.

Pass:	es:di	Buffer of size TocCategoryStruct to be filled in.
-------	--------------	--



TocGetFileHandle

■ 354

Returns: CF Set if not found, otherwise clear.
Destroyed: Nothing.
Library: **config.def**

■ TocGetFileHandle

Return the TOC file handle.

Pass: Nothing.
Returns: **bx** TOC file handle.
Destroyed: Nothing.
Library: **config.def**

■ TocNameArrayAdd

Add an element to a TOC name array.

Pass: **ax:di** **DBItem** (nameArray) if **ax** = 0, then the map item will be used.
 cx:dx buffer containing data to add
 ds: fptr to name to search for
Returns: **bx** Element number.
Destroyed: Nothing.
Library: **config.def**

■ TocNameArrayFind

Find a name in the passed name array.

Pass: **ax:di** **DBItem** (nameArray) in which to find name. If **ax** is zero, then the map item will be used.
 cx:dx Buffer to fill with data. If **cx** is zero, will not return any data.
 ds:si fptr to name to search for.
Returns: **bx** Name token, or CA_NULL_ELEMENT if not found.
Destroyed: Nothing.
Library: **config.def**

■ TocNameArrayGetElement

Return data about an element, given its number.

Pass: **ax:di** **DBItem** (nameArray) in which to find name. If **ax** is zero, then the map item will be used.

Assembly Reference

	bx	Element number
	cx:dx	Buffer to fill with data. If cx is zero, will not return any data.
Returns:	ax	Length of data returned in cx:dx .
Destroyed:	Nothing.	
Library:	config.def	

■ TocSortedNameArrayAdd

Add an element to the name array, inserting it in the proper order.

Pass:	di	VM handle of name array
	ds:si	name
	cx:dx	data to add, pass cx =zero if no data
	bx	NameArrayAddFlags
Returns:	ax	New element number
Destroyed:	Nothing	
Library:	config.def	

■ TocSortedNameArrayFind

Find a name in a sorted name array.

Pass:	di	VM handle of SortedNameArray
	ds:si	name to find
	cx:dx	buffer for data (cx = null to not store data)
	bl	SortedNameArrayFindFlags
Returns:	CF	Set if found; clear otherwise.
	ax	Element number if found; otherwise element number where element would appear if it were in the list.
Destroyed:	Nothing.	
Library:	config.def	

■ TocUpdateCategory

Create the category if it doesn't exist, and update the file lists by scanning the current directory for files.

Pass:	ss:bp	TocUpdateCategoryParams structure
	CWD	Current working directory is directory where files reside.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	config.def	



TokenDefineToken

■ 356

■ TokenDefineToken

This routine adds a new token and moniker list to the token database. If the token already exists in the token database, the old token will be replaced. This routine may only be called by the thread capable of locking the clock which the passed Moniker or MonikerList resides in.

Pass: **ax, bx, si** Six bytes of token. The **ax** and **bx** registers contain the four characters of the token and **si** contains the manufacturer ID.
 cx:dx Handle:chunk of moniker list
 bp TokenFlags.

Returns: Nothing.

Destroyed: Nothing. This routine may legally move locked LMem blocks (token database items), invalidating any stored segment pointers to them.

Library: **token.def**

■ TokenExitTokenDB

Close the token database file.

Pass: Nothing.

Returns: Nothing.

Destroyed: Nothing

Library: **token.def**

■ TokenGetTokenInfo

Get information about a token.

Pass: **ax, bx, si** Six bytes of token.

Returns: **CF** Clear if token exists in database, set otherwise.
 bp **TokenFlags** for the token (if found).

Destroyed: Nothing.

Library: **token.def**

■ TokenInitTokenDB

Open the local token database file read/write and, if the path for a globally shared token database appears in the .INI file, open that file shared-multiple read-only.

Pass: Nothing.

Returns: **CF** Set on error; clear on success.

Assembly Reference

dx On error, this will be a **TokenError** value, one of
 ERROR_OPENING_SHARED_TOKEN_DATABASE_FILE,
 ERROR_OPENING_LOCAL_TOKEN_DATABASE_FILE, and
 BAD_PROTOCOL_IN_SHARED_TOKEN_DATABASE_FILE

Destroyed: Nothing.

Library: **token.def**

■ TokenListTokens

Make a list of the tokens in the token.db file and return it in a memory block as an array of GeodeToken structures. Along with the list, the number of items in the list is returned. Because groups are mixed in with tokens, we have to do a preliminary pass to count the tokens, then allocate space and run through again grabbing tokens.

Pass: **ax** Zero if only tokens with GString monikers are requested.
 Non zero to request all monikers.
 bx Number of bytes to reserve for header in created block. If
 zero, token list will begin at top of returned block.
 cx **ManufacturerID** of tokens for list, if the
 TRF_ONLY_PASSED_MANUFID is set in **ax**

Returns: **bx** Handle of global memory block containing the list.
 ax Number of items in list.

Destroyed: Nothing.

Library: **token.def**

■ TokenLoadMoniker

This routine loads a specified token's moniker.

If you ask that this routine create an LMem block for you, and **ds** or **es** is pointing to that LMem block, you must fix **ds** or **es** yourself. E.g.:

push ds:[LMBH_handle] ; save LMem block handle

(set up params)

call TokenLoadMoniker

pop bx

call MemDerefDS

Pass: **ax, bx, si** Six bytes of token.
 dh DisplayType.
 cx:di Moniker destination.
 If **cx** is zero, then a new global memory chunk will be
 allocated for the moniker.
 If **di** is zero, then **cx** is interpreted as the handle of the



TokenLoadToken

■ 358

		LMem block in which to allocate an LMem chunk for the moniker.
		Otherwise, cx:di is interpreted as the address to copy the moniker to.
	ss:bp	Search flags and buffer size. Note that these arguments will be removed from the stack by this routine.
Pass on stack:		(Pushed in this order):
	(word) VisMonikerSearchFlags .	
	(word) Size of buffer	
Returns:	CF	Clear if token exists in database, set otherwise.
	cx	Number of bytes in moniker.
	di	Global memory block handle, or LMem chunk handle.
Destroyed:	Nothing.	
Library:	token.def	

■ TokenLoadToken

Load **TokenEntry** structure for a token into a buffer.

If you ask that this routine create an LMem block for you, and **ds** or **es** is pointing to that LMem block, you must fix **ds** or **es** yourself. E.g.:
push ds:[0] ; save LMem block handle
call TokenLoadToken
pop bx
call MemDerefDS

Pass:	ax, bx, si	Six bytes of token.
	cx:di	Moniker destination.
		If cx is zero, then a new global memory chunk will be allocated for the moniker.
		If di is zero, then cx is interpreted as the handle of the LMem block in which to allocate an LMem chunk for the moniker.
		Otherwise, cx:di is interpreted as the address to copy the moniker to.
Returns:	CF	Clear if token exists in database, set otherwise.
	cx	Number of bytes in TokenEntry.
	di	Global memory block handle, or LMem chunk handle.
Destroyed:	Nothing.	
Library:	token.def	

■ TokenLockTokenMoniker

Lock moniker for drawing.

Assembly Reference

Pass:	cx:dx ax	Group:Item for drawing. Zero if token is in shared token DB file; non-zero if token is in local token DB file.
Returns:	*ds:bx	Segment:Chunk of moniker.
Destroyed:	Nothing.	
Library:	token.def	

■ TokenLookupMoniker

Get the specific moniker for a token, given display type and other attributes.

Pass:	ax, bx, si dh bp	Six bytes of token. The ax and bx registers hold the token characters, and si holds the manufacturer ID. DisplayType VisMonikerSearchFlags (VSMF_COPY_CHUNK and VMSF_REPLACE_LIST ignored).
Returns:	CF cx:dx ax	Clear if token exists in database, set otherwise. Group:Item of moniker (if found). Zero if token found in shared token DB file; non-zero if found in local token DB file.
Destroyed:	Nothing.	
Library:	token.def	

■ TokenRemoveToken

Get information about a token.

Pass:	ax, bx, si	Six bytes of token. The ax and bx registers hold the token characters, and si holds the manufacturer ID.
Returns:	CF	Clear if token successfully deleted.
Destroyed:	Nothing.	
Library:	token.def	

■ TokenUnlockTokenMoniker

This routine unlocks a moniker that had been locked with **TokenLockMoniker()**. Pass a pointer to the locked moniker, as returned by the locking routine.

Pass:	ds	Segment of moniker.
Returns:	Nothing.	
Destroyed:	Nothing.	



UserAddAutoExec

■ 360

Library: **token.def**

■ UserAddAutoExec

Add an application to the list of those that are to be loaded when the system is booted. This works with the “execOnStartup” field of the initialization field. Welcome is an example of an application that might be executed on startup.

Pass: **ds:si** Name of application to be loaded on startup. The geode should reside in SP_APPLICATION or SP_SYS_APPLICATION.

Returns: Nothing.

Destroyed: Nothing.

Library: **ui.def**

■ UserAddItemToGroup

Add a font GenItem to the list, set it usable and set its action/data

Pass: ***ds:si** Parent

bx Handle of parent block

dx Chunk of font entry (i.e. **^lGenListEntry**)

cx Action/data for entry (**FontID**)

Returns: **ds** Updated to point at segment of same block as on entry.

Destroyed: Nothing.

Library: **ui.def**

■ UserAllocObjBlock

Allocate a block on the heap, to be used for holding UI objects.

Pass: **bx** Handle of thread to run block (0 for current thread).

Returns: **bx** Handle of block.

Destroyed: Nothing.

Library: **ui.def**

■ UserCallApplication

Call application object of process which owns block passed.

Pass: **ax** Message to send to application.

cx, dx, bp Data to send on to application.

ds Any object block (for fixup).

Assembly Reference

Returns:	CF	If there was no call, or if message was not handled, will return clear. Otherwise, the message handler will set the return value.
	ds	Updated to point at segment of same block as on entry.
Destroyed:	Nothing.	
Library:	ui.def	

■ UserCallFlow

Call the UI flow object.

Pass:	ax	Message to pass to system object.
	di	Flags as in ObjMessage0 .
	cx, dx, bp	Data to pass to message handler.
Returns:	CF	Set by message handler.
	di, cx, dx, bp	Data returned by message handler.
	ds, es	Updated segments (depending on flags passed in di).
Destroyed:	Nothing.	
Library:	ui.def	

■ UserCheckAcceleratorChar

Returns carry set if passed an accelerator-type character.

Pass:	c1	Character.
	d1	CharFlags .
	dh	ShiftState .
	bp	(high byte) scan code: (low byte) ToggleState .
Returns:	CF	Set if accelerator character.
Destroyed:	Nothing.	
Library:	ui.def	

■ UserCheckInsertableCtrlChar

Checks passed key to see if it is a control character that maps to an insertable ASCII character.

Pass:	cx	Character value.
	d1	CharFlags .
	dh	ShiftState .
	bp	Low byte: ToggleState
Returns:	Nothing.	



UserCopyChunkOut

■ 362

Destroyed: Nothing.

Library: **ui.def**

■ UserCopyChunkOut

This routine copies part of a local memory chunk to another location.

Pass:	ds:bp	Pointer to source chunk.
	cx	Zero to request that a global memory block be allocated, and that the chunk's contents be copied to this block; Otherwise, meaning of cx depends on dx .
	dx	Zero to specify that cx should be treated as the handle of an LMem block to copy to; Otherwise, cx:dx will be treated as the address to copy to.
	ax	Offset specifying where in chunk to start copying. Use zero to start at the beginning.
	bx	Flag specifying whether a null terminator should be appended at the end of the copy. One to request a null terminator, zero to omit it.
	di	Offset specifying where to stop copying. This is an offset in chunk past the end. Use zero to copy to the end.
Returns:	ax	Chunk handle, if one created. Note that if created, the copied chunk is marked as dirty. The caller must clear the flags to set it otherwise.
	cx	Number of characters copied (not including added null terminator, if any).
	ds	Updated to point at segment of same block as on entry (only relevant if copying to lmem chunk).
Destroyed:	bx, dx, di, bp.	
Library:	ui.def	

■ UserCreateDialog

Duplicates a template dialog block, attaches the dialog to an application object, and sets it fully usable. The dialog at this point may be used with **UserDoDialog()**. The dialog should be removed and destroyed by the caller when no longer needed.

Pass:	bx:si	Template object block, chunk offset of GenInteractionClass within it to invoke. The block must be sharable, read-only, and the top GenInteraction must not be linked into any generic tree.
Returns:	bx:si	Created, fully usable dialog (or zero if unable to create).

Assembly Reference

Destroyed: Nothing.

Warning: This routine may resize LMem and/or object blocks, moving them on the heap and invalidating stored segment pointers to them.

Library: **ui.def**

■ UserCreateInkDestinationInfo

This routine creates an **InkDestinationInfo** structure to be returned with MSG_META_QUERY_IF_PRESS_IS_INK.

Pass: **cx, dx** optr
bp gstate for ink to be drawn through (or zero)
ax width/height of ink (or zero for default)
bx:di virtual fptr of callback routine (to be passed to **ProcCallFixedOrMovable**) to determine whether a stroke is a gesture or not (BX:DI=0 if none)

Returns: **bp** handle of an **InkDestinationInfo** structure (or zero if couldn't allocate).

Destroyed: Nothing

Library: **ui.def**

■ UserCreateItem

Create a GenItem for a given string.

Pass: **es:di** ptr to font string (NULL terminated)
***ds:si** parent
bx block of parent
ds pointing to a "fixupable" block
dx mask OCF_IGNORE_DIRTY if created entry should be marked ignore dirty, 0 if not

Returns: **dx** lmem handle of new list entry
ds updated to point at segment of same block as on entry

Destroyed: Nothing.

Library: **ui.def**

■ UserDestroyDialog

Duplicates a template dialog block, attaches the dialog to an application object, and sets it fully usable. The dialog at this point may be used with **UserDoDialog()**. The dialog should be removed and destroyed by the caller when no longer needed.



UserDiskRestore

■ 364

Pass: **bx:si** Dialog to destroy as object block:chunk offset.
Returns: Nothing.
Destroyed: Nothing.
Library: **ui.def**

■ UserDiskRestore

Front-end for **DiskRestore** that automatically passes a callback function to prompt for the disk, if **DiskRestore** can't do it by itself.

Pass: **ds:si** Buffer to which the disk handle was saved.
Returns: **CF** Set if disk could not be restored; clear if disk restored.
ax On error, **DiskRestoreError**; on success, handle of disk for this invocation of GEOS.
Destroyed:
Library: **ui.def**

■ UserDoDialog

This routine allows the application to invoke a dialog (GenInteraction set up to be a modal dialog) and block until the user responds. The passed object must be linked into a generic tree and be fully usable. Where possible, use MSG_GEN_INTERACTION_INITIATE instead. All objects making up the dialog must reside within a single block. The dialog must be self-contained. I.e. it may not rely on messages sent or called on objects outside of itself.

Pass: **bx:si** The optr of the GenInteractionClass object to invoke. Must be linked into a generic tree and be fully usable before this routine may be called on it.
Returns: **ax** InteractionCommand response value.
Destroyed: Nothing.
Library: **ui.def**

■ UserGetDefaultMonikerFont

Get the UI moniker font, size for the passed object.

Pass: **ds:si** Object to get the display type for (for future expansion possibilities).
Returns: **cx** **FontID**.
dx Point size.
Destroyed: Nothing.

Assembly Reference

Library: **ui.def**

■ UserGetDisplayType

Get the display type for the passed object. Currently reads the global variable **uiDisplayType**, set by GenScreen in MSG_GEN_SCREEN_SET_VIDEO_DRIVER.

Pass: ***ds:si** Object to get the display type for (for future expansion possibilities)

Returns: **ah** **DisplayType**
al *flag*: true (i.e., non-zero) if **uiDisplayType** has been set (should only be false before first screen object is put up)

Destroyed: Nothing.

Library: **ui.def**

■ UserGetInitFileCategory

Utility routine to fetch .ini category for an object. Test application optimization flag for single category, to avoid recursive search if possible.

Pass: ***ds:si** Object needing .ini category
cx:dx Pointer to buffer needing filled

Returns: **CF** Set if buffer filled.

Destroyed: Nothing.

Library: **ui.def**

■ UserGetKbdAcceleratorMode

Returns keyboard accelerator mode status.

Pass: Nothing.

Returns: **ZF** Clear if accelerator mode on; set if off.

Destroyed: Nothing.

Library: **ui.def**

■ UserGetOverstrikeMode

Returns overstrike mode status.

Pass: Nothing.

Returns: **ZF** Clear if overstrike mode on; set if off.

Destroyed: Nothing.



UserGetSpecUIProtocolRequirement

■ 366

Library: **ui.def**

■ UserGetSpecUIProtocolRequirement

Returns any protocol number that should be passed to **GeodeUseLibrary** in any attempt to load a specific user interface for use with this geode.

Pass: Nothing.

Returns: **bx** Major protocol number.
ax Minor protocol number.

Destroyed: Nothing.

Library: **ui.def**

■ UserHaveProcessCopyChunkIn

This routine figures out which process runs the destination block and sends MSG_PROCESS_COPY_CHUNK_IN to it.

Pass: **dx** Number of bytes on stack
ss:bp Pointer to **CopyChunkInFrame** structure.

Returns: **ax** Chunk handle of created chunk
cx Number of bytes copied over
es,ds Updated if they moved (were the destination block)

Destroyed: Nothing.

Library: **ui.def**

■ UserHaveProcessCopyChunkOut

This routine figures out which process runs the source block and sends MSG_PROCESS_COPY_CHUNK_OUT to it. The source optr must be in an object block (the **otherInfo** field must be a thread handle).

Pass: **dx** Number of bytes on stack
ss:bp Pointer to **CopyChunkOutFrame** structure.

Returns: **ax** Chunk handle of created chunk/block handle (if any)
cx Number of bytes copied
es,ds Updated if they moved (were the destination block)

Destroyed: Nothing.

Library: **ui.def**

■ UserHaveProcessCopyChunkOver

This routine figures out which process runs the destination block and sends MSG_PROCESS_COPY_CHUNK_OVER to it.

Assembly Reference

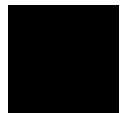
Pass:	dx	Number of bytes on stack
	ss:bp	Pointer to CopyChunkOverFrame structure.
Returns:	ax	Chunk handle of created chunk/block handle (if any)
	cx	Number of bytes copied
	es,ds	Updated if they moved (were the destination block)
Destroyed:	Nothing.	
Library:	ui.def	

■ UserLoadApplication

Loads a GEOS application. Changes to standard application directory before attempting GeodeLoad on filename passed. Stores the filename being launched into the **AppLaunchBlock**, so that information needed to restore this application instance will be around later if needed.

Ownership of the launch block is transferred to the new geode and will be freed by it. If the application cannot be loaded, the block will be freed here. On no account should a passed AppLaunchBlock be referred to after this function returns.

Pass:	ah	AppLaunchFlags (zero for default). The ALF_SEND_LAUNCH_REQUEST_TO_UI_TO_HANDLE should be set if the actual launch should be done later by the UI, in a safe memory situation (no error code will be returned in this case). If this flag is clear, then the caller should be calling from a fixed memory space, such that none of their movable code segments are locked. This is to provide the most favorable conditions for the new application to be loaded in.
	cx	Application attach mode message. This may be one of the following: Zero: Use <i>ALB_appMode</i> in AppLaunchBlock passed, or if there is none, use the default mode. If this is non-zero, any <i>ALB_appMode</i> in the launch block is overridden. MSG_GEN_PROCESS_RESTORE_FROM_STATE : State file must be passed, no data should be passed. MSG_GEN_PROCESS_OPEN_APPLICATION : State file should normally not be passed, although one could be to accomplish UI templates. A data file may be passed into the application as well. MSG_GEN_PROCESS_OPEN_ENGINE : State file normally should not be passed. The data file on which the engine will operate must be passed. If zero, the default data file should be used (this is enforced by the application, not GenProcessClass).



UserLoadExtendedDriver

■ 368

	dx	Block handle of structure AppLaunchBlock (must be sharable) or zero for default case. This default case results in a mode of MSG_GEN_PROCESS_OPEN_APPLICATION, no data file, no template state file, launch to take place in the current default field, current directory is the data directory passed to the application.
	ds:si	If the pathname is not in the AppLaunchBlock , then this may be a pointer to the absolute path of the file to load, or the file name of a file in either SP_APPLICATION or SP_SYS_APPLICATION.
	si	If the full pathname, filename, and diskhandle are stored in the AppLaunchBlock , then si is -1.
	bx	If the path is specified in ds:si , then bx contains the disk handle, or a standard path (SP_APPLICATION or SP_SYS_APPLICATION). Otherwise, this register is ignored.
Returns:	bx	Geode process handle.
	CF	Clear if no error; set if error.
	ax	If no error, segment of geode's core block. If there was an error, this register will hold the error code, of type GeodeLoadError .
Destroyed:	Nothing.	
Library:	ui.def	

■ UserLoadExtendedDriver

Load an extended driver given the category of the .INI file in which to find the "device" and "driver" keys for the thing.

Pass:	ax	StandardPath enum for directory to look in
	bx	Value to pass in bx to DRE_TEST_DEVICE and DRE_SET_DEVICE; may be garbage if driver being loaded doesn't expect anything.
	cx:dx	Protocol number expected
	ds:si	Category
Returns:	CF	Clear if successful; set on error.
	bx	On success, handle of loaded and initialized driver.
	ax	On error, GeodeLoadError .
Destroyed:	cx, dx, di, si.	
Library:	ui.def	

■ UserMessageIM

Send a message to the input manager.

Assembly Reference

Pass:	ax	Message to send.
	di	Flags as in ObjMessage() .
	cx, dx, bp	Data to pass with message.
Returns:	CF	Returned as set by message handler.
	di, cx, dx, bp	Data returned by message handler.
	ds, es	Updated segments (depending on flags passed in di).
Destroyed:	Nothing.	
Library:	ui.def	

■ UserRegisterForTextContext

Registers the passed object to receive context data.

Pass:	^lcx:dx	Object to register
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	ui.def	

■ UserRemoveAutoExec

Remove an application from the list of those to be launched on start-up.

Pass:	ds:si	Name of application to remove.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	ui.def	

■ UserScreenRegister

Register another screen for GenScreen.

Pass:	cx	Handle of root window for screen
	dx	Handle of video driver for screen
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	ui.def	

■ UserSendToApplicationViaProcess

Call the application object, but only after a method has been passed fully through the owning application's process.



UserSetDefaultMonikerFont

■ 370

Pass:	*ds:si	Generic object whose application object we'd like to send a method to delayed via stack
	ax	Message to send to application object
	cx, dx, bp	Message's arguments.
Returns:	ds	Updated to point at same segment of same block as on entry.
Destroyed:	Nothing.	
Library:	ui.def	

■ UserSetDefaultMonikerFont

Set the font and font size to use when drawing UI monikers.

Pass:	cx	FontID.
	dx	Point size.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	ui.def	

■ UserSetOverstrikeMode

Sets the overstrike mode in the initialization file.

Pass:	al	Zero for no overstrike. 0xff to turn it on.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	ui.def	

■ UserStandardSound

Play a standard sound.

Pass:	di	StandardSoundType.
	cx	If playing a stream, this is the stream handle. If playing a buffer, this is the segment of the buffer.
	dx	If playing a note, this is the frequency. If playing a buffer, this is the offset of the buffer.
	ds	If playing a note, this is its duration. DGroup.
Returns:	Nothing.	
Destroyed:	di .	
Library:	ui.def	

Assembly Reference

■ UserUnregisterForTextContext

Unregisters the passed object to receive context data.

Pass: `^lcx:dx` Object to unregister.
Returns: Nothing.
Destroyed: `Nothing.`
Library: **ui.def**

■ UtilAsciiToHex32

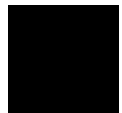
Converts a null-terminated ASCII string into a dword. The string may be signed or unsigned.

Pass: `ds:si` String to convert.
Returns: `CF` Set if error, clear if no error.
 `dx:ax` DWord value if no error. The `ax` register will hold a **UtilAsciiToHexError** value otherwise.
Destroyed: `Nothing.`
Library: **ui.def**

■ UtilHex32ToAscii

Converts 32-bit unsigned number to its ASCII representation. The number may be signed or unsigned.

Pass: `dx:ax` String to convert.
 `cx` **UtilHexToAsciiFlags**, allowing the placement of leading zeros and/or a null terminator.
 `es:di` Buffer in which to place string. Should be of minimum size `UHTA_NO_NULL_TERM_BUFFER_SIZE` or `UHTA_NULL_TERM_BUFFER_SIZE`.
Returns: `CF` Set if error, clear if no error.
 `dx:ax` DWord value if no error. The `ax` register will hold a **UtilAsciiToHexError** value otherwise.
 `cx` Length of the string, not including *null*.
Destroyed: `Nothing.`
Library: **ui.def**



■ VisCompInitialize

This routine initializes a VisComp object. This does parent class initialization first, followed by an initialization of the VisComp part. It initializes the composite linkage and marks the visible object as a composite.

Pass: ***ds:si** Instance data.
 es Segment of VisCompClass.
 ax, bx Ignored (ergo, may safely be called using **CallMod**).
 Returns: Nothing.
 Destroyed: **ax, bx, cx, dx, bp, si, di, ds, es**.
 Library: **vCompC.def**

■ VisCompMakePressesInk

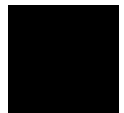
This routine is a handler for subclasses of VisCompClass that wish to make presses that are not on a child to be ink.

Pass: ***ds:si** Instance data.
 es Segment of VisCompClass.
 ax MSG_META_QUERY_IF_PRESS_IS_INK.
 cx, dx Press position.
 Returns: **bp** As returned by child object, if press was over a child object.
 Zero if press not over child object.
 ax **InkReturnValue**.
 Destroyed: Nothing.
 Library: **vCompC.def**

■ VisCompMakePressesNotInk

This routine is a handler for subclasses of VisCompClass that wish to make presses that are not on a child to be normal (i.e., not ink).

Pass: ***ds:si** Instance data.
 es Segment of VisCompClass.
 ax MSG_META_QUERY_IF_PRESS_IS_INK.
 cx, dx Press position.
 Returns: **bp** As returned by child object, if press was over a child object.
 Zero if press not over child object.
 ax **InkReturnValue**.
 Destroyed: Nothing.



VisInitialize

■ 374

Library: **vCompC.def**

■ VisInitialize

This routine initializes the VisInstance part of a visual object's instance data. This includes setting the size of the object to zero (bounds being (0, 0) to (-1, -1)) and marking the object invalid in all ways (image, window, and geometry).

Pass: ***ds:si** Instance data.
 es Segment of VisClass.
 ax, bx Ignored (ergo, this routine may be called using **CallMod**).
Returns: **si** Intact.
Destroyed: **ax, cx, dx, bp**.
Library: **visC.def**

■ VisObjectHandlesInkReply

This is a message handler to be used by those objects that want ink.

Pass: **ss:bp** **VisCallChildrenInBoundsFrame** structure.
Pass on stack: **VisCallChildrenInBoundsFrame** structure.
Returns: Nothing.
Destroyed: **ax, cx, di**.
Library: **visC.def**

■ VisTextGraphicCompressGraphic

This routine compresses the bitmaps in a VisTextGraphic.

Pass: All arguments passed on the stack.
Pass on stack: **VisTextGraphicCompressParams** structure.
Returns: **dx:ax** **VMChain** of GString in destination file.
Destroyed: Nothing
Library: **vTextC.def**

■ VMAlloc

Creates and allocates space for a VM block within a previously existing VM file. The block will not be initialized.

Before you use this block, make sure to lock it down with **VMLock**.

Pass: **bx** VM file handle.

Assembly Reference

	ax	User ID. This can be any word-length data that the application wishes to associate with the VM block. (This ID can be used with VMFind .)
	cx	Number of bytes to allocate for the block. This may be 0, in which case no associated memory will be assigned to the block.
Returns:	ax	VM block handle, marked dirty if memory is allocated within the block.
Destroyed:	Nothing.	
Library:	vm.def	

■ VMAllocLMem

Allocates a VM block and initializes it to contain a local memory heap. If you want a fixed data header space, you must pass the total size to allocate (including the **LMemBlockHeader**; otherwise, pass zero indicating that only enough space for an **LMemBlockHeader** will be allocated in the local memory block header.

You do not need to specify a block size, since the heap will automatically expand itself.

Pass:	ax	Type of LMem heap to create (LMemType).
	bx	VM file handle.
	cx	Size of block handle (or 0 for default).
Returns:	ax	VM block handle.
Destroyed:	Destroyed.	
Library:	vm.def	

■ VMAttach

Attaches an existing block of memory to a VM block, deleting whatever data was stored there before.

Pass:	bx	VM file handle.
	ax	VM block handle (or 0 to allocate a new VM block). Any data previously associated with that block will be lost.
	cx	Handle of global memory block to attach.
Returns:	ax	VM block.
Destroyed:	Nothing.	
Library:	vm.def	



VMCheckForModifications

■ 376

■ VMCheckForModifications

Checks whether a VM file has been marked as modified.

Pass: **bx** VM file handle.
Returns: **CF** Set if file is modified.
Destroyed: Nothing.
Library: **vm.def**

■ VMClose

Closes a VM file. This routine updates all dirty blocks, frees all global memory blocks attached to the file and closes the file.

Make sure to update the file before closing it. If **VMClose** encounters an error when closing the file, it will still close the file and free its memory anyway.

Pass: **al** **FILE_NO_ERRORS** or 0.
bx VM file handle.
Returns: **CF** Set on error.
ax **VMStatus** (which may possibly be an error code).
Destroyed: **bx** (if file was actually closed).
Library: **vm.def**

■ VMCompareVMChains

Compares two VM chains or DB items.

Pass: **bx** VM file handle #1.
ax:bp VM chain #1.
dx VM file handle #2.
cx:di VM chain #2.
Returns: **CF** Set if the VM chains are equal.
Destroyed: Nothing.
Library: **vm.def**

■ VMCopyVMBlock

Creates a duplicate of a VM block into the specified destination file. This destination file may be the same as the source file. The duplicate VM block user ID will remain the same as the original block's user.

Pass: **bx** VM file handle.

Assembly Reference

	ax	VM block handle.
	dx	Destination file handle.
Returns:	ax	New block handle.
Destroyed:	Nothing.	
Library:	vm.def	

■ VMCopyVMChain

Creates a duplicate of a VM chain (or DB item) in the specified destination file. This destination file may be the same as the source file. All blocks in the duplicate will have the same user ID number.

Pass:	bx	Source file.
	ax:bp	Source VM chain. (If this is a DB item, bp will contain the item number, otherwise it will be zero.)
	dx	Destination file.
Returns:	ax:bp	Destination VM chain (or DB item) created.
Destroyed:	Nothing.	
Library:	vm.def	

■ VMDetach

Detaches a VM block from a VM file, returning the handle of the detached global memory block (which may have been allocated). The VM block will contain no data after the detach.

Pass:	bx	VM file handle.
	ax	VM block handle.
	cx	Owner of memory handle (0 for the current thread's geode).
Returns:	di	Handle of global memory block.
Destroyed:	ax	
Library:	vm.def	

■ VMDirty

Marks a locked VM block as dirty.

Pass:	bp	Locked VM memory handle containing the VM block.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	vm.def	



VMFind

■ 378

■ VMFind

Given a VM block's user ID, locates and returns the first VM block handle whose user ID matches.

Pass:	bx	VM file handle.
	ax	User ID.
	cx	0 to find the first block with the given ID; otherwise, a VM block handle to find the *next* block with the given ID.
Returns:	CF	Clear if found, set otherwise.
	ax	VM block handle if found, else ax = 0.
Destroyed:	Nothing.	
Library:	vm.def	

■ VMFree

Frees a VM block handle. If a global memory block is attached to the VM block, that is freed also.

Pass:	bx	VM file handle.
	ax	VM block handle.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	vm.def	

■ VMFreeVMChain

Frees a VM chain (or DB item). If freeing a VM chain, all blocks in the chain will be freed.

Pass:	bx	VM file handle.
	ax:bp	VM chain.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	vm.def	

■ VMGetAttributes

Returns the **VMAttributes** associated with the specified VM file.

Pass:	bx	VM file handle.
Returns:	al	VMAttributes .
Destroyed:	Nothing.	

Assembly Reference

Library: **vm.def**

■ VMGetDirtyState

Returns the dirty state of a VM file, in a word-length value that specifies both if the file has been dirtied since the last save and whether it has been dirtied since the last save, auto-save, or update.

Pass: **bx** VM file handle.

Returns: **al** Non-zero if the file has been marked dirty since the last save.
ah Non-zero if the file has been marked dirty since the last save, auto-save, or update.

Destroyed: Nothing.

Library: **vm.def**

■ VMGetMapBlock

Returns the VM block handle of the VM file's map block.

Pass: **bx** VM file handle.

Returns: **ax** VM block handle of map block (or 0 if there is none).

Destroyed: Nothing.

Library: **vm.def**

■ VMGrabExclusive

Provides the current thread with exclusive access to the VM file.

Pass: **bx** VM file handle.
ax **VMOperation** for the operation to be performed.
cx Timeout value in increments of 1/10th of a second. Pass zero to wait for as long as it takes.

Returns: **ax** **VMStartExclusiveReturnValue**
cx existing **VMOperation** (if the routine was time-out'd).

Destroyed: Nothing.

Library: **vm.def**

■ VMInfo

Fetches information about a VM block. Returns the memory handle, block size, and user ID of the specified VM block.

Pass: **bx** VM file handle.
ax VM block handle.



VMLock

■ 380

Returns:	CF	Clear if block handle is valid. Set if block handle is free, out of range, or otherwise illegal. No other registers will be altered if this is the case.
	CX	Size of block. (This size is not a guarantee that the block will remain the same size after this routine returns. It must be locked with VMLock to ensure this.)
	AX	Associated memory handle, if any (or 0 if none).
	DI	User ID of the block.
Destroyed:	Nothing.	
Library:	vm.def	

■ VMLock

Locks the given VM block into the global memory heap.

Pass:	bx	VM file handle.
	ax	VM block handle.
Returns:	ax	Segment of locked VM block.
	bp	Memory handle of locked VM block.
Destroyed:	Nothing.	
Library:	vm.def	

■ VMMemBlockToVMBlock

Returns the VM block and VM file associated with a given VM memory handle.

Pass:	bx	VM memory handle.
Returns:	ax	VM block handle.
	bx	VM file handle.
Destroyed:	Nothing.	
Library:	vm.def	

■ VMModifyUserID

Changes the user ID of the passed VM block.

Pass:	bx	VM file handle.
	ax	VM block handle.
	CX	New user ID.
Returns:	Nothing.	
Destroyed:	Nothing.	

Assembly Reference

Library: **vm.def**■ **VMOpen**

Opens (or creates) a VM file, returning the handle of the opened file.

VMOpen looks for the file in the thread's current working directory (unless creating a temporary file).

Pass:	ah	VMOpenType.
	al	VMAccessFlags.
	cx	Compression threshold percentage passed as an integer. (Pass zero to use the system default.)
	ds:dx	Pointer to file name to open (null-terminated text string).
Returns:	CF	Set on error.
	ax	VMStatus.
	bx	VM file handle.
Destroyed:	Nothing.	
Library:	vm.def	

■ **VMPreserveBlocksHandle**

Keeps the same global memory block with this VM block until the block is explicitly detached or the VM block is freed.

Pass:	bx	VM file handle.
	ax	VM block handle.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	vm.def	

■ **VMReleaseExclusive**

Relinquishes a thread's exclusive access to a VM file.

Pass:	bx	VM file handle (or override).
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	vm.def	

■ **VMRevert**

Reverts a file to its last-saved state.

Pass:	bx	VM file handle.
-------	-----------	-----------------

VMSave

■ 382

Returns: Nothing.
Destroyed: Nothing.
Library: **vm.def**

■ VMSave

Updates and saves a VM file, freeing all backup blocks.

Pass: **bx** VM file handle.
Returns: **CF** Set on error.
ax (If **CF** is set) error code.
Destroyed: Nothing.
Library: **vm.def**

■ VMSaveAs

Saves a VM file under a new name. The old file is reverted to its last-saved state.

Pass: **ah** **VMOpenType**.
al **VMAccessFlags**.
bx VM file handle.
cx Compression threshold percentage passed as an integer.
(Pass zero to use the system default.)
ds:dx Pointer to new file name (null-terminated string).
Returns: **CF** Set on error.
bx Handle for new file.
ax **VMStatus**.
Destroyed: **cx, dx**
Library: **vm.def**

■ VMSetAttributes

Changes a VM file's **VMAttributes** settings, also returning the new attributes in a word-length record.

Pass: **bx** VM file handle.
al Bits in **VMAttributes** record to set.
ah Bits in **VMAttributes** to clear.
Returns: **al** New **VMAttributes**.
Destroyed: Nothing.
Library: **vm.def**

Assembly Reference

■ VMSetExecThread

Sets the thread that will execute methods of all objects within the passed VM file.

Pass: **bx** VM file handle.
 ax Thread handle.

Returns: Nothing.

Destroyed: Nothing.

Library: **vm.def**

■ VMSetMapBlock

Sets the map block of a VM file.

Pass: **bx** VM file handle.
 ax VM block handle of map block.

Returns: Nothing.

Destroyed: Nothing.

Library: **vm.def**

■ VMSetReloc

Sets the (fixed-memory) data relocation routine to be called whenever a block is brought into memory from a VM file (or written to memory).

Pass: **bx** VM file handle.
 cx:dx Address of routine to call.

Returns: Nothing.

Destroyed: Nothing.

Callback Routine Specifications:

Passed:	ax	Memory handle.
	bx	VM file handle.
	di	Block handle of loaded block.
	dx	Segment address of block.
	cx	VMRelocType .
	bp	User ID of block.

Return: Block relocated/unrelocated.

May Destroy: **ax, bx, cx, dx, si, di, bp, ds, es**

Library: **vm.def**

VMUnlock

■ 384

■ VMUnlock

Unlocks a locked VM block. Note that the block's global memory handle is passed (not it's VM handle).

Pass: **bp** Memory handle of locked VM block.

Returns: Nothing.

Destroyed: Nothing except, possibly **ds** and **es** if error-checking. (If segment error-checking is on, and either **ds** or **es** is pointing to a block that has become unlocked, then that register will be set to NULL_SEGMENT upon return from this procedure.)

Library: **vm.def**

■ VMUpdate

Updates all dirty blocks within a VM file to the disk. (This is known as flushing all changes onto disk.)

Pass: **bx** VM file handle.

Returns: **CF** Clear if successful, set otherwise.
ax (If **CF** is set): error code.

Destroyed: Nothing.

Library: **vm.def**

■ VMVMBlockToMemBlock

Returns the handle of the global memory block attached to a specified VM block. If no global block is currently attached, it will allocate and attach one.

Pass: **bx** VM file handle (or override).

ax VM block handle.

Returns: **ax** Global memory block handle.

Destroyed: Nothing.

Library: **vm.def**

■ VTFClearSmartQuotes

Clear the variable that prohibits smart quotes.

Pass: ***ds:si** Instance data of a VisText object (or subclass).

Returns:

Destroyed:

Assembly Reference

Library: **vTextC.def**

■ WarningNotice

A place for Swat to place a breakpoint to catch taken invocations of the WARNING family of macros

Pass: Nothing.
 Returns: Nothing.
 Destroyed: **Nothing.**
 Library: **ec.def**

■ WinAckUpdate

Acknowledges an update in the window without performing any visual updating. This is equivalent to calling **GrBeginUpdate**, then **GrEndUpdate** but does not require a GState to do so.

This routine should be used when responding to a MSG_META_EXPOSED when the application does not wish to perform any update drawing.

Pass: **di** **WindowHandle.**
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **win.def**

■ WinApplyRotation

Applies the passed rotation to the window's transformation matrix.

Pass: **di** **WindowHandle** or GState handle.
dx.ax Angle to rotate (as a **WWFixed** value).
si **WinInvalFlag.**
 (WIF_INVALIDATE to invalidate the window.
 WIF_DONT_INVALIDATE to avoid invalidating the window.)
 Returns: **CF** Set if **di** is a gstate or a window that is closing.
 Destroyed: Nothing.
 Library: **win.def**

■ WinApplyScale

Applies the passed scale factor to the window's transformation matrix.

Pass: **di** **WindowHandle.**
dx.cx X-scale factor (**WWFixed** value).



WinApplyTransform

■ 386

	bx.ax	Y-scale factor (WWFixed value).
	si	WinInvalFlag (WIF_INVALIDATE to invalidate the window. WIF_DONT_INVALIDATE to avoid invalidating the window.)
Returns:	CF	Set if di is a gstate or a window that is closing.
Destroyed:	Nothing.	
Library:	win.def	

■ WinApplyTransform

Concatenates the passed transformation matrix (**TMatrix**) with the window's current transformation matrix, forming the window's new transformation matrix.

Pass:	di	WindowHandle.
	ds:si	Pointer to new TMatrix to use.
	cx	WinInvalFlag (WIF_INVALIDATE to invalidate the window. WIF_DONT_INVALIDATE to avoid invalidating the window.)
Returns:	Nothing.	
Destroyed:	ax, bx	
Library:	win.def	

■ WinApplyTranslation

Applies the passed translation to the window's transformation matrix.

Pass:	di	WindowHandle.
	dx.cx	X translation to apply (WWFixed value).
	bx.ax	Y translation to apply (WWFixed value).
	si	WinInvalFlag (WIF_INVALIDATE to invalidate the window. WIF_DONT_INVALIDATE to avoid invalidating the window.)
Returns:	CF	Set if di is a gstate or a window that is closing.
Destroyed:	Nothing.	
Library:	win.def	

■ WinApplyTranslationDWord

Applies a 32-bit translation to a window's transformation matrix.

Pass:	di	WindowHandle.
	dx.cx	X translation to apply (32-bit integer).

Assembly Reference

	bx.ax	Y translation to apply (32-bit integer).
	si	WinInvalFlag (WIF_INVALIDATE to invalidate the window. WIF_DONT_INVALIDATE to avoid invalidating the window.)
Returns:	CF	Set if di is a gstate or a window that is closing.
Destroyed:	Nothing.	
Library:	win.def	

■ WinChangeAck

Called to acknowledge a MSG_META_WIN_CHANGE, this function generates “Enter” and “Leave” events for any windows which the mouse may have moved across.

Pass:	cx, dx	Screen location to traverse to.
	bp	Window handle of tree to traverse to.
Returns:	cx:dx	Enter/leave Output Descriptor for that window (zero if none).
	bp	Handle of window that mouse pointer is in.
Destroyed:	Nothing.	
Library:	win.def	
Warning:	This routine may resize LMem and/or object blocks, moving them on the heap and invalidating stored segment pointers and current register or stored offsets to them.	

■ WinChangePriority

Changes a window’s priority.

Pass:	ax	WinPassFlags.
	dx	Layer ID.
	di	WindowHandle.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	win.def	

■ WinClose

Closes and frees the specified window.

Pass:	di	WindowHandle.
Returns:	Nothing.	
Destroyed:	di	



WinDecRefCount

■ 388

Library: **win.def**

■ WinDecRefCount

Handle acknowledge of window death. To be called by whoever receives a MSG_META_WIN_DEC_REF_COUNT.

Pass: **di** Window handle.

Returns: Nothing.

Destroyed: **Nothing.**

Library: **win.def**

■ WinEnsureChangeNotification

Ensures that if the window that has the implied mouse grab has changed since the last MSG_META_WIN_CHANGE, then another MSG_META_WIN_CHANGE will be sent out to update the system.

Pass: Nothing.

Returns: Nothing.

Destroyed: Nothing.

Library: **win.def**

■ WinGeodeGetFlags

Returns the **GeodeWinFlags** associated with a geode.

Pass: **bx** Geode handle.

Returns: **ax** **GeodeWinFlags**.

Destroyed: Nothing.

Library: **win.def**

■ WinGeodeGetInputObj

Returns the optr of the input object associated with the specified geode. (If there is no such object, this routine returns a null optr.)

Pass: **bx** Geode handle.

Returns: **cx:dx** Optr of Input object (or 0 if none).

Destroyed: Nothing.

Library: **win.def**

Assembly Reference

■ WinGeodeGetParentObj

Returns the optr of the parent object associated with the passed geode. (If there is no such parent object, this routine returns a null optr.)

Pass: **bx** Geode handle.
 Returns: **cx:dx** Parent object (or 0 if none).
 Destroyed: Nothing.
 Library: **win.def**

■ WinGeodeSetActiveWin

Sets the passed window within the specified geode to be that geode's "active" window.

Pass: **bx** Geode handle.
di **WindowHandle**.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **win.def**

■ WinGeodeSetFlags

Sets the **GeodeWinFlags** associated with the specified geode.

Pass: **bx** Geode handle.
ax **GeodeWinFlags**.
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **win.def**

■ WinGeodeSetInputObj

Sets the passed geode's input object to the specified optr.

Pass: **bx** Geode handle.
cx:dx Input object (or 0 to set it to none).
 Returns: Nothing.
 Destroyed: Nothing.
 Library: **win.def**



Sets the passed geode's parent object to the specified optr.

Library: **win.def**Library: **win.def**

cx:dx	Address of strategy routine.
--------------	------------------------------

WIT_FLAGS		
	al	WinRegFlags.
	ah	WinPtrFlags.
WIT_LAYER_ID		
	ax	Layer ID.
WIT_PARENT_WIN		
WIT_FIRST_CHILD_WIN		
WIT_LAST_CHILD_WIN		
WIT_PREV_SIBLING_WIN		
WIT_NEXT_SIBLING_WIN		
	ax	Appropriate window link.
WIT_PRIORITY		
	al	WinPriorityData.

Destroyed: Nothing.

Library: **win.def**

■ WinGetTransform

Returns the transformation matrix of the specified window.

Pass: **di** **WindowHandle.**
ds:si Pointer to buffer to hold **TMatrix**. (There should be room for 6 **WWFixed** arguments.)

Returns: Buffer at **ds:si** filled in the following order:

Original Matrix:

[e11 e12 0]
[e21 e22 0]
[e31 e32 1]

Order of returned array:

[e11 e12 e21 e22 e31 e32]

Destroyed: Nothing.

Library: **win.def**

■ WinGetWinScreenBounds

Returns the bounds of the on-screen portion of the specified window.

Pass: **di** **WindowHandle.**

Returns: **CF** Set if **di** is a gstate or is a window that is closing.
ax Left position of window.
bx Top position of window.



WinGrabChange

■ 392

	cx	Right position of window.
	dx	Bottom position of window.
Destroyed:	Nothing.	
Library:	win.def	

■ WinGrabChange

Allows an object to grab pointer events for windows within the system.

Pass:	bx:si	Object to send change events to.
Returns:	INTs ON CF	Set if grabbed, clear if grab is unsuccessful.
Destroyed:	Nothing.	
Library:	win.def	

■ WinInvalReg

Invalidates a portion of the specified window, indicated by the passed region (or rectangle).

Pass:	ax, bx, cx, dx	Parameters for region. (Bounds if a rectangular region.) These coordinates must be WINDOW coordinates.
	bp:si	Region. (Zero if rectangular.)
	di	WindowHandle.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	win.def	

■ WinLocatePoint

Searches through a window tree, returning the window in which the passed point lies within, along with other information about the window.

Pass:	di	WindowHandle of window to start search at.
	cx	X screen position.
	dx	Y screen position.
Returns:	di	WindowHandle of window containing the passed point (or NULL_WINDOW).
	bx:si	Optr associated with window.
	cx	Horizontal (X) absolute position of window.
	dx	Vertical (Y) absolute position of window.

Assembly Reference

Destroyed: **ax, bx, cx, dx**

Library: **win.def**

■ WinMove

Moves a window, either relative to its current position or in an absolute manner (relative to the parent).

Pass: **ax** Horizontal units to move window.
bx Vertical units to move window.
si **WinPassFlags**. (WPF_ABS is set if this move is an absolute position, clear if this is a relative move.)
di **WindowHandle**.

Returns: Nothing.

Destroyed: Nothing.

Library: **win.def**

■ WinOpen

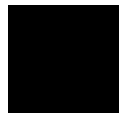
Allocates and initializes a window and (optionally) its associated GState.

The region/rectangle passed is expressed in units relative to the parent's window. (I.e. the width of a rectangle should be *right - left + 1*.)

Pass: **al** Color index (or red value if using RGB).
ah **WinColorFlags**.
b1 Green value (if using RGB values).
bh Blue value (if using RGB values).
cx:dx Optr of input object (object responsible for handling mouse input for this window). This object must be run by the same thread as the owning geode's input object.
di:bp Optr of exposure object.
si **WinPassFlags**.

Pass on stack:

word Layer ID.
word Geode which should own this window. Pass zero for the current running geode.
word **WindowHandle** of parent (or handle of the video driver if there is no parent).
word High word of region (0 for rectangular window).
word Low word of region (0 for rectangular window).
word PARAM_3 for region (bottom if rectangular).
word PARAM_2 for region (right if rectangular).
word PARAM_1 for region (top if rectangular).



WinRealizePalette

■ 394

	word	PARAM_0 for region (left if rectangular).
Returns:	bx	Handle to allocated and opened window.
	di	Handle to allocated and opened GState (if any).
Destroyed:	ax, cx, dx, si, bp	
Library:	win.def	

■ WinRealizePalette

Realize the palette for this window in hardware.

Pass:	di	WindowHandle of window.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	win.def	

■ WinReleaseChange

Releases an object from being notified of window changes.

Pass:	bx:si	Object to release change notification.
Returns:	INTs ON.	
Destroyed:	Nothing.	
Library:	win.def	

■ WinResize

Resizes a window. It is also possible to move it at the same time.

Pass:	ax, bx, cx, dx	Parameters of the region (or bounds if the region is instead a rectangle).
	bp:si	Region (or 0 for a rectangle).
	di	WindowHandle.
Pass on stack:	WinPassFlags (with mask WPF_ABS to perform an absolute resize/move).	
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	win.def	

Assembly Reference

■ WinScroll

Scrolls a document within a window by the passed values. Actual displacement values may be different than the passed values due to rounding and optimizations.

This routine will not work for rotated windows.

Pass:	di	WindowHandle.
	dx.cx	Horizontal displacement (WWFixed value).
	bx.ax	Vertical displacement (WWFixed value).
Returns:	dx.cx	Actual horizontal displacement applied.
	bx.ax	Actual vertical displacement applied.
Destroyed:	Nothing.	
Library:	win.def	

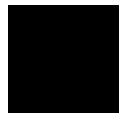
■ WinSetInfo

Sets information within a window. Passed register arguments depend on the **WinInfoType** passed in **si**.

Pass:	di	WindowHandle.
	si	WinInfoType.
	WIT_PRIVATE_DATA:	
	ax, bx, cx, dx	Private data.
	WIT_COLOR	Color index (or red value for RGB).
		WCF_TRANSPARENT if none, WIN_RGB if RGB colors.
		Low bits store the color map mode.
	WIT_INPUT_OBJ	
	cx:dx	New optr of input object.
	WIT_EXPOSURE_OBJ	
	cx:dx	New optr of exposure object.
	WIT_STRATEGY	
	cx:dx	Address of strategy routine.
Returns:	CF	Set if di is a GState or references a window that is closing.
Destroyed:	Nothing.	
Library:	win.def	

■ WinSetNullTransform

Replaces a window's transformation matrix with a null (identity) transformation.



WinSetPtrImage

■ 396

Pass:	di cx	WindowHandle. WinInvalFlag (WIF_INVALIDATE to invalidate the window. WIF_DONT_INVALIDATE to avoid invalidating the window.)
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	win.def	

■ WinSetPtrImage

Sets the pointer image within the range of the passed window.

Pass:	bp cx:dx di	PIL_GADGET or PIL_WINDOW. Optr to PointerDef in sharable memory block. (If cx = 0, dx = PtrImageValue .) WindowHandle.
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	win.def	

■ WinSetTransform

Sets the transformation matrix of a window. Any previous transformation matrix is lost.

Pass:	di ds:si cx	WindowHandle. Pointer to new TMatrix . WinInvalFlag (WIF_INVALIDATE to invalidate the window. WIF_DONT_INVALIDATE to avoid invalidating the window.)
Returns:	Nothing.	
Destroyed:	Nothing.	
Library:	win.def	

■ WinSuspendUpdate

Suspends the sending of update messages to the window. If an update is already in progress, it will be allowed to continue and suspend behavior will commence afterward.

Pass:	di	WindowHandle.
Returns:	Nothing.	

Assembly Reference

Destroyed: Nothing.

Library: **win.def**

■ WinSysSetActiveGeode

Sets the passed geode to be the system's "active" geode.

Pass: **bx** Geode handle to make active (0 sets none active).

Returns: Nothing.

Destroyed: Nothing.

Library: **win.def**

■ WinTransform

Translates the passed document coordinates into screen coordinates, ignoring the effect of any transformation in the associated GState.

Pass: **ax** X coordinate (in document coordinates).

bx Y coordinate (in document coordinates).

di **WindowHandle**.

Returns: **CF** Set on error.

ax On success, x screen coordinate; on error, an error code.

bx On success, y screen coordinate; destroyed on error.

Destroyed: Nothing (except possibly **bx**, see above).

Library: **win.def**

■ WinTransformDWord

Translates the passed 32-bit document coordinates into 32-bit screen coordinates.

Pass: **dx.cx** x document coordinate (32-bit integer).

bx.ax y document coordinate (32-bit integer).

di **WindowHandle**.

Returns: **dx.cx** x screen coordinate (32-bit integer).

bx.ax y screen coordinate (32-bit integer).

CF Set if **di** is a gstate or a window that is closing.

Destroyed: Nothing.

Library: **win.def**



WinUnSuspendUpdate

■ 398

■ WinUnSuspendUpdate

Cancels any previous **WinSuspendUpdate** call, allowing update drawing to the window.

Pass: **di** **WindowHandle.**

Returns: Nothing.

Destroyed: Nothing.

Library: **win.def**

■ WinUntransform

Translates a coordinate pair from screen coordinates into document coordinates, ignoring the effects in the associated GState.

Pass: **ax** X screen coordinate.
bx Y screen coordinate.
di **WindowHandle.**

Returns: **CF** Set on error.
ax On success, x screen coordinate; on error, an error code.
bx On success, y screen coordinate; destroyed on error.

Destroyed: Nothing (except possibly **bx**; see above).

Library: **win.def**

■ WinUntransformDWord

Translates a 32-bit coordinate pair into 32-bit screen coordinates.

Pass: **dx.cx** X screen coordinate (32-bit integer).
bx.ax Y screen coordinate (32-bit integer).
di **WindowHandle.**

Returns: **CF** Set if **di** was a GState or a window that is closing. (**ax**, **bx**, **cx**, and **dx** remain unchanged).

dx.cx On success, x document coordinate (32-bit integer), otherwise unchanged.

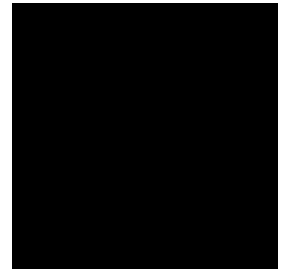
bx.ax On success, y document coordinate (32-bit integer), otherwise unchanged.

Destroyed: Nothing.

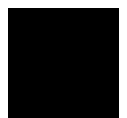
Library: **win.def**

Assembly Reference

Structures



3



■ ActionDescriptor

```
ActionDescriptor    struct
    AD_OD            optr
    AD_message       word
ActionDescriptor    ends
```

This structure describes an event, storing both the message to send and the optr of the destination for that message.

Library: **ui.def**

■ ActivateCreateFlags

```
ActivateCreateFlags    record
    ACF_NOTIFY         :1                ;notify selected objects that
                                         ;tool is activating
ActivateCreateFlags    end
```

Library: **grobj.def**

■ ActivationData

```
ActivationData struc
    AD_dialog          optr              ; On-screen "Activating..." dialog.
    AD_appLaunchBlock  hptr              ; Initial AppLaunchBlock - not used once
                                         ; geode is known (NOT a reference, i.e.
                                         ; MemRefCount is *not* incremented for
                                         ; this usage, so don't decrement later)
    AD_geode           hptr              ; Geode having a dialog put up for it
    AD_savedBlankMonikerlptr ; Saved blank moniker if not currently in use in
                                         ; the "Activating" dialog

ActivationData         ends
```

Library: **grobj.def**

■ ActiveSearchSpellType

```
ActiveSearchSpellType  etype    byte, 0, 1
    ASST_NOTHING_ACTIVE    enum    ActiveSearchSpellType
    ASST_SPELL_ACTIVE      enum    ActiveSearchSpellType
    ASST_SEARCH_ACTIVE      enum    ActiveSearchSpellType
```

Library: **Objects/vTextC.def**



■ AddChildRelativeParams

```
AddChildRelativeParams    struct
    ACRP_child      optr      ;the object to add
    ACRP_parent     optr      ;the visual parent to use
    ACRP_buildFlags SpecBuildFlags ;the spec build flags to use
AddChildRelativeParams    ends
```

Library: **Objects/visC.def**

■ AddUndoActionFlags

```
AddUndoActionFlags        record
    AUAF_NOTIFY_BEFORE_FREEING          :1
    AUAF_NOTIFY_IF_FREED_WITHOUT_BEING_PLAYED_BACK :1
    :14 ; Unused bits
AddUndoActionFlags        end
```

AUAF_NOTIFY_BEFORE_FREEING

Set this flag if you want to be notified when (before) the associated action is freed.

AUAF_NOTIFY_IF_FREED_WITHOUT_BEING_PLAYED_BACK

Set this flag if you want to be notified if the action is freed without being played back.

Library: **Objects/gProcC.def**

■ AddUndoActionStruct

```
AddUndoActionStruct        struct
    AUAS_data              UndoActionStruct
    AUAS_output            optr
    AUAS_flags             AddUndoActionFlags
    even
AddUndoActionStruct        ends
```

This structure provides several pieces of information vital to processes which will be working with the “undo” mechanism.

AUAS_output stores the optr of the object to be sent MSG_META_CLIPBOARD_UNDO.

Library: **Objects/gProcC.def**

■ AddVarDataParams

```
AddVarDataParams    struct
    AVDP_data        fptr;
    AVDP_dataSize    word;
    AVDP_dataType    word;
AddVarDataParams    ends
```

AVDP_data stores a pointer to data to initialize the vardata with, or null if no extra data is available. *AVDP_data* may also be null if the extra data should be initialized to zero.

AVDP_dataSize stores the size of the extra data, if any.

AVDP_dataType stores the VarData type.

Library: **Objects/metaC.def**

■ AdjustType

```
AdjustType            etype byte, 0, 1
    AT_NORMAL          enum      AdjustType
    AT_PASTE           enum      AdjustType
    AT_QUICK           enum      AdjustType
```

Library: **Objects/vTextC.def**

■ AfterAddedToGroupData

```
AfterAddedToGroupData struct
    AATGD_group        optr;
    AATGD_centerAdjust PointDWFixed;
AfterAddedToGroupData ends
```

AATGD_group stores the optr of the group object.

AATGD_centerAdjust stores the value to subtract from the center of the child to position it correctly.

Library: **grobj.def**

■ AfterEditAction

```
AfterEditAction      etype byte, 0
    DONT_SELECT_AFTER_EDIT  enum  AfterEditAction
    SELECT_AFTER_EDIT       enum  AfterEditAction
```

Library: **grobj.def**



■ **AlignParams**

```
AlignParams          struct
    AP_x              DWFixed
    AP_y              DWFixed
    AP_spacingX       DWFixed
    AP_spacingY       DWFixed
    AP_type            AlignType
AlignParams          ends
```

Library: **grobj.def**

■ **AlignToGridType**

```
AlignToGridType      record
    ATGT_LEFT         :1
    ATGT_H_CENTER     :1
    ATGT_RIGHT        :1
    ATGT_TOP          :1
    ATGT_V_CENTER     :1
    ATGT_BOTTOM       :1
AlignToGridType      end
```

Library: **grobj.def**

■ **AlignType**

```
AlignType            record
    AT_ALIGN_X        :1
    AT_DISTRIBUTE_X   :1
    AT_CLRW           CenterLeftRightWidth:2
    AT_ALIGN_Y        :1
    AT_DISTRIBUTE_Y   :1
    AT_CTBH           CenterTopBottomHeight:2
AlignType            end
```

Library: **grobj.def**

■ AnotherToolActivatedFlags

```
AnotherToolActivatedFlags      record
    ATAF_STANDARD_POINTER      :1
    ATAF_SHAPE                  :1
    ATAF_GUARDIAN               :1
AnotherToolActivatedFlags      end
```

This record provides basic information about tool activation. Selected or edited GrObj objects will use this information to determine whether to remain selected or edited.

ATAF_STANDARD_POINTER

A pointer tool intended to work on the normal move and resize handles of an object.

ATAF_SHAPE

A shape drawing tool, rectangle ellipse...

ATAF_GUARDIAN

A Vis guardian object

Library: **grobj.def**

■ AppAttachFlags

```
AppAttachFlags      record
    AAF_RESTORING_FROM_STATE :1
    AAF_STATE_FILE_PASSED   :1
    AAF_DATA_FILE_PASSED    :1
    AAF_RESTORING_FROM_QUIT :1
                                :12
AppAttachFlags      end
```

These flags are passed in MSG_GEN_PROCESS_RESTORE_FROM_STATE, MSG_GEN_PROCESS_OPEN_APPLICATION, and MSG_GEN_PROCESS_OPEN_ENGINE.

AAF_RESTORING_FROM_STATE

Set if this application was invoked with MSG_GEN_PROCESS_RESTORE_FROM_STATE. The mode chosen to restore to was extracted from the GenApplication object. The flag AAF_STATE_FILE_PASSED will always be set if this flag is.

AAF_STATE_FILE_PASSED

Set if a state file was passed into this application when invoked. This will be set if AAF_RESTORING_FROM_STATE, but may also be set if the application has been invoked with a "template" state file.



AAF_DATA_FILE_PASSED

Set if a data file, whose name is in the **AppLaunchBlock**, has been passed into the invocation of this application.

AAF_RESTORING_FROM_QUIT

Set if the application was in the process of quitting, got to engine mode, and is now being restarted to application mode again. If this is set, **AAF_RESTORING_FROM_STATE** will also be set.

Library: **Objects/gProcC.def**

■ AppInstanceReference

```
AppInstanceReference    struct
    AIR_filename        char  PATH_BUFFER_SIZE dup (?)
    AIR_stateFile       char  FILE_LONGNAME_BUFFER_SIZE dup (?)
    AIR_diskHandle      word
    AIR_savedDiskData   byte  0
AppInstanceReference    ends
```

This structure stores information needed to reload an instance of an application. This structure is stored in the application object itself and copied into the field when the application is forcefully detached.

AIR_filename stores the file name of the application to launch. The path name is relative to the **SP_APPLICATION** directory, though you can override this behavior by specifying an absolute path.

AIR_stateFile specifies the name of the state file for the application. The state file name is assumed to be in the **SP_STATE** directory. If the first byte of this instance data is "0", then there is no state file for this application and it cannot be relaunched.

AIR_diskHandle specifies the disk handle where the application is located. In the field, if **AppInstanceReference** is a placeholder structure *AIR_diskHandle* stores the handle of the application object we are waiting to detach.

AIR_savedDiskData stores the start of data saved by **DiskSave** when instance data is saved to state.

Library: **Objects/gProcC.def**

■ AppLaunchBlock

```
AppLaunchBlock    struct
    ALB_appRef      AppInstanceReference
    ALB_appMode     word
    ALB_launchFlags AppLaunchFlags
    ALB_diskHandle  word
    ALB_path        char  PATH_BUFFER_SIZE dup (?)
```

Reference book

```

ALB_dataFile           FileLongName
ALB_genParent          optr
ALB_userLoadAckAD      ActionDescriptor
ALB_userLoadAckID      word
ALB_extraData          word
AppLaunchBlock        ends

```

This structure is used when an application is first starting up. It is the argument of various messages, including MSG_META_ATTACH, which will be intercepted by system classes. The first fields (*ALB_appRef*, *ALB_appMode*, and *ALB_launchFlags*) are preserved in the application's state file. The other information must be set correctly upon launch.

ALB_appRef stores the **AppInstanceReference** which specifies the pathname to both the application and its associated state file.

ALB_appMode stores the attach mode message used to invoke the application. This should be one of the following:

MSG_GEN_PROCESS_RESTORE_FROM_STATE

State file *must* be passed; no data file should be passed.

MSG_GEN_PROCESS_OPEN_APPLICATION

State file normally should not be passed, although one might pass a state file to use UI templates. A data file may be passed as well.

MSG_GEN_PROCESS_OPEN_ENGINE

State file normally should not be passed. The data file on which the engine will operate *must* be passed. If this is zero, the default data file should be used. (The default data file is specified by the application, not **GenProcessClass**.)

ALB_launchFlags stores the **AppLaunchFlags** that specify the type of launch desired for the application.

ALB_diskHandle stores the disk handle for the data path. (This is set as the application's current path in MSG_META_ATTACH.)

ALB_path stores the directory path for the application to use as its default starting path. (This is also set as the application's current path in MSG_META_ATTACH.)

ALB_dataFile stores the name of the associated data file to be opened (or zero if none). The file name is relative to *ALB_path*.

ALB_genParent stores the generic parent of the launching application (or zero to specify the default field). (This optr should be null when sent to MSG_GEN_FIELD_LAUNCH_APPLICATION.)



ALB_userLoadAckAD stores the **ActionDescriptor** to activate once the application is successfully launched (used in conjunction with *ALF_SEND_LAUNCH_REQUEST_TO_UI_TO_HANDLE*). This **ActionDescriptor** should be set to zero if no action should be sent. The event sent will pass the following information:

cx - **GeodeHandle** (if launched successfully).
dx - Error (0 if no error).
bp - ID passed in *ALB_userLoadAckID*.

This **ActionDescriptor** should be set to zero if no action should be sent.

ALB_userLoadAckID stores the ID sent out via *ALB_userLoadAckAD*, if any.

ALB_extraData stores extra data to send to the process, if any (possibly a handle to a block containing extra arguments).

Library: **Objects/gProcC.def**

■ AppLaunchFlags

```
AppLaunchFlags      record
    ALF_SEND_LAUNCH_REQUEST_TO_UI_TO_HANDLE :1
    ALF_OPEN_IN_BACK      :1
    ALF_DESK_ACCESSORY    :1
    ALF_DO_NOT_OPEN_ON_TOP :1
    ALF_OVERRIDE_MULTIPLE_INSTANCE :1
    ALF_OPEN_FOR_IACP_ONLY :1
                                :2
```

```
AppLaunchFlags      end
```

ALF_SEND_LAUNCH_REQUEST_TO_UI_TO_HANDLE

If this bit is set, the application will not immediately be launched, but instead the UI will be sent a method which will cause it to do so. Because of this, no error is returned. (This flag should *not* be passed to the application itself; it is used only by *UserLoadApplication*.)

ALF_OPEN_IN_BACK

Set to open application behind other applications. It will also ensure that if an application has multiple *GenPrimaries*, (each with a different *Layer ID*), the *GenPrimaries* will be restored in the correct order (one behind the other). This flag is normally set when restoring from state.

ALF_DESK_ACCESSORY

Set to open application as a “desk accessory”, in a layer above normal applications.

ALF_DO_NOT_OPEN_ON_TOP

Set to prevent application from opening on top with the focus.

Reference book

ALF_OVERRIDE_MULTIPLE_INSTANCE

Set to prevent application, when running in a UILM_MULTIPLE_INSTANCE field, from asking the user whether to start another copy. This flag is used especially by the express menu.

ALF_OPEN_FOR_IACP_ONLY

This flag is used only for MSG_GEN_PROCESS_OPEN_APPLICATION mode connections. This flag is set if the application is being launched via **IACPConnect()** for a specific task only, and should close once the task is complete, as indicated by the IACP connection closing (unless there is some other reason for the application to stay open, such as other application-mode IACP connections). This flag should be clear wherever **IACPConnect()** is being used to open an application with the intention that the application is to remain open after the IACP connection is closed.

Library: **Objects/gProcC.def**

■ ApplicationStates

```
ApplicationStates  record
                                :1
    AS_TRANSPARENT             :1
    AS_HAS_FULL_SCREEN_EXCL    :1
    AS_QUIT_DETACHING          :1
    AS_AVOID_TRANSPARENT_DETACH :1
    AS_TRANSPARENT_DETACHING   :1
    AS_REAL_DETACHING          :1
    AS_QUITTING               :1
    AS_DETACHING               :1
    AS_FOCUSABLE               :1
    AS_MODELABLE               :1
    AS_NOT_USER_INTERACTIBLE   :1
    AS_RECEIVED_APP_OBJECT_DETACH :1
    AS_ATTACHED_TO_STATE_FILE  :1
    AS_ATTACHING               :1
ApplicationStates  end
```

AS_TRANSPARENT

Set if the application is running in UILM_TRANSPARENT mode.

AS_HAS_FULL_SCREEN_EXCL

Set if the application has the full screen.

AS_SINGLE_INSTANCE

Set if the application is not capable of being launched multiple times.

AS_QUIT_DETACHING

Set if the detach sequence has been initiated as the result of a *quit*.



AS_AVOID_TRANSPARENT_DETACH

Set if this application should ignore transparent detaches.

AS_TRANSPARENT_DETACHING

Set if this application is being transparently-detached, that is, being shutdown to state because another application has been started in this application's field and that field is in UILM_TRANSPARENT mode.

AS_REAL_DETACHING

Set while the application is irreversibly detaching, after the UI has been detached and the application's GS_USABLE bit has been cleared.

AS_QUITTING

Set if the application is currently quitting.

AS_DETACHING

Set if the app object is detaching.

AS_FOCUSABLE

Set if the application may receive the "Focus" exclusive. If set, the application will be given the focus exclusive within its field, when launched, or clicked in by the user. This bit is set TRUE by default. This bit is copied to the **GeodeWinFlags** stored as part of the geode upon load, which act to guide the window system.

AS_MODELABLE

Set if the application may receive the "Model" exclusive. If set, the application will be given the model exclusive within its field, when launched, or clicked in by the user. Unless you're doing something odd, you'll want to have this match the state of your GA_TARGETABLE bit. This bit is set TRUE by default. This bit is copied to the **GeodeWinFlags** stored as part of the geode upon load, which act to guide the window system.

AS_NOT_USER_INTERACTABLE

Clear if this is a standard application which has at least one primary window or other interactable window on-screen. If this bit is set, then the UI need not provide options to navigate or select this application for user interaction.

AS_RECEIVED_APP_OBJECT_DETACH

Set if we have received a MSG_GEN_APPLICATION_OBJECT_DETACH.

AS_ATTACHED_TO_STATE_FILE

Set if we are attached to a state file.

AS_ATTACHING

Set if the application is in the process of attaching.

Library: **Objects/gAppC.def**

Reference book

■ AppMeasurementType

```
AppMeasurementType  etype      byte
    AMT_US           enum       AppMeasurementType, MEASURE_US
    AMT_METRIC       enum       AppMeasurementType, MEASURE_METRIC
    AMT_DEFAULT      enum       AppMeasurementType, 0xff ; use system default
```

Library: **Objects/gAppC.def**

■ AppNavigationID

```
AppNavigationID  etype      word, NAVIGATION_ID_APP_START
    NAVIGATION_ID_START_OF_RANGE equ 0x8000
```

This mask is OR-ed into a navigation ID number (which is given in HINT_NAVIGATION_ID); this bit specifies that the ID number serves as the start of a range.

Library: **Objects/genC.def**

■ AppUIData

```
AppUIData          struct
    AUID_specificUI      hptr          ;handle of specific UI
    AUID_displayScheme    DisplayScheme <>
    AUID_noTargetNoFocusReg  fptr.Region ;cursor for no target, no focus
    AUID_targetNoFocusReg   fptr.Region ;cursor for target, no focus
    AUID_noTargetFocusReg   fptr.Region ;cursor for no target, focus
    AUID_targetFocusReg     fptr.Region ;cursor for target, focus
    AUID_textKbdBindings    fptr          ;VisText kbd bindings.
    AUID_textEditCursor     optr.PointerDef ;handle:chunk to PointerDef
                                          ;(cursor) for text editing

AppUIData          ends
```

This structure stores the UI data stored with each process.

Library: **Objects/gAppC.def**

■ ApplicationOptFlags

```
ApplicationOptFlags  record
    AOF_MULTIPLE_INIT_FILE_CATEGORIES:1
                                :7
```

```
ApplicationOptFlags  ends
```

AOF_MULTIPLE_INIT_FILE_CATEGORIES

This is an optimization flag for UserGetIniCategory. Keep it clear if an application has only one init file category.

Library: **Objects/gAppC.def**



■ ArcCloseType

ArcCloseType	etype	word
ACT_OPEN	enum	ArcCloseType ; illegal for filled arcs
ACT_CHORD	enum	ArcCloseType ; draw/fill as a chord
ACT_PIE	enum	ArcCloseType ; draw/fill as a pie

Library: **graphics.def**

■ ArcBasicInit

```

ArcBasicInit      struct
  ABI_arcCloseType ArcCloseType
  ABI_startAngle   WWFixed
  ABI_endAngle     WWFixed
  ABI_startPoint   PointWWFixed
  ABI_endPoint     PointWWFixed
  ABI_midPoint     PointWWFixed
  ABI_radius       WWFixed
ArcBasicInitends

```

Library: **grobj.def**

■ ArcParams

```

ArcParams      struct
  AP_close      ArcCloseType      ; how the arc should be closed
  AP_left       sword              ; ellipse bounding box: left
  AP_top        sword              ;                               top
  AP_right      sword              ;                               right
  AP_bottom     sword              ;                               bottom
  AP_angle1     sword              ; start angle for arc
  AP_angle2     sword              ; ending angle for arc
ArcParams      ends

```

ArcParams is a structure passed to several arc construction routines.

Library: **graphics.def**

■ AreaAttr

```

AreaAttr      struct
  AA_colorFlag  ColorFlag CF_INDEX ; RGB or INDEX
  AA_color      RGBValue <0,0,0>   ; RGB values or index
  AA_mask       SystemDrawMask      ; draw mask
  AA_mapMode    ColorMapMode        ; color map mode
AreaAttr      ends

```

This structure is used with **GrSetAreaAttr**.

Library: **graphics.def**

Reference book

■ ArgumentStackElement

```
ArgumentStackElement    struct
    ASE_type      EvalStackArgumentType ; The type of argument.
    ASE_data      EvalStackArgumentData ; The associated data.
ArgumentStackElement    ends
```

Library: **parse.def**

■ BackgroundColors

```
BackgroundColors    struc
    BC_unselectedColor1    byte ;the two colors to use when unselected
    BC_unselectedColor2    byte
    BC_selectedColor1      byte ;the two colors to use when selected
    BC_selectedColor2      byte
BackgroundColors    ends
```

Library: **Objects/genC.def**

■ BasicGrab

```
BasicGrab    struct
    BG_OD      optr
    BG_data     word
BasicGrab    ends
```

This structure is used for grab mechanisms where a single optr has the grab at any moment in time, and when methods should be sent out to notify optrs of their gaining or losing of the grab. The *BG_data* field is solely to keep the struct the same size as the **MouseGrab** structures, so that common routines may operate on the different structures.

Library: **Objects/uiInputC.def**

■ BasicInit

```
BasicInit    struct
    BI_center    PointDWFxed
    BI_width     WWFixed
    BI_height    WWFixed
    BI_transform GrObjTransMatrix
    align        word
BasicInit    ends
```

Library: **grobj.def**



■ **BBFixed**

```
BBFixed struct
    BBF_frac    byte
    BBF_int     byte
BBFixed ends
```

This structure stores an 8 bit/8 bit fixed point number.

Library: **geos.def**

■ **BCCToolboxFeatures**

```
BCCToolboxFeatures    record
BCCToolboxFeatures    end
```

Library: **Objects/Text/tCtrlC.def**

■ **BCFeatures**

```
BCFeatures            record
    BCF_LIST           :1
    BCF_CUSTOM         :1
BCFeatures            end
```

Library: **Objects/Text/tCtrlC.def**

■ **BCToolboxFeatures**

```
BCToolboxFeatures    record
BCToolboxFeatures    end
```

Library: **Objects/Text/tCtrlC.def**

■ **Bitmap**

```
Bitmap struct
    B_width      word
    B_height     word
    B_compact    BMCompact BMC_UNCOMPACTED
    B_type       BMType <0,0,0,0,BMF_MONO>
Bitmap ends
```

This structure stores information about a simple graphics bitmap.

B_width and *B_height* store the width and height of the bitmap, in points (pixels).

 **Reference book**

B_compact stores the method of compaction in use by this bitmap.

B_type stores the bitmap type (**BMType**).

Library: **graphics.def**

■ BitmapGuardianBitmapPointerActiveStatus

```
BitmapGuardianBitmapPointerActiveStatus etype byte, 0
  BGBPAS_ACTIVE      enum      BitmapGuardianBitmapPointerActiveStatus
  BGBPAS_INACTIVE    enum      BitmapGuardianBitmapPointerActiveStatus
```

Library: **grobj.def**

■ BitmapGuardianFlags

```
BitmapGuardianFlags      record
  BGF_POINTER_ACTIVE      :1
  BGF_REAL_ESTATE_RESIZE  :1
BitmapGuardianFlags      end
```

BGF_POINTER_ACTIVE

This flag specifies that a floater is a **BitmapPointer**, so the bitmap should display handles instead of a dotted box for its edit indicator and it should respond to clicks on those handles. The **BitmapPointer** is used for changing the bitmap width and height.

BGF_REAL_ESTATE_RESIZE

This flag specifies that the current resize action is actually a real estate resize.

Library: **grobj.def**

■ BitmapGuardianSpecificInitializationData

```
BitmapGuardianSpecificInitializationData struct
  BGSID_toolClass      fptr.ClassStruct
  BGSID_activeStatus    VisWardToolActiveStatus
BitmapGuardianSpecificInitializationData ends
```

Library: **grobj.def**



■ BitmapMode

```
BitmapMode      record
    BM_EDIT_MASK      :14
    BM)CLUSTERED_DITHER :1
    BM)CLUSTERED_DITHER :1
BitmapMode      end
```

BM_EDIT_MASK

This flag specifies whether the mask is edited.

BM_CLUSTERED_DITHER

This flag specifies that the bitmap uses a clustered dither instead of a dispersed dither.

Library: **graphics.def**

■ BLTMode

```
BLTMode etype      word
    BLTM_COPY      enum BLTMode      ; 0 = copy image
    BLTM_MOVE      enum BLTMode      ; 1 = move image
    BLTM_CLEAR      enum BLTMode      ; 2 = clear source rect
```

Library: **graphics.def**

■ BMCompact

```
BMCompact      etype      byte
    BMC_UNCOMPACTED      enum BMCompact      ; 0 = no compaction
    BMC_PACKBITS      enum BMCompact      ; 1 = Mac packbits
    BMC_USER_DEFINED      enum BMCompact, 0x80 ; >0x80 = user defined
                                                ; compaction
```

This data structure is used to specify what sort of compaction is used to store a graphics bitmap.

Library: **graphics.def**

■ BMDestroy

```
BMDestroy      etype      byte
    BMD_KILL_DATA      enum BMDestroy      ; 0 = free bitmap (HugeArray)
    BMD_LEAVE_DATA      enum BMDestroy      ; 1 = leave bitmap data alone
```

Library: **graphics.def**

■ **BMFormat**

```

BMFormat          etype          byte, 0
    BMF_MONO       enum BMFormat   ; 0 = monochrome
    BMF_4BIT       enum BMFormat   ; 1 = 4-bit (EGA,VGA)
    BMF_8BIT       enum BMFormat   ; 2 = 8-bit (MCGA,SVGA)
    BMF_24BIT      enum BMFormat   ; 3 = 24-bit (high end cards)
    BMF_4CMYK      enum BMFormat   ; 4 = 4-bit CMYK (printers)
    BMF_3CMY       enum BMFormat   ; 5 = 3-bit CMY (printers)

```

This type determines a graphic bitmap's depth.

Library: **graphics.def**

■ **BMTType**

```

BMTType  record
    BMT_PALETTE      :1
    BMT_HUGE         :1
    BMT_MASK         :1
    BMT_COMPLEX      :1
    BMT_FORMAT       BMFormat:3
BMTType  end

```

This record stores various facts about a graphics bitmap.

BMT_PALETTE

This flag indicates that this 0 no palette stored with bitmap ; 1 palette supplied. (This bit is ignored if BMT_COMPLEX = 0.)

BMT_HUGE

This flag indicates that the bitmap is stored in a **HugeArray**.

BMT_MASK

This flag specifies that a bitmap mask is stored along with bitmap data.

BMT_COMPLEX

This flag specifies that this is not a simple bitmap. This flag must set to use a palette.

BMT_FORMAT

The type of bitmap format (**BMFormat**) is specified here.

Library: **graphics.def**

■ **BooleanByte**

```

BooleanByte      etype          byte
    BB_FALSE      enum          BooleanByte, 0
    BB_TRUE       enum          BooleanByte, 255

```



Library: **geos.def**

■ BooleanWord

```
BooleanWord      etype      word
  BW_FALSE      enum      BooleanWord, 0
  BW_TRUE       enum      BooleanWord, 0ffffh
```

Library: **geos.def**

■ BoundingRectData

```
BoundingRectData  struct
  BRD_rect      RectDWFxed
  CheckHack     < (offset BRD_rect eq 0) >
  BRD_destGState hptr.GState
  BRD_parentGState hptr.GState
  BRD_initialized word
BoundingRectData  ends
```

BRD_initialized is zero if the rectangle has not been initialized. This entry is generally ignored except by groups.

Library: **grobj.def**

■ BranchReplaceParams

```
BranchReplaceParams struct
  BRP_searchParam dd (?)
  BRP_replaceParam dd (?)
  BRP_type      BranchReplaceParamType
BranchReplaceParams ends
```

BRP_searchParam stores the search parameter, which is compared with instance data. Single word compare values should be stored in the first word; single byte values should be stored in the first byte.

BRP_replaceParam stores the replace parameter, which replaces any instance data which matches the search parameter. Single word compare values should be stored in first word, single byte in first byte.

BRP_type stores the type of replace operation (**BranchReplaceParamType**).

Library: **Objects/genC.def**

■ BranchReplaceParamType

```
BranchReplaceParamType etype word
  BRPT_OUTPUT_OTPTr enum BranchReplaceParamType
```

This type is passed with MSG_GEN_BRANCH_REPLACE_PARAMS to specify the type of replacement operation to effect.

The type BRPT_OUTPUT_OPTR affects all optr's stored in output optr fields and action descriptors within the generic branch, replacing and relocating them. Generic linkage itself is *not* affected.

The following generic objects recognize this replacement operation:

GenTrigger, GenList, GenValue, GenText: action optr's,
GenView: output optr's.

Library: **Objects/genC.def**

■ Button

```
Button      etype      byte
  BUTTON_0   enum       Button
  BUTTON_1   enum       Button
  BUTTON_2   enum       Button
  BUTTON_3   enum       Button
```

Library: **input.def**

■ ButtonInfo

```
ButtonInfo      record
  BI_PRESS       :1
  BI_DOUBLE_PRESS :1
  BI_B3_DOWN     :1
  BI_B2_DOWN     :1
  BI_B1_DOWN     :1
  BI_B0_DOWN     :1
  BI_BUTTON      Button:2
ButtonInfo      end
```

This record defines the active state of a mouse's buttons.

Library: **input.def**

■ C_CallbackStruct

```
C_CallbackStruct  struc
  C_callbackType  CallbackType
  C_params        fptr
  C_returnDS      word
  C_u             C_CallbackUnion
  align word
C_CallbackStruct ends
```



Library: **parse.def**

■ C_CallbackUnion

```
C_CallbackUnion      union
  CT_ftt             CT_FFT_CallbackStruct
  CT_ntt             CT_NTT_CallbackStruct
  CT_cne             CT_CNE_CallbackStruct
  CT_cns             CT_CNS_CallbackStruct
  CT_ef              CT_EF_CallbackStruct
  CT_ln              CT_LN_CallbackStruct
  CT_ul              CT_UL_CallbackStruct
  CT_ff              CT_FF_CallbackStruct
  CT_fn              CT_FN_CallbackStruct
  CT_cc              CT_CC_CallbackStruct
  CT_ec              CT_EX_CallbackStruct
  CT_ntc             CT_NTC_CallbackStruct
  CT_ftc             CT_FTC_CallbackStruct
  CT_dc              CT_DC_CallbackStruct
  CT_sf              CT_SF_CallbackStruct
C_CallbackUnion      end
```

Library: **parse.def**

■ CallbackMessageData

```
CallbackMessageData  struct
  CBMD_callBackOD     optr
  CBMD_callBackMessage word
  CBMD_groupOD        optr
  CBMD_childOD        optr
  CBMD_extraData1     word
  CBMD_extraData2     word
CallbackMessageData  ends
```

Library: **grobj.def**

■ CallbackType

```

CallbackType      etype      byte, 0, 1
CT_FUNCTION_TO_TOKEN      enum      CallbackType
; Description:
;   Converts a function name to a function ID token.
; Pass:
;   ss:bp      = Pointer to ParserParameters structure
;   ds:si      = Pointer to the text of the identifier
;   cx         = Length of the identifier
; Return:
;   carry set if the text is a function name
;   di         = The Function-ID for the identifier
;
CT_NAME_TO_TOKEN      enum      CallbackType
; Description:
;   Converts a name to a name ID token.
; Pass:
;   ss:bp      = Pointer to ParserParameters structure
;   ds:si      = Pointer to the text of the name
;   cx         = Length of the name
; Return:
;   cx         = Token for the name
;   Carry set on error
;   al         = Error code
;
CT_CHECK_NAME_EXISTS      enum      CallbackType
; Description:
;   Checks whether a name already exists
; Pass:
;   ss:bp      = Pointer to ParserParameters structure
;   ds:si      = Pointer to the text of the name
;   cx         = Length of the name
; Return:
;   carry set if the name exists
;   carry clear otherwise
;
CT_CHECK_NAME_SPACE      enum      CallbackType
; Description:
;   Signals the need to allocate a certain number of names.
;   This avoids the problem of getting part way through
;   a set of name glaciations for an expression and running out
;   of space for the names.
; Pass:
;   ss:bp      = Pointer to ParserParameters structure
;   cx         = Number of names we want to allocate
; Return:
;   Carry set on error
;   al         = Error code
;

```

```

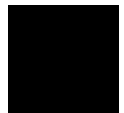
CT_EVAL_FUNCTION          enum   CallbackType
; Description:
;   Evaluates a function with parameters.
; Pass:
;   ss:bp          = Pointer to EvalParameters structure
;   cx              = Number of arguments
;   si              = Function ID
;   es:di           = Operator stack
;   es:bx           = Argument stack
; Return:
;   carry set on error
;   al              = Error code
;
CT_LOCK_NAME              enum   CallbackType
;
; Description:
;   Locks a name definition.
; Pass:
;   ss:bp          = Pointer to ParserParameters structure
;   cx              = Name token
; Return:
;   carry set on error
;   al              = Error code
;   ds:si           = Pointer to the definition
;
CT_UNLOCK                 enum   CallbackType
; Description:
;   Unlocks a name/function definition.
; Pass:
;   ss:bp          = Pointer to ParserParameters structure
;   ds              = Segment address of the data to unlock
;
CT_FORMAT_FUNCTION        enum   CallbackType
; Description:
;   Formats a function name into a buffer.
; Pass:
;   ss:bp          = Pointer to ParserParameters structure
;   es:di           = Buffer to store the text
;   cx              = Function token
;   dx              = Maximum number of characters that can be written
; Return:
;   es:di           = Pointer past the inserted text
;   dx              = Number of characters that were written
;
CT_FORMAT_NAME            enum   CallbackType
; Description:
;   Formats a name into a buffer.
; Pass:
;   ss:bp          = Pointer to ParserParameters structure
;   es:di           = Place to store the text

```

```

;   cx           = Name token
;   dx           = Maximum number of characters that can be written
; Return:
;   ss:bp        = Pointer to ParserParameters structure
;   es:di        = Pointer past the inserted text
;   dx           = Number of characters that were written
;
CT_CREATE_CELL      enum   CallbackType
; Description:
;   Creates a new empty cell. Used by the dependency code to
;   create a cell to add dependencies to.
; Pass:
;   ss:bp        = Pointer to ParserParameters structure
;   dx           = Row of the cell to create
;   cx           = Column of the cell to create
; Return:
;   carry set on error
;   al           = Error code
;
CT_EMPTY_CELL       enum   CallbackType
; Description:
;   Removes a cell if it's appropriate. This is called when a cell
;   has its last dependency removed.
; Pass:
;   ss:bp        = Pointer to ParserParameters structure
;   dx           = Row of the cell that now has no dependencies
;   cx           = Column of the cell that now has no dependencies
; Return:
;   carry set on error
;   al           = Error code
;
CT_NAME_TO_CELL     enum   CallbackType
; Description:
;   Converts a name into a cell to enable the addition of
;   dependencies to it.
; Pass:
;   ss:bp        = Pointer to ParserParameters structure
;   cx           = Name token
; Return:
;   dx           = Row of the cell containing the names
;                dependencies
;   cx           = Column of the cell containing the names
;                dependencies
;
CT_FUNCTION_TO_CELL enum   CallbackType
; Description:
;   Converts a function into a cell to enable the addition of
;   dependencies to it.
; Pass:
;   ss:bp        = Pointer to ParserParameters structure

```



```

;      cx      = Function-ID
; Return:
;      dx      = Row of the cell containing the functions ;
;              dependencies
;              = 0 if no dependency is required
;      cx      = Column of the cell containing the functions
;              dependencies.
;
CT_DEREF_CELL      enum   CallbackType
; Description:
;   Returns the contents of a cell. The callback is responsible for
;   popping the cell reference off the stack.
; Pass:
;   es:bx      = Pointer to the argument stack
;   es:di      = Pointer to operator/function stack
;   ss:bp      = Pointer to EvalParameters structure
;   dx         = Row of the cell to dereference
;   cx         = Column of the cell to dereference
; Return:
;   es:bx      = New pointer to the argument stack
;   carry set on error
;   al         = Error code
;
CT_SPECIAL_FUNCTION  enum   CallbackType
; Description:
;   Returns the value of one of the special functions.
; Pass:
;   es:bx      = Pointer to the argument stack
;   es:di      = Pointer to operator/function stack
;   ss:bp      = Pointer to EvalParameters structure
;   cx         = Special function
; Return:
;   es:bx      = New pointer to the argument stack
;   carry set on error
;   al         = Error code

```

Library: **parse.def**

■ CBitmap

```
CBitmap struct
  CB_simple      Bitmap <>          ; simple bitmap structure
  CB_startScan   word 0              ; starting row number
  CB_numScans    word 1              ; Number of scans of data in this slice
  ;
  ; the following three offsets are offsets from the start of the bitmap
  ; structure
  ;
  CB_devInfo     word 0              ; offset to device info
  CB_data        word (size CBitmap) ; offset to start of data
  CB_palette     word 0              ; offset to color table
  ; (if bit 6 set in B_type of Bitmap
  ; structure)
  CB_xres        word 72             ; x resolution (DPI)
  CB_yres        word 72             ; y resolution (DPI)
CBitmap ends
```

This structure stores information for a “complex” bitmap. (For a simple bitmap see the reference entry for **Bitmap**.)

CBitmap holds bitmap specifics such as resolution information, a palette, and mask data.

Library: **graphics.def**

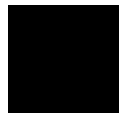
■ CDependencyStruct

```
CDependencyStruct struct
  DP_parameters   DependencyParameters
  DP_callbackPtr  fptr
  DP_callbackStruct C_CallbackStruct
CDependencyStruct ends
```

Library: **parse.def**

■ CellFunctionParameterFlags

```
CellFunctionParameterFlags record
  CFPF_DIRTY      :1      ;If set, the parameter block is dirty.
                  :4      ;Unused.
  CFPF_NO_FREE_COUNT :3    ;Set temporarily in RangeEnum to make sure
                          ;that a callback doesn't attempt to free
                          ;anything. These bits count the number of
                          ;calls to a non-special RangeEnum
CellFunctionParameterFlags end
```



Library: **cell.def**

■ CellFunctionParameters

```
CellFunctionParameters    struct
    CFP_flags             CellFunctionParameterFlags
    CFP_file              word
    CFP_rowBlocks         RowBlockList
CellFunctionParameters    ends
```

This structure is used to pass specifics about a cell file to the cell library routines. Some of the data in the **CellFunctionParameters** structure is opaque to the application (namely, the *CFP_flags*) and should not be modified by the application.

CFP_flags stores flags that are set and modified by the cell library. The only flag that is allowed to be checked or changed is the *CFPF_dirty* flag. The cell library routines set this bit whenever it changes the **CellFunctionParameters** structure, indicating that the structure needs to be resaved. After it is saved, you may clear this bit.

CFP_file must contain the VM file handle of the cell file. This field must be set each time you open the file. The cell library routines will always act on the file specified here, *not* on the VM override file (if any).

CFP_rowBlocks contains an array of VM block handles, one for every existing or potential row block. The length of this array is *N_ROW_BLOCKS* (defined in **cell.h**). When you create a cell file, initialize all of these handles to zero; do not access or change this field thereafter.

Warning: The cell library expects the **CellFunctionParameters** structure to remain motionless for the duration of the call. Therefore, if you allocate it as a DB item in a cell file, you must *not* have the structure be an ungrouped item.

Library: **cell.def**

■ CellRange

```
CellRange                struct
    CR_start              CellReference <>
    CR_end                CellReference <>
CellRange                ends
```

This structure specifies a rectangular range of cells

Library: **parse.def**

■ CellReference

```
CellReference      struct
    CR_row         CellRowColumn <>
    CR_column      CellRowColumn <>
CellReference      ends
```

Cell references can be absolute, relative, or mixed. If the cell reference is absolute, this structure specifies a particular cell; if the reference is relative, this structure specifies an offset from a previous cell position.

Library: **parse.def**

■ CellRowColumn

```
CellRowColumn      record
    CRC_ABSOLUTE   :1          ; Set if the reference is absolute
    CRC_VALUE      :15         ; The value of the row/column
CellRowColumn      end
```

Library: **parse.def**

■ CenterLeftRightWidth

```
CenterLeftRightWidth  etype      byte
    CLRW_CENTER       enum        CenterLeftRightWidth
    CLRW_LEFT         enum        CenterLeftRightWidth
    CLRW_RIGHT        enum        CenterLeftRightWidth
    CLRW_WIDTH        enum        CenterLeftRightWidth
```

Library: **grobj.def**

■ CenterTopBottomHeight

```
CenterTopBottomHeight etype      byte
    CTBH_CENTER       enum        CenterTopBottomHeight
    CTBH_TOP          enum        CenterTopBottomHeight
    CTBH_BOTTOM       enum        CenterTopBottomHeight
    CTBH_HEIGHT       enum        CenterTopBottomHeight
```

Library: **grobj.def**



■ CEvalStruct

```
CEvalStruct      struc
    CE_parameters      EvalParameters
    CE_callbackPtr      fptr
    CE_callbackStruct   C_CallbackStruct
CEvalStruct      ends
```

Library: **parse.def**

■ CFormatStruct

```
CFormatStruct    struc
    CF_parameters      FormatParameters
    CF_callbackPtr      fptr
    CF_callbackStruct   C_CallbackStruct
CFormatStruct    ends
```

Library: **parse.def**

■ CharSet

```
CharacterSet      etype      byte
    CS_BSW          enum CharSet, 0x00;Extended BSW set (printable) (Chars)
    CS_CONTROL       enum CharSet, 0xff;Control codes (non-printable) (VChar)
    CS_UI_FUNCS      enum CharSet, 0xfe;Special UI functions, not actually key
                                ;presses, defined in ui.def (UChar)

    VC_ISANSI        = CS_BSW
    VC_ISCTRL         = CS_CONTROL
    VC_ISUI           = CS_UI_FUNCS
```

Library: **input.def**

■ CharChoiceInformation

```
CharChoiceInformation  struct
    CCI_numChoices      word
    CCI_firstPoint      word
    CCI_lastPoint       word
    CCI_data             fptr.word
CharChoiceInformation  ends
```

CCI_numChoices stores the number of choices for this character (can be 0)

CCI_firstPoint stores the offset to the first point in the ink data corresponding to this char.

CCI_lastPoint stores the offset to the last point in the ink data corresponding to this char.

CCI_data stores the actual pointer to the characters.

Library: **hwr.def**

■ CharFlags

```
CharFlags          record
  CF_STATE_KEY      :1      ;Set if state key (shift/toggle modifier)
                   :2
  CF_EXTENDED       :1      ;TRUE: extended key
  CF_TEMP_ACCENT    :1      ;Set if temporary accent char
  CF_FIRST_PRESS    :1      ;Set if initial key press
  CF_REPEAT_PRESS   :1      ;Set if repeated key press
  CF_RELEASE        :1      ;Set if key release (may be set in conjunction
                           ;with the other two, by monitors or UI to lessen
                           ;number of events)

CharFlags          end
```

Library: **input.def**



■ Chars

Chars	etype	byte
C_NULL	enum Chars,	0x0 ;NULL
C_CTRL_A	enum Chars,	0x1 ;<ctrl>-A
C_CTRL_B	enum Chars,	0x2 ;<ctrl>-B
C_CTRL_C	enum Chars,	0x3 ;<ctrl>-C
C_CTRL_D	enum Chars,	0x4 ;<ctrl>-D
C_CTRL_E	enum Chars,	0x5 ;<ctrl>-E
C_CTRL_F	enum Chars,	0x6 ;<ctrl>-F
C_CTRL_G	enum Chars,	0x7 ;<ctrl>-G
C_CTRL_H	enum Chars,	0x8 ;<ctrl>-H
C_TAB	enum Chars,	0x9 ; TAB
C_LINEFEED	enum Chars,	0xa ; LINE FEED
C_CTRL_K	enum Chars,	0xb ;<ctrl>-K
C_CTRL_L	enum Chars,	0xc ;<ctrl>-L
C_ENTER	enum Chars,	0xd ; ENTER or CR
C_SHIFT_OUT	enum Chars,	0xe ;<ctrl>-N
C_SHIFT_IN	enum Chars,	0xf ;<ctrl>-O
C_CTRL_P	enum Chars,	0x10 ;<ctrl>-P
C_CTRL_Q	enum Chars,	0x11 ;<ctrl>-Q
C_CTRL_R	enum Chars,	0x12 ;<ctrl>-R
C_CTRL_S	enum Chars,	0x13 ;<ctrl>-S
C_CTRL_T	enum Chars,	0x14 ;<ctrl>-T
C_CTRL_U	enum Chars,	0x15 ;<ctrl>-U
C_CTRL_V	enum Chars,	0x16 ;<ctrl>-V
C_CTRL_W	enum Chars,	0x17 ;<ctrl>-W
C_CTRL_X	enum Chars,	0x18 ;<ctrl>-X
C_CTRL_Y	enum Chars,	0x19 ;<ctrl>-Y
C_CTRL_Z	enum Chars,	0x1a ;<ctrl>-Z
C_ESCAPE	enum Chars,	0x1b ;ESC
; common shortcuts for low 32 codes		
C_NUL	=	C_NULL
C_STX	=	C_CTRL_B
C_ETX	=	C_CTRL_C
C_BEL	=	C_CTRL_G
C_BS	=	C_CTRL_H
C_HT	=	C_CTRL_I
C_VT	=	C_CTRL_K
C_FF	=	C_CTRL_L
C_SO	=	C_CTRL_N
C_SI	=	C_CTRL_O
C_DC1	=	C_CTRL_Q
C_DC2	=	C_CTRL_R
C_DC3	=	C_CTRL_S
C_DC4	=	C_CTRL_T
C_CAN	=	C_CTRL_X
C_EM	=	C_CTRL_Y

```

C_ESC          = C_ESCAPE
;
; some alternative names:
;
C_CR           = C_ENTER
C_CTRL_M      = C_ENTER
C_CTRL_I      = C_TAB
C_CTRL_J      = C_LINEFEED
C_LF          = C_LINEFEED
C_CTRL_N      = C_SHIFT_OUT
C_CTRL_O      = C_SHIFT_IN

C_NULL_WIDTH   enum Chars, 0x19    ; null width character
C_GRAPHIC      enum Chars, 0x1a    ; Graphic in text.
C_THINSPACE    enum Chars, 0x1b    ; 1/4 width space
C_ENSPACE      enum Chars, 0x1c    ; En-space, fixed width
C_EMSPACE      enum Chars, 0x1d    ; Em-space, fixed width.

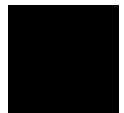
C_SECTION_BREAK enum Chars, C_CTRL_K
C_PAGE_BREAK    enum Chars, C_CTRL_L

C_COLUMN_BREAK =          C_PAGE_BREAK

C_NONBRKHYPHEN enum Chars, 0x1e    ; Non breaking hyphen.
C_OPHTYPHEN     enum Chars, 0x1f    ; Optional hyphen, only drawn
                                           ; at end of line.

C_FS           = C_ENSPACE
C_FIELD_SEP     = C_FS
;
; the standard ASCII chars:
;
C_SPACE         enum Chars, ' '
C_EXCLAMATION   enum Chars, '!'
C_QUOTE         enum Chars, '"'
C_NUMBER_SIGN   enum Chars, '#'
C_DOLLAR_SIGN   enum Chars, '$'
C_PERCENT       enum Chars, '%'
C_AMPERSAND     enum Chars, '&'
C_SNG_QUOTE     enum Chars, 0x27
C_LEFT_PAREN    enum Chars, '('
C_RIGHT_PAREN   enum Chars, ')'
C_ASTERISK      enum Chars, '*'
C_PLUS         enum Chars, '+'
C_COMMA         enum Chars, ','
C_MINUS         enum Chars, '-'
C_PERIOD        enum Chars, '.'
C_SLASH         enum Chars, '/'
C_ZERO          enum Chars, '0'
C_ONE           enum Chars, '1'
C_TWO           enum Chars, '2'

```



C_THREE	enum Chars, '3'
C_FOUR	enum Chars, '4'
C_FIVE	enum Chars, '5'
C_SIX	enum Chars, '6'
C_SEVEN	enum Chars, '7'
C_EIGHT	enum Chars, '8'
C_NINE	enum Chars, '9'
C_COLON	enum Chars, ':'
C_SEMICOLON	enum Chars, ';'
C_LESS_THAN	enum Chars, '<'
C_EQUAL	enum Chars, '='
C_GREATER_THAN	enum Chars, '>'
C_QUESTION_MARK	enum Chars, '?'
C_AT_SIGN	enum Chars, '@'
C_CAP_A	enum Chars, 'A'
C_CAP_B	enum Chars, 'B'
C_CAP_C	enum Chars, 'C'
C_CAP_D	enum Chars, 'D'
C_CAP_E	enum Chars, 'E'
C_CAP_F	enum Chars, 'F'
C_CAP_G	enum Chars, 'G'
C_CAP_H	enum Chars, 'H'
C_CAP_I	enum Chars, 'I'
C_CAP_J	enum Chars, 'J'
C_CAP_K	enum Chars, 'K'
C_CAP_L	enum Chars, 'L'
C_CAP_M	enum Chars, 'M'
C_CAP_N	enum Chars, 'N'
C_CAP_O	enum Chars, 'O'
C_CAP_P	enum Chars, 'P'
C_CAP_Q	enum Chars, 'Q'
C_CAP_R	enum Chars, 'R'
C_CAP_S	enum Chars, 'S'
C_CAP_T	enum Chars, 'T'
C_CAP_U	enum Chars, 'U'
C_CAP_V	enum Chars, 'V'
C_CAP_W	enum Chars, 'W'
C_CAP_X	enum Chars, 'X'
C_CAP_Y	enum Chars, 'Y'
C_CAP_Z	enum Chars, 'Z'
C_LEFT_BRACKET	enum Chars, '['
C_BACKSLASH	enum Chars, 0x5c
C_RIGHT_BRACKET	enum Chars, ']'
C_ASCII_CIRCUMFLEX	enum Chars, '^'
C_UNDERSCORE	enum Chars, '_'
C_BACKQUOTE	enum Chars, '`'
C_SMALL_A	enum Chars, 'a'
C_SMALL_B	enum Chars, 'b'
C_SMALL_C	enum Chars, 'c'
C_SMALL_D	enum Chars, 'd'

```

C_SMALL_E      enum Chars, 'e'
C_SMALL_F      enum Chars, 'f'
C_SMALL_G      enum Chars, 'g'
C_SMALL_H      enum Chars, 'h'
C_SMALL_I      enum Chars, 'i'
C_SMALL_J      enum Chars, 'j'
C_SMALL_K      enum Chars, 'k'
C_SMALL_L      enum Chars, 'l'
C_SMALL_M      enum Chars, 'm'
C_SMALL_N      enum Chars, 'n'
C_SMALL_O      enum Chars, 'o'
C_SMALL_P      enum Chars, 'p'
C_SMALL_Q      enum Chars, 'q'
C_SMALL_R      enum Chars, 'r'
C_SMALL_S      enum Chars, 's'
C_SMALL_T      enum Chars, 't'
C_SMALL_U      enum Chars, 'u'
C_SMALL_V      enum Chars, 'v'
C_SMALL_W      enum Chars, 'w'
C_SMALL_X      enum Chars, 'x'
C_SMALL_Y      enum Chars, 'y'
C_SMALL_Z      enum Chars, 'z'
C_LEFT_BRACE   enum Chars, '{'
C_VERTICAL_BAR enum Chars, '|'
C_RIGHT_BRACE  enum Chars, '}'
C_ASCII_TILDE  enum Chars, '~'
C_DELETE       enum Chars, 0x7f
;
; some alternative names:
;
C_HYPHEN       = C_MINUS
C_GRAVE        = C_BACKQUOTE
C_UA_DIERESIS  enum Chars, 0x80
C_UA_RING      enum Chars, 0x81
C_UC_CEDILLA   enum Chars, 0x82
C_UE_ACUTE     enum Chars, 0x83
C_UN_TILDE     enum Chars, 0x84
C_UO_DIERESIS  enum Chars, 0x85
C_UU_DIERESIS  enum Chars, 0x86
C_LA_ACUTE     enum Chars, 0x87
C_LA_GRAVE     enum Chars, 0x88
C_LA_CIRCUMFLEX enum Chars, 0x89
C_LA_DIERESIS  enum Chars, 0x8a
C_LA_TILDE     enum Chars, 0x8b
C_LA_RING      enum Chars, 0x8c
C_LC_CEDILLA   enum Chars, 0x8d
C_LE_ACUTE     enum Chars, 0x8e
C_LE_GRAVE     enum Chars, 0x8f
C_LE_CIRCUMFLEX enum Chars, 0x90
C_LE_DIERESIS  enum Chars, 0x91

```



C_LI_ACUTE	enum Chars, 0x92
C_LI_GRAVE	enum Chars, 0x93
C_LI_CIRCUMFLEX	enum Chars, 0x94
C_LI_DIERESIS	enum Chars, 0x95
C_LN_TILDE	enum Chars, 0x96
C_LO_ACUTE	enum Chars, 0x97
C_LO_GRAVE	enum Chars, 0x98
C_LO_CIRCUMFLEX	enum Chars, 0x99
C_LO_DIERESIS	enum Chars, 0x9a
C_LO_TILDE	enum Chars, 0x9b
C_LU_ACUTE	enum Chars, 0x9c
C_LU_GRAVE	enum Chars, 0x9d
C_LU_CIRCUMFLEX	enum Chars, 0x9e
C_LU_DIERESIS	enum Chars, 0x9f
C_DAGGER	enum Chars, 0xa0
C_DEGREE	enum Chars, 0xa1
C_CENT	enum Chars, 0xa2
C_STERLING	enum Chars, 0xa3
C_SECTION	enum Chars, 0xa4
C_BULLET	enum Chars, 0xa5
C_PARAGRAPH	enum Chars, 0xa6
C_GERMANDBLS	enum Chars, 0xa7
C_REGISTERED	enum Chars, 0xa8
C_COPYRIGHT	enum Chars, 0xa9
C_TRADEMARK	enum Chars, 0xaa
C_ACUTE	enum Chars, 0xab
C_DIERESIS	enum Chars, 0xac
C_NOTEQUAL	enum Chars, 0xad
C_U_AE	enum Chars, 0xae
C_UO_SLASH	enum Chars, 0xaf
C_INFINITY	enum Chars, 0xb0
C_PLUSMINUS	enum Chars, 0xb1
C_LESSEQUAL	enum Chars, 0xb2
C_GREATEREQUAL	enum Chars, 0xb3
C_YEN	enum Chars, 0xb4
C_L_MU	enum Chars, 0xb5
C_L_DELTA	enum Chars, 0xb6
C_U_SIGMA	enum Chars, 0xb7
C_U_PI	enum Chars, 0xb8
C_L_PI	enum Chars, 0xb9
C_INTEGRAL	enum Chars, 0xba
C_ORDFEMININE	enum Chars, 0xbb
C_ORDMASCULINE	enum Chars, 0xbc
C_U_OMEGA	enum Chars, 0xbd
C_L_AE	enum Chars, 0xbe
C_LO_SLASH	enum Chars, 0xbf
C_QUESTIONDOWN	enum Chars, 0xc0
C_EXCLAMDOWN	enum Chars, 0xc1
C_LOGICAL_NOT	enum Chars, 0xc2
C_ROOT	enum Chars, 0xc3

C_FLORIN	enum Chars, 0xc4
C_APPROX_EQUAL	enum Chars, 0xc5
C_U_DELTA	enum Chars, 0xc6
C_GUILLEDBLLEFT	enum Chars, 0xc7
C_GUILLEDBLRIGHT	enum Chars, 0xc8
C_ELLIPSIS	enum Chars, 0xc9
C_NONBRKSPACE	enum Chars, 0xca
C_UA_GRAVE	enum Chars, 0xcb
C_UA_TILDE	enum Chars, 0xcc
C_UO_TILDE	enum Chars, 0xcd
C_U_OE	enum Chars, 0xce
C_L_OE	enum Chars, 0xcf
C_ENDASH	enum Chars, 0xd0
C_EMDASH	enum Chars, 0xd1
C_QUOTEDBLLEFT	enum Chars, 0xd2
C_QUOTEDBLRIGHT	enum Chars, 0xd3
C_QUOTESNGLEFT	enum Chars, 0xd4
C_QUOTESNGRIGHT	enum Chars, 0xd5
C_DIVISION	enum Chars, 0xd6
C_DIAMONDBULLET	enum Chars, 0xd7
C_LY_DIERESIS	enum Chars, 0xd8
C_UY_DIERESIS	enum Chars, 0xd9
C_FRACTION	enum Chars, 0xda
C_CURRENCY	enum Chars, 0xdb
C_GUILSNGLEFT	enum Chars, 0xdc
C_GUILSNGRIGHT	enum Chars, 0xdd
C_LY_ACUTE	enum Chars, 0xde
C_UY_ACUTE	enum Chars, 0xdf
C_DBLDAGGER	enum Chars, 0xe0
C_CNTR_DOT	enum Chars, 0xe1
C_SNGQUOTELOW	enum Chars, 0xe2
C_DBLQUOTELOW	enum Chars, 0xe3
C_PERTHOUSAND	enum Chars, 0xe4
C_UA_CIRCUMFLEX	enum Chars, 0xe5
C_UE_CIRCUMFLEX	enum Chars, 0xe6
C_UA_ACUTE	enum Chars, 0xe7
C_UE_DIERESIS	enum Chars, 0xe8
C_UE_GRAVE	enum Chars, 0xe9
C_UI_ACUTE	enum Chars, 0xea
C_UI_CIRCUMFLEX	enum Chars, 0xeb
C_UI_DIERESIS	enum Chars, 0xec
C_UI_GRAVE	enum Chars, 0xed
C_UO_ACUTE	enum Chars, 0xee
C_UO_CIRCUMFLEX	enum Chars, 0xef
C_LOGO	enum Chars, 0xf0
C_UO_GRAVE	enum Chars, 0xf1
C_UU_ACUTE	enum Chars, 0xf2
C_UU_CIRCUMFLEX	enum Chars, 0xf3
C_UU_GRAVE	enum Chars, 0xf4
C_LI_DOTLESS	enum Chars, 0xf5

```

C_CIRCUMFLEX      enum Chars, 0xf6
C_TILDE           enum Chars, 0xf7
C_MACRON          enum Chars, 0xf8
C_BREVE           enum Chars, 0xf9
C_DOTACCENT       enum Chars, 0xfa
C_RING            enum Chars, 0xfb
C_CEDILLA         enum Chars, 0xfc
C_HUNGARUMLAT    enum Chars, 0xfd
C_OGONEK          enum Chars, 0xfe
C_CARON           enum Chars, 0xff
;
; some alternative names:
;
C_PARTIAL_DIFF    = C_L_DELTA
C_SUM             = C_U_SIGMA
C_PRODUCT         = C_U_PI
C_RADICAL         = C_ROOT
C_LOZENGE         = C_DIAMONDBULLET

```

Library: **char.def**

■ CharTableData

```

CharTableData      struct
    CTD_line1      optr
    CTD_line2      optr
    CTD_line3      optr
    CTD_line4      optr
    CTD_line5      optr
CharTableData      ends

```

This structure is used during notification of the pen object.

Library: **Objects/gPenICC.def**

■ ChunkArrayHeader

```

ChunkArrayHeader    struct
    CAH_count      word
    CAH_elementSize word
    CAH_curOffset  word
    CAH_offset      word
ChunkArrayHeader    ends

```

Every chunk array begins with a **ChunkArrayHeader**. This structure contains the basic information about the associated chunk array. Applications should never change the contents of the **ChunkArrayHeader**; only the chunk array routines should do this. However, applications can examine the header if they wish.

CAH_count stores the number of elements in the chunk array.

CAH_elementSize stores the size of each element in the chunk array if the elements are each of the same size. If the elements are variable-sized, this entry will be zero.

CAH_curOffset stores bookkeeping information pointing to the current element in use during an enumeration.

CAH_offset stores the offset from the start of the chunk to the first element in the array.

Library: **chunkarr.def**

■ ChunkMapList

```
ChunkMapList      struc
    CML_source     word
    CML_dest       word
ChunkMapList      ends
```

Library: **impex.def**

■ ClassFlags

```
ClassFlags        record
    CLASSF_HAS_DEFAULT      :1      ; Set if dword before the class record
                                ; contains an fptr of a default method
                                ; handler to deal with any unrecognized
                                ; method send to an object of the class.

    CLASSF_MASTER_CLASS     :1      ; Set if class is a master class
    CLASSF_VARIANT_CLASS    :1      ; Set if superclass varies
    CLASSF_DISCARD_ON_SAVE  :1      ; Set if class data can be discarded
                                ; when object is saved
    CLASSF_NEVER_SAVED      :1      ; Set if objects of this class
                                ; are never saved. This is a signal
                                ; to Esp that it needn't build up
                                ; a relocation table for the class
    CLASSF_HAS_RELOC        :1      ; Set if dword after method table is
                                ; routine to call to relocate or
                                ; unrelocate an object. Routine is
                                ; passed MSG_META_RELOCATE or
                                ; MSG_META_UNRELOCATE in AX.
    CLASSF_C_HANDLERS       :1      ; Handlers are written in C and must
                                ; be called with the C convention
                                :1
ClassFlags        end
```

This record is stored in the **ClassStruct** structure's *Class_flags* field. These flags are internal and may not be set or retrieved directly.



Library: **object.def**

■ ClassStruct

```
ClassStruct      struct
  Class_superClass      fptr.ClassStruct
  Class_masterOffset    word
  Class_methodCount     word
  Class_instanceSize    word
  Class_vdRelocTable    nptr.VarObjRelocation
  Class_relocTable      nptr.ObjRelocation
  Class_flags           ClassFlags
  Class_masterMethods   byte
  Class_methodTable     label word
ClassStruct      ends
```

This structure contains the arguments which define a class. It is internal and used only very rarely by anything other than the kernel and the UI.

Class_superClass stores the **ClassStruct** of this class's superclass.

Class_masterOffset stores the offset to the master class data.

Class_methodCount stores the number of methods defined for this class. This is used to determine the size of the method table, which follows this **ClassStruct**.

Class_instanceSize stores the size of the entire master group's instance data.

Class_vdRelocTable stores the offset to the class' relocatable vardata table.

Class_relocTable stores the offset to the class' relocatable instance data table.

Class_flags stores the **ClassFlags** in use by the class.

Class_masterMessages stores internal flags that Esp uses to indicate the presence of method handlers for a given master level.

Class_methodTable marks the start of the class' method table.

Library: **object.def**

■ ClipboardItemFlags

```
ClipboardItemFlags record
  CIF_UNUSED      :1
  CIF_QUICK       :1
  CIF_UNUSED2     :14
ClipboardItemFlags end
```

Library: **Objects/clipbrd.def**

Reference book

■ ClipboardItemFormat

ClipboardItemFormat	etype	word
CIF_TEXT	enum	ClipboardItemFormat
CIF_GRAPHICS_STRING	enum	ClipboardItemFormat
CIF_FILES	enum	ClipboardItemFormat
CIF_SPREADSHEET	enum	ClipboardItemFormat
CIF_INK	enum	ClipboardItemFormat
CIF_GROBJ	enum	ClipboardItemFormat
CIF_GEODEX	enum	ClipboardItemFormat
CIF_BITMAP	enum	ClipboardItemFormat
CIF_SOUND_SYNT	enum	ClipboardItemFormat
CIF_SOUND_SAMPLE	enum	ClipboardItemFormat

CIF_TEXT

The contents of the clipboard are null terminated text (with possible formatting information).

CIF_GRAPHICS_STRING

The contents of the clipboard is a standard GEOS graphics string.

CIF_FILES

The contents of the clipboard are in an internal desktop format for direct-manipulation file operations.

Library: **geoworks.def**

■ ClipboardItemFormatID

ClipboardItemFormatID	struct
CIFID_manufacturer	ManufacturerID
CIFID_type	ClipboardItemFormat
ClipboardItemFormatID	ends

Format IDs are identified by two words. One is a manufacturer ID and the other is a manufacturer-specific value that specifies the actual format.

Library: **Objects/clipbrd.def**



■ ClipboardItemFormatInfo

```
ClipboardItemFormatInfo  struct
;
; two words of format identification
;
CIFI_format      ClipboardItemFormatID
;
; two words of format-specific extra data
;      (not used for CIF_TEXT, gstring size for CIF_GRAPHICS_STRING,
;      not used for CIF_FILES)
;
CIFI_extra1      word
CIFI_extra2      word
;
; VM block handle of first block in linked chain of data blocks
;
CIFI_vmChain     dword
;
; token of application that knows how to render this format
; (not currently used)
;
CIFI_renderer    GeodeToken <>
ClipboardItemFormatInfo  ends
```

A clipboard item header contains all of the data for the item in all formats supported by the owner. Each format is identified by a structure that stores the format type, two words of format-specific extra data, and the VM block handle of the first VM block in a chain of VM data blocks for the format.

Library: **Objects/clipbrd.def**

■ ClipboardItemHeader

```
ClipboardItemHeader      struct
CIH_owner               optr
CIH_flags               ClipboardItemFlags
CIH_name                ClipboardItemNameBuffer
CIH_formatCount         word
CIH_sourceID            optr
CIH_formats             ClipboardItemFormatInfo CLIPBOARD_MAX_FORMATS dup (<>)
CIH_reserved            dword
ClipboardItemHeader      ends
```

This structure is passed to **ClipboardRegisterItem**, **ClipboardRequestItemFormat**, **ClipboardDoneWithItem** and returned from **ClipboardQueryItem**.

CIH_owner stores the owner of the transfer item - this is cleared when a clipboard item is saved to disk when shutting down. Note that only normal transfer items persist across shutdown.

CIH_flags stores the quick/normal (quick item is only temporary).

CIH_name stores the name of this clipboard item.

CIH_formatCount stores the number of data formats available.

CIH_sourceID stores the optr of additional info about transfer item source (used to determine default move/copy behavior during quick transfer). “source document ID” -- most things will want to put the optr of the parent GenDocument object here.

CIH_formats stores the data formats available ordered from most informative (includes VM block handles containing data for the format) (all formats for a given transfer item must be in the same VM file).

CIH_reserved is reserved for future expansion (must be 0 for now).

Library: **Objects/clipbrd.def**

■ ClipboardQuickNotifyFlags

```
ClipboardQuickNotifyFlags    record
    CQNF_ERROR                :1
    CQNR_SOURCE_EQUAL_DEST    :1
    CQNR_MOVE                  :1
    CQNR_COPY                  :1
    CQNR_NO_OPERATION          :1
    CQNR_UNUSED                :11
ClipboardQuickNotifyFlags    end
```

These flags return information about the success or failure of a quick transfer operation.

Library: **Objects/clipbrd.def**

■ ClipboardQuickTransferFeedback

```
ClipboardQuickTransferFeedback    etype word
    CQTF_SET_DEFAULT            enum    ClipboardQuickTransferFeedback
    CQTF_CLEAR_DEFAULT          enum    ClipboardQuickTransferFeedback
    CQTF_MOVE                   enum    ClipboardQuickTransferFeedback
    CQTF_COPY                   enum    ClipboardQuickTransferFeedback
    CQTF_CLEAR                  enum    ClipboardQuickTransferFeedback
```

CQTF_SET_DEFAULT
Sets the default modal cursor used during a clipboard quick-transfer operation. (This is used internally.)



CQTF_CLEAR_DEFAULT
Clears the default modal cursor during a clipboard quick-transfer operation.
(This is used internally.)

CQTF_MOVE
Sets the move cursor during a clipboard quick-transfer operation.

CQTF_COPY
Sets the copy cursor during a clipboard quick-transfer operation.

CQTF_CLEAR
Clears any move/copy cursors present.

Library: **Objects/clipbrd.def**

■ ClipboardQuickTransferFlags

```
ClipboardQuickTransferFlags    record
    CQTF_IN_PROGRESS           :1      ; internal
    CQTF_COPY_ONLY             :1      ; if the source only supports copying
    CQTF_USE_REGION            :1      ; use region
    CQTF_NOTIFICATION          :1      ; set if the quick-transfer source wants
                                      ; to be notified when the transfer item
                                      ; has been processed.
                                :12
ClipboardQuickTransferFlags    end
```

Library: **Objects/clipbrd.def**

■ ClipboardQuickTransferRegionInfo

```
ClipboardQuickTransferRegionInfo    struct
    CQTRI_paramAX              word
    CQTRI_paramBX              word
    CQTRI_paramCX              word
    CQTRI_paramDX              word
    CQTRI_regionPos            Point
    CQTRI_strategy             dword
    CQTRI_region               dword
ClipboardQuickTransferRegionInfo    ends
```

This structure stores the stack parameters used in **ClipboardStartQuickTransfer** if the **ClipboardQuickTransferFlags** in use include **CQTF_USE_REGION**.

Library: **Objects/clipbrd.def**

■ Color

```

Color      etype      byte
C_BLACK    enum Color  ; black color index
C_BLUE     enum Color  ; dark blue color index
C_GREEN    enum Color  ; dark green color index
C_CYAN     enum Color  ; dark cyan color index
C_RED      enum Color  ; dark red color index
C_VIOLET   enum Color  ; dark violet color index
C_BROWN    enum Color  ; brown color index
C_LIGHT_GRAY enum Color ; light gray color index
C_DARK_GRAY enum Color ; dark gray color index
C_LIGHT_BLUE enum Color ; light blue color index
C_LIGHT_GREEN enum Color ; light green color index
C_LIGHT_CYAN enum Color ; light cyan color index
C_LIGHT_RED enum Color  ; light red color index
C_LIGHT_VIOLET enum Color ; light violet color index
C_YELLOW   enum Color  ; yellow color index
C_WHITE    enum Color  ; white color index

C_LIGHT_GREY = C_LIGHT_GRAY ; alternate spelling
C_DARK_GREY  = C_DARK_GRAY  ; alternate spelling

MAX_CF_INDEX = C_WHITE

C_BW_GREY    = 0x84          ; "color" to pass to black
                                ; and white driver to get
                                ; 50% pattern (in dither mode)

;      Additional color enums for use as color indices

C_GRAY_0     enum Color, 0x10; start of grey ramp, 0.0%
C_GRAY_7     enum Color, 0x11; start of grey ramp, 6.3%
C_GRAY_13    enum Color, 0x12; start of grey ramp, 13.3%
C_GRAY_20    enum Color, 0x13; start of grey ramp, 20.0%
C_GRAY_27    enum Color, 0x14; start of grey ramp, 26.7%
C_GRAY_33    enum Color, 0x15; start of grey ramp, 33.3%
C_GRAY_40    enum Color, 0x16; start of grey ramp, 40.0%
C_GRAY_47    enum Color, 0x17; start of grey ramp, 46.7%
C_GRAY_53    enum Color, 0x18; start of grey ramp, 53.3%
C_GRAY_60    enum Color, 0x19; start of grey ramp, 60.0%
C_GRAY_68    enum Color, 0x1a; start of grey ramp, 67.7%
C_GRAY_73    enum Color, 0x1b; start of grey ramp, 73.3%
C_GRAY_80    enum Color, 0x1c; start of grey ramp, 80.0%
C_GRAY_88    enum Color, 0x1d; start of grey ramp, 87.7%
C_GRAY_93    enum Color, 0x1e; start of grey ramp, 93.3%
C_GRAY_100   enum Color, 0x1f; start of grey ramp, 100.0%

C_UNUSED_0   enum Color, 0x20; 8 unused spots
C_UNUSED_1   enum Color, 0x21

```



```

C_UNUSED_2      enum Color, 0x22
C_UNUSED_3      enum Color, 0x23
C_UNUSED_4      enum Color, 0x24
C_UNUSED_5      enum Color, 0x25
C_UNUSED_6      enum Color, 0x26
C_UNUSED_7      enum Color, 0x27
C_R0_G0_B0      enum Color, 0x28; start of 6x6x6 RGB cube
C_R0_G0_B1      enum Color, 0x29
C_R0_G0_B2      enum Color, 0x2a
C_R0_G0_B3      enum Color, 0x2b
C_R0_G0_B4      enum Color, 0x2c
C_R0_G0_B5      enum Color, 0x2d
C_R0_G1_B0      enum Color, 0x2e
C_R0_G1_B1      enum Color, 0x2f
C_R0_G1_B2      enum Color, 0x30
C_R0_G1_B3      enum Color, 0x31
C_R0_G1_B4      enum Color, 0x32
C_R0_G1_B5      enum Color, 0x33
C_R0_G2_B0      enum Color, 0x34
C_R0_G2_B1      enum Color, 0x35
C_R0_G2_B2      enum Color, 0x36
C_R0_G2_B3      enum Color, 0x37
C_R0_G2_B4      enum Color, 0x38
C_R0_G2_B5      enum Color, 0x39
C_R0_G3_B0      enum Color, 0x3a
C_R0_G3_B1      enum Color, 0x3b
C_R0_G3_B2      enum Color, 0x3c
C_R0_G3_B3      enum Color, 0x3d
C_R0_G3_B4      enum Color, 0x3e
C_R0_G3_B5      enum Color, 0x3f
C_R0_G4_B0      enum Color, 0x40
C_R0_G4_B1      enum Color, 0x41
C_R0_G4_B2      enum Color, 0x42
C_R0_G4_B3      enum Color, 0x43
C_R0_G4_B4      enum Color, 0x44
C_R0_G4_B5      enum Color, 0x45
C_R0_G5_B0      enum Color, 0x46
C_R0_G5_B1      enum Color, 0x47
C_R0_G5_B2      enum Color, 0x48
C_R0_G5_B3      enum Color, 0x49
C_R0_G5_B4      enum Color, 0x4a
C_R0_G5_B5      enum Color, 0x4b
C_R1_G0_B0      enum Color, 0x4c
C_R1_G0_B1      enum Color, 0x4d
C_R1_G0_B2      enum Color, 0x4e
C_R1_G0_B3      enum Color, 0x4f
C_R1_G0_B4      enum Color, 0x50
C_R1_G0_B5      enum Color, 0x51
C_R1_G1_B0      enum Color, 0x52
C_R1_G1_B1      enum Color, 0x53

```

C_R1_G1_B2	enum Color, 0x54
C_R1_G1_B3	enum Color, 0x55
C_R1_G1_B4	enum Color, 0x56
C_R1_G1_B5	enum Color, 0x57
C_R1_G2_B0	enum Color, 0x58
C_R1_G2_B1	enum Color, 0x59
C_R1_G2_B2	enum Color, 0x5a
C_R1_G2_B3	enum Color, 0x5b
C_R1_G2_B4	enum Color, 0x5c
C_R1_G2_B5	enum Color, 0x5d
C_R1_G3_B0	enum Color, 0x5e
C_R1_G3_B1	enum Color, 0x5f
C_R1_G3_B2	enum Color, 0x60
C_R1_G3_B3	enum Color, 0x61
C_R1_G3_B4	enum Color, 0x62
C_R1_G3_B5	enum Color, 0x63
C_R1_G4_B0	enum Color, 0x64
C_R1_G4_B1	enum Color, 0x65
C_R1_G4_B2	enum Color, 0x66
C_R1_G4_B3	enum Color, 0x67
C_R1_G4_B4	enum Color, 0x68
C_R1_G4_B5	enum Color, 0x69
C_R1_G5_B0	enum Color, 0x6a
C_R1_G5_B1	enum Color, 0x6b
C_R1_G5_B2	enum Color, 0x6c
C_R1_G5_B3	enum Color, 0x6d
C_R1_G5_B4	enum Color, 0x6e
C_R1_G5_B5	enum Color, 0x6f
C_R2_G0_B0	enum Color, 0x70
C_R2_G0_B1	enum Color, 0x71
C_R2_G0_B2	enum Color, 0x72
C_R2_G0_B3	enum Color, 0x73
C_R2_G0_B4	enum Color, 0x74
C_R2_G0_B5	enum Color, 0x75
C_R2_G1_B0	enum Color, 0x76
C_R2_G1_B1	enum Color, 0x77
C_R2_G1_B2	enum Color, 0x78
C_R2_G1_B3	enum Color, 0x79
C_R2_G1_B4	enum Color, 0x7a
C_R2_G1_B5	enum Color, 0x7b
C_R2_G2_B0	enum Color, 0x7c
C_R2_G2_B1	enum Color, 0x7d
C_R2_G2_B2	enum Color, 0x7e
C_R2_G2_B3	enum Color, 0x7f
C_R2_G2_B4	enum Color, 0x80
C_R2_G2_B5	enum Color, 0x81
C_R2_G3_B0	enum Color, 0x82
C_R2_G3_B1	enum Color, 0x83
C_R2_G3_B2	enum Color, 0x84
C_R2_G3_B3	enum Color, 0x85

C_R2_G3_B4	enum Color, 0x86
C_R2_G3_B5	enum Color, 0x87
C_R2_G4_B0	enum Color, 0x88
C_R2_G4_B1	enum Color, 0x89
C_R2_G4_B2	enum Color, 0x8a
C_R2_G4_B3	enum Color, 0x8b
C_R2_G4_B4	enum Color, 0x8c
C_R2_G4_B5	enum Color, 0x8d
C_R2_G5_B0	enum Color, 0x8e
C_R2_G5_B1	enum Color, 0x8f
C_R2_G5_B2	enum Color, 0x90
C_R2_G5_B3	enum Color, 0x91
C_R2_G5_B4	enum Color, 0x92
C_R2_G5_B5	enum Color, 0x93
C_R3_G0_B0	enum Color, 0x94
C_R3_G0_B1	enum Color, 0x95
C_R3_G0_B2	enum Color, 0x96
C_R3_G0_B3	enum Color, 0x97
C_R3_G0_B4	enum Color, 0x98
C_R3_G0_B5	enum Color, 0x99
C_R3_G1_B0	enum Color, 0x9a
C_R3_G1_B1	enum Color, 0x9b
C_R3_G1_B2	enum Color, 0x9c
C_R3_G1_B3	enum Color, 0x9d
C_R3_G1_B4	enum Color, 0x9e
C_R3_G1_B5	enum Color, 0x9f
C_R3_G2_B0	enum Color, 0xa0
C_R3_G2_B1	enum Color, 0xa1
C_R3_G2_B2	enum Color, 0xa2
C_R3_G2_B3	enum Color, 0xa3
C_R3_G2_B4	enum Color, 0xa4
C_R3_G2_B5	enum Color, 0xa5
C_R3_G3_B0	enum Color, 0xa6
C_R3_G3_B1	enum Color, 0xa7
C_R3_G3_B2	enum Color, 0xa8
C_R3_G3_B3	enum Color, 0xa9
C_R3_G3_B4	enum Color, 0xaa
C_R3_G3_B5	enum Color, 0xab
C_R3_G4_B0	enum Color, 0xac
C_R3_G4_B1	enum Color, 0xad
C_R3_G4_B2	enum Color, 0xae
C_R3_G4_B3	enum Color, 0xaf
C_R3_G4_B4	enum Color, 0xb0
C_R3_G4_B5	enum Color, 0xb1
C_R3_G5_B0	enum Color, 0xb2
C_R3_G5_B1	enum Color, 0xb3
C_R3_G5_B2	enum Color, 0xb4
C_R3_G5_B3	enum Color, 0xb5
C_R3_G5_B4	enum Color, 0xb6
C_R3_G5_B5	enum Color, 0xb7

C_R4_G0_B0	enum Color, 0xb8
C_R4_G0_B1	enum Color, 0xb9
C_R4_G0_B2	enum Color, 0xba
C_R4_G0_B3	enum Color, 0xbb
C_R4_G0_B4	enum Color, 0xbc
C_R4_G0_B5	enum Color, 0xbd
C_R4_G1_B0	enum Color, 0xbe
C_R4_G1_B1	enum Color, 0xbf
C_R4_G1_B2	enum Color, 0xc0
C_R4_G1_B3	enum Color, 0xc1
C_R4_G1_B4	enum Color, 0xc2
C_R4_G1_B5	enum Color, 0xc3
C_R4_G2_B0	enum Color, 0xc4
C_R4_G2_B1	enum Color, 0xc5
C_R4_G2_B2	enum Color, 0xc6
C_R4_G2_B3	enum Color, 0xc7
C_R4_G2_B4	enum Color, 0xc8
C_R4_G2_B5	enum Color, 0xc9
C_R4_G3_B0	enum Color, 0xca
C_R4_G3_B1	enum Color, 0xcb
C_R4_G3_B2	enum Color, 0xcc
C_R4_G3_B3	enum Color, 0xcd
C_R4_G3_B4	enum Color, 0xce
C_R4_G3_B5	enum Color, 0xcf
C_R4_G4_B0	enum Color, 0xd0
C_R4_G4_B1	enum Color, 0xd1
C_R4_G4_B2	enum Color, 0xd2
C_R4_G4_B3	enum Color, 0xd3
C_R4_G4_B4	enum Color, 0xd4
C_R4_G4_B5	enum Color, 0xd5
C_R4_G5_B0	enum Color, 0xd6
C_R4_G5_B1	enum Color, 0xd7
C_R4_G5_B2	enum Color, 0xd8
C_R4_G5_B3	enum Color, 0xd9
C_R4_G5_B4	enum Color, 0xda
C_R4_G5_B5	enum Color, 0xdb
C_R5_G0_B0	enum Color, 0xdc
C_R5_G0_B1	enum Color, 0xdd
C_R5_G0_B2	enum Color, 0xde
C_R5_G0_B3	enum Color, 0xdf
C_R5_G0_B4	enum Color, 0xe0
C_R5_G0_B5	enum Color, 0xe1
C_R5_G1_B0	enum Color, 0xe2
C_R5_G1_B1	enum Color, 0xe3
C_R5_G1_B2	enum Color, 0xe4
C_R5_G1_B3	enum Color, 0xe5
C_R5_G1_B4	enum Color, 0xe6
C_R5_G1_B5	enum Color, 0xe7
C_R5_G2_B0	enum Color, 0xe8
C_R5_G2_B1	enum Color, 0xe9



C_R5_G2_B2	enum Color, 0xea
C_R5_G2_B3	enum Color, 0xeb
C_R5_G2_B4	enum Color, 0xec
C_R5_G2_B5	enum Color, 0xed
C_R5_G3_B0	enum Color, 0xee
C_R5_G3_B1	enum Color, 0xef
C_R5_G3_B2	enum Color, 0xf0
C_R5_G3_B3	enum Color, 0xf1
C_R5_G3_B4	enum Color, 0xf2
C_R5_G3_B5	enum Color, 0xf3
C_R5_G4_B0	enum Color, 0xf4
C_R5_G4_B1	enum Color, 0xf5
C_R5_G4_B2	enum Color, 0xf6
C_R5_G4_B3	enum Color, 0xf7
C_R5_G4_B4	enum Color, 0xf8
C_R5_G4_B5	enum Color, 0xf9
C_R5_G5_B0	enum Color, 0xfa
C_R5_G5_B1	enum Color, 0xfb
C_R5_G5_B2	enum Color, 0xfc
C_R5_G5_B3	enum Color, 0xfd
C_R5_G5_B4	enum Color, 0xfe

Library: **color.def**

■ ColoredObjectOrientation

ColoredObjectOrientation	etype	byte
COO_AREA_ORIENTED	enum	ColoredObjectOrientation
COO_TEXT_ORIENTED	enum	ColoredObjectOrientation
COO_LINE_ORIENTED	enum	ColoredObjectOrientation

Library: **Objects/colorC.def**

■ ColorFlag

ColorFlag	etype	byte
CF_INDEX	enum	ColorFlag ; set color with index
CF_GRAY	enum	ColorFlag ; set color with gray value
CF_SAME	enum	ColorFlag ; don't change the color (hatch)
CF_CMY	enum	ColorFlag ; set color with CMY value
CF_RGB	enum	ColorFlag, 0x80 ; set color with RGB values

Several color-related commands accept colors in a variety of formats. The **ColorFlag** type is used to specify how the color is being described. The **ColorFlag** is normally used as part of a **ColorQuad**. See **ColorQuad** for information about how to interpret color specifications using **ColorFlags**.

Library: **color.def**

■ ColorMapMode

```
ColorMapMode      record
  CMM_ON_BLACK    :1          ; 1 if drawing on black
                  :1
  CMM_MAP_TYPE    ColorMapType:1 ; color mapping mode.
ColorMapMode      end
```

Library: **graphics.def**

■ ColorMapType

```
ColorMapType      etype      byte
  CMT_CLOSEST     enum      ColorMapType; Map to closest solid color
  CMT_DITHER      enum      ColorMapType; Map to dither pattern
```

Library: **graphics.def**

■ ColorModifiedStates

```
ColorModifiedStates record
  CMS_COLOR_CHANGED      :1
  CMS_DRAW_MASK_CHANGED  :1
  CMS_PATTERN_CHANGED    :1
ColorModifiedStates      end
```

Library: **colorC.def**

■ ColorQuad

```
ColorQuad      struct
  CQ_redOrIndex byte
  CQ_info        ColorFlag
  CQ_green       byte
  CQ_blue        byte
ColorQuad      ends
```

This structure represents a color.

CQ_info determines how the color is being described.

If *CQ_info* includes the CF_INDEX flag, the color is specified by an index value which matches a specific color in the palette. This index is stored in the *CQ_redOrIndex* field; *CQ_green* and *CQ_blue* are ignored if the color is an index value.



If *CQ_info* includes the CF_RGB flag, the color is specified by separate RGB components. *CQ_redOrIndex* stores the red value and *CQ_green* and *CQ_blue* store the green and blue components, respectively.

If *CQ_info* contains the CF_GRAY flag, the color is being expressed as a grey scale. This is basically an optimized way of describing RGB colors where the red, green, and blue components are equal. The *CQ_redOrIndex* field contains the brightness, a number between 0 and 255. The *CQ_green* and *CQ_blue* fields are ignored.

When defining hatch patterns, it is possible to have a CF_SAME info field. This means that the hatch lines should used the “same” color when drawing. That is, when hatching text, the text color will be used; when filling an area, the area color will be used. The *CQ_redOrIndex*, *CQ_green*, and *CQ_blue* fields are all ignored.

Library: **color.def**

■ ColorScheme

```
ColorScheme      record
  CS_lightColor   Color:4
  CS_darkColor    Color:4
ColorScheme      end
```

Library: **Objects/visC.def**

■ ColorToolboxPreferences

```
ColorToolboxPreferences  record
                                :2
  CTP_INDEX_ORIENTATION   :2      ;ColoredObjectOrientation
  CTP_DRAW_MASK_ORIENTATION :2
  CTP_PATTERN_ORIENTATION :1
  CTP_IS_POPUP             :1
ColorToolboxPreferences  end
```

Library: **colorC.def**

■ ColorTransfer

```
ColorTransfer      struct
    CT_data        RGBDelta 5*5*5 dup (?)    ; 375 bytes of data.
ColorTransfer      ends
```

A color correction table is a 5x5x5 cube of RGB difference values. The correction is done by doing a lookup in the 3D table and applying the **RGBDelta** values to the original input values.

Library: **color.def**

■ ColumnArrayElement

```
ColumnArrayElement struct
    CAE_column      byte          ; The column number in which the cell resides.
    CAE_data        DBaseItem     ; The item containing the cell data.
ColumnArrayElement ends
```

Library: **cell.def**

■ ColumnArrayHeader

```
ColumnArrayHeader struct
    CAH_numEntries  word          ; Number of entries in the array.
    CAH_rowFlags    word          ; Flags that exist for each row.
ColumnArrayHeader ends
```

Library: **cell.def**

■ CommonParameters

```
CommonParameters  struct
    CP_row          word          ; Current row
    CP_column       word          ; Current column
    CP_maxRow       word          ; Largest legal row value
    CP_maxColumn    word          ; Largest legal column value
    CP_callback     dword         ; One general purpose callback
    CP_cellParams   dword         ; Pointer to the cell parameters
CommonParameters  ends
```

This structure stores basic information that is useful to many of the parse callback routines. It should always be placed at the base of the parameter structures.

Library: **parse.def**



■ CommonTransferParams

```
CommonTransferParams    struct
    CTP_range            VisTextRange
    CTP_pasteFrame       word            ;ptr to frame if quick paste.
                                   ;0 otherwise.
    CTP_vmFile           word            ;VM file handle
    CTP_vmBlock          word            ;VM block handle
CommonTransferParams    ends
```

This structure stores parameters sent on the stack to all transfer routines.

Library: **Objects/vTextC.def**

■ CompChildFlags

```
CompChildFlags          record
    CCF_MARK_DIRTY       :1,            ; Marks chunk and modified objects as
                                   ; dirty
    CCF_REFERENCE        :15           ; Object # we should add new object
                                   ; before (if > # objects, then add new
                                   ; object last)

    CCO_FIRST            equ           0x0000
    CCO_LAST             equ           0x7FFF ;NOTE - will not work if the object
                                   ;already has 32767 children.
CompChildFlags          end
```

This record is used when adding, moving, or removing children in an object tree.

CCF_MARK_DIRTY indicates whether the object should be marked dirty at the end of the operation.

CCF_REFERENCE stores a child number; when adding or moving a child, this is the child number after which the new object should be inserted. It can be any number less than 32768 or either of the two constants CCO_FIRST and CCO_LAST specifying the absolute first or last position.

Library: **Objects/metaC.def**

■ CompPart

```
CompPart                struct
    CP_firstChild        optr            ; 0 = no children.
CompPart                ends
```

Library: **Objects/metaC.def**

■ CompSizeHintArgs

```
CompSizeHintArgs    struct
    CSHA_width      SpecWidth <>      ; Width of the composite.
    CSHA_height     SpecHeight <>     ; Height of each child.
    CSHA_count      sword             ; Number of children of a composite.
CompSizeHintArgs    ends
```

This structure is used for HINT_FIXED_SIZE, HINT_MINIMUM_SIZE, HINT_MAXIMUM_SIZE and HINT_INITIAL_SIZE.

Library: **Objects/genC.def**

■ ContextData

```
ContextData         struct
    CD_object        optr
    CD_numChars      dword
    CD_range         VisTextRange
    CD_selection     VisTextRange
    CD_contextData   label char
ContextData         ends
```

CD_object stores the optr of the object the context is coming from.

CD_numChars stores the number of chars in the text object.

CD_range stores the range of characters that this context represents.

CD_selection stores the current text selection.

CD_contextData stores the null-terminated data.

Library: **Objects/vTextC.def**

■ ContextLocation

```
ContextLocation     etype      word
    CL_STARTING_AT_POSITION      enum      ContextLocation
    CL_ENDING_AT_POSITION        enum      ContextLocation
    CL_CENTERED_AROUND_POSITION  enum      ContextLocation
    CL_CENTERED_AROUND_SELECTION enum      ContextLocation
    CL_CENTERED_AROUND_SELECTION_START enum ContextLocation
    CL_SELECTED_WORD             enum      ContextLocation
```

This type is used to identify a context location within a **GetContextParams** structure.

CL_STARTING_AT_POSITION

Retrieves *GCP_numCharsToGet* characters starting at *GCP_position*.

CL_ENDING_AT_POSITION

Retrieves text ending at the passed selection.



CL_CENTERED_AROUND_POSITION
Retrieves *GCP_numCharsToGet* characters centered around *GCP_position*.

CL_CENTERED_AROUND_SELECTION
Retrieves *GCP_numCharsToGet* characters centered around the selection

CL_CENTERED_AROUND_SELECTION_START
Retrieves *GCP_numCharsToGet* characters centered around the start of the selection

CL_SELECTED_WORD
Retrieves the selection or surrounding word.

Library: **Objects/vTextC.def**

■ ContextValues

ContextValues etype word, 0

Library: **ec.def**

■ CopyChunkFlags

CopyChunkFlags record

 CCF_DIRTY :1

 CCF_MODE CopyChunkMode:2

 CCF_SIZE :13 ; number of bytes to copy (Not used for

 ; CCM_OPTR).

CopyChunkFlags end

 CCF_DIRTY

 If set, any created chunk is set DIRTY. If clear, any created chunk is set

 IGNORE_DIRTY

Library: **Objects/processC.def**

■ CopyChunkInFrame

CopyChunkInFrame struct

 CCIF_copyFlags CopyChunkFlags

 CCIF_source dword

 CCIF_destBlock hpPtr

CopyChunkInFrame ends

This structure is passed on the stack to MSG_PROCESS_COPY_CHUNK_IN.

CCIF_destBlock must be in an object block.

Library: **Objects/process.def**

■ CopyChunkMode

CopyChunkMode	etype	byte
CCM_OPTR	enum	CopyChunkMode
CCM_HPTR	enum	CopyChunkMode
CCM_FPTR	enum	CopyChunkMode
CCM_STRING	enum	CopyChunkMode

CCM_OPTR

The chunk being copied is in the form of an object block and chunk offset.

CCM_HPTR

The chunk being copied is in the form of a memory block and chunk offset.

CCM_FPTR

The chunk being copied is in the form of a segment and chunk offset.

Library: **Objects/processC.def**

■ CopyChunkOutFrame

CopyChunkOutFrame	struct
CCOF_copyFlags	CopyChunkFlags
CCOF_source	optr
CCOF_dest	dword
CopyChunkOutFrame	ends

This structure is passed on the stack to MSG_PROCESS_COPY_CHUNK_OUT.

Library: **Objects/processC.def**

■ CopyChunkOverFrame

CopyChunkOverFrame	struct
CCOVF_copyFlags	CopyChunkFlags
CCOVF_source	dword
CCOVF_dest	optr ; If 0, then creates a new chunk.
CopyChunkOverFrame	ends

This structure is passed on the stack to MSG_PROCESS_COPY_CHUNK_OVER.

Library: **Objects/processC.def**

■ CountryType

CountryType	etype	word, 1, 1
CT_UNITED_STATE	enum	CountryType
CT_CANADA	enum	CountryType
CT_UNITED_KINGDOM	enum	CountryType
CT_GERMANY	enum	CountryType
CT_FRANCE	enum	CountryType



CT_SPAIN	enum	CountryType
CT_ITALY	enum	CountryType
CT_DENMARK	enum	CountryType
CT_NETHERLANDS	enum	CountryType

Library: **localize.def**

■ CParserReturnStruct

```
CParserReturnStruct struc
    PRS_errorCode      byte
    PRS_textOffsetStart word
    PRS_textOffsetEnd  word
    PRS_lastTokenPtr   fptr
CParserReturnStruct ends
```

Library: **parse.def**

■ CParserStruct

```
CParserStruct      struc
    C_parameters    ParserParameters
    C_callbackPtr   fptr
    C_callbackStruct C_CallbackStruct
CParserStruct      ends
```

Library: **parse.def**

■ CPUFlags

```
CPUFlags          record
    :4
    CPU_OVERFLOW   :1
    CPU_DIRECTION  :1
    CPU_INTERRUPT  :1
    CPU_TRAP       :1
    CPU_SIGN       :1
    CPU_ZERO       :1
    :1
    CPU_AUX_CARRY  :1
    :1
    CPU_PARITY     :1
    :1
    CPU_CARRY      :1
CPUFlags          end
```

Library: **geos.def**

■ CRangeEnumCallbackParams

```
CRangeEnumCallbackParams struct
    CRECP_rangeParams      fptr.CRangeEnumParams
    CRECP_row              word                ;current row
    CRECP_column           word                ;current column
    CRECP_cellData         fptr                ;NULL if no data or REF_NO_LOCK
                                         ;passed
    CRECP_rangeFlags       RangeEnumFlags     ;Range flags.
CRangeEnumCallbackParams ends
```

This structure is a C version of what the **RangeEnum** callback function is called with.

Library: **cell.def**

■ CRangeEnumParams

```
CRangeEnumParams struct
    CREP_params            RangeEnumParams
    CREP_locals            fptr
    CREP_callback          fptr.far ; This field is used internally.
CRangeEnumParams ends
```

This structure is a C version of **RangeEnumParams**.

Library: **cell.def**

■ CreateExpressMenuControlItemFeature

```
CreateExpressMenuControlItemFeature etype word
CEMCIF_GEOS_TASKS_LIST    enum    CreateExpressMenuControlItemFeature
CEMCIF_DOS_TASKS_LIST     enum    CreateExpressMenuControlItemFeature
CEMCIF_CONTROL_PANEL      enum    CreateExpressMenuControlItemFeature
CEMCIF_UTILITIES_PANEL    enum    CreateExpressMenuControlItemFeature
```

Library: **Objects/eMenuC.def**



■ CreateExpressMenuItemParams

```
CreateExpressMenuItemParams struct
    CEMCIP_feature          CreateExpressMenuItemFeature
    CEMCIP_class            fptr.ClassStruct
    CEMCIP_itemPriority      CreateExpressMenuItemPriority
    CEMCIP_responseMessage  word
    CEMCIP_responseDestinationoptr
    CEMCIP_responseData     word
    CEMCIP_field            optr
CreateExpressMenuItemParams ends
```

CEMCIP_feature stores the feature to which the item is to be created. Only `EMCF_GEOS_TASKS_LIST`, `EMCF_DOS_TASKS_LIST`, `EMCF_CONTROL_PANEL`, and `EMCF_UTILITIES_PANEL` are allowed.

CEMCIP_class stores the class of the object to create. This class must be a subclass of `GenItemClass` for `CEMCIF_GEOS_TASKS_LIST`, a subclass of `GenTriggerClass` for `CEMCIF_DOS_TASKS_LIST`, or a subclass of `GenClass` for `CEMCIF_CONTROL_PANEL` and `CEMCIF_UTILITIES_PANEL`.

CEMCIP_itemPriority specifies the relative position for the newly created item. Lower numbers will be added in front (above) higher numbers. Use `CEMCIP_STANDARD_PRIORITY` for default position

CEMCIP_responseMessage stores the message to send with newly created object's `optr`.

CEMCIP_responseDestination stores the destination for the response message.

CEMCIP_responseData stores an opaque word of data copied to *CEMCIRP_data* field to help destination figure out what it should do with the new item.

CEMCIP_field stores the `optr` of a `GenField`. Only Express Menu Control objects associated with this `GenField` object will be affected. Pass 0 if the `GenField` the Express Menu Control is associated with doesn't matter.

Library: **Objects/eMenuC.def**

■ CreateExpressMenuItemPriority

```
CreateExpressMenuItemPriority etype word
    CEMCIP_SPOOL_CONTROL_PANEL enum CreateExpressMenuItemPriority, 100h
    CEMCIP_NETMSG_SEND_MESSAGE enum CreateExpressMenuItemPriority, 200h
    CEMCIP_SAVER_SCREEN_SAVER  enum CreateExpressMenuItemPriority, 300h
    CEMCIP_SAVER_SCREEN_LOCK   enum CreateExpressMenuItemPriority, 400h
    CEMCIP_STANDARD_PRIORITY    enum CreateExpressMenuItemPriority, \
                                CCO_LAST
```

Library: **Objects/eMenuC.def**

■ CreateExpressMenuItemResponseParams

```
CreateExpressMenuItemResponseParams struct
    CEMCIRP_newItem      optr
    CEMCIRP_data          word
    CEMCIRP_expressMenuControl optr
CreateExpressMenuItemResponseParams ends
```

This structure stores the parameters for the response message sent in **CreateExpressMenuItemParams**.

CEMCIRP_newItem stores the optr of the newly created item.;

CEMCIRP_data stores an opaque word of data copied from *CEMCIP_responseData* field to help the destination figure out what it should do with the new item.

CEMCIRP_expressMenuControl stores the optr of the Express Menu Control object that created the new item.

Library: **Objects/eMenuC.def**

■ CreateVisMonikerFlags

```
CreateVisMonikerFlags      record
    CVMF_DIRTY      :1
                   :7
CreateVisMonikerFlags      end
```

Library: **Objects/visC.def**

■ CreateVisMonikerFrame

```
CreateVisMonikerFrame      struct
    CVMF_source      dword
    CVMF_sourceType   VisMonikerSourceType
    even
    CVMF_dataType     VisMonikerDataType
    even
    CVMF_length       word
```



```

CVMF_width      word
CVMF_height     word
CVMF_flags      CreateVisMonikerFlags
even
CreateVisMonikerFrame  ends

```

This structure contains parameters passed to MSG_VIS_CREATE_VIS_MONIKER and MSG_GEN_CREATE_VIS_MONIKER.

CVMF_source stores the source for the moniker. This source may be an optr, hptr, or fptr, depending on the *CVMF_sourceType*.

CVMF_sourceType stores the **VisMonikerSourceType**, which specifies whether the moniker in *CVMF_source* is an optr, hptr, or fptr.

CVMF_dataType specifies whether the source is a VisMoniker, text string, graphics string, or GeodeToken.

CVMF_length stores the byte size of the source. This size is not used if *CVMF_sourceType* is VMST_OPTR. If the source type is VMDT_TEXT and *CVMF_length* is 0, text is assumed to be null-terminated. If the source type is VMDT_GSTRING and *CVMF_length* is 0, the length of the gstring is computed by scanning the gstring.

CVMF_width stores the width of the source if the source type is VMDT_GSTRING. If 0, the width of gstring is computed by scanning the gstring.

CVMF_height stores the height of the source if the source type is VMDT_GSTRING. If 0, the height of the gstring is computed by scanning the gstring.

CVMF_flags stores flags indicating whether to create the new moniker chunk dirty.

Library: **Objects/visC.def**

■ CSFeatures

```

CSFeatures      record
  CSF_FILLED_LIST  :1
  CSF_INDEX        :1
  CSF_RGB          :1
  CSF_DRAW_MASK    :1
  CSF_PATTERN      :1
CSFeatures      end

```

Library: **Objects/colorC.def**

Reference book

■ CStoolboxFeatures

```
CStoolboxFeatures    record
    CSTF_INDEX       :1
    CSTF_DRAW_MASK   :1
    CSTF_PATTERN      :1
CStoolboxFeatures    end
```

Library: **colorC.def**

■ CT_CC_CallbackStruct

```
CT_CC_CallbackStruct    struc      ; Structure for CT_CREATE_CELL
    CC_row              word
    CC_column           word
    CC_errorOccurred    byte
    CC_error            byte
CT_CC_CallbackStruct    ends
```

Library: **parse.def**

■ CT_CNE_CallbackStruct

```
CT_CNE_CallbackStruct    struc;      ; Structure for CT_CHECK_NAME_EXISTS
    CNE_text            fptr
    CNE_length          word
    CNE_nameExists      byte
CT_CNE_CallbackStruct    ends
```

Library: **parse.def**

■ CT_CNS_CallbackStruct

```
CT_CNS_CallbackStruct    struc      ; Structure for CT_CHECK_NAME_SPACE
    CNS_numToAllocate   word
    CNS_enoughSpace     byte
    CNS_errorOccurred   byte
    CNS_error           byte
CT_CNS_CallbackStruct    ends
```

Library: **parse.def**



■ **CT_DC_CallbackStruct**

```
CT_DC_CallbackStruct      struc      ;Structure for CT_DEREF_CELL
    DC_argStack           fptr
    DC_opFnStack          fptr
    DC_row                word
    DC_column             byte
    DC_derefFlags         DerefFlags
    DC_newArgStack        fptr
    DC_errorOccurred      byte
    DC_error              byte
CT_DC_CallbackStruct      ends
```

Library: **parse.def**

■ **CT_EC_CallbackStruct**

```
CT_EC_CallbackStruct      struc      ; Structure for CT_EMPTY_CELL
    EC_row               word
    EC_column            word
    EC_errorOccurred     byte
    EC_error             byte
CT_EC_CallbackStruct      ends
```

Library: **parse.def**

■ **CT_EF_CallbackStruct**

```
CT_EF_CallbackStruct      struc      ; Structure for CT_EVAL_FUNCTION
    EF_numArgs           word
    EF_funcID            word
    EF_opStack           fptr
    EF_argStack          fptr
    EF_errorOccurred     byte
    EF_error             byte
CT_EF_CallbackStruct      ends
```

Library: **parse.def**

■ CT_FF_CallbackStruct

```

CT_FF_CallbackStruct    struc    ; Structure for CT_FORMAT_FUNCTION
    FF_funcID           word
    FF_maxChars         word
    FF_resultPtr        fptr
    FF_numWritten       word
CT_FF_CallbackStruct    ends
  
```

Library: **parse.def**

■ CT_FN_CallbackStruct

```

CT_FN_CallbackStruct    struc    ; Structure for CT_FORMAT_NAME
    FN_textPtr          fptr
    FN_nameToken        word
    FN_maxChars         word
    FN_resultPtr        fptr
    FN_numWritten       word
CT_FN_CallbackStruct    ends
  
```

Library: **parse.def**

■ CT_FTC_CallbackStruct

```

CT_FTC_CallbackStruct    struc    ; Structure for CT_FUNCTION_TO_CELL
    FTC_funcID          word
    FTC_row             word
    FTC_column          word
    FTC_errorOccurred   byte
    FTC_error           byte
CT_FTC_CallbackStruct    ends
  
```

Library: **parse.def**

■ CT_FTT_CallbackStruct

```

CT_FTT_CallbackStruct    struc    ; Structure for CT_FUNCTION_TO_TOKEN
    FTT_text            fptr
    FTT_length          word
    FTT_isFunctionName   byte
    FTT_funcID          word
CT_FTT_CallbackStruct    ends
  
```

Library: **parse.def**



■ CT_LN_CallbackStruct

```
CT_LN_CallbackStruct    struc    ; Structure for CT_LOCK_NAME
    LN_nameToken        word
    LN_defPtr           dword
    LN_errorOccurred    byte
    LN_error            byte
CT_LN_CallbackStruct    ends
```

Library: **parse.def**

■ CT_NTC_CallbackStruct

```
CT_NTC_CallbackStruct  struc    ; Structure for CT_NAME_TO_CELL
    NTC_nameToken       word
    NTC_row             word
    NTC_column          word
CT_NTC_CallbackStruct  ends
```

Library: **parse.def**

■ CT_NTT_CallbackStruct

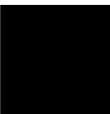
```
CT_NTT_CallbackStruct  struc    ; Structure for CT_NAME_TO_TOKEN
    NTT_text            fptr
    NTT_length          word
    NTT_nameID          word
    NTT_errorOccurred   byte
    NTT_error           byte
CT_NTT_CallbackStruct  ends
```

Library: **parse.def**

■ CT_SF_CallbackStruct

```
CT_SF_CallbackStruct   struc    ; Structure for CT_SPECIAL_FUNCTION
    SF_argStack         fptr
    SF_opFnStack        fptr
    SF_specialFunction   SpecialFunction
    SF_newArgStack       fptr
    SF_errorOccurred     byte
    SF_error            byte
CT_SF_CallbackStruct   ends
```

Library: **parse.def**

 **Reference book**

■ CT_UL_CallbackStruct

```
CT_UL_CallbackStruct    struc    ; Structure for CT_UNLOCK
    UL_dataPtr          fptr
CT_UL_CallbackStruct    ends
```

Library: **parse.def**

■ CurrencyFormatFlags

```
CurrencyFormatFlags    record
                                :2
    CFF_LEADING_ZERO      :1
    CFF_SPACE_AROUND_SYMBOL :1

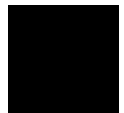
    ; these four are set together in one preference manager gadget.
    CFF_USE_NEGATIVE_SIGN  :1
    CFF_SYMBOL_BEFORE_NUMBER :1
    CFF_NEGATIVE_SIGN_BEFORE_NUMBER :1
    CFF_NEGATIVE_SIGN_BEFORE_SYMBOL :1
CurrencyFormatFlags    end
```

Library: **localize.def**

■ CustomDialogBoxFlags

```
CustomDialogBoxFlags    record
    CDBF_SYSTEM_MODAL      :1
    CDBF_DIALOG_TYPE        CustomDialogType:2
    CDBF_INTERACTION_TYPE    GenInteractionType:4
    CDBF_DESTRUCTIVE_ACTION  :1=0
    ; This flag signals that the affirmative response to this dialog
    ; denotes a destructive action, and shouldn't be given the
    ; interaction default. A HINT_TRIGGER_DESTRUCTIVE_ACTION will be
    ; placed on the trigger having an IC_YES interaction command.
    ; This flag can only be used on a GIT_MULTIPLE_RESPONSE dialog.
                                :8
CustomDialogBoxFlags    end
```

Library: **uDialog.def**



■ CustomDialogType

CustomDialogType	etype	byte
CDT_QUESTION	enum	CustomDialogType
CDT_WARNING	enum	CustomDialogType
CDT_NOTIFICATION	enum	CustomDialogType
CDT_ERROR	enum	CustomDialogType

This type specifies the type of dialog box brought up by **UserStandardDialog**. These types are used in determining any special graphics strings that the dialog box may display.

CDT_QUESTION

This type specifies that the dialog asks the user a question (e.g. "Save changes to 'ftpoom' before quitting?"). The associated text should normally end in a question mark.

CDT_WARNING

This type specifies that the dialog warns the user of an impending action. (e.g. "This action can cause loss of data.").

CDT_NOTIFICATION

This type specifies that the dialog performs a generic notification to the user. (e.g. "New mail has arrived.").

CDT_ERROR

This type specifies that the dialog states an error condition (e.g. "cannot open file"). Typically, error dialog boxes beep when the dialog is displayed.

Library: **uDialog.def**

3Assembly Routines

■ DACPlayFlags

```
DACPlayFlags      record
    DACPF_CATENATE :1
                  :7
DACPlayFlags      end
```

Library: **sound.def**

■ DACReferenceByte

```
DACReferenceByte  etype      word, 0, 1
    DACRB_NO_REFERENCE_BYTE  enum  DACReferenceByte
    DACRB_WITH_REFERENCE_BYTE enum  DACReferenceByte
```

Library: **sound.def**

■ DACSampleFormat

```
DACSampleFormat  etype      word, 0, 1
    DACSF_8_BIT_PCM      enum  DACSampleFormat
    DACSF_2_TO_1_ADPCM   enum  DACSampleFormat
    DACSF_3_TO_1_ADPCM   enum  DACSampleFormat
    DACSF_4_TO_1_ADPCM   enum  DACSampleFormat
```

Library: **sound.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ DateTimeFormat

DateTimeFormat	etype	word
DTF_LONG	enum	DateTimeFormat ; Sunday, March 5th, 1990
DTF_LONG_CONDENSED	enum	DateTimeFormat ; Sun, Mar 5, 1990
DTF_LONG_NO_WEEKDAY	enum	DateTimeFormat ; March 5th, 1990
DTF_LONG_NO_WEEKDAY_CONDENSED	enum	DateTimeFormat ; Mar 5, 1990
DTF_SHORT	enum	DateTimeFormat ; 3/5/90
DTF_ZERO_PADDED_SHORT	enum	DateTimeFormat ; 03/05/90
DTF_MD_LONG	enum	DateTimeFormat ; Sunday, March 5th
DTF_MD_LONG_NO_WEEKDAY	enum	DateTimeFormat ; March 5th
DTF_MD_SHORT	enum	DateTimeFormat ; 3/5
DTF_MY_LONG	enum	DateTimeFormat ; March 1990
DTF_MY_SHORT	enum	DateTimeFormat ; 3/90
DTF_YEAR	enum	DateTimeFormat ; 1990
DTF_MONTH	enum	DateTimeFormat ; March
DTF_DAY	enum	DateTimeFormat ; 5th
DTF_WEEKDAY	enum	DateTimeFormat ; Wednesday
;		
DTF_END_DATE_FORMATS	=	DateTimeFormat
DTF_START_TIME_FORMATS	=	DateTimeFormat
DTF_HMS	enum	DateTimeFormat ; 1:05:31 PM
DTF_HM	enum	DateTimeFormat ; 1:05 PM
DTF_H	enum	DateTimeFormat ; 1 PM
DTF_MS	enum	DateTimeFormat ; 5:31
DTF_HMS_24HOUR	enum	DateTimeFormat ; 13:05:31
DTF_HM_24HOUR	enum	DateTimeFormat ; 13:05
DTF_END_TIME_FORMATS	=	DateTimeFormat

These date-time formats are generic. The examples shown to the right are for US based date-times only. In other countries, date-time formats may be substantially different.

Library: **localize.def**

■ DBaseItem

DBaseItem	struct
DBI_group	word ; The group in which the data resides.
DBI_item	word ; The item within that group.
DBaseItem	ends

This structure defines a dbase item.

Library: **cell.def**

■ **DBGGroupAndItem**

```
DBGGroupAndItem    struct
    DBGI_item      word
    DBGI_group     word
DBGGroupAndItem    ends
```

Library: **dbase.def**

■ **dbptr**

```
dbptr    struct
    dbitem word
    dbggroup word
dbptr    ends
```

Library: **config.def**

■ **DCCFeatures**

```
DCCFeatures    record
    DCCF_DROP_CAP    :1
DCCFeatures    end
```

These feature flags are used with ATTR_GEN_CONTROL_REQUIRE_UI and ATTR_GEN_CONTROL_PROHIBIT_UI.

Library: **Objects/Text/tCtrlC.def**

■ **DCCToolboxFeatures**

```
DCCToolboxFeatures record
DCCToolboxFeatures end
```

Library: **Objects/Text/tCtrlC.def**

■ **DDFixed**

```
DDFixed struct
    DDF_frac    dword
    DDF_int     sdword
DDFixed ends
```

This structure defines a 32 bit/32 bit fixed point number.

Library: **geos.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ DebugObjDuplicateResourceInfo

```
DebugObjDuplicateResourceInfo  struct
    DODRI_tempOwner char    GEODE_NAME_SIZE dup(?)
    ; The permanent name of the geode owning the template that this block
    ; was duplicated from.
    DODRI_tempResource word
    ; The handle of the template resource, unrelocated relative to the above.
DebugObjDuplicateResourceInfo  ends
```

Info used to allow the debugger to print symbolic information for objects in the block this object lies in. The EC version of **ObjDuplicateResource**, after duplicating an object block, finds the first object & adds this piece of vardata to it, thereby tagging the block for the debugger with info regarding where it came from. The data stored is the unrelocated library, and unrelocated resource within that library, so that the entry is valid across sessions. The debugger, when displaying info about an object in a block that doesn't have a symbol, uses this to be able to show a reference like:

^h4020h:EditCopyTrigger

Library: **metaC.def**

■ DefaultFieldNameUsage

```
DefaultFieldNameUsage  etype  byte,    0
    DFNU_FIELD          enum    DefaultFieldNameUsage
    DFNU_COLUMN         enum    DefaultFieldNameUsage
    DFNU_FIXED          enum    DefaultFieldNameUsage
```

Library: **impex.def**

■ Dependency

```
Dependency  struct
    D_row      word    ; Row of the dependency.
    D_column   byte    ; Column of the dependency.
Dependency  ends
```

Library: **parse.def**

■ DependencyBlock

```
DependencyBlock  struct
    DB_size  word    ; Size of the block containing the dependency.
DependencyBlock  ends
```

Library: **parse.def**

■ DependencyListHeader

```

DependencyListHeader      struct
    DLH_next      dword      ; DBase item containing next block in the list.
DependencyListHeader      ends
  
```

DLH_next. The “next” link must come first in this structure. It must be at the same position as the cells dependency list header (which must also fall at the start of the cell data).

Library: **parse.def**

■ DependencyParameters

```

DependencyParameters      struct
    DP_common          CommonParameters <>
    ;
    ; Possible callbacks:
    ;     CT_CREATE_CELL, CT_EMPTY_CELL, CT_NAME_TO_CELL,
    ;     CT_FUNCTION_TO_TOKEN
    ;
    ; Everything else here is used exclusively by the dependency list code.
    ; Applications do not need to initialize it and should not depend on
    ; the values returned in this part of the stack frame.
    ;
    DP_dep              dword      ; Dbase item containing the dependency list.
    DP_prev              dword      ; Dbase item containing the previous block.
    DP_prevIsCell        byte      ; If non-zero, previous entry is the cell.
    DP_chunk              word      ; Chunk handle of the current dependency.
    align                word
DependencyParameters      ends
  
```

This structure is passed to the dependency code.

Library: **parse.def**

■ Derefflags

```

Derefflags                record
    DF_DONT_POP_ARGUMENT    :1      ; Set = don't pop arg from arg stack.
Derefflags                end
  
```

Library: **parse.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **DestinationClassArgs**

```
DestinationClassArgs    struct
    DCA_class    fptr.ClassStruct
DestinationClassArgs    ends
```

Library: **Objects/genC.def**

■ **DetachDataEntry**

```
DetachDataEntry    struct
    DDE_ackCount    word
    DDE_callerID    word
    DDE_ackOD    optr
    DDE_completeMsg word ; Message to send to itself when ackCount goes to zero.
DetachDataEntry    ends
```

Library: **Objects/metaC.def**

■ **DevicePresent**

```
DevicePresent    etype    word
    DP_NOT_PRESENT    enum    DevicePresent, 0xffff
    DP_CANT_TELL    enum    DevicePresent, 0
    DP_PRESENT    enum    DevicePresent, 1
    DP_INVALID_DEVICE    enum    DevicePresent, 0xfffe
```

DP_NOT_PRESENT

Driver knows that a device is not present.

DP_CANT_TELL

Driver cannot determine if a device is present.

DP_PRESENT

Driver knows that a device is present.

DP_INVALID_DEVICE

An unknown device string was passed.

Library: **driver.def**

■ DirPathInfo

```

DirPathInfo      record
    DPI_EXISTS_LOCALLY      :1
    DPI_ENTRY_NUMBER_IN_PATH :7
    DPI_STD_PATH            StandardPath:8
DirPathInfo      end
  
```

DPI_EXISTS_LOCALLY

File exists in directory under primary tree (usually a local directory, not a server-based one).

Library: **file.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ DiskCopyCallback

```

DiskCopyCallback    etype          word, 0
DCC_GET_SOURCE_DISK    enum DiskCopyCallback
;      Desc:      prompt the user to insert the source disk.
;      Pass:
;      ax - DCC_GET_SOURCE_DISK
;      dl - 0 based drive number
;      Return:
;      ax = 0 to continue, non-0 to abort

DCC_REPORT_NUM_SWAPS    enum DiskCopyCallback
;      Desc:      tell the user how many times to swap disks
;                  in order to accomplish the copy.
;      Pass:
;      ax - DCC_REPORT_NUM_SWAPS
;      dx - number of swaps required
;      Return:
;      ax = 0 to continue, non-0 to abort

DCC_GET_DEST_DISK      enum DiskCopyCallback
;      Desc:      prompt the user to insert the destination disk.
;      Pass:
;      ax - DCC_GET_DEST_DISK
;      dl - 0 based drive number
;      Return:
;      ax = 0 to continue, non-0 to abort

DCC_VERIFY_DEST_DESTRUCTION    enum DiskCopyCallback
;      Desc:      make sure the user really wants to biff the formatted disk
;                  inserted as the destination.
;      Pass:
;      ax - DCC_VERIFY_DEST_DESTRUCTION
;      bx - disk handle of destination disk
;      dl - 0 based drive number
;      Return:
;      ax = 0 to continue, non-0 to abort

DCC_REPORT_FORMAT_PCT    enum DiskCopyCallback
;      Desc:      During the formatting phase of the copy, report how much of
;                  the format is complete.
;      Pass:
;      ax - DCC_REPORT_FORMAT_PCT
;      dx - percentage of destination disk formatted
;      Return:
;      ax = 0 to continue, non-0 to abort

DCC_REPORT_READ_PCT      enum DiskCopyCallback
;      Desc:      Report how much of the source disk has been read.
;      Pass:

```

```

;          ax - DCC_REPORT_READ_PCT
;          dx - percentage of source disk read
;      Return:
;          ax = 0 to continue, non-0 to abort
DCC_REPORT_WRITE_PCT      enum DiskCopyCallback
;      Desc:   Report how much of the destination disk has been written.
;      Pass:
;          ax - DCC_REPORT_WRITE_PCT
;          dx - percentage of destination disk written
;      Return:
;          ax = 0 to continue, non-0 to abort

```

This type, when passed to the routine **DiskCopy**, specifies the type of callback operation to use with this routine.

Library: **disk.def**

■ DiskCopyError

DiskCopyError	etype	word, FormatError
ERR_INVALID_SOURCE_DRIVE	enum	DiskCopyError
ERR_INVALID_DEST_DRIVE	enum	DiskCopyError
ERR_SOURCE_DRIVE_DOESNT_SUPPORT_DISK_COPY	enum	DiskCopyError
ERR_DEST_DRIVE_DOESNT_SUPPORT_DISK_COPY	enum	DiskCopyError
ERR_DRIVES_HOLD_DIFFERENT_FILESYSTEM_TYPES	enum	DiskCopyError
ERR_SOURCE_DISK_INCOMPATIBLE_WITH_DEST_DRIVE	enum	DiskCopyError
ERR_SOURCE_DISK_NOT_FORMATTED	enum	DiskCopyError
ERR_COULD_NOT_REGISTER_FORMATTED_DESTINATION_DISK	enum	DiskCopyError
ERR_DISKCOPY_INSUFFICIENT_MEM	enum	DiskCopyError
ERR_CANT_READ_FROM_SOURCE	enum	DiskCopyError
ERR_CANT_WRITE_TO_DEST	enum	DiskCopyError
ERR_OPERATION_CANCELLED	enum	DiskCopyError
ERR_DISK_IS_IN_USE	enum	DiskCopyError
ERR_INVALID_SOURCE_DRIVE	enum	DiskCopyError

These enumerated types begin at the point where **FormatError** ends, so an error formatting the destination disk can be returned immediately without conversion.

ERR_INVALID_SOURCE_DRIVE

The passed source drive doesn't exist.

ERR_INVALID_DEST_DRIVE

The passed destination drive doesn't exist.

ERR_DRIVES_HOLD_DIFFERENT_FILESYSTEM_TYPES

The two drives are managed by different file system drivers, so a disk cannot be copied between the two with this function.

Revision: ■

Draft Dated (4/18/94)

Reference book



ERR_SOURCE_DISK_INCOMPATIBLE_WITH_DEST_DRIVE

The source disk is formatted in a manner that the destination drive does not support.

ERR_SOURCE_DISK_NOT_FORMATTED

The source disk was not properly formatted.

ERR_COULD_NOT_REGISTER_FORMATTED_DESTINATION_DISK

The copy operation attempted to register the source disk after prompting for it and the registration failed.

Library: **disk.def**

■ DiskCopyFlags

```
DiskCopyFlags      record
  DCF_GREEDY       :1
  :7
DiskCopyFlags      end
```

DCF_GREEDY

If set, the copy operation will use as much memory as necessary and possible to minimize disk swaps. This flag is applicable only if source and destination drives are the same.

Library: **disk.def**

■ DiskFindResult

```
DiskFindResult      etype      word
  DFR_UNIQUE         enum      DiskFindResult
  DFR_NOT_UNIQUE     enum      DiskFindResult
  DFR_NOT_FOUND      enum      DiskFindResult
```

Library: **disk.def**

■ DiskFormatFlags

```
DiskFormatFlags      record
  :13
  DFF_CALLBACK_PCT_DONE :1
  DFF_CALLBACK_CYL_HEAD :1
  DFF_FORCE_ERASE      :1
DiskFormatFlags      end
```

DFF_CALLBACK_PCT_DONE

This flag is set if the disk format should call the callback with the percent-complete of the operation.

DFF_CALLBACK_CYL_HEAD

This flag is set if the disk format should call the callback with the current cylinder and head.

DFF_FORCE_ERASE

This flag is set if we wish to force erasure of the entire disk.

Library: **disk.def**

■ DiskInfoStruct

```

DiskInfoStruct      struct
    DIS_blockSize    word          ; number of bytes in which file system allocations
                                ; are performed. Useful as an efficient
                                ; buffer size for disk transfers, with
                                ; certain restrictions.
    DIS_freeSpace     sdword        ; number of bytes free on the disk.
    DIS_totalSpace    sdword        ; number of bytes on the entire disk.
    DIS_name          char          VOLUME_BUFFER_SIZE dup(?)
DiskInfoStruct      ends
  
```

Library: **disk.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ DiskRestoreError

```
DiskRestoreError    etype          word, 0, 1
DRE_DISK_IN_DRIVE    enum DiskRestoreError
;    This is the value returned by DiskRestore itself and the callback
;    routine (if called at all) if the disk is in the drive.
;    In the callback's case, of course, it cannot be sure that the disk
;    is in the drive; it merely thinks it still is.
DRE_DRIVE_NO_LONGER_EXISTS    enum DiskRestoreError
;    The drive in which the disk was registered no longer exists and the
;    file system driver either isn't loaded or couldn't restore the
drive.
DRE_REMOVABLE_DRIVE_DOESNT_HOLD_DISK    enum DiskRestoreError
;    The disk was in a removable-media drive and that drive doesn't
;    contain the disk.
DRE_USER_CANCELED_RESTORE    enum DiskRestoreError
;    This type is solely for callback routines to use, as it implies the
;    user was asked, which DiskRestore will not do.
DRE_COULDNT_CREATE_NEW_DISK_HANDLE    enum DiskRestoreError
;    The operation attempted to create the new disk handle after
;    deciding the disk was in the drive, but had some difficulty finding
;    the disk name, etc.
DRE_REMOVABLE_DRIVE_IS_BUSY    enum DiskRestoreError
;    The disk was in a removable-media drive that is currently marked
;    busy, so the system could neither confirm nor deny that it holds
;    the saved disk. Try again later.
DRE_NOT_ATTACHED_TO_SERVER    enum DiskRestoreError
;    The disk was from a network server to which we are not logged in.
DRE_PERMISSION_DENIED    enum DiskRestoreError
;    The disk was on a network which is (now) denying access to it.
DRE_ALL_DRIVES_USED    enum DiskRestoreError
;    The disk was on a network volume that isn't mounted, but there is
;    no drive left to which it can be mapped.
```

Library: **disk.def**

■ DisplayAspectRatio

```
DisplayAspectRatio    etype          byte
DAR_NORMAL            enum    DisplayAspectRatio;VGA, MCGA
DAR_SQUISHED          enum    DisplayAspectRatio;EGA, HGCA
DAR_VERY_SQUISHED     enum    DisplayAspectRatio;CGA
```

Library: **win.def**

■ DisplayClass

DisplayClass	etype	byte	
DC_TEXT	enum	DisplayClass	;denotes that display is ;character only (Not implemented)
DC_GRAY_1	enum	DisplayClass	;1 bit/pixel gray scale
DC_GRAY_2	enum	DisplayClass	;2 bit/pixel gray scale
DC_GRAY_4	enum	DisplayClass	;4 bit/pixel gray scale
DC_GRAY_8	enum	DisplayClass	;8 bit/pixel gray scale
DC_COLOR_2	enum	DisplayClass	;2 bit/pixel color index
DC_COLOR_4	enum	DisplayClass	;4 bit/pixel color index
DC_COLOR_8	enum	DisplayClass	;8 bit/pixel color index
DC_CF_RGB	enum	DisplayClass	;color with RGB values

Library: **win.def**

■ DisplayScheme

DisplayScheme	struct	
DS_colorScheme	ColorScheme	;passed in al
DS_displayType	DisplayType	;passed in ah
DS_unused	word	;passed in bx
DS_fontID	FontID	;passed in cx
DS_pointSize	sword	;passed in dx
DisplayScheme	ends	

DS_fontID & *DS_pointSize* are conveniently set up so they'll be in registers **cx:dx** to conform to graphics routines.

Library: **Objects/visC.def**

■ DisplaySize

DisplaySize	etype	byte	
DS_TINY	enum	DisplaySize	;tiny screens: CGA, 256x320
DS_STANDARD	enum	DisplaySize	;standard screens: EGA,VGA,HGC,MCGA
DS_LARGE	enum	DisplaySize	;large screens: 800x600 SVGA
DS_HUGE	enum	DisplaySize	;huge screens

Library: **win.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ DisplayType

```
DisplayType      record
    DT_DISP_SIZE      DisplaySize:2
    DT_DISP_ASPECT_RATIO      DisplayAspectRatio:2
    DT_DISP_CLASS      DisplayClass:4
DisplayType      end

    DT_DISP_SIZE
    Size of display.

    DT_DISP_ASPECT_RATIO
    Aspect ratio of display.

    DT_DISP_CLASS
    Class of driver (or closest match).
```

Library: **win.def**

■ DistanceUnit

```
DistanceUnit      etype      byte
    DU_POINTS      enum DistanceUnit
        ;U.S. points (72 per inch)
        ;Display format is "###.### pt"
        ;Entry format is "###.### pt"
    DU_INCHES      enum DistanceUnit
        ;Display format is "##.### in"
        ;Entry format is "##.### in" or "##.###" (double quotes = inches)
    DU_CENTIMETERS      enum DistanceUnit
        ;Display format is "###.### cm"
        ;Entry format is "###.### cm"
    DU_MILLIMETERS      enum DistanceUnit
        ;Display format is "###.### mm"
        ;Entry format is "###.### mm"
    DU_PICAS      enum DistanceUnit      ;U.S. picas (12 points)
        ;Display format is "###.### pi"
        ;Entry format is "###.### pi"
    DU_EUR_POINTS      enum DistanceUnit      ;European points
        ;Display format is "###.### ep"
        ;Entry format is "###.### ep"
    DU_CICEROS      enum DistanceUnit
        ;Display format is "###.### ci" (should fraction be in e points ???)
        ;Entry format is "###.### ci"
    DU_POINTS_OR_MILLIMETERS      enum DistanceUnit
        ;Depends on units for app
    DU_INCHES_OR_CENTIMETERS      enum DistanceUnit
        ;Depends on units for app

LOCAL_DISTANCE_BUFFER_SIZE= 32
```

Library: **localize.def**

■ DocQuitStatus

```
DocQuitStatus      etype      word
DQS_OK             enum       DocQuitStatus
DQS_CANCEL         enum       DocQuitStatus
DQS_DELAYED        enum       DocQuitStatus
DQS_SAVE_ERROR     enum       DocQuitStatus
```

Library: **Objects/gDocGrpC.def**

■ DocumentCommonParams

```
DocumentCommonParams  struct
DCP_name              FileLongName
DCP_diskHandle        word
DCP_path              PathName
DCP_docAttrs          GenDocumentAttrs
DCP_flags              DocumentOpenFlags
DCP_connection        IACPConnection ; OPEN_DOC & SEARCH_FOR_DOC only: IACP
                                   ; connection that requested the open.
                                   ; 0 if open requested by the user.

DocumentCommonParams  ends
```

Not all of these flags are always needed, but they are all in one structure to simplify passing the flags around.

Library: **Objects/gDocC.def**

■ DocumentFileChangedParams

```
DocumentFileChangedParams  struct
DFCP_name              FileLongName
DFCP_diskHandle        hpctr
DFCP_path              PathName
DFCP_display           optr
DFCP_document          optr

DocumentFileChangedParams  ends
```

Library: **Objects/gDocCtrl.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ DocumentOpenFlags

```
DocumentOpenFlags  record
    DOF_CREATE_FILE_IF_FILE_DOES_NOT_EXIST  :1
    DOF_FORCE_TEMPLATE_BEHAVIOR              :1
    DOF_SAVE_AS_OVERWRITE_EXISTING_FILE      :1
    DOF_REOPEN                              :1
    DOF_RAISE_APP_AND_DOC                    :1
    DOF_FORCE_REAL_EMPTY_DOCUMENT            :1
    DOF_OPEN_FOR_IACP_ONLY                   :1
    DOF_NO_ERROR_DIALOG:1                   :1
    DOF_NO_DOC_SEARCH                        :1
DocumentOpenFlags  end
```

Flags used for document OPEN operations:

DOF_CREATE_FILE_IF_FILE_DOES_NOT_EXIST

This bit controls behavior when the file to open does not exist. Setting this bit causes a new file to be created if this case.

DOF_FORCE_TEMPLATE_BEHAVIOR

Forces the document to be treated as a template

Flags used for document SAVE AS operations:

DOF_SAVE_AS_OVERWRITE_EXISTING_FILE

This bit controls behavior when the file to be Save As'd to exists. Setting this bit causes the file to be overwritten.

DOF_REOPEN

We are re-opening the file.

Flags used for document SEARCH_FOR_DOC operations:

DOF_RAISE_APP_AND_DOC

Raise the application & document to the top, too, regardless of non-zero nature of *DCP_connection*.

The following two fields are used when creating new documents.

DOF_FORCE_REAL_EMPTY_DOCUMENT

DOF_OPEN_FOR_IACP_ONLY

Set if document was opened for purpose of handling an IACP request only. If user opens in the meantime, this bit is cleared.

The following two fields are Internal, and you should try to ignore them.

DOF_NO_ERROR_DIALOG

Internal.

DOF_NO_DOC_SEARCH
Internal.

Library: **Objects/gDocC.def**

■ DosCodePage

DosCodePage	etype	word
CODE_PAGE_US	enum	DosCodePage, 437
CODE_PAGE_MULTILINGUAL	enum	DosCodePage, 850
CODE_PAGE_PORTUGUESE	enum	DosCodePage, 860
CODE_PAGE_CANADIAN_FRENCH	enum	DosCodePage, 863
CODE_PAGE_NORDIC	enum	DosCodePage, 865
CODE_PAGE_SJIS	enum	DosCodePage, 932

Library: **localize.def**

■ DosExecArgAndMemReqsStruct

Library:

■ DosExecFlags

DosExecFlags	record
DEF_PROMPT	:1
DEF_FORCED_SHUTDOWN	:1
DEF_INTERACTIVE	:1
DEF_INTERACTIVE	:1
DEF_SWAP_EXEC	:1
DEF_SWAP_TSR	:1
DEF_MEM_REQ	:1
	:2

DosExecFlags end

DEF_PROMPT

Set if we want to prompt the user to strike a key to return to GEOS.

DEF_FORCED_SHUTDOWN

Set if we want to force the user to shutdown (cannot abort, program must be run).

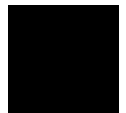
DEF_INTERACTIVE

Set if program being run is interactive shell and we should change \$PROMPT to tell the user to type "exit" to return to GEOS.

Revision: ■

Draft Dated (4/18/94)

Reference book



DEF_SWAP_EXEC

Set if GEOS should be swapped out instead of shutdown.

DEF_SWAP_TSR

Set if GEOS should swap itself out and TSR, without executing the program.

DEF_MEM_REQ

Set if a **DosExecArgAndMemReqStruct** is passed to DosExec in es:di instead of the argument string

Library: **system.def**

■ DosExecMemReq

```
DosExecMemReq    struct
    DEMR_minimum    word            ; minimum memory requirement
    DEMR_optimal    word            ; optimal memory requirement
DosExecMemReq    ends
```

Library:

■ DosExecMemReqsStruct

```
DosExecMemReqsStruct    struct
    DEMRS_tsr            BooleanByte    BB_FALSE    ; program is a TSR
    DEMRS_conventional    DosExecMemReq    <0,0>    ; conventional
    DEMRS_upper          DosExecMemReq    <0,0>    ; upper
    DEMRS_ems            DosExecMemReq    <0,0>    ; EMS
    DEMRS_xms            DosExecMemReq    <0,0>    ; XMS
    DEMRS_extended       DosExecMemReq    <0,0>    ; raw extended
DosExecMemReqsStruct    ends
```

Library:

■ DrawFlags

```
DrawFlags          record
    DF_EXPOSED              :1
    DF_OBJECT_SPECIFIC      :1
    DF_PRINT                :1
    DF_DONT_DRAW_CHILDREN   :1
    DF_DISPLAY_TYPE         DisplayClass:4
DrawFlags          end
```

DF_EXPOSED

Set if the current draw is the result of a MSG_META_EXPOSED being

Revision: ■

Draft Dated (4/18/94)

processed; if this is the case, we are in-between calls made to the window system of **GrBeginUpdate** and **GrEndUpdate**.

DF_OBJECT_SPECIFIC

This bit is used differently under different objects.

For scrolling list objects:

This flag is set if a composite which controls the drawing of its own children should *not* draw its children. i.e. the draw should act on only the composite itself.

For text objects:

This flag is used by the text object to draw all of its lines, not just the ones which aren't masked out. This is used to get rotated text up on the screen.

DF_PRINT

This flag is set if the draw is a result of a **MSG_META_EXPOSED_FOR_PRINT**. If this bit is set then **DF_EXPOSED** will be set also.

DF_DONT_DRAW_CHILDREN

This flag is set if composites should not be drawing their children.

DF_DISPLAY_TYPE

This flag is set to **DisplayClass** (*not* **DisplayType**).

Library: **Objects/visC.def**

■ DrawMonikerFlags

```
DrawMonikerFlags    record
    DMF_TEXT_ONLY           :1
    DMF_UNDERLINE_ACCELERATOR :1
    DMF_CLIP_TO_MAX_WIDTH   :1
    DMF_NONE                 :1
    DMF_Y_JUST               Justification:2
    DMF_X_JUST               Justification:2
DrawMonikerFlags    end
```

DMF_TEXT_ONLY

Set if we can only draw a text moniker on this object.

DMF_UNDERLINE_ACCELERATOR

Underlines accelerator key if set.

DMF_CLIP_TO_MAX_WIDTH

Causes the moniker to be clipped to its maximum width (passed elsewhere, if used).

DMF_NONE

TRUE to draw at current pen position.

Revision: ■

Draft Dated (4/18/94)

Reference book



DMF_Y_JUST
Vertical justification.

DMF_X_JUST
Horizontal justification.

Library: **Objects/visC.def**

■ DriveExtendedStatus

```
DriveExtendedStatus      record
    DES_LOCAL_ONLY       :1
    DES_READ_ONLY        :1
    DES_FORMATTABLE      :1
    DES_ALIAS             :1
    DES_BUSY              :1
    DES_EXTERNAL          DriveStatus:8
```

```
DriveExtendedStatus      end
```

DES_LOCAL_ONLY
Set if device cannot be viewed over a network.

DES_READ_ONLY
Set if device is read-only.

DES_FORMATTABLE
Set if disks can be formatted in the drive. If set, implies disks can be copied in the drive using **DiskCopy**.

DES_ALIAS
Set if drive is actually an alias for a path on another drive.

DES_BUSY
Set if drive will be busy for an extended period of time (e.g. when disk is being formatted).

DES_EXTERNAL
Externally-visible status flags.

Library: **drive.def**

■ DriverAttrs

```
DriverAttrs              record
    DA_FILE_SYSTEM        :1
    DA_CHARACTER           :1
    DA_HAS_EXTENDED_INFO  :1
                                :13
```

```
DriverAttrs              end
```

DA_FILE_SYSTEM
Driver is used primarily for file access.

DA_CHARACTER

Driver is used primarily with character-oriented devices.

DA_HAS_EXTENDED_INFO

Driver has **DriverExtendedInfo** structure, with attendant mandatory functions.

Library: **driver.def**

■ DriverEscCode

```
DriverEscCode      etype      word, 8000h, 1
    DRV_ESC_QUERY_ESC      enum    DriverEscCode; query for escape cpde
                                ; support
```

Library: **driver.def**

■ DriverExtendedFunction

```
DriverExtendedFunction  etype      word, DriverFunction, 2
    DRE_TEST_DEVICEenumDriverExtendedFunction
    ;      PASS:      dx:si      = pointer to null-terminated device name string
    ;      RETURN:  ax      = DevicePresent
    ;      carry set if DP_INVALID_DEVICE, clear otherwise
    ;      DESTROYS:di
    ;
    ;      This function tests the existence of a particular device the driver
    ;      supports.

    DRE_SET_DEVICEenumDriverExtendedFunction
    ;      PASS:      dx:si      = pointer to null-terminated device name string
    ;      RETURN:  nothing
    ;      DESTROYS:di
    ;
    ;      This function informs the driver which of its many devices it is to
    ;      support.
```

Library: **driver.def**

■ DriverExtendedInfoStruct

```
DriverExtendedInfoStruct  struct
    DEIS_common      DriverInfoStruct
    DEIS_resource      hptr.DriverExtendedInfoTable
DriverExtendedInfoStruct  ends
```

This structure is used by preferences to locate the names of devices supported by a particular driver.

Revision: ■

Draft Dated (4/18/94)

Reference book



An extended driver is one that can handle multiple types of devices, identified by ASCII strings that the driver provides. The specific device to be supported is specified by a DRE_SET_DEVICE call after the driver is loaded.

DEIS_common stores the regular driver information within a **DriverInfoStruct**.

DEIS_resource stores the resource containing additional driver information. The resource must be sharable so other geodes can lock it down. It should also be read-only. The **DriverExtendedInfoTable** is stored in a separate resource to keep the amount of fixed memory required to a minimum.

Library: **driver.def**

■ DriverExtendedInfoTable

```
DriverExtendedInfoTable  struct
    DEIT_common           LMemBlockHeader
    DEIT_numDevices       word
    DEIT_nameTable        nptr.lptr.char
    DEIT_infoTable        nptr.word
DriverExtendedInfoTable  ends
```

DEIT_common stores the common LMem header info. Just use {} to define this field and Esp will skip this field when defining the structure.

DEIT_numDevices stores the number of device types supported by the driver

DEIT_nameTable stores the pointer to the table of *DEIT_numDevices* pointers to the device names themselves. As indicated by this being an nptr, the table and the names lie in this same resource.

DEIT_infoTable stores the pointer to the table of *DEIT_numDevices*, with extra words containing driver-specific data for each device. The type of data stored here is specified by the driver-type-specific interface .def file (e.g. **mouseDriver.def**).

Library: **driver.def**

■ DriverFunction

```

DriverFunction      etype      word, 0, 2
DR_INIT            enum DriverFunction;Initialize driver
;      PASS:cx = di passed to GeodeLoad. Garbage if loaded via
;              GeodeUseDriver
;      dx = bp passed to GeodeLoad. Garbage if loaded via
;              GeodeUseDriver
;      RETURN:carry set if driver initialization failed. Driver will be
;              unloaded by the system.
;              carry clear if initialization successful.
;
;      DESTROYS:bp, ds, es, ax, di, si, cx, dx
;

DR_EXIT            enum DriverFunction;Exit driver
;      PASS:   nothing
;      RETURN: nothing
;      DESTROYS:ax, bx, cx, dx, si, di, ds, es
;
;      NOTES:If the driver has GA_SYSTEM set, the handler for this function
;              must be in fixed memory and may not use anything in movable
;              memory.

DRIVER_SUSPEND_ERROR_BUFFER_SIZE equ 128

DR_SUSPEND         enum DriverFunction
;      SYNOPSIS:Prepare the device for going into stasis while GEOS
;              is task-switched out. Typical actions include disabling
;              interrupts or returning to text-display mode.
;
;      PASS:   cx:dx      = buffer in which to place reason for refusal, if
;              suspension refused (DRIVER_SUSPEND_ERROR_BUFFER_SIZE
;              bytes long)
;      RETURN: carry set if suspension refused:
;              cx:dx      = buffer filled with null-terminated reason,
;              standard GEOS character set.
;              carry clear if suspension approved
;      DESTROYS:ax, di
;

DR_UNsuspend       enum DriverFunction
;      SYNOPSIS:Reconnect to the device when GEOS is task-switched
;              back in.
;
;      PASS:   nothing
;      RETURN: nothing
;      DESTROYS:ax, di
;
; Protocol number for "DriverFunction" interface. All other driver protocols

```

Revision: ■

Draft Dated (4/18/94)

Reference book



```
; will be based on this number.
;
DRIVER_PROTO_MAJOREqu      2
DRIVER_PROTO_MINOREqu      0
```

Library: **driver.def**

■ **DriverInfoStruct**

```
DriverInfoStruct    struct
    DIS_strategy      fptr.far
    DIS_driverAttributes  DriverAttrs
    DIS_driverType     DriverType
DriverInfoStruct    ends
```

This structure defines the characteristics of a particular driver. In general, applications will not need to access this structure unless they use a driver directly.

DIS_strategy stores the address of the strategy routine which calls the proper driver.

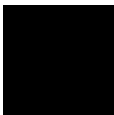
DIS_driverAttributes stores the device attributes for the driver.

DIS_driverType stores the type of driver (video, stream, printer, etc.).

Library: **driver.def**

■ **DriverType**

```
DriverType          etype      word, 1
    DRIVER_TYPE_VIDEO      enum   DriverType
    DRIVER_TYPE_INPUT      enum   DriverType
    DRIVER_TYPE_MASS_STORAGE  enum   DriverType
    DRIVER_TYPE_STREAM     enum   DriverType
    DRIVER_TYPE_FONT       enum   DriverType
    DRIVER_TYPE_OUTPUT     enum   DriverType
    DRIVER_TYPE_LOCALIZATION  enum   DriverType
    DRIVER_TYPE_FILE_SYSTEM  enum   DriverType
    DRIVER_TYPE_PRINTER     enum   DriverType
    DRIVER_TYPE_SWAP       enum   DriverType
    DRIVER_TYPE_POWER_MANAGEMENT  enum   DriverType
    DRIVER_TYPE_TASK_SWITCH  enum   DriverType
    DRIVER_TYPE_NETWORK     enum   DriverType
    DRIVER_TYPE_SOUND       enum   DriverType
    DRIVER_TYPE_PAGER       enum   DriverType
    DRIVER_TYPE_PCMCIA      enum   DriverType
    DRIVER_TYPE_FEP         enum   DriverType
```



Library: **driver.def**

■ **DriveStatus**

```
DriveStatus          record
    DS_PRESENT        :1          ; Set if physical drive exists
    DS_MEDIA_REMOVABLE :1          ; Set if disk can be removed from
                                   ; the drive.
    DS_NETWORK        :1          ; Set if drive is on the network (or
                                   ; accessed via network protocols),
                                   ; so disk cannot be formatted or
                                   ; copied.
                                   :1
    DS_TYPE            DriveType:4 ; Type of drive
DriveStatus          end
```

Library: **drive.def**

■ **DriveType**

```
DriveType            etype      byte
    DRIVE_5_25        enum      DriveType
    DRIVE_3_5         enum      DriveType
    DRIVE_FIXED       enum      DriveType
    DRIVE_RAM         enum      DriveType
    DRIVE_CD_ROM      enum      DriveType
    DRIVE_8           enum      DriveType
    DRIVE_PCMCIA      enum      DriveType
    DRIVE_UNKNOWN     enum      DriveType, 0xf
```

Library: **drive.def**

■ **DTCFeatures**

```
DTCFeatures          record
    DTCF_LIST         :1
    DTCF_CUSTOM       :1
DTCFeatures          end
```

These features flags (used with ATTR_GEN_CONTROL_REQUIRE_UI and ATTR_GEN_CONTROL_PROHIBIT_UI).

Library: **Objects/Text/tCtrlC.def**

■ DTCToolboxFeatures

```
DTCToolboxFeatures  record
DTCToolboxFeatures  end
```

Library: **Objects/Text/tCtrlC.def**

■ DWFixed

```
DWFixed  struct
    DWF_frac    word
    DWF_int     sdword
DWFixed  ends
```

This structure stores a 32 bit/16 bit fixed point number.

Library: **geos.def**

■ ElementArrayHeader

```
ElementArrayHeader struct
    EAH_meta      ChunkArrayHeader
    EAH_freePtr   word
ElementArrayHeader ends
```

This structure must be at the front of every element array. Since element arrays are special kinds of chunk arrays, the **ElementArrayHeader** must itself begin with a **ChunkArrayHeader**.

EAH_meta stores the **ChunkArrayHeader**.

EAH_freePtr stores the first free element within the element array. Applications should not examine or change this field.

Library: **chunkarr.def**

■ EMCDDetachData

```
EMCDDetachData  struct
    EMCDD_ackEvent    hpPtr
    EMCDD_childBlock  hpPtr
```

EMCDD_ackEvent

Recorded MSG_META_ACK that will be sent back by everyone on the EXPRESS_MENU_CHANGE list.

EMCDD_childBlock

Handle of child block to be freed when MSG_META_DETACH_COMPLETE comes in, saying that

everyone who could possibly care has acknowledged the detach, so it's safe to actually free the child block, which the GenControl object won't do when it receives a MSG_META_DETACH.

Library: **eMenuC.def**

■ EMCFeatures

```
EMCFeatures      record
  EMCF_GEOS_TASKS_LIST      :1
  EMCF_DESK_ACCESSORY_LIST  :1
  EMCF_MAIN_APPS_LIST       :1
  EMCF_OTHER_APPS_LIST      :1
  EMCF_CONTROL_PANEL        :1
  EMCF_DOS_TASK_LIST        :1
  EMCF_UTILITIES            :1
  EMCF_EXIT_TO_DOS          :1
EMCFeatures      end
```

Library: **eMenuC.def**

■ EmptyRowBlock

```
EmptyRowBlock    struct
  ERB_header      LMemBlockHeader <>
  ERB_handles     word N_ROWS_PER_ROW_BLOCK dup (0)
EmptyRowBlock    ends
```

An empty row block is an LMem block with space for several handles.

Library: **cell.def**

■ EndCreatePassFlags

```
EndCreatePassFlags record
  ECPF_ADJUSTED_CREATE      :1
EndCreatePassFlags end
```

ECPF_ADJUSTED_CREATE
The UIF_A_ADJUST flag was set in the START_SELECT operation that began the object creation.

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ EndCreateReturnFlags

```
EndCreateReturnFlags      record
    ECRF_NOT_CREATING      :1      ;object was not in create mode.
    ECRF_DESTROYED         :1      ;object was destroyed
EndCreateReturnFlags      end
```

Library: **grobj.def**

■ EndOfSongFlags

```
EndOfSongFlags            record
    EOSF_UNLOCK            :1 = 0    ; unlock block at EOS?
    EOSF_DESTROY           :1 = 0    ; destroy sound at EOS?
                                :6
EndOfSongFlags            end
```

Library: **sound.def**

■ EnsureActiveFTPRIORITYPreferenceData

```
EnsureActiveFTPRIORITYPreferenceData      struct
    EAFPPD_priority      word
    EAFPPD_avoidOpPtr     optr
EnsureActiveFTPRIORITYPreferenceData      ends
```

Library:

■ EnsureNoMenusInStayUpModeParams

```
EnsureNoMenusInStayUpModeParams      struc
    ENMISUMP_menuCount      word      ;Number of menus released so far
EnsureNoMenusInStayUpModeParams      ends
```

Library: **ui.def**

■ EntryPointRelocation

```
EntryPointRelocation      struct
    EPR_geodeName      char GEODE_NAME_SIZE dup (?)
    EPR_entryNumber     word
EntryPointRelocation      ends
```

This structure is passed to **ObjRelocateEntryPoint**.

Library: **object.def**

Revision: ■

Draft Dated (4/18/94)

■ EnvelopeOrientation

EnvelopeOrientation	etype	byte, 0, 1
EO_PORTAIT	enum	EnvelopeOrientation
EO_LANDSCAPE	enum	EnvelopeOrientation

Library: **print.def**

■ ErrorCheckingFlags

```
ErrorCheckingFlags record
  ECF_UNUSED_1:1
  ECF_UNUSED_2:1
  ECF_ANAL_VMEM:1
  ECF_FREE:1           ;Ensure that all free blocks are 0xcccc
  ECF_HIGH:1           ; lot of random extra checking (old NORMAL)
  ECF_LMEM:1           ;Internal lmem checking
  ECF_BLOCK_CHECKSUM:1 ;Checksum on a particular block
  ECF_GRAPHICS:1       ;Misc graphics stuff
  ECF_SEGMENT:1        ;Extensive segment checking
  ECF_NORMAL:1         ;Misc kernel error checking
  ECF_VMEM:1           ;VM file consistency
  ECF_APP:1            ;Application error checking (if implemented
                      ;by applications)
  ECF_LMEM_MOVE:1      ;Force lmem blocks to move whenever possible
  ECF_UNLOCK_MOVE:1   ;Force unlocked blocks to move
  ECF_VMEM_DISCARD:1  ;Force clean VM blocks to be discarded
ErrorCheckingFlags end
```

Library: **ec.def**

■ EvalErrorData

```
EvalErrorData      struct
  EED_errorCode    ParserScannerEvaluatorError
EvalErrorData      ends
```

Library: **parse.def**



■ **EvalFlags**

```

EvalFlags          record
    EF_MAKE_DEPENDENCIES      :1      ; Make dependencies instead of
                                   ; recalculating
    EF_ONLY_NAMES             :1      ; Only name dependencies
    EF_KEEP_LAST_CELL         :1      ; Don't dereference the last cell
    EF_NO_NAMES               :1      ; Only non-name dependencies
    ;
    ; This flag is set inside the evaluator and shouldn't be used by
    ; applications.
    ;
    EF_ERROR_PUSHED           :1      ; Set: if an error was pushed on the arg
                                   ; stack
                                   :3
EvalFlags          end

```

Library: **parse.def**

■ **EvalFunctionData**

```

EvalFunctionData   struct
    EFD_functionID  FunctionID      ; Function ID if a function
    EFD_nArgs       word            ; Number of arguments
EvalFunctionData   ends

```

Library: **parse.def**

■ **EvalNameData**

```

EvalNameData       struct
    END_name        word
EvalNameData       ends

```

Library: **parse.def**

■ **EvalOperatorData**

```

EvalOperatorData   struct
    EOD_opType      OperatorType    ; Type of operator
EvalOperatorData   ends

```

Library: **parse.def**

■ EvalParameters

```

EvalParameters      struct
    EP_common        CommonParameters <>
    ;
    ; Possible callbacks:
    ;     CT_LOCK_NAME, CT_LOCK_FUNCTION, CT_UNLOCK
    ;
    EP_flags          EvalFlags <> ; Evaluator flags
    ;
    ; Everything below this point is initialized by the Evaluator.
    ;
    EP_fpStack        word          ; Floating point stack pointer
    EP_depHandle      word          ; Block handle of dependency block
    EP_nestedLevel    word          ; Levels of nesting

    EVAL_MAX_NESTED_LEVELS = 32
    EP_nestedAddresses dword EVAL_MAX_NESTED_LEVELS dup (?)
    align              word
EvalParameters      ends

```

This structure provides information when the evaluator is invoked. This structure is passed in a stack frame.

Library: **parse.def**

■ EvalRangeData

```

EvalRangeData      struct
    ERD_firstCell    CellReference <>
    ERD_lastCell     CellReference <>
EvalRangeData      ends

```

Library: **parse.def**

■ EvalStackArgumentData

```

EvalStackArgumentData union
    ESAD_string        EvalStringData
    ESAD_range         EvalRangeData
    ESAD_error         EvalErrorData
EvalStackArgumentData end

```

Library: **parse.def**

Revision: ■
Draft Dated (4/18/94)

Reference book



■ **EvalStackArgumentType**

```
EvalStackArgumentType      record
    ESAT_EMPTY      :1          ; Set: Argument came from an empty cell
    ;
    ; Only one of the following will ever be set at a time for
    ; arguments on the evaluator argument stack.
    ;
    ESAT_ERROR      :1          ; Set: Argument is an error
    ESAT_RANGE      :1          ; Set: Argument is a range
    ESAT_STRING      :1          ; Set: Argument is a string
    ESAT_NUMBER      :1          ; Set: Argument is a number
                                :1
    ;
    ; Numbers have some possible sub-types
    ;
    ESAT_NUM_TYPENumberType:2    ; The type of the number
EvalStackArgumentType      end
```

Library: **parse.def**

■ **EvalStackOperatorData**

```
EvalStackOperatorData      union
    ESOD_operator      EvalOperatorData
    ESOD_function      EvalFunctionData
EvalStackOperatorData      end
```

Library: **parse.def**

■ **EvalStackOperatorType**

```
EvalStackOperatorType      etype      byte, 0, 1
    ESOT_OPERATOR      enum      EvalStackOperatorType
    ESOT_FUNCTION      enum      EvalStackOperatorType
    ESOT_OPEN_PAREN      enum      EvalStackOperatorType
    ESOT_TOP_OF_STACK      enum      EvalStackOperatorType
```

Library: **parse.def**

■ **EvalStringData**

```
EvalStringData      struct
    ESD_length      word          ; Length of the string. (String data follows.)
EvalStringData      ends
```

Library: **parse.def**

■ EvaluatePositionNotes

```
EvaluatePositionNotes    record
    EPN_PADDING           :14
    EPN_SELECTION_LOCK_SET :1      ;Object's selection lock is set.
    EPN_BLOCKS_LOWER_OBJECTS :1    ;GrObj blocks, covers up or otherwise
                                   ;completely obscures objects
                                   ;underneath it at the position

EvaluatePositionNotes    end
```

Library: **grobj.def**

■ EvaluatePositionRating

```
EvaluatePositionRating    etype    byte, 0
    EVALUATE_NONE          enum     EvaluatePositionRating
    EVALUATE_SUB_LOW       enum     EvaluatePositionRating
    EVALUATE_LOW           enum     EvaluatePositionRating
    EVALUATE_SUB_MEDIUM    enum     EvaluatePositionRating
    EVALUATE_MEDIUM        enum     EvaluatePositionRating
    EVALUATE_SUB_HIGH      enum     EvaluatePositionRating
    EVALUATE_HIGH          enum     EvaluatePositionRating

    EVALUATE_NONE
    Point is not interesting at all.

    EVALUATE_SUB_LOW
    This type is not currently used.

    EVALUATE_LOW
    Point is in a rectangle bounding an object.

    EVALUATE_SUB_MEDIUM
    This type is not currently used.

    EVALUATE_MEDIUM
    Point is inside an enclosed but not a filled object.

    EVALUATE_SUB_HIGH
    This type is not currently used.

    EVALUATE_HIGH
    Point is on a line or a filled - or partially filled - area.
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ ExitFlags

```
ExitFlags          record
    EF_PANIC        :1      ; Set if the exit is unstable; in this case we
                           ; choose not to write out the .ini file.
    EF_RUN_DOS      :1      ; Set if we should run a DOS program upon exit
    EF_OLD_EXIT     :1      ; Set if we should use old-style (int 20h) exit
                           ; call (if accidentally run under DOS 1.x).
    EF_RESET        :1      ; Set if we should reset the machine instead of
                           ; exiting.
    EF_RESTART      :1      ; Set if should reload GEOS at the end.
ExitFlags          end
```

Library: **system.def**

■ ExportControlAttrs

```
ExportControlAttrs record
    ECA_IGNORE_INPUT      :1      ; ignore input while export occurs
                           :15
ExportControlAttrs end
```

Library: **impex.def**

■ ExportControlFeatures

```
ExportControlFeatures record
    EXPORTCF_EXPORT_TRIGGER      :1 ; export trigger
    EXPORTCF_FORMAT_OPTIONS      :1 ; export format UI parent, under which is
                           ; placed any UI specific to the currently
                           ; selected format
    EXPORTCF_BASIC               :1 ; export file selector, export format list,
                           ; export file name, and export app UI parent,
                           ; under which is placed any UI specific to
                           ; the app
    EXPORTCF_GLYPH               :1 ; glyph at top of export dialog box
ExportControlFeatures          end
```

These Feature flags are used with ATTR_GEN_CONTROL_REQUIRE_UI and ATTR_GEN_CONTROL_PROHIBIT_UI.

Library: **impex.def**

■ ExportControlToolboxFeatures

```
ExportControlToolboxFeatures  record
    EXPORTCTF_DIALOG_BOX      :1
ExportControlToolboxFeatures  end
```

Library: **impex.def**

■ ExtSelFlags

```
ExtSelFlags                    record
    ESF_INITIAL_SELECTION      :1      ;set if initial selection -- will update
                                      ; all items between anchor and extent
    ESF_XOR_INDIVIDUAL_ITEMS   :1      ;whether xoring changed items. If
                                      ; clear, sets items within new
                                      ; selection, clears others.
    ESF_CLEAR_UNSELECTED_ITEMS:1      ;set to clear non-selected items.
    ESF_SELECT                  :1      ;set when items in the selection
                                      ; should be turned on, rather than
                                      ; off.
                                      :4
ExtSelFlags                    end
```

Library: **Objects/gItemGC.def**

■ FACFeatures

```
FACFeatures                    record
    FACF_FONT_WEIGHT           :1
    FACF_FONT_WIDTH            :1
    FACF_TRACK_KERNING         :1
FACFeatures                    end
```

Library: **Objects/Text/tCtrlC.def**

■ FACToolboxFeatures

```
FACToolboxFeatures  record
FACToolboxFeatures  end
```

Library: **Objects/Text/tCtrlC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FatalErrors

```
FatalErrors          etype word, 0
    CAN_NOT_USE_CHUNKSIZEPTR_MACRO_ON_EMPTY_CHUNKS    enum FatalErrors
    CHUNK_ARRAY_BAD_ELEMENT                            enum FatalErrors
    MACRO_REQUIRES_FIXED_SIZE_ELEMENTS                 enum FatalErrors
    CANNOT_USE_DBCS_IN_THIS_VERSION                   enum FatalErrors
    ; Double Byte characters (DBCS) are not supported in this version of PC/GEOS
```

FatalErrors is an enumerated type into which named error codes are placed. The members placed in the type should be accessible to all modules of a patient or be assigned ranges by the programmer. That makes no difference to Swat. NOTE: definition must lie outside the StartKernel/EndKernel bounds if Swat is to receive the type unmolested.

FatalErrors are errors global to the entire system.

Library: **ec.def**

■ FCFeatures

```
FCFeatures          record
    FCF_SHORT_LIST   :1
    FCF_LONG_LIST    :1
FCFeatures          end
```

Library: **Objects/Text/tCtrlC.def**

■ FCToolboxFeatures

```
FCToolboxFeatures   record
    FCTF_TOOL_LIST   :1
FCToolboxFeatures   end
```

Library: **Objects/Text/tCtrlC.def**

■ FEDosInfo

```

FEDosInfo      struct
  FEDI_attributes      FileAttrs      ; file's attributes
  FEDI_modified        FileDateAndTime ; file's modification timestamp
  FEDI_fileSize        dword          ; file's size in bytes
  FEDI_name            FileLongName   ; file's name and extension in
                                      ; the form of a null terminated
                                      ; string
  FEDI_pathInfo        DirPathInfo
FEDosInfo      ends

```

This structure is used with **FileEnum**.

Library: **fileEnum.def**

■ FENAMEAndAttr

```

FENAMEAndAttr  struct
  FENAA_attr    FileAttrs
  FENAA_name    FileLongName
FENAMEAndAttr  ends

```

This structure is used with **FileEnum**.

Library: **fileEnum.def**

■ FFA_stackFrame

```

FFA_stackFrame  union
  FFA_float      FloatFloatToAsciiData
  FFA_dateTime   FloatFloatToDateTimeData
FFA_stackFrame  end

```

Library: **math.def**

■ FFCFeatures

```

FFCFeatures      record
                                      :14
  FCF_FORMAT_LIST :1
  FCF_DEFINE_FORMATS :1
FFCFeatures      end

```

Library: **math.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FieldBGFormatType

```
FieldBGFormatType  etype word
    FBGFT_STANDARD_GSTRING  enum FieldBGFormatType
    ;Just a standard graphics string.
    FBGFT_BITMAP_SLICES     enum FieldBGFormatType
    ;(Not currently supported)
```

Library: **backgrnd.def**

■ FieldInfo

```
FieldInfo          struct
    FI_nChars       word           ; Number of characters in the field
    FI_position     word           ; X position of field on line
    FI_width        word           ; Width of the field
    FI_tab          TabReference  ; Reference to a tab in the ruler
FieldInfo          ends
```

Library: **text.def**

■ FileAccess

```
FileAccess         etype byte, 0
    FA_READ_ONLY   enum FileAccess
    FA_WRITE_ONLY  enum FileAccess
    FA_READ_WRITE  enum FileAccess
```

Library: **file.def**

■ FileAccessFlags

```
FileAccessFlags    record
    :1=0,           ; Must be 0.
    FAF_EXCLUDE    FileExclude:3, ; What others may not do.
    :2=0,           ; Must be 0.
    FAF_MODE       FileAccess:2,  ; What caller wants to do.
FileAccessFlags    end
```

Library: **file.def**

■ FileAddStandardPathFlags

```
FileAddStandardPathFlags      record
    :16
FileAddStandardPathFlags      end
```

Library:

■ FileAttrs

```
FileAttrs                      record
    :1=0,
    FA_LINK                    :1          ; File is a link
    FA_ARCHIVE                 :1,         ; File requires backup (modified since FA_ARCHIVE
                                           ; last cleared)
    FA_SUBDIR                  :1,         ; File is actually a subdirectory
    FA_VOLUME                  :1,         ; File is actually a volume label
    FA_SYSTEM                  :1,         ; File is for the system (kernel, e.g.)
    FA_HIDDEN                  :1,         ; File should not be seen by regular searches.
    FA_RDONLY                  :1,         ; File may not be written
FileAttrs                      end
```

Library: **file.def**

■ FileChangeBatchNotificationData

```
FileChangeBatchNotificationData  struct
    FCBND_end                  nptr
    FCBND_items                label FileChangeBatchNotificationItem
FileChangeBatchNotificationData  ends
```

FCBND_end stores the ending offset of the array of
FileChangeBatchNotificationItem structures.

Library: **gcnlst.def**

■ FileChangeBatchNotificationItem

```
FileChangeBatchNotificationItem  struct
    FCBNI_type                 FileChangeNotificationType
    FCBNI_disk                 word
    FCBNI_id                   FileID
    FCBNI_name                 label FileLongName ; Only present if required by FCBNI_type.
FileChangeBatchNotificationItem  ends
```

Library: **gcnlst.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **FileChangeNotificationData**

```
FileChangeNotificationData    struct
    FCND_disk    word ; handle for disk on which the change occurred
    FCND_id      FileID    ; 32-bit identifier for the directory in
                           ; which the change occurred, or for the file
                           ; to which the change occurred.
    FCND_name    FileLongName ; For those notifications that require it,
                           ; the virtual name of the file or directory
                           ; to which the change occurred.
FileChangeNotificationData    ends
```

Library: **gcncist.def**

■ FileChangeNotificationType

```
FileChangeNotificationType    etype word
    FCNT_CREATE                enum FileChangeNotificationType
                                ; File or directory created. FCND_id is the id of the containing
                                ; directory. FCND_name is the name of the new file or directory.

    FCNT_RENAME                enum FileChangeNotificationType
                                ; File or directory renamed. FCND_id is the identifier for the file
                                ; or directory, and FCND_name is its new name.

    FCNT_OPEN                  enum FileChangeNotificationType
                                ; A file has been closed. FCND_id is the 32-bit identifier for the
                                ; file. FCND_name is undefined and may not be present. This is
                                ; generated only if someone has called
                                ; FileEnableOpenCloseNotification.

    FCNT_DELETE                enum FileChangeNotificationType
                                ; File or directory deleted. FCND_id is the 32-bit identifier for
                                ; the file or directory that was deleted. FCND_name is undefined and
                                ; may not be present.

    FCNT_CONTENTS              enum FileChangeNotificationType
                                ; File contents changed. This is sent only when FileCommit or
                                ; FileClose is called for a file and the file has been modified.
                                ; FCND_id is the 32-bit identifier for the affected file. FCND_name
                                ; is undefined and may not be present.

    FCNT_ATTRIBUTES            enum FileChangeNotificationType
                                ; File attributes changed. This is sent once all changes have been
                                ; made during a given FileSetAttributes, FileSetHandleExtAttributes,
                                ; or FileSetPathExtAttributes call. FCND_id is the 32-bit identifier
                                ; for the affected file. FCND_name is undefined and may not be
                                ; present.

    FCNT_DISK_FORMAT           enum FileChangeNotificationType
                                ; A disk has been formatted. FCND_id and FCND_name are undefined and
                                ; may not be present.

    FCNT_CLOSE                 enum FileChangeNotificationType
                                ; A file has been closed. FCND_id is the 32-bit identifier for the
                                ; file. FCND_name is undefined and may not be present. This is
                                ; generated only if someone has called
                                ; FileEnableOpenCloseNotification.

    FCNT_BATCH                 enum FileChangeNotificationType
                                ; The block is a FileChangeBatchNotificationData block, holding
                                ; multiple notifications. Notifications are collected into a batch
                                ; when a thread calls FileBatchChangeNotifications. All the
                                ; notifications are sent when it calls FileFlushChangeNotifications.
```

Revision: ■

Draft Dated (4/18/94)

Reference book



```
; Any application performing a substantial number of changes to
; the file system (e.g. deleting a directory tree) should tell the
; system to batch its notifications and flush them when it's all
; done. This will reduce the number of handles required to alert all
; interested parties to the changes.
```

Library: **gcnlst.def**

■ FileCreateFlags

```
FileCreateFlags    record
    FCF_NATIVE      :1
    FCF_NATIVE_WITH_EXT_ATTRS :1
                    :4
    FCF_MODE        FileCreateMode:2
FileCreateFlags    end
```

FCF_NATIVE

Create file to be compatible with the file system on which it resides. This may mean that most extended attributes are not supported for the file, unless the file system itself supports them (which DOS file systems do not).

FCF_NATIVE_WITH_EXT_ATTRS

Create file with a name compatible with the file system on which it resides, but support extended attributes. The driver may place restrictions on what sort of name may be used, and will return ERROR_INVALID_NAME if the name passed falls beyond the pale.

FCF_MODE

How the file should be created.

Library: **file.def**

■ FileCreateMode

```
FileCreateMode    etype byte, 0
    FILE_CREATE_TRUNCATE    enum FileCreateMode
    FILE_CREATE_NO_TRUNCATE enum FileCreateMode
    FILE_CREATE_ONLY        enum FileCreateMode
```

Library: **file.def**

■ **FileDate**

```
FileDate          record
    FD_YEAR       :7,          ; year since 1980
    FD_MONTH      :4,          ; month (1-12)
    FD_DAY        :5,          ; day of the month (1-31)
FileDate          end
```

Library: **file.def**

■ **FileDateAndTime**

```
FileDateAndTime   struct
    FDAT_date      FileDate
    FDAT_time      FileTime
FileDateAndTime   ends
```

Library: **file.def**

■ **FileEnumCallbackData**

```
FileEnumCallbackData  struct
    FECD_attrs      label FileExtAttrDesc
FileEnumCallbackData  ends
```

FECD_attrs stores the array of extended-attribute descriptors for the current file. The end of the array is signaled by a **FileExtAttrDesc** with *FEA_END_OF_LIST* in its *FEAD_attr* field. All the attribute values lie in the same segment as the **FileEnumCallbackData**, so their *FEAD_value.segment* will be *ds* unless the file doesn't have that particular attribute, in which case *FEAD_value.segment* will be 0.

Library: **fileEnum.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FileEnumParams

```
FileEnumParams      struct
    FEP_searchFlags      FileEnumSearchFlags 0
    FEP_returnAttrs      fptr.FileExtAttrDesc 0
    FEP_returnSize       word 0
    FEP_matchAttrs       fptr.FileExtAttrDesc0
    FEP_bufSize          word FE_BUFSIZE_UNLIMITED
    ;
    FE_BUFSIZE_UNLIMITED equ 1 ; Value to pass in FEP_bufSize
                                ; to place no limit on the
                                ; number of files for which
                                ; to return data.

    FEP_skipCount        word 0
    FEP_callback         fptr.far 0
    FEP_callbackAttrs     fptr.FileExtAttrDesc
    FEP_cbData1          dword 0
    FEP_cbData2          dword 0
    FEP_headerSize       word 0
    even
FileEnumParams      ends
```

FEP_searchFlags stores the flags to control the **FileEnum** search operation.

FEP_returnAttrs stores the array of attributes that are to be returned from the **FileEnum** operation. The entries in the returned buffer can be of an arbitrary size; the size is controlled by the *FEP_returnSize* parameter. Each extended attribute returned for a file that matches is copied into the entry at an offset given by the *FEAD_value.offset* for the attribute. The number of bytes copied cannot exceed the value of *FEAD_size*.

If the segment is 0, the offset is of type **FileEnumStandardReturnType**, specifying the attributes to be returned for a standard structure (as defined later in this file). *FEP_returnSize* should still be either the size of the appropriate structure, or larger if that structure has been incorporated into a larger one of your own. The standard return type *FESRT_COUNT_ONLY* exists so you receive no information about the files that match, just their number (in **dx**). One of *FEP_bufSize* or *FEP_returnSize* should be 0 in this case.

The last entry in the array has *FEA_END_OF_LIST* as its *FEAD_attr*.

FEP_returnSize stores the size of each entry in the returned buffer.

FEP_matchAttrs stores the array of attributes that are to be matched by **FileEnum** itself. For attributes that are records (and hence a word or smaller), *FEAD_value.offset* holds the bits that must be set in the attribute, while *FEAD_value.segment* holds the bits that must *not* be set in the

attribute's actual value. For all other attributes, *FEAD_value* is a pointer to the exact value to match. *FEAD_size* gives the size of that value.

The last entry in the array has *FEA_END_OF_LIST* as its *FEAD_attr*. If all the checks are to be performed by the callback, or if all files are desired, regardless of their attributes, *FEP_matchAttrs.segment* may be passed as 0. *FEP_matchAttrs.offset* may be anything in this case.

FEP_bufSize stores the *number* of structures that *FEP_buffer* can hold. This is used as the maximum number of files to find. The actual size of the buffer (in bytes) is determined by this and the *FEP_returnType*. If set to 0, the *dx* returned is a count of the matching files in the directory.

FEP_skipCount stores the number of matches to skip before storing matching entries in *FEP_buffer*. This can be used to make several passes through the files in a directory. Each pass will process the next *X* number of files in the directory:

```
FileEnum(skipCount=0, FEP_bufSize=20)
process(FEP_buffer)
FileEnum(skipCount=20, FEP_bufSize=20)
process(FEP_buffer)
FileEnum(skipCount=40, FEP_bufSize=20)
process(FEP_buffer)
FileEnum(skipCount=60, FEP_bufSize=20)
process(FEP_buffer)
```

This means that a buffer that only holds 20 files may be used as opposed to a buffer of unknown size which would otherwise be needed to hold return structures for all files in the directory.

Skip count optimization - if the *FESF_REAL_SKIP* bit is set, then this is the actual number of files to skip, matching or not. If *FESF_REAL_SKIP* is clear, *FEP_skipCount* is the number of *matching* files to skip. The real skip count is faster because the match condition does not need to be checked.

With *FESF_REAL_SKIP* set: When **FileEnum** returns after filling in *FEP_bufSize* number of matching entries, *di* will be updated to the real number of files passed through in order to get those *FEP_bufSize* files.

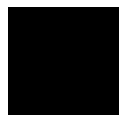
Starting with *di* at 0, **FileEnum** will increment *di* once for each file checked. When **FileEnum** returns, save *di* for the next time **FileEnum** is called.

FEP_callback stores the address of the callback routine to determine if the file should be accepted by **FileEnum**. The callback is performed *after all regularly specified tests have accepted the file*. Therefore, this callback routine is the last step when checking acceptance of the file.

Revision: ■

Draft Dated (4/18/94)

Reference book



Callback Routine Specifications:

Passed:	ds	= segment of FileEnumCallbackData
	bp	= inherited stack frame, which must be passed to any FileEnum helper routines the callback calls.

The callback routine can look at the **FileEnumStructure** passed by using:

```
FooCallback proc far params:FileEnumParams
enter inherit far
```

Return:

- carry clear to accept file
- carry set to reject file

Callback routine should destroy no registers and should not change passed structures. Only relevant if mask FESF_CALLBACK is set.

If FESF_CALLBACK is set and *FEP_callback.segment* is 0, *FEP_callback.offset* is a **FileEnumStandardCallback**. *FEP_callbackAttrs* is ignored in this case, as the system knows what extra attributes are required by each standard callback. See the description of each FESC_* constant to find what should be passed in *FEP_cbData1* and *FEP_cbData2*.

FEP_callbackAttrs specifies an array of attributes the callback routine will need to examine if segment is non-zero, and FESF_CALLBACK is set in *FEP_searchFlags*. **FileEnum** will always pass the callback all attributes given in either the *FEP_returnAttrs* or *FEP_matchAttrs* array. This array is for attributes for which you can't give an exact value (thus they can't be in *FEP_matchAttrs*) and of which you don't actually need to make a record (thus they can't be in *FEP_returnAttrs*).

The last entry in the array has FEA_END_OF_LIST as its *FEAD_attr*. If no additional attributes are required when FESF_CALLBACK is set, *FEP_callbackAttrs.segment* must be zero.

FEP_cbData1 and *FEP_cbData2* allow the caller of **FileEnum** to pass data to the callback routine.

FEP_headerSize stores the amount of space to leave at the start of the return block if FESF_LEAVE_HEADER set.

Library: **fileEnum.def**

■ FileEnumSearchFlags

```
FileEnumSearchFlags      record
    FESF_DIRS             :1      ; accept directories
    FESF_NON_GEOS         :1      ; accept non-GEOS files
    FESF_GEOS_EXECS       :1      ; accept GEOS executables
    FESF_GEOS_NON_EXECS   :1      ; accept GEOS non-executables (data files)
    FESF_REAL_SKIP        :1      ; use FEP_skipCount is real skip count
                                ; (see FileEnum for explanation)
    FESF_CALLBACK         :1      ; use FEP_callback field
    FESF_LOCK_CB_DATA     :1      ; for use in FileEnumPtr only; if set,
                                ; FEP_cbData1 and FEP_cbData2 are assumed
                                ; to be far pointers to movable or fixed
                                ; memory that must be locked before
                                ; FileEnum is called.
    FESF_LEAVE_HEADER     :1      ; if set, then FEP_headerSize indicates
                                ; number of bytes at the beginning of the
                                ; return block that should be left 0 by
                                ; FileEnum, to form a header to be filled
                                ; in by the caller.

FileEnumSearchFlags      end
```

Library: **fileEnum.def**

■ FileEnumStandardCallback

```
FileEnumStandardCallback etype      word, 0
    FESC_WILDCARD enum      FileEnumStandardCallback
                                ;FEP_cbData1 is a far pointer to a null-terminated string
                                ;containing a virtual filename, with the special characters * and ?
                                ;interpreted as meaning 0-or-more-of-any-character and
                                ;any-character, respectively.
                                ;
                                ;Note that the match occurs in the virtual namespace, so "*. *" will
                                ;not match all files, as it will in standard DOS, but rather all
                                ;files that have a . in their virtual name.
                                ;
                                ;FEP_cbData2.low should be non-zero to perform the match in a
                                ;case-insensitive fashion, or zero to be case-sensitive.
```

Library: **fileEnum.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **FileEnumStandardReturnType**

```
FileEnumStandardReturnType    etype word, 0
    FESRT_COUNT_ONLY          enum FileEnumStandardReturnType
    FESRT_DOS_INFO             enum FileEnumStandardReturnType
    FESRT_NAME                 enum FileEnumStandardReturnType
    FESRT_NAME_AND_ATTR        enum FileEnumStandardReturnType
```

Library: **fileEnum.def**

■ FileError

FileError	etype	word
ERROR_UNSUPPORTED_FUNCTION	enum	FileError, 1;MS-DOS error
ERROR_FILE_NOT_FOUND	enum	FileError, 2;MS-DOS error
ERROR_PATH_NOT_FOUND	enum	FileError, 3;MS-DOS error
ERROR_TOO_MANY_OPEN_FILES	enum	FileError, 4;MS-DOS error
ERROR_ACCESS_DENIED	enum	FileError, 5;MS-DOS error
ERROR_INSUFFICIENT_MEMORY	enum	FileError, 8;MS-DOS & FileEnum error
ERROR_INVALID_DRIVE	enum	FileError, 15;MS-DOS error
ERROR_IS_CURRENT_DIRECTORY	enum	FileError, 16;MS-DOS error
ERROR_DIFFERENT_DEVICE	enum	FileError, 17;MS-DOS error
ERROR_NO_MORE_FILES	enum	FileError, 18;MS-DOS error
ERROR_WRITE_PROTECTED	enum	FileError, 19;MS-DOS critical error
ERROR_UNKNOWN_VOLUME	enum	FileError, 20;MS-DOS critical error
ERROR_DRIVE_NOT_READY	enum	FileError, 21;MS-DOS critical error
ERROR_CRC_ERROR	enum	FileError, 23;MS-DOS critical error
ERROR_SEEK_ERROR	enum	FileError, 25;MS-DOS critical error
ERROR_UNKNOWN_MEDIA	enum	FileError, 26;MS-DOS critical error
ERROR_SECTOR_NOT_FOUND	enum	FileError, 27;MS-DOS critical error
ERROR_WRITE_FAULT	enum	FileError, 29;MS-DOS critical error
ERROR_READ_FAULT	enum	FileError, 30;MS-DOS critical error
ERROR_GENERAL_FAILURE	enum	FileError, 31;MS-DOS critical error
ERROR_SHARING_VIOLATION	enum	FileError, 32;
ERROR_ALREADY_LOCKED	enum	FileError, 33;'share.exe' error
ERROR_SHARING_OVERFLOW	enum	FileError, 36;'share.exe' error
ERROR_NETWORK_CONNECTION_BROKEN	enum	FileError, 55
ERROR_NETWORK_ACCESS_DENIED	enum	FileError, 65
ERROR_NETWORK_NOT_LOGGED_IN	enum	FileError, 78
ERROR_SHORT_READ_WRITE	enum	FileError, 128;PC GEOS error
ERROR_INVALID_NAME	enum	FileError, 129;PC GEOS error
ERROR_FILE_EXISTS	enum	FileError, 130
ERROR_DOS_EXEC_IN_PROGRESS	enum	FileError, 131; DosExec
ERROR_FILE_IN_USE	enum	FileError, 132
ERROR_ARGS_TOO_LONG	enum	FileError, 133;DosExec
ERROR_DISK_UNAVAILABLE	enum	FileError, 134;Validation of disk in ;drive aborted by user.
ERROR_DISK_STALE	enum	FileError, 135;Drive disk was on has been ;removed.
ERROR_FILE_FORMAT_MISMATCH	enum	FileError, 136;Attempted to create a file ;with FILE_CREATE_TRUNCATE or ;FILE_CREATE_NO_TRUNCATE and its ;current state doesn't match that ;desired by the FCF_NATIVE flag.
ERROR_CANNOT_MAP_NAME	enum	FileError, 137;file system driver was ;unable to map the virtual 32-char ;name to a suitable name appropriate to ;the file system.
ERROR_DIRECTORY_NOT_EMPTY	enum	FileError, 138;Attempted to delete a ;directory that still contained files.

Revision: ■

Draft Dated (4/18/94)

Reference book



ERROR_ATTR_NOT_SUPPORTED	enum FileError, 139;Requested an extended ;attribute that is not supported by the ;file system or the file.
ERROR_ATTR_NOT_FOUND	enum FileError, 140;Requested an extended ;attribute that is not present for the ;file.
ERROR_ATTR_SIZE_MISMATCH	enum FileError, 141;Requested an attribute ;without providing the correct amount of ;space/data to get/set it.
ERROR_ATTR_CANNOT_BE_SET	enum FileError, 142;Attempted to set an ; extended attribute that cannot be set: ; FEA_SIZE ; FEA_NAME ; FEA_DOS_NAME ; FEA_GEODE_ATTRS ; FEA_PATH_INFO ; FEA_FILE_ID
ERROR_CANNOT_MOVE_DIRECTORY support	enum FileError, 143;file system doesn't ; moving of directories in ; FileMove, and PC/GEOS doesn't ; provide the functionality ; itself.
ERROR_PATH_TOO_LONG	enum FileError, 144;Attempted to create a ;directory that would be unreachable, ; owing to path-length ; restrictions of the file system
ERROR_ARGS_INVALID	enum FileError, 145;DosExec: argument string ; contained a character that ; could not be mapped to the ; current DOS character set.
ERROR_CANNOT_FIND_COMMAND_INTERPRETER	enum FileError, 146 ;DosExec: program to run is a ; batch file, but system was ; unable to locate the command ; interpreter (COMMAND.COM) to ; run the command.
ERROR_NO_TASK_DRIVER_LOADED	enum FileError, 147 ;DosExec: cannot run a DOS program as ; no task-switching driver was loaded.
ERROR_LINK_ENCOUNTED	enum FileError, 148 ; A link was encountered and needs to ; be traversed
ERROR_NOT_A_LINK	enum FileError, 149 ; A link function was called on a file ; that's not a link.
ERROR_TOO_MANY_LINKS	enum FileError, 150 ; A path contains too many links. Most ; likely, one of the elements of the path ; is a link to itself.

Library: **file.def**

■ FileExclude

```
FileExclude      etype byte, 0
FE_COMPAT        enum FileExclude
FE_EXCLUSIVE     enum FileExclude
FE_DENY_WRITE    enum FileExclude
FE_DENY_READ     enum FileExclude
FE_NONE          enum FileExclude
```

Library: **file.def**

■ FileExtAttrDesc

```
FileExtAttrDesc  struct
FEAD_attr        FileExtendedAttribute
FEAD_value       fptr
FEAD_size        word
FEAD_name        fptr.char
FileExtAttrDesc  ends
```

This structure stores a description of extended attributes that should be changed to reflect new values.

FEAD_attr stores the **FileExtendedAttribute** that is to be altered (or *FEA_CUSTOM* to alter a custom attribute). *FEAD_value* stores the pointer to a buffer containing the new value.

FEAD_size stores the size of that buffer.

FEAD_name stores the pointer to a null-terminated ASCII name of an attribute if the attribute is a custom one (*FEA_CUSTOM*).

FileEnum can be passed arrays of **FileExtAttrDesc** structures. In this case, the number of elements in the array is not passed; the last element should have its *FEAD_attr* field set to *FEA_END_OF_LIST*.

Library: **file.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FileExtendedAttribute

```
FileExtendedAttribute      etype word, 0
FEA_MODIFICATION          enum FileExtendedAttribute; FileDateAndTime
FEA_FILE_ATTR              enum FileExtendedAttribute; FileAttrs
FEA_SIZE                   enum FileExtendedAttribute; dword
FEA_FILE_TYPE              enum FileExtendedAttribute; GeosFileType
FEA_FLAGS                  enum FileExtendedAttribute; GeosFileHeaderFlags
FEA_RELEASE                enum FileExtendedAttribute; ReleaseNumber
FEA_PROTOCOL               enum FileExtendedAttribute; ProtocolNumber
FEA_TOKEN                  enum FileExtendedAttribute; GeodeToken
FEA_CREATOR                enum FileExtendedAttribute; GeodeToken
FEA_USER_NOTES              enum FileExtendedAttribute; char array
                                ; FileUserNotes
FEA_NOTICE                 enum FileExtendedAttribute; char array
                                ; FileCopyrightNotice
FEA_CREATION               enum FileExtendedAttribute; FileDateAndTime
FEA_PASSWORD               enum FileExtendedAttribute; char array
                                ; FilePassword
FEA_CUSTOM                 enum FileExtendedAttribute; ?
FEA_NAME                   enum FileExtendedAttribute; char array
                                ; (FileLongName)
FEA_GEODE_ATTR             enum FileExtendedAttribute; GeodeAttrs. a hack
                                ; for FileEnum...
FEA_PATH_INFO              enum FileExtendedAttribute; DirPathInfo. a hack
                                ; for FileEnum...
FEA_FILE_ID                enum FileExtendedAttribute; 32-bit ID of
                                ; file
FEA_DESKTOP_INFO           enum FileExtendedAttribute; FileDesktopInfo
FEA_DRIVE_STATUS           enum FileExtendedAttribute; DriveExtendedStatus
FEA_DISK                   enum FileExtendedAttribute; Disk handle
;
; these next are supported only by some file systems and are intended for
; specialized use (e.g. a desktop program) not for most applications.
;
FEA_DOS_NAME                enum FileExtendedAttribute; actual DOS name of
                                ;the file, if it's on
                                ;a DOS file system.
                                ;8.3. name in DOS
                                ;character set
FEA_OWNER                  enum FileExtendedAttribute; null-terminated name
                                ; of owner of the file.
                                ; FileOwnerName
FEA_RIGHTS                  enum FileExtendedAttribute; null-terminated
                                ; description of access
                                ; rights to the file.
                                ; FileAccessRights
FEA_LAST_VALID              equ FileExtendedAttribute-1
FEA_MULTIPLE                enum FileExtendedAttribute,-2 ; Special value for
```

```

; FileGetExtAttr and
; FileSetExtAttr to
; get/set multiple
; attributes for a
; file.
FEA_END_OF_LIST          enum FileExtendedAttribute,-1; Marker for the last
                           ; entry in an array of
                           ; FileExtAttrDesc
                           ; structures

```

Library: **file.def**

■ FileOpenAndReadFlags

```

FileOpenAndReadFlags      record
    ; These three flags are processed in order:

    FOARF_ADD_CRLF:1
        ; Append a CR/LF to the buffer, unless the buffer already ends
        ; with a CR/LF.

    FOARF_ADD_EOF:1
        ; Append an MSDOS_TEXT_FILE_EOF to the buffer.

    FOARF_NULL_TERMINATE:1
        ; null-terminate the buffer.

    :6
    FOARF_ACCESS FileAccessFlags:7
FileOpenAndReadFlags      end

```

Library: **file.def**

■ FilePathID

```

FilePathID                struct
    FPID_disk              word          ; disk handle
    FPID_id                FileID       ; ID for path on that disk.
FilePathID                ends

```

These structures act as elements of arrays returned by
FileGetCurrentPathIDs.

Library: **file.def**

Revision: ■
Draft Dated (4/18/94)

Reference book



■ FilePosMode

```
FilePosMode          etype byte, 0
    FILE_POS_START    enum FilePosMode
    FILE_POS_RELATIVE enum FilePosMode
    FILE_POS_END       enum FilePosMode
```

Library: **file.def**

■ FileSelectorAttrs

```
FileSelectorAttrs    record
    FSA_ALLOW_CHANGE_DIRS      :1
                                :1
    FSA_SHOW_FIXED_DISKS_ONLY  :1
    FSA_SHOW_FILES_DISABLED    :1
    FSA_HAS_CLOSE_DIR_BUTTON   :1
    FSA_HAS_OPEN_DIR_BUTTON    :1
    FSA_HAS_DOCUMENT_BUTTON    :1
    FSA_HAS_CHANGE_DIRECTORY_LIST :1
    FSA_HAS_CHANGE_DRIVE_LIST  :1
    FSA_HAS_FILE_LIST          :1
    FSA_USE_VIRTUAL_ROOT       :1
                                :5
```

```
FileSelectorAttrs    end
```

FSA_ALLOW_CHANGE_DIRS

Allows changing to different directories. If not set, directories are not opened automatically when the user double-clicks on them. It is up to the application to send MSG_FILE_SELECTOR_OPEN_ENTRY to the GenFileSelector.

FSA_SHOW_FIXED_DISKS_ONLY

Show only fixed disks in the volume existing.

FSA_SHOW_FILES_DISABLED

When showing a file, don't allow the user to select it; useful for Save As operations.

FSA_HAS_CLOSE_DIR_BUTTON

Set if the corresponding gadget.

FSA_HAS_OPEN_DIR_BUTTON

Appear in the file selector

FSA_HAS_DOCUMENT_BUTTON

FSA_HAS_CHANGE_DIRECTORY_LIST

FSA_HAS_CHANGE_DRIVE_LIST

FSA_HAS_FILE_LIST

FSA_USE_VIRTUAL_ROOT

Set if information in
ATTR_GEN_FILE_SELECTOR_VIRTUAL_ROOT should be used
(allows turning on and off 'virtual root' feature without
changing variable data).

Library: **Objects/gFSelC.def**

■ FileSelectorFileCriteria

FileSelectorFileCriteria record

```

;
; Types of files to include in the listing
;
FSFC_DIRS                :1      ; include directories
FSFC_NON_GEOS_FILES      :1      ; include non-GEOS files
FSFC_GEOS_EXECUTABLES    :1      ; include GEOS executables
FSFC_GEOS_NON_EXECUTABLES :1      ; include GEOS non-executables
;
; for files and (if FSFC_USE_MASK_FOR_DIRS is set) directories
;
FSFC_MASK_CASE_SENSITIVE :1
;
; for all files (FSFC_NON_GEOS_FILES and/or FSFC_GEOS_EXECUTABLES
; and/or FSFC_GEOS_NON_EXECUTABLES)
;
FSFC_FILE_FILTER          :1
FSFC_FILTER_IS_C          :1
;
; for GEOS files (FSFC_GEOS_EXECS_FILES and/or
; FSFC_GEOS_NON_EXECUTABLES)
;
FSFC_TOKEN_NO_ID          :1
;
; for directories (FSFC_DIRS)
;
FSFC_USE_MASK_FOR_DIRS    :1
                          :7

```

FileSelectorFileCriteria end

This record defines the file selection criteria of a GenFileSelector. This information is stored in the file selector's *GFSI_fileCriteria* instance field.

FSFC_MASK_CASE_INSENSITIVE

Match files against the mask in a case-insensitive manner.

FSFC_FILE_FILTER

Use the filter routine in addition to evaluating each file accepted by other selection criteria.

Revision: ■

Draft Dated (4/18/94)

Reference book



FSFC_FILTER_IS_C

The filter routine returned by
MSG_GEN_FILE_SELECTOR_GET_FILTER_ROUTINE is written in C and
obeys the Pascal calling convention.

FSFC_TOKEN_NO_ID

Ignore manufacturer ID when comparing tokens (for FSFC_TOKEN_MATCH
and FSFC_CREATOR_MATCH).

FSFC_USE_MASK_FOR_DIRS

Also use ATTR_GEN_FILE_SELECTOR_NAME_MASK attribute for directories.
(This ATTR is normally applied only to files.)

Library: **Objects/gFSelC.def**

■ FileTime

```
FileTime      record
  FT_HOUR      :5,          ; hour (24-hour clock)
  FT_MIN       :6,          ; minute (0-59)
  FT_2SEC      :5,          ; 2-second (0-29 giving 0-58 seconds, even
                           ;seconds only)
FileTime      end
```

Library: **file.def**

■ FindNoteHeader

```
FindNoteHeader struct
  FNH_count    word          ; The number of matching notes.
  FNH_data     label dword
FindNoteHeader ends
```

Library: **pen.def**

■ FloatAsciiToFloatFlags

```
FloatAsciiToFloatFlags record
  :6
  FAF_PUSH_RESULT :1
  FAF_STORE_NUMBER :1
FloatAsciiToFloatFlags end
```

This record is used by the **FloatFloatToAscii** routine. This routine converts
ASCII text into a floating point (FP) number.

FAF_PUSH_RESULT

This flag specifies that the resulting FP number should be pushed onto the FP stack.

FAF_STORE_NUMBER

This flag specifies that the resulting FP number should be stored in the buffer passed in **FloatFloatToAscii**.

Library: **math.def**

■ FloatCtrlInfoStruc

```
FloatCtrlInfoStruc      struc
;
; passed values
;
FCIS_listEntryNum      word
FCIS_fmtToken          word
FCIS_listOD            dword
FCIS_fmtArrayHan       word
FCIS_fmtArraySeg       word
;
; returned values
;
FCIS_fmtParamsHan      word
FloatCtrlInfoStruc     ends
```

FCIS_listEntryNum stores the zero-based position of the format entry to retrieve in the Float Format controller.

FCIS_fmtToken stores the token of the format entry to retrieve in the Float Format controller.

FCIS_listOD stores the optr of the dynamic list within the Float Format controller. This list stores the format entry names.

FCIS_fmtArrayHan stores the VM file handle of the array storing user-defined formats. This handle must be passed to all routines that operate on formats even if they do not directly access this format array (i.e. even if there are no user-defined formats).

FCIS_fmtArraySeg stores the VM block handle of the array storing user-defined formats. This segment must be passed to all routines that operate on formats even if they do not directly access this format array.

FCIS_fmtParamsHan stores the handle to a **FormatParams** structure returned by the routine.

Library: **math.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FloatErrorType

```
FloatErrorType      etype byte, FLOAT_ERROR_CODES_ENUM_START, 1
    FLOAT_POS_INFINITY      enum FloatErrorType
    FLOAT_NEG_INFINITY      enum FloatErrorType
    FLOAT_GEN_ERR           enum FloatErrorType
```

Library: **math.def**

■ FloatExponent

```
FloatExponent      record
    FE_SIGN          :1          ; set if number is negative
    FE_EXPONENT      :15        ; the exponent is biased by 3fff
FloatExponent      end
```

This record defines the high word of a GEOS 80 bit FP number. This high word stores the sign (FE_SIGN) and a 15 bit exponent (FE_EXPONENT).

You can check if the FP number has overflowed or underflowed by checking FE_EXPONENT against FP_NAN.

Library: **math.def**

■ FloatFloatToAsciiData

```
FloatFloatToAsciiData      struct
;
; Fields for caller to set up. All fields must be initialized.
;
FFA_params                  FloatFloatToAsciiParams
;
; Possibly useful information returned by FloatFloatToAscii.
;
FFA_startNumber             word
FFA_decimalPoint            word
FFA_endNumber               word
FFA_numChars                word
FFA_startExponent           word
FFA_bufSize                 word ;internal use only
FFA_savedI                  word ;internal use only
FFA_numSign                 word ;internal use only
FFA_startSigCount           byte ;internal use only
FFA_sigCount                byte ;internal use only
FFA_noMoreSigInfo           byte ;internal use only
FFA_startDecCount           byte ;internal use only
FFA_decCount                byte ;internal use only
FFA_decExponent             word ;internal use only
FFA_curExponent             word ;internal use only
```

```

FFA_useCommas      byte    ;internal use only
FFA_charsToComma   byte    ;internal use only
FFA_commaChar      char    ;internal use only
FFA_decimalChar    char    ;internal use only
FloatFloatToAsciiData  ends

```

This structure contains the **FloatFloatToAsciiParams** and some fields for internal use. This structure exists as a member of the *FFA_stackFrame* union.

FFA_params must be set up by the caller. All fields in that structure must be initialized.

FFA_startNumber returns the offset to the start of numeric characters. (This field is set by **FloatFloatToAscii::FloatDoPreNumeric**.)

FFA_decimalPoint returns the offset to the decimal point (0 if there is no decimal point). (This field is set by **FloatFloatToAscii::StuffDecimalPoint**.)

FFA_endNumber stores the offset to the end of numeric characters. (This field is set by **FloatFloatToAscii::FloatDoPostNumeric**.)

FFA_numChars stores the total number of characters in the ASCII string excluding the null terminator, or 0 if there was an error. (This field is set by **FloatFloatToAscii** in two locations.)

FFA_startExponent stores the offset to the “E” character in an exponentiated numeric value, or 0 if there is no exponent. Applications can check this to see if the exponent format was used.

Library: **math.def**

■ FloatFloatToAsciiFormatFlags

```

FloatFloatToAsciiFormatFlags  record
    FFAF_FLOAT_RESERVED      :1
    FFAF_FROM_ADDR           :1
                                :4
    FFAF_DONT_USE_SCIENTIFIC  :1
    ;boolean bits, phrased so that a 0 will give the default
    FFAF_SCIENTIFIC           :1
    FFAF_PERCENT              :1
    FFAF_USE_COMMAS           :1
    FFAF_NO_TRAIL_ZEROS       :1
    FFAF_NO_LEAD_ZERO         :1

```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

FFAF_HEADER_PRESENT          :1
FFAF_TRAILER_PRESENT         :1
FFAF_SIGN_CHAR_TO_FOLLOW_HEADER :1
FFAF_SIGN_CHAR_TO_PRECEDE_TRAILER :1
FloatFloatToAsciiFormatFlags end

```

FFAF_FLOAT_RESERVED

This flag must be 0 to perform a float-to-ASCII operation (a 1 indicates a time-date operation).

FFAF_FROM_ADDR

This flag is set if the number should be taken from a passed address rather than the top of the FP stack.

FFAF_DONT_USE_SCIENTIFIC

This flag informs **FloatFloatToAscii** to format the number as a fixed point number by padding the number with zeros as necessary. The routine will force scientific anyway if the resulting string exceeds some large limit.

FFAF_SCIENTIFIC

Set if the FP number should always be formatted in scientific notation.

FFAF_PERCENT

Set if the FP number should be formatted as a percentage

FFAF_USE_COMMAS

Set if the format should use commas to demarcate the thousands separators.

FFAF_NO_TRAIL_ZEROS

Set if the format should use trailing zeros to pad the FP number.

FFAF_NO_LEAD_ZERO

Set if a zero should precede the decimal point if the FP number is between -1 and 1.

FFAF_HEADER_PRESENT

This flag indicates that the format includes a header. Setting this flag speeds formatting.

FFAF_TRAILER_PRESENT

This flag indicates that the format includes a trailer. Setting this flag speeds formatting.

FFAF_SIGN_CHAR_TO_FOLLOW_HEADER

This flag indicates the position of the sign character(s).

FFAF_SIGN_CHAR_TO_PRECEDE_TRAILER

This flag indicates the position of the sign character(s).

Library: **math.def**

■ FloatFloatToAsciiParams

```
FloatFloatToAsciiParams  struct
;
;Fields for caller to set up. All fields must be initialized.
;
formatFlags      FloatFloatToAsciiFormatFlags
decimalOffset    byte
totalDigits      byte
decimalLimit     byte
preNegative      char SIGN_STR_LEN+1 dup (?)
postNegative     char SIGN_STR_LEN+1 dup (?)
prePositive      char SIGN_STR_LEN+1 dup (?)
postPositive     char SIGN_STR_LEN+1 dup (?)
;
; HEADER AND TRAILER FOLLOW
; If these aren't present then only the bytes above need be stored
; per format
;
header           char PAD_STR_LEN+1 dup (?)
trailer          char PAD_STR_LEN+1 dup (?)
align            word
FloatFloatToAsciiParams  ends
```

formatFlags stores a record of Boolean bits specifying how the caller wants the string to look.

decimalOffset stores the number of decimal places that the caller want the decimal point to be offset. E.g. Caller may want offset of -6 to display numbers in terms of “millions.”

- DECIMAL_PRECISION <= *decimalOffset* <= DECIMAL_PRECISION.

totalDigits stores the maximum number of digits for the number to contain (integer + decimal portions). The ASCII string is truncated if length(string) > number.

Generally, *totalDigits* <= DECIMAL_PRECISION if
 FFAF_DONT_USE_SCIENTIFIC is not used.
 totalDigits <= MAX_DIGITS_FOR_HUGE_NUMBERS if
 FFAF_DONT_USE_SCIENTIFIC is used.

By the way, a significant digit is a decimal digit derived from the floating point number's mantissa and it may precede or follow a decimal point. The IEEE format is only capable of DECIMAL_PRECISION number of significant digits. If the *totalDigits* is greater than DECIMAL_PRECISION, the excess digits will be 0.

Revision: ■

Draft Dated (4/18/94)

Reference book



decimalLimit stores the maximum number of decimal digits. The number will be rounded to meet this limit. E.g. 345.678 with a *decimalLimit* of 2 will return 345.68 in fixed format and 3.46E+2 in scientific format.

$0 \leq \text{decimalLimit} \leq \text{DECIMAL_PRECISION}$.

preNegative stores the character(s) used to precede a negative number. The string is expected to be null terminated. E.g. for parenthesized negatives *preNegative* may be '('; for arithmetic negatives *preNegative* may be '-'. Set *preNegative* to 0 if no character is desired.

(Note: the +1 in "SIGN_STR_LEN+1" is for the null terminator.)

postNegative stores the character(s) used to terminate a negative number. The string is expected to be null terminated. E.g. for parenthesized negatives, *postNegative* may be ')'. Set *postNegative* to 0 if no character is desired.

(Note: the +1 in "SIGN_STR_LEN+1" is for the null terminator.)

prePositive stores the character(s) used to precede a positive number. The string is expected to be null terminated. E.g. for arithmetic positives *prePositive* may be '+'. Set *prePositive* to 0 if no character is desired.

(Note: the +1 in "SIGN_STR_LEN+1" is for the null terminator.)

postPositive stores the character(s) used to terminate a positive number. The string is expected to be null terminated. Set *postPositive* to 0 if no character is desired.

(Note: the +1 in "SIGN_STR_LEN+1" is for the null terminator.)

header stores the characters that should precede the number. The string is expected to be null terminated. Whether or not this string follows or precedes the sign is determined by FFAF_SIGN_CHAR_TO_FOLLOW_HEADER.

(Note: the +1 in "PAD_STR_LEN+1" is for the null terminator.)

trailer stores the characters that should follow the number. The string is expected to be null terminated. Whether or not this string follows or precedes the sign is determined by FFAF_SIGN_CHAR_TO_PRECEDE_TRAILER.

(Note: the +1 in "PAD_STR_LEN+1" is for the null terminator.)

Library: **math.def**

■ FloatFloatToAsciiParams_Union

```
FloatFloatToAsciiParams  union
    FFAP_FLOAT            FloatFloatToAsciiParams
    FFAP_DATE_TIME       FloatFloatToDateTimeParams
FloatFloatToAsciiParams  end
```

This union is used by **FloatFloatToAscii** to determine if the FP number should be converted into ASCII text or into a date-time.

Library: **math.def**

■ FloatFloatToDateTimeData

```
FloatFloatToDateTimeData  struct
    FFA_dateTimeParams      FloatFloatToDateTimeParams
FloatFloatToDateTimeData  ends
```

FloatFloatToDateTimeData contains the **FloatFloatToDateTimeParams** and (possibly) fields for internal use in the future. This structure exists as a member of the *FFA_stackFrame* union.

Library: **math.def**

■ FloatFloatToDateTimeFlags

```
FloatFloatToDateTimeFlags  record
    ; these first 2 bits must not move
    FFDT_DATE_TIME_OP       :1
    FFDT_FROM_ADDR          :1

    FFDT_FORMAT              :14
FloatFloatToDateTimeFlags  end
```

FFDT_DATE_TIME_OP

This flag must set to indicate to **FloatFloatToAscii** that a date operation is desired.

FFDT_FROM_ADDR

This flag is set if the FP number to convert should be taken from a passed address rather than the top of the FP stack.

FFDT_FORMAT stores the **DateTimeFormat** format to use.

Library: **math.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FloatFloatToDateTimeParams

```
FloatFloatToDateTimeParams    struct
    FFA_dateTimeFlags    FloatFloatToDateTimeFlags
    FFA_year              word
    FFA_month             byte
    FFA_day               byte
    FFA_weekday           byte
    FFA_hours             byte
    FFA_minutes           byte
    FFA_seconds           byte
FloatFloatToDateTimeParams    ends
```

The **FloatFloatToDateTimeParams** portion of **FloatFloatToDateTimeData** needs to be initialized before the call to **FloatFloatToAscii**.

The **FloatFloatToDateTimeParams** is part of the **FloatFloatToDateTimeData** structure which in turn is a member of the *FFA_stackFrame* union.

Library: **math.def**

■ FloatFormatErrors

```
FloatFormatErrors    etype byte, 0
    FLOAT_FORMAT_NO_ERROR          enum FloatFormatErrors
    FLOAT_FORMAT_TOO_MANY_FORMATS enum FloatFormatErrors
    FLOAT_FORMAT_CANNOT_ALLOC      enum FloatFormatErrors
    ;
    ; picked with FORMAT_ID_PREDEF in mind
    ;
    FLOAT_FORMAT_FORMAT_NAME_NOT_FOUND = 7fffh
    FLOAT_FORMAT_PARAMS_MATCH          = TRUE
    FLOAT_FORMAT_PARAMS_DONT_MATCH     = FALSE
```

Library: **math.def**

■ FloatNum

```
FloatNum    struct
    F_mantissa_wd0    word            ; offset 0
    F_mantissa_wd1    word            ; offset 2
    F_mantissa_wd2    word            ; offset 4
    F_mantissa_wd3    word            ; offset 6
    F_exponent        FloatExponent <> ; offset 8
FloatNum    ends
```

This structure defines a GEOS 80 bit floating point number.

Library: **math.def**

■ FloatStackType

```
FloatStackType      etype byte, 0
  FLOAT_STACK_GROW    enum FloatStackType
  FLOAT_STACK_WRAP    enum FloatStackType
  FLOAT_STACK_ERROR    enum FloatStackType

  FLOAT_STACK_DEFAULT_TYPE equ FLOAT_STACK_GROW
```

This value defines the type of FP stack that should be used by the current thread.

FLOAT_STACK_GROW indicates that the FP stack should grow as needed, This is the default.

FLOAT_STACK_WRAP indicates that the FP stack should “wrap around” and overwrite existing numbers on the bottom of the stack as needed.

FLOAT_STACK_ERROR indicates that FP stack should initiate an error whenever it reaches its maximum size.

Library: **math.def**

■ FloatingKeyboardInfo

```
FloatingKeyboardInfo struct
  FKI_defaultPosition word
      ; If set, the keyboard will be moved to the default position before
      ; being brought onscreen
  FKI_sysModal word
      ; If set, the currently focused window is system modal, so the
      ; window needs to be moved to a higher layer
FloatingKeyboardInfo ends
```

Library: **gAppC.def**

■ FontAttrs

```
FontAttrs      record
  FA_USEFUL      FontUseful:1      ;TRUE: “useful” font
  FA_FIXED_WIDTH FontPitch:1      ;TRUE: fixed width
  FA_ORIENT      FontOrientation:1 ;TRUE: landscape orientation
  FA_OUTLINE     FontSource:1      ;TRUE: outline defined
  FA_FAMILY      FontFamily:4      ;font family
FontAttrs      end
```

Library: **font.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FontEnumFlags

```
FontEnumFlags      record
    FEF_ALPHABETIZE :1      ;TRUE: alphabetize list
    FEF_USEFUL      :1      ;TRUE: find "useful" fonts only
    FEF_FIXED_WIDTH :1      ;TRUE: find fixed-width fonts only
    FEF_FAMILY      :1      ;TRUE: match FontFamily
    FEF_STRING      :1      ;TRUE: match string
    FEF_DOWNCASE    :1      ;TRUE: downcase returned strings
    FEF_BITMAPS     :1      ;TRUE: find fonts with bitmaps
    FEF_OUTLINES    :1      ;TRUE: find fonts with outlines.
FontEnumFlags      end
```

Library: **font.def**

■ FontEnumStruct

```
FontEnumStruct      struct
    FES_ID           FontID
    FES_name         char FONT_NAME_LEN dup (?) ; null terminated string
FontEnumStruct      ends
```

This structure is returned by **GrEnumFonts**.

Library: **font.def**

■ FontFamily

```
FontFamily          etype byte
    FF_SERIF         enum FontFamily
    FF_SANS_SERIF    enum FontFamily
    FF_SCRIPT        enum FontFamily
    FF_ORNAMENT      enum FontFamily
    FF_SYMBOL        enum FontFamily
    FF_MONO          enum FontFamily
    FF_SPECIAL       enum FontFamily
    FF_NON_PORTABLE  enum FontFamily
```

Library: **fontID.def**

■ FontGroup

```
FontGroup          etype word, 0, FID_FAMILY_DIVISIONS
  FG_SERIF          enum FontGroup, FF_SERIF          *FID_FAMILY_DIVISIONS
  FG_SANS_SERIF     enum FontGroup, FF_SANS_SERIF      *FID_FAMILY_DIVISIONS
  FG_SCRIPT         enum FontGroup, FF_SCRIPT         *FID_FAMILY_DIVISIONS
  FG_ORNAMENT       enum FontGroup, FF_ORNAMENT       *FID_FAMILY_DIVISIONS
  FG_SYMBOL         enum FontGroup, FF_SYMBOL         *FID_FAMILY_DIVISIONS
  FG_MONO          enum FontGroup, FF_MONO           *FID_FAMILY_DIVISIONS
  FG_SPECIAL        enum FontGroup, FF_SPECIAL        *FID_FAMILY_DIVISIONS
  FG_NON_PORTABLE  enum FontGroup, FF_NON_PORTABLE    *FID_FAMILY_DIVISIONS
```

Library: **fontID.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FontID

FontID	etype	word	
FID_INVALID			enum FontID, 0x0000; invalid font ID
FID_PRINTER_PROP_SANS			enum FontID, 0xf200
FID_PRINTER_PROP_SERIF			enum FontID, 0xf000
FID_BITSTREAM_LETTER_GOTHIC			enum FontID, 0x3a03
FID_PS_LETTER_GOTHIC			enum FontID, 0x2a03
FID_DTC_LETTER_GOTHIC			enum FontID, 0x1a03
FID_BITSTREAM_PRESTIGE_ELITE			enum FontID, 0x3a02
FID_PS_PRESTIGE_ELITE			enum FontID, 0x2a02
FID_DTC_PRESTIGE_ELITE			enum FontID, 0x1a02
FID_BITSTREAM_AMERICAN_TYPEWRITER			enum FontID, 0x3a01
FID_PS_AMERICAN_TYPEWRITER			enum FontID, 0x2a01
FID_DTC_AMERICAN_TYPEWRITER			enum FontID, 0x1a01
FID_BITSTREAM_URW_MONO			enum FontID, 0x3a00
FID_PS_COURIER			enum FontID, 0x2a00
FID_DTC_URW_MONO			enum FontID, 0x1a00
FID_BITSTREAM_FUN_DINGBATS			enum FontID, 0x380d
FID_PS_FUN_DINGBATS			enum FontID, 0x280d
FID_DTC_FUN_DINGBATS			enum FontID, 0x180d
FID_BITSTREAM_CHEQ			enum FontID, 0x380c
FID_PS_CHEQ			enum FontID, 0x280c
FID_DTC_CHEQ			enum FontID, 0x180c
FID_BITSTREAM_BUNDESBahn_PI_3			enum FontID, 0x380b
FID_PS_BUNDESBahn_PI_3			enum FontID, 0x280b
FID_DTC_BUNDESBahn_PI_3			enum FontID, 0x180b
FID_BITSTREAM_BUNDESBahn_PI_2			enum FontID, 0x380a
FID_PS_BUNDESBahn_PI_2			enum FontID, 0x280a
FID_DTC_BUNDESBahn_PI_2			enum FontID, 0x180a
FID_BITSTREAM_BUNDESBahn_PI_1			enum FontID, 0x3809
FID_PS_BUNDESBahn_PI_1			enum FontID, 0x2809
FID_DTC_BUNDESBahn_PI_1			enum FontID, 0x1809
FID_BITSTREAM_U_GREEK_MATH_PI			enum FontID, 0x3808
FID_PS_U_GREEK_MATH_PI			enum FontID, 0x2808
FID_DTC_U_GREEK_MATH_PI			enum FontID, 0x1808
FID_BITSTREAM_U_NEWS_COMM_PI			enum FontID, 0x3807
FID_PS_U_NEWS_COMM_PI			enum FontID, 0x2807
FID_DTC_U_NEWS_COMM_PI			enum FontID, 0x1807
FID_BITSTREAM_ACE_I			enum FontID, 0x3806
FID_PS_ACE_I			enum FontID, 0x2806
FID_DTC_ACE_I			enum FontID, 0x1806
FID_BITSTREAM_SONATA			enum FontID, 0x3805
FID_PS_SONATA			enum FontID, 0x2805
FID_DTC_SONATA			enum FontID, 0x1805
FID_BITSTREAM_CARTA			enum FontID, 0x3804
FID_PS_CARTA			enum FontID, 0x2804
FID_DTC_CARTA			enum FontID, 0x1804
FID_BITSTREAM_MICR			enum FontID, 0x3803
FID_PS_MICR			enum FontID, 0x2803

Revision: ■

Draft Dated (4/18/94)

FID_DTC_MICR	enum FontID, 0x1803
FID_BITSTREAM_ZAPF_DINGBATS	enum FontID, 0x3802
FID_PS_ZAPF_DINGBATS	enum FontID, 0x2802
FID_DTC_ZAPF_DINGBATS	enum FontID, 0x1802
FID_BITSTREAM_DINGBATS	enum FontID, 0x3801
FID_PS_DINGBATS	enum FontID, 0x2801
FID_DTC_DINGBATS	enum FontID, 0x1801
FID_BITSTREAM_URW_SYMBOLPS	enum FontID, 0x3800
FID_PS_SYMBOL	enum FontID, 0x2800
FID_DTC_URW_SYMBOLPS	enum FontID, 0x1800
FID_BITSTREAM_JUNIPER	enum FontID, 0x367f
FID_PS_JUNIPER	enum FontID, 0x267f
FID_DTC_JUNIPER	enum FontID, 0x167f
FID_BITSTREAM_COTTONWOOD	enum FontID, 0x367e
FID_PS_COTTONWOOD	enum FontID, 0x267e
FID_DTC_COTTONWOOD	enum FontID, 0x167e
FID_BITSTREAM_BANCO	enum FontID, 0x367d
FID_PS_BANCO	enum FontID, 0x267d
FID_DTC_BANCO	enum FontID, 0x167d
FID_BITSTREAM_ARCADIA	enum FontID, 0x367c
FID_PS_ARCADIA	enum FontID, 0x267c
FID_DTC_ARCADIA	enum FontID, 0x167c
FID_BITSTREAM_ZIPPER	enum FontID, 0x367b
FID_PS_ZIPPER	enum FontID, 0x267b
FID_DTC_ZIPPER	enum FontID, 0x167b
FID_BITSTREAM_WEIFZ_RUNDGOTIFCH	enum FontID, 0x367a
FID_PS_WEIFZ_RUNDGOTIFCH	enum FontID, 0x267a
FID_DTC_WEIFZ_RUNDGOTIFCH	enum FontID, 0x167a
FID_BITSTREAM_WASHINGTON	enum FontID, 0x3679
FID_PS_WASHINGTON	enum FontID, 0x2679
FID_DTC_WASHINGTON	enum FontID, 0x1679
FID_BITSTREAM_VICTORIAN	enum FontID, 0x3678
FID_PS_VICTORIAN	enum FontID, 0x2678
FID_DTC_VICTORIAN	enum FontID, 0x1678
FID_BITSTREAM_VEGAS	enum FontID, 0x3677
FID_PS_VEGAS	enum FontID, 0x2677
FID_DTC_VEGAS	enum FontID, 0x1677
FID_BITSTREAM_VARIO	enum FontID, 0x3676
FID_PS_VARIO	enum FontID, 0x2676
FID_DTC_VARIO	enum FontID, 0x1676
FID_BITSTREAM_VAG_RUNDSCHRIFT	enum FontID, 0x3675
FID_PS_VAG_RUNDSCHRIFT	enum FontID, 0x2675
FID_DTC_VAG_RUNDSCHRIFT	enum FontID, 0x1675
FID_BITSTREAM_TRAJANUS	enum FontID, 0x3674
FID_PS_TRAJANUS	enum FontID, 0x2674
FID_DTC_TRAJANUS	enum FontID, 0x1674
FID_BITSTREAM_TITUS	enum FontID, 0x3673
FID_PS_TITUS	enum FontID, 0x2673
FID_DTC_TITUS	enum FontID, 0x1673
FID_BITSTREAM_TIME_SCRIPT	enum FontID, 0x3672

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_PS_TIME_SCRIPT	enum FontID, 0x2672
FID_DTC_TIME_SCRIPT	enum FontID, 0x1672
FID_BITSTREAM_THUNDERBIRD	enum FontID, 0x3671
FID_PS_THUNDERBIRD	enum FontID, 0x2671
FID_DTC_THUNDERBIRD	enum FontID, 0x1671
FID_BITSTREAM_THOROWGOOD	enum FontID, 0x3670
FID_PS_THOROWGOOD	enum FontID, 0x2670
FID_DTC_THOROWGOOD	enum FontID, 0x1670
FID_BITSTREAM_TARRAGON	enum FontID, 0x366f
FID_PS_TARRAGON	enum FontID, 0x266f
FID_DTC_TARRAGON	enum FontID, 0x166f
FID_BITSTREAM_TANGO	enum FontID, 0x366e
FID_PS_TANGO	enum FontID, 0x266e
FID_DTC_TANGO	enum FontID, 0x166e
FID_BITSTREAM_SYNCHRO	enum FontID, 0x366d
FID_PS_SYNCHRO	enum FontID, 0x266d
FID_DTC_SYNCHRO	enum FontID, 0x166d
FID_BITSTREAM_SUPERSTAR	enum FontID, 0x366c
FID_PS_SUPERSTAR	enum FontID, 0x266c
FID_DTC_SUPERSTAR	enum FontID, 0x166c
FID_BITSTREAM_STOP	enum FontID, 0x366b
FID_PS_STOP	enum FontID, 0x266b
FID_DTC_STOP	enum FontID, 0x166b
FID_BITSTREAM_STILLA_CAPS	enum FontID, 0x366a
FID_PS_STILLA_CAPS	enum FontID, 0x266a
FID_DTC_STILLA_CAPS	enum FontID, 0x166a
FID_BITSTREAM_STILLA	enum FontID, 0x3669
FID_PS_STILLA	enum FontID, 0x2669
FID_DTC_STILLA	enum FontID, 0x1669
FID_BITSTREAM_STENTOR	enum FontID, 0x3668
FID_PS_STENTOR	enum FontID, 0x2668
FID_DTC_STENTOR	enum FontID, 0x1668
FID_BITSTREAM_SQUIRE	enum FontID, 0x3667
FID_PS_SQUIRE	enum FontID, 0x2667
FID_DTC_SQUIRE	enum FontID, 0x1667
FID_BITSTREAM_SPRINGFIELD	enum FontID, 0x3666
FID_PS_SPRINGFIELD	enum FontID, 0x2666
FID_DTC_SPRINGFIELD	enum FontID, 0x1666
FID_BITSTREAM_SLIPSTREAM	enum FontID, 0x3665
FID_PS_SLIPSTREAM	enum FontID, 0x2665
FID_DTC_SLIPSTREAM	enum FontID, 0x1665
FID_BITSTREAM_SINALOA	enum FontID, 0x3664
FID_PS_SINALOA	enum FontID, 0x2664
FID_DTC_SINALOA	enum FontID, 0x1664
FID_BITSTREAM_SHELLEY	enum FontID, 0x3663
FID_PS_SHELLEY	enum FontID, 0x2663
FID_DTC_SHELLEY	enum FontID, 0x1663
FID_BITSTREAM_SERPENTINE	enum FontID, 0x3662
FID_PS_SERPENTINE	enum FontID, 0x2662
FID_DTC_SERPENTINE	enum FontID, 0x1662

FID_BITSTREAM_RUBBER_STAMP	enum FontID, 0x3661
FID_PS_RUBBER_STAMP	enum FontID, 0x2661
FID_DTC_RUBBER_STAMP	enum FontID, 0x1661
FID_BITSTREAM_ROMIC	enum FontID, 0x3660
FID_PS_ROMIC	enum FontID, 0x2660
FID_DTC_ROMIC	enum FontID, 0x1660
FID_BITSTREAM_RIALTO	enum FontID, 0x365f
FID_PS_RIALTO	enum FontID, 0x265f
FID_DTC_RIALTO	enum FontID, 0x165f
FID_BITSTREAM_REVUE	enum FontID, 0x365e
FID_PS_REVUE	enum FontID, 0x265e
FID_DTC_REVUE	enum FontID, 0x165e
FID_BITSTREAM_QUENTIN	enum FontID, 0x365d
FID_PS_QUENTIN	enum FontID, 0x265d
FID_DTC_QUENTIN	enum FontID, 0x165d
FID_BITSTREAM_PRO_ARTE	enum FontID, 0x365c
FID_PS_PRO_ARTE	enum FontID, 0x265c
FID_DTC_PRO_ARTE	enum FontID, 0x165c
FID_BITSTREAM_PRINCETOWN	enum FontID, 0x365b
FID_PS_PRINCETOWN	enum FontID, 0x265b
FID_DTC_PRINCETOWN	enum FontID, 0x165b
FID_BITSTREAM_PRESIDENT	enum FontID, 0x365a
FID_PS_PRESIDENT	enum FontID, 0x265a
FID_DTC_PRESIDENT	enum FontID, 0x165a
FID_BITSTREAM_PREMIER	enum FontID, 0x3659
FID_PS_PREMIER	enum FontID, 0x2659
FID_DTC_PREMIER	enum FontID, 0x1659
FID_BITSTREAM_POST_ANTIQUA	enum FontID, 0x3658
FID_PS_POST_ANTIQUA	enum FontID, 0x2658
FID_DTC_POST_ANTIQUA	enum FontID, 0x1658
FID_BITSTREAM_PLAZA	enum FontID, 0x3657
FID_PS_PLAZA	enum FontID, 0x2657
FID_DTC_PLAZA	enum FontID, 0x1657
FID_BITSTREAM_PLAYBILL	enum FontID, 0x3656
FID_PS_PLAYBILL	enum FontID, 0x2656
FID_DTC_PLAYBILL	enum FontID, 0x1656
FID_BITSTREAM_PICCADILLY	enum FontID, 0x3655
FID_PS_PICCADILLY	enum FontID, 0x2655
FID_DTC_PICCADILLY	enum FontID, 0x1655
FID_BITSTREAM_PEIGNOT	enum FontID, 0x3654
FID_PS_PEIGNOT	enum FontID, 0x2654
FID_DTC_PEIGNOT	enum FontID, 0x1654
FID_BITSTREAM_PAPYRUS	enum FontID, 0x3653
FID_PS_PAPYRUS	enum FontID, 0x2653
FID_DTC_PAPYRUS	enum FontID, 0x1653
FID_BITSTREAM_PADDINGTION	enum FontID, 0x3652
FID_PS_PADDINGTION	enum FontID, 0x2652
FID_DTC_PADDINGTION	enum FontID, 0x1652
FID_BITSTREAM_OKAY	enum FontID, 0x3651
FID_PS_OKAY	enum FontID, 0x2651

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_DTC_OKAY	enum FontID, 0x1651
FID_BITSTREAM_ODIN	enum FontID, 0x3650
FID_PS_ODIN	enum FontID, 0x2650
FID_DTC_ODIN	enum FontID, 0x1650
FID_BITSTREAM_OCTOPUSS	enum FontID, 0x364f
FID_PS_OCTOPUSS	enum FontID, 0x264f
FID_DTC_OCTOPUSS	enum FontID, 0x164f
FID_BITSTREAM_MOTTER_FEMINA	enum FontID, 0x364e
FID_PS_MOTTER_FEMINA	enum FontID, 0x264e
FID_DTC_MOTTER_FEMINA	enum FontID, 0x164e
FID_BITSTREAM_MICROGRAMMA	enum FontID, 0x364d
FID_PS_MICROGRAMMA	enum FontID, 0x264d
FID_DTC_MICROGRAMMA	enum FontID, 0x164d
FID_BITSTREAM_MACHINE	enum FontID, 0x364c
FID_PS_MACHINE	enum FontID, 0x264c
FID_DTC_MACHINE	enum FontID, 0x164c
FID_BITSTREAM_LINOTEXT	enum FontID, 0x364b
FID_PS_LINOTEXT	enum FontID, 0x264b
FID_DTC_LINOTEXT	enum FontID, 0x164b
FID_BITSTREAM_LIBERTY	enum FontID, 0x364a
FID_PS_LIBERTY	enum FontID, 0x264a
FID_DTC_LIBERTY	enum FontID, 0x164a
FID_BITSTREAM_LAZYBONES	enum FontID, 0x3649
FID_PS_LAZYBONES	enum FontID, 0x2649
FID_DTC_LAZYBONES	enum FontID, 0x1649
FID_BITSTREAM_LATIN_WIDE	enum FontID, 0x3648
FID_PS_LATIN_WIDE	enum FontID, 0x2648
FID_DTC_LATIN_WIDE	enum FontID, 0x1648
FID_BITSTREAM_KNIGHTSBRIDGE	enum FontID, 0x3647
FID_PS_KNIGHTSBRIDGE	enum FontID, 0x2647
FID_DTC_KNIGHTSBRIDGE	enum FontID, 0x1647
FID_BITSTREAM_KAPITELLIA	enum FontID, 0x3646
FID_PS_KAPITELLIA	enum FontID, 0x2646
FID_DTC_KAPITELLIA	enum FontID, 0x1646
FID_BITSTREAM_KALLIGRAPHIA	enum FontID, 0x3645
FID_PS_KALLIGRAPHIA	enum FontID, 0x2645
FID_DTC_KALLIGRAPHIA	enum FontID, 0x1645
FID_BITSTREAM_ICE_AGE	enum FontID, 0x3644
FID_PS_ICE_AGE	enum FontID, 0x2644
FID_DTC_ICE_AGE	enum FontID, 0x1644
FID_BITSTREAM_ICONE	enum FontID, 0x3643
FID_PS_ICONE	enum FontID, 0x2643
FID_DTC_ICONE	enum FontID, 0x1643
FID_BITSTREAM_HORNDON	enum FontID, 0x3642
FID_PS_HORNDON	enum FontID, 0x2642
FID_DTC_HORNDON	enum FontID, 0x1642
FID_BITSTREAM_HORATIO	enum FontID, 0x3641
FID_PS_HORATIO	enum FontID, 0x2641
FID_DTC_HORATIO	enum FontID, 0x1641
FID_BITSTREAM_HIGHLIGHT	enum FontID, 0x3640

FID_PS_HIGHLIGHT	enum FontID, 0x2640
FID_DTC_HIGHLIGHT	enum FontID, 0x1640
FID_BITSTREAM_HADFIELD	enum FontID, 0x363f
FID_PS_HADFIELD	enum FontID, 0x263f
FID_DTC_HADFIELD	enum FontID, 0x163f
FID_BITSTREAM_GLASER_STENCIL	enum FontID, 0x363e
FID_PS_GLASER_STENCIL	enum FontID, 0x263e
FID_DTC_GLASER_STENCIL	enum FontID, 0x163e
FID_BITSTREAM_GILL_KAYO	enum FontID, 0x363d
FID_PS_GILL_KAYO	enum FontID, 0x263d
FID_DTC_GILL_KAYO	enum FontID, 0x163d
FID_BITSTREAM_GALADRIEL	enum FontID, 0x363c
FID_PS_GALADRIEL	enum FontID, 0x263c
FID_DTC_GALADRIEL	enum FontID, 0x163c
FID_BITSTREAM_FUTURA_DISPLAY	enum FontID, 0x363b
FID_PS_FUTURA_DISPLAY	enum FontID, 0x263b
FID_DTC_FUTURA_DISPLAY	enum FontID, 0x163b
FID_BITSTREAM_FUTURA_C_BLACK	enum FontID, 0x363a
FID_PS_FUTURA_C_BLACK	enum FontID, 0x263a
FID_DTC_FUTURA_C_BLACK	enum FontID, 0x163a
FID_BITSTREAM_FRANKFURTER	enum FontID, 0x3639
FID_PS_FRANKFURTER	enum FontID, 0x2639
FID_DTC_FRANKFURTER	enum FontID, 0x1639
FID_BITSTREAM_FLORA	enum FontID, 0x3638
FID_PS_FLORA	enum FontID, 0x2638
FID_DTC_FLORA	enum FontID, 0x1638
FID_BITSTREAM_FLANGE	enum FontID, 0x3637
FID_PS_FLANGE	enum FontID, 0x2637
FID_DTC_FLANGE	enum FontID, 0x1637
FID_BITSTREAM_FLASH	enum FontID, 0x3636
FID_PS_FLASH	enum FontID, 0x2636
FID_DTC_FLASH	enum FontID, 0x1636
FID_BITSTREAM_FLAMENCO	enum FontID, 0x3635
FID_PS_FLAMENCO	enum FontID, 0x2635
FID_DTC_FLAMENCO	enum FontID, 0x1635
FID_BITSTREAM_FETTE_GOTILCH	enum FontID, 0x3634
FID_PS_FETTE_GOTILCH	enum FontID, 0x2634
FID_DTC_FETTE_GOTILCH	enum FontID, 0x1634
FID_BITSTREAM_FETTE_FRAKTUR	enum FontID, 0x3633
FID_PS_FETTE_FRAKTUR	enum FontID, 0x2633
FID_DTC_FETTE_FRAKTUR	enum FontID, 0x1633
FID_BITSTREAM_ENVIRO	enum FontID, 0x3632
FID_PS_ENVIRO	enum FontID, 0x2632
FID_DTC_ENVIRO	enum FontID, 0x1632
FID_BITSTREAM_EINHORN	enum FontID, 0x3631
FID_PS_EINHORN	enum FontID, 0x2631
FID_DTC_EINHORN	enum FontID, 0x1631
FID_BITSTREAM_ECKMANN	enum FontID, 0x3630
FID_PS_ECKMANN	enum FontID, 0x2630
FID_DTC_ECKMANN	enum FontID, 0x1630

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_BITSTREAM_DYNAMO	enum FontID, 0x362f
FID_PS_DYNAMO	enum FontID, 0x262f
FID_DTC_DYNAMO	enum FontID, 0x162f
FID_BITSTREAM_DOM_CASUAL	enum FontID, 0x362e
FID_PS_DOM_CASUAL	enum FontID, 0x262e
FID_DTC_DOM_CASUAL	enum FontID, 0x162e
FID_BITSTREAM_DAVIDA	enum FontID, 0x362d
FID_PS_DAVIDA	enum FontID, 0x262d
FID_DTC_DAVIDA	enum FontID, 0x162d
FID_BITSTREAM_CROISSANT	enum FontID, 0x362c
FID_PS_CROISSANT	enum FontID, 0x262c
FID_DTC_CROISSANT	enum FontID, 0x162c
FID_BITSTREAM_CRILLEE	enum FontID, 0x362b
FID_PS_CRILLEE	enum FontID, 0x262b
FID_DTC_CRILLEE	enum FontID, 0x162b
FID_BITSTREAM_COUNTDOWN	enum FontID, 0x362a
FID_PS_COUNTDOWN	enum FontID, 0x262a
FID_DTC_COUNTDOWN	enum FontID, 0x162a
FID_BITSTREAM_CORTEZ	enum FontID, 0x3629
FID_PS_CORTEZ	enum FontID, 0x2629
FID_DTC_CORTEZ	enum FontID, 0x1629
FID_BITSTREAM_CONFERENCE	enum FontID, 0x3628
FID_PS_CONFERENCE	enum FontID, 0x2628
FID_DTC_CONFERENCE	enum FontID, 0x1628
FID_BITSTREAM_COMPANY	enum FontID, 0x3627
FID_PS_COMPANY	enum FontID, 0x2627
FID_DTC_COMPANY	enum FontID, 0x1627
FID_BITSTREAM_COLUMNNA_SOLID	enum FontID, 0x3626
FID_PS_COLUMNNA_SOLID	enum FontID, 0x2626
FID_DTC_COLUMNNA_SOLID	enum FontID, 0x1626
FID_BITSTREAM_CITY	enum FontID, 0x3625
FID_PS_CITY	enum FontID, 0x2625
FID_DTC_CITY	enum FontID, 0x1625
FID_BITSTREAM_CIRKULUS	enum FontID, 0x3624
FID_PS_CIRKULUS	enum FontID, 0x2624
FID_DTC_CIRKULUS	enum FontID, 0x1624
FID_BITSTREAM_CHURCHWARD_BRUSH	enum FontID, 0x3623
FID_PS_CHURCHWARD_BRUSH	enum FontID, 0x2623
FID_DTC_CHURCHWARD_BRUSH	enum FontID, 0x1623
FID_BITSTREAM_CHROMIUM_ONE	enum FontID, 0x3622
FID_PS_CHROMIUM_ONE	enum FontID, 0x2622
FID_DTC_CHROMIUM_ONE	enum FontID, 0x1622
FID_BITSTREAM_CHOC	enum FontID, 0x3621
FID_PS_CHOC	enum FontID, 0x2621
FID_DTC_CHOC	enum FontID, 0x1621
FID_BITSTREAM_CHISEL	enum FontID, 0x3620
FID_PS_CHISEL	enum FontID, 0x2620
FID_DTC_CHISEL	enum FontID, 0x1620
FID_BITSTREAM_CHESTERFIELD	enum FontID, 0x361f
FID_PS_CHESTERFIELD	enum FontID, 0x261f

FID_DTC_CHESTERFIELD	enum FontID, 0x161f
FID_BITSTREAM_CAROUSEL	enum FontID, 0x361e
FID_PS_CAROUSEL	enum FontID, 0x261e
FID_DTC_CAROUSEL	enum FontID, 0x161e
FID_BITSTREAM_CAMELLIA	enum FontID, 0x361d
FID_PS_CAMELLIA	enum FontID, 0x261d
FID_DTC_CAMELLIA	enum FontID, 0x161d
FID_BITSTREAM_CABARET	enum FontID, 0x361c
FID_PS_CABARET	enum FontID, 0x261c
FID_DTC_CABARET	enum FontID, 0x161c
FID_BITSTREAM_BUXOM	enum FontID, 0x361b
FID_PS_BUXOM	enum FontID, 0x261b
FID_DTC_BUXOM	enum FontID, 0x161b
FID_BITSTREAM_BUSTER	enum FontID, 0x361a
FID_PS_BUSTER	enum FontID, 0x261a
FID_DTC_BUSTER	enum FontID, 0x161a
FID_BITSTREAM_BOTTLENECK	enum FontID, 0x3619
FID_PS_BOTTLENECK	enum FontID, 0x2619
FID_DTC_BOTTLENECK	enum FontID, 0x1619
FID_BITSTREAM_BLOCK	enum FontID, 0x3618
FID_PS_BLOCK	enum FontID, 0x2618
FID_DTC_BLOCK	enum FontID, 0x1618
FID_BITSTREAM_BINNER	enum FontID, 0x3617
FID_PS_BINNER	enum FontID, 0x2617
FID_DTC_BINNER	enum FontID, 0x1617
FID_BITSTREAM_BERNHARD_ANTIQU	enum FontID, 0x3616
FID_PS_BERNHARD_ANTIQU	enum FontID, 0x2616
FID_DTC_BERNHARD_ANTIQU	enum FontID, 0x1616
FID_BITSTREAM_BELSHAW	enum FontID, 0x3615
FID_PS_BELSHAW	enum FontID, 0x2615
FID_DTC_BELSHAW	enum FontID, 0x1615
FID_BITSTREAM_BARCELONA	enum FontID, 0x3614
FID_PS_BARCELONA	enum FontID, 0x2614
FID_DTC_BARCELONA	enum FontID, 0x1614
FID_BITSTREAM_BAUHAUS	enum FontID, 0x3613
FID_PS_BAUHAUS	enum FontID, 0x2613
FID_DTC_BAUHAUS	enum FontID, 0x1613
FID_BITSTREAM_AUGUSTEA_OPEN	enum FontID, 0x3612
FID_PS_AUGUSTEA_OPEN	enum FontID, 0x2612
FID_DTC_AUGUSTEA_OPEN	enum FontID, 0x1612
FID_BITSTREAM_AMERICAN_UNCIAL	enum FontID, 0x3611
FID_PS_AMERICAN_UNCIAL	enum FontID, 0x2611
FID_DTC_AMERICAN_UNCIAL	enum FontID, 0x1611
FID_BITSTREAM_ULTE_SCHWABACHER	enum FontID, 0x3610
FID_PS_ULTE_SCHWABACHER	enum FontID, 0x2610
FID_DTC_ULTE_SCHWABACHER	enum FontID, 0x1610
FID_BITSTREAM_ARNOLD_BOCKLIN	enum FontID, 0x360f
FID_PS_ARNOLD_BOCKLIN	enum FontID, 0x260f
FID_DTC_ARNOLD_BOCKLIN	enum FontID, 0x160f
FID_BITSTREAM_ALGERIAN	enum FontID, 0x360e

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_PS_ALGERIAN	enum FontID, 0x260e
FID_DTC_ALGERIAN	enum FontID, 0x160e
FID_BITSTREAM_PUMP	enum FontID, 0x360d
FID_PS_PUMP	enum FontID, 0x260d
FID_DTC_PUMP	enum FontID, 0x160d
FID_BITSTREAM_MARIAGE	enum FontID, 0x360c
FID_PS_MARIAGE	enum FontID, 0x260c
FID_DTC_MARIAGE	enum FontID, 0x160c
FID_BITSTREAM_OLD_TOWN	enum FontID, 0x360b
FID_PS_OLD_TOWN	enum FontID, 0x260b
FID_DTC_OLD_TOWN	enum FontID, 0x160b
FID_BITSTREAM_HOBO	enum FontID, 0x360a
FID_PS_HOBO	enum FontID, 0x260a
FID_DTC_HOBO	enum FontID, 0x160a
FID_BITSTREAM_GOUDY_HEAVYFACE	enum FontID, 0x3609
FID_PS_GOUDY_HEAVYFACE	enum FontID, 0x2609
FID_DTC_GOUDY_HEAVYFACE	enum FontID, 0x1609
FID_BITSTREAM_DATA_70	enum FontID, 0x3608
FID_PS_DATA_70	enum FontID, 0x2608
FID_DTC_DATA_70	enum FontID, 0x1608
FID_BITSTREAM_LCD	enum FontID, 0x3607
FID_PS_LCD	enum FontID, 0x2607

FID_DTC_LCD	enum FontID, 0x1607
FID_BITSTREAM_BALLOON	enum FontID, 0x3606
FID_PS_BALLOON	enum FontID, 0x2606
FID_DTC_BALLOON	enum FontID, 0x1606
FID_BITSTREAM_BLIPO_C_BLACK	enum FontID, 0x3605
FID_PS_BLIPO_C_BLACK	enum FontID, 0x2605
FID_DTC_BLIPO_C_BLACK	enum FontID, 0x1605
FID_BITSTREAM_COOPER_C_BLACK	enum FontID, 0x3604
FID_PS_COOPER_C_BLACK	enum FontID, 0x2604
FID_DTC_COOPER_C_BLACK	enum FontID, 0x1604
FID_BITSTREAM_COPPERPLATE	enum FontID, 0x3603
FID_PS_COPPERPLATE	enum FontID, 0x2603
FID_DTC_COPPERPLATE	enum FontID, 0x1603
FID_BITSTREAM_STENCIL	enum FontID, 0x3602
FID_PS_STENCIL	enum FontID, 0x2602
FID_DTC_STENCIL	enum FontID, 0x1602
FID_BITSTREAM_OLD_ENGLISH	enum FontID, 0x3601
FID_PS_OLD_ENGLISH	enum FontID, 0x2601
FID_DTC_OLD_ENGLISH	enum FontID, 0x1601
FID_BITSTREAM_BROADWAY	enum FontID, 0x3600
FID_PS_BROADWAY	enum FontID, 0x2600
FID_DTC_BROADWAY	enum FontID, 0x1600
FID_BITSTREAM_NUPITAL_SCRIPT	enum FontID, 0x3430
FID_PS_NUPITAL_SCRIPT	enum FontID, 0x2430
FID_DTC_NUPITAL_SCRIPT	enum FontID, 0x1430
FID_BITSTREAM_MEDICI_SCRIPT	enum FontID, 0x342f
FID_PS_MEDICI_SCRIPT	enum FontID, 0x242f
FID_DTC_MEDICI_SCRIPT	enum FontID, 0x142f
FID_BITSTREAM_CHARME	enum FontID, 0x342e
FID_PS_CHARME	enum FontID, 0x242e
FID_DTC_CHARME	enum FontID, 0x142e
FID_BITSTREAM_CASCADE_SCRIPT	enum FontID, 0x342d
FID_PS_CASCADE_SCRIPT	enum FontID, 0x242d
FID_DTC_CASCADE_SCRIPT	enum FontID, 0x142d
FID_BITSTREAM_LITHOS	enum FontID, 0x342c
FID_PS_LITHOS	enum FontID, 0x242c
FID_DTC_LITHOS	enum FontID, 0x142c
FID_BITSTREAM_TEKTON	enum FontID, 0x342b
FID_PS_TEKTON	enum FontID, 0x242b
FID_DTC_TEKTON	enum FontID, 0x142b
FID_BITSTREAM_VLADIMIR_SCRIPT	enum FontID, 0x342a
FID_PS_VLADIMIR_SCRIPT	enum FontID, 0x242a
FID_DTC_VLADIMIR_SCRIPT	enum FontID, 0x142a
FID_BITSTREAM_VAN_DIJK	enum FontID, 0x3429
FID_PS_VAN_DIJK	enum FontID, 0x2429
FID_DTC_VAN_DIJK	enum FontID, 0x1429
FID_BITSTREAM_SLOGAN	enum FontID, 0x3428
FID_PS_SLOGAN	enum FontID, 0x2428
FID_DTC_SLOGAN	enum FontID, 0x1428
FID_BITSTREAM_SHAMROCK	enum FontID, 0x3427

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_PS_SHAMROCK	enum FontID, 0x2427
FID_DTC_SHAMROCK	enum FontID, 0x1427
FID_BITSTREAM_ROMAN_SCRIPT	enum FontID, 0x3426
FID_PS_ROMAN_SCRIPT	enum FontID, 0x2426
FID_DTC_ROMAN_SCRIPT	enum FontID, 0x1426
FID_BITSTREAM_RAGE	enum FontID, 0x3425
FID_PS_RAGE	enum FontID, 0x2425
FID_DTC_RAGE	enum FontID, 0x1425
FID_BITSTREAM_PRESENT_SCRIPT	enum FontID, 0x3424
FID_PS_PRESENT_SCRIPT	enum FontID, 0x2424
FID_DTC_PRESENT_SCRIPT	enum FontID, 0x1424
FID_BITSTREAM_PHYLLIS_INITIALS	enum FontID, 0x3423
FID_PS_PHYLLIS_INITIALS	enum FontID, 0x2423
FID_DTC_PHYLLIS_INITIALS	enum FontID, 0x1423
FID_BITSTREAM_PHYLLIS	enum FontID, 0x3422
FID_PS_PHYLLIS	enum FontID, 0x2422
FID_DTC_PHYLLIS	enum FontID, 0x1422
FID_BITSTREAM_PEPITA	enum FontID, 0x3421
FID_PS_PEPITA	enum FontID, 0x2421
FID_DTC_PEPITA	enum FontID, 0x1421
FID_BITSTREAM_PENDRY_SCRIPT	enum FontID, 0x3420
FID_PS_PENDRY_SCRIPT	enum FontID, 0x2420
FID_DTC_PENDRY_SCRIPT	enum FontID, 0x1420
FID_BITSTREAM_PALETTE	enum FontID, 0x341f
FID_PS_PALETTE	enum FontID, 0x241f
FID_DTC_PALETTE	enum FontID, 0x141f
FID_BITSTREAM_PALACE_SCRIPT	enum FontID, 0x341e
FID_PS_PALACE_SCRIPT	enum FontID, 0x241e
FID_DTC_PALACE_SCRIPT	enum FontID, 0x141e
FID_BITSTREAM_NEVISON_CASUAL	enum FontID, 0x341d
FID_PS_NEVISON_CASUAL	enum FontID, 0x241d
FID_DTC_NEVISON_CASUAL	enum FontID, 0x141d
FID_BITSTREAM_HILL	enum FontID, 0x341c
FID_PS_HILL	enum FontID, 0x241c
FID_DTC_HILL	enum FontID, 0x141c
FID_BITSTREAM_LINOSCRIP	enum FontID, 0x341b
FID_PS_LINOSCRIP	enum FontID, 0x241b
FID_DTC_LINOSCRIP	enum FontID, 0x141b
FID_BITSTREAM_LINDSAY	enum FontID, 0x341a
FID_PS_LINDSAY	enum FontID, 0x241a
FID_DTC_LINDSAY	enum FontID, 0x141a
FID_BITSTREAM_LE_GRIFFE	enum FontID, 0x3419
FID_PS_LE_GRIFFE	enum FontID, 0x2419
FID_DTC_LE_GRIFFE	enum FontID, 0x1419
FID_BITSTREAM_KUNSTLERSCHREIBSCHRIFT	enum FontID, 0x3418
FID_PS_KUNSTLERSCHREIBSCHRIFT	enum FontID, 0x2418
FID_DTC_KUNSTLERSCHREIBSCHRIFT	enum FontID, 0x1418
FID_BITSTREAM_JULIA_SCRIPT	enum FontID, 0x3417
FID_PS_JULIA_SCRIPT	enum FontID, 0x2417
FID_DTC_JULIA_SCRIPT	enum FontID, 0x1417

FID_BITSTREAM_ISBELL	enum FontID, 0x3416
FID_PS_ISBELL	enum FontID, 0x2416
FID_DTC_ISBELL	enum FontID, 0x1416
FID_BITSTREAM_ISADORA	enum FontID, 0x3415
FID_PS_ISADORA	enum FontID, 0x2415
FID_DTC_ISADORA	enum FontID, 0x1415
FID_BITSTREAM_HOGARTH_SCRIPT	enum FontID, 0x3414
FID_PS_HOGARTH_SCRIPT	enum FontID, 0x2414
FID_DTC_HOGARTH_SCRIPT	enum FontID, 0x1414
FID_BITSTREAM_HARLOW	enum FontID, 0x3413
FID_PS_HARLOW	enum FontID, 0x2413
FID_DTC_HARLOW	enum FontID, 0x1413
FID_BITSTREAM_GLASTONBURY	enum FontID, 0x3412
FID_PS_GLASTONBURY	enum FontID, 0x2412
FID_DTC_GLASTONBURY	enum FontID, 0x1412
FID_BITSTREAM_GILLIES_GOTHIC	enum FontID, 0x3411
FID_PS_GILLIES_GOTHIC	enum FontID, 0x2411
FID_DTC_GILLIES_GOTHIC	enum FontID, 0x1411
FID_BITSTREAM_FREESTYLE_SCRIPT	enum FontID, 0x3410
FID_PS_FREESTYLE_SCRIPT	enum FontID, 0x2410
FID_DTC_FREESTYLE_SCRIPT	enum FontID, 0x1410
FID_BITSTREAM_ENGLISCHE_SCHREIBSCHRIFT	enum FontID, 0x340f
FID_PS_ENGLISCHE_SCHREIBSCHRIFT	enum FontID, 0x240f
FID_DTC_ENGLISCHE_SCHREIBSCHRIFT	enum FontID, 0x140f
FID_BITSTREAM_DEMIAN	enum FontID, 0x340e
FID_PS_DEMIAN	enum FontID, 0x240e
FID_DTC_DEMIAN	enum FontID, 0x140e
FID_BITSTREAM_CANDICE	enum FontID, 0x340d
FID_PS_CANDICE	enum FontID, 0x240d
FID_DTC_CANDICE	enum FontID, 0x140d
FID_BITSTREAM_BRONX	enum FontID, 0x340c
FID_PS_BRONX	enum FontID, 0x240c
FID_DTC_BRONX	enum FontID, 0x140c
FID_BITSTREAM_BRODY	enum FontID, 0x340b
FID_PS_BRODY	enum FontID, 0x240b
FID_DTC_BRODY	enum FontID, 0x140b
FID_BITSTREAM_BIBLE_SCRIPT	enum FontID, 0x340a
FID_PS_BIBLE_SCRIPT	enum FontID, 0x240a
FID_DTC_BIBLE_SCRIPT	enum FontID, 0x140a
FID_BITSTREAM_ARISTON	enum FontID, 0x3409
FID_PS_ARISTON	enum FontID, 0x2409
FID_DTC_ARISTON	enum FontID, 0x1409
FID_BITSTREAM_ANGLIA	enum FontID, 0x3408
FID_PS_ANGLIA	enum FontID, 0x2408
FID_DTC_ANGLIA	enum FontID, 0x1408
FID_BITSTREAM_MISTRAL	enum FontID, 0x3407
FID_PS_MISTRAL	enum FontID, 0x2407
FID_DTC_MISTRAL	enum FontID, 0x1407
FID_BITSTREAM_BALMORAL	enum FontID, 0x3406
FID_PS_BALMORAL	enum FontID, 0x2406

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_DTC_BALMORAL	enum FontID, 0x1406
FID_BITSTREAM_COMMERCIAL_SCRIPT	enum FontID, 0x3405
FID_PS_COMMERCIAL_SCRIPT	enum FontID, 0x2405
FID_DTC_COMMERCIAL_SCRIPT	enum FontID, 0x1405
FID_BITSTREAM_KAUFMANN	enum FontID, 0x3404
FID_PS_KAUFMANN	enum FontID, 0x2404
FID_DTC_KAUFMANN	enum FontID, 0x1404
FID_BITSTREAM_PARK_AVENUE	enum FontID, 0x3403
FID_PS_PARK_AVENUE	enum FontID, 0x2403
FID_DTC_PARK_AVENUE	enum FontID, 0x1403
FID_BITSTREAM_BRUSH_SCRIPT	enum FontID, 0x3402
FID_PS_BRUSH_SCRIPT	enum FontID, 0x2402
FID_DTC_BRUSH_SCRIPT	enum FontID, 0x1402
FID_BITSTREAM_VIVALDI	enum FontID, 0x3401
FID_PS_VIVALDI	enum FontID, 0x2401
FID_DTC_VIVALDI	enum FontID, 0x1401
FID_BITSTREAM_ZAPF_CHANCERY	enum FontID, 0x3400
FID_PS_ZAPF_CHANCERY	enum FontID, 0x2400
FID_DTC_ZAPF_CHANCERY	enum FontID, 0x1400
FID_BITSTREAM_AVANTE_GARDE_CONDENSED	enum FontID, 0x323d
FID_PS_AVANTE_GARDE_CONDENSED	enum FontID, 0x223d
FID_DTC_AVANTE_GARDE_CONDENSED	enum FontID, 0x123d
FID_BITSTREAM_INSIGNIA	enum FontID, 0x323c
FID_PS_INSIGNIA	enum FontID, 0x223c
FID_DTC_INSIGNIA	enum FontID, 0x123c
FID_BITSTREAM_INDUSTRIA	enum FontID, 0x323b
FID_PS_INDUSTRIA	enum FontID, 0x223b
FID_DTC_INDUSTRIA	enum FontID, 0x123b
FID_BITSTREAM_DORIC_BOLD	enum FontID, 0x323a
FID_PS_DORIC_BOLD	enum FontID, 0x223a
FID_DTC_DORIC_BOLD	enum FontID, 0x123a
FID_BITSTREAM_AKZINDENZ_GROTESK	enum FontID, 0x3239
FID_PS_AKZINDENZ_GROTESK	enum FontID, 0x2239
FID_DTC_AKZINDENZ_GROTESK	enum FontID, 0x1239
FID_BITSTREAM_GROTESK	enum FontID, 0x3238
FID_PS_GROTESK	enum FontID, 0x2238
FID_DTC_GROTESK	enum FontID, 0x1238
FID_BITSTREAM_TEMPO	enum FontID, 0x3237
FID_PS_TEMPO	enum FontID, 0x2237
FID_DTC_TEMPO	enum FontID, 0x1237
FID_BITSTREAM_SYNTAX	enum FontID, 0x3236
FID_PS_SYNTAX	enum FontID, 0x2236
FID_DTC_SYNTAX	enum FontID, 0x1236
FID_BITSTREAM_STONE_SANS	enum FontID, 0x3235
FID_PS_STONE_SANS	enum FontID, 0x2235
FID_DTC_STONE_SANS	enum FontID, 0x1235
FID_BITSTREAM_SERIF_GOTHIC	enum FontID, 0x3234
FID_PS_SERIF_GOTHIC	enum FontID, 0x2234
FID_DTC_SERIF_GOTHIC	enum FontID, 0x1234
FID_BITSTREAM_PRIMUS_ANTIQUA	enum FontID, 0x3233

Revision: ■

Draft Dated (4/18/94)

FID_PS_PRIMUS_ANTIQUA	enum FontID, 0x2233
FID_DTC_PRIMUS_ANTIQUA	enum FontID, 0x1233
FID_BITSTREAM_PRIMUS	enum FontID, 0x3232
FID_PS_PRIMUS	enum FontID, 0x2232
FID_DTC_PRIMUS	enum FontID, 0x1232
FID_BITSTREAM_PRAXIS	enum FontID, 0x3231
FID_PS_PRAXIS	enum FontID, 0x2231
FID_DTC_PRAXIS	enum FontID, 0x1231
FID_BITSTREAM_PANACHE	enum FontID, 0x3230
FID_PS_PANACHE	enum FontID, 0x2230
FID_DTC_PANACHE	enum FontID, 0x1230
FID_BITSTREAM_OCR_B	enum FontID, 0x322f
FID_PS_OCR_B	enum FontID, 0x222f
FID_DTC_OCR_B	enum FontID, 0x122f
FID_BITSTREAM_OCR_A	enum FontID, 0x322e
FID_PS_OCR_A	enum FontID, 0x222e
FID_DTC_OCR_A	enum FontID, 0x122e
FID_BITSTREAM_NEWTEXT	enum FontID, 0x322d
FID_PS_NEWTEXT	enum FontID, 0x222d
FID_DTC_NEWTEXT	enum FontID, 0x122d
FID_BITSTREAM_NEWS_GOTHIC	enum FontID, 0x322c
FID_PS_NEWS_GOTHIC	enum FontID, 0x222c
FID_DTC_NEWS_GOTHIC	enum FontID, 0x122c
FID_BITSTREAM_NEUZEIT_GROTESK	enum FontID, 0x322b
FID_PS_NEUZEIT_GROTESK	enum FontID, 0x222b
FID_DTC_NEUZEIT_GROTESK	enum FontID, 0x122b
FID_BITSTREAM_MIXAGE	enum FontID, 0x322a
FID_PS_MIXAGE	enum FontID, 0x222a
FID_DTC_MIXAGE	enum FontID, 0x122a
FID_BITSTREAM_MAXIMA	enum FontID, 0x3229
FID_PS_MAXIMA	enum FontID, 0x2229
FID_DTC_MAXIMA	enum FontID, 0x1229
FID_BITSTREAM_LUCIDA_SANS	enum FontID, 0x3228
FID_PS_LUCIDA_SANS	enum FontID, 0x2228
FID_DTC_LUCIDA_SANS	enum FontID, 0x1228
FID_BITSTREAM_LITERA	enum FontID, 0x3227
FID_PS_LITERA	enum FontID, 0x2227
FID_DTC_LITERA	enum FontID, 0x1227
FID_BITSTREAM_KABEL	enum FontID, 0x3226
FID_PS_KABEL	enum FontID, 0x2226
FID_DTC_KABEL	enum FontID, 0x1226
FID_BITSTREAM_HOLSATIA	enum FontID, 0x3225
FID_PS_HOLSATIA	enum FontID, 0x2225
FID_DTC_HOLSATIA	enum FontID, 0x1225
FID_BITSTREAM_HELVETICA_INSERTAT	enum FontID, 0x3224
FID_PS_HELVETICA_INSERTAT	enum FontID, 0x2224
FID_DTC_HELVETICA_INSERTAT	enum FontID, 0x1224
FID_BITSTREAM_NEUE_HELVETICA	enum FontID, 0x3223
FID_PS_NEUE_HELVETICA	enum FontID, 0x2223
FID_DTC_NEUE_HELVETICA	enum FontID, 0x1223

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_BITSTREAM_HELVETICA	enum FontID, 0x3222
FID_PS_HELVETICA	enum FontID, 0x2222
FID_DTC_HELVETICA	enum FontID, 0x1222
FID_BITSTREAM_HAAS_UNICA	enum FontID, 0x3221
FID_PS_HAAS_UNICA	enum FontID, 0x2221
FID_DTC_HAAS_UNICA	enum FontID, 0x1221
FID_BITSTREAM_GOUDY_SANS	enum FontID, 0x3220
FID_PS_GOUDY_SANS	enum FontID, 0x2220
FID_DTC_GOUDY_SANS	enum FontID, 0x1220
FID_BITSTREAM_GOTHIC	enum FontID, 0x321f
FID_PS_GOTHIC	enum FontID, 0x221f
FID_DTC_GOTHIC	enum FontID, 0x121f
FID_BITSTREAM_GILL_SANS	enum FontID, 0x321e
FID_PS_GILL_SANS	enum FontID, 0x221e
FID_DTC_GILL_SANS	enum FontID, 0x121e
FID_BITSTREAM_GILL	enum FontID, 0x321d
FID_PS_GILL	enum FontID, 0x221d
FID_DTC_GILL	enum FontID, 0x121d
FID_BITSTREAM_FUTURA	enum FontID, 0x321c
FID_PS_FUTURA	enum FontID, 0x221c
FID_DTC_FUTURA	enum FontID, 0x121c
FID_BITSTREAM_FOLIO	enum FontID, 0x321b
FID_PS_FOLIO	enum FontID, 0x221b
FID_DTC_FOLIO	enum FontID, 0x121b
FID_BITSTREAM_FLYER	enum FontID, 0x321a
FID_PS_FLYER	enum FontID, 0x221a
FID_DTC_FLYER	enum FontID, 0x121a
FID_BITSTREAM_FETTE_MIDSCHRIFT	enum FontID, 0x3219
FID_PS_FETTE_MIDSCHRIFT	enum FontID, 0x2219
FID_DTC_FETTE_MIDSCHRIFT	enum FontID, 0x1219
FID_BITSTREAM_FETTE_ENGSCHRIFT	enum FontID, 0x3218
FID_PS_FETTE_ENGSCHRIFT	enum FontID, 0x2218
FID_DTC_FETTE_ENGSCHRIFT	enum FontID, 0x1218
FID_BITSTREAM_ERAS	enum FontID, 0x3217
FID_PS_ERAS	enum FontID, 0x2217
FID_DTC_ERAS	enum FontID, 0x1217
FID_BITSTREAM_DIGI_GROTESK	enum FontID, 0x3216
FID_PS_DIGI_GROTESK	enum FontID, 0x2216
FID_DTC_DIGI_GROTESK	enum FontID, 0x1216
FID_BITSTREAM_CORINTHIAN	enum FontID, 0x3215
FID_PS_CORINTHIAN	enum FontID, 0x2215
FID_DTC_CORINTHIAN	enum FontID, 0x1215
FID_BITSTREAM_COMPACTA	enum FontID, 0x3214
FID_PS_COMPACTA	enum FontID, 0x2214
FID_DTC_COMPACTA	enum FontID, 0x1214
FID_BITSTREAM_CLEARFACE_GOTHIC	enum FontID, 0x3213
FID_PS_CLEARFACE_GOTHIC	enum FontID, 0x2213
FID_DTC_CLEARFACE_GOTHIC	enum FontID, 0x1213
FID_BITSTREAM_OPTIMA	enum FontID, 0x3212
FID_PS_OPTIMA	enum FontID, 0x2212



FID_DTC_OPTIMA	enum FontID, 0x1212
FID_BITSTREAM_CHELMSFORD	enum FontID, 0x3211
FID_PS_CHELMSFORD	enum FontID, 0x2211
FID_DTC_CHELMSFORD	enum FontID, 0x1211
FID_BITSTREAM_CASTLE	enum FontID, 0x3210
FID_PS_CASTLE	enum FontID, 0x2210
FID_DTC_CASTLE	enum FontID, 0x1210
FID_BITSTREAM_BRITANNIC	enum FontID, 0x320f
FID_PS_BRITANNIC	enum FontID, 0x220f
FID_DTC_BRITANNIC	enum FontID, 0x120f
FID_BITSTREAM_BERLINER_GROTESK	enum FontID, 0x320e
FID_PS_BERLINER_GROTESK	enum FontID, 0x220e
FID_DTC_BERLINER_GROTESK	enum FontID, 0x120e
FID_BITSTREAM_BENGUIAT_GOTHIC	enum FontID, 0x320d
FID_PS_BENGUIAT_GOTHIC	enum FontID, 0x220d
FID_DTC_BENGUIAT_GOTHIC	enum FontID, 0x120d
FID_BITSTREAM_AVANTE_GARDE	enum FontID, 0x320c
FID_PS_AVANTE_GARDE	enum FontID, 0x220c
FID_DTC_AVANTE_GARDE	enum FontID, 0x120c
FID_BITSTREAM_ANZEIGEN_GROTESK	enum FontID, 0x320b
FID_PS_ANZEIGEN_GROTESK	enum FontID, 0x220b
FID_DTC_ANZEIGEN_GROTESK	enum FontID, 0x120b
FID_BITSTREAM_ANTIQUOLIVE	enum FontID, 0x320a
FID_PS_ANTIQUOLIVE	enum FontID, 0x220a
FID_DTC_ANTIQUOLIVE	enum FontID, 0x120a
FID_BITSTREAM_ALTERNATE_GOTHIC	enum FontID, 0x3209
FID_PS_ALTERNATE_GOTHIC	enum FontID, 0x2209
FID_DTC_ALTERNATE_GOTHIC	enum FontID, 0x1209
FID_BITSTREAM_AKZIDENZ_GROTESK_BUCH	enum FontID, 0x3208
FID_PS_AKZIDENZ_GROTESK_BUCH	enum FontID, 0x2208
FID_DTC_AKZIDENZ_GROTESK_BUCH	enum FontID, 0x1208
FID_BITSTREAM_AKZIDENZ_GROTESK	enum FontID, 0x3207
FID_PS_AKZIDENZ_GROTESK	enum FontID, 0x2207
FID_DTC_AKZIDENZ_GROTESK	enum FontID, 0x1207
FID_BITSTREAM_AVENIR	enum FontID, 0x3206
FID_PS_AVENIR	enum FontID, 0x2206
FID_DTC_AVENIR	enum FontID, 0x1206
FID_BITSTREAM_UNIVERS	enum FontID, 0x3205
FID_PS_UNIVERS	enum FontID, 0x2205
FID_DTC_UNIVERS	enum FontID, 0x1205
FID_BITSTREAM_FRANKLIN_GOTHIC	enum FontID, 0x3204
FID_PS_FRANKLIN_GOTHIC	enum FontID, 0x2204
FID_DTC_FRANKLIN_GOTHIC	enum FontID, 0x1204
FID_BITSTREAM_ANGRO	enum FontID, 0x3203
FID_PS_ANGRO	enum FontID, 0x2203
FID_DTC_ANGRO	enum FontID, 0x1203
FID_BITSTREAM_EUROSTILE	enum FontID, 0x3202
FID_PS_EUROSTILE	enum FontID, 0x2202
FID_DTC_EUROSTILE	enum FontID, 0x1202
FID_BITSTREAM_FRUTIGER	enum FontID, 0x3201

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_PS_FRUTIGER	enum FontID,	0x2201
FID_DTC_FRUTIGER	enum FontID,	0x1201
FID_BITSTREAM_URW_SANS	enum FontID,	0x3200
FID_PS_URW_SANS	enum FontID,	0x2200
FID_DTC_URW_SANS	enum FontID,	0x1200
FID_BITSTREAM_GALLIARD_ROMAN_ITALIC	enum FontID,	0x307e
FID_PS_GALLIARD_ROMAN_ITALIC	enum FontID,	0x207e
FID_DTC_GALLIARD_ROMAN_ITALIC	enum FontID,	0x107e
FID_BITSTREAM_GRANJON	enum FontID,	0x307d
FID_PS_GRANJON	enum FontID,	0x207d
FID_DTC_GRANJON	enum FontID,	0x107d
FID_BITSTREAM_GARTH_GRAPHIC	enum FontID,	0x307c
FID_PS_GARTH_GRAPHIC	enum FontID,	0x207c
FID_DTC_GARTH_GRAPHIC	enum FontID,	0x107c
FID_BITSTREAM_BAUER_BODONI	enum FontID,	0x307b
FID_PS_BAUER_BODONI	enum FontID,	0x207b
FID_DTC_BAUER_BODONI	enum FontID,	0x107b
FID_BITSTREAM_BELWE	enum FontID,	0x307a
FID_PS_BELWE	enum FontID,	0x207a
FID_DTC_BELWE	enum FontID,	0x107a
FID_BITSTREAM_CHARLEMAGNE	enum FontID,	0x3079
FID_PS_CHARLEMAGNE	enum FontID,	0x2079
FID_DTC_CHARLEMAGNE	enum FontID,	0x1079
FID_BITSTREAM_TRAJAN	enum FontID,	0x3078
FID_PS_TRAJAN	enum FontID,	0x2078
FID_DTC_TRAJAN	enum FontID,	0x1078
FID_BITSTREAM_ADOBE_GARAMOND	enum FontID,	0x3077
FID_PS_ADOBE_GARAMOND	enum FontID,	0x2077
FID_DTC_ADOBE_GARAMOND	enum FontID,	0x1077
FID_BITSTREAM_ZAPF_INTERNATIONAL	enum FontID,	0x3076
FID_PS_ZAPF_INTERNATIONAL	enum FontID,	0x2076
FID_DTC_ZAPF_INTERNATIONAL	enum FontID,	0x1076
FID_BITSTREAM_ZAPF_BOOK	enum FontID,	0x3075
FID_PS_ZAPF_BOOK	enum FontID,	0x2075
FID_DTC_ZAPF_BOOK	enum FontID,	0x1075
FID_BITSTREAM_WORCESTER_ROUND	enum FontID,	0x3074
FID_PS_WORCESTER_ROUND	enum FontID,	0x2074
FID_DTC_WORCESTER_ROUND	enum FontID,	0x1074
FID_BITSTREAM_WINDSOR	enum FontID,	0x3073
FID_PS_WINDSOR	enum FontID,	0x2073
FID_DTC_WINDSOR	enum FontID,	0x1073
FID_BITSTREAM_WEISS	enum FontID,	0x3072
FID_PS_WEISS	enum FontID,	0x2072
FID_DTC_WEISS	enum FontID,	0x1072
FID_BITSTREAM_WEIDEMANN	enum FontID,	0x3071
FID_PS_WEIDEMANN	enum FontID,	0x2071
FID_DTC_WEIDEMANN	enum FontID,	0x1071
FID_BITSTREAM_WALBAUM	enum FontID,	0x3070
FID_PS_WALBAUM	enum FontID,	0x2070
FID_DTC_WALBAUM	enum FontID,	0x1070

Revision: ■

Draft Dated (4/18/94)

FID_BITSTREAM_VOLTA	enum FontID, 0x306f
FID_PS_VOLTA	enum FontID, 0x206f
FID_DTC_VOLTA	enum FontID, 0x106f
FID_BITSTREAM_VENDOME	enum FontID, 0x306e
FID_PS_VENDOME	enum FontID, 0x206e
FID_DTC_VENDOME	enum FontID, 0x106e
FID_BITSTREAM_VELJOVIC	enum FontID, 0x306d
FID_PS_VELJOVIC	enum FontID, 0x206d
FID_DTC_VELJOVIC	enum FontID, 0x106d
FID_BITSTREAM_ADOBE_UTOPIA	enum FontID, 0x306c
FID_PS_ADOBE_UTOPIA	enum FontID, 0x206c
FID_DTC_ADOBE_UTOPIA	enum FontID, 0x106c
FID_BITSTREAM_USHERWOOD	enum FontID, 0x306b
FID_PS_USHERWOOD	enum FontID, 0x206b
FID_DTC_USHERWOOD	enum FontID, 0x106b
FID_BITSTREAM_URW_ANTIQUA	enum FontID, 0x306a
FID_PS_URW_ANTIQUA	enum FontID, 0x206a
FID_DTC_URW_ANTIQUA	enum FontID, 0x106a
FID_BITSTREAM_TIMES_NEW_ROMAN	enum FontID, 0x3069
FID_PS_TIMES_NEW_ROMAN	enum FontID, 0x2069
FID_DTC_TIMES_NEW_ROMAN	enum FontID, 0x1069
FID_BITSTREAM_TIMELESS	enum FontID, 0x3068
FID_PS_TIMELESS	enum FontID, 0x2068
FID_DTC_TIMELESS	enum FontID, 0x1068
FID_BITSTREAM TIFFANY	enum FontID, 0x3067
FID_PS TIFFANY	enum FontID, 0x2067
FID_DTC TIFFANY	enum FontID, 0x1067
FID_BITSTREAM_TIEPOLO	enum FontID, 0x3066
FID_PS_TIEPOLO	enum FontID, 0x2066
FID_DTC_TIEPOLO	enum FontID, 0x1066
FID_BITSTREAM_SWIFT	enum FontID, 0x3065
FID_PS_SWIFT	enum FontID, 0x2065
FID_DTC_SWIFT	enum FontID, 0x1065
FID_BITSTREAM_STYMIE	enum FontID, 0x3064
FID_PS_STYMIE	enum FontID, 0x2064
FID_DTC_STYMIE	enum FontID, 0x1064
FID_BITSTREAM_STRATFORD	enum FontID, 0x3063
FID_PS_STRATFORD	enum FontID, 0x2063
FID_DTC_STRATFORD	enum FontID, 0x1063
FID_BITSTREAM_STONE_SERIF	enum FontID, 0x3062
FID_PS_STONE_SERIF	enum FontID, 0x2062
FID_DTC_STONE_SERIF	enum FontID, 0x1062
FID_BITSTREAM_STONE_INFORMAL	enum FontID, 0x3061
FID_PS_STONE_INFORMAL	enum FontID, 0x2061
FID_DTC_STONE_INFORMAL	enum FontID, 0x1061
FID_BITSTREAM_STEMPEL_SCHNEIDLER	enum FontID, 0x3060
FID_PS_STEMPEL_SCHNEIDLER	enum FontID, 0x2060
FID_DTC_STEMPEL_SCHNEIDLER	enum FontID, 0x1060
FID_BITSTREAM_SOUVENIR	enum FontID, 0x305f
FID_PS_SOUVENIR	enum FontID, 0x205f

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_DTC_SOUVENIR	enum FontID, 0x105f
FID_BITSTREAM_SLIMBACH	enum FontID, 0x305e
FID_PS_SLIMBACH	enum FontID, 0x205e
FID_DTC_SLIMBACH	enum FontID, 0x105e
FID_BITSTREAM_SERIFA	enum FontID, 0x305d
FID_PS_SERIFA	enum FontID, 0x205d
FID_DTC_SERIFA	enum FontID, 0x105d
FID_BITSTREAM_SABON_ANTIQUA	enum FontID, 0x305c
FID_PS_SABON_ANTIQUA	enum FontID, 0x205c
FID_DTC_SABON_ANTIQUA	enum FontID, 0x105c
FID_BITSTREAM_SABON	enum FontID, 0x305b
FID_PS_SABON	enum FontID, 0x205b
FID_DTC_SABON	enum FontID, 0x105b
FID_BITSTREAM_ROMANA	enum FontID, 0x305a
FID_PS_ROMANA	enum FontID, 0x205a
FID_DTC_ROMANA	enum FontID, 0x105a
FID_BITSTREAM_ROCKWELL	enum FontID, 0x3059
FID_PS_ROCKWELL	enum FontID, 0x2059
FID_DTC_ROCKWELL	enum FontID, 0x1059
FID_BITSTREAM_RENAULT	enum FontID, 0x3058
FID_PS_RENAULT	enum FontID, 0x2058
FID_DTC_RENAULT	enum FontID, 0x1058
FID_BITSTREAM_RALEIGH	enum FontID, 0x3057
FID_PS_RALEIGH	enum FontID, 0x2057
FID_DTC_RALEIGH	enum FontID, 0x1057
FID_BITSTREAM_QUORUM	enum FontID, 0x3056
FID_PS_QUORUM	enum FontID, 0x2056
FID_DTC_QUORUM	enum FontID, 0x1056
FID_BITSTREAM_PROTEUS	enum FontID, 0x3055
FID_PS_PROTEUS	enum FontID, 0x2055
FID_DTC_PROTEUS	enum FontID, 0x1055
FID_BITSTREAM_PLANTIN	enum FontID, 0x3054
FID_PS_PLANTIN	enum FontID, 0x2054
FID_DTC_PLANTIN	enum FontID, 0x1054
FID_BITSTREAM_PERPETUA	enum FontID, 0x3053
FID_PS_PERPETUA	enum FontID, 0x2053
FID_DTC_PERPETUA	enum FontID, 0x1053
FID_BITSTREAM_PACELLA	enum FontID, 0x3052
FID_PS_PACELLA	enum FontID, 0x2052
FID_DTC_PACELLA	enum FontID, 0x1052
FID_BITSTREAM_NOVARESE	enum FontID, 0x3051
FID_PS_NOVARESE	enum FontID, 0x2051
FID_DTC_NOVARESE	enum FontID, 0x1051
FID_BITSTREAM_NIMROD	enum FontID, 0x3050
FID_PS_NIMROD	enum FontID, 0x2050
FID_DTC_NIMROD	enum FontID, 0x1050
FID_BITSTREAM_NIKIS	enum FontID, 0x304f
FID_PS_NIKIS	enum FontID, 0x204f
FID_DTC_NIKIS	enum FontID, 0x104f
FID_BITSTREAM_NAPOLEAN	enum FontID, 0x304e



FID_PS_NAPOLEAN	enum FontID, 0x204e
FID_DTC_NAPOLEAN	enum FontID, 0x104e
FID_BITSTREAM_MODERN_NO_216	enum FontID, 0x304d
FID_PS_MODERN_NO_216	enum FontID, 0x204d
FID_DTC_MODERN_NO_216	enum FontID, 0x104d
FID_BITSTREAM_MODERN	enum FontID, 0x304c
FID_PS_MODERN	enum FontID, 0x204c
FID_DTC_MODERN	enum FontID, 0x104c
FID_BITSTREAM_MINISTER	enum FontID, 0x304b
FID_PS_MINISTER	enum FontID, 0x204b
FID_DTC_MINISTER	enum FontID, 0x104b
FID_BITSTREAM_MESSIDOR	enum FontID, 0x304a
FID_PS_MESSIDOR	enum FontID, 0x204a
FID_DTC_MESSIDOR	enum FontID, 0x104a
FID_BITSTREAM_MERIDIEN	enum FontID, 0x3049
FID_PS_MERIDIEN	enum FontID, 0x2049
FID_DTC_MERIDIEN	enum FontID, 0x1049
FID_BITSTREAM_MEMPHIS	enum FontID, 0x3048
FID_PS_MEMPHIS	enum FontID, 0x2048
FID_DTC_MEMPHIS	enum FontID, 0x1048
FID_BITSTREAM_MELIOR	enum FontID, 0x3047
FID_PS_MELIOR	enum FontID, 0x2047
FID_DTC_MELIOR	enum FontID, 0x1047
FID_BITSTREAM_MARCONI	enum FontID, 0x3046
FID_PS_MARCONI	enum FontID, 0x2046
FID_DTC_MARCONI	enum FontID, 0x1046
FID_BITSTREAM_MAGNUS	enum FontID, 0x3045
FID_PS_MAGNUS	enum FontID, 0x2045
FID_DTC_MAGNUS	enum FontID, 0x1045
FID_BITSTREAM_MAGNA	enum FontID, 0x3044
FID_PS_MAGNA	enum FontID, 0x2044
FID_DTC_MAGNA	enum FontID, 0x1044
FID_BITSTREAM_MADISON	enum FontID, 0x3043
FID_PS_MADISON	enum FontID, 0x2043
FID_DTC_MADISON	enum FontID, 0x1043
FID_BITSTREAM_LUCIDA	enum FontID, 0x3042
FID_PS_LUCIDA	enum FontID, 0x2042
FID_DTC_LUCIDA	enum FontID, 0x1042
FID_BITSTREAM_LUBALIN_GRAPH	enum FontID, 0x3041
FID_PS_LUBALIN_GRAPH	enum FontID, 0x2041
FID_DTC_LUBALIN_GRAPH	enum FontID, 0x1041
FID_BITSTREAM_LIFE	enum FontID, 0x3040
FID_PS_LIFE	enum FontID, 0x2040
FID_DTC_LIFE	enum FontID, 0x1040
FID_BITSTREAM_LEAWOOD	enum FontID, 0x303f
FID_PS_LEAWOOD	enum FontID, 0x203f
FID_DTC_LEAWOOD	enum FontID, 0x103f
FID_BITSTREAM_KORINNA	enum FontID, 0x303e
FID_PS_KORINNA	enum FontID, 0x203e
FID_DTC_KORINNA	enum FontID, 0x103e

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_BITSTREAM_JENSON_OLD_STYLE	enum FontID, 0x303d
FID_PS_JENSON_OLD_STYLE	enum FontID, 0x203d
FID_DTC_JENSON_OLD_STYLE	enum FontID, 0x103d
FID_BITSTREAM_JANSON	enum FontID, 0x303c
FID_PS_JANSON	enum FontID, 0x203c
FID_DTC_JANSON	enum FontID, 0x103c
FID_BITSTREAM_JAMILLE	enum FontID, 0x303b
FID_PS_JAMILLE	enum FontID, 0x203b
FID_DTC_JAMILLE	enum FontID, 0x103b
FID_BITSTREAM_ITALIA	enum FontID, 0x303a
FID_PS_ITALIA	enum FontID, 0x203a
FID_DTC_ITALIA	enum FontID, 0x103a
FID_BITSTREAM_IMPRESSUM	enum FontID, 0x3039
FID_PS_IMPRESSUM	enum FontID, 0x2039
FID_DTC_IMPRESSUM	enum FontID, 0x1039
FID_BITSTREAM_HOLLANDER	enum FontID, 0x3038
FID_PS_HOLLANDER	enum FontID, 0x2038
FID_DTC_HOLLANDER	enum FontID, 0x1038
FID_BITSTREAM_HIROSHIGE	enum FontID, 0x3037
FID_PS_HIROSHIGE	enum FontID, 0x2037
FID_DTC_HIROSHIGE	enum FontID, 0x1037
FID_BITSTREAM_HAWTHORN	enum FontID, 0x3036
FID_PS_HAWTHORN	enum FontID, 0x2036
FID_DTC_HAWTHORN	enum FontID, 0x1036
FID_BITSTREAM_GOUDY	enum FontID, 0x3035
FID_PS_GOUDY	enum FontID, 0x2035
FID_DTC_GOUDY	enum FontID, 0x1035
FID_BITSTREAM_GAMMA	enum FontID, 0x3034
FID_PS_GAMMA	enum FontID, 0x2034
FID_DTC_GAMMA	enum FontID, 0x1034
FID_BITSTREAM_GALLIARD	enum FontID, 0x3033
FID_PS_GALLIARD	enum FontID, 0x2033
FID_DTC_GALLIARD	enum FontID, 0x1033
FID_BITSTREAM_FRIZ_QUADRATA	enum FontID, 0x3032
FID_PS_FRIZ_QUADRATA	enum FontID, 0x2032
FID_DTC_FRIZ_QUADRATA	enum FontID, 0x1032
FID_BITSTREAM_FENICE	enum FontID, 0x3031
FID_PS_FENICE	enum FontID, 0x2031
FID_DTC_FENICE	enum FontID, 0x1031
FID_BITSTREAM_EXCELSIOR	enum FontID, 0x3030
FID_PS_EXCELSIOR	enum FontID, 0x2030
FID_DTC_EXCELSIOR	enum FontID, 0x1030
FID_BITSTREAM_ESPRIT	enum FontID, 0x302f
FID_PS_ESPRIT	enum FontID, 0x202f
FID_DTC_ESPRIT	enum FontID, 0x102f
FID_BITSTREAM_ELAN	enum FontID, 0x302e
FID_PS_ELAN	enum FontID, 0x202e
FID_DTC_ELAN	enum FontID, 0x102e
FID_BITSTREAM_EGYPTIENNE	enum FontID, 0x302d
FID_PS_EGYPTIENNE	enum FontID, 0x202d

FID_DTC_EGYPTIENNE	enum FontID, 0x102d
FID_BITSTREAM_EGIZIO	enum FontID, 0x302c
FID_PS_EGIZIO	enum FontID, 0x202c
FID_DTC_EGIZIO	enum FontID, 0x102c
FID_BITSTREAM_EDWARDIAN	enum FontID, 0x302b
FID_PS_EDWARDIAN	enum FontID, 0x202b
FID_DTC_EDWARDIAN	enum FontID, 0x102b
FID_BITSTREAM_EDISON	enum FontID, 0x302a
FID_PS_EDISON	enum FontID, 0x202a
FID_DTC_EDISON	enum FontID, 0x102a
FID_BITSTREAM_DIGI_ANTIQUA	enum FontID, 0x3029
FID_PS_DIGI_ANTIQUA	enum FontID, 0x2029
FID_DTC_DIGI_ANTIQUA	enum FontID, 0x1029
FID_BITSTREAM_DEMOS	enum FontID, 0x3028
FID_PS_DEMOS	enum FontID, 0x2028
FID_DTC_DEMOS	enum FontID, 0x1028
FID_BITSTREAM_CUSHING	enum FontID, 0x3027
FID_PS_CUSHING	enum FontID, 0x2027
FID_DTC_CUSHING	enum FontID, 0x1027
FID_BITSTREAM_CORONA	enum FontID, 0x3026
FID_PS_CORONA	enum FontID, 0x2026
FID_DTC_CORONA	enum FontID, 0x1026
FID_BITSTREAM_CONGRESS	enum FontID, 0x3025
FID_PS_CONGRESS	enum FontID, 0x2025
FID_DTC_CONGRESS	enum FontID, 0x1025
FID_BITSTREAM_CONCORDE_NOVA	enum FontID, 0x3024
FID_PS_CONCORDE_NOVA	enum FontID, 0x2024
FID_DTC_CONCORDE_NOVA	enum FontID, 0x1024
FID_BITSTREAM_CONCORDE	enum FontID, 0x3023
FID_PS_CONCORDE	enum FontID, 0x2023
FID_DTC_CONCORDE	enum FontID, 0x1023
FID_BITSTREAM_CLEARFACE	enum FontID, 0x3022
FID_PS_CLEARFACE	enum FontID, 0x2022
FID_DTC_CLEARFACE	enum FontID, 0x1022
FID_BITSTREAM_CLARENDON	enum FontID, 0x3021
FID_PS_CLARENDON	enum FontID, 0x2021
FID_DTC_CLARENDON	enum FontID, 0x1021
FID_BITSTREAM_CHELTENHAM	enum FontID, 0x3020
FID_PS_CHELTENHAM	enum FontID, 0x2020
FID_DTC_CHELTENHAM	enum FontID, 0x1020
FID_BITSTREAM_CENTURY_OLD_STYLE	enum FontID, 0x301f
FID_PS_CENTURY_OLD_STYLE	enum FontID, 0x201f
FID_DTC_CENTURY_OLD_STYLE	enum FontID, 0x101f
FID_BITSTREAM_CENTURY	enum FontID, 0x301e
FID_PS_CENTURY	enum FontID, 0x201e
FID_DTC_CENTURY	enum FontID, 0x101e
FID_BITSTREAM_CENTENNIAL	enum FontID, 0x301d
FID_PS_CENTENNIAL	enum FontID, 0x201d
FID_DTC_CENTENNIAL	enum FontID, 0x101d
FID_BITSTREAM_CAXTON	enum FontID, 0x301c

Revision: ■

Draft Dated (4/18/94)

Reference book



FID_PS_CAXTON	enum FontID, 0x201c
FID_DTC_CAXTON	enum FontID, 0x101c
FID_BITSTREAM_ADOBE_CASLON	enum FontID, 0x301b
FID_PS_ADOBE_CASLON	enum FontID, 0x201b
FID_DTC_ADOBE_CASLON	enum FontID, 0x101b
FID_BITSTREAM_CASLON	enum FontID, 0x301a
FID_PS_CASLON	enum FontID, 0x201a
FID_DTC_CASLON	enum FontID, 0x101a
FID_BITSTREAM_CANDIDA	enum FontID, 0x3019
FID_PS_CANDIDA	enum FontID, 0x2019
FID_DTC_CANDIDA	enum FontID, 0x1019
FID_BITSTREAM_BOOKMAN	enum FontID, 0x3018
FID_PS_BOOKMAN	enum FontID, 0x2018
FID_DTC_BOOKMAN	enum FontID, 0x1018
FID_BITSTREAM_BASKERVILLE_HANDCUT	enum FontID, 0x3017
FID_PS_BASKERVILLE_HANDCUT	enum FontID, 0x2017
FID_DTC_BASKERVILLE_HANDCUT	enum FontID, 0x1017
FID_BITSTREAM_BASKERVILLE	enum FontID, 0x3016
FID_PS_BASKERVILLE	enum FontID, 0x2016
FID_DTC_BASKERVILLE	enum FontID, 0x1016
FID_BITSTREAM_BASILIA	enum FontID, 0x3015
FID_PS_BASILIA	enum FontID, 0x2015
FID_DTC_BASILIA	enum FontID, 0x1015
FID_BITSTREAM_BARBEDOR	enum FontID, 0x3014
FID_PS_BARBEDOR	enum FontID, 0x2014
FID_DTC_BARBEDOR	enum FontID, 0x1014
FID_BITSTREAM_AUREALIA	enum FontID, 0x3013
FID_PS_AUREALIA	enum FontID, 0x2013
FID_DTC_AUREALIA	enum FontID, 0x1013
FID_BITSTREAM_NEW_ASTER	enum FontID, 0x3012
FID_PS_NEW_ASTER	enum FontID, 0x2012
FID_DTC_NEW_ASTER	enum FontID, 0x1012
FID_BITSTREAM_ASTER	enum FontID, 0x3011
FID_PS_ASTER	enum FontID, 0x2011
FID_DTC_ASTER	enum FontID, 0x1011
FID_BITSTREAM_AMERICANA	enum FontID, 0x3010
FID_PS_AMERICANA	enum FontID, 0x2010
FID_DTC_AMERICANA	enum FontID, 0x1010
FID_BITSTREAM_AACHEN	enum FontID, 0x300f
FID_PS_AACHEN	enum FontID, 0x200f
FID_DTC_AACHEN	enum FontID, 0x100f
FID_BITSTREAM_NICOLAS_COCHIN	enum FontID, 0x300e
FID_PS_NICOLAS_COCHIN	enum FontID, 0x200e
FID_DTC_NICOLAS_COCHIN	enum FontID, 0x100e
FID_BITSTREAM_COCHIN	enum FontID, 0x300d
FID_PS_COCHIN	enum FontID, 0x200d
FID_DTC_COCHIN	enum FontID, 0x100d
FID_BITSTREAM_ALBERTUS	enum FontID, 0x300c
FID_PS_ALBERTUS	enum FontID, 0x200c
FID_DTC_ALBERTUS	enum FontID, 0x100c

FID_BITSTREAM_ACCOLADE	enum FontID, 0x300b
FID_PS_ACCOLADE	enum FontID, 0x200b
FID_DTC_ACCOLADE	enum FontID, 0x100b
FID_BITSTREAM_PALATINO	enum FontID, 0x300a
FID_PS_PALATINO	enum FontID, 0x200a
FID_DTC_PALATINO	enum FontID, 0x100a
FID_BITSTREAM_GOUDY_OLD_STYLE	enum FontID, 0x3009
FID_PS_GOUDY_OLD_STYLE	enum FontID, 0x2009
FID_DTC_GOUDY_OLD_STYLE	enum FontID, 0x1009
FID_BITSTREAM_BERKELEY_OLD_STYLE	enum FontID, 0x3008
FID_PS_BERKELEY_OLD_STYLE	enum FontID, 0x2008
FID_DTC_BERKELEY_OLD_STYLE	enum FontID, 0x1008
FID_BITSTREAM_ARSIS	enum FontID, 0x3007
FID_PS_ARSIS	enum FontID, 0x2007
FID_DTC_ARSIS	enum FontID, 0x1007
FID_BITSTREAM_UNIVERSITY_ROMAN	enum FontID, 0x3006
FID_PS_UNIVERSITY_ROMAN	enum FontID, 0x2006
FID_DTC_UNIVERSITY_ROMAN	enum FontID, 0x1006
FID_BITSTREAM_BEMBO	enum FontID, 0x3005
FID_PS_BEMBO	enum FontID, 0x2005
FID_DTC_BEMBO	enum FontID, 0x1005
FID_BITSTREAM_GARAMOND	enum FontID, 0x3004
FID_PS_GARAMOND	enum FontID, 0x2004
FID_DTC_GARAMOND	enum FontID, 0x1004
FID_BITSTREAM_GLYPHA	enum FontID, 0x3003
FID_PS_GLYPHA	enum FontID, 0x2003
FID_DTC_GLYPHA	enum FontID, 0x1003
FID_BITSTREAM_BODONI	enum FontID, 0x3002
FID_PS_BODONI	enum FontID, 0x2002
FID_DTC_BODONI	enum FontID, 0x1002
FID_BITSTREAM_CENTURY_SCHOOLBOOK	enum FontID, 0x3001
FID_PS_CENTURY_SCHOOLBOOK	enum FontID, 0x2001
FID_DTC_CENTURY_SCHOOLBOOK	enum FontID, 0x1001
FID_BITSTREAM_URW_ROMAN	enum FontID, 0x3000
FID_PS_TIMES_ROMAN	enum FontID, 0x2000
FID_DTC_URW_ROMAN	enum FontID, 0x1000
FID_WINDOWS	enum FontID, 0x0a01
FID_BISON	enum FontID, 0x0a00
FID_LED	enum FontID, 0x0600
FID_PMSYSTEM	enum FontID, 0x0203
FID_BERKELEY	enum FontID, 0x0202
FID_UNIVERSITY	enum FontID, 0x0201
FID_CHICAGO	enum FontID, 0x0200
FID_ROMA	enum FontID, 0x0001

The lower twelve bits for any particular font face are the same. (For example SCHOOLBOOK faces end with 001. The first 4 bits define the particular maker. Thus, each particular face may have up to 16 different makers.

Revision: ■

Draft Dated (4/18/94)

Reference book



The only exceptions to this naming scheme are the printer and bitstream fonts, and FID_INVALID, which is a special case and is set to all zeros.

Library: **fontID.def**

■ FontIDRecord

```
FontIDRecord      record
  FIDR_maker      :4
  FIDR_ID         :12
FontIDRecord      end
```

Library: **font.def**

■ FontMaker

```
FontMaker      etype word, 0, FID_MAKER_DIVISIONS
  FM_BITMAP    enum FontMaker
  FM_NIMBUSQ    enum FontMaker
  FM_ADOBE     enum FontMaker
  FM_BITSTREAM  enum FontMaker
  FM_AGFA      enum FontMaker
  FM_PUBLIC    enum FontMaker, 0xc000
  FM_ATECH     enum FontMaker, 0xd000
  FM_MICROLOGIC enum FontMaker, 0xe000
  FM_PRINTER   enum FontMaker, 0xf000
```

Library: **fontID.def**

■ FontMap

```
FontMap etype      byte, 0
  FM_EXACT      enum FontMap, 0
  FM_DONT_USE   enum FontMap, 0xff
```

Library: **fontID.def**

■ FontOrientation

```
FontOrientation etype byte
  FO_NORMAL      enum FontOrientation; normal straight up & down font
  FO_LANDSCAPE   enum FontOrientation; rotated 90 degrees.
```

Library: **font.def**

■ FontPitch

```
FontPitch          etype byte
FP_PROPORTIONAL    enum FontPitch; proportional font.
FP_FIXED           enum FontPitch; Fixed pitch font.
```

Library: **font.def**

■ FontSource

```
FontSource          etype byte
FS_BITMAP           enum FontSource; bitmap data
FS_OUTLINE          enum FontSource; outline data
```

Library: **font.def**

■ FontUseful

```
FontUseful          etype byte
FU_NOT_USEFUL       enum FontUseful; not useful for menus
FU_USEFUL           enum FontUseful; useful for menus
```

Library: **font.def**

■ FontWeight

```
FontWeight          etype byte
FW_MINIMUM          enum FontWeight, 75
FW_NORMAL           enum FontWeight, 100
FW_MAXIMUM           enum FontWeight, 125
```

Library: **font.def**

■ FontWidth

```
FontWidth           etype byte
FWI_MINIMUM         enum FontWidth, 25
FWI_NARROW          enum FontWidth, 75
FWI_CONDENSED       enum FontWidth, 85
FWI_MEDIUM          enum FontWidth, 100
FWI_WIDE            enum FontWidth, 125
FWI_EXPANDED        enum FontWidth, 150
FWI_MAXIMUM         enum FontWidth, 200
```

This type defines the width of the font as a percentage of its normal width.

Library: **font.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **FormatArrayHeader**

```
FormatArrayHeader  struc
    FAH_signature      word
    FAH_numFormatEntries word      ; format array entries that have
                                ; been allocated (possibly free)

    FAH_numUserDefEntries word
    FAH_formatArrayEnd  word      ;offset to end of format array
FormatArrayHeader  ends
```

Library: **math.def**

■ **FormatEntry**

```
FormatEntry          struc
    FE_params          FormatParams
    FE_listEntryNumber word      ; list entry number
                                ; This will allow us to get the right
                                ; entry given the list entry
    FE_used             byte      ; boolean, 0 if entry is free
    FE_sig              word      ; signature for EC purposes
FormatEntry          ends
```

Library: **math.def**

■ FormatError

FormatError	etype	word
FMT_DONE	enum	FormatError, 0
FMT_READY	enum	FormatError
FMT_RUNNING	enum	FormatError
FMT_DRIVE_NOT_READY	enum	FormatError
FMT_ERR_WRITING_BOOT	enum	FormatError
FMT_ERR_WRITING_ROOT_DIR	enum	FormatError
FMT_ERR_WRITING_FAT	enum	FormatError
FMT_ABORTED	enum	FormatError
FMT_SET_VOLUME_NAME_ERR	enum	FormatError
FMT_CANNOT_FORMAT_FIXED_DISKS_IN_CUR_RELEASE	enum	FormatError
FMT_BAD_PARTITION_TABLE	enum	FormatError ;fixed disk
FMT_ERR_READING_PARTITION_TABLE	enum	FormatError ;fixed disk
FMT_ERR_NO_PARTITION_FOUND	enum	FormatError ;fixed disk
FMT_ERR_MULTIPLE_PRIMARY_PARTITIONS	enum	FormatError ;fixed disk
FMT_ERR_NO_EXTENDED_PARTITION_FOUND	enum	FormatError ;fixed disk
FMT_ERR_CANNOT_ALLOC_SECTOR_BUFFER	enum	FormatError
FMT_ERR_DISK_IS_IN_USE	enum	FormatError
FMT_ERR_WRITE_PROTECTED	enum	FormatError
FMT_ERR_DRIVE_CANNOT_SUPPORT_GIVEN_FORMAT	enum	FormatError
FMT_ERR_INVALID_DRIVE_SPECIFIED	enum	FormatError
FMT_ERR_DRIVE_CANNOT_BE_FORMATTED	enum	FormatError
FMT_ERR_DISK_UNAVAILABLE	enum	FormatError
FMT_ERR_CANNOT_FORMAT_TRACK	enum	FormatError ; catch-all

Library: **disk.def**

■ FormatInfoStruc

FormatInfoStruc	struc
FIS_signature	word
FIS_userDefFmtArrayFileHan	word
FIS_userDefFmtArrayBlkHan	word
FIS_childBlk	word
FIS_chooseFmtListChunk	word
FIS_features	FFCFeatures
FIS_editFlag	byte
FIS_curSelection	word
FIS_curToken	word
FIS_curParams	FormatParams ;
FormatInfoStruc	ends

This structure is passed as a block to the Float Format create and edit code.
This structure is used by the Float Format controller to specify the specific format to act on.

Revision: ■

Draft Dated (4/18/94)

Reference book



When passed to routines, *FIS_userDefFmtArrayFileHan* and *FIS_userDefFmtArrayBlkHan* must be properly set up to hold the VM file and block handles of the user-defined format array.

FIS_signature stores internal signatures used by error-checking code.

FIS_userDefFmtArrayFileHan stores the VM file handle of the user-defined format array. This must be properly set up even if no user-defined formats are to be used.

FIS_userDefFmtArrayBlkHan stores the VM block handle of the user-defined format array. This must be properly set up even if no user-defined formats are to be used.

FIS_childBlk and *FIS_chooseFmtListChunk* store the optr to the dynamic list object within the Float Format controller.

FIS_features stores the features list of the Float Format controller.

FIS_editFlag stores a non-zero value if the format entry is currently being used. If this entry is later freed, this edit flag is set to zero to indicate that this entry is available for other formats.

FIS_curSelection stores the current selection of the Float Format controller's dynamic list.

FIS_curToken stores the current **FormatIdType** of the format entry. In many cases, this token is passed in this structure and *FIS_curParams* is filled in with the matching **FormatParams**.

FIS_curParams stores the current **FormatParams** of the format entry.

Library: **math.def**

■ FormatNameParams

```
FormatNameParams    struct
  FNP_listEntry      word                ; the entry number in the defined
                                          ; list
  FNP_textLength      word                ; length of the format name
  FNP_text            byte FORMAT_NAME_LENGTH dup (?)
  FNP_token           word                ; the token of the format
  align              word
FormatNameParams    ends
```

Library: **math.def**

■ FormatOption

```
FormatOption      record
                  :2
    FO_COMMA      :1
    FO_PCT        :1
    FO_LEAD_ZERO  :1
    FO_TRAIL_ZERO :1
    FO_HEADER_SIGN_POS :1
    FO_TRAILER_SIGN_POS :1
FormatOption      end
```

Library: **math.def**

■ FormatParams

```
FormatParams      struc
    FP_params      FloatFloatToAsciiParams_Union
    FP_formatName  char FORMAT_NAME_LENGTH+1 dup (?)
    FP_nameHan     word
    FP_nameOff     word
    FP_listEntryNum word
    FP_signature   word ; internal
FormatParams      ends
```

This structure stores the formatting parameters used by the Float Format controller to format FP numbers into text.

FP_params stores the **FloatFloatToAscii** parameters to use when formatting the FP number into text.

FP_formatName stores the text name of the format entry to display in the Float Format controller's dynamic list. This text is stored at the optr defined by *FP_nameHan* and *FP_nameOff*.

FP_nameHan and *FP_nameOff* store the optr to the text strings where the format names are kept.

FP_listEntryNumber stores the zero-based position of the format entry within the dynamic list.

Library: **parse.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ FormatParameters

```
FormatParameters    struct
  FP_common    CommonParameters <>
  FP_nChars    word           ; Number of bytes left in the buffer.
FormatParameters    ends
```

Library: **parse.def**

■ FRSPFlags

```
FRSPFlags            record
                                :14
  FRSPF_ADD_DRIVE_NAME    :1
  FRSPF_RETURN_FIRST_DIR  :1
FRSPFlags            end
```

FRSPF_ADD_DRIVE_NAME

Set if **FileResolveStandardPath** should prepend name of the drive in which the file or directory was found to the path.

FRSPF_RETURN_FIRST_DIR

Set if **FileResolveStandardPath** should not check to see whether the passed path actually exists, but instead assume it exists in the first existing directory along the standard path.

Library: **file.def**

■ FTVMCGrab

```
FTVMCGrab            struct
  FVMC_OD            optr
  FVMC_flags          MetaAlterFTVMCExclFlags
FTVMCGrab            ends
```

This structure is a variation on the basic **MetaAlterFTVMCExclFlags** record, adding the optr of the Focus/Target/Model hierarchical grab.

Library: **uiInputC.def**

■ FunctionID

FunctionID	etype word, 0, 2
FUNCTION_ID_ABS	enum FunctionID
FUNCTION_ID_ACOS	enum FunctionID
FUNCTION_ID_ACOSH	enum FunctionID
FUNCTION_ID_AND	enum FunctionID
FUNCTION_ID_ASIN	enum FunctionID
FUNCTION_ID_ASINH	enum FunctionID
FUNCTION_ID_ATAN	enum FunctionID
FUNCTION_ID_ATAN2	enum FunctionID
FUNCTION_ID_ATANH	enum FunctionID
FUNCTION_ID_AVG	enum FunctionID
FUNCTION_ID_CHAR	enum FunctionID
FUNCTION_ID_CHOOSE	enum FunctionID
FUNCTION_ID_CLEAN	enum FunctionID
FUNCTION_ID_CODE	enum FunctionID
FUNCTION_ID_COLS	enum FunctionID
FUNCTION_ID_COS	enum FunctionID
FUNCTION_ID_COSH	enum FunctionID
FUNCTION_ID_COUNT	enum FunctionID
FUNCTION_ID_CTERM	enum FunctionID
FUNCTION_ID_DATE	enum FunctionID
FUNCTION_ID_DATEVALUE	enum FunctionID
FUNCTION_ID_DAY	enum FunctionID
FUNCTION_ID_DDB	enum FunctionID
FUNCTION_ID_ERR	enum FunctionID
FUNCTION_ID_EXACT	enum FunctionID
FUNCTION_ID_EXP	enum FunctionID
FUNCTION_ID_FACT	enum FunctionID
FUNCTION_ID_FALSE	enum FunctionID
FUNCTION_ID_FIND	enum FunctionID
FUNCTION_ID_FV	enum FunctionID
FUNCTION_ID_HLOOKUP	enum FunctionID
FUNCTION_ID_HOUR	enum FunctionID
FUNCTION_ID_IF	enum FunctionID
FUNCTION_ID_INDEX	enum FunctionID
FUNCTION_ID_INT	enum FunctionID
FUNCTION_ID_IRR	enum FunctionID
FUNCTION_ID_ISERR	enum FunctionID
FUNCTION_ID_ISNUMBER	enum FunctionID
FUNCTION_ID_ISSTRING	enum FunctionID
FUNCTION_ID_LEFT	enum FunctionID
FUNCTION_ID_LENGTH	enum FunctionID
FUNCTION_ID_LN	enum FunctionID
FUNCTION_ID_LOG	enum FunctionID
FUNCTION_ID_LOWER	enum FunctionID
FUNCTION_ID_MAX	enum FunctionID
FUNCTION_ID_MID	enum FunctionID
FUNCTION_ID_MIN	enum FunctionID

Revision: ■

Draft Dated (4/18/94)

Reference book



FUNCTION_ID_MINUTE	enum FunctionID
FUNCTION_ID_MOD	enum FunctionID
FUNCTION_ID_MONTH	enum FunctionID
FUNCTION_ID_N	enum FunctionID
FUNCTION_ID_NA	enum FunctionID
FUNCTION_ID_NOW	enum FunctionID
FUNCTION_ID_NPV	enum FunctionID
FUNCTION_ID_OR	enum FunctionID
FUNCTION_ID_PI	enum FunctionID
FUNCTION_ID_PMT	enum FunctionID
FUNCTION_ID_PRODUCT	enum FunctionID
FUNCTION_ID_PROPER	enum FunctionID
FUNCTION_ID_PV	enum FunctionID
FUNCTION_ID_RANDOM_N	enum FunctionID
FUNCTION_ID_RANDOM	enum FunctionID
FUNCTION_ID_RATE	enum FunctionID
FUNCTION_ID_REPEAT	enum FunctionID
FUNCTION_ID_REPLACE	enum FunctionID
FUNCTION_ID_RIGHT	enum FunctionID
FUNCTION_ID_ROUND	enum FunctionID
FUNCTION_ID_ROWS	enum FunctionID
FUNCTION_ID_SECOND	enum FunctionID
FUNCTION_ID_SIN	enum FunctionID
FUNCTION_ID SINH	enum FunctionID
FUNCTION_ID_SLN	enum FunctionID
FUNCTION_ID_SQRT	enum FunctionID
FUNCTION_ID_STD	enum FunctionID
FUNCTION_ID_STDP	enum FunctionID
FUNCTION_ID_STRING	enum FunctionID
FUNCTION_ID_SUM	enum FunctionID
FUNCTION_ID_SYD	enum FunctionID
FUNCTION_ID_TAN	enum FunctionID
FUNCTION_ID_TANH	enum FunctionID
FUNCTION_ID_TERM	enum FunctionID
FUNCTION_ID_TIME	enum FunctionID
FUNCTION_ID_TIMEVALUE	enum FunctionID
FUNCTION_ID_TODAY	enum FunctionID
FUNCTION_ID_TRIM	enum FunctionID
FUNCTION_ID_TRUE	enum FunctionID
FUNCTION_ID_TRUNC	enum FunctionID
FUNCTION_ID_UPPER	enum FunctionID
FUNCTION_ID_VALUE	enum FunctionID
FUNCTION_ID_VAR	enum FunctionID
FUNCTION_ID_VARP	enum FunctionID
FUNCTION_ID_VLOOKUP	enum FunctionID
FUNCTION_ID_WEEKDAY	enum FunctionID
FUNCTION_ID_YEAR	enum FunctionID
FUNCTION_ID_FILENAME	enum FunctionID
FUNCTION_ID_PAGE	enum FunctionID
FUNCTION_ID_PAGES	enum FunctionID

```
FUNCTION_ID_DEGREES          enum FunctionID
FUNCTION_ID_RADIANS          enum FunctionID
;
; External functions (defined by the application) start here.
;
FUNCTION_ID_FIRST_EXTERNAL_FUNCTIONenumFunctionID, 0x8000
```

Library: **parse.def**

■ FunctionType

```
FunctionType                record
                             :7
    FT_PRINT                 :1
    FT_TRIGONOMETRIC        :1
    FT_LOGICAL               :1
    FT_STATISTICAL          :1
    FT_STRING                :1
    FT_TIME_DATE             :1
    FT_FINANCIAL             :1
    FT_MATH                  :1
    FT_INFORMATION          :1
FunctionType                end
```

Library: **parse.def**





Reference book

Revision: ■

Draft Dated (4/18/94)

3Assembly Reference

■ GadgetSizeHintArgs

```
GadgetSizeHintArgs      struct
    GSHA_width           SpecWidth <>      ;Width of the composite
    GSHA_height          SpecHeight <>     ;Height of each child
GadgetSizeHintArgs      ends
```

Library: **Objects/genC.def**

■ GCCFeatures

```
GCCFeatures             record
    GCCF_HORIZONTAL_GUIDES :1
    GCCF_VERTICAL_GUIDES  :1
GCCFeatures             end
```

Library: **ruler.def**

■ GCMIcon

```
GCMIcon  etype          byte, 0
    GCMI_NONE  enum GCMIcon
    GCMI_EXIT  enum GCMIcon
    GCMI_HELP  enum GCMIcon
```

Library: **Objects/genC.def**

■ GCM_info

```
GCM_info etype          word, 0, 2
    GCMI_MIN_X  enum GCM_info;min x (left side bearing)
    GCMI_MIN_Y  enum GCM_info;min y (descent)
    GCMI_MAX_X  enum GCM_info;max x
    GCMI_MAX_Y  enum GCM_info;max y (ascent)
```

Library: **font.def**

■ GCNDriveChangeNotificationType

```
GCNDriveChangeNotificationType  etype word
    GCNDCNT_CREATED              enum GCNDriveChangeNotificationType
    GCNDCNT_DESTROYED            enum GCNDriveChangeNotificationType
```

Revision: ■

Draft Dated (4/18/94)

Reference book



Library: **gcnlist.def**

■ GCNExpressMenuNotificationType

```
GCNExpressMenuNotificationType    etype word
GCNEMNT_CREATED                  enum GCNExpressMenuNotificationType
GCNEMNT_DESTROYED                enum GCNExpressMenuNotificationType
```

Library: **gcnlist.def**

■ GCNListBlockHeader

```
GCNListBlockHeader struct
    GCNLBH_lmemHeader    LMemBlockHeader
    GCNLBH_listOfLists   lptr.GCNListOfListsHeader
GCNListBlockHeader ends
```

This structure begins a kernel's GCN list block.

Library: **gcnlist.def**

■ GCNListElement

```
GCNListElement    struct
    GCNLE_item     optr
GCNListElement    ends
```

This structure stores an element within a GCN list.

Library: **Objects/metaC.def**

■ GCNListHeader

```
GCNListHeader    struct
    GCNLH_meta        ChunkArrayHeader
    GCNLH_statusEvent  hptr
    GCNLH_statusData   hptr
    GCNLH_statusCount  word
GCNListHeader    ends
```

This structure defines a single GCN list (which resides in a chunk).

GCNLH_statusEvent stores a copy of the last notification event sent to this list via **GCNListSendStatus**. This event will be sent automatically to any object adding itself to the list. (This functionality is not yet used.)

GCNLH_statusData stores a copy of the extra data block, if any, passed in the above status event. This data block must be sharable, & have a reference count.

GCNLH_statusCount is incremented each time status is set for this list. This status count is used in the UI to avoid setting a status of NULL between changes in the target. If GCNLSF_IGNORE_IF_STATUS_TRANSITIONING is set in a 'Send' request, the GenApplication object will only set a NULL status if no status updates have been made after the time it takes to clear the process's queue when an object loses the target.

Library: **Objects/metaC.def**

■ GCNListMessageParams

```
GCNListMessageParams    struct
    GCNLMP_ID            GCNListType
    GCNLMP_block         hpPtr.GCNDataBlockHeader
    GCNLMP_event         hpPtr
    GCNLMP_flags         GCNListSendFlags
GCNListMessageParams    ends
```

GCNLMP_ID stores a list identifier - a combination of a **ManufacturerID** and a Manufacturer list type.

GCNLMP_block stores the handle of the extra data block, if used. (If there is no extra data block, this should be 0.) Blocks of this type must have a reference count, which may be initialized with **MemInitRefCount** and incremented for any new usage with **MemIncRefCount**. Methods in which the blocks are passed are considered a new usage and must have **MetaClass** handlers which call **MemDecRefCount**. Current messages supporting this behavior:

```
MSG_META_NOTIFY_WITH_DATA_BLOCK
MSG_NOTIFY_FILE_CHANGE.
```

GCNLMP_event stores a classed event to send to the list.

GCNLMP_flags stores the flags to pass on to **GCNListSend** or a similar primitive routine.

Library: **Objects/metaC.def**

■ GCNListOfListsElement

```
GCNListOfListsElement    struct
    GCNLOLE_ID            GCNListType
    GCNLOLE_list          lpPtr.GCNListHeader
GCNListOfListsElement    ends
```

This structure defines an element in a GCN list of lists.

Library: **Objects/metaC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GCNListOfListsHeader

```
GCNListOfListsHeader    struct
    GCNLOL_meta          ChunkArrayHeader
    GCNLH_data            label GCNListOfListsElement
GCNListOfListsHeader    ends
```

This structure starts a GCN lists of lists (and resides in a chunk). The label marks the start of multiple **GCNListOfListsElement** structures.

Library: **Objects/metaC.def**

■ GCNListParams

```
GCNListParams           struct
    GCNLP_ID             GCNListType
    GCNLP_optr           optr
GCNListParams           ends
```

GCNLP_ID stores the list identifier, which consists of a **ManufacturerID** and its associated Manufacturer list type.

GCNLP_optr stores the optr of the object to be added or removed from the list.

Library: **Objects/metaC.def**

■ GCNListSendFlags

```
GCNListSendFlags       record
    GCNLSF_SET_STATUS           :1
    GCNLSF_IGNORE_IF_STATUS_TRANSITIONING :1
    GCNLSF_FORCE_QUEUE         :1
                                :13
GCNListSendFlags       end
```

GCNLSF_SET_STATUS

During a **GCNListSend**, this flag additionally saves the message as the list's current "status". This "status" message will be automatically sent to any object adding itself to the list at a later point in time.

GCNLSF_IGNORE_IF_STATUS_TRANSITIONING

This flag is an optimization bit used to avoid a lull in status when transitioning between two different sources. This case may arise when the source is the current target object and one has just lost and another may soon gain the exclusive. (The bit should be set only when sending the "null"/"lost"/"not selected" status, as this is the event that should be tossed if another non-null status comes along shortly.)

Implementation is *not* provided by the kernel primitive routines, which ignore this bit, but may be provided by objects managing their own GCN lists. GenApplication objects respond to this bit by delaying the request until after

the UI and application queues have been cleared; then they only set the status as indicated if no other status has been set since the first request. Other objects may use their own logic to implement this optimization as is appropriate. Mechanisms which can not tolerate the delayed nature of this optimization, or require that all changes be registered, should not pass this bit set.

GCNLSF_FORCE_QUEUE

This flag informs **GCNListSend** to place the message on the event queue for the destination, even if the destination is run by the same thread as that sending the message.

Library: **Objects/metaC.def**

■ GCNListType

```
GCNListType      struct
  GCNLT_manuf    ManufacturerID
  GCNLT_type     word
GCNListType      ends
```

This structure defines a specific GCN list type. A GCN list type consists of a manufacturer ID describing each unique manufacturer and a specific list type defined for that manufacturer ID.

Library: **Objects/metaC.def**

■ GCNListTypeFlags

```
GCNListTypeFlags record
  ; high bits hold the list type.
                                :15
  GCNLTF_SAVE_TO_STATE        :1    ; set to indicate that list should be
                                ; saved to state.
GCNListTypeFlags end
```

Library: **Objects/metaC.def**

■ GCNShutdownControlType

```
GCNShutdownControlType etype word
  GCNSCT_SUSPEND        enum GCNShutdownControlType
  ; Task-switcher wishes to suspend the system.
  GCNSCT_SHUTDOWN       enum GCNShutdownControlType
  ; Task-switcher or other entity wishes to shut the system down to state.
  GCNSCT_UNSUSPEND      enum GCNShutdownControlType
  ; System has been unsuspended. No acknowledgement required.
```

Revision: ■

Draft Dated (4/18/94)

Reference book



Library: **gcnlist.def**

■ GCNStandardListType

```
GCNStandardListType      etype word, 0, 2
GCNSLT_FILE_SYSTEM      enum GCNStandardListType
; This notification is sent out when the file system changes.
```

Library: **gcnlist.def**

■ GDCFeatures

```
GDCFeatures              record
    GDCF_NEW              :1          ;replaced with switch documents in
                                ;transparent mode
    GDCF_OPEN_CLOSE      :1
    GDCF_QUICK_BACKUP    :1
    GDCF_SAVE            :1
    GDCF_SAVE_AS         :1
    GDCF_COPY            :1
    GDCF_EXPORT          :1
    GDCF_REVERT          :1
    GDCF_RENAME          :1          ;requires an auto-savable file
    GDCF_EDIT_USER_NOTES :1
    GDCF_SET_TYPE        :1
    GDCF_SET_PASSWORD    :1
    GDCF_SAVE_AS_TEMPLATE :1
    GDCF_SET_EMPTY_DOCUMENT :1
    GDCF_SET_DEFAULT_DOCUMENT :1
GDCFeatures              end
```

Library: **Objects/gDocCtrl.def**

■ GDCTask

```
GDCTask etype byte
    GDCT_NONE          enum GDCTask
    GDCT_NEW           enum GDCTask
    GDCT_OPEN          enum GDCTask
    GDCT_USE_TEMPLATE  enum GDCTask
    GDCT_SAVE_AS       enum GDCTask
    GDCT_COPY_TO       enum GDCTask
    GDCT_DIALOG        enum GDCTask
    GDCT_TYPE          enum GDCTask
    GDCT_PASSWORD      enum GDCTask
```

Library: **gDocCtrl.def**

■ GDCToolboxFeatures

```
GDCToolboxFeatures record
    GDCTF_NEW_EMPTY      :1
    GDCTF_USE_TEMPLATE    :1
    GDCTF_OPEN            :1
    GDCTF_CLOSE           :1
    GDCTF_SAVE            :1
    GDCTF_QUICK_BACKUP    :1
GDCToolboxFeatures end
```

Library: **Objects/gDocCtrl.def**

■ GDF_saved

```
GDF_saved      struct
    GDFS_nChars word      ; Number of characters to draw
    GDFS_drawPos PointWBFixed ; X/Y position to draw at
    GDFS_baseline WBFixed   ; Baseline for text
    GDFS_limit   word       ; Limit for underline or strike-through
    GDFS_flags   HyphenFlags
    align        word
GDF_saved      ends
```

This structure stores information about a graphics string and is used in the **GrDrawTextString** operation.

Library: **text.def**

■ GDF_vars

```
GDF_vars      struct
    GDFV_saved      GDF_saved
    GDFV_styleCallback fptr.far
    GDFV_textOffset dword
    GDFV_other       dword
    GDFV_textPointer dword
    align            word
GDF_vars      ends
```

This structure is passed to **GrDrawTextField**.

GDFV_saved stores the information to save for this graphics strings.

GDFV_styleCallback stores the callback routine for style changes.

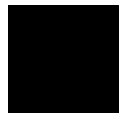
Callback Routine Specifications:

Passed:	ss:bp	<i>GDF_vars</i>
	bx:di	TextAttr buffer to fill in
	si	Offset into the field.

Revision: ■

Draft Dated (4/18/94)

Reference book



Return: **cx** Zero if this is the first call
Buffer pointed at by **bx:di** filled in.
cx Number of characters in this run
ds:si Pointer to the text at offset **si** in the field.

May Destroy: Nothing

GDFV_textOffset stores the offset to the start of the text to draw.

GDFV_other stores application specific data.

GDFV_textPointer stores the current text pointer (set by callback).

Library: **text.def**

■ **GDICFeatures**

```
GDICFeatures      record
  GDCF_OVERLAPPING_MAXIMIZED :1
  GDCF_TILE                  :1
  GDCF_DISPLAY_LIST          :1
GDICFeatures      end
```

Library: **Objects/gDCtrlC.def**

■ **GDICToolboxFeatures**

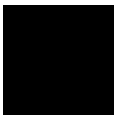
```
GDICToolboxFeatures record
  GDCTF_OVERLAPPING_MAXIMIZED :1
  GDCTF_TILE                   :1
  GDCTF_DISPLAY_LIST          :1
GDICToolboxFeatures end
```

Library: **Objects/gDCtrlC.def**

■ **GECFeatures**

```
GECFeatures      record
  GECF_UNDO        :1
  GECF_CUT         :1
  GECF_COPY        :1
  GECF_PASTE       :1
  GECF_SELECT_ALL  :1
  GECF_DELETE      :1
GECFeatures      end
```

Library: **Objects/gEditCC.def**



■ GECToolboxFeatures

```

GECToolboxFeatures record
    GECTF_UNDO      :1
    GECTF_CUT       :1
    GECTF_COPY      :1
    GECTF_PASTE     :1
    GECTF_SELECT_ALL:1
    GECTF_DELETE    :1
GECToolboxFeatures end

```

Library: **Objects/gEditCC.def**

■ GenAppDoDialogParams

```

GenAppDoDialogParams struct
    GADDP_dialog      StandardDialogParams
    GADDP_finishOD    optr           ; OD to send method to.
    GADDP_message     word          ; method to send.
GenAppDoDialogParams ends

```

Library: **Objects/gAppC.def**

■ GenAppIACPConnection

```

GenAppIACPConnection struc
    GAIACPC_connection IACPConnection
    ; The IACP connection
    GAIACPC_appMode    word
    ; The type of connection -- MSG_GEN_PROCESS_OPEN_APPLICATION or engine
    ; mode message)
GenAppIACPConnection ends

```

Library:

■ GenAppUpdateFeaturesParams

```

GenAppUpdateFeaturesParams struct
    GAUFP_featuresOn    word
    GAUFP_featuresChanged word
    GAUFP_level         UIInterfaceLevel
    GAUFP_oldLevel      UIInterfaceLevel
    GAUFP_appOpening    word
    GAUFP_table         fptr       ; table of fptrs to GenAppUsabilityTuple

```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

GAUFP_tableLength      word
GAUFP_levelTable       fptr.GenAppUsabilityTuple
GAUFP_reparentObject   optr
GAUFP_unReparentObject optr
GenAppUpdateFeaturesParams ends

```

Library: **Objects/gAppC.def**

■ GenAppUsabilityCommand

```

GenAppUsabilityCommand etype byte
GAUC_USABILITY          enum GenAppUsabilityCommand
GAUC_RECALC_CONTROLLER  enum GenAppUsabilityCommand
GAUC_REPARENT           enum GenAppUsabilityCommand
GAUC_POPUP              enum GenAppUsabilityCommand
GAUC_TOOLBAR            enum GenAppUsabilityCommand
GAUC_RESTART            enum GenAppUsabilityCommand

```

GAUC_USABILITY

Indicates that the controller should be usable if the feature is ON. (This is the default behavior.)

GAUC_RECALC_CONTROLLER

Indicates that the controller needs to have its features recalculated if the feature bit this table represents changes.

GAUC_REPARENT

Indicates that the controller should be moved to the *GAUFP_reparentObject*.

GAUC_POPUP

Indicates that the controller should be made a popup menu if the feature is ON (unless reverse is set).

GAUC_TOOLBAR

Indicates that the controller is a GenBoolean that corresponds to a toolbar state. Turning the feature off or on forces the GenBoolean to send an apply in addition to the normal behavior.

GAUC_RESTART

Indicates that this generic object needs to be restarted by setting it not-usable and then setting it usable.

Library: **gAppC.def**

■ GenAppUsabilityTuple

```
GenAppUsabilityTuple      struct
    GAUT_flags      GenAppUsabilityTupleFlags
    GAUT_objChunk    lptr
    GAUT_objResId    word
GenAppUsabilityTuple      ends
```

Library: **Objects/gAppC.def**

■ GenAppUsabilityTupleFlags

```
GenAppUsabilityTupleFlags      record
                                :2
    GAUTF_END_OF_LIST          :1
    GAUTF_OFF_IF_BIT_ON        :1
    GAUTF_COMMAND              GenAppUsabilityCommand:4
GenAppUsabilityTupleFlags      end
```

Library: **Objects/gAppC.def**

■ GenAttrs

```
GenAttrs      record
    GA_SIGNAL_INTERACTION_COMPLETE      :1
    GA_INITIATES_BUSY_STATE             :1
    GA_INITIATES_INPUT_HOLD_UP          :1
    GA_INITIATES_INPUT_IGNORE           :1
    GA_READ_ONLY                        :1
    GA_KBD_SEARCH_PATH                  :1
    GA_TARGETABLE                       :1
    GA_NOTIFY_VISIBILITY                 :1
GenAttrs      end
```

GA_SIGNAL_INTERACTION_COMPLETE

This flag is set to indicate that this GenTrigger completes user interaction with the associated GenInteraction when activated. This causes a MSG_GEN_GUP_INTERACTION_COMMAND with IC_INTERACTION_COMPLETE to be sent to the GenTrigger itself (eventually making its way up to the associated GenInteraction) after the trigger's action message is sent out. The specific UI (in the MSG_GEN_GUP_INTERACTION_COMMAND handler) will then determine whether this dialog should be dismissed or not.

This should be set for any ATTR_GEN_TRIGGER_INTERACTION_COMMAND trigger with IC_APPLY, IC_OK, IC_YES, IC_NO, or IC_STOP.

Revision: ■

Draft Dated (4/18/94)

Reference book



(This flag should not be set for IC_RESET triggers, as their usefulness depends on the dialog staying on-screen after their activation.) GA_SIGNAL_INTERACTION_COMPLETE should also be set for any other HINT_SEEK_REPLY_BAR triggers that should dismiss the dialog after usage.

This flag should not be used for GIA_INITIATED_VIA_USER_DO_DIALOG GenInteractions as the command triggers in those dialogs, by definition, signal interaction completion.

GA_INITIATES_BUSY_STATE

Set for gadgets whose invocation starts a long enough operation that we'd like to change the cursor to show busy. Results in a MSG_GEN_APPLICATION_MARK_BUSY being sent to the app object, followed by a MSG_GEN_APPLICATION_MARK_NOT_BUSY being sent to the same object but via the application's queue.

GA_INITIATES_INPUT_HOLD_UP

Set for gadgets whose invocation results in the application thread modifying the UI gadgtry slightly (typically enabling and disabling options). This flag causes input to be held up until the application has completed whatever its response is, so that the user cannot click twice on something that the app will disable after processing the first click.

Note: This functions stops input from being processed for all applications, so when using this bit, be sure that the gadget's application method handler is quick, or at least does not perform any prolonged operation.

Initiating a trigger with this flag results in a MSG_GEN_APPLICATION_HOLD_UP_INPUT being sent to the application object, followed by a MSG_GEN_APPLICATION_RESUME_INPUT being sent to the same object but via the application's queue.

GA_INITIATES_INPUT_IGNORE

This flags is set for gadgets whose invocation starts a long enough operation that we want to change the cursor to show that the app is busy and cannot take input. This flag causes the application to enter a modal state even if there isn't an application-modal dialog box up (i.e. any activity is eaten with a beep).

Initiating a trigger with this flag results in a MSG_GEN_APPLICATION_IGNORE_INPUT being sent to the app object, followed by a

MSG_GEN_APPLICATION_ACCEPT_INPUT being sent to the same object but via the app's queue.

GA_READ_ONLY

If set, the generic object is presumed to be a read-only version of the gadget (i.e. a text object that is not editable, a scrolling list whose items cannot be selected, a non-editable GenRange, without up/down arrows, etc.

GA_KBD_SEARCH_PATH

Set if there is a reason to look for keyboard accelerators along this section of the generic tree.

GA_TARGETABLE

Set if this object is a target of some sort and can receive the "Target" exclusive within its target level. If set, most specific UI's will automatically grab the Target for the object whenever the user interacts with it in some way, such as clicking on it.

GA_NOTIFY_VISIBILITY

Set if this object should send notification when it becomes visible and not visible. See the documentation with ATTR_GEN_VISIBILITY_DATA for more details.

Library: **Objects/genC.def**

■ **GenBranchInfo**

```
GenBranchInfo      record
  GBI_USABLE        :1
  GBI_BRANCH_MINIMIZED :1
                   :14
GenBranchInfo      end
```

GBI_USABLE

This bit is cleared if any generic parent found is not usable.

GBI_BRANCH_MINIMIZED

This bit is set if the object is within a branch which the specific UI has set the SA_BRANCH_MINIMIZED in. (This flag is only valid if GBI_USABLE is set.)

Library: **Objects/visC.def**

■ **GenControlBuildFlags**

```
GenControlBuildFlags  record
  GCBF_SUSPEND_ON_APPLY      :1
  GCBF_USE_GEN_DESTROY      :1
  GCBF_SPECIFIC_UI          :1
  GCBF_CUSTOM_ENABLE_DISABLE :1
  GCBF_ALWAYS_UPDATE        :1
```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

GCBF_EXPAND_TOOL_WIDTH_TO_FIT_PARENT      :1
GCBF_ALWAYS_INTERACTABLE                  :1
GCBF_ALWAYS_ON_GCN_LIST                   :1
GCBF_MANUALLY_REMOVE_FROM_ACTIVE_LIST    :1
GCBF_IS_ON_ACTIVE_LIST                    :1
GCBF_IS_ON_START_LOAD_OPTIONS             :1
GCBF_NOT_REQUIRED_TO_BE_ON_SELF_LOAD_OPTIONS_LIST :1
GCBF_DO_NOT_DESTROY_CHILDREN_WHEN_CLOSED  :1
                                           :3
GenControlBuildFlags      end

GCBF_SUSPEND_ON_APPLY
    This flag indicates that the object should be sent
    MSG_META_{SUSPEND,UNSUSPEND} at the beginning and end
    of MSG_GEN_APPLY.

GCBF_USE_GEN_DESTROY
    This flag specifies that unused objects cannot be destroyed
    using LMemFree.

GCBF_SPECIFIC_UI
    This flag specifies that the controller is at least partly
    implemented in the specific UI and therefore needs special
    treatment.

GCBF_CUSTOM_ENABLE_DISABLE
    This flag specifies that the GenControl should not set itself
    enabled or disabled based on
    MSG_GEN_CONTROL_ENABLE_DISABLE.

    Note: controllers that have this bit set and contain keyboard
    shortcuts must be marked GS_ENABLED initially.

GCBF_ALWAYS_UPDATE
    This flag forces MSG_GEN_CONTROL_UPDATE_UI to always be
    sent, even if the data block is 0.

GCBF_EXPAND_TOOL_WIDTH_TO_FIT_PARENT
    This flag expands the width of the tool control so that children
    can take advantage of extra space.

GCBF_ALWAYS_INTERACTABLE
    This flag indicates that the controller has set its interactable
    flag; this forces the controller to remain on its GCN lists, even
    if no part of it is visible. This flag must be set in conjunction
    with GCBF_IS_ON_ACTIVE_LIST.

GCBF_ALWAYS_ON_GCN_LIST
    This flag specifies that the controller should remain on the

```

specified GCN lists at all times. This flag must be set in conjunction with GCBF_IS_ON_ACTIVE_LIST.

GCBF_MANUALLY_REMOVE_FROM_ACTIVE_LIST

This flag specifies that the controller should not be removed from the active list in the MSG_META_DETACH handler.

GCBF_IS_ON_ACTIVE_LIST

This flag specifies that this controller is on the MGCNLT_ACTIVE_LIST.

GCBF_IS_ON_START_LOAD_OPTIONS_LIST

This flag is set if the controller is on the GAGCNLT_STARTUP_LOAD_OPTIONS list.

GCBF_NOT_REQUIRED_TO_BE_ON_SELF_LOAD_OPTIONS_LIST

This flag is set if the controller does not have to be on the GAGCNLT_SELF_LOAD_OPTIONS GCN list.

GCBF_DO_NOT_DESTROY_CHILDREN_WHEN_CLOSED

This controller's children will not be discarded when it is closed.

Library: **Objects/gCtrlC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GenControlBuildInfo

```

GenControlBuildInfo      struct
;
; General information
;
GCBI_flags                GenControlBuildFlags
GCBI_initFileKey          fptr.char           ;key to store data in
GCBI_gcnList              fptr.GCNListType     ;list of gcn lists to add to
GCBI_gcnCount             word                ;size of gcn list
GCBI_notificationList     fptr.NotificationType ;list of supported types
GCBI_notificationCount    word
GCBI_controllerName       optr
;
; Information for building normal visual representation
;
GCBI_dupBlock             hptr                ;handle of UI resource to
                                           ;duplicate or 0 for none
GCBI_childList            fptr.GenControlChildInfo
GCBI_childCount           word                ;number of children to add
GCBI_featuresList         fptr.GenControlFeaturesInfo
GCBI_featuresCount        word                ;size of features list
GCBI_features             word                ;bitmask for default features
;
;Information for building toolbox
;
GCBI_toolBlock            hptr                ;handle of UI resource
                                           ;containing tools
GCBI_toolList             fptr.GenControlChildInfo
GCBI_toolCount            word                ;number of tools to add
GCBI_toolFeaturesList     fptr.GenControlFeaturesInfo
GCBI_toolFeaturesCount    word                ;size of tools features list
GCBI_toolFeatures         word                ;bitmask for default features
GCBI_helpContext          fptr.char           ;if non-zero then add
                                           ;ATTR_GEN_HELP_CONTEXT with
                                           ;this string being the context
                                           ;reserved for future expansion
GCBI_reservedbyte 8 dup (0)
GenControlBuildInfo      ends

```

Library: **Objects/gCtrlC.def**

■ GenControlChildFlags

```
GenControlChildFlags    record
    GCCF_NOTIFY_WHEN_ADDING    :1
    GCCF_ALWAYS_ADD          :1
    GCCF_IS_DIRECTLY_A_FEATURE    :1
GenControlChildFlags    end
```

Library: **Objects/gCtrlC.def**

■ GenControlChildInfo

```
GenControlChildInfo    struct
    GCCI_object          lptr
    GCCI_featureMask      word
    GCCI_flags            GenControlChildFlags
GenControlChildInfo    ends
```

GCCI_featureMask stores a bitmask of the feature that this object exhibits or a bitmask of the combination of tools that compose this object.

Library: **Objects/gCtrlC.def**

■ GenControlFeatureFlags

```
GenControlFeatureFlags    record
    :8
GenControlFeatureFlags    end
```

Library: **Objects/gCtrlC.def**

■ GenControlFeaturesInfo

```
GenControlFeaturesInfo    struct
    GCFI_object          lptr
    GCFI_name            optr
    GCFI_flags            GenControlFeatureFlags
GenControlFeaturesInfo    ends
```

GCFI_object stores the lptr of the controller's associated object.

GCFI_name stores an optr to a reference chunk. This chunk contains a reference to the name of the feature (if the feature is allowed be changed).

Library: **Objects/gCtrlC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **GenControlInteractableFlags**

```

GenControlInteractableFlags  record
    GCIF_CONTROLLER  :1          ;Controller object itself is interactable and
                                ;may need to be enabled/disabled
                                :13
    GCIF_TOOLBOX_UI  :1          ;Toolbox UI is interactable
    GCIF_NORMAL_UI   :1          ;Normal UI is interactable
GenControlInteractableFlags  end

```

Library: **Objects/gCtlC.def**

■ GenControlScalableUICommand

```
GenControlScalableUICommand  etype byte
    GCSUIC_SET_NORMAL_FEATURES_IF_APP_FEATURE_ON enum GenControlScalableUICommand
    ; if (GCSUIE_appFeature is ON)
    ;     menu features = GCSUIE_newFeatures
    ;
    GCSUIC_SET_TOOLBOX_FEATURES_IF_APP_FEATURE_ON enum
    GenControlScalableUICommand
    ; if (GCSUIE_appFeature is ON)
    ;     tool features = GCSUIE_newFeatures
    ;
    GCSUIC_SET_NORMAL_FEATURES_IF_APP_FEATURE_OFF enum
    GenControlScalableUICommand
    ; if (GCSUIE_appFeature is OFF)
    ;     menu features = GCSUIE_newFeatures
    ;
    GCSUIC_SET_TOOLBOX_FEATURES_IF_APP_FEATURE_OFF enum
    GenControlScalableUICommand
    ; if (GCSUIE_appFeature is OFF)
    ;     tool features = GCSUIE_newFeatures
    ;
    GCSUIC_SET_NORMAL_FEATURES_IF_APP_LEVEL enum GenControlScalableUICommand
    ; if (app level >= GCSUIE_appFeature)
    ;     menu features = GCSUIE_newFeatures
    ;
    GCSUIC_SET_TOOLBOX_FEATURES_IF_APP_LEVEL enum GenControlScalableUICommand
    ; if (app level >= GCSUIE_appFeature)
    ;     tool features = GCSUIE_newFeatures
    ;
    GCSUIC_ADD_NORMAL_FEATURES_IF_APP_FEATURE_ON enum
    GenControlScalableUICommand
    ; if (GCSUIE_appFeature is ON)
    ;     menu features |= GCSUIE_newFeatures
    ;
    GCSUIC_ADD_TOOLBOX_FEATURES_IF_APP_FEATURE_ON enum
    GenControlScalableUICommand
    ; if (GCSUIE_appFeature is ON)
    ;     tool features |= GCSUIE_newFeatures
```

This type is passed with the **GenControlScalableUIEntry** structure.

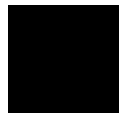
GCSUIC_SET_NORMAL_FEATURES_IF_APP_FEATURE_ON

If the particular feature within *GCSUIE_appFeature* is set, then the normal (menu) features within *GCSUIE_newFeatures* are set absolutely. If you would rather have these new features added to already existing features, use **GCSUIC_ADD_NORMAL_FEATURES_IF_APP_FEATURE_ON**.

Revision: ■

Draft Dated (4/18/94)

Reference book



GCSUIC_SET_TOOLBOX_FEATURES_IF_APP_FEATURE_ON

If the particular feature within *GCSUIE_appFeature* is set, then the toolbox features within *GCSUIE_newFeatures* are set absolutely. If you would rather have these new features added to already existing features, use **GCSUIC_ADD_TOOLBOX_FEATURES_IF_APP_FEATURE_ON**.

GCSUIC_SET_NORMAL_FEATURES_IF_APP_FEATURE_OFF

If the particular feature within *GCSUIE_appFeature* is clear, then the normal (menu) features within *GCSUIE_newFeatures* are set absolutely.

GCSUIC_SET_TOOLBOX_FEATURES_IF_APP_FEATURE_OFF

If the particular feature within *GCSUIE_appFeature* is clear, then the toolbox features within *GCSUIE_newFeatures* are set absolutely.

GCSUIC_SET_NORMAL_FEATURES_IF_APP_LEVEL

If (app level >= *GCSUIE_appFeature*) then the normal (menu) features within *GCSUIE_newFeatures* are set absolutely.

GCSUIC_SET_TOOLBOX_FEATURES_IF_APP_LEVEL

If (app level >= *GCSUIE_appFeature*) then the toolbox features within *GCSUIE_newFeatures* are set absolutely.

GCSUIC_ADD_NORMAL_FEATURES_IF_APP_FEATURE_ON

If the particular feature within *GCSUIE_appFeature* is set, then the normal (menu) features within *GCSUIE_newFeatures* are added to any already existing features. If you would rather have these new features set absolutely, use **GCSUIC_SET_NORMAL_FEATURES_IF_APP_FEATURE_ON**.

GCSUIC_ADD_TOOLBOX_FEATURES_IF_APP_FEATURE_ON

If the particular feature within *GCSUIE_appFeature* is set, then the toolbox features within *GCSUIE_newFeatures* are added to any already existing features. If you would rather have these new features set absolutely, use **GCSUIC_SET_TOOLBOX_FEATURES_IF_APP_FEATURE_ON**.

Library: **gCtlC.def**

■ GenControlScalableUIEntry

```
GenControlScalableUIEntry    struct
    GCSUIE_command            GenControlScalableUICommand
    GCSUIE_appFeature          word                ;feature bit to check.
    GCSUIE_newFeatures         word                ;new features bits to use.
GenControlScalableUIEntry    ends
```

Library: **Objects/gCtrlC.def**

■ GenControlScanInfo

```
GenControlScanInfo          struct
    GCSI_userAdded            word
    GCSI_userRemoved          word
    GCSI_appRequired          word
    GCSI_appProhibited        word
GenControlScanInfo          ends
```

Library: **Objects/gCtrlC.def**

■ GenControlStatusChange

```
GenControlStatusChange      record
                                :13
    GCSF_HIGHLIGHTED_TOOLGROUP_SELECTED :1
    GCSF_TOOLBOX_FEATURES_CHANGED       :1
    GCSF_NORMAL_FEATURES_CHANGED        :1
GenControlStatusChange      end
```

GCSF_HIGHLIGHTED_TOOLGROUP_SELECTED

Set if user has clicked, or in some other manner, “selected” the toolgroup of a particular controller. This flag is used by GenToolControl to provide the shortcut for the user of scrolling the ToolGroup list to this selection.

GCSF_TOOLBOX_FEATURES_CHANGED

This flag is set if toolbox features have been added or removed

GCSF_NORMAL_FEATURES_CHANGED

This flag is set if normal features have been added or removed.

Library: **Objects/gCtrlC.def**

■ GenControlUIType

```
GenControlUIType            etype word
    GCUIT_NORMAL            enum      GenControlUIType
    GCUIT_TOOLBOX           enum      GenControlUIType
```

Revision: ■

Draft Dated (4/18/94)

Reference book



GCUIT_NORMAL

This type indicates that the “normal UI” components are set. Generally, this includes menu items or features within a dialog box.

GCUIT_TOOLBOX

This type indicates that the toolbox components are set. A toolbox generally consists of “Tiny” sized triggers or items within popup lists.

Library: **gCtrlC.def**

■ GenControlUpdateUIParams

```
GenControlUpdateUIParams struct
    GCUUIP_manufacturer      ManufacturerID
    GCUUIP_changeType        word
    GCUUIP_dataBlock         hpPtr
    GCUUIP_features          word
    GCUUIP_toolboxFeatures   word
    GCUUIP_childBlock        hpPtr
    GCUUIP_toolBlock         hpPtr
GenControlUpdateUIParams ends
```

GCUUIP_features stores the features list from the GenControl's temporary instance data TEMP_GEN_CONTROL_INSTANCE. This entry is clear if GCIF_NORMAL_UI is not set in TEMP_GEN_CONTROL_INSTANCE.

GCUUIP_toolboxFeatures stores the tools features list from the GenControl's TEMP_GEN_CONTROL_INSTANCE. This entry is clear if GCIF_TOOLBOX_UI is not set in TEMP_GEN_CONTROL_INSTANCE.

GCUUIP_childBlock stores the optr of the child block (from TEMP_GEN_CONTROL_INSTANCE).

Library: **Objects/gCtrlC.def**

■ GenControlUserData

```
GenControlUserData struct
    GCUD_flags               GenControlUserFlags
    GCUD_userAddedUI         word
    GCUD_userRemovedUI       word
    GCUD_userAddedToolboxUI  word
    GCUD_userRemovedToolboxUI word
GenControlUserData ends
```

Library: **Objects/gCtrlC.def**

Revision: ■

Draft Dated (4/18/94)

■ **GenControlUserFlags**

```
GenControlUserFlags    record
                        :14
    GCUF_USER_TOOLBOX_UI    :1
    GCUF)USER_UI          :1
GenControlUserFlags    end
```

Library: **Objects/gCtrlC.def**

■ **GenDefaultMonikerType**

```
GenDefaultMonikerType  etype word
    ; monikers used for various levels in the Set User Level dialog box.
    GDMT_LEVEL_0        enum GenDefaultMonikerType
    GDMT_LEVEL_1        enum GenDefaultMonikerType
    GDMT_LEVEL_2        enum GenDefaultMonikerType
    GDMT_LEVEL_3        enum GenDefaultMonikerType
    ;
    ; moniker used for Help triggers in dialog boxes, etc.
    GDMT_HELP           enum GenDefaultMonikerType
    ;
    ; moniker used for Help triggers in the title bar of the primary.
    GDMT_HELP_PRIMARY   enum GenDefaultMonikerType
```

Library: **Objects/genC.def**

■ **GenDisplayAttrs**

```
GenDisplayAttrs    record
    GDA_USER_DISMISSABLE    :1
                        :7
GenDisplayAttrs    end
```

GDA_USER_DISMISSABLE

This flag is set if user is allowed to dismiss this window. Dismissing the display will close the window. A GenDisplay's user dismissable behavior does not affect iconification operations. This attribute is implemented in some specific UIs (e.g. Open Look) by providing a push-pin which may be unpinned. Other specific UIs (e.g. CUA) provide a "CLOSE" option in the system menu.

Library: **Objects/gDispC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GenDisplayControlAttributes

```
GenDisplayControlAttributes    record
    GDCA_MAXIMIZED_NAME_ON_PRIMARY    :1
                                      :7
GenDisplayControlAttributes    end
```

GDCA_MAXIMIZED_NAME_ON_PRIMARY

This flag sets the moniker of a maximized display is the long term moniker of the primary.

Library: **Objects/gDCtrlC.def**

■ GenDocumentAttrs

```
GenDocumentAttrs    record
;
; These bits reflect permanent attributes of the document
;
GDA_READ_ONLY            :1      ;File is opened read-only
GDA_READ_WRITE           :1      ;File is opened read-write
GDA_FORCE_DENY_WRITE    :1      ;File is opened "force deny write"
GDA_SHARED_MULTIPLE      :1      ;File opened "shared multiple"
GDA_SHARED_SINGLE        :1      ;File opened "shared single"
;
; These bits reflect temporary states of the document -- these bits are set
; by the document control object
;
GDA_UNTITLED             :1      ;File does not have a real (user) name
GDA_DIRTY                 :1      ;File has been modified
GDA_CLOSING              :1      ;File is being closed
GDA_ATTACH_TO_DIRTY_FILE :1      ;File is attached to a dirty file
GDA_SAVE_FAILED          :1      ;"Save" failed, revert is not possible
GDA_OPENING              :1      ;Document is being opened
GDA_AUTO_SAVE_STOPPED    :1      ;Auto-save has been stopped
GDA_MODEL                 :1      ;Document has the model exclusive
GDA_ON_WRITABLE_MEDIA    :1
GDA_BACKUP_EXISTS        :1      ;A document backup file exists
;
; These bits reflect temporary states of the document -- these bits are set
; by the application
;
GDA_PREVENT_AUTO_SAVE     :1      ;Do not auto save (temporary state set
                                   ;by the application)
GenDocumentAttrs    end
```

Library: **Objects/gDocC.def**

■ GenDocumentChangePasswordParams

```
GenDocumentChangePasswordParams    struct
    GDCLPP_password  char (MAX_PASSWORD_SIZE+2) dup (?)
GenDocumentChangePasswordParams    ends
```

Library: **Objects/gDocC.def**

■ GenDocumentControlAttrs

```
GenDocumentControlAttrs    record
;
; File attributes
;
GDCA_MULTIPLE_OPEN_FILES    :1      ; Allows multiple files to be opened
GDCA_MODE                    GenDocumentControlMode:2
GDCA_DOS_FILE_DENY_WRITE    :1      ; If GDCA_VM_FILE is not set, then open
                                   ; a standard DOS file deny-write
GDCA_VM_FILE                  :1      ; Documents stored in VM files
GDCA_NATIVE                   :1      ; If GDCA_VM_FILE is not set, documents
                                   ; are stored in a format native to the
                                   ; file system
GDCA_SUPPORTS_SAVE_AS_REVERT :1      ; Document uses "save as"
;
; Current state
;
GDCA_DOCUMENT_EXISTS          :1      ; At least one document exists
GDCA_CURRENT_TASK GDCTask     :4      ; Current task being performed
GDCA_DO_NOT_SAVE_FILES        :1      ; Working model support...
GDCA_FORCE_DEMAND_PAGING      :1      ; Forces demand-paging of documents,
                                   ; even on systems that force documents
                                   ; completely into memory.
                                   :2
GenDocumentControlAttrs    end
```

Library: **Objects/gDocCtrl.def**

■ GenDocumentControlFeatures

```
GenDocumentControlFeatures    record
; File features
GDCLF_READ_ONLY_SUPPORTS_SAVE_AS_REVERT    :1
GDCLF_SINGLE_FILE_CLEAN_CAN_NEW_OPEN       :1
GDCLF_SUPPORTS_TEMPLATES                   :1
```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

GDCF_SUPPORTS_USER_SETTABLE_EMPTY_DOCUMENT      :1
GDCF_SUPPORTS_USER_SETTABLE_DEFAULT_DOCUMENT    :1
GDCF_SUPPORTS_USER_MAKING_SHARED_DOCUMENTS       :1
GDCF_NAME_ON_PRIMARY                             :1
                                                    :9

GenDocumentControlFeatures      end

    GDCF_READ_ONLY_SUPPORTS_SAVE_AS_REVERT
    If set, the document control allows read-only files to be edited.

    GDCF_SINGLE_FILE_CLEAN_CAN_NEW_OPEN
    If set, the document control allows the user to use “new” or “open” to create
    another document even if multiple files are not allowed. The current
    document must be clean.

    GDCF_SUPPORTS_TEMPLATES
    If set, the document control supports template documents.

Library:      Objects/gDocCtrl.def

```

■ GenDocumentControlMode

```

GenDocumentControlMode      etype byte
    GDCM_VIEWER              enum GenDocumentControlMode
    GDCM_SHARED_SINGLE       enum GenDocumentControlMode
    GDCM_SHARED_MULTIPLE     enum GenDocumentControlMode

```

Library: **Objects/gDocCtrl.def**

■ GenDocumentGetVariableParams

```

GenDocumentGetVariableParams      struct
    GDGVP_position      PointDWord      ;object position
    GDGVP_buffer        fptr.char       ;buffer for result
    GDGVP_graphic       fptr.VisTextGraphic ;graphic
    GDGVP_object        optr            ;source object
GenDocumentGetVariableParams      ends

```

Library: **Objects/gDocC.def**

■ GenDocumentGroupAttrs

```
GenDocumentGroupAttrs    record
    GDGA_VM_FILE          :1 ;Documents stored in VM files
    GDGA_NATIVE           :1 ;If document not in VM file,
                           ;then should be in format
                           ;native to file system.
    GDGA_SUPPORTS_AUTO_SAVE :1 ;Use auto-save
    GDGA_AUTOMATIC_CHANGE_NOTIFICATION :1 ;Automatically provide change
                           ;notification
    GDGA_AUTOMATIC_DIRTY_NOTIFICATION :1 ;Use automatic mechanism for
                           ;VM dirty notification
    GDGA_APPLICATION_THREAD :1 ;Set if AppDocumentControl runs
                           ;in the application thread
    GDGA_VM_FILE_CONTAINS_OBJECTS :1 ;Set if appropriate VM
                           ;attributes for storing objects
                           ;should be set in the VM file
    GDGA_CONTENT_DOES_NOT_MANAGE_CHILDREN :1 ;VisContent does not manage its
                           ;children
    GDGA_LARGE_CONTENT :1 ;VisContent uses large model
    GDGA_AUTOMATIC_UNDO_INTERACTION :1 ;Sends out undo set-context
                           ; messages automatically
                           :6
GenDocumentGroupAttrs    end
```

Library: **Objects/gDocGrpC.def**

■ GenDocumentOperation

```
GenDocumentOperation    etype word
    GDO_NORMAL           enum GenDocumentOperation
    GDO_SAVE_AS          enum GenDocumentOperation
    GDO_REVERT           enum GenDocumentOperation
    GDO_REVERT_QUICK     enum GenDocumentOperation
    GDO_ATTACH           enum GenDocumentOperation
    GDO_DETACH           enum GenDocumentOperation
    GDO_NEW              enum GenDocumentOperation
    GDO_OPEN             enum GenDocumentOperation
    GDO_SAVE             enum GenDocumentOperation
    GDO_CLOSE            enum GenDocumentOperation
    GDO_AUTO_SAVE        enum GenDocumentOperation
```

Library: **Objects/gDocC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GenDocumentType

```
GenDocumentType      etype word
    GDT_NORMAL          enum GenDocumentType
    GDT_READ_ONLY       enum GenDocumentType
    GDT_TEMPLATE         enum GenDocumentType
    GDT_READ_ONLY_TEMPLATE enum GenDocumentType
    GDT_PUBLIC           enum GenDocumentType
    GDT_MULTI_USER       enum GenDocumentType
```

Library: **Objects/gDocC.def**

■ GenDynamicListPosition

```
GenDynamicListPosition etype word
    GDLP_FIRST          enum GenDynamicListPosition, 00000h
    GDLP_LAST           enum GenDynamicListPosition, 0ffffh
```

Library: **Objects/gDListC.def**

■ GeneralConsumerModeFlags

```
GeneralConsumerModeFlags record
    :2
    GCMF_LEFT_ICON      GCMIcon:3      ; Indicates which icon to show on
                                   ; the left side of the title bar.
    GCMF_RIGHT_ICON     GCMIcon:3      ; Indicates which icon to show on
                                   ; the right side of the title bar.
GeneralConsumerModeFlags end
```

Library: **Objects/genC.def**

■ GeneralEvent

```
GeneralEvent          etype word, 0, 2
    GE_NO_EVENT        enum GeneralEvent
    GE_END_OF_SONG     enum GeneralEvent
    GE_SET_PRIORITY    enum GeneralEvent
    GE_SET_TEMPO       enum GeneralEvent
    GE_SEND_NOTIFICATION enum GeneralEvent
    GE_V_SEMAPHORE     enum GeneralEvent
```

This types stores events that are required in the sound stream, but are not actually involved in the generation of specific sounds.

GE_NO_EVENT

This event generates exceptionally long durations.

GE_END_OF_SONG

This event marks the end of the song. Any event or delta-time after an EOS mark will be ignored.

GE_SET_PRIORITY

This event changes the priority of the stream. All following events will be evaluated at that priority.

GE_SET_TEMPO

This event changes the tempo of the song from that point onward. Any delta-Tempo following that event will use the new value as the # of msec per 64th beats.

GE_SEND_NOTIFICATION

This event causes the stream to send a given message to a given object. The message will be placed at the end of the queue.

GE_V_SEMAPHORE

This event causes the stream to V the semaphore handle.

Library: **sound.def**

■ GenFieldFlags

```
GenFieldFlags      record
  GFF_DETACHING      :1
  GFF_LOAD_BITMAP    :1
  GFF_RESTORING_APPS :1
  GFF_NEEDS_WORKSPACE_MENU :1
  GFF_HAS_DEFAULT_LAUNCHER :1
  GFF_NEED_DEFAULT_LAUNCHER :1
  GFF_QUIT_ON_CLOSE :1
  GFF_LOAD_DEFAULT_LAUNCHER_WHEN_NEXT_PROCESS_EXITS :1
GenFieldFlags      end
```

These flags affect one of the system objects - the GenField object. As such, there will be no need for your application to set or alter these flags.

GFF_DETACHING

This flag is set if MSG_META_DETACH has been sent to the GenField object. This flag is cleared when the detach is complete.

GFF_LOAD_BITMAP

This flag is set if we want to draw a bitmap on this field.

GFF_RESTORING_APPS

This flag is set if we are currently restoring applications.

Revision: ■

Draft Dated (4/18/94)

Reference book



GFF_NEEDS_WORKSPACE_MENU

This flag is set if an application express menu is needed for the field.

GFF_HAS_DEFAULT_LAUNCHER

This flag is set if this field should start a default launcher. The name of this launcher is stored in GEOS.INI file under key 'defaultLauncher' and category specified by ATTR_GEN_INIT_FILE_CATEGORY.

GFF_NEED_DEFAULT_LAUNCHER

Set if the field detached because it had no focusable apps available, so we need to start the default launcher when we restore it.

GFF_QUIT_ON_CLOSE

Set if the field is in the process of doing a 'quitOnClose'.

GFF_LOAD_DEFAULT_LAUNCHER_WHEN_NEXT_PROCESS_EXITS

We tried to load the default launcher, but couldn't because the system was too busy - wait until a process exits, then try again.

Library: **Objects/gFieldC.def**

■ GenFilePath

```
GenFilePath      struct
  GFP_disk       word SP_TOP
  GFP_path       PathName
GenFilePath      ends
```

GFP_disk stores the handle of the disk on which the path resides. This may be initialized to a **StandardPath** constant.

GFP_path stores the absolute path (or relative path if *GFP_disk* is a **StandardPath** constant) to the directory.

Library: **Objects/genC.def**

■ GenFileSelectorEntryFlags

```
GenFileSelectorEntryFlags  record
  GFSEF_TYPE                GenFileSelectorEntryType:2
  GFSEF_OPEN                 :1
  GFSEF_NO_ENTRIES           :1
  GFSEF_ERROR                 :1
  GFSEF_TEMPLATE             :1
  GFSEF_SHARED_MULTIPLE      :1
```

```

GFSEF_SHARED_SINGLE      :1
GFSEF_READ_ONLY          :1
GFSEF_PARENT_DIR         :1
                           :6
GenFileSelectorEntryFlags      end

    GFSEF_TYPE
        This flags stores the type of entry selected.

    GFSEF_OPEN
        The selection should be opened. (User has double-clicked).

    GFSEF_NO_ENTRIES
        No entries are within the file selector's list.

    GFSEF_ERROR
        The file selector encountered an error opening a selection entry
        (through MSG_GEN_FILE_SELECTOR_OPEN_ENTRY or a
        double-click).

    GFSEF_TEMPLATE
        This flag is set if the file is a template (from GFHF_TEMPLATE).

    GFSEF_SHARED_MULTIPLE
        This flag is set if the file is shared with multiple writers (from
        GFHF_SHARED_MULTIPLE).

    GFSEF_SHARED_SINGLE
        This flag is set if the file is shared with a single writer (from
        GFHF_SHARED_SINGLE).

    GFSEF_READ_ONLY
        This flag is set if the file is read-only (from FA_RDONLY).

    GFSEF_PARENT_DIR
        This flag is set if the current selection is the parent directory
        entry (first entry).

```

Library: **Objects/gFSelC.def**

■ GenFileSelectorEntryType

```

GenFileSelectorEntryType  etype byte, 0
    GFSET_FILE             enum GenFileSelectorEntryType
    GFSET_SUBDIR           enum GenFileSelectorEntryType
    GFSET_VOLUME           enum GenFileSelectorEntryType

```

Library: **Objects/gFSelC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **GenFileSelectorFileAttrs**

```
GenFileSelectorFileAttrs      struct
    GFSFA_match      FileAttrs      ; Attributes that must match
    GFSFA_mismatch    FileAttrs      ; Attributes that must not match
GenFileSelectorFileAttrs      ends
```

Library: **Object/gFSelC.def**

■ **GenFileSelectorGeodeAttrs**

```
GenFileSelectorGeodeAttrs     struct
    GFSGA_match      GeodeAttrs     ; Attributes that must match
    GFSGA_mismatch    GeodeAttrs     ; Attributes that must not match
GenFileSelectorGeodeAttrs     ends
```

Library: **Objects/gFSelC.def**

■ **GenFileSelectorScalableUICommand**

```
GenFileSelectorScalableUICommand  etype byte
    GFSSUIC_SET_FEATURES_IF_APP_FEATURE_ON      enum
    GenFileSelectorScalableUICommand
    GFSSUIC_SET_FEATURES_IF_APP_FEATURE_OFF      enum
    GenFileSelectorScalableUICommand
    GFSSUIC_ADD_FEATURES_IF_APP_FEATURE_ON      enum
    GenFileSelectorScalableUICommand
    GFSSUIC_SET_FEATURES_IF_APP_LEVEL            enum
    GenFileSelectorScalableUICommand
    GFSSUIC_ADD_FEATURES_IF_APP_LEVEL            enum
    GenFileSelectorScalableUICommand
```

Library: **Objects/gFSelC.def**

■ **GenFileSelectorScalableUIEntry**

```
GenFileSelectorScalableUIEntry    struct
    GFSSUIE_command      GenFileSelectorScalableUICommand
    GFSSUIE_appFeature    word
    GFSSUIE_fsFeatures    FileSelectorAttrs
GenFileSelectorScalableUIEntry    ends
```

Library: **Objects/gFSelC.def**

■ GenFileSelectorType

```
GenFileSelectorType      etype byte
    GFST_DOCUMENTS      enum GenFileSelectorType
    GFST_EXECUTABLES    enum GenFileSelectorType
    GFST_NON_GEOS_FILES  enum GenFileSelectorType
    GFST_ALL_FILES      enum GenFileSelectorType
```

Library: **Objects/gDocCtrl.def**

■ GenFindObjectWithMonikerFlags

```
GenFindObjectWithMonikerFlags record
    GFOWMF_EXACT_MATCH      :1
    GFOWMF_SKIP_THIS_NODE   :1
                                :14
```

```
GenFindObjectWithMonikerFlags end
```

GFOWMF_EXACT_MATCH

If set, text within the searched moniker must match the passed text completely. The passed text will not match if it represents only a portion of an object's moniker text.

GFOWMF_SKIP_THIS_NODE

If set, the search operation will skip this object and just check objects below it in the generic tree.

Library: **Objects/genC.def**

■ GenGadgetAttributes

```
GenGadgetAttributes      record
    GGA_COMPOSITE         :1
                                :7
```

```
GenGadgetAttributes      end
```

GGA_COMPOSITE

This flag is set if gadget object should become a VisComp. If set then all generic children will become visual children.

Library: **Objects/gGadgetC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GenInteractionAttrs

```
GenInteractionAttrs      record
    GIA_NOT_USER_INITIATABLE :1
    GIA_INITIATED_VIA_USER_DO_DIALOG :1
    GIA_MODAL :1
    GIA_SYS_MODAL :1
    :4
GenInteractionAttrs      end
```

GIA_NOT_USER_INITIATABLE

This flag is set to indicate that a dialog GenInteraction should build an activation trigger that brings up the dialog. Instead, the dialog must be brought up with MSG_GEN_INTERACTION_INITIATE. In this case, the GenInteraction should be a child of GenPrimary or GenApplication.

GIA_INITIATED_VIA_USER_DO_DIALOG

This flag is set to indicate that a dialog GenInteraction will be displayed using the routine **UserDoDialog**. Input hold up, ignore, & busy states are overridden by default to allow users to interact with this type of GenInteraction (and prevent user-lock-out which could potentially occur in these cases).

GIA_MODAL

This flag is set to indicate that a dialog GenInteraction needs to be modal. This modality indicates that the application has been coded in such a way that it cannot allow a dialog box to stay on-screen while allowing the user to work in other areas of the application. (E.g. selection information in the dialog box is not updated if the user were to change the selection).

GIA_SYS_MODAL

This flag sets a dialog GenInteraction modal at the system level. Only use this flag if no other way can be found to perform the required operation, as it will halt input to all other parts of the system.

Library: **Objects/gInterC.def**

■ GenInteractionDiscardInfo

```
GenInteractionDiscardInfo      struct
    GIDI_inUse                 word
    ; If non-zero, the interaction is onscreen, or is about to go
    ; onscreen, so it should not be discarded.

    GIDI_discardCount          word
    ; Count of MSG_GEN_DESTROY_AND_DISCARD_BLOCK messages that have to
    ; come in before the block is discarded.
GenInteractionDiscardInfo      ends
```

Library: **gInterC.def**

■ GenInteractionGroupType

```
GenInteractionGroupType      etype byte
    GIGT_FILE_MENU            enum    GenInteractionGroupType
    ; Set to indicate that this GenInteraction is the File menu. Can
    ; contain DocumentControl or other file-related commands.
    ;
    GIGT_EDIT_MENU            enum    GenInteractionGroupType
    ; Set to indicate that this GenInteraction is the Edit menu. Can
    ; contain EditControl or other edit commands.
    ;
    GIGT_VIEW_MENU            enum    GenInteractionGroupType
    ; Set to indicate that this GenInteraction is the View menu. Can
    ; contain ViewControl or other view commands.
    ;
    GIGT_OPTIONS_MENU         enum    GenInteractionGroupType
    ; Set to indicate that this GenInteraction is the Options menu. Can
    ; contain application options.
    ;
    GIGT_WINDOW_MENU          enum    GenInteractionGroupType
    ; Set to indicate that this GenInteraction is the Window menu. Can
    ; contain GenDisplayControl or other window commands.
    ;
    GIGT_HELP_MENU            enum    GenInteractionGroupType
    ; Set to indicate that this GenInteraction is the Help menu.
    ;
    GIGT_PRINT_GROUP          enum    GenInteractionGroupType
    ; Set to indicate that this GenInteraction is the Print group. Can
    ; contain PrintControl or other print commands.
    ;
```

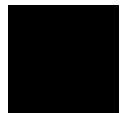
GIGT_FILE_MENU

This indicates that this GenInteraction acts as the File menu.

Revision: ■

Draft Dated (4/18/94)

Reference book



The GenInteraction can contain a DocumentControl or other file-related commands.

GIGT_EDIT_MENU

This indicates that this GenInteraction acts as the Edit menu. The GenInteraction can contain an EditControl or other edit commands.

GIGT_VIEW_MENU

This indicates that this GenInteraction acts as the View menu. The GenInteraction can contain a ViewControl or other view commands.

GIGT_OPTIONS_MENU

This indicates that this GenInteraction acts as the Options menu. The GenInteraction can contain application options.

GIGT_WINDOW_MENU

This indicates that this GenInteraction acts as the Window menu. The GenInteraction can contain a GenDisplayControl or other window commands.

GIGT_HELP_MENU

This indicates that this GenInteraction acts as the Help menu.

GIGT_PRINT_GROUP

This indicates that this GenInteraction acts as the Print group. The GenInteraction can contain a PrintControl or other print commands.

Library: **Objects/gInterC.def**

■ GenInteractionType

GenInteractionType etype byte

GIT_ORGANIZATIONAL	enum GenInteractionType
GIT_PROPERTIES	enum GenInteractionType
GIT_PROGRESS	enum GenInteractionType
GIT_COMMAND	enum GenInteractionType
GIT_NOTIFICATION	enum GenInteractionType
GIT_AFFIRMATION	enum GenInteractionType
GIT_MULTIPLE_RESPONSE	enum GenInteractionType

GIT_ORGANIZATIONAL

This indicates that this GenInteraction is only used for grouping its children. This type has two chief uses:

- 1) For geometry purposes.
Collecting a group of generic objects together under a GenInteraction allows you to specify geometry for that group

via hints. The group will be dealt with as a single entity for geometry purposes.

2) For holding other GenInteractions.

Organizational interactions can act as a place-holder to put non-user-initiatable dialogs. This leaves the dialogs in the generic tree in a non-visible location where they can be initiated. As you generally don't want this organizational interaction to be visible itself, you will most likely want to mark it as a GIV_DIALOG and GIA_NOT_USER_INITIATABLE as well. (Optimization note: This also has the side effect of avoiding the loading in of the resources that the child dialog boxes reside in, should this object receive MSG_SPEC_BUILD.);

This type does not support a reply bar, even if GIV_DIALOG is used to specify that this GenInteraction should become a dialog box. OSF/Motif will ensure that there is a way for the user to dismiss any GIT_ORGANIZATIONAL interaction that does becomes a visible dialog.

GIT_PROPERTIES

This indicates that this interaction contains UI used for object properties. This type supports the IC_APPLY and IC_RESET **InteractionCommand** types which can be used for apply and reset functionality. If the interaction is built as a dialog box, the specific UI (such as OSF/Motif) will create "Apply" and "Close" triggers that reference the IC_APPLY and IC_DISMISS interaction commands.

In OSF/Motif, the following table shows the standard triggers that will be provided with various hints (given a GIT_PROPERTIES GenInteraction that is built as a dialog box):

GIT_PROGRESS

This indicates that this interaction is used to report progress information relating to an operation. If the interaction is built as a dialog box, the specific UI (such as OSF/Motif) may create a "Stop" trigger that contains the IC_STOP interaction command. Input hold up, ignore, & busy states are overridden by default to allow users to interact with modal implementations of this style GenInteraction.

GIT_COMMAND

This indicates that this interaction contains commands to other parts of the application. If the interaction is built as a dialog box, the specific UI (such as OSF/Motif) may create a "Close" trigger that contains the IC_DISMISS interaction command. Any additional command triggers must be provided with ATTR_GEN_TRIGGER_INTERACTION_COMMAND and HINT_SEEK_REPLY_BAR.

Revision: ■

Draft Dated (4/18/94)

Reference book



Table 12-1 *Default Triggers supplied with a GIT_PROPERTIES dialog box.*

Hint	Triggers Supplied
None (delayed properties)	“Apply” “Close”
_UNRELATED_PROPERTIES or _FAST_RESPONSE_PROPERTIES	None (Immediate properties)
_RELATED_PROPERTIES, _SLOW_RESPONSE_PROPERTIES or _REQUIRES_VALIDATION	“Apply” “Close”
_COMPLEX_PROPERTIES *and* (_RELATED_PROPERTIES, _SLOW_RESPONSE_PROPERTIES or _REQUIRES_VALIDATION)	“Apply” “Reset” “Close”
_SINGLE_USAGE *and* (_RELATED_PROPERTIES, _SLOW_RESPONSE_PROPERTIES or _REQUIRES_VALIDATION)	“OK” “Cancel”
_SINGLE_USAGE *and* (_COMPLEX_PROPERTIES, _SLOW_RESPONSE_PROPERTIES or _REQUIRES_VALIDATION)	“OK” “Reset” “Cancel”
Modal Dialog Box	“OK” “Cancel”
Modal Dialog and _COMPLEX_PROPERTIES	“OK” “Reset” “Cancel”

GIT_NOTIFICATION

This indicates that this interaction is used to report notification of events. When built as a dialog box, certain specific UIs (such as OSF/Motif) may create an “OK” trigger that contains the IC_OK interaction command. The specific UI will not provide “Close” or “Cancel” triggers to dismiss the interaction without a response from the user. Input hold up, ignore, and busy states are overridden by default to allow users to interact with modal implementations of this type of GenInteraction.

GIT_AFFIRMATION

This indicates that this interaction is used to ask the user to confirm an operation. The interaction should include a prompt in the form of a question. When built as a dialog box, certain specific UIs (such as OSF/Motif) may create “Yes” and “No” triggers that contain the IC_YES and IC_NO interaction

commands. The specific UI will not provide “Close” or “Cancel” triggers to dismiss the interaction without a user response. Input hold up, ignore, and busy states are overridden by default to allow users to interact with modal implementations of this type of GenInteraction.

GIT_MULTIPLE_RESPONSE

This indicates that this interaction contains multiple responses (two or more) of which one must be chosen. When built as a dialog box, you will need to add custom response triggers with

ATTR_GEN_TRIGGER_INTERACTION_COMMAND and

HINT_SEEK_REPLY_BAR. The specific UI will not provide “Close” or “Cancel” triggers to dismiss the interaction without a user response. Input hold up, ignore, and busy states are overridden by default to allow users to interact with modal implementations of this type of GenInteraction.

Library: **Objects/gInterC.def**

■ GenInteractionVisibility

GenInteractionVisibility etype byte

GIV_NO_PREFERENCE	enum GenInteractionVisibility
GIV_POPUP	enum GenInteractionVisibility
GIV_SUB_GROUP	enum GenInteractionVisibility
GIV_CONTROL_GROUP	enum GenInteractionVisibility
GIV_DIALOG	enum GenInteractionVisibility
GIV_POPOUT	enum GenInteractionVisibility

GIV_NO_PREFERENCE

This type specifies no visual preference for this interaction. The specific UI will determine how this GenInteraction should appear depending on hints, location in the visual/generic tree, and the type of generic children, etc.

GIV_POPUP

This type specifies that this interaction should appear as a popup (menu or popup list). Popups are normally hidden from view until activated and then only remain on-screen for the duration of the operation. They are not, in general, independently displayable windows.

GIV_SUB_GROUP

This type specifies that this interaction should appear as a sub-group within a larger window. The specific visual implementation of a sub-group depends on the visual implementation of the parent.

GIV_CONTROL_GROUP

This type specifies that this interaction contains controls and therefore should not be placed directly within a popup

Revision: ■

Draft Dated (4/18/94)

Reference book



interaction. The interaction may be built out as a sub-group within a larger window or as a separate dialog box.

GIV_DIALOG

This type specifies that this interaction should appear as a dialog box. The specific UI will create an activation trigger that brings up the dialog box unless GIA_NOT_USER_INITIATABLE is set. This trigger will appear in the location a normal child would appear.

GIV_POPOUT

This type specifies that this interaction can be 'popped out' into a dialog box from a sub-group implementation. The interaction will normally behave as a GIV_SUB_GROUP until it is 'popped out'. Then it behaves as a GIV_DIALOG.

MSG_GEN_INTERACTION_POP_IN and

MSG_GEN_INTERACTION_POP_OUT may be used to 'pop' the interaction in and out.

ATTR_GEN_INTERACTION_POPPED_OUT is used to indicate that the interaction is popped out. If set within the .ui file, the interaction will be popped out upon startup.

Library: **Objects/gInterC.def**

■ GenItemGroupBehaviorType

GenItemGroupBehaviorType etype byte, 0

GIGBT_EXCLUSIVE	enum GenItemGroupBehaviorType
GIGBT_EXCLUSIVE_NONE	enum GenItemGroupBehaviorType
GIGBT_EXTENDED_SELECTION	enum GenItemGroupBehaviorType
GIGBT_NON_EXCLUSIVE	enum GenItemGroupBehaviorType

GIGBT_EXCLUSIVE

This type specifies an exclusive selection list. In this mode, one and only one item may be selected at any time; anytime the user selects one item, any other will become deselected. The user may also not deselect a currently selected item.

GIGBT_EXCLUSIVE_NONE

This type specifies an exclusive-none selection list. In this mode, a user can de-select an already selected item, leaving the list with no items selected. GenItemGroups can show a none-selected state by returning GIGS_NONE (-1) in place of the selected item's identifier.

GIGBT_EXTENDED_SELECTION

This type specifies an exclusive selection list, but also contains the ability to extend the selection of items. If the user drags across items, or extends the selection (usually done by holding

a key down while clicking) several items can be selected. This is sometimes useful for selecting a target for an operation where choosing one item is good enough for a novice but selecting multiple items can be useful for an experienced user. As is the case with GIGBT_NON_EXCLUSIVE, an application will need to deal with sending and receiving item groups of identifiers for multiply selected items.

GIGBT_NON_EXCLUSIVE
 This type specifies a non-exclusive selection list, allowing the user to select multiple items with no constraints. If you have less than 16 GenItems, you may want to consider a GenBooleanGroup instead. Use MSG_GEN_ITEM_GROUP_GET_MULTIPLE_SELECTIONS and MSG_GEN_ITEM_GROUP_SET_MULTIPLE_SELECTIONS to handle multiple selections.

Library: **Objects/gItemGC.def**

■ **GenItemGroupStateFlags**

```
GenItemGroupStateFlags    record
    GIGSF_INDETERMINATE    :1
    GIGSF_MODIFIED         :1
                           :6
GenItemGroupStateFlags    end
```

GIGSF_INDETERMINATE
 This flag is set if the current selection is indeterminate. The current selection in this case refers to the initial state at the beginning of the data being represented. This indeterminate state refers to the item group as a whole even if in non-exclusive mode.

GIGSF_MODIFIED
 As stored in instance data, and sent in MSG_GEN_APPLY:
 This flag is set whenever the group itself should be marked as “modified.” This flag is cleared anytime the state is set and the flag should be set anytime the user changes the state of the group. Redundant selections will not change an item group’s state in GIGBT_EXCLUSIVE or GIGBT_EXTENDED_SELECTION style groups. This flag is also cleared when the item group receives MSG_GEN_APPLY or any selection setting message from the application. This state may further be set using MSG_GEN_ITEM_GROUP_SET_MODIFIED_STATE. It may be checked using MSG_GEN_ITEM_GROUP_IS_MODIFIED. The apply message is normally only sent out on MSG_GEN_PPLY if

this bit is non-zero, though this behavior can be overridden using ATTR_GEN_SEND_APPLY_MSG_ON_APPLY_EVEN_IF_NOT_MODIFIED.

As sent in status message:

GIGSF_MODIFIED will be set if the user has done something to change the state of the item group. If the user just clicks on the current selection in a GIGBT_EXCLUSIVE or GIGBT_EXTENDED_SELECTION item group, then this bit will be clear. If message is the result of a MSG_GEN_ITEM_GROUP_SEND_STATUS_MSG being sent, then this bit will be the value passed in that message.

Library: **Objects/gItemGC.def**

■ GenItemGroupUpdateExtSelParams

```
GenItemGroupUpdateExtSelParams    struc
    GIGUESP_anchorItem            word    ;anchor item
    GIGUESP_extentItem            word    ;extent item
    GIGUESP_prevExtentItem        word    ;previous extent item
    GIGUESP_setSelMsg             word    ;message to send to change items
                                      ; dh non-zero to select
                                      ; dh zero to de-select.

    GIGUESP_flags                 ExtSelFlags
    GIGUESP_passFlags             byte    ;internal: flags to pass to
                                      ; message in dl
GenItemGroupUpdateExtSelParams    ends
```

Library: **Objects/gItemGC.def**

■ GenMonikerMessageFrame

```
GenMonikerMessageFrame    struct
    GMMF_xInset             word
    GMMF_yInset             word
    GMMF_xMaximum           word
    GMMF_yMaximum           word
    GMMF_gState             hptr.GState
    GMMF_textHeight         word
    GMMF_monikerFlags       DrawMonikerFlags
GenMonikerMessageFrame    ends
```

This structure stores the parameters used in MSG_GEN_DRAW_MONIKER and MSG_GEN_GET_MONIKER_POS.

GMMF_xInset stores the x inset to the start of where to draw the moniker, if top or bottom justifying.

GMMF_yInset stores the y inset to the start of where to draw the moniker, if left or right justifying.

GMMF_xMaximum and *GMMF_yMaximum* store the maximum size of the moniker, if VMF_CLIP_TO_MAXIMUM_WIDTH is set in the GMFP_monikerFlags. Moniker will be clipped to that width.

GMMF_gState stores the **GState** to use to draw the moniker.

GMMF_textHeight stores the height of the system text, if we happen to have this information available. (This enables the messages to be processed faster; otherwise pass 0.)

GMMF_monikerFlags stores the various justification and miscellaneous flags for drawing the moniker.

Library: **Objects/genC.def**

■ GenOptionsParams

```
GenOptionsParams    struct
  GOP_category      char INI_CATEGORY_BUFFER_SIZE dup (?)
  GOP_key           char INI_CATEGORY_BUFFER_SIZE dup (?)
GenOptionsParams    ends
```

Library: **Objects/genC.def**

■ GenPathDiskRestoreArgs

```
GenPathDiskRestoreArgs  struct
  GPDRA_pathType      word
  GPDRA_savedDiskType word
  GPDRA_driveName     fptr.char
  GPDRA_diskName      fptr.char
  GPDRA_errorCode     DiskRestoreError
GenPathDiskRestoreArgs  ends
```

GPDRA_pathType stores the vardata tag under which the path itself is stored.

GPDRA_savedDiskType stores the vardata tag under which the disk handle is saved.

GPDRA_driveName stores the drive name (in a null-terminated text string with a trailing '.').

GPDRA_diskName stores the disk name in a null-terminated text string.

Revision: ■

Draft Dated (4/18/94)

Reference book



GPDRA_errorCode stores the error code that will be returned to **DiskRestore**.

Library: **Objects/genC.def**

■ GenSaveWindowInfo

```
GenSaveWindowInfo  struct
    GSWI_winPosition    SpecWinSizePair
    GSWI_winSize        SpecWinSizePair
    GSWI_winPosSizeState word
GenSaveWindowInfo  ends
```

Library: **Objects/genC.def**

■ GenScanItemsFlags

```
GenScanItemsFlags  record
    GSIF_FROM_START          :1
    GSIF_FORWARD              :1
    GSIF_WRAP_AROUND         :1
    GSIF_INITIAL_ITEM_FOUND   :1
    GSIF_USABLE_AND_ENABLED_ITEM_FOUND :1
    GSIF_EXISTING_ITEMS_ONLY  :1
    GSIF_DYNAMIC_LIST         :1
GenScanItemsFlags  end
```

GSIF_FROM_START

Set if there is no initial item. This allows us to easily get to the beginning (with GSIF_FORWARD set) or end (with GSIF_FORWARD clear) of the list by passing zero for the scan amount.

GSIF_FORWARD

Direction of scan. The scan amount passed will be negated if this is set.

GSIF_WRAP_AROUND

Flag for wrapping around to the beginning if we go past the end, or to the end if we go past the beginning.

GSIF_INITIAL_ITEM_FOUND

Internal.

GSIF_USABLE_AND_ENABLED_ITEM_FOUND

Internals.

GSIF_EXISTING_ITEMS_ONLY

Look through currently existing items only, even if in a dynamic list.

GSIF_DYNAMIC_LIST

Set if item group handler is dealing with a dynamic list, so that when the scan fails, it returns various data back to the dynamic list rather than returning the first or last item.

Library: **Objects/gItemGC.def**

■ GenStates

```
GenStates          record
  GS_USABLE        :1
  GS_ENABLED       :1
                  :6
GenStates          end
```

GS_USABLE

This flag is setable by the application and indicates whether the entire generic branch starting with this object should be considered part of the application's user interface at this time. If this bit is clear, then neither the object nor any of its children will appear or may be interacted with. The specific UI and visual state of any object which is made NOT_USABLE will be destroyed and the object will be treated as if it were in generic form only.

GS_ENABLED

This flag indicates whether the user can directly interact with an object. This flag is used in generic objects to show options not currently available, which is typically represented by "greying out" the object's moniker.

Library: **Objects/genC.def**

■ GenTextAttrs

```
GenTextAttrs      record
  GTA_SINGLE_LINE_TEXT      :1
  GTA_USE_TAB_FOR_NAVIGATION :1
  GTA_INIT_SCROLLING        :1
  GTA_NO_WORD_WRAPPING      :1
  GTA_ALLOW_TEXT_OFF_END    :1
  GTA_TAIL_ORIENTED         :1
  GTA_DONT_SCROLL_TO_CHANGES :1
                          :1
GenTextAttrs      end
```

GTA_SINGLE_LINE_TEXT

This flag indicates that the text has zero or only one carriage

Revision: ■

Draft Dated (4/18/94)

Reference book



return. Scrolling in some specific UIs (such as OpenLook) gets implemented horizontally if this is set.

GTA_USE_TAB_FOR_NAVIGATION

This flag indicates that TAB is used to navigate around your text's parent window, rather than inserted in the text field. For simple text objects you will nearly always want this.

GTA_INIT_SCROLLING

This flag forces the text object into a scrolling text area. This flag supersedes other size flags. The scrolling box is allowed to be resized vertically from a height of one upward.

GTA_NO_WORD_WRAPPING

This flag disallows word wrapping.

GTA_ALLOW_TEXT_OFF_END

This flag is set if text may overflow past the end of the text box.

GTA_TAIL_ORIENTED

This flag is set if we prefer the tail end of the text to be visible over the top end, given that option. In a scrolling text box, this means we always keep the end of the tail visible while text is being added or deleted at the bottom of the text field, if the end of the text field is currently visible.

GTA_DONT_SCROLL_TO_CHANGES

Usually if there is a scrollable text field, any insertion or deletion of text will be made visible, by scrolling the text if necessary. Setting this flag will turn this behavior off; text can be getting inserted at the end of a document without automatically scrolling there.

Library: **Objects/gTextC.def**

■ **GenTextCustomMargins**

```
GenTextCustomMargins    struc
    GTCM_lrMargin    byte    ;margin on the left and right of the text.
    GTCM_rbMargin    byte    ;margin on the top and bottom of the text.
GenTextCustomMargins    ends
```

Library: **Objects/gTextC.def**

■ GenTextStateFlags

```
GenTextStateFlags  record
    GTSF_INDETERMINATE    :1
    GTSF_MODIFIED         :1
                        :6
GenTextStateFlags  end
```

GTSF_INDETERMINATE

This flag is set if the current text is indeterminate. This means that for whatever data is being represented, there is more than one text. *GTXI_text* in this case should refer to the state at the beginning of the data being represented.

GTSF_MODIFIED

This flag stores a GenText's modified state:

As stored in instance data and sent out in MSG_GEN_APPLY:

This flag is cleared anytime the object's state is set and this flag is set anytime the user changes the state of the object . The flag is also automatically cleared on MSG_GEN_APPLY or MSG_GEN_TEXT_SET_TEXT. This state may be manually modified using MSG_GEN_TEXT_SET_MODIFIED_STATE. It may be checked using MSG_GEN_TEXT_IS_MODIFIED. The apply message is normally only sent out on MSG_GEN_APPLY if this bit is non-zero, though this behavior can be overridden using ATTR_GEN_SEND_APPLY_MSG_ON_APPLY_EVEN_IF_NOT_MODIFIED.

As sent in status message:

GVSF_MODIFIED will be set if the user has done something to change the state of the item group. If message is the result of a MSG_GEN_TEXT_SEND_STATUS_MSG being sent, then this bit will be passed in that message.

Library: **Objects/gTextC.def**

■ GenUpwardQueryType

```
GenUpwardQueryType  etype word
    GUQT_UI_FOR_APPLICATION  enum GenUpwardQueryType
    GUQT_UI_FOR_SCREEN      enum GenUpwardQueryType
    GUQT_UI_FOR_FIELD       enum GenUpwardQueryType
    GUQT_UI_FOR_MISC        enum GenUpwardQueryType
```

Library: **Objects/genC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GenValueDisplayFormat

GenValueDisplayFormat	etype byte
GVDF_INTEGER	enum GenValueDisplayFormat
GVDF_DECIMAL	enum GenValueDisplayFormat
GVDF_POINTS	enum GenValueDisplayFormat
GVDF_INCHES	enum GenValueDisplayFormat
GVDF_CENTIMETERS	enum GenValueDisplayFormat
GVDF_MILLIMETERS	enum GenValueDisplayFormat
GVDF_PICAS	enum GenValueDisplayFormat
GVDF_EUR_POINTS	enum GenValueDisplayFormat
GVDF_CICEROS	enum GenValueDisplayFormat
GVDF_POINTS_OR_MILLIMETERS	enum GenValueDisplayFormat
GVDF_INCHES_OR_CENTIMETERS	enum GenValueDisplayFormat

GVDF_INTEGER

Value will be displayed as an integer value. The value will be displayed with no decimal places, regardless of any fraction in the current value or the presence of ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_DECIMAL

Value will be displayed as a decimal value. By default, the value's fraction, if any, will be displayed using 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_POINTS

Value will be displayed using distance units, in points, regardless of whether metric or US units are set for the given application. The stored value is considered to be in units of "Points" (i.e. 1/72 of an inch). By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_INCHES

Value will be displayed using distance units, in inches, regardless of whether metric or US units are set for the given application. The stored value is considered to be in units of "Points" (i.e. 1/72 of an inch) but is translated into inches for display in the value's text field. By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_CENTIMETERS

Value will be displayed using distance units, in centimeters, regardless of whether metric or US units are set for the given application. The stored value is considered to be in units of

“Points” (i.e. 1/72 of an inch) but is translated into centimeters for display in the value’s text field. By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_MILLIMETERS

Value will be displayed using distance units, in millimeters, regardless of whether metric or US units are set for the given application. The stored value is considered to be in units of “Points” (i.e. 1/72 of an inch) but is translated into millimeters for display in the value’s text field. By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_PICAS

Value will be displayed using distance units, in picas, regardless of whether metric or US units are set for the given application. The stored value is considered to be in units of “Points” (i.e. 1/72 of an inch) but is translated into picas for display in the value’s text field. By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_EUR_POINTS

Value will be displayed using distance units, in European points, regardless of whether metric or US units are set for the given application. The stored value is considered to be in units of “Points” (i.e. 1/72 of an inch) but is translated into European points for display in the value’s text field. By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_CICEROS

Value will be displayed using distance units, in ciceros, regardless of whether metric or US units are set for the given application. The stored value is considered to be in units of “Points” (i.e. 1/72 of an inch) but is translated into Ciceros for display in the value’s text field. By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_POINTS_OR_MILLIMETERS

Value will be displayed using distance units, as points or millimeters, depending on whether metric or US units are set for the given application. The stored value is considered to be in units of “Points” (i.e. 1/72 of an inch) but is translated into points of millimeters for display in the value’s text field. By



default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

GVDF_INCHES_OR_CENTIMETERS

Value will be displayed using distance units, as inches or centimeters, depending on whether metric or US units are set for the given application. The stored value is considered to be in units of “Points” (i.e. 1/72 of an inch) but is translated into inches or millimeters for display in the value’s text field. By default, the value will be displayed with up to 3 decimal places. This can be changed with ATTR_GEN_VALUE_DECIMAL_PLACES.

Library: **Objects/gValueC.def**

■ GenValueIntervals

```
GenValueIntervals  struc
    GVI_numMajorIntervals    word    ; Total number of major intervals
                                ; to display over the range.
    GVI_numMinorIntervals    word    ; Total number of minor intervals
                                ; to display over the range.
GenValueIntervals  ends
```

Library: **Objects/gValueC.def**

■ GenValueStateFlags

```
GenValueStateFlags  record
    GVSF_INDETERMINATE      :1
    GVSF_MODIFIED           :1
    GVSF_OUT_OF_DATE        :1
                                :5
GenValueStateFlags  end
```

GVSF_INDETERMINATE

This flag is set if the current value is indeterminate. *GVLL* value in this case should refer to the initial state of the GenValue before it was set indeterminate.

GVSF_MODIFIED

This flag is set if the value has been modified.

GVSF_OUT_OF_DATE

This flag is set when the GenValue object’s internal value is out of date with what has been typed by the user. The GenValue object does not update its internal value on every user key

press, since the typed value may be temporarily out-of-range as they type. The GenValue is then updated on a return press ;by the user, a query for the value, on an increment or decrement, or as it is being taken offscreen. The GVSF_OUT_OF_DATE flag is most useful on status messages, to instruct the status message recipient to not try to process the value passed. This allows the GenValue to send a status message when the user first types in a GenValue, in order to let the recipient know that the GenValue has been modified.

As stored in instance data, and sent in MSG_GEN_APPLY:

This modified bit is cleared anytime the object's state is set; the flag is set anytime the user changes the state of the object. The flag is automatically cleared on MSG_GEN_APPLY or MSG_GEN_VALUE_SET_VALUE. This state may further be modified using MSG_GEN_VALUE_SET_MODIFIED_STATE. It may be checked using MSG_GEN_VALUE_IS_MODIFIED. The apply message is normally only sent out on MSG_GEN_APPLY if this bit is non-zero, though this behavior can be overridden using ATTR_GEN_SEND_APPLY_MSG_ON_APPLY_EVEN_IF_NOT_MODIFIED

As sent in status message:

GVSF_MODIFIED will be set if the user has done something to change the state of the GenValue. If the status message is the result of a MSG_GEN_VALUE_SEND_STATUS_MSG being sent, then this bit will be the value passed in that message.

GVSF_OUT_OF_DATE

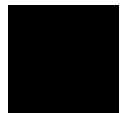
This flag is set when the GenValue object's internal value is out of date with what has been typed by the user. The GenValue object does not update its internal value on every user key press, since the typed value may be temporarily out-of-range as they type. The GenValue is then updated on a return press by the user, a query for the value, on an increment or decrement, or as it is being taken offscreen. The GVSF_OUT_OF_DATE flag is most useful on status messages, to instruct the status message recipient to not try to process the value passed. This allows the GenValue to send a status message when the user first types in a GenValue, in order to let the recipient know that the GenValue has been modified.

Library: **Objects/gValueC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GenValueType

```

GenValueType      etype word
    GVT_VALUE      enum GenValueType      ; The current value
    GVT_MINIMUM     enum GenValueType      ; The minimum value
    GVT_MAXIMUM     enum GenValueType      ; The maximum value
    GVT_INCREMENT   enum GenValueType      ; The increment value
    GVT_LONG        enum GenValueType      ; The longest value we can
                                           ; create
    GVT_RANGE_LENGTH enum GenValueType      ; The end of the displayed
                                           ; range, if applicable
    GVT_RANGE_END   enum GenValueType      ; The last value in the range,
                                           ; if applicable
    GVT_VALUE_AS_RATIO_OF_AVAILABLE_RANGE
                                           enum GenValueType      ; The current value, expressed
                                           ; as a ratio of max-min-
                                           ; range (if any).
```

Library: **Objects/gValueC.def**

■ GenViewAttrs

```

GenViewAttrs      record
/*
    * General GenView Attributes
    */
    GVA_CONTROLLED          :1
    GVA_GENERIC_CONTENTS    :1
    GVA_TRACK_SCROLLING     :1
    GVA_DRAG_SCROLLING      :1
    GVA_NO_WIN_FRAME        :1
    GVA_SAME_COLOR_AS_PARENT_WIN :1
    GVA_VIEW_FOLLOWS_CONTENT_GEOMETRY :1
/*
    * Attributes that follow may only be used if GVA_GENERIC_CONTENTS is
    * not set.
    */
    GVA_WINDOW_CCORDINATE_MOUSE_EVENTS :1
    GVA_DONT_SEND_PTR_EVENTS           :1
    GVA_DONT_SEND_KBD_RELEASES         :1
    GVA_SEND_ALL_KBD_CHARS              :1
    GVA_FOCUSABLE                      :1
    GVA_SCALE_TO_FIT                   :1
    GVA_ADJUST_FOR_ASPECT_RATIO        :1
    GVA_ADJUST_FOR_ASPECT_RATIO        :2
end
```

GVA_CONTROLLED

This flag is set if the view is connected to a controller object and

Revision: ■

Draft Dated (4/18/94)

Reference book

therefore should send out notification as appropriate and should add itself to the appropriate GCN list.

GVA_GENERIC_CONTENTS

This flag is set if the content of the GenView is a GenContentClass object (and its children are therefore generic objects, not visual objects). If this bit is set, the mouse grab mode and pointer event sending mode is set by the specific UI, overriding whatever is passed in the instance data. Other generic messages (such as MSG_SPEC_SPEC_BUILD_BRANCH) will be sent to the content as appropriate.

GVA_TRACK_SCROLLING

This flag is set if scrolling events should be sent to the content, so it can control more carefully where scrolling leaves the document origin. See MSG_META_CONTENT_TRACK_SCROLLING for more info.

GVA_DRAG_SCROLLING

This flag is set if so that the user can select and drag out of a subview window scrolling the window appropriately. This drag scrolling operates independently of the objects in the window; no special handling by the output objects should be needed.

GVA_NO_WIN_FRAME

This flag is set if the specific UI shouldn't draw a frame around the subview window.

GVA_SAME_COLOR_AS_PARENT_WIN

This flag is set if the background color of the view should be whatever color the parent window's background is. This flag should nearly always be set if GVA_GENERIC_CONTENTS is set, so the generic objects appear correctly underneath the view!

GVA_VIEW_FOLLOWS_CONTENT_GEOMETRY

This flag is set if the view, in a non-scrollable direction, should follow the size of the content. Content object must be running in the same thread as the view, so that MSG_VIS_RECALC_SIZE can be called on the content.

Attributes that may be used when GVA_GENERIC_CONTENTS is not set.

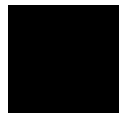
GVA_WINDOW_COORDINATE_MOUSE_EVENTS

This flag is set if mouse events should be sent in window coordinates instead of document coordinates (that is, as the offset in screen pixels from the upper left-hand corner of the view). This bit may be used transparently in conjunction with a VisContentClass content, by setting

Revision: ■

Draft Dated (4/18/94)

Reference book



VCNA_WINDOW_COORDINATE_MOUSE_EVENTS in the VisContent. This arrangement allows the VisContent to provide both 32-bit and fractional mouse data to visible objects within the content.

Alternatively, if the content is a process, then the process is responsible for converting incoming mouse data back into document coordinates. This may be done by storing the current document origin and scale factor as sent out by the View, and then using the equation:

$$Documentlocation = \frac{ViewWindowCoordinate}{ScaleFactor} + ViewOrigin$$

View Window Coordinate is the coordinate value passed in the 16-bit mouse event when GVA_WINDOW_COORDINATE_MOUSE_EVENTS is set in the GenView.

Scale Factor is current view scale factor, as sent to the process in MSG_META_CONTENT_VIEW_SCALE_FACTOR_CHANGED.

View Origin is the location in the document that currently appears at the upper left corner of the view window, as sent to the process in MSG_META_CONTENT_VIEW_ORIGIN_CHANGED.

Note: If *GVI_docBounds* lies outside of the 16-bit graphics space, then this flag MUST be used, since standard mouse events cannot pass 32-bit document positions.

GVA_DONT_SEND_PTR_EVENTS

This flag reflects an optimization to avoid sending pointer events to the application, if not needed.

GVA_DONT_SEND_KBD_RELEASES

This flag reflects an optimization to avoid sending keyboard releases to the application, if not needed.

GVA_SEND_ALL_KBD_CHARS

This flag forces all key presses to go to the content (if it has the focus), regardless of what those keypresses are. Usually the view will check first for mnemonics, accelerators, or other special specific UI keys, and will not pass the key press down if it gets handled by the UI in one of these ways. If applications set this flag, it is their responsibility in their MSG_META_KBD_CHAR handler to return a MSG_FUP_KBD_CHAR back to the view so it can finish the keyboard handling. Also, applications (such as GeoWrite) that intermix keypresses and other functions (such as changing the font or style) may have problems getting these messages in the correct order if the keypress has to go across threads to the content first, then back

Revision: ■

Draft Dated (4/18/94)

to the UI to check for accelerators. For this reason, a lot of applications may not want to use this flag. This flag also allows odd keyboard characters to be allowed as accelerators in generic objects. (Usually, only ctrl characters and a few others are acceptable as accelerators) In summary, the differences between each approach:

SEND_ALL_KBD_CHARS clear GenView gets key press, Checks ctrl chars for accelerators and other specific-UI actions. If char not used in these ways, sends it to content.	SEND_ALL_KBD_CHARS set GenView gets key press, Sends to content. Checks *all* chars FUPped back by content for accelerators and other specific-UI actions.
--	---

GVA_FOCUSABLE

This flag indicates that the view is allowed to have the focus. This flag is set by default upon instantiation or declaration in the **.ui** file. In general, you will only want to clear this flag for custom gadgets which should not be keyboard navigable.

GVA_SCALE_TO_FIT

This flag indicates that the view is operating in “scale to fit” mode. In this mode the *y* scale factor will be adjusted so that the entire document fits in the view, and the *x* scale factor will be adjusted accordingly. This behavior can be modified by either giving a page size (via ATTR_GEN_VIEW_PAGE_SIZE) to scale into the view, specifying that the document should fit entirely in *x* with the *y* scale factor following (via ATTR_GEN_VIEW_SCALE_TO_FIT_BASED_ON_X) or that both the document should fit in both *x* and *y* (via ATTR_GEN_VIEW_SCALE_TO_FIT_BOTH_DIMENSIONS).

GVA_ADJUST_FOR_ASPECT_RATIO

This flag indicates that scaling is adjusted to match the aspect ratio of the screen.

Library: **Objects/gViewC.def**

■ GenViewControlAttrs

```
GenViewControlAttrs      record
    GVCA_ADJUST_ASPECT_RATIO  :1
    GVCA_APPLY_TO_ALL         :1
    GVCA_SHOW_HORIZONTAL      :1
    GVCA_SHOW_VERTICAL        :1
                                :12
GenViewControlAttrs      end
```

Library: **Objects/gViewCC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GenViewControlSpecialScaleFactor

```
GenViewControlSpecialScaleFactor  etype word
  GVCSSF_TO_FIT  enum GenViewControlSpecialScaleFactor
```

Library: **Objects/gViewCC.def**

■ GenViewDimensionAttrs

```
GenViewDimensionAttrs  record
  GVDA_SCROLLABLE      :1
  GVDA_SPLITTABLE      :1
  GVDA_TAIL_ORIENTED   :1
  GVDA_DONT_DISPLAY_SCROLLBAR :1
  GVDA_NO_LARGER_THAN_CONTENT :1
  GVDA_NO_SMALLER_THAN_CONTENT :1
  GVDA_SIZE_A_MULTIPLE_OF_INCREMENT :1
  GVDA_KEEP_ASPECT_RATIO :1
GenViewDimensionAttrs  end
```

GVDA_SCROLLABLE

This flag is set if the view is scrollable in the given dimension. The view will force itself to be as big as its document size, so that nothing is obscured.

GVDA_SPLITTABLE

This flag is set if the view is splittable in the given dimension.

GVDA_TAIL_ORIENTED

This flag is set if the document prefers to be displayed at its end. The window will scroll to stay at the bottom of the document when you resize or change the document length, but only if you are currently at the bottom. If you move to the top or middle of the document no scrolling will be done. Currently, tail orientation does not work across threads; if you want to do this, you can try doing it via a tracking.

GVDA_DONT_DISPLAY_SCROLLBAR

This flag instructs the view to hide any scrollers in the given dimension, even if the view is scrollable.

GVDA_NO_LARGER_THAN_CONTENT

This flag is set if the view will not get larger than is needed to fit the content in the given dimension, based on the current value of *GVI_docBounds*. By default there are no restrictions on the size of the view.

GVDA_NO_SMALLER_THAN_CONTENT

This flag is set if the view will stay large enough to display the entire content in the given dimension, based on the current value of *GVI_docBounds*. By default there are no restrictions on the size of the view.

GVDA_SIZE_A_MULTIPLE_OF_INCREMENT

This flag is set if we want to truncate the view window's size in this direction to a multiple of the increment amount in this direction. Subclass *MSG_GEN_VIEW_CALC_WIN_SIZE* if you need finer adjustments of the view window size.

GVDA_KEEP_ASPECT_RATIO

This flag is set if we want to keep the aspect ratio of the port windows the same as they are in the open size. If set in *vertAttrs*, then we'll keep the width and use the ratio to calculate the height; and vice versa for *horizAttrs*.

Library: **Objects/gViewC.def**

■ GenViewInkType

```
GenViewInkType      etype byte
GVIT_PRESSES_ARE_NOT_INK      enum GenViewInkType
GVIT_INK_WITH_STANDARD_OVERRIDE      enum GenViewInkType
GVIT_PRESSES_ARE_INK      enum GenViewInkType
GVIT_QUERY_OUTPUT      enum GenViewInkType
```

GVIT_PRESSES_ARE_NOT_INK

The type specifies that the output of the view cannot handle ink. If you are using the large document model, you should subclass *MSG_NOTIFY_DATA_GROUP* with the notification type *GWNT_INK* on the *VisContent* attached to the view. Otherwise, ink will be sent to the first child with the target.

GVIT_INK_WITH_STANDARD_OVERRIDE

This type specifies that the output of the view can handle ink, but the user can override it by holding the mouse down for some user-specified amount of time before moving the mouse.

GVIT_PRESSES_ARE_INK

This type specifies that any mouse presses are ink and the output of the view can handle ink. Note: If you use the following value, you must handle *MSG_META_QUERY_IF_PRESS_IS_INK*. Note also that a *MSG_NOTIFY_DATA_GROUP* with the notification type of *GWNT_INK* can come even if no *MSG_META_QUERY_IF_PRESS_IS_INK* has been received. (This can happen if the user starts drawing just outside of the view but then draws inside the view.)

Revision: ■

Draft Dated (4/18/94)

Reference book



GVIT_QUERY_OUTPUT

This type specifies that the output of the view only wants presses to be ink under certain conditions. MSG_META_QUERY_IF_PRESS_IS_INK is still sent to the output.

Library: **Objects/gViewC.def**

■ GeodeAttrs

```
GeodeAttrs          record
    GA_PROCESS          :1      ; Has initial thread
    GA_LIBRARY          :1      ; Exports routines
    GA_DRIVER           :1      ; Has DriverTable
    GA_KEEP_FILE_OPEN   :1      ; .geo file must stay open (resource(s)
                                ; discardable or initially discarded)
    GA_SYSTEM           :1      ; Compiled into kernel
    GA_MULTI_LAUNCHABLE :1      ; May be loaded more than once
    GA_APPLICATION      :1      ; A user-launched application
    GA_DRIVER_INITIALIZED :1     ; If DRIVER aspect initialized (DR_INIT
                                ; sent to strategy routine)
    GA_LIBRARY_INITIALIZED :1    ; If LIBRARY aspect initialized
                                ; (library entry point called)
    GA_GEODE_INITIALIZED :1     ; If all aspects initialized.
    GA_USES_COPROC      :1      ; Uses coprocessor if available
    GA_REQUIRES_COPROC  :1      ; Requires coprocessor/emulator to run
    GA_HAS_GENERAL_CONSUMER_MODE :1 ; Can be run in GCM mode
    GA_ENTRY_POINTS_IN_C :1     ; Library/driver entry point in C
                                :2
GeodeAttrs          end
```

Library: **geode.def**

■ GeodeDefaultDriverType

```
GeodeDefaultDriverType etype word, 0, 2
    GDDT_FILE_SYSTEM    enum GeodeDefaultDriverType
    GDDT_KEYBOARD        enum GeodeDefaultDriverType
    GDDT_MOUSE           enum GeodeDefaultDriverType
    GDDT_VIDEO           enum GeodeDefaultDriverType
    GDDT_MEMORY_VIDEO    enum GeodeDefaultDriverType
    GDDT_POWER_MANAGEMENT enum GeodeDefaultDriverType
    GDDT_TASK            enum GeodeDefaultDriverType
```

Library: **driver.def**

■ GeodeGetInfoType

```
GeodeGetInfoType    etype word, 0, 2
    GGIT_ATTRIBUTES          enum GeodeGetInfoType
    GGIT_TYPE                enum GeodeGetInfoType
    GGIT_GEODE_RELEASE       enum GeodeGetInfoType
    GGIT_GEODE_PROTOCOL      enum GeodeGetInfoType
    GGIT_TOKEN_ID            enum GeodeGetInfoType
    GGIT_PERM_NAME_AND_EXT   enum GeodeGetInfoType
    GGIT_PERM_NAME_ONLY     enum GeodeGetInfoType
```

Library: **geode.def**

■ GeodeGrab

```
GeodeGrab          struct
    GG_OD           optr
    GG_geode        hptr
GeodeGrab          ends
```

This structure stores a top-level grab for controlling input flow to the geode.

Library: **Objects/uiInputC.def**

■ GeodeHeapVars

```
GeodeHeapVars      struc
    GHV_heapSpace          word
    ;
    ; Heap space requirement, as copied from EFH_heapSpace from the
    ; ExecutableFileHeader of applications. Roughly, the amount of space
    ; on the heap that this application uses, in paragraphs. The system
    ; sums the total of all "heapSpace" requirements when trying to decide
    ; whether to let another app load or not.
```

Library: **geode.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GeodeLoadError

GeodeLoadError	etype word	
GLE_PROTOCOL_IMPORTER_TOO_RECENT		enum GeodeLoadError
GLE_PROTOCOL_IMPORTER_TOO_OLD		enum GeodeLoadError
GLE_FILE_NOT_FOUND		enum GeodeLoadError
GLE_LIBRARY_NOT_FOUND		enum GeodeLoadError
GLE_FILE_READ_ERROR		enum GeodeLoadError
GLE_NOT_GEOS_FILE		enum GeodeLoadError
GLE_NOT_GEOS_EXECUTABLE_FILE		enum GeodeLoadError
GLE_ATTRIBUTE_MISMATCH		enum GeodeLoadError
GLE_MEMORY_ALLOCATION_ERROR		enum GeodeLoadError
GLE_NOT_MULTI_LAUNCHABLE		enum GeodeLoadError
GLE_LIBRARY_PROTOCOL_ERROR		enum GeodeLoadError
GLE_LIBRARY_LOAD_ERROR		enum GeodeLoadError
GLE_DRIVER_INIT_ERROR		enum GeodeLoadError
GLE_LIBRARY_INIT_ERROR		enum GeodeLoadError
GLE_DISK_TOO_FULL		enum GeodeLoadError
GLE_FIELD_DETACHING		enum GeodeLoadError
GLE_INSUFFICIENT_HEAP_SPACE		enum GeodeLoadError

Library: **geode.def**

■ GeodeToken

GeodeToken	struct
GT_chars	TokenChars
GT_manufID	ManufacturerID
GeodeToken	ends

This structure defines a token identifier for an application. Together the two fields uniquely identify the application.

GT_chars stores the four character identifier.

GT_manufID stores the **ManufacturerID**.

Library: **geode.def**

■ GeosFileHeaderFlags

```

GeosFileHeaderFlags      record
    GFHF_TEMPLATE        :1
    GFHF_SHARED_MULTIPLE :1 ; Also called "multi-user"
    GFHF_SHARED_SINGLE   :1 ; Also called "public"
                           :1
    GFHF_HIDDEN           :1 ; This file is hidden. This flag does
                           ; not replace the DOS "hidden"
                           ; attribute -- the two may be
                           ; set/cleared independently of
                           ; each-other.
    GFHF_DBCS             :1 ;TRUE: DBCS filename, etc.
                           :10
GeosFileHeaderFlags      end

```

Library: **file.def**

■ GeosFileType

```

GeosFileType      etype word
    GFT_NOT_GEOS_FILE      enum GeosFileType      ; Not a geos file. defined as
                                                ; 0 so one can reasonably
                                                ; look at FEA_FILE_TYPE

    GFT_EXECUTABLE         enum GeosFileType      ; Something we can execute
    GFT_VM                 enum GeosFileType      ; Managed by VMem
    GFT_DATA               enum GeosFileType      ; Raw byte-stream of data
    GFT_DIRECTORY          enum GeosFileType      ; Directory
    GFT_OLD_VM             enum GeosFileType      ; VM file from PC/GEOS 1.X.
                                                ; Only FEA_NAME and
                                                ; FEA_FILE_TYPE are supported
                                                ; from the set of
                                                ; GEOS-specific extended attrs

```

Library: **file.def**



■ GeoWorksGenAppGCNListType

```

GeoWorksGenAppGCNListType      etype word, FIRST_GEN_APP_GCN_LIST_TYPE, 2
;
; GenToolControl/GenControl communication related
;
GAGCNLT_SELF_LOAD_OPTIONS      enum GeoWorksGenAppGCNListType
GAGCNLT_GEN_CONTROL_NOTIFY_STATUS_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SELECT_STATE_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_EDIT_CONTROL_NOTIFY_UNDO_STATE_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_CHAR_ATTR_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_PARA_ATTR_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_TYPE_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_SELECTION_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_COUNT_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_STYLE_TEXT_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_STYLE_SHEET_TEXT_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_STYLE_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_FONT_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_POINT_SIZE_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_FONT_ATTR_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_JUSTIFICATION_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_FG_COLOR_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_BG_COLOR_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_PARA_COLOR_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_BORDER_COLOR_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SEARCH_SPELL_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SEARCH_REPLACE_CHANGE  enum GeoWorksGenAppGCNListType
;
; Chart related
GAGCNLT_APP_TARGET_NOTIFY_CHART_TYPE_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_CHART_GROUP_FLAGS  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_CHART_AXIS_ATTRIBUTES  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_CHART_MARKER_SHAPE  enum GeoWorksGenAppGCNListType
;
; Grobj related
GAGCNLT_APP_TARGET_NOTIFY_GROBJ_CURRENT_TOOL_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_GROBJ_BODY_SELECTION_STATE_CHANGE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_GROBJ_AREA_ATTR_CHANGE  enum GeoWorksGenAppGCNListType

```

```

GAGCNLT_APP_TARGET_NOTIFY_GROBJ_LINE_ATTR_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_GROBJ_TEXT_ATTR_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_STYLE_GROBJ_CHANGE enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_STYLE_SHEET_GROBJ_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_GROBJ_BODY_INSTRUCTION_FLAGS_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_GROBJ_GRADIENT_ATTR_CHANGE
enum GeoWorksGenAppGCNListType
;
; Ruler related
GAGCNLT_APP_TARGET_NOTIFY_RULER_TYPE_CHANGE enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_RULER_GRID_CHANGE enum GeoWorksGenAppGCNListType
GAGCNLT_TEXT_RULER_OBJECTS enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_RULER_GUIDE_CHANGE enum GeoWorksGenAppGCNListType
;
; VisBitmap related
GAGCNLT_APP_TARGET_NOTIFY_BITMAP_CURRENT_TOOL_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_BITMAP_CURRENT_FORMAT_CHANGE
enum GeoWorksGenAppGCNListType
;
; Flat file library related
GAGCNLT_APP_TARGET_NOTIFY_FLAT_FILE_FIELD_PROPERTIES_STATUS_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_FLAT_FILE_FIELD_LIST_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_FLAT_FILE_RCP_STATUS_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_FLAT_FILE_FIELD_APPEARANCE_CHANGE
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_FLAT_FILE_DUMMY_CHANGE_2
enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_FLAT_FILE_DUMMY_CHANGE_3
enum GeoWorksGenAppGCNListType
;
; Spool library related
GAGCNLT_APP_NOTIFY_DOC_SIZE_CHANGE enum GeoWorksGenAppGCNListType
GAGCNLT_APP_NOTIFY_PAPER_SIZE_CHANGE enum GeoWorksGenAppGCNListType
;
; Used in GenViewControl
GAGCNLT_APP_TARGET_NOTIFY_VIEW_STATE_CHANGE enum GeoWorksGenAppGCNListType
GAGCNLT_CONTROLLED_GEN_VIEW_OBJECTS enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_INK_STATE_CHANGE enum GeoWorksGenAppGCNListType
GAGCNLT_CONTROLLED_INK_OBJECTS enum GeoWorksGenAppGCNListType
;
; Float library related
GAGCNLT_APP_TARGET_NOTIFY_PAGE_STATE_CHANGE enum GeoWorksGenAppGCNListType

```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

;
; GenDocumentControl related
GAGCNLT_APP_TARGET_NOTIFY_DOCUMENT_CHANGE      enum GeoWorksGenAppGCNListType
;
; GenDisplayControl related
GAGCNLT_APP_TARGET_NOTIFY_DISPLAY_CHANGE        enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_DISPLAY_LIST_CHANGE  enum GeoWorksGenAppGCNListType
;
; Spline Library Notification Lists
GAGCNLT_APP_TARGET_NOTIFY_SPLINE_MARKER_SHAPE  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPLINE_POINT         enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPLINE_POLYLINE      enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPLINE_SMOOTHNESS    enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPLINE_OPEN_CLOSE_CHANGE
                                                enum GeoWorksGenAppGCNListType
;
; Spreadsheet Library Notification Lists
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_ACTIVE_CELL_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_EDIT_BAR_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_SELECTION_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_CELL_WIDTH_HEIGHT_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_DOC_ATTR_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_CELL_ATTR_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_CELL_NOTES_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_DATA_RANGE_CHANGE
                                                enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_TEXT_NAME_CHANGE     enum GeoWorksGenAppGCNListType
GAGCNLT_FLOAT_FORMAT_CHANGE                   enum GeoWorksGenAppGCNListType
GAGCNLT_DISPLAY_OBJECTS_WITH_RULERS           enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_APP_CHANGE          enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_LIBRARY_CHANGE       enum GeoWorksGenAppGCNListType
;
; UI Notification Lists
GAGCNLT_WINDOWS                              enum GeoWorksGenAppGCNListType
GAGCNLT_STARTUP_LOAD_OPTIONS                  enum GeoWorksGenAppGCNListType
;
; CARD LLibrary Notification Lists
GAGCNLT_APP_TARGET_NOTIFY_CARD_BACK_CHANGE    enum GeoWorksGenAppGCNListType
;
GAGCNLT_NOTIFY_FOCUS_TEXT_OBJECT              enum GeoWorksGenAppGCNListType
GAGCNLT_NOTIFY_TEXT_CONTEXT                   enum GeoWorksGenAppGCNListType
;
; Help Notification Lists

```



```
GAGCNLT_NOTIFY_HELP_CONTEXT_CHANGE      enum GeoWorksGenAppGCNListType
;
GAGCNLT_FLOAT_FORMAT_INIT                enum GeoWorksGenAppGCNListType
GAGCNLT_ALWAYS_INTERACTABLE_WINDOWS      enum GeoWorksGenAppGCNListType
GAGCNLT_USER_DO_DIALOGS                  enum GeoWorksGenAppGCNListType
GAGCNLT_MODAL_WIN_CHANGE                  enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_NAME_CHANGE
                                          enum GeoWorksGenAppGCNListType
GAGCNLT_CONTROLLERS_WITHIN_USER_DO_DIALOGS
                                          enum GeoWorksGenAppGCNListType
GAGCNLT_FOCUS_WINDOW_KBD_STATUS          enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_PAGE_INFO_STATE_CHANGE
                                          enum GeoWorksGenAppGCNListType
GAGCNLT_APP_TARGET_NOTIFY_CURSOR_POSITION_CHANGE
                                          enum GeoWorksGenAppGCNListType
```

The UI library's `GenApplicationClass` supports its very own GCN (General Change Notification) system separate from the kernel's. Lists within this system are identified by a **GAGCNListType**, whose enumerations are separate from that of the kernel's GCN system.

This section contains the enumerations of the *GCNLT_type* field for **GCNListType** used within a `GenApplication` for the case of *GCNLT_manuf* = `MANUFACTURER_ID_GEOWORKS`.

GAGCNLT_SELF_LOAD_OPTIONS

Objects on this list don't need to receive `MSG_META_LOAD_OPTIONS` on startup, but do need to receive `MSG_META_SAVE_OPTIONS`. (`MSG_META_SAVE_OPTIONS` will be sent when the `GenApplication` itself receives `MSG_META_SAVE_OPTIONS`.) This must be done by the application, usually when a "Save Options" trigger is activated. Objects on this list can be of any class, as `MetaClass` defines the options behavior, though objects will most likely be of a Generic UI class (`GenClass` provides the **.ini** file behavior).

Any controller not on the `GAGCNLT_STARTUP_LOAD_OPTIONS` list will need to be placed on this list. A controller should not appear in both lists.

Other objects on this list will be those that support options like `GenBooleanGroups`, `GenItemGroups`, or a `GenInteraction` with `ATTR_GEN_INIT_FILE_PROPAGATE_TO_CHILDREN`. These should have the appropriate `ATTR_GEN_INIT_FILE_KEY`. The `GenApplication` object should have an appropriate `ATTR_GEN_INIT_FILE_CATEGORY`.

GAGCNLT_GEN_CONTROL_NOTIFY_STATUS_CHANGE

Objects on this list (generally `GenToolControl` objects) are kept

Revision: ■

Draft Dated (4/18/94)

Reference book



up to date on the status of all GenControl objects. Any specific status events are *not* sent with this list -- only notification that a change has occurred. This notification passes the data type **NotifyGenControlStatusChange**.

GAGCNLT_APP_TARGET_NOTIFY_SELECT_STATE_CHANGE
Objects on this list receive notification about the selection state of the current target object. This notification passes the data type **NotifySelectStateChange**.

GCNLT_EDIT_CONTROL_NOTIFY_UNDO_STATE_CHANGE
Objects on this list receive notification of the undo state of the current process. This notification passes the data type **NotifyUndoStateChange**.

GAGCNLT_APP_TARGET_NOTIFY_TEXT_CHAR_ATTR_CHANGE
This notification passes the data type **VisTextNotifyCharAttrChange**.

GAGCNLT_APP_TARGET_NOTIFY_TEXT_PARA_ATTR_CHANGE
This notification passes the data type **VisTextNotifyParaAttrChange**.

GAGCNLT_APP_TARGET_NOTIFY_TEXT_TYPE_CHANGE
This notification passes the data type **VisTextNotifyTypeChange**.

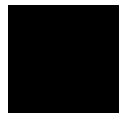
GAGCNLT_APP_TARGET_NOTIFY_TEXT_SELECTION_CHANGE
This notification passes the data type **VisTextNotifySelectionChange**.

GAGCNLT_APP_TARGET_NOTIFY_TEXT_COUNT_CHANGE
This notification passes the data type **VisTextNotifyCountChange**.

GAGCNLT_APP_TARGET_NOTIFY_STYLE_TEXT_CHANGE
Objects on this list receive notification when the current style could have changed (i.e. a different style could be the current style). This notification passes the data type **NotifyStyleChange**.

GAGCNLT_APP_TARGET_NOTIFY_STYLE_SHEET_TEXT_CHANGE
Objects on this list receive notification when the style sheet could have changed (i.e. a style was created, deleted or modified). This notification passes the data type **NotifyStyleSheetChange**.

- GAGCNLT_APP_TARGET_NOTIFY_TEXT_STYLE_CHANGE
This notification passes the data type **NotifyTextStyleChange**.
- GAGCNLT_APP_TARGET_NOTIFY_FONT_CHANGE
This notification passes the data type **NotifyFontChange**.
- GAGCNLT_APP_TARGET_NOTIFY_POINT_SIZE_CHANGE
This notification passes the data type **NotifyPointSizeChange**.
- GAGCNLT_APP_TARGET_NOTIFY_FONT_ATTR_CHANGE
This notification passes the data type **NotifyFontAttrChange**.
- GAGCNLT_APP_TARGET_NOTIFY_JUSTIFICATION_CHANGE
This notification passes the data type **NotifyJustificationChange**.
- GAGCNLT_APP_TARGET_NOTIFY_TEXT_FG_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GAGCNLT_APP_TARGET_NOTIFY_TEXT_BG_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GAGCNLT_APP_TARGET_NOTIFY_PARA_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GAGCNLT_APP_TARGET_NOTIFY_BORDER_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GAGCNLT_APP_NOTIFY_DOC_SIZE_CHANGE
This notification passes the data type **NotifyPageSetupChange**.
- GAGCNLT_APP_NOTIFY_PAPER_SIZE_CHANGE
This notification passes the data type **NotifyPageSetupChange**.
- GAGCNLT_CONTROLLED_GEN_VIEW_OBJECTS
Objects on this list are GenView objects that have the ATTR_GEN_VIEW_INTERACT_WITH_CONTROLLER attribute.
Note: If an object is not a GenView object, it shouldn't be on this list.
- GAGCNLT_APP_TARGET_NOTIFY_INK_STATE_CHANGE
Objects on this list (controllers) are notified when ink objects come up.



- GAGCNLT_CONTROLLED_INK_OBJECTS
Objects on this list are Ink objects that have the IF_CONTROLLED bit set. Note: If you aren't an Ink object, you shouldn't be on this list.
- GAGCNLT_APP_TARGET_NOTIFY_PAGE_STATE_CHANGE
Objects on this list are notified when the selection state of the current target object changes. This notification passes the data type **NotifyPageStateChange**
- GAGCNLT_APP_TARGET_NOTIFY_DOCUMENT_CHANGE
Objects on this list are notified about the state of the current target document. This notification passes the data type **NotifyPageStateChange**.
- GAGCNLT_APP_TARGET_NOTIFY_DISPLAY_CHANGE
Objects on this list are notified about the state of the current target display. This notification passes the data type **NotifyDisplayChange**.
- GAGCNLT_APP_TARGET_NOTIFY_DISPLAY_LIST_CHANGE
Objects on this list are notified about the state of the current target display. This notification passes the data type **NotifyDisplayChange**.
- GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_ACTIVE_CELL_CHANGE
This notification passes the data type **NotifySSheetActiveCellChanged**.
- GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_EDIT_BAR_CHANGE
This notification passes the data **NotifySSheetEditBarChanged**.
- GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_SELECTION_CHANGE
This notification passes the data type **NotifySSheetSelectionChanged**.
- GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_CELL_WIDTH_HEIGHT_CHANGE
This notification passes the data type **NotifySSheetCellWidthHeightChange**.
- GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_DOC_ATTR_CHANGE
This notification passes the data type **NotifySSheetDocAttrChange**.
- GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_CELL_ATTR_CHANGE
This notification passes the data type **NotifySSheetCellAttrChange**.

GAGCNLT_APP_TARGET_NOTIFY_SPREADSHEET_DATA_RANGE_CHANGE

This notification passes the data type
NotifySSheetDataRangeChange.

GAGCNLT_APP_TARGET_NOTIFY_TEXT_NAME_CHANGE

This notification passes the data type
VisTextNotifyNameChange.

GAGCNLT_DISPLAY_OBJECTS_WITH_RULERS

Objects on this list should include all GenDisplays that have rulers.

GAGCNLT_WINDOWS

Objects on this list are windowed objects that should be displayed on-screen. In your **.ui** file, any windows that you want to appear when the application starts up should be included on this list. In most cases, this will be only be a GenPrimary. When windows are opened, they are automatically added to this list. The list is saved to state and used to re-open all windows that were on-screen when the application shuts down. The only message sent to this list is MSG_META_UPDATE_WINDOW. The classes of objects on this list are GenPrimaryClass, GenDisplayClass, and GenInteractionClass. GenDisplayGroup does not need to be on this list, as it is not an independently displayable window.

GAGCNLT_STARTUP_LOAD_OPTIONS

Objects on this list need to receive MSG_META_LOAD_OPTIONS on startup. MSG_META_LOAD_OPTIONS is automatically sent by the UI when the application starts up.

MSG_META_SAVE_OPTIONS will be sent to the objects on the list when the GenApplicationitself receives a MSG_META_SAVE_OPTIONS. This must be done by the application, usually when a “Save Options” trigger is activated. Objects on this list can be of any class, as MetaClass defines the options behavior, though objects will most likely be generic objects (GenClass provides the **.ini** file behavior).

The GenViewControl must be placed on this list.

Other objects on this list will be those that support options like GenBooleanGroups, GenItemGroups, or a GenInteractions with ATTR_GEN_INIT_FILE_PROPAGATE_TO_CHILDREN. These should have the appropriate ATTR_GEN_INIT_FILE_KEY. The GenApplication object should have an appropriate ATTR_GEN_INIT_FILE_CATEGORY.

Revision: ■

Draft Dated (4/18/94)

Reference book



GAGCNLT_ALWAYS_INTERACTABLE_WINDOWS

Objects on this are windows that should always remain interactable (even when modal windows are on screen).

GAGCNLT_USER_DO_DIALOGS

Objects on this list include all dialog boxes initiated via **UserDoDialog**.

GAGCNLT_MODAL_WIN_CHANGE

Objects on this list need notification upon modal window changes within the application.

Library: **geoworks.def**

■ GeoWorksMetaGCNListType

```
GeoWorksMetaGCNListType  etype word, FIRST_META_GCN_LIST_TYPE, 2
MGCNLT_ACTIVE_LIST       enum GeoWorksMetaGCNListType
MGCNLT_APP_STARTUP       enum GeoWorksMetaGCNListType
```

MGCNLT_ACTIVE_LIST

Objects on this list need to receive MSG_META_ATTACH, MSG_META_DETACH, and/or MSG_META_QUIT. Currently, this represents only certain controllers. These controllers need MSG_META_ATTACH, so they should be placed on the MGCNLT_ACTIVE_LIST list in the **.ui** file. Other objects may add themselves dynamically to this list, if they only need to receive MSG_META_DETACH or MSG_META_QUIT.

These are the current controllers that need to be on the list:

GenToolControl
GenDocumentControl
GenDisplayControl
TextRulerControl
TabControl

Aside from controller-specific reasons, a controller would need to receive MSG_META_ATTACH (and thus would need to be on the active list) if it is always interactable (GCBF_ALWAYS_INTERACTABLE) or if it is always on its GCN lists (GCBF_ALWAYS_ON_GCN_LIST). Objects on this list may be of any class, as MSG_META_ATTACH, MSG_META_DETACH, and MSG_META_QUIT are defined for MetaClass.

MGCNLT_APP_STARTUP

Objects on this list need to receive MSG_META_APP_STARTUP and MSG_META_APP_SHUTDOWN to know when the application has been started or is about to exit regardless of whether the app will become/was available to the user. For

example, the GenDocumentControl needs to receive these messages so it can open and manipulate a passed document even when the application is launched in engine mode to perform some query on the document. Objects on this list may be of any class, as MSG_META_APP_STARTUP and MSG_META_APP_SHUTDOWN are defined for MetaClass.

Library: **geoworks.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GeoWorksNotificationType

```
GeoWorksNotificationType  etype word
    GWNT_INK                                enum GeoWorksNotificationType
    GWNT_GEN_CONTROL_NOTIFY_STATUS_CHANGE   enum GeoWorksNotificationType
    ;
    ;GenEditControl related.
    ;
    GWNT_SELECT_STATE_CHANGE                enum GeoWorksNotificationType
    GWNT_UNDO_STATE_CHANGE                  enum GeoWorksNotificationType
    ;
    ;StyleSheetControl related.
    ;
    GWNT_STYLE_CHANGE                       enum GeoWorksNotificationType
    GWNT_STYLE_SHEET_CHANGE                 enum GeoWorksNotificationType
    ;
    ;High-level types for the Text object
    ;
    GWNT_TEXT_CHAR_ATTR_CHANGE              enum GeoWorksNotificationType
    GWNT_TEXT_PARA_ATTR_CHANGE              enum GeoWorksNotificationType
    GWNT_TEXT_TYPE_CHANGE                   enum GeoWorksNotificationType
    GWNT_TEXT_SELECTION_CHANGE              enum GeoWorksNotificationType
    GWNT_TEXT_COUNT_CHANGE                  enum GeoWorksNotificationType
    ;
    ;Low-level types for the Text object
    ;
    GWNT_TEXT_STYLE_CHANGE                  enum GeoWorksNotificationType
    GWNT_FONT_CHANGE                        enum GeoWorksNotificationType
    GWNT_POINT_SIZE_CHANGE                  enum GeoWorksNotificationType
    GWNT_FONT_ATTR_CHANGE                   enum GeoWorksNotificationType
    GWNT_JUSTIFICATION_CHANGE                enum GeoWorksNotificationType
    GWNT_TEXT_FG_COLOR_CHANGE               enum GeoWorksNotificationType
    GWNT_TEXT_BG_COLOR_CHANGE               enum GeoWorksNotificationType
    GWNT_TEXT_PARA_COLOR_CHANGE              enum GeoWorksNotificationType
    GWNT_TEXT_BORDER_COLOR_CHANGE            enum GeoWorksNotificationType
    GWNT_SEARCH_REPLACE_ENABLE_CHANGE        enum GeoWorksNotificationType
    GWNT_SPELL_ENABLE_CHANGE                 enum GeoWorksNotificationType
    ;
    ;Chart library notification types
    ;
    GWNT_CHART_TYPE_CHANGE                  enum GeoWorksNotificationType
    GWNT_CHART_GROUP_FLAGS                  enum GeoWorksNotificationType
    GWNT_CHART_AXIS_ATTRIBUTES              enum GeoWorksNotificationType
    ;
    ;Grobj library notification types
    ;
    GWNT_GROBJ_CURRENT_TOOL_CHANGE           enum GeoWorksNotificationType
    GWNT_GROBJ_BODY_SELECTION_STATE_CHANGE   enum GeoWorksNotificationType
    GWNT_GROBJ_AREA_ATTR_CHANGE             enum GeoWorksNotificationType
    GWNT_GROBJ_LINE_ATTR_CHANGE             enum GeoWorksNotificationType
```

```

GWNT_GROBJ_TEXT_ATTR_CHANGE          enum      GeoWorksNotificationType
GWNT_GROBJ_BODY_INSTRUCTION_FLAGS_CHANGE  enum      GeoWorksNotificationType
GWNT_GROBJ_GRADIENT_ATTR_CHANGE        enum      GeoWorksNotificationType
;
;Ruler library notification types
;
GWNT_RULER_TYPE_CHANGE                 enum      GeoWorksNotificationType
GWNT_RULER_GRID_CHANGE                 enum      GeoWorksNotificationType
GWNT_RULER_GUIDE_CHANGE                enum      GeoWorksNotificationType
;
;Bitmap library notification types
;
GWNT_BITMAP_CURRENT_TOOL_CHANGE         enum      GeoWorksNotificationType
GWNT_BITMAP_CURRENT_FORMAT_CHANGE       enum      GeoWorksNotificationType
;
;Flat File library notification types
;
GWNT_FLAT_FILE_FIELD_PROPERTIES_STATUS_CHANGEenum  GeoWorksNotificationType
GWNT_FLAT_FILE_FIELD_LIST_CHANGE        enum      GeoWorksNotificationType
GWNT_FLAT_FILE_RCP_STATUS_CHANGE        enum      GeoWorksNotificationType
GWNT_FLAT_FILE_FIELD_APPEARANCE_CHANGE  enum      GeoWorksNotificationType
GWNT_FLAT_FILE_DUMMY_CHANGE_2          enum      GeoWorksNotificationType
GWNT_FLAT_FILE_DUMMY_CHANGE_3          enum      GeoWorksNotificationType
;
;Spool library notification types
;
GWNT_SPOOL_DOC_OR_PAPER_SIZE            enum      GeoWorksNotificationType
;
;View control notification types
;
GWNT_VIEW_STATE_CHANGE                  enum      GeoWorksNotificationType
;
;Ink control notification types
;
GWNT_INK_HAS_TARGET                     enum      GeoWorksNotificationType
;
;Page control notification types
;
GWNT_PAGE_STATE_CHANGE                  enum      GeoWorksNotificationType
;
;Document control notification types
;
GWNT_DOCUMENT_CHANGE                    enum      GeoWorksNotificationType
;
;Display control notification types
;
GWNT_DISPLAY_CHANGE                     enum      GeoWorksNotificationTyp
GWNT_DISPLAY_LIST_CHANGE                enum      GeoWorksNotificationType
;
;Spline library notification types

```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

;
GWNT_SPLINE_MARKER_SHAPE          enum    GeoWorksNotificationType
GWNT_SPLINE_POINT                  enum    GeoWorksNotificationType
GWNT_SPLINE_POLYLINE               enum    GeoWorksNotificationType
GWNT_SPLINE_SMOOTHNESS             enum    GeoWorksNotificationType
GWNT_SPLINE_OPEN_CLOSE_CHANGE     enum    GeoWorksNotificationType
;
;
GWNT_UNUSED_1                      enum    GeoWorksNotificationType
;
; Spreadsheet control notification types
;
GWNT_SPREADSHEET_ACTIVE_CELL_CHANGE      enum    GeoWorksNotificationType
GWNT_SPREADSHEET_EDIT_BAR_CHANGE         enum    GeoWorksNotificationType
GWNT_SPREADSHEET_SELECTION_CHANGE        enum    GeoWorksNotificationType
GWNT_SPREADSHEET_CELL_WIDTH_HEIGHT_CHANGE  enum    GeoWorksNotificationType
GWNT_SPREADSHEET_DOC_ATTR_CHANGE         enum    GeoWorksNotificationType
GWNT_SPREADSHEET_CELL_ATTR_CHANGE        enum    GeoWorksNotificationType
GWNT_SPREADSHEET_CELL_NOTES_CHANGE       enum    GeoWorksNotificationType
GWNT_SPREADSHEET_DATA_RANGE_CHANGE       enum    GeoWorksNotificationType
;
; Float library notification types
;
GWNT_FLOAT_FORMAT_CHANGE              enum    GeoWorksNotificationType
;
; Impex mapping control notification types
;
GWNT_MAP_APP_CHANGE                  enum    GeoWorksNotificationType
GWNT_MAP_LIBRARY_CHANGE              enum    GeoWorksNotificationType
;
; Transfer notification types
;
GWNT_TEXT_NAME_CHANGE                enum    GeoWorksNotificationType
;
; Card library notification types
;
GWNT_CARD_BACK_CHANGE                enum    GeoWorksNotificationType
;
;
GWNT_TEXT_OBJECT_HAS_FOCUS            enum    GeoWorksNotificationType
GWNT_TEXT_CONTEXT                     enum    GeoWorksNotificationType
GWNT_TEXT_REPLACE_WITH_HWR            enum    GeoWorksNotificationType
;
; Help notification types
;
GWNT_HELP_CONTEXT_CHANGE              enum    GeoWorksNotificationType
;
GWNT_FLOAT_FORMAT_INIT                enum    GeoWorksNotificationType
;
; Hard Icon Bar notification types

```

;		
GWNT_HARD_ICON_BAR_FUNCTION	enum	GeoWorksNotificationType
GWNT_STARTUP_INDEXED_APP	enum	GeoWorksNotificationType
;		
;		
GWNT_SPOOL_PRINTING_COMPLETE	enum	GeoWorksNotificationType
GWNT_MODAL_WIN_CHANGE	enum	GeoWorksNotificationType
GWNT_SPREADSHEET_NAME_CHANGE	enum	GeoWorksNotificationType
GWNT_DOCUMENT_OPEN_COMPLETE	enum	GeoWorksNotificationType
GWNT_EMAIL_SCAN_INBOX	enum	GeoWorksNotificationType
GWNT_FOCUS_WINDOW_KBD_STATUS	enum	GeoWorksNotificationType
GWNT_TAB_DOUBLE_CLICK	enum	GeoWorksNotificationType
GWNT_PAGE_INFO_STATE_CHANGE	enum	GeoWorksNotificationType
GWNT_CURSOR_POSITION_CHANGE	enum	GeoWorksNotificationType
GWNT_FAX_NEW_JOB_CREATED	enum	GeoWorksNotificationType
GWNT_FAX_NEW_JOB_COMPLETED	enum	GeoWorksNotificationType
GWNT_EMAIL_DATABASE_CHANGE	enum	GeoWorksNotificationType
GWNT_EMAIL_STATUS_CHANGE	enum	GeoWorksNotificationType
GWNT_EMAIL_PAGE_PANEL_UPDATE	enum	GeoWorksNotificationType
GWNT_PCCOM_DISPLAY_CHAR	enum	GeoWorksNotificationType
GWNT_PCCOM_DISPLAY_STRING	enum	GeoWorksNotificationType
GWNT_PCCOM_EXIT	enum	GeoWorksNotificationType

GWNT_INK

Objects on this list receive notification of data collected as ink. This notification passes the handle of a data block holding an **InkHeader** structure (containing a series of ink points) in **bp**. If the handle is null, the system could not allocate memory to hold all the points, or was intercepted by an Input Monitor.

Note: If a monitor intercepts MSG_META_NOTIFY_WITH_DATA_BLOCK with GWNT_INK, it must still pass it on, but may pass on **bp=0** if it wants to consume the ink data itself.

Format of data:

InkHeader <>

Point<>

Point<>

...

...

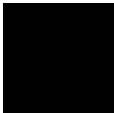
The high bit of the *x* coord is set to denote the end of a line segment.

The points are all in the screen coordinates ; objects may want to convert them into their own window coordinates using **WinUntransform()**.

Revision: ■

Draft Dated (4/18/94)

Reference book



GWNT_GEN_CONTROL_NOTIFY_STATUS_CHANGE

Objects on this list receive notification of a status change in GenControl objects. This notification passes the data type **NotifyGenControlStatusChange**.

GWNT_SELECT_STATE_CHANGE

Objects on this list are notified when a selection state has changed within a GenEditControl (cut/copy/paste). This type uses MSG_META_NOTIFY_WITH_DATA_BLOCK. This notification passes the data type **NotifySelectStateChange**.

GWNT_UNDO_STATE_CHANGE

This notification passes the data type **NotifyUndoStateChange**.

GWNT_STYLE_CHANGE

Objects on this list (a style sheet control) are notified when a style change occurs. This notification passes the data type **NotifyStyleChange**.

GWNT_STYLE_SHEET_CHANGE

This notification passes the data type **NotifyStyleSheetChange**.

GWNT_TEXT_CHAR_ATTR_CHANGE

This notification passes the data type **VisTextNotifyCharAttrChange**.

GWNT_TEXT_PARA_ATTR_CHANGE

This notification passes the data type **VisTextNotifyParaAttrChange**.

GWNT_TEXT_TYPE_CHANGE

This notification passes the data type **VisTextNotifyTypeChange**.

GWNT_TEXT_SELECTION_CHANGE

This notification passes the data type **VisTextNotifySelectionChange**.

GWNT_TEXT_COUNT_CHANGE

This notification passes the data type **VisTextNotifyCountChange**.

GWNT_TEXT_STYLE_CHANGE

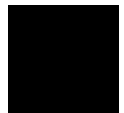
This notification passes the data type **NotifyTextStyleChange**.

- GWNT_FONT_CHANGE
This notification passes the data type **NotifyFontChange**.
- GWNT_POINT_SIZE_CHANGE
This notification passes the data type **NotifyPointSizeChange**.
- GWNT_FONT_ATTR_CHANGE
This notification passes the data type **NotifyFontAttrChange**.
- GWNT_JUSTIFICATION_CHANGE
This notification passes the data type **NotifyJustificationChange**.
- GWNT_TEXT_FG_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GWNT_TEXT_BG_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GWNT_TEXT_PARA_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GWNT_TEXT_BORDER_COLOR_CHANGE
This notification passes the data type **NotifyColorChange**.
- GWNT_SEARCH_REPLACE_ENABLE_CHANGE
This notification passes the data type **NotifySearchReplaceEnableChange**.
- GWNT_SPELL_ENABLE_CHANGE
This notification passes the data type **NotifySpellEnableChange**.
- GWNT_CHART_TYPE_CHANGE
Objects on this list receive notification when the chart type changes.
- GWNT_SPOOL_DOC_OR_PAPER_SIZE
This notification passes the data type **PageSizeReport**.
- GWNT_VIEW_STATE_CHANGE
This notification passes the data type **NotifyViewStateChange**.
- GWNT_INK_HAS_TARGET
Send with MSG_META_NOTIFY (bp = non-zero if we have the target).

Revision: ■

Draft Dated (4/18/94)

Reference book



- GWNT_PAGE_STATE_CHANGE
This notification passes the data type **NotifyPageStateChange**.
- GWNT_DOCUMENT_CHANGE
This notification passes the data type **NotifyDocumentChange**.
- GWNT_DISPLAY_CHANGE
This notification passes the data type **NotifyDisplayChange**.
- GWNT_DISPLAY_LIST_CHANGE
This notification passes the data type **NotifyDisplayListChange**.
- GWNT_SPREADSHEET_ACTIVE_CELL_CHANGE
This notification passes the data type **NotifySSheetActiveCellChange**.
- GWNT_SPREADSHEET_EDIT_BAR_CHANGE
This notification passes the data type **NotifySSheetEditBarChange**.
- GWNT_SPREADSHEET_SELECTION_CHANGE
This notification passes the data type **NotifySSheetSelectionChange**.
- GWNT_SPREADSHEET_CELL_WIDTH_HEIGHT_CHANGE
This notification passes the data type **NotifySSheetCellWidthHeightChange**.
- GWNT_SPREADSHEET_DOC_ATTR_CHANGE
This notification passes the data type **NotifySSheetDocAttrChange**.
- GWNT_SPREADSHEET_CELL_ATTR_CHANGE
This notification passes the data type **NotifySSheetCellAttrChange**.
- GWNT_SPREADSHEET_DATA_RANGE_CHANGE
This notification passes the data type **NotifySSheetDataRangeChange**.
- GWNT_TEXT_NAME_CHANGE
This notification passes the data type **VisTextNotifyNameChange**.
- GWNT_TEXT_REPLACE_WITH_HWR
This notification passes the data type **InkHeader** (and ink data), followed by a **ReplaceWithHWRData** structure.

GWNT_HELP_CONTEXT_CHANGE

This notification passes the data type **NotifyHelpChange**.

GWNT_HARD_ICON_BAR_FUNCTION

GenApplication objects on this list will perform the indicated function when receiving this notification.

GWNT_STARTUP_INDEXED_APP

Objects on this list start up the passed application when receiving this notification.

GWNT_SPOOL_PRINTING_COMPLETE

Objects on this list receive notification that printing has been completed. The spooler does not send this out, but instead delays MSG_META_SEND_CLASSSED_EVENTS that are sent to it having this as the encapsulated message, until printing is completed. This list is used in remote (IACP) printing.

GWNT_MODAL_WIN_CHANGE

Objects on this list receive notification that the modal status of the application has changed in some way, by becoming modal, non-modal, or simply changing which window is modal.

GWNT_SPREADSHEET_NAME_CHANGE

Notification that a name has been added, deleted or changed.

GWNT_DOCUMENT_OPEN_COMPLETE

The GenDocument does not send this out, but rather delays MSG_META_SEND_CLASSSED_EVENT messages that are sent to it having this as the encapsulated message, until the document has either been opened (and is ready to be printed), or has had the open operation aborted somehow.

NOTE: used for remote (IACP) printing—as of yet, there is no need for the document to delay this message, and so code to do that is not present.

GWNT_EMAIL_SCAN_INBOX

Notify an email geode that it should check for new mail.

GWNT_FOCUS_WINDOW_KBD_STATUS

On pen systems, this GCN Notification is sent from focus windows to the GAGCNLT_FOCUS_WINDOW_KBD_STATUS GCN List with the **NotifyFocusWindowKbdStatus** structure to tell the system what the floating keyboard should do.

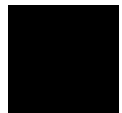
GWNT_TAB_DOUBLE_CLICK

Sent when a tab is double clicked on. This is an inelegant solution to allow backwards compatibility

Revision: ■

Draft Dated (4/18/94)

Reference book



GWNT_PAGE_INFO_STATE_CHANGE
Sent when page info (size, margin) changes.
Data type: **NotifyPageInfoChange**

GWNT_CURSOR_POSITION_CHANGE
Data type: **VisTextCursorPositionChange**

GWNT_FAX_NEW_JOB_CREATED
Notify the fax spooler that there is a new fax file created.

GWNT_FAX_NEW_JOB_COMPLETED
Notify the fax spooler that the new fax file is completely generated.

GWNT_EMAIL_DATABASE_CHANGE
Notify email app that database has changed.

GWNT_EMAIL_STATUS_CHANGE
Notify email app of current communications state.

GWNT_EMAIL_PAGE_PANEL_UPDATE
Ask the page panel to enable or disable features.

GWNT_PCCOM_DISPLAY_CHAR
Sent when there is a character ready to be displayed
Data type: char

GWNT_PCCOM_DISPLAY_STRING
Sent when there is a string ready to be displayed
Data type: **MemHandle** of block containing the null-terminated string.

GWNT_PCCOM_EXIT
Sent when the remote machine has sent the exit command and pccom has exited.
Data type: **PCComReturnType** from the call to PCCOMEXIT.

Library: **geoworks.def**

■ GeoWorksPrefDialogGCNListType

```
GeoWorksPrefDialogGCNListType etype word, first PrefDialogMessages, 2
PDGCNLT_DIALOG_CHANGE          enum GeoWorksPrefDialogGCNListType
; MSG_PREF_NOTIFY_DIALOG_CHANGE sent out.
```

Library: **config.def**

■ GeoWorksVisContentGCNListType

```
GeoWorksVisContentGCNListType etype word ,
FIRST_VIS_CONTENT_GCN_LIST_TYPE, 2
    VCGCNLT_TARGET_NOTIFY_TEXT_PARA_ATTR_CHANGEenum
                                GeoWorksVisContentGCNListType
;Data type: VisTextNotifyParaAttrChange
```

Library: **geoworks.def**

■ GestureType

```
GestureType etype word
    GT_NO_GESTURE enum GestureType
    GT_DELETE_CHARS enum GestureType
    GT_SELECT_CHARS enum GestureType
    GT_V_CROSSOUT enum GestureType
    GT_H_CROSSOUT enum GestureType
    GT_BACKSPACE enum GestureType
```

Library: **hwr.def**

■ GetContextParams

```
GetContextParams struct
    GCP_replyObj optr ;Output to reply to via MSG_META_CONTEXT
    GCP_numCharsToGet word ;Maximum number of characters to return
    GCP_location ContextLocation
    GCP_position dword
GetContextParams ends
```

Library: **Objects/vTextC.def**

■ GetItemMonikerParams

```
GetItemMonikerParams struct
    GIMP_identifier word
    GIMP_bufferSize word
    GIMP_buffer fptr.char
GetItemMonikerParams ends
```

Library: **config.def**

Revision: ■
Draft Dated (4/18/94)

Reference book



■ GetMaskType

```
GetMaskType          etype byte
GMT_ENUM             enum GetMaskType
GMT_BUFFER           enum GetMaskType
```

Library: **graphics.def**

■ GetPalType

```
GetPalType          etype byte
GPT_ACTIVE          enum GetPalType
GPT_DEFAULT         enum GetPalType
```

Library: **color.def**

■ GetPathType

```
GetPathType        etype    word
GPT_CURRENT        enum GetPathType    ; current path
GPT_CLIP           enum GetPathType    ; clip path
GPT_WIN_CLIP       enum GetPathType    ; win clip path
```

Use this type with **GrGetPath** to determine which sort of path to get.

Library:

■ GetSearchSpellObjectParam

```
GetSearchSpellObjectParam  record
    GSSOP_RELAYED_FLAG      :1
                           :11
    GSSOP_TYPE              GetSearchSpellObjectType:4
GetSearchSpellObjectParam  end
```

Library: **Objects/vTextC.def**

■ GetSearchSpellObjectType

```
GetSearchSpellObjectType  etype word
GSSOT_FIRST_OBJECT       enum GetSearchSpellObjectType
GSSOT_LAST_OBJECT        enum GetSearchSpellObjectType
GSSOT_NEXT_OBJECT        enum GetSearchSpellObjectType
GSSOT_PREV_OBJECT        enum GetSearchSpellObjectType
```

GSSOT_FIRST_OBJECT

This type indicates to the spell checker to start spell checking from the first object encountered in the document when the

user clicks on “Check Entire Document.” It is also used by the search code to wrap a search to the beginning after it has reached the end.

GSSOT_LAST_OBJECT

This type indicates to the spell checker to wrap a backwards search around to the end.

GSSOT_NEXT_OBJECT

This type indicates to the spell checker to go to the next object in which to continue spell checking. At the end of the chain of objects, it should return 0:0.

GSSOT_PREV_OBJECT

This type indicates to the spell checker to go to the previous object when continuing spell checking. After reaching the start of the chain, it should return 0:0.

Library: **Objects/vTextC.def**

■ GetVarDataParams

```
GetVarDataParams    struct
    GVDP_buffer      fptr
    GVDP_bufferSize  word
    GVDP_dataType    word
GetVarDataParams    ends
```

GVDP_buffer stores the pointer to the buffer to fill with data from the VarData entry. This must be passed unless *GVDP_bufferSize* is 0.

GVDP_bufferSize stores the size of the above buffer (to allow us to prevent overflow). This must be set to zero if no buffer is passed.

GVDP_dataType stores the VarData type whose data should be returned.

Library: **Objects/metaC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GFM_info

```
GFM_info          etype word, 0, 2
  GFMI_HEIGHT      enum GFM_info    ;height of font box
  GFMI_MEAN         enum GFM_info    ;top of lowers
  GFMI_DESCENT      enum GFM_info    ;descent of lowers
  GFMI_BASELINE     enum GFM_info    ;baseline offset
  GFMI_LEADING      enum GFM_info    ;external leading
  GFMI_AVERAGE_WIDTH enum GFM_info    ;average char width
  GFMI_ASCENT       enum GFM_info    ;ascent line to baseline
  GFMI_MAX_WIDTH    enum GFM_info    ;widest char width
  GFMI_MAX_ADJUSTED_HEIGHT enum GFM_info ;height, adjusted, with
                                     ;above/below
  GFMI_UNDER_POS    enum GFM_info    ;offset to underline
  GFMI_UNDER_THICKNESS enum GFM_info ;thickness of underline
  GFMI_ABOVE_BOX    enum GFM_info    ;height of above box
  GFMI_ACCENT       enum GFM_info    ;height of accent
  GFMI_DRIVER       enum GFM_info    ;driver ID
  GFMI_KERN_COUNT   enum GFM_info    ;# of kerning pairs
  GFMI_FIRST_CHAR   enum GFM_info    ;first character in font
  GFMI_LAST_CHAR    enum GFM_info    ;last character in font
  GFMI_DEFAULT_CHAR enum GFM_info    ;default character for font
  GFMI_STRIKE_POS   enum GFM_info    ;strike-through position
  GFMI_BELOW_BOX    enum GFM_info    ;height of below box
```

Library: **font.def**

■ GFSTempDataEntry

```
GFSTempDataEntry  struct
  GFSTDE_selectionNumber  word
  GFSTDE_selectionFlags   GenFileSelectorEntryFlags
GFSTempDataEntry  ends
```

Library: **Objects/gFSelC.def**

■ GOAACFeatures

```
GOAACFeatures      record
    GOAACF_MM_CLEAR      :1
    GOAACF_MM_COPY       :1
    GOAACF_MM_NOP        :1
    GOAACF_MM_AND        :1
    GOAACF_MM_INVERT     :1
    GOAACF_MM_XOR        :1
    GOAACF_MM_SET        :1
    GOAACF_MM_OR         :1
    GOAACF_TRANSPARENCY  :1
GOAACFeatures      end
```

Library: **grobj.def**

■ GOArcCFeatures

```
GOArcCFeatures      record
    GOACF_START_ANGLE    :1
    GOACF_END_ANGLE      :1
    GOACF_PIE_TYPE       :1
    GOACF_CHORD_TYPE     :1
GOArcCFeatures      end
```

Library: **grobj.def**

■ GOATGCFeatures

```
GOATGCFeatures      record
    GOATGCF_ALIGN_TO_GRID :1
GOATGCFeatures      end
```

Library: **grobj.def**

■ GOCCFeatures

```
GOCCFeatures      record
    GOCCF_CONVERT_TO_BITMAP :1
    GOCCF_CONVERT_TO_GRAPHIC :1
    GOCCF_CONVERT_FROM_GRAPHIC :1
GOCCFeatures      end
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GOCDCFeatures

```
GOCDCFeatures      record
    GOCDCF_REPITIONS      :1
    GOCDCF_MOVE           :1
    GOCDCF_SCALE          :1
    GOCDCF_ROTATE         :1
    GOCDCF_SKEW           :1
GOCDCFeatures      end
```

Library: **grobj.def**

■ GOCSCFeatures

```
GOCSCFeatures      record
    GOCSCF_NUM_POLYGON_SIDES :1
    GOCSCF_POLYGON_RADIUS   :1
    GOCSCF_NUM_STAR_POINTS  :1
    GOCSCF_STAR_RADII       :1
GOCSCFeatures      end
```

Library: **grobj.def**

■ GODACFeatures

```
GODACFeatures      record
    GODACF_SET_DEFAULT_ATTRIBUTES :1
GODACFeatures      end
```

Library: **grobj.def**

■ GODEpthCFeatures

```
GODEpthCFeatures   record
    GODEpthCF_BRING_TO_FRONT :1
    GoDepthCF_SEND_TO_BACK   :1
    GoDepthCF_SHUFFLE_UP     :1
    GoDepthCF_SHUFFLE_DOWN   :1
GODEpthCFeatures   end
```

Library: **grobj.def**

■ GODMCFeatures

```
GODMCFeatures      record
  GODMCF_DRAFT_MODE  :1
GODMCFeatures      end
```

Library: **grobj.def**

■ GOFCFFeatures

```
GOFCFFeatures      record
  GOFDCF_FLIP_HORIZONTAL  :1
  GOFDCF_FLIP_VERTICAL    :1
GOFCFFeatures      end
```

Library: **grobj.def**

■ GOGCFFeatures

```
GOGCFFeatures      record
  GOGCF_GROUP           :1
  GOGCF_UNGROUP         :1
GOGCFFeatures      end
```

Library: **grobj.def**

■ GOHCFFeatures

```
GOHCFFeatures      record
  GOHCF_SMALL_HANDLES    :1
  GOHCF_MEDIUM_HANDLES   :1
  GOHCF_LARGE_HANDLES    :1
  GOHCF_INVISIBLE_HANDLES :1
GOHCFFeatures      end
```

Library: **grobj.def**

■ GOHSCFeatures

```
GOHSCFeatures      record
  GOHSCF_HIDE   :1
  GOHSCF_SHOW   :1
GOHSCFeatures      end
```

Revision: ■

Draft Dated (4/18/94)

Reference book



Library: **grobj.def**

■ GOLACFeatures

```
GOLACFeatures      record
  GOLACF_WIDTH_INDEX      :1
  GOLACF_WIDTH_VALUE      :1
  GOLACF_STYLE             :1
  GOLACF_ARROWHEAD_TYPE   :1
  GOLACF_ARROWHEAD_WHICH_END :1
GOLACFeatures      end
```

Library: **grobj.def**

■ GOLACToolboxFeatures

```
GOLACToolboxFeatures      record
  GOLACTF_WIDTH_INDEX      :1
  GOLACTF_STYLE             :1
GOLACToolboxFeatures      end
```

Library: **grobj.def**

■ GOPICFeatures

```
GOPICFeatures      record
  GOPICF_PASTE_INSIDE      : 1
  GOPICF_BREAKOUT_PASTE_INSIDE : 1
GOPICFeatures      end
```

Library: **grobj.def**

■ GOPICToolboxFeatures

```
GOPICToolboxFeatures      record
  GOPICTF_PASTE_INSIDE      : 1
  GOPICTF_BREAKOUT_PASTE_INSIDE : 1
GOPICToolboxFeatures      end
```

Library: **grobj.def**

■ GOTCFeatures

```
GOTCFeatures      record
    GOTCF_PTR      :1
    GOTCF_ROTATE_PTR :1
    GOTCF_ZOOM      :1
    GOTCF_TEXT      :1
    GOTCF_LINE      :1
    GOTCF_RECT      :1
    GOTCF_ROUNDED_RECT :1
    GOTCF_ELLIPSE   :1
    GOTCF_ARC       :1
    GOTCF_POLYLINE   :1
    GOTCF_POLYCURVE  :1
    GOTCF_SPLINE     :1
GOTCFeatures      end
```

Library: **grobj.def**

■ GOTransformCFeatures

```
GOTransformCFeatures record
    GOTCF_UNTRANSFORM :1
GOTransformCFeatures end
```

Library: **grobj.def**

■ GPCFeatures

```
GPCFeatures      record
    GPCF_GOTO_PAGE   :1
    GPCF_NEXT_PAGE   :1
    GPCF_PREVIOUS_PAGE :1
GPCFeatures      end
```

Library: **gPageCC.def**

■ GPCToolboxFeatures

```
GPCToolboxFeatures record
    GPCTF_PREVIOUS_PAGE :1
    GPCTF_GOTO_PAGE     :1
    GPCTF_NEXT_PAGE     :1
GPCToolboxFeatures end
```

Revision: ■

Draft Dated (4/18/94)

Reference book



Library: **gPageCC.def**

■ GPICFeatures

```
GPICFeatures      record
  GPICF_KEYBOARD      :1
  GPICF_CHAR_TABLE    :1
  GPICF_CHAR_TABLE_SYMBOLS :1
  GPICF_CHAR_TABLE_INTERNATIONAL :1
  GPICF_CHAR_TABLE_MATH :1
  GPICF_CHAR_TABLE_CUSTOM :1
  GPICF_HWR_ENTRY_AREA :1
GPICFeatures      end
```

Library: **gPenICC.def**

■ GPICToolboxFeatures

```
GPICToolboxFeatures record
  GPICTF_INITIATE :1
GPICToolboxFeatures end
```

Library: **grobj.def**

■ GraphicPattern

```
GraphicPattern      struct
  GP_type      PatternType
  GP_data      byte
GraphicPattern      ends
```

This structure stores a system hatch pattern.

Library: **graphics.def**

■ Grid

```
Grid      struct
  G_x      WWFixed      ;pixels between horiz gridlines
  G_y      WWFixed      ;pixels between vert gridlines
Grid      ends
```

Library: **ruler.def**

■ **GridOptions**

```
GridOptions      record
  GO_SHOW_GRID   :1
  GO_SNAP_TO_GRID :1
                  :6
GridOptions      end
```

Library: **ruler.def**

■ **GrInfoType**

```
GrInfoType      etype word, 0, 2
  GIT_PRIVATE_DATA      enum GrInfoType
  GIT_WINDOW            enum GrInfoType
```

Library: **graphics.def**

■ **GrObjActionModes**

```
GrObjActionModes      record
  GOAM_RESIZE          :1
  GOAM_MOVE            :1
  GOAM_ROTATE          :1
  GOAM_CHOOSE          :1
  GOAM_ACTION_ACTIVATED :1
  GOAM_ACTION_PENDING  :1
  GOAM_ACTION_HAPPENING :1
  GOAM_CREATE          :1
GrObjActionModes      end
```

Library: **grobj.def**

■ **GrObjActionNotificationStruct**

```
GrObjActionNotificationStruct      struct
  GOANS_suspendCount      word      ; If non-zero, then defer sending out
                                   ; action notification.
  GOANS_optr              optr      ; OD to send message to.
GrObjActionNotificationStruct      ends
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjActionNotificationType

```
GrObjActionNotificationType  etype word
    GOANT_NULL                enum GrObjActionNotificationType
                                ; Reserve zero as a special value
    GOANT_SELECTED            enum GrObjActionNotificationType
    GOANT_UNSELECTED          enum GrObjActionNotificationType
    GOANT_CREATED              enum GrObjActionNotificationType
    GOANT_MOVED                enum GrObjActionNotificationType
    GOANT_RESIZED              enum GrObjActionNotificationType
    GOANT_ROTATED              enum GrObjActionNotificationType
    GOANT_SKEWED               enum GrObjActionNotificationType
    GOANT_TRANSFORMED          enum GrObjActionNotificationType
    GOANT_ATTRED               enum GrObjActionNotificationType
    GOANT_SPEC_MODIFIED        enum GrObjActionNotificationType
    GOANT_PASTED               enum GrObjActionNotificationType
    GOANT_DELETED              enum GrObjActionNotificationType
    GOANT_WRAP_CHANGED         enum GrObjActionNotificationType
    GOANT_UNDO_GEOMETRY        enum GrObjActionNotificationType
    GOANT_UNDO_DELETE          enum GrObjActionNotificationType
    GOANT_REDO_DELETE          enum GrObjActionNotificationType
    GOANT_PRE_MOVE             enum GrObjActionNotificationType
    GOANT_PRE_RESIZE           enum GrObjActionNotificationType
    GOANT_PRE_ROTATE           enum GrObjActionNotificationType
    GOANT_PRE_SKEW             enum GrObjActionNotificationType
    GOANT_PRE_TRANSFORM        enum GrObjActionNotificationType
    GOANT_PRE_SPEC_MODIFY      enum GrObjActionNotificationType
    GOANT_QUERY_DELETE         enum GrObjActionNotificationType
    GOANT_PRE_WRAP_CHANGE      enum GrObjActionNotificationType
```

Library: **grobj.def**

■ GrObjAlignDistributeControlFeatures

```
GrObjAlignDistributeControlFeatures record
    GOADCF_ALIGN_LEFT          :1
    GOADCF_ALIGN_CENTER_HORIZONTALLY :1
    GOADCF_ALIGN_RIGHT         :1
    GOADCF_ALIGN_WIDTH         :1

    GOADCF_ALIGN_TOP           :1
    GOADCF_ALIGN_CENTER_VERTICALLY :1
    GOADCF_ALIGN_BOTTOM        :1
    GOADCF_ALIGN_HEIGHT        :1

    GOADCF_DISTRIBUTE_LEFT     :1
    GOADCF_DISTRIBUTE_CENTER_HORIZONTALLY :1
    GOADCF_DISTRIBUTE_RIGHT    :1
    GOADCF_DISTRIBUTE_WIDTH    :1
```

Revision: ■

Draft Dated (4/18/94)

```
GOADCF_DISTRIBUTE_TOP :1
GOADCF_DISTRIBUTE_CENTER_VERTICALLY :1
GOADCF_DISTRIBUTE_BOTTOM :1
GOADCF_DISTRIBUTE_HEIGHT :1
GrObjAlignDistributionControlFeatures end
```

Library: **grobj.def**

■ GrObjAnchoredScaleData

```
GrObjAnchoredScaleData struct
GOASD_scale GrObjScaleData
GOASD_scaleAnchor GrObjHandleSpecification
align word
GrObjAnchoredScaleData ends
```

Library: **grobj.def**

■ GrObjAnchoredSkewData

```
GrObjAnchoredSkewData struct
GOASD_degrees GrObjSkewData
GOASD_skewAnchor GrObjHandleSpecification
align word
GrObjAnchoredSkewData ends
```

Library: **grobj.def**

■ GrObjAreaAttrElementType

```
GrObjAreaAttrElementType etype byte
GOAAET_BASE enum GrObjAreaAttrElementType
;GrObjBaseAreaAttrElement
GOAAET_GRADIENT enum GrObjAreaAttrElementType
;GrObjGradientAreaAttrElement
```

Library: **grobj.def**



■ GrObjAreaAttrInfoRecord

```
GrObjAreaAttrInfoRecord  record
    GOAAIR_TRANSPARENT    :1
GrObjAreaAttrInfoRecord  end
```

GOAAIR_TRANSPARENT

This flag indicates a GrObj area is transparent. If this flag is set, then there is no need to redraw the background behind the object.

Library: **grobj.def**

■ GrObjAttrFlags

```
GrObjAttrFlags            record
                                :6
    GOAF_DONT_COPY_LOCKS    :1
    GOAF_HAS_PASTE_INSIDE_CHILDREN :1
    GOAF_PASTE_INSIDE       :1
    GOAF_INSERT_DELETE_MOVE_ALLOWED :1
    GOAF_INSERT_DELETE_RESIZE_ALLOWED:1
    GOAF_INSERT_DELETE_DELETE_ALLOWED:1
    GOAF_INSTRUCTION        :1
    GOAF_MULTIPLICATIVE_RESIZE :1
    GOAF_WRAP                GrObjWrapTextType:2
GrObjAttrFlags            end
```

GOAF_DONT_COPY_LOCKS

This flag indicates that when the object is written out to the transfer format, any locks will not be copied.

GOAF_HAS_PASTE_INSIDE_CHILDREN

This flag indicates that the object contains paste inside children. (This is only relevant for objects in groups.)

GOAF_PASTE_INSIDE

Meaningless unless object is in a group. This flag indicates that the object was pasted inside a group and will be drawn clipped to the group's normal children. (This is only relevant for objects in groups.)

GOAF_INSERT_DELETE_MOVE_ALLOWED

This flag indicates that this GrObj can be moved as a result of a MSG_GO_INSERT_OR_DELETE_SPACE.

GOAF_INSERT_DELETE_RESIZE_ALLOWED

This flag indicates that this GrObj can be resized as a result of MSG_GO_INSERT_OR_DELETE_SPACE.

GOAF_INSERT_DELETE_DELETE_ALLOWED

This flag indicates that this GrObj can be deleted as a result of a MSG_GO_INSERT_OR_DELETE_SPACE.

GOAF_INSTRUCTION

This flag indicates that this object is used for instructions in a template.

GOAF_MULTIPLICATIVE_RESIZE

This flag indicates whether resize deltas are added to object coordinates. If true, a scale factor is calculated and applied to objects transform. If false, then resize deltas are not added to object coordinates.

GOAF_WRAP

This type indicates how to wrap text with respect to the object.

Library: **grobj.def**

■ GrObjAttributeManagerArrayDesc

```
GrObjAttributeManagerArrayDesc    struct
    GOAMAD_areaAttrArrayHandle    word
    GOAMAD_areaDefaultElement     word
    GOAMAD_lineAttrArrayHandle    word
    GOAMAD_lineDefaultElement     word
    GOAMAD_grObjStyleArrayHandle  word
    GOAMAD_charAttrArrayHandle    word
    GOAMAD_charDefaultElement     word
    GOAMAD_paraAttrArrayHandle    word
    GOAMAD_paraDefaultElement     word
    GOAMAD_typeArrayHandle        word
    GOAMAD_typeDefaultElementword word
    GOAMAD_graphicArrayHandle     word
    GOAMAD_nameArrayHandle        word
    GOAMAD_textStyleArrayHandle   word
GrObjAttributeManagerArrayDesc    ends
```

GOAMAD_areaAttrArrayHandle stores the VM block handle of the element array for area attributes. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, an area attribute array will be created.

GOAMAD_areaDefaultElement stores the element number of the default area attributes in the area attribute array. This value is ignored if 0 is passed in *GOAMAD_areaAttrArrayHandle*.

GOAMAD_lineAttrArrayHandle stores the VM block handle of the element array for line attributes. The associated chunk must be at the

Revision: ■

Draft Dated (4/18/94)

Reference book



GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a line attribute array will be created.

GOAMAD_lineDefaultElement stores the element number of the default line attributes in the line attribute array. This value is ignored if 0 is passed in *GOAMAD_inlineAttrArrayHandle*.

GOAMAD_grObjStyleArrayHandle stores the VM block handle of the element array for grobj styles. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a style array will be created.

GOAMAD_charAttrArrayHandle stores the VM block handle of the element array for character attributes. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a character attribute array will be created.

GOAMAD_charDefaultElement stores the element number of the default attributes in the character array. This value is ignored if 0 is passed in *GOAMAD_charAttrArrayHandle*.

GOAMAD_paraAttrArrayHandle stores the VM block handle of the element array for paragraph attributes. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a paragraph attribute array will be created.

GOAMAD_paraDefaultElement stores the element number of the default attributes in the paragraph array. This value is ignored if 0 is passed in *GOAMAD_paraAttrArrayHandle*.

GOAMAD_typeArrayHandle stores the VM block handle of the element array for types. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a type array will be created.

GOAMAD_typeDefaultElement stores the element number of the default attributes in a type array. This value is ignored if 0 is passed in *GOAMAD_paraAttrArrayHandle*.

GOAMAD_graphicArrayHandle stores the VM block handle of the element array for graphics. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a graphic array will be created.

GOAMAD_nameArrayHandle stores the VM block handle of the element array for names. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a name array will be created.

GOAMAD_textStyleArrayHandle stores the VM block handle of the element array for text styles. The associated chunk must be at the GROBJ_VM_ELEMENT_ARRAY_CHUNK offset. If 0, a style array will be created.

Library: **grobj.def**

■ **GrObjBaseAreaAttrDiffs**

```
GrObjBaseAreaAttrDiffs    record
  GOBAAD_MULTIPLE_ELEMENT_TYPES      :1
  GOBAAD_MULTIPLE_STYLE_ELEMENTS     :1
  GOBAAD_MULTIPLE_COLORS              :1
  GOBAAD_MULTIPLE_BACKGROUND_COLORS  :1
  GOBAAD_MULTIPLE_MASKS              :1
  GOBAAD_MULTIPLE_PATTERNS           :1
  GOBAAD_MULTIPLE_DRAW_MODES         :1
  GOBAAD_MULTIPLE_INFOS              :1

  GOBAAD_MULTIPLE_GRADIENT_END_COLORS :1
  GOBAAD_MULTIPLE_GRADIENT_TYPES     :1
  GOBAAD_MULTIPLE_GRADIENT_INTERVALS :1
  GOBAAD_MULTIPLE_GRADIENT_COLORS    :4
  GOBAAD_FIRST_RECIPIENT              :1
GrObjBaseAreaAttrDiffs    end
```

GOBAAD_FIRST_RECIPIENT
This flag indicates that this GrObj is the first one to receive this data buffer (and should thereafter clear it).

Library: **grobj.def**

■ GrObjBaseAreaAttrElement

```
GrObjBaseAreaAttrElement struct
    GOBAAE_styleElement    StyleSheetElementHeader
    GOBAAE_r               byte
    GOBAAE_g               byte
    GOBAAE_b               byte
    GOBAAE_mask            SystemDrawMask
    GOBAAE_drawMode        MixMode
    GOBAAE_pattern         GraphicPattern
    GOBAAE_backR           byte
    GOBAAE_backG           byte
    GOBAAE_backB           byte
    GOBAAE_aaeType         GrObjAreaAttrElementType
    GOBAAE_areaInfo        GrObjAreaAttrInfoRecord
    GOBAAE_reservedByte    byte           ; Currently unused. Must be 0.
    GOBAAE_reserved        word          ; Currently unused. Must be 0.
GrObjBaseAreaAttrElement ends
```

Library: **grobj.def**

■ GrObjBaseLineAttrDiffs

```
GrObjBaseLineAttrDiffs    record
    GOBLAD_MULTIPLE_STYLE_ELEMENTS          :1
    GOBLAD_MULTIPLE_ELEMENT_TYPES           :1
    GOBLAD_MULTIPLE_COLORS                  :1
    GOBLAD_MULTIPLE_ENDS                    :1
    GOBLAD_MULTIPLE_JOINS                   :1
    GOBLAD_MULTIPLE_WIDTHS                  :1
    GOBLAD_MULTIPLE_MASKS                   :1
    GOBLAD_MULTIPLE_STYLES                  :1
    GOBLAD_ARROWHEAD_ON_START               :1
    GOBLAD_ARROWHEAD_ON_END                :1
    GOBLAD_ARROWHEAD_FILLED                 :1
    GOBLAD_ARROWHEAD_FILL_WITH_AREA_ATTRIBUTES :1
    GOBLAD_MULTIPLE_MITER_LIMITS            :1
    GOBLAD_MULTIPLE_ARROWHEAD_ANGLES        :1
    GOBLAD_MULTIPLE_ARROWHEAD_LENGTHS      :1
    GOBLAD_FIRST_RECIPIENT                  :1
GrObjBaseLineAttrDiffs    end
```

Library: **grobj.def**

■ GrObjBaseLineAttrElement

```
GrObjBaseLineAttrElement struct
    GOBLAE_styleElement    StyleSheetElementHeader
    GOBLAE_r                byte
    GOBLAE_g                byte
    GOBLAE_b                byte
    GOBLAE_end              LineEnd
    GOBLAE_join             LineJoin
    GOBLAE_width            WWFixed
    GOBLAE_mask             SystemDrawMask
    GOBLAE_style            LineStyle
    GOBLAE_miterLimit       WWFixed
    GOBLAE_laeType          GrObjLineAttrElementType
    GOBLAE_lineInfo         GrObjLineAttrInfoRecord
    GOBLAE_arrowheadAngle   byte
    GOBLAE_arrowheadLength  byte
    GOBLAE_reserved         word        ; Currently unused. Must be 0.
GrObjBaseLineAttrElement ends
```

Library: **grobj.def**

■ GrObjBlockHandleElement

```
GrObjBlockHandleElement struct
    GOBHE_blockHandle      hptr
    GOBHE_potentialSize     word
GrObjBlockHandleElement ends
```

Library: **grobj.def**

■ GrObjBodyAddGrObjFlags

```
GrObjBodyAddGrObjFlags record
    GOBAGOF_DRAW_LIST_POSITION :1
    GOBAGOF_REFERENCE           :15
GrObjBodyAddGrObjFlags end
```

GOBAGOF_DRAW_LIST_POSITION

If this flag is set, GOBAGOF_REFERENCE refers to the GrObj's position in the draw list; if the flag is clear, GOBAGOF_REFERENCE refers to its position in the reverse draw list.

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjBodyCreateGrObjParams

```
GrObjBodyCreateGrObjParams    struct
    GBCGP_class                fptr.ClassStruct
    padding1                    word                ; This padding ensures that width and
                                                    ; height align with GrObjInitializedData.

    padding2                    word
    padding3                    word
    padding4                    word
    GBCGP_width                 WWFixed
    GBCGP_height                WWFixed
GrObjBodyCreateGrObjParams    ends
```

Library: **grobj.def**

■ GrObjBodyCustomDuplicateParams

```
GrObjBodyCustomDuplicateParams    struct
    GBCDP_repetitions           word
    GBCDP_move                  PointDWFixed
    GBCDP_rotation              WWFixed
    GBCDP_rotateAnchor          GrObjHandleSpecification
    GBCDP_skew                  GrObjAnchoredSkewData
    GBCDP_scale                 GrObjAnchoredScaleData
    align                       word
GrObjBodyCustomDuplicateParams    ends
```

Library: **grobj.def**

■ GrObjBodyFlags

```
GrObjBodyFlags    record
    GBF_HAS_ACTION_NOTIFICATION    :1
    GBF_DEFAULT_TARGET             :1
    GBF_DEFAULT_FOCUS              :1
GrObjBodyFlags    end
```

GBF_HAS_ACTION_NOTIFICATION

This flag is set if the GrObj body contains an action notification
optr within its vardata.

GBF_DEFAULT_TARGET

This flag is set if the GrObjBody will grab the default target
upon a MSG_VIS_OPEN.

GBF_DEFAULT_FOCUS

This flag is set if the GrObjBody will grab the default focus upon a MSG_VIS_OPEN.

Library: **grobj.def**

■ GrObjBodyNotifyInstructionFlags

```
GrObjBodyNotifyInstructionFlags  struct
    GBNIF_flags                   GrObjDrawFlags
    GBNIF_handleSize              byte    ;this field added post-Zoomer
    align word
GrObjBodyNotifyInstructionFlags  ends
```

Library: **grobj.def**

■ GrObjBodyPasteCallBackStruct

```
GrObjBodyPasteCallBackStruct  struct
    GOBPCBS_message              word
    GOBPCBS_optr                 optr
GrObjBodyPasteCallBackStruct  ends
```

Library: **grobj.def**

■ GrObjBodyUnsuspendOps

```
GrObjBodyUnsuspendOps         record
    GBUO_UI_NOTIFY              GrObjUINotificationTypes: width GrObjUINotificationTypes
GrObjBodyUnsuspendOps         end

    GBUO_UI_NOTIFY
    This field stores notifications that should be sent when the GrObjBody
    suspend counts returns to zero.
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjCreateControlFeatures

```
GrObjCreateControlFeatures    record
    GOCCF_RECTANGLE           :1
    GOCCF_ELLIPSE             :1
    GOCCF_LINE                 :1
    GOCCF_ROUNDED_RECTANGLE    :1
    GOCCF_ARC                  :1
    GOCCF_TRIANGLE            :1
    GOCCF_HEXAGON              :1
    GOCCF_OCTOGON              :1
    GOCCF_FIVE_POINTED_STAR    :1
    GOCCF_EIGHT_POINTED_STAR   :1
GrObjCreateControlFeatures    end
```

Library: **grobj.def**

■ GrObjCreateGStateType

```
GrObjCreateGStateType        etype word
    BODY_GSTATE               enum GrObjCreateGStateType,0
                                ;Has only translation to body upper left
                                ;in it. Useful for un-transforming device
                                ;coordinates into document coordinates.
    PARENT_GSTATE              enum GrObjCreateGStateType
                                ;Has body translation and group
                                ;transformations in it.
    OBJECT_GSTATE              enum GrObjCreateGStateType
```

Library: **grobj.def**

■ GrObjDefiningData

```
GrObjDefiningData            struct
    GODD_attrFlags             GrObjAttrFlags
    GODD_locks                 GrObjLocks
    GODD_areaToken             word
    GODD_lineToken             word
    GODD_normalTransform        ObjectTransform
GrObjDefiningData            ends
```

This structure represents the subset of GrObj instance data required to recover the object from a transfer item.

Library: **grobj.def**

■ GrObjDrawFlags

```
GrObjDrawFlags      record
                                : 7
    GODF_DRAW_QUICK_VIEW      : 1
    GODF_DRAW_CLIP_ONLY       : 1
    GODF_DRAW_WRAP_TEXT_INSIDE_ONLY : 1
    GODF_DRAW_WRAP_TEXT_AROUND_ONLY : 1
    GODF_DRAW_WITH_INCREASED_RESOLUTION : 1
    GODF_DRAW_INSTRUCTIONS     : 1
    GODF_DRAW_SELECTED_OBJECTS_ONLY : 1
    GODF_DRAW_OBJECTS_ONLY     : 1
    GODF_PRINT_INSTRUCTIONS    : 1
GrObjDrawFlags      end
```

GODF_DRAW_QUICK_VIEW

If this flag is set, GrObjs will draw themselves with MSG_GO_DRAW_QUICK_VIEW. This message results in much faster drawing but is not WYSIWYG.

GODF_DRAW_CLIP_ONLY

If this flag is set, GrObjs will only draw their clip area.

GODF_DRAW_WRAP_TEXT_INSIDE_ONLY

If this flag is set, only GrObjs with GOAF_WRAP set to GOWTT_WRAP_INSIDE will draw.

GODF_DRAW_WRAP_TEXT_AROUND_ONLY

If this flag is set, only GrObjs with GOAF_WRAP set to GOWTT_WRAP_AROUND_RECT or GOWTT_WRAP_AROUND_TIGHTLY will draw.

GODF_DRAW_WITH_INCREASED_RESOLUTION

If this flag is set, the object should draw with more resolution (if possible). This flag is used for printing and when the view is scaled.

GODF_DRAW_INSTRUCTIONS

If this flag is set, GrObjs with the GOAF_INSTRUCTION bit set will draw.

GODF_DRAW_SELECTED_OBJECTS_ONLY

If this flag is set, only selected objects will draw.

GODF_DRAW_OBJECTS_ONLY

If this flag is set, then only the GrObjs themselves should be drawn. (I.e. don't draw grid lines, sprites, handles, etc.)

GODF_PRINT_INSTRUCTIONS

If this flag is set, instructions should be printed.

Revision: ■

Draft Dated (4/18/94)

Reference book



Library: **grobj.def**

■ GrObjDuplicateControlFeatures

```
GrObjDuplicateControlFeatures    record
    GODCF_DUPLICATE : 1
    GODCF_DUPLICATE_IN_PLACE : 1
GrObjDuplicateControlFeatures    end
```

Library: **grobj.def**

■ GrObjDuplicateControlToolboxFeatures

```
GrObjDuplicateControlToolboxFeatures    record
    GODCTF_DUPLICATE : 1
    GODCTF_DUPLICATE_IN_PLACE : 1
GrObjDuplicateControlToolboxFeatures    end
```

Library:

■ GrObjEntryPointRelocation

```
GrObjEntryPointRelocation    struct
    GOEPR_fullRelocation    EntryPointRelocation
    GOEPR_grObjEntryPoint    word
GrObjEntryPointRelocation    ends
```

Library: **grobj.def**

■ GrObjFileStatus

```
GrObjFileStatus    record
    GOFS_MOUSE_GRAB : 1    ;True if body has mouse grab
    GOFS_SYS_TARGETED : 1    ;Body has the system target excl
    GOFS_TARGETED : 1    ;Body or one of its children has target
    GOFS_OPEN : 1    ;True if file is open.
GrObjFileStatus    end
```

Library: **grobj.def**

■ GrObjFullAreaAttrElement

```
GrObjFullAreaAttrElement      struct
    GOFAAE_base      GrObjBaseAreaAttrElement
    GOFAAE_future     byte FUTURE_AREA_ATTR_ELEMENT_DATA_SIZE dup (?)
GrObjFullAreaAttrElement      ends
```

This structure is used to allow future routines to access larger **GrObjBaseAreaAttrElement** structures.

Library: **grobj.def**

■ GrObjFullLineAttrElement

```
GrObjFullLineAttrElement      struct
    GOFLAE_base      GrObjBaseLineAttrElement
    GOFLAE_future     byte FUTURE_LINE_ATTR_ELEMENT_DATA_SIZE dup (?)
GrObjFullLineAttrElement      ends
```

This structure is used to allow future routines to access larger **GrObjBaseLineAttrElement** structures.

Library: **grobj.def**

■ GrObjFunctionsActive

```
GrObjFunctionsActive          record
    GOFA_RULER_HAS_SEEN_EVENT :1
    GOFA_VIEW_ZOOMED          :1
    GOFA_SNAP_TO              :1
    GOFA_FROM_CENTER          :1
    GOFA_ABOUT_OPPOSITE       :1
    GOFA_CONSTRAIN            :1
    GOFA_ADJUST               :1
    GOFA_EXTEND               :1
                                :2
GrObjFunctionsActive          end
```

GOFA_RULER_HAS_SEEN_EVENT

If set, the mouse event has been sent to the ruler already. This is used to prevent snapping the mouse more often than is needed. For example, when moving multiple GrObjs, you only want to snap the mouse once and not for each GrObj.

GOFA_VIEW_ZOOMED

If set, all drawing operations should be done in high resolution mode.

GOFA_SNAP_TO

If set, operations should be snapped to the grid.

Revision: ■

Draft Dated (4/18/94)

Reference book



GOFA_FROM_CENTER

If set, any resize or create operations should be performed from the center.

GOFA_ABOUT_OPPOSITE

If set, rotations should be performed about the opposite corner.

GOFA_CONSTRAIN

If set, constrain resize, rotate, etc.

GOFA_ADJUST

Same as UIFA_ADJUST.

GOFA_EXTEND

Same as UIFA_EXTEND.

Library: **grobj.def**

■ GrObjGradientAreaAttrElement

```
GrObjGradientAreaAttrElement struct
  GOGAAE_base          GrObjBaseAreaAttrElement
  GOGAAE_type          GrObjGradientType
  GOGAAE_endR          byte          ;ending color red byte
  GOGAAE_endG          byte          ;ending color green byte
  GOGAAE_endB          byte          ;ending color blue byte
  GOGAAE_numIntervals  word
  GOGAAE_reserved      word
GrObjGradientAreaAttrElement ends
```

Library: **grobj.def**

■ GrObjGradientAttrDiffs

```
GrObjGradientAttrDiffs record
  GGAD_MULTIPLE_END_COLORS :1
  GGAD_MULTIPLE_TYPES      :1
  GGAD_MULTIPLE_INTERVALS :1
                          :4
  GGAD_FIRST_RECIPIENT     :1
GrObjGradientAttrDiffs end
```

GGAD_FIRST_RECIPIENT

If set, the GrObj knows that it's the first one to receive this data buffer (and should clear it).

Library: **grobj.def**

■ GrObjGradientFillControlFeatures

```
GrObjGradientFillControlFeatures  record
    GOGFCF_HORIZONTAL_GRADIENT    :1
    GOGFCF_VERTICAL_GRADIENT      :1
    GOGFCF_RADIAL_RECT_GRADIENT    :1
    GOGFCF_RADIAL_ELLIPSE_GRADIENT:1
    GOGFCF_NUM_INTERVALS          :1
GrObjGradientFillControlFeatures  end
```

Library: **grobj.def**

■ GrObjGradientType

```
GrObjGradientType  etype byte
    GOGT_NONE                enum GrObjGradientType
    GOGT_LEFT_TO_RIGHT        enum GrObjGradientType
    GOGT_TOP_TO_BOTTOM        enum GrObjGradientType
    GOGT_RADIAL_RECT          enum GrObjGradientType
    GOGT_RADIAL_ELLIPSE       enum GrObjGradientType
```

Library: **grobj.def**

■ GrObjHandleAnchorData

```
GrObjHandleAnchorData  struct
    GOHAD_anchor         PointDWFxed
    GOHAD_handle         GrObjHandleSpecification
    align                word
GrObjHandleAnchorData  ends
```

Library: **grobj.def**

■ GrObjHandleSpecification

```
GrObjHandleSpecification  record
    GOHS_HANDLE_LEFT      :1
    GOHS_HANDLE_TOP       :1
    GOHS_HANDLE_RIGHT     :1
    GOHS_HANDLE_BOTTOM    :1
GrObjHandleSpecification  end
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjInitializeData

```
GrObjInitializeData      struct
    GOID_position      PointDWFxed
    GOID_width          WWFixed
    GOID_height         WWFixed
GrObjInitializeData      ends
```

GOID_position stores the position of the upper left corner of the object in *parent* coordinates.

GOID_width stores the width of the object in points.

GOID_height stores the height of the object in points.

Library: **grobj.def**

■ GrObjInstructionControlFeatures

```
GrObjInstructionControlFeatures      record
    GOICF_DRAW                      :1
    GOICF_PRINT                     :1
    GOICF_MAKE_EDITABLE             :1
    GOICF_MAKE_UNEDITABLE           :1
    GOICF_DELETE                    :1
GrObjInstructionControlFeatures      end
```

Library: **grobj.def**

■ GrObjLineAttrElementType

```
GrObjLineAttrElementType  etype byte
    GOLAET_BASE             enum GrObjLineAttrElementType
;GrObjBaseLineAttrElement
```

Library: **grobj.def**

■ GrObjLineAttrInfoRecord

```
GrObjLineAttrInfoRecord      record
    GOLAIR_ARROWHEAD_ON_START                      :1
    GOLAIR_ARROWHEAD_ON_END                        :1
    GOLAIR_ARROWHEAD_FILLED                        :1
    GOLAIR_ARROWHEAD_FILL_WITH_AREA_ATTRIBUTES     :1
                                                    :4
GrObjLineAttrInfoRecord      end
```

Library: **grobj.def**

■ GrObjLocks

```
GrObjLocks      record
  GOL_COPY:1      ;True if object may not be transferred to the
                  ;clipboard
  GOL_LOCK        :1      ;True if object cannot have its locks changed
  GOL_SHOW        :1      ;True if object can't be drawn/selected/edited
                  ;Used with MSG_GB_HIDE_UNSELECTED_OBJECTS
                  ;and MSG_GB_SHOW_ALL_OBJECTS
  GOL_WRAP        :1      ;True if can't change wrap type
  GOL_MOVE        :1      ;True if object cannot be moved
  GOL_RESIZE      :1      ;True if object cannot be resized
  GOL_ROTATE      :1      ;True if object cannot be rotated
  GOL_SKEW        :1      ;True if object cannot be skewed
  GOL_EDIT        :1      ;True if object cannot be edit
  GOL_DELETE      :1      ;True if object cannot be deleted
  GOL_SELECT      :1      ;True if object cannot be selected
  GOL_ATTRIBUTE   :1      ;True if object cannot change attributes
  GOL_GROUP       :1      ;True if object cannot be grouped
  GOL_UNGROUP     :1      ;True if group cannot be ungrouped
  GOL_DRAW        :1      ;True if object cannot be drawn
  GOL_PRINT       :1      ;True if object cannot be printed
GrObjLocks      end
```

Library: **grobj.def**

■ GrObjMessageOptimizationFlags

```
GrObjMessageOptimizationFlags record
  GOMOF_GET_DWF_SELECTION_HANDLE_BOUNDS_FOR_TRIVIAL_REJECT:1
  GOMOF_SPECIAL_RESIZE_CONSTRAIN      :1
  GOMOF_INVALIDATE_LINE               :1
  GOMOF_INVALIDATE_AREA               :1
  GOMOF_INVALIDATE                   :1
  GOMOF_NOTIFY_ACTION                 :1
  GOMOF_SEND_UI_NOTIFICATION          :1
  GOMOF_DRAW_FG_AREA                 :1
  GOMOF_DRAW_FG_LINE                 :1
  GOMOF_DRAW_BG                     :1
GrObjMessageOptimizationFlags end
```

Each bit below corresponds to an often used, but seldom subclassed message. Instead of sending the message to itself, objects will call a utility routine (such as **GrObjOptInvalidate**). The routine will check the objects GOMOF flags and if the corresponding bit is set it will send the message to itself. Otherwise the routine will perform the default functionality.

Revision: ■

Draft Dated (4/18/94)

Reference book



These bits should only be set in a MSG_META_INITIALIZE handler as they are class- and not object-specific. They are not copied to the clipboard.

GOMOF_GET_DWF_SELECTION_HANDLE_BOUNDS_FOR_TRIVIAL_REJECT

If set, sends

MSG_GO_GET_DWF_SELECTION_HANDLE_BOUNDS_FOR_TRIVIAL_REJECT.

GOMOF_SPECIAL_RESIZE_CONSTRAIN

If set, sends MSG_GO_SPECIAL_RESIZE_CONSTRAIN.

GOMOF_INVALIDATE_LINE

If set, sends MSG_GO_INVALIDATE_LINE.

GOMOF_INVALIDATE_AREA

If set, sends MSG_GO_INVALIDATE_AREA.

GOMOF_INVALIDATE

If set, sends MSG_GO_INVALIDATE.

GOMOF_NOTIFY_ACTION

If set, sends MSG_GO_NOTIFY_ACTION.

GOMOF_SEND_UI_NOTIFICATION

If set, sends MSG_GO_SEND_UI_NOTIFICATION.

The following bits correspond to several drawing messages that are almost always sent during the handling of MSG_GO_DRAW. Under certain conditions these message are not sent. If the line or area mask is 0 then the line or area drawing messages are not sent. If the area mask is solid, then the background message is not sent. If you wish to force one of these messages to be sent anyway then set its corresponding bit. For example, the text object sets the GOMOF_DRAW_BG bit because the foreground is text which doesn't completely cover the background rectangle, so the background should always be drawn.

Note: if the background is transparent then the background message will never be sent, regardless of the presence of the GOMOF_DRAW_BG bit.

GOMOF_DRAW_FG_AREA

If set, sends MSG_GO_DRAW_FG_AREA(_HI_RES).

GOMOF_DRAW_FG_LINE

If set, sends MSG_GO_DRAW_FG_LINE(_HI_RES).

GOMOF_DRAW_BG

If set, sends MSG_GO_DRAW_BG.

Library: **grobj.def**

■ GrObjMouseDown

```
GrObjMouseDown      struct
  GOMD_point         PointDWFxed           ; This field must be first.
  GOMD_buttonInfo    ButtonInfo           ; Copy of ButtonInfo
  GOMD_uiFA          UIFunctionsActive     ; Copy of UIFunctionsActive
  GOMD_goFA          GrObjFunctionsActive
  GOMD_gstate        hptr.GState
GrObjMouseDown      ends
```

Library: **grobj.def**

■ GrObjMouseReturnType

```
GrObjMouseReturnType  etype byte
  GOMRF_HANDLE        enum GrObjMouseReturnType ;Mouse position is over a handle
                                                             ;of a selected object.
  GOMRF_BOUNDS        enum GrObjMouseReturnType ;Mouse position is over the
                                                             ;bounds of a grobject
  GOMRF_NOTHING       enum GrObjMouseReturnType ;Mouse position isn't over
                                                             ;anything interesting.
```

These types are defined in the order in which they will be checked. As soon as one of the conditions is met the message returns without checking the remaining types.

Library: **grobj.def**

■ GrObjNotifyAreaAttrChange

```
GrObjNotifyAreaAttrChange  struct
  GNAAC_areaAttr          GrObjBaseAreaAttrElement
  GNAAC_areaAttrDiffs     GrObjBaseAreaAttrDiffs
GrObjNotifyAreaAttrChange  ends
```

Library: **grobj.def**

■ GrObjNotifyCurrentTool

```
GrObjNotifyCurrentTool    struct
  GONCT_toolClass         fptr.ClassStruct
  GONCT_specInitData      word
GrObjNotifyCurrentTool    ends
```

Library: **grobj.def**

Revision: ■
Draft Dated (4/18/94)

Reference book



■ GrObjNotifyGradientAttrChange

```
GrObjNotifyGradientAttrChange    struct
    GONGAC_type                  GrObjGradientType
    GONGAC_endR                  byte                ;ending color red byte
    GONGAC_endG                  byte                ;ending color green byte
    GONGAC_endB                  byte                ;ending color blue byte
    GONGAC_numIntervals          word
    GONGAC_diffs                 GrObjGradientAttrDiffs
    align                        word
GrObjNotifyGradientAttrChange    ends
```

Library: **grobj.def**

■ GrObjNotifyLineAttrChange

```
GrObjNotifyLineAttrChange        struct
    GNLAC_lineAttr              GrObjBaseLineAttrElement
    GNLAC_lineAttrDiffs         GrObjBaseLineAttrDiffs
GrObjNotifyLineAttrChange        ends
```

Library: **grobj.def**

■ GrObjNotifySelectionStateChange

```
GrObjNotifySelectionStateChange   struct
    GONSSC_selectionState       GrObjSelectionState
    GONSSC_selectionStateDiffs  GrObjSelectionStateDiffs
    GONSSC_grObjFlagsDiffs      GrObjAttrFlags
    GONSSC_locksDiffs           GrObjLocks
    GONSSC_arcCloseType         ArcCloseType
    GONSSC_arcStartAngle        WWFixed
    GONSSC_arcEndAngle          WWFixed
GrObjNotifySelectionStateChange   ends
```

Library: **grobj.def**

■ **GrObjNudgeControlFeatures**

```
GrObjNudgeControlFeatures      record
    GONCF_NUDGE_LEFT           :1
    GONCF_NUDGE_RIGHT          :1
    GONCF_NUDGE_UP             :1
    GONCF_NUDGE_DOWN           :1
    GONCF_CUSTOM_MOVE          :1
GrObjNudgeControlFeatures      end
```

Library: **grobj.def**

■ **GrObjObjManipData**

```
GrObjObjManipData      struct
    GOOMD_actionGrObj      optr
    GOOMD_origMousePt      PointDWFxed
    GOOMD_oppositeHandle    GrObjHandleSpecification
    GOOMD_grabbedHandle     GrObjHandleSpecification
    GOOMD_initialAngle      WWFxed
    GOOMD_oppositeAnchor    PointDWFxed
    GOOMD_oppositeInitialAngleWWFxed
GrObjObjManipData      ends
```

Library: **grobj.def**

■ **GrObjObscureAttrControlFeatures**

```
GrObjObscureAttrControlFeatures      record
    GOOACF_INSTRUCTIONS           :1
    GOOACF_INSERT_OR_DELETE_MOVE :1
    GOOACF_INSERT_OR_DELETE_RESIZE:1
    GOOACF_INSERT_OR_DELETE_DELETE:1
    GOOACF_DONT_WRAP              :1
    GOOACF_WRAP_INSIDE            :1
    GOOACF_WRAP_AROUND_RECT       :1
    GOOACF_WRAP_TIGHTLY          :1
GrObjObscureAttrControlFeatures      end
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjOptimizationFlags

```
GrObjOptimizationFlags    record
    GOOF_ADDED_TO_BODY      :1
    GOOF_IN_GROUP           :1
    GOOF_GROBJ_INVALID      :1
    GOOF_ATTRIBUTE_MANAGER  :1
    GOOF_FLOATER            :1
    GOOF_HAS_ACTION_NOTIFICATION :1
    GOOF_HAS_UNBALANCED_PARENT_DIMENSIONS:1
GrObjOptimizationFlags    end
```

GOOF_ADDED_TO_BODY

If set, the Grobj has been added to a body, or the group it is in has been added to a body.

GOOF_IN_GROUP

If set, the GrObj is within a group.

GOOF_GROBJ_INVALID

If set, the object is incomplete and cannot be drawn, or it is invalidated. It may be missing its normal transform or have no attributes, etc.

GOOF_ATTRIBUTE_MANAGER

If set, the object is an attribute manager.

GOOF_FLOATER

If set, GrObj is a floater. If this flag is set, we don't need to dirty the object because floater objects are not actually in the document.

GOOF_HAS_ACTION_NOTIFICATION

If set, object has an action notification OD in it's vardata.

GOOF_HAS_UNBALANCED_PARENT_DIMENSIONS

If set, the object contains the vardata entry ATTR_GO_PARENT_DIMENSIONS_OFFSET. This ATTR holds the offset from the object's center to the center of the parent dimensions.

Library: **grobj.def**

■ GrObjPointerImageSituation

```
GrObjPointerImageSituation    etype byte
    GOPIS_NORMAL              enum GrObjPointerImageSituation
    GOPIS_EDIT                enum GrObjPointerImageSituation
    GOPIS_CREATE              enum GrObjPointerImageSituation
    GOPIS_MOVE                enum GrObjPointerImageSituation
    GOPIS_RESIZE_ROTATE       enum GrObjPointerImageSituation
```

Library: **grobj.def**

■ GrObjResizeMouseData

```
GrObjResizeMouseData         struct
    GORSMD_point              PointDWFixed           ; Must be first.
    GORSMD_anchor             GrObjHandleSpecification
    GORSMD_grabbed            GrObjHandleSpecification
    GORSMD_goFA               GrObjFunctionsActive
    GORSMD_gstate             hptr.GState
    align                     word
GrObjResizeMouseData         ends
```

Library: **grobj.def**

■ GrObjRotateControlFeatures

```
GrObjRotateControlFeatures    record
    GORCF_45_DEGREES_CW       :1
    GORCF_90_DEGREES_CW       :1
    GORCF_135_DEGREES_CW      :1
    GORCF_180_DEGREES         :1
    GORCF_135_DEGREES_CCW     :1
    GORCF_90_DEGREES_CCW      :1
    GORCF_45_DEGREES_CCW      :1
    GORCF_CUSTOM_ROTATION     :1
GrObjRotateControlFeatures    end
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjRotateMouseData

```
GrObjRotateMouseData    struct
    GORMD_degrees        WWFixed
    GORMD_anchor          GrObjHandleSpecification
    GORMD_goFA            GrObjFunctionsActive
    GORMD_gstate          hptr.GState
    align                 word
GrObjRotateMouseData    ends
```

This structure is the stack frame passed with rotate message.

Library: **grobj.def**

■ GrObjScaleControlFeatures

```
GrObjScaleControlFeatures    record
    GOSCF_HALF_WIDTH        :1
    GOSCF_HALF_HEIGHT       :1
    GOSCF_DOUBLE_WIDTH      :1
    GOSCF_DOUBLE_HEIGHT     :1
    GOSCF_CUSTOM_SCALE      :1
GrObjScaleControlFeatures    end
```

Library: **grobj.def**

■ GrObjScaleData

```
GrObjScaleData              struct
    GOSD_xScale             WWFixed
    GOSD_yScale             WWFixed
GrObjScaleData              ends
```

Library: **grobj.def**

■ GrObjSelectionState

```
GrObjSelectionState         struct
    GSS_numSelected          word
    GSS_classSelected        fptr.ClassStruct
    GSS_flags                GrObjSelectionStateFlags
    GSS_grObjFlags           GrObjAttrFlags
    GSS_locks                GrObjLocks
    align                    word
GrObjSelectionState         ends
```

Library: **grobj.def**

■ GrObjSelectionStateDiffs

```
GrObjSelectionStateDiffs record
    GSSD_MULTIPLE_CLASSES      :1
    GSSD_MULTIPLE_ARC_CLOSE_TYPES :1
    GSSD_MULTIPLE_ARC_START_ANGLES :1
    GSSD_MULTIPLE_ARC_END_ANGLES :1
                                :4
GrObjSelectionStateDiffs end
```

Library: **grobj.def**

■ GrObjSelectionStateFlags

```
GrObjSelectionStateFlags record
    GSSF_EDITING      :1 ;True if an object is being edited.
    GSSF_UNGROUPABLE  :1 ;True if at least one of the objects
                        ;selected can be ungrouped.
    GSSF_TEXT_SELECTED :1 ;True if at least one of the objects
                        ;selected is some sort of text object
    GSSF_BITMAP_SELECTED :1 ;True if at least one of the objects
                        ;selected is some sort of bitmap object
    GSSF_SPLINE_SELECTED :1 ;True if at least one of the objects
                        ;selected is some sort of spline object
    GSSF_ARC_SELECTED   :1 ;True if at least one of the objects
                        ;selected is some sort of arc object
GrObjSelectionStateFlags end
```

Library: **grobj.def**

■ GrObjsInRectData

```
GrObjsInRectData struct
    GOIRD_tempMessage      word
    GOIRD_tempMessageDX    word
    GOIRD_inRectMessage    word
    GOIRD_inRectMessageDX  word
    GOIRD_rect              RectDWord
    GOIRD_special           GrObjsInRectSpecial
    align                  word
GrObjsInRectData ends
```

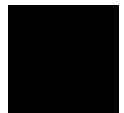
GOIRD_tempMessage stores the message to send to an object that has its GOTM_TEMP_HANDLES bit set.

GOIRD_tempMessageDX stores the word of data that can be passed with the above temporary message in **dx**.

Revision: ■

Draft Dated (4/18/94)

Reference book



GOIRD_inRectMessage stores the message to send to an object if it is found to reside within the **Rectangle** specified by *GOIRD_rect*.

GOIRD_inRectMessageDX stores the word of data that can be passed with the above message in **dx**.

GOIRD_rect stores the **Rectangle** that we are checking whether the object resides within.

GOIRD_special stores special instructions for processing children.

Library: **grobj.def**

■ GrObjsInRectSpecial

```
GrObjsInRectSpecial      record
  GOIRS_IGNORE_TEMP      :1
  GOIRS_IGNORE_RECT      :1
  GOIRS_XOR_CHECK        :1
GrObjsInRectSpecial      end
```

GOIRS_IGNORE_TEMP

If set, do not send the GrObj's Temp Message to objects with GOTM_TEMP_HANDLES set.

GOIRS_IGNORE_RECT

If set, do not send the GrObj's InRect Message to objects within the rectangle.

GOIRS_XOR_CHECK

If set and both the Temp and InRect conditions apply, then send neither message. Otherwise send both messages. (The Temp Message will always be sent first.)

Library: **grobj.def**

■ GrObjSkewControlFeatures

```
GrObjSkewControlFeatures record
  GOSCF_LEFT             :1
  GOSCF_RIGHT            :1
  GOSCF_UP               :1
  GOSCF_DOWN             :1
  GOSCF_CUSTOM_SKEW      :1
GrObjSkewControlFeatures end
```

Library: **grobj.def**

■ GrObjSkewData

```
GrObjSkewData      struct
    GOSD_xDegrees   WWFixed
    GOSD_yDegress   WWFixed
GrObjSkewData      ends
```

Library: **grobj.def**

■ GrObjStyleElement

```
GrObjStyleElement  struct
    GSE_meta          NameArrayElement
    GSE_baseStyle      word
    GSE_flags          StyleElementFlags
    GSE_reserved       byte 6 dup (?)
    GSE_privateData    GrObjStylePrivateData
    GSE_areaAttrToken  word
    GSE_lineAttrToken  word
    GSE_name           label char
GrObjStyleElement  ends
```

Library: **grobj.def**

■ GrObjStyleFlags

```
GrObjStyleFlags    record
    GSF_AREA_COLOR_RELATIVE  :1
    GSF_AREA_MASK_RELATIVE   :1
    GSF_LINE_COLOR_RELATIVE   :1
    GSF_LINE_MASK_RELATIVE    :1
    GSF_LINE_WIDTH_RELATIVE   :1
                                :11
GrObjStyleFlags     end
```

Library: **grobj.def**

■ GrObjStylePrivateData

```
GrObjStylePrivateData  struct
    GSPD_flags          GrObjStyleFlags
    GSPD_unused         byte 2 dup (0)
GrObjStylePrivateData  ends
```

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjTempModes

```
GrObjTempModes      record
    GOTM_SELECTED      :1
    GOTM_EDITED        :1
    GOTM_EDIT_INDICATOR_DRAWN :1
    GOTM_HANDLES_DRAWN :1
    GOTM_TEMP_HANDLES  :1
    GOTM_SPRITE_DRAWN  :1
    GOTM_SPRITE_DRAWN_HI_RES :1
    GOTM_SYS_TARGET    :1
GrObjTempModes      end
```

GOTM_SELECTED

If set, the GrObj is in the selection list.

GOTM_EDITED

If set, the GrObj is currently being edited. This is equivalent to the GrObj having the application target.

GOTM_EDIT_INDICATOR_DRAWN

If set, the object has drawn some indicator to show the user that it is being edited.

GOTM_HANDLES_DRAWN

If set, the GrObj's selection handles have been drawn.

GOTM_TEMP_HANDLES

If set, the GrObj's handles are drawn in a temporary state. This flag is set by MSG_GO_DRAW_HANDLES_FORCE, MSG_GO_DRAW_HANDLES_OPPOSITE and cleared by MSG_GO_DRAW_HANDLES_MATCH, MSG_GO_DRAW_HANDLES and MSG_GO_UNDRAW_HANDLES. This functionality is mainly used when drag selecting to cut down the number of objects that methods must be sent to.

GOTM_SPRITE_DRAWN

If set, the object's sprite has been drawn.

GOTM_SPRITE_DRAWN_HI_RES

If set, the object's sprite was drawn at a higher resolution. This bit is meaningless if GOTM_SPRITE_DRAWN is not set.

GOTM_SYS_TARGET

If set, the GrObjBody has the system target. This bit is meaningless unless GOTM_SELECTED or GOTM_EDITED is set. (The GrObjBody only updates objects in selection list and with the application target.)

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

■ GrObjTextArrays

```
GrObjTextArrays      struct
    GOTA_charAttrArray    word
    GOTA_paraAttrArray    word
    GOTA_typeArray        word
    GOTA_graphicArray     word
    GOTA_nameArray        word
    GOTA_textStyleArray   word
GrObjTextArrays      ends
```

Library: **grobj.def**

■ GrObjTiledDataFlags

```
GrObjTiledDataFlags      record
                        :7
    GOTDF_VAR_DATA      :1
GrObjTiledDataFlags      end
```

Library: **grobj.def**

■ GrObjTransferBlockHeader

```
GrObjTransferBlockHeader      struct
    GOTBH_meta              VMChainTree
    GOTBH_size              PointDWord    ; Width and height of the cut.

    GOTBH_firstLMem         label word
    GOTBH_areaAttrArray     dword
    GOTBH_lineAttrArray     dword
    GOTBH_styleArray        dword
    GOTBH_charAttrRuns      dword
    GOTBH_paraAttrRuns      dword
    GOTBH_textStyleArray    dword
    GOTBH_lastLMem          label word

    GOTBH_textGraphicsTree  dword
GrObjTransferBlockHeader      ends
```

This structure heads a “transfer” item, which is used by the Clipboard in Cut/Copy/Paste operations.

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjTransferDataDirectory

```
GrObjTransferDataDirectory    struct
    GOTDD_tiledDataFlags      GrObjTiledDataFlags
    GOTDD_protocol             byte
GrObjTransferDataDirectory    ends
```

Library: **grobj.def**

■ GrObjTransferParams

```
GrObjTransferParams           struct
    GTP_ssp                    StyleSheetParams
    GTP_textSSP                VisTextSaveStyleSheetParams
    GTP_selectionCenterDOCUMENT PointDWFfixed
    GTP_optBlock               hptr
    GTP_treeBlock              hptr
    GTP_curSlot                word
    GTP_id                     dword
    GTP_curSize                word
    GTP_curPos                 word
GrObjTransferParams           ends
```

Library: **grobj.def**

■ GrObjTransMatrix

```
GrObjTransMatrix             struct
    GTM_e11                    WWFixed
    GTM_e12                    WWFixed
    GTM_e21                    WWFixed
    GTM_e22                    WWFixed
GrObjTransMatrix             ends
```

Library: **grobj.def**

■ GrObjUINotificationTypes

```
GrObjUINotificationTypes record
    GOUINT_STYLE           :1      ;True if style notification needs to be
                                   ;sent
    GOUINT_AREA            :1      ;True if area notification needs to be sent
    GOUINT_LINE            :1      ;True if line notification needs to be sent
    GOUINT_GROBJ_SELECT    :1      ;True if grobj specific selection state
                                   ;notification needs to be sent
    GOUINT_STYLE_SHEET     :1      ;True if style notification needs to be
                                   ;sent
    GOUINT_SELECT          :1      ;True if edit menu notification need be
                                   ;sent
    :10                    ;unused
GrObjUINotificationTypes end
```

Library: **grobj.def**

■ GrObjUndoAppType

```
GrObjUndoAppType struct
    GOUAT_freeMessage      word
    GOUAT_undoMessage      word
GrObjUndoAppType ends
```

Library: **grobj.def**

■ GrObjVisGuardianCreateMode

```
GrObjVisGuardianCreateMode    etype byte, 0
GOVGCM_NO_CREATE              enum GrObjVisGuardianCreateMode
GOVGCM_GUARDIAN_CREATE        enum GrObjVisGuardianCreateMode
GOVGCM_VIS_WARD_CREATE        enum GrObjVisGuardianCreateMode
```

GOVGCM_NO_CREATE

This type indicates that new objects cannot be created with the guardian.

GOVGCM_GUARDIAN_CREATE

This type indicates that creating a new object is handled by the guardian and consists of dragging open a rectangular area.

GOVGCM_VIS_WARD_CREATE

This type indicates that creating a new object is handled by the ward and mouse events during the create operation should be sent to the ward.

Library: **grobj.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ GrObjVisGuardianFlags

```
GrObjVisGuardianFlags    record
    GOVGF_VIS_BOUNDS_HAVE_CHANGED    :1
    GOVGF_LARGE                :1
    GOVGF_UNUSED                :1
    GOVGF_ALSO_UNUSED           :1
    GOVGF_APPLY_OBJECT_TO_VIS_TRANSFORM :1
    GOVGF_CAN_EDIT_EXISTING_OBJECTS :1
    GOVGF_CREATE_MODE           GrObjGuardianCreateMode:2
GrObjVisGuardianFlags    end
```

GOVGF_VIS_BOUNDS_HAVE_CHANGED

If set, the visible bounds of the ward have changed since the last time the bit was cleared. Each guardian uses this bit differently to help it determine when to send out GOANT_RESIZE action notifications.

GOVGF_LARGE

If set, send large mouse events to the object. If clear, send small mouse events instead.

GOVGF_APPLY_OBJECT_TO_VIS_TRANSFORM

If set, you must send MSG_GOVG_APPLY_OBJECT_TO_VIS_TRANSFORM to the object; otherwise a utility routine can be used.

GOVGF_CAN_EDIT_EXISTING_OBJECTS

If set, the floater can edit existing objects in the document.

Library: **grobj.def**

■ GrObjWrapTextType

```
GrObjWrapTextType    etype byte, 0
    GOWTT_DONT_WRAP        enum GrObjWrapTextType
    GOWTT_WRAP_AROUND_RECT enum GrObjWrapTextType
    GOWTT_WRAP_AROUND_TIGHTLY enum GrObjWrapTextType
    GOWTT_WRAP_INSIDE      enum GrObjWrapTextType
```

Library: **grobj.def**

■ GroupAddGrObjFlags

```
GroupAddGrObjFlags record
    GAGOF_RELATIVE :1
    GAGOF_REFERENCE :15
GroupAddGrObjFlags end
```

GAGOF_RELATIVE

This flag indicates that the position of the center of the object is already relative to the center of the group. Otherwise, the object center is absolute and must be adjusted when it is added.

Library: **grobj.def**

■ GroupUnsuspendOps

```
GroupUnsuspendOps record
    GUO_EXPAND :1
GroupUnsuspendOps end
```

GUO_EXPAND

If set, the group should send MSG_GROUP_EXPAND to itself when its suspend count reaches zero.

Library: **grobj.def**

■ GSControl

```
GSControl record
    :6,
    GSC_PARTIAL :1 ; Just do one element. If element is a complex
                    ; bit map, do just one piece. (This flag works
                    ; only with GrCopyGString.)
    GSC_ONE :1, ; just do one element
    GSC_MISC :1, ; return on MISC opcode
    GSC_LABEL :1, ; return on GR_LABEL opcode
    GSC_ESCAPE :1, ; return on GR_ESCAPE opcode
    GSC_NEW_PAGE :1, ; return when we get to a NEW_PAGE
    GSC_XFORM :1, ; return on TRANSFORMATIONopcode
    GSC_OUTPUT :1, ; return on OUTPUT opcode
    GSC_ATTR :1, ; return on ATTRIBUTE opcode
    GSC_PATH :1, ; return on PATH opcode
GSControl end
```

Library: **gstring.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **GSElemInfo**

```
GSElemInfo          struct
    GSEI_size        word
    GSEI_play         nptr
    GSEI_kern         fptr
GSElemInfo          ends
```

Library: **gstring.def**

■ **GSRefCountAndFlags**

```
GSRefCountAndFlags  record
    GSRCAF_USE_DOC_CLIP_REGION:1 ;If set, then use GrSetDocClipRect
                                ;instead of GrSetClipRect
    GSRCAF_REF_COUNT          :7
GSRefCountAndFlags  end
```

Library: **Objects/vTextC.def**

■ **GSRetType**

```
GSRetType          etype word
    GSRT_COMPLETE  enum GSRetType
    GSRT_ONE       enum GSRetType
    GSRT_MISCEnum  GSRetType
    GSRT_LABELenum GSRetType
    GSRT_ESCAPE    enum GSRetType
    GSRT_NEW_PAGE  enum GSRetType
    GSRT_XFORM     enum GSRetType
    GSRT_OUTPUT    enum GSRetType
    GSRT_ATTR      enum GSRetType
    GSRT_PATH      enum GSRetType
    GSRT_FAULT     enum GSRetType, 0xffff
```

Library: **gstring.def**

■ GStringElement

GStringElement etype byte, 0, 1

```
DefGSElement     macro     \
                     rootName, gseEnum, playRout, varType, varOff, altKern
gseEnum enum GStringElement
```

```
DefGSElement EndGString, GR_END_GSTRING, PENull
DefGSElement Comment,    GR_COMMENT,        Pecomment, bytes, OC_size
DefGSElement NullOp,     GR_NULL_OP,        PENoArgs
DefGSElement SetGStringBounds, GR_SET_GSTRING_BOUNDS, PETwoCoords
DefGSElement             ,        GR_MISC_4,        PENull
DefGSElement             ,        GR_MISC_5,        PENull
DefGSElement             ,        GR_MISC_6,        PENull
DefGSElement             ,        GR_MISC_7,        PENull
DefGSElement             ,        GR_MISC_8,        PENull
DefGSElement             ,        GR_MISC_9,        PENull
DefGSElement             ,        GR_MISC_A,        PENull
DefGSElement             ,        GR_MISC_B,        PENull
DefGSElement             ,        GR_MISC_C,        PENull
DefGSElement Label,       GR_LABEL,         PEWordAttr
DefGSElement Escape,      GR_ESCAPE,        Pecomment, bytes, OE_escSize
DefGSElement NewPage,    GR_NEW_PAGE,     PEByteAttr
DefGSElement ApplyRotation, GR_APPLY_ROTATION, PERotate
DefGSElement ApplyScale, GR_APPLY_SCALE,     PETransScale
DefGSElement ApplyTranslation, GR_APPLY_TRANSLATION, PETransScale
DefGSElement ApplyTransform, GR_APPLY_TRANSFORM, PETMatrix
DefGSElement ApplyTranslationDWord, GR_APPLY_TRANSLATION_DWORD, PETransScale
DefGSElement SetTransform, GR_SET_TRANSFORM,    PETMatrix
DefGSElement SetNullTransform, GR_SET_NULL_TRANSFORM, PENoArgs
DefGSElement SetDefaultTransform, GR_SET_DEFAULT_TRANSFORM, PENoArgs
DefGSElement InitDefaultTransform, GR_INIT_DEFAULT_TRANSFORM, PENoArgs
DefGSElement SaveTransform, GR_SAVE_TRANSFORM, PENoArgs
DefGSElement RestoreTransform, GR_RESTORE_TRANSFORM, PENoArgs
DefGSElement             ,        GR_XFORM_1B, PENull
DefGSElement             ,        GR_XFORM_1C, PENull
DefGSElement             ,        GR_XFORM_1D, PENull
DefGSElement             ,        GR_XFORM_1E, PENull
DefGSElement             ,        GR_XFORM_1F, PENull
DefGSElement DrawLine, GR_DRAW_LINE,        PETwoCoords
DefGSElement DrawLineTo, GR_DRAW_LINE_TO, PEOneCoord
DefGSElement DrawRelLineTo, GR_DRAW_REL_LINE_TO, PERelCoord
DefGSElement DrawHLine, GR_DRAW_HLINE, PEDrawHalfLine
DefGSElement DrawHLineTo, GR_DRAW_HLINE_TO, PEDrawHalfLine
DefGSElement DrawVLine, GR_DRAW_VLINE, PEDrawHalfLine
DefGSElement DrawVLineTo, GR_DRAW_VLINE_TO, PEDrawHalfLine
DefGSElement DrawPolyline, GR_DRAW_POLYLINE, PEPolyCoord, coords, ODPL_count
DefGSElement DrawArc,    GR_DRAW_ARC, PEDrawArcs
```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

DefGSElement DrawArc3Point, GR_DRAW_ARC_3POINT, PEDrawArcs
DefGSElement DrawArc3PointTo, GR_DRAW_ARC_3POINT_TO, PEDrawArcs
DefGSElement DrawRelArc3PointTo, GR_DRAW_REL_ARC_3POINT_TO, PEDrawArcs
DefGSElement DrawRect, GR_DRAW_RECT, PETwoCoords
DefGSElement DrawRectTo, GR_DRAW_RECT_TO, PEOneCoord
DefGSElement DrawRoundRect, GR_DRAW_ROUND_RECT, PEDrawRoundRects
DefGSElement DrawRoundRectTo, GR_DRAW_ROUND_RECT_TO, PEDrawRoundRects
DefGSElement DrawSpline, GR_DRAW_SPLINE, PEPolyCoord, coords, ODS_count
DefGSElement DrawSplineTo, GR_DRAW_SPLINE_TO, PEPolyCoord, coords,
ODST_count
DefGSElement DrawCurve, GR_DRAW_CURVE, PECurve
DefGSElement DrawCurveTo, GR_DRAW_CURVE_TO, PECurve
DefGSElement DrawRelCurveTo, GR_DRAW_REL_CURVE_TO, PECurve
DefGSElement DrawEllipse, GR_DRAW_ELLIPSE, PETwoCoords
DefGSElement DrawPolygon, GR_DRAW_POLYGON, PEPolyCoord, coords, ODPG_count
DefGSElement DrawPoint, GR_DRAW_POINT, PEOneCoord
DefGSElement DrawPointAtCP, GR_DRAW_POINT_CP, PENOArgs
DefGSElement BrushPolyline, GR_BRUSH_POLYLINE, PEPolyCoord, coords,
OBPL_count
DefGSElement DrawChar, GR_DRAW_CHAR, PEDrawChar
DefGSElement DrawCharAtCP, GR_DRAW_CHAR_CP, PEDrawChar
DefGSElement DrawText, GR_DRAW_TEXT, PEDrawText, bytes, ODT_len
DefGSElement DrawTextAtCP, GR_DRAW_TEXT_CP, PEDrawText, bytes, ODTCP_len
DefGSElement DrawTextField, GR_DRAW_TEXT_FIELD, PETextField
DefGSElement DrawTextPtr, GR_DRAW_TEXT_PTR, PETextPtr,,,GrDrawText

DefGSElement DrawTextOpPtr, GR_DRAW_TEXT_OPTR, PETextOpPtr,,,GrDrawText
DefGSElement DrawPath, GR_DRAW_PATH, PENOArgs
DefGSElement FillRect, GR_FILL_RECT, PETwoCoords
DefGSElement FillRectTo, GR_FILL_RECT_TO, PEOneCoord
DefGSElement FillRoundRect, GR_FILL_ROUND_RECT, PEDrawRoundRects
DefGSElement FillRoundRectTo, GR_FILL_ROUND_RECT_TO, PEDrawRoundRects
DefGSElement FillArc, GR_FILL_ARC, PEDrawArcs
DefGSElement FillPolygon, GR_FILL_POLYGON, PEPolyCoord, coords, OFP_count
DefGSElement FillEllipse, GR_FILL_ELLIPSE, PETwoCoords
DefGSElement FillPath, GR_FILL_PATH, PEByteAttr
DefGSElement FillArc3Point, GR_FILL_ARC_3POINT, PEDrawArcs
DefGSElement FillArc3PointTo, GR_FILL_ARC_3POINT_TO, PEDrawArcs
DefGSElement FillBitmap, GR_FILL_BITMAP, PEBitmap, bytes, OFB_size
DefGSElement FillBitmapAtCP, GR_FILL_BITMAP_CP, PEBitmap, bytes, OFBCP_size
DefGSElement FillBitmapOpPtr, GR_FILL_BITMAP_OPTR,
PEBitmapOpPtr,,,GrFillBitmap
DefGSElement FillBitmapPtr, GR_FILL_BITMAP_PTR, PEBitmapPtr,,,GrFillBitmap
DefGSElement DrawBitmap, GR_DRAW_BITMAP, PEBitmap, bytes, ODB_size
DefGSElement DrawBitmapAtCP, GR_DRAW_BITMAP_CP, PEBitmap, bytes, ODBCP_size
DefGSElement DrawBitmapOpPtr, GR_DRAW_BITMAP_OPTR,
PEBitmapOpPtr,,,GrDrawBitmap
DefGSElement DrawBitmapPtr, GR_DRAW_BITMAP_PTR, PEBitmapPtr,,,GrDrawBitmap
DefGSElement BitmapSlice, GSE_BITMAP_SLICE,
PESlice, bytes, OBS_size, GrDrawBitmap

```

```

DefGSElement      ,      GR_OUTPUT_55,      PENull
DefGSElement      ,      GR_OUTPUT_56,      PENull
DefGSElement      ,      GR_OUTPUT_57,      PENull
DefGSElement      ,      GR_OUTPUT_58,      PENull
DefGSElement      ,      GR_OUTPUT_59,      PENull
DefGSElement      ,      GR_OUTPUT_5A,      PENull
DefGSElement      ,      GR_OUTPUT_5B,      PENull
DefGSElement      ,      GR_OUTPUT_5C,      PENull
DefGSElement      ,      GR_OUTPUT_5D,      PENull
DefGSElement      ,      GR_OUTPUT_5E,      PENull
DefGSElement      ,      GR_OUTPUT_5F,      PENull
DefGSElement SaveState, GR_SAVE_STATE, PENoArgs
DefGSElement RestoreState, GR_RESTORE_STATE, PENoArgs
DefGSElement SetMixMode, GR_SET_MIX_MODE, PEByteAttr
DefGSElement MoveTo, GR_MOVE_TO, PEOneCoord
DefGSElement RelMoveTo, GR_REL_MOVE_TO, PERelCoord
DefGSElement CreatePalette, GR_CREATE_PALETTE, PENoArgs
DefGSElement DestroyPalette, GR_DESTROY_PALETTE, PENoArgs
DefGSElement SetPaletteEntry, GR_SET_PALETTE_ENTRY, PEOneCoord
DefGSElement SetPalette, GR_SET_PALETTE, PEPalette, bytes, OSP_num
DefGSElement SetLineColor, GR_SET_LINE_COLOR, PE3ByteAttr
DefGSElement SetLineMask, GR_SET_LINE_MASK, PEByteAttr
DefGSElement SetLineColorMap, GR_SET_LINE_COLOR_MAP, PEByteAttr
DefGSElement SetLineWidth, GR_SET_LINE_WIDTH, PEOneCoord
DefGSElement SetLineJoin, GR_SET_LINE_JOIN, PEByteAttr
DefGSElement SetLineEnd, GR_SET_LINE_END, PEByteAttr
DefGSElement SetLineAttr, GR_SET_LINE_ATTR, PEAttr
DefGSElement SetMiterLimit, GR_SET_MITER_LIMIT, PEOneCoord
DefGSElement SetLineStyle, GR_SET_LINE_STYLE, PELineStyle
DefGSElement
SetLineColorIndex, GR_SET_LINE_COLOR_INDEX, PEByteAttr, , , GrSetLineColor
DefGSElement
SetCustomLineMask, GR_SET_CUSTOM_LINE_MASK, PECustomMask, , , GrSetLineMask
DefGSElement
SetCustomLineStyle, GR_SET_CUSTOM_LINE_STYLE, PECustomStyle, words, OSCLS_count,
GrSetLineStyle
DefGSElement SetAreaColor, GR_SET_AREA_COLOR, PE3ByteAttr
DefGSElement SetAreaMask, GR_SET_AREA_MASK, PEByteAttr
DefGSElement SetAreaColorMap, GR_SET_AREA_COLOR_MAP, PEByteAttr
DefGSElement SetAreaAttr, GR_SET_AREA_ATTR, PEAttr
DefGSElement SetAreaColorIndex, GR_SET_AREA_COLOR_INDEX,
PEByteAttr, , , GrSetAreaColor
DefGSElement
SetCustomAreaMask, GR_SET_CUSTOM_AREA_MASK, PECustomMask, , , GrSetAreaMask
DefGSElement SetAreaPattern, GR_SET_AREA_PATTERN, PESetPattern
DefGSElement
SetCustomAreaPattern, GR_SET_CUSTOM_AREA_PATTERN, PESetCustPattern, bytes, OSCAP
_size, GrSetAreaPattern
DefGSElement SetTextColor, GR_SET_TEXT_COLOR, PE3ByteAttr
DefGSElement SetTextMask, GR_SET_TEXT_MASK, PEByteAttr

```

Revision: ■

Draft Dated (4/18/94)

Reference book



```

DefGSElement SetTextColorMap, GR_SET_TEXT_COLOR_MAP, PByteAttr
DefGSElement SetTextStyle, GR_SET_TEXT_STYLE, PWordAttr
DefGSElement SetTextMode, GR_SET_TEXT_MODE, PWordAttr
DefGSElement SetTextSpacePad, GR_SET_TEXT_SPACE_PAD, PSpacePad
DefGSElement SetTextAttr, GR_SET_TEXT_ATTR, PAttr
DefGSElement SetFont, GR_SET_FONT, PSetFont
DefGSElement
SetTextColorIndex, GR_SET_TEXT_COLOR_INDEX, PByteAttr, , , GrSetTextColor
DefGSElement
SetCustomTextMask, GR_SET_CUSTOM_TEXT_MASK, PCustomMask, , , GrSetTextMask
DefGSElement SetTrackKern, GR_SET_TRACK_KERN, PWordAttr
DefGSElement SetFontWeight, GR_SET_FONT_WEIGHT, PByteAttr
DefGSElement SetFontWidth, GR_SET_FONT_WIDTH, PByteAttr
DefGSElement SetSuperscriptAttr, GR_SET_SUPERSCRIPT_ATTR, PWordAttr
DefGSElement SetSubscriptAttr, GR_SET_SUBSCRIPT_ATTR, PWordAttr
DefGSElement SetTextPattern, GR_SET_TEXT_PATTERN, PSetPattern
DefGSElement
SetCustomTextPattern, GR_SET_CUSTOM_TEXT_PATTERN, PSetCustPattern, bytes, OSCTP
_size, GrSetTextPattern
DefGSElement      ,      GR_ATTR_8E,      PNull
DefGSElement      ,      GR_ATTR_8F,      PNull

DefGSElement      ,      GR_ATTR_90,      PNull
DefGSElement      ,      GR_ATTR_91,      PNull
DefGSElement      ,      GR_ATTR_92,      PNull
DefGSElement      ,      GR_ATTR_93,      PNull
DefGSElement      ,      GR_ATTR_94,      PNull
DefGSElement      ,      GR_ATTR_95,      PNull
DefGSElement      ,      GR_ATTR_96,      PNull
DefGSElement      ,      GR_ATTR_97,      PNull
DefGSElement      ,      GR_ATTR_98,      PNull
DefGSElement      ,      GR_ATTR_99,      PNull
DefGSElement      ,      GR_ATTR_9A,      PNull
DefGSElement      ,      GR_ATTR_9B,      PNull
DefGSElement      ,      GR_ATTR_9C,      PNull
DefGSElement      ,      GR_ATTR_9D,      PNull
DefGSElement      ,      GR_ATTR_9E,      PNull
DefGSElement      ,      GR_ATTR_9F,      PNull
DefGSElement BeginPath, GR_BEGIN_PATH, POneCoord
DefGSElement EndPath, GR_END_PATH, PNoArgs
DefGSElement SetClipRect, GR_SET_CLIP_RECT, PClipRect
DefGSElement SetWinClipRect, GR_SET_WIN_CLIP_RECT, PClipRect
DefGSElement CloseSubPath, GR_CLOSE_SUB_PATH, PNoArgs
DefGSElement SetClipPath, GR_SET_CLIP_PATH, PPathArgs
DefGSElement SetWinClipPath, GR_SET_WIN_CLIP_PATH, PPathArgs
DefGSElement SetStrokePath, GR_SET_STROKE_PATH, PNoArgs
DefGSElement      ,      GR_PATH_A8,      PNull
DefGSElement      ,      GR_PATH_A9,      PNull
DefGSElement      ,      GR_PATH_AA,      PNull
DefGSElement      ,      GR_PATH_AB,      PNull

```

```

DefGSElement      ,      GR_PATH_AC,      PENull
DefGSElement      ,      GR_PATH_AD,      PENull
DefGSElement      ,      GR_PATH_AE,      PENull
DefGSElement      ,      GR_PATH_AF,      PENull

```

Library: **gstring.def**

■ GStringErrorType

```

GStringErrorType    etype word
GSET_NO_ERROR       enum GStringErrorType ; there was no error
GSET_DISK_FULL      enum GStringErrorType ; disk became full, file truncated

```

Library: **gstring.def**

■ GStringKillType

```

GStringKillType     etype byte
GSKT_KILL_DATA      enum GStringKillType  ; delete the data too
GSKT_LEAVE_DATA     enum GStringKillType  ; leave the data alone

```

Library: **gstring.def**

■ GStringSetPosType

```

GStringSetPosType   etype byte
GSSPT_SKIP_1        enum GStringSetPosType ; advance 1 element
GSSPT_RELATIVE       enum GStringSetPosType ; advance N elements
GSSPT_BEGINNING      enum GStringSetPosType ; set to start of gstring
GSSPT_END            enum GStringSetPosType ; set to end of gstring

```

Library: **gstring.def**

■ GStringType

```

GStringType         etype byte
GST_CHUNK            enum GStringType      ; write to a memory chunk
GST_STREAM           enum GStringType      ; write to a stream
GST_VMEM             enum GStringType      ; write to a vmem block
GST_PTR              enum GStringType      ; static memory (read only)
GST_PATH             enum GStringType      ; write to a path (&store in an lmem
; chunk). INTERNAL ONLY!

```

Library: **gstring.def**

Revision: ■
Draft Dated (4/18/94)

Reference book



■ **GTCFeatures**

```
GTCFeatures      record
    GTCF_TOOL_DIALOG:1
GTCFeatures      end
```

Library: **Objects/gToolCC.def**

■ **GTP_vars**

```
GTP_vars      struct
    GTPPL_style      TextMetricStyles      ; Must be first
    GTPPL_object      dword      ; Object segment address
    GTPPL_startPosition      word      ; Passed position into gstring
    GTPPL_charCount      word      ; Start offset in string
    align      word
GTP_vars      ends
```

This structure is passed to **GrTextPosition**.

Library: **text.def**

■ **Guide**

```
Guide      struct
    Guide_location      DWordFixed
Guide      ends
```

Library: **ruler.def**

■ **GVCFeatures**

```
GVCFeatures      record
    GVCF_MAIN_100      :1
    GVCF_MAIN_SCALE_TO_FIT      :1
    GVCF_ZOOM_IN      :1
    GVCF_ZOOM_OUT      :1
    GVCF_REDUCE      :1
    GVCF_100      :1
    GVCF_ENLARGE      :1
    GVCF_BIG_ENLARGE      :1
    GVCF_SCALE_TO_FIT      :1
    GVCF_ADJUST_ASPECT_RATIO      :1
    GVCF_APPLY_TO_ALL      :1
```

```

GVCF_SHOW_HORIZONTAL      :1
GVCF_SHOW_VERTICAL        :1
GVCF_CUSTOM_SCALE         :1
GVCF_REDRAW                :1
GVCFfeatures               end

```

Library: **Objects/gViewCC.def**

■ **GVCToolboxFeatures**

```

GVCToolboxFeatures record
  GVCTF_100                :1
  GVCTF_SCALE_TO_FIT       :1
  GVCTF_ZOOM_IN            :1
  GVCTF_ZOOM_OUT           :1
  GVCTF_REDRAW             :1
  GVCTF_PAGE_LEFT          :1
  GVCTF_PAGE_RIGHT         :1
  GVCTF_PAGE_UP            :1
  GVCTF_PAGE_DOWN         :1
  GVCTF_ADJUST_ASPECT_RATIO :1
  GVCTF_APPLY_TO_ALL       :1
  GVCTF_SHOW_HORIZONTAL    :1
  GVCTF_SHOW_VERTICAL      :1
GVCToolboxFeatures end

```

Library: **Objects/gViewCC.def**

Revision: ■

Draft Dated (4/18/94)

Reference book



■ **Reference book**

Revision: ■
Draft Dated (4/18/94)

■ HandleUpdateMode

```
HandleUpdateMode    etype byte, 0
    HUM_NOW          enum HandleUpdateMode
    HUM_MANUAL        enum HandleUpdateMode
```

Library: **grobj.def**

■ HatchDash

```
HatchDash          struct
    HD_on           WWFixed      ; length of dash to be drawn
    HD_off          WWFixed      ; space to skip until next dash
HatchDash          ends
```

A **HatchPattern** consists of one or more **HatchLine** structures, which in turn may contain zero or more **HatchDash** structures.

Library: **graphics.def**

■ HatchLine

```
HatchLine          struct
    HL_origin       PointWWFixed ; origin of line
    HL_deltaX       WWFixed      ; X offset to next line
    HL_deltaY       WWFixed      ; Y offset to next line
    HL_angle        WWFixed      ; angle at which line is to be drawn
    HL_color        ColorQuad    ; color of line
    HL_numDashes    word         ; number of dash pairs
    HL_dashData     label HatchDash ; array of pairs of on/off lengths
HatchLine          ends
```

Library: **graphics.def**

■ HatchPattern

```
HatchPattern       struct
    HP_numLines     word         ; number of line records in this pattern
    HP_lineData     label HatchLine ; array of 1 or more hatch lines
HatchPattern       ends
```

A **HatchPattern** consists of one or more **HatchLine** structures, which in turn may contain zero or more **HatchDash** structures.

Library: **graphics.def**



■ HCFeatures

```
HCFeatures      record
    HCF_LIST      :1
HCFeatures      end
```

Library: **Objects/Text/tCtrlC.def**

■ HCToolboxFeatures

```
HCToolboxFeatures  record
    HCTF_TOGGLE    :1
HCToolboxFeatures  end
```

Library: **Objects/Text/tCtrlC.def**

■ HeapAllocFlags

```
HeapAllocFlags      record
    HAF_ZERO_INIT      :1      ; Initialize new memory to 0
    HAF_LOCK            :1      ; Return with block locked
    HAF_NO_ERR          :1      ; Caller can't handle errors
    HAF_UI              :1      ; If HAF_OBJECT_RESOURCE, set HM_otherInfo
                                ; to the handle of the UI as that's who's
                                ; to operate objects in the block
    HAF_READ_ONLY       :1      ; Data in block will not/may not be
                                ; modified
    HAF_OBJECT_RESOURCE :1      ; Block contains objects
    HAF_CODE            :1      ; Block contains executable code
    HAF_CONFORMING      :1      ; Block contains code that may be executed
                                ; by a less privileged entity
HeapAllocFlags      end
```

Library: **heap.def**

■ HeapCongestion

```
HeapCongestion      etype word, 0, 2
    HC_SCRUBBING      enum HeapCongestion ;Heap is being scrubbed. Might be a
                                ;good idea to free some things.
                                ;** Not currently used **
    HC_CONGESTED      enum HeapCongestion ; Couldn't nuke enough memory to
                                ;satisfy the heap scrubber.
    HC_DESPERATE      enum HeapCongestion ;Heap is perilously close to
                                ;overflowing. Nuke stuff *now*.
```

Library: **heap.def**

■ HeapFlags

```
HeapFlags      record
    HF_FIXED    :1      ; Block won't ever move
    HF_SHARABLE :1      ; May be locked by other than owner
    HF_DISCARDABLE :1    ; May be discarded if space needed
    HF_SWAPABLE :1      ; May be swapped if space needed
    HF_LMEM     :1      ; Managed by LMem module
    HF_DEBUG    :1      ; Swat cares what happens to it -- DO NOT PASS
                        ; THIS FLAG. IT IS RESERVED FOR INTERNAL USE
                        ; BY THE DEBUGGER
    HF_DISCARDED :1      ; Discarded and must be brought in fresh from
                        ; executable/resource file
    HF_SWAPPED  :1      ; Swapped to memory or disk.
HeapFlags      end
```

Library: **heap.def**

■ HeightJustification

```
HeightJustification  etype byte
    HJ_TOP_JUSTIFY_CHILDREN      enum HeightJustification
    HJ_BOTTOM_JUSTIFY_CHILDREN   enum HeightJustification
    HJ_CENTER_CHILDREN_VERTICALLY enum HeightJustification
    HJ_FULL_JUSTIFY_CHILDREN_VERTICALLY enum HeightJustification
```

Library: **Objects/vCompC.def**

■ HelpEntry

```
HelpEntry      struct
    HE_string   byte
HelpEntry      ends
```

This structure defines a help chunk.

Library: **Objects/genC.def**

■ HierarchicalGrab

```
HierarchicalGrab  struct
    HG_OD          optr
    HG_flags       HierarchicalGrabFlags <>
HierarchicalGrab  ends
```



Library: **Objects/uiInputC.def**

■ HierarchicalGrabFlags

```
HierarchicalGrabFlags    record
    HGF_SYS_EXCL         :1
    HGF_APP_EXCL         :1
    HGF_GRAB              :1
    HGF_OTHER_INFO       :12
HierarchicalGrabFlags    end
```

HGF_SYS_EXCL

Not passed anywhere, but stored in hierarchical grab structure, it indicates that the object has the exclusive within the System.

HGF_APP_EXCL

Not passed anywhere, but stored in hierarchical grab structure, it indicates that the object has the exclusive within the Application.

HGF_GRAB

This bit as passed to **FlowAlterHierarchicalGrab** indicates whether the object wishes to grab or release the exclusive it has within the node. Stored in a grab, it indicates that an object has the exclusive with the node (i.e. is redundant with the fact that there is an OD stored in the grab).

HGF_OTHER_INFO

Use defined by the type of **HierarchicalGrab**. This data is stored in the *HG_flags* field, whenever **FlowAlterHierarchicalGrab** is called to grab the exclusive for an object.

Library: **uiInputC.def**

■ HoldUpInputFlags

```
HoldUpInputFlags    record
    HUIF_FLUSHING_QUEUE :1
    HUIF_HOLD_UP_MODE_DISABLED :1
                                :6
HoldUpInputFlags    end
```

HUIF_FLUSHING_QUEUE

Set if the HoldUpInputQueue is in the process of being flushed. Used to allow reentrant calls into **FlowFlushHoldUpInputQueue**.

HUIF_HOLD_UP_MODE_DISABLED

Set on call to **FlowDisableHoldUpInput**. Forces input data to flow normally until cleared. Used only by the system object when a system-modal dialog box is put on screen, to ensure that user can interact with it.

Library: **uiInputC.def**

■ HugeArrayDirectory

```
HugeArrayDirectory      struct
  HAD_header      LMemBlockHeader <>;
  HAD_data         word           ; VM block link to first data block
  HAD_dir          lptr.ChunkArrayHeader ; chunk handle to ChunkArray
  HAD_xdir         word           ; link to next dir block
  HAD_self         word           ; vm block handle of self
  HAD_size         word           ; element size, 0=variable
HugeArrayDirectory      ends
```

This structure is allocated at the beginning of the directory block.

Library: **hugearr.def**

■ HWRBoxData

```
HWRBoxData      struct
  HWRBD_mode      HWRMode
  HWRBD_top       sword
  HWRBD_bottom    sword
HWRBoxData      ends
```

Library: **hwr.def**

■ HWRContext

```
HWRContext      union
  HWRC_none      HWRNoneData
  HWRC_lined     HWRLineData
  HWRC_boxed     HWRBoxData
  HWRC_grid      HWRGridData
HWRContext      end
```

Library: **hwr.def**



■ HWRGridData

```
HWRGridData      struct
    HWRGD_mode    HWRMode
    HWRGD_bounds  Rectangle
    HWRGD_xOffset sword
    HWRGD_yOffset sword
HWRGridData      ends
```

HWRGD_bounds stores the bounds of the grid area (in same coordinates as the ink data).

HWRGD_yOffset stores the X/Y offsets between grid lines.

Library: **hwr.def**

■ HWRLineData

```
HWRLineData      struct
    HWRLD_mode    HWRMode
    HWRLD_line    sword
HWRLineData      ends
```

Library: **hwr.def**

■ HWRMode

```
HWRMode  etype word
    HM_NONE      enum HWRMode
    ; The user is writing in a multi-line object - no guidelines
    HM_LINE      enum HWRMode
    ; The user has a reference line to write on
    HM_BOX       enum HWRMode
    ; The user has a box to write into
    HM_GRID      enum HWRMode
    ; The user has a grid to write chars into (one char per box)
```

Library: **hwr.def**

■ HWRNoneData

```
HWRNoneData      struct
    HWRND_mode    HWRMode
HWRNoneData      ends
```

Library: **hwr.def**

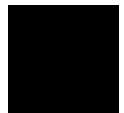
■ HWRRoutine

```

HWRRoutine          etype word
    HWRR_BEGIN_INTERACTION      enum HWRRoutine
    HWRR_END_INTERACTION        enum HWRRoutine
;
;      Most HWR drivers can not handle multiple clients at once. Clients
;      should call HWRR_BEGIN_INTERACTION before any other HWR calls, and
;      HWRR_END_INTERACTION after their HWR calls.
;
;      NOTE: Assume that after you call HWRR_END_INTERACTION, all of the
;      parameters you've set up (points added, filters activated) will
;      be destroyed.
;
;      Pass:          nothing
;      Returns:       HWRR_BEGIN_INTERACTION returns AX=0 to show that
;                     everything is fine, else there is an error.
;                     If HWRR_BEGIN_INTERACTION returns an error, do
;                     not call HWRR_END_INTERACTION.
;
HWRR_RESET           enum HWRRoutine
;
;      Resets the library in preparation of sending a new set of ink data
;      to it. This nukes all old points, and re-enables the entire
;      character set.
;
;      Pass:          nothing
;      Returns:       nothing
;      Destroyed:     nothing
;

HWRR_DISABLE_CHAR_RANGE      enum HWRRoutine
;
;      Disables the passed range of characters - this means that strokes
;      will not be recognized as these characters.
;
;      Pass: (on stack)
;
;      word - first char in range to disable
;      word - last char in range to disable
;
;      Return: nothing
;
HWRR_ENABLE_CHAR_RANGE       enum HWRRoutine
;
;      Enables the passed range of characters - this means that strokes
;      can be recognized by these characters.
;
;      Pass: (on stack)
;

```



```

;           word - first char in range to disable
;           word - last char in range to disable
;
;       Return: nothing
;
HWRR_SET_CHAR_FILTER_CALLBACK    enum HWRRoutine
;
;       Calls the passed callback routine with characters.
;
;       Pass:   (push in this order)
;               fptr to callback routine
;               fptr to callback data
;
;       Return: nothing
;
;       Callback is passed (on stack):
;
;           word          number of choices for character
;           word          offset of first point in char
;           word          offset of last point in char
;           fptr          array of 16-bit characters
;           fptr          callback data
;
;       Callback should return:
;
;           AX =          character chosen (it does not necessarily have to
;                           be one of the characters in the passed array)
;
;       Callback can destroy: ax, bx, cx, dx
;
HWRR_SET_STRING_FILTER_CALLBACK  enum HWRRoutine
;
;       This allows the application to specify his own filter routine on
;       an entire word basis (as opposed to a char by char basis)
;
;       NOTE: If the app specifies a "WHOLE_WORD" filter callback, it should
;       not also specify a "CHAR_FILTER" callback, as the "CHAR_FILTER"
;       callback will not be called.
;
;       Pass:   (on stack - push in this order)
;               fptr to callback routine
;               fptr to callback data
;
;       Returns:nothing
;
;       Callback routine is passed (on stack - pascal model):
;
;           word          number of characters recognized
;           fptr          array of CharChoiceInformation structures
;           fptr          callback data

```

Reference book

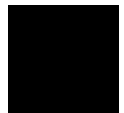
```

;
;   Callback routine returns:
;
;       AX - handle of block containing null-terminated
;           ink data
;
;   Callback routine can destroy:
;
;       AX, BX, CX, DX
HWRR_ADD_POINT          enum HWRRoutine
;
;   This allows the application to add a point to the list of
;   points being collected and recognized
;
;   Pass: (on stack)
;           InkXCoord          ;X coordinate
;           word               ;Y coordinate
;           dword              ;time stamp
;                               (normally passed as 0, but can be passed
;                               as an actual value for certain real-time
;                               applications, such as signature verification)
;   Returns: nothing
;
HWRR_ADD_POINTS          enum HWRRoutine
;
;   This adds a bunch of points at once.
;
;   Pass: (on stack)
;           word               num points
;           fptr               array of InkPoint structures
;   Return: nothing
;
HWRR_DO_GESTURE_RECOGNITION  enum HWRRoutine
;
;   Checks to see if the points are a single gesture.
;
;   Pass: nothing
;   Return: AX <- return
;

HWRR_DO_SINGLE_CHAR_RECOGNITION enum HWRRoutine
;
;   This returns a single char that was recognized from the ink input
;
;   Pass: nothing
;   Return: AX <- character
;

HWRR_DO_MULTIPLE_CHAR_RECOGNITION  enum HWRRoutine
;

```



```

;      This returns a null-terminated string that was recognized from the
;      input.
;
;      Pass: nothing
;      Return: AX <- handle of block containing null-terminated ink data
;
HWRR_GET_HWR_MANUF_ID          enum HWRRoutine
;
;      Returns the manufacturer of the HWR. This may be useful if you want
;      to call certain special features that only exist in certain drivers.
;      (For example, if one was writing a signature verification
application
;      that required a specific HWR driver).
;
;      Pass: nothing
;      Returns: AX - ManufacturerID
;

HWRR_SET_CONTEXT              enum HWRRoutine
;
;      Puts the hwr engine in line/grid/boxed mode.
;
;      Pass (on stack)
;      fptr to HWRContext union
HWRR_SET_LANGUAGE             enum HWRRoutine
;
;      Sets the default recognition language of the HWR
;      Pass:                (push in this order)
;                          word - StandardLanguage
;      Returns:             nothing

```

Library: **hwr.def**

■ HyphenationPoints

```

HyphenationPoints  struct
    HP_wordLen      word
    HP_array         label byte
HyphenationPoints  ends

```

HP_array marks the start of a null-terminated array of hyphenation points.

Library: **Objects/vTextC.def**

■ HyphenFlags

```
HyphenFlags      record
  HF_AUTO_HYPHEN  :1      ; set when auto-hyphen exists at EOL
HyphenFlags      end
```

Library: **text.def**

■ IACPConnectError

```
IACPConnectError  etype word, GeodeLoadError
  IACPCE_CANNOT_FIND_SERVER  enum IACPConnectError
  IACPCE_NO_SERVER           enum IACPConnectError
```

IACPCE_CANNOT_FIND_SERVER

Asked to start server w/o specifying location of app, and IACP was unable to find an application with the same token as the list.

IACPCE_NO_SERVER

Didn't ask IACP to start server, and no server is registered for the list.

Library: **iaccp.def**

■ IACPConnectFlags

```
IACPConnectFlags  record
                                     :10
  IACPCF_OBEY_LAUNCH_MODEL          :1
  IACPCF_CLIENT_OD_SPECIFIED        :1
  IACPCF_FIRST_ONLY                 :1
  IACPCF_SERVER_MODE IACPServerMode :3
IACPConnectFlags  end
```

IACPCF_OBEY_LAUNCH_MODEL

Set if IACP should obey any launch model for the field that would cause it to ask the user whether an existing server should be used, or a new one started. **AppLaunchBlock** must be passed with *ALB_appMode* set to MSG_GEN_PROCESS_OPEN_APPLICATION and IACPCF_SERVER_MODE set to IACPSM_USER_INTERACTIBLE.

IACPCF_CLIENT_OD_SPECIFIED

Set to indicate ^lcx:dx contains client OD for the IACP connection. If not set, the client OD is set to the application object of the process on whose thread the **IACPConnect** call is made.



IACPCF_FIRST_ONLY

Set to connect only to the first server on the list, else connects to all of them.

IACPCF_SERVER_MODE

Mode in which server is expected to be operating (IACPSM_IN_FLUX not allowed). Higher-numbered modes are expected to support requests for lower-numbered modes.

Library: **iACP.def**

■ IACPDocCloseAckParams

```
IACPDocCloseAckParams    struct
  IDCAP_docObj            optr
  IDCAP_connection        IACPConnection
  IDCAP_serverNum         word
  IDCAP_status            word           ; DocQuitStatus
IACPDocCloseAckParams    ends
```

Library: **iACP.def**

■ IACPDocOpenAckParams

```
IACPDocOpenAckParams    struct
  IDOAP_docObj            optr
  IDOAP_connection        IACPConnection
  IDOAP_serverNum         word
IACPDocOpenAckParams    ends
```

IDOAP_docObj the optr of the document object managing the document.

IDOAP_connection stores the IACP connection over which the request to open was received.

IDOAP_serverNum stores the server number the document object's GenApplication object is for that connection (0 if connection goes through some other object).

Library: **iACP.def**

■ IACPServerFlags

```
IACPServerFlags          record
  IACPSF_MULTIPLE_INSTANCES:1
  ; Set if application may have multiple instances of itself launched
  :7
IACPServerFlags          end
```

Library: **iarp.def**

■ IACPServerMode

```
IACPServerMode etype byte
IACPSM_NOT_USER_INTERACTIBLE      enum IACPServerMode
IACPSM_IN_FLUX                     enum IACPServerMode
IACPSM_USER_INTERACTIBLE           enum IACPServerMode
```

The mode in which an IACP server is operating. A user-interactable server is expected to cope with messages addressed to a non-user-interactable server, but the reverse is not true.

Library: **iarp.def**

■ IACPSide

```
IACPSide etype word
IACPS_CLIENT      enum IACPSide
IACPS_SERVER      enum IACPSide
```

Specifies which side of an IACP connection is sending a message via **IACPSendMessage**.

Library: **iarp.def**

■ IEEE64

```
IEEE64 struct
IEEE64_wd0 word
IEEE64_wd1 word
IEEE64_wd2 word
IEEE64_wd3 word
IEEE64 ends
```

Library: **math.def**

■ ImageBitSize

```
ImageBitSize etype byte
IBS_1         enum ImageBitSize ; 1 to 1 mapping
IBS_2         enum ImageBitSize ; 2 x 2 pixels
IBS_4         enum ImageBitSize ; 4 x 4 pixels
IBS_8         enum ImageBitSize ; 8 x 8 pixels
IBS_16        enum ImageBitSize ; 16 x 16 pixels
```

Library: **graphics.def**



■ ImageFlags

```
ImageFlags      record
  IF_DRAW_IMAGE  :1,      ; reserved for internal use (set to zero)
  IF_HUGE        :1,      ; reserved for internal use (set to zero)
                  :1,
  IF_IGNORE_MASK :1       ; set to draw all pixels, regardless of mask
  IF_BORDER      :1,      ; set if border desired around each pixel
  IF_BITSIZES    ImageBitSize:3; size of each pixel
ImageFlags      end
```

Library: **graphics.def**

■ IMCFeatures

```
IMCFeatures      record
                  :7
  IMCF_MAP        :1
IMCFeatures      end
```

Library: **impex.def**

■ ImpexDataClasses

```
ImpexDataClasses record
  IDC_TEXT        :1
  IDC_GRAPHICS    :1
  IDC_SPREADSHEET :1
  IDC_FONT        :1
                  :12
ImpexDataClasses end
```

Library: **impex.def**

■ ImpexFileSelectionData

```
ImpexFileSelectionData struct
  IFSD_selection FileLongName
  IFSD_path       PathName
  IFSD_disk       word
  IFSD_type       GenFileSelectorEntryFlags
ImpexFileSelectionData ends
```

This structure is passed with
MSG_IMPORT_EXPORT_FILE_SELECTION_INFO.

Library: **impex.def**

Reference book

■ ImpexMapFileInfoHeader

```
ImpexMapFileInfoHeader    struc
    IMFTH_base             LMemBlockHeader
    IMFTH_fieldChunk       word
    IMFTH_numFields        word
    IMFTH_flag             DefaultFieldNameUsage
ImpexMapFileInfoHeader    ends
```

Library: **impex.def**

■ ImpexMapFlags

```
ImpexMapFlags             record
    IMF_IMPORT             :1
    IMF_EXPORT             :1
                           :6
ImpexMapFlags             end
```

Library: **impex.def**

■ ImpexTranslationParams

```
ImpexTranslationParams    struct
    ITP_impexOD            optr   ; OD of Import/ExportControl
    ITP_returnMsg          word   ; message to return to above
    ITP_dataClass          word   ; what class type (ImpexDataClasses)
    ITP_transferVMFile     hptr   ; VM file w/ transfer format
    ITP_transferVMChain    dword  ; VM chain w/ transfer format
    ITP_internal           dword  ; two words of internal data
    ITP_manufacturerID     ManufacturerID;
    ITP_clipboardFormat     ClipboardItemFormat
ImpexTranslationParams    ends
```

Library: **ieCommon.def**

■ ImportControlAttrs

```
ImportControlAttrs       record
    ICA_IGNORE_INPUT      :1      ; ignore input while import occurs.
    ICA_NON_DOCUMENT_IMPORT :1      ; non-document imports only
                           :14
ImportControlAttrs       end
```

Library: **impex.def**



■ **ImportControlFeatures**

```

ImportControlFeatures      record
    IMPORTCF_PREVIEW_TRIGGER  :1      ; not currently used
    IMPORTCF_IMPORT_TRIGGER   :1      ; import trigger
    IMPORTCF_FORMAT_OPTIONS   :1      ; import format UI parent,
                                      ; under which is placed any
                                      ; UI specific to the
                                      ; currently selected format
    IMPORTCF_FILE_MASK        :1      ; import file mask
    IMPORTCF_BASIC             :1      ; import file selector,
                                      ; import format list, and
                                      ; import app UI parent, under
                                      ; which is placed any UI
                                      ; specific to the app
    IMPORTCF_GLYPH            :1      ; glyph at top of import
                                      ; dialog box
ImportControlFeatures      end

```

Library: **impex.def**

■ **ImportControlToolboxFeatures**

```

ImportControlToolboxFeatures  record
    IMPORTCTF_DIALOG_BOX      :1
ImportControlToolboxFeatures  end

```

Library: **impex.def**

■ **InitFileCharConvert**

```

InitFileCharConvert          etype byte
    IFCC_INTACT               enum InitFileCharConvert  ; leave intact
    IFCC_UPCASE               enum InitFileCharConvert  ; upcase all chars
    IFCC_DOWNCASE             enum InitFileCharConvert  ; downcase all chars

```

Library: **initfile.def**

■ InitFileReadFlags

```
InitFileReadFlags  record
    IFRF_CHAR_CONVERT          InitFileCharConvert:2    ; character conversion

    IFRF_READ_ALL:1
    ; Instructs the initfile routine to read from all the init files,
    ; where appropriate. Only used in InitFileEnumStringSection,
    ; which will enumerate over all string sections in all init files

    IFRF_FIRST_ONLY:1
    ; Read from the first init file only.

    IFRF_SIZE                  :12                      ; size of buffer
InitFileReadFlags  end
```

Library: **initfile.def**

■ InkBackgroundType

```
InkBackgroundType  etype word, 0, 2
    IBT_NO_BACKGROUND          enum InkBackgroundType
    IBT_NARROW_LINED_PAPER     enum InkBackgroundType
    IBT_MEDIUM_LINED_PAPER     enum InkBackgroundType
    IBT_WIDE_LINED_PAPER       enum InkBackgroundType
    IBT_NARROW_STENO_PAPER     enum InkBackgroundType
    IBT_MEDIUM_STENO_PAPER     enum InkBackgroundType
    IBT_WIDE_STENO_PAPER       enum InkBackgroundType
    IBT_SMALL_GRID             enum InkBackgroundType
    IBT_MEDIUM_GRID            enum InkBackgroundType
    IBT_LARGE_GRID             enum InkBackgroundType
    IBT_SMALL_CROSS_SECTION    enum InkBackgroundType
    IBT_MEDIUM_CROSS_SECTION   enum InkBackgroundType
    IBT_LARGE_CROSS_SECTION    enum InkBackgroundType
    IBT_TO_DO_LIST             enum InkBackgroundType
    IBT_PHONE_MESSAGE          enum InkBackgroundType
    IBT_CUSTOM_BACKGROUND      enum InkBackgroundType
```

Library: **pen.def**



■ **InkControlFeatures**

```
InkControlFeatures  record
    ICF_PENCIL_TOOL      :1
    ICF_ERASER_TOOL      :1
    ICF_SELECTION_TOOL    :1
InkControlFeatures  end
```

Library: **pen.def**

■ **InkControlToolboxFeatures**

```
InkControlToolboxFeatures  record
    ICTF_PENCIL_TOOL      :1
    ICTF_ERASER_TOOL      :1
    ICTF_SELECTION_TOOL    :1
InkControlToolboxFeatures  end
```

Library: **pen.def**

■ **InkDBFrame**

```
InkDBFrame          struct
    IDBF_bounds      Rectangle
    IDBF_VMFile       hptr
    IDBF_DBGGroupAndItem  DBGroupAndItem
    IDBF_DBExtra      word
InkDBFrame          ends
```

IDBF_bounds stores the bounds of the ink to write out. If you want all of the ink rather than a portion of it, pass <(0,0), (0xffff, 0xffff)> as the bounds.

IDBF_VMFile stores the VM File to write to or read from (depending on the operation).

IDBF_DBGGroupAndItem stores the DB Item to save to or load from (or 0 to create a new one).

IDBF_DBExtra stores the extra space to skip at the start of the block.

Library: **pen.def**

■ InkDestinationInfo

```
InkDestinationInfo struct
    IDI_destObj          optr
    IDI_gstate           hptr.GState
    IDI_brushSize        word
    IDI_gestureCallback  dword
InkDestinationInfo ends
```

IDI_destObj stores the optr of the object that the ink should be sent to.

IDI_gstate stores the gstate to draw through. (This is optional, and can be set to 0 if ink can go all over the screen).

IDI_brushSize stores the width/height parameter of the ink lines (see GrBrushPolyline). Use 0 for default behavior.

IDI_gestureCallback stores the virtual far pointer to the callback routine.

Callback Routine Specifications:

Pass on stack: (Pascal calling convention):

fptr	arrayOfInkPoints
word	numPoints
word	numStrokes

numStrokes specifies the number of strokes entered by the user. If you only support single-stroke gestures, you can check this to quickly exit if the user has entered multiple strokes.

Return: **ax** Non-zero if the ink is a gesture

Library: **Objects/uiInputC.def**

■ InkDestinationInfoParams

```
InkDestinationInfoParams struct
    IDIP_dest          optr
    IDIP_brushSize      word
    IDIP_color          Color
    IDIP_reserved1      byte
    IDIP_createGState   byte
    IDIP_reserved2      byte
InkDestinationInfoParams ends
```

IDIP_dest and *IDIP_brushSize* are the same as the arguments passed to **UserCreateInkDestinationInfo**.

Library: **Objects/gViewC.def**



■ InkFlags

```
InkFlags record
    IF_HAS_MOUSE_GRAB          :1
    IF_SELECTING               :1
    IF_HAS_TARGET              :1
    IF_HAS_SYS_TARGET          :1
    IF_DIRTY                   :1
    IF_ONLY_CHILD_OF_CONTENT   :1
    IF_CONTROLLED              :1
    IF_INVALIDATE_ERASURES     :1
    IF_HAS_UNDO                :1
                                :7
```

InkFlags end

IF_HAS_MOUSE_GRAB
Set if the object has grabbed the mouse.

IF_SELECTING
Set if doing a selection.

IF_HAS_TARGET
Set if this object has the target.

IF_HAS_SYS_TARGET
Set if this object has the target.

IF_DIRTY
Set when we are dirty.

IF_ONLY_CHILD_OF_CONTENT
Set if this is the only child of a VisContent, in which case it should use some optimizations to reply to ink at the view level.

IF_CONTROLLED
Set if this object is to be used in conjunction with an InkControl object.

IF_INVALIDATE_ERASURES
Set if we should invalidate the bounds of all erasures, in case there is a subclass that needs to redraw the background.

IF_HAS_UNDO
Set if this object should be undoable.

Library: **pen.def**

■ InkGrab

```
InkGrab struct
    IG_OD      optr
    IG_gState   hptr
InkGrab ends
```

This structure is used by the ink code within the Flow object.

Library: **Objects/uiInputC.def**

■ InkHeader

```
InkHeader struct
    IH_count    word
    IH_bounds    Rectangle
    IH_destination optr
    IH_reserved  dword
    IH_data      label Point
InkHeader ends
```

This structure defines the data block format for the GWNT_INK notification type.

IH_count stores the number of ink points collected.

IH_bounds stores the bounds of the ink on the screen.

IH_destination stores the optr of the destination object for the ink. Objects can use this to determine whether the ink was sent to them directly, or just because it overlapped the screen. This field is set by the flow object.

IH_reserved is reserved for future use.

Library: **Input.def**

■ InkPoint

```
InkPoint struct
    IP_x    InkXCoord
    IP_y    word
InkPoint ends
```

Library: **hwr.def**



■ InkReturnValue

```
InkReturnValue      etype word
    IRV_NO_REPLY          enum InkReturnValue, 0
    IRV_NO_INK            enum InkReturnValue
    IRV_INK_WITH_STANDARD_OVERRIDE enum InkReturnValue
    IRV_DESIRES_INK       enum InkReturnValue
    IRV_WAIT              enum InkReturnValue
```

IRV_NO_REPLY

VisComp objects use **VisCallChildUnderPoint** to send MSG_META_QUERY_IF_PRESS_IS_INK to its children, and VisCallChildUnderPoint returns this value (zero) if there was not child under the point. No object should actually return this value.

IRV_NO_INK

Return this if the object wants the MSG_META_START_SELECT to be passed on to it.

IRV_INK_WITH_STANDARD_OVERRIDE

Return this if the object normally wants ink (the text object does this), but the user can force mouse events instead by pressing the pen and holding for some user-adjustable amount of time).

IRV_DESIRES_INK

Return this if the object does not want the MSG_META_START_SELECT (it should be captured as ink).

IRV_WAIT

This should be the last item in the enumerated type for EC reasons.

Return this value if the object under the point is run by a different thread, and you want to hold up input (don't do anything with the MSG_META_START_SELECT) until an object sends a MSG_GEN_APPLICATION_INK_QUERY_REPLY to the application object.

Library: **uiInputC.def**

■ InkStrokeSize

```
InkStrokeSize      struct
    ISS_width       byte
    ISS_height      byte
InkStrokeSize      ends
```

Library: **pen.def**

■ InkTool

```
InkTool  etype word, 0, 2
  IT_PENCIL    enum InkTool ;Default tool
  IT_ERASER    enum InkTool
  IT_SELECTOR  enum InkTool
```

Library: **pen.def**

■ InkXCoord

```
InkXCoord      record
  IXC_TERMINATE_STROKE  :1
  IXC_X_COORD           :15
InkXCoord      end
```

Library: **hwr.def**

■ InsertChildFlags

```
InsertChildFlags  record
  ICF_MARK_DIRTY  :1,                ;Marks chunk and modified object as
                                      ; dirty
                                      :13,
  ICF_OPTIONS      InsertChildOption:2 ;Options for how to add the child
InsertChildFlags  end
```

Library: **Objects/metaC.def**

■ InsertChildOption

```
InsertChildOption  etype byte
  ICO_FIRST          enum InsertChildOption
  ICO_LAST           enum InsertChildOption
  ICO_BEFORE_REFERENCE  enum InsertChildOption
  ICO_AFTER_REFERENCE  enum InsertChildOption
```

Library: **Objects/metaC.def**



■ InsertDeleteSpaceParams

```
InsertDeleteSpaceParams  struct
    IDSP_position    PointDWFxed
    IDSP_space       PointDWFxed
    IDSP_type        InsertDeleteSpaceTypes
InsertDeleteSpaceParams  ends
```

Library: **Objects/visC.def**

■ InsertDeleteSpaceTypes

```
InsertDeleteSpaceTypes  record
                                :11
    IDST_MOVE_OBJECTS_INSIDE_DELETED_SPACE_BY_AMOUNT_DELETED :1
    IDST_MOVE_OBJECTS_INTERSECTING_DELETED_SPACE             :1
    IDST_RESIZE_OBJECTS_INTERSECTING_SPACE                    :1
    IDST_DELETE_OBJECTS_SHRUNK_TO_ZERO_SIZE                   :1
    IDST_MOVE_OBJECTS_BELOW_AND_RIGHT_OF_INSERT_POINT_OR_DELETED_SPACE :1
InsertDeleteSpaceTypes  end
```

IDST_MOVE_OBJECTS_INSIDE_DELETED_SPACE_BY_AMOUNT_DELETED
Move objects that are in the deleted space by the amount of space being deleted.

IDST_MOVE_OBJECTS_INTERSECTING_DELETED_SPACE
Move objects that intersect the deleted space so that their left and top are aligned with the left and top of the deleted space

IDST_RESIZE_OBJECTS_INTERSECTING_SPACE
If inserting space and line extending down and/or to right from insert point intersects object then add inserted space to size of object. If deleting space and deleted space intersects object then remove space from object. Object can be shrunk to zero width and height.

IDST_DELETE_OBJECTS_SHRUNK_TO_ZERO_SIZE
If object is shrunk to zero width OR height during delete space then delete it.

IDST_MOVE_OBJECTS_BELOW_AND_RIGHT_OF_INSERT_POINT_OR_DELETED_SPACE
If inserting space and object is below and or to right of insert point then move object down and right the amount of inserted space. If deleting space and object is below or to right of deleted space then move object up and to left the amount of the deleted space. In most uses of this message, this bit will be set.

Library: **Objects/visC.def**

Reference book



■ InstrumentPatch

```
InstrumentPatch      etype dword, 0, size InstrumentEnvelope
; MIDI patch 1- 8 = Piano
IP_ACOUSTIC_GRAND_PIANO          enum InstrumentPatch
IP_BRIGHT_ACOUSTIC_PIANO        enum InstrumentPatch
IP_ELECTRIC_GRAND_PIANO          enum InstrumentPatch
IP_HONKY_TONK_PIANO              enum InstrumentPatch
IP_ELECTRIC_PIANO_1              enum InstrumentPatch
IP_ELECTRIC_PIANO_2              enum InstrumentPatch
IP_HARPSICORD                    enum InstrumentPatch
IP_CLAVICORD                     enum InstrumentPatch
; MIDI patch 9- 16 = Chromatic Percussion
IP_CELESTA                       enum InstrumentPatch
IP_GLOCKENSPIEL                  enum InstrumentPatch
IP_MUSIC_BOX                     enum InstrumentPatch
IP_VIBRAPHONE                    enum InstrumentPatch
IP_MARIMBA                       enum InstrumentPatch
IP_XYLOPHONE                     enum InstrumentPatch
IP_TUBULAR_BELLS                 enum InstrumentPatch
IP_DULCIMER                      enum InstrumentPatch
; MIDI patch 17- 24 = Organ
IP_DRAWBAR_ORGAN                 enum InstrumentPatch
IP_PERCUSSIVE_ORGAN              enum InstrumentPatch
IP_ROCK_ORGAN                    enum InstrumentPatch
IP_CHURCH_ORGAN                  enum InstrumentPatch
IP_REED_ORGAN                    enum InstrumentPatch
IP_ACCORDIAN                     enum InstrumentPatch
IP_HARMONICA                     enum InstrumentPatch
IP_TANGO_ACCORDION               enum InstrumentPatch
; MIDI patch 25- 32 = Guitar
IP_ACOUSTIC_NYLON_GUITAR          enum InstrumentPatch
IP_ACOUSTIC_STEEL_GUITAR          enum InstrumentPatch
IP_ELECTRIC_JAZZ_GUITAR           enum InstrumentPatch
IP_ELECTRIC_CLEAN_GUITAR          enum InstrumentPatch
IP_ELECTRIC_MUTED_GUITAR          enum InstrumentPatch
IP_OVERDRIVEN_GUITAR             enum InstrumentPatch
IP_DISTORTION_GUITAR             enum InstrumentPatch
IP_GUITAR_HARMONICS              enum InstrumentPatch
; MIDI patch 33- 40 = Bass
IP_ACOUSTIC_BASS                  enum InstrumentPatch
IP_ELECTRIC_FINGERED_BASS         enum InstrumentPatch
IP_ELECTRIC_PICKED_BASS          enum InstrumentPatch
IP_FRETLESS_BASS                 enum InstrumentPatch
IP_SLAP_BASS_1                   enum InstrumentPatch
IP_SLAP_BASS_2                   enum InstrumentPatch
IP_SYNTH_BASS_1                  enum InstrumentPatch
IP_SYNTH_BASS_2                  enum InstrumentPatch
; MIDI patch 41- 48 = Strings
IP_VIOLIN                        enum InstrumentPatch
```

IP_VIOLA	enum InstrumentPatch
IP_CELLO	enum InstrumentPatch
IP_CONTRABASS	enum InstrumentPatch
IP_TREMELO_STRINGS	enum InstrumentPatch
IP_PIZZICATO_STRINGS	enum InstrumentPatch
IP_ORCHESTRAL_HARP	enum InstrumentPatch
IP_TIMPANI	enum InstrumentPatch
; MIDI patch 49- 56 = Ensemble	
IP_STRING_ENSEMBLE_1	enum InstrumentPatch
IP_STRING_ENSEMBLE_2	enum InstrumentPatch
IP_SYNTH_STRINGS_1	enum InstrumentPatch
IP_SYNTH_STRINGS_2	enum InstrumentPatch
IP_CHIOR_AAHS	enum InstrumentPatch
IP_VOICE_OOHS	enum InstrumentPatch
IP_SYNTH_VOICE	enum InstrumentPatch
IP_ORCHESTRA_HIT	enum InstrumentPatch
; MIDI patch 57- 64 = Brass	
IP_TRUMPET	enum InstrumentPatch
IP_TROMBONE	enum InstrumentPatch
IP_TUBA	enum InstrumentPatch
IP_MUTED_TRUMPET	enum InstrumentPatch
IP_FRENCH_HORN	enum InstrumentPatch
IP_BRASS_SECTION	enum InstrumentPatch
IP_SYNTH_BRASS_1	enum InstrumentPatch
IP_SYNTH_BRASS_2	enum InstrumentPatch
; MIDI patch 65- 72 = Reed	
IP_SOPRANO_SAX	enum InstrumentPatch
IP_ALTO_SAX	enum InstrumentPatch
IP_TENOR_SAX	enum InstrumentPatch
IP_BARITONE_SAX	enum InstrumentPatch
IP_OBOE	enum InstrumentPatch
IP_ENGLISH_HORN	enum InstrumentPatch
IP_BASSOON	enum InstrumentPatch
IP_CLARINET	enum InstrumentPatch
; MIDI patch 73- 80 = Pipe	
IP_PICCOLO	enum InstrumentPatch
IP_FLUTE	enum InstrumentPatch
IP_RECORDER	enum InstrumentPatch
IP_PAN_FLUTE	enum InstrumentPatch
IP_BLOWN_BOTTLE	enum InstrumentPatch
IP_SHAKUHACHI	enum InstrumentPatch
IP_WHISTLE	enum InstrumentPatch
IP_OCARINA	enum InstrumentPatch
; MIDI patch 81- 88 = Synth Lead	
IP_LEAD_SQUARE	enum InstrumentPatch
IP_LEAD_SAWTOOTH	enum InstrumentPatch
IP_LEAD_CALLIOPE	enum InstrumentPatch
IP_LEAD_CHIFF	enum InstrumentPatch
IP_LEAD_CHARANG	enum InstrumentPatch
IP_LEAD_VOICE	enum InstrumentPatch

IP_LEAD_FIFTHS	enum InstrumentPatch
IP_LEAD_BASS_LEAD	enum InstrumentPatch
; MIDI patch 89- 96 = Synth Pad	
IP_PAD_NEW_AGE	enum InstrumentPatch
IP_PAD_WARM	enum InstrumentPatch
IP_PAD_POLYSYNTH	enum InstrumentPatch
IP_PAD_CHOIR	enum InstrumentPatch
IP_PAD_BOWED	enum InstrumentPatch
IP_PAD_METALLIC	enum InstrumentPatch
IP_PAD_HALO	enum InstrumentPatch
IP_PAD_SWEEP	enum InstrumentPatch
; MIDI patch 97-104 = Synth Effects	
IP_FX_RAIN	enum InstrumentPatch
IP_FX_SOUNDTRACK	enum InstrumentPatch
IP_FX_CRYSTAL	enum InstrumentPatch
IP_FX_ATMOSPHERE	enum InstrumentPatch
IP_FX_BRIGHTNESS	enum InstrumentPatch
IP_FX_GOBLINS	enum InstrumentPatch
IP_FX_ECHOES	enum InstrumentPatch
IP_FX_SCI_FI	enum InstrumentPatch
; MIDI patch 105-112 = Ethnic	
IP_SITAR	enum InstrumentPatch
IP_BANJO	enum InstrumentPatch
IP_SHAMISEN	enum InstrumentPatch
IP_KOTO	enum InstrumentPatch
IP_KALIMBA	enum InstrumentPatch
IP_BAG_PIPE	enum InstrumentPatch
IP_FIDDLE	enum InstrumentPatch
IP_SHANAI	enum InstrumentPatch
; MIDI patch 113-120 = Percussive	
IP_TINKLE_BELL	enum InstrumentPatch
IP_AGOGO	enum InstrumentPatch
IP_STEEL_DRUMS	enum InstrumentPatch
IP_WOODBLOCK	enum InstrumentPatch
IP_TAIKO_DRUM	enum InstrumentPatch
IP_MELODIC_TOM	enum InstrumentPatch
IP_SYNTH_DRUM	enum InstrumentPatch
IP_REVERSE_CYMBAL	enum InstrumentPatch
; MIDI patch 132-128 = SoundEffects	
IP_GUITAR_FRET_NOISE	enum InstrumentPatch
IP_BREATH_NOISE	enum InstrumentPatch
IP_SEASHORE	enum InstrumentPatch
IP_BIRD_TWEET	enum InstrumentPatch
IP_TELEPHONE_RING	enum InstrumentPatch
IP_HELICOPTER	enum InstrumentPatch
IP_APPLAUSE	enum InstrumentPatch
IP_GUNSHOT	enum InstrumentPatch
; MIDI Percussion Map (Channel 10)	
; Keys 35-42	
IP_ACOUSTIC_BASS_DRUM	enum InstrumentPatch

[illegible]

IP_OPEN_TRIANGLE

enum InstrumentPatch

Library: **sound.def**

■ InteractionCommand

InteractionCommand etype word

IC_NULL	enum InteractionCommand
IC_DISMISS	enum InteractionCommand
IC_INTERACTION_COMPLETE	enum InteractionCommand
IC_APPLY	enum InteractionCommand
IC_RESET	enum InteractionCommand
IC_OK	enum InteractionCommand
IC_YES	enum InteractionCommand
IC_NO	enum InteractionCommand
IC_STOP	enum InteractionCommand
IC_EXIT	enum InteractionCommand
IC_HELP	enum InteractionCommand

IC_NULL

Special interaction command for use with **UserDoDialog** and **UserStandardDialog**, et. al. When returned as the dialog response, this indicates that the interaction was terminated by the system (for example, the system shut down while the box was on-screen). This should not be sent with MSG_GEN_INTERACTION_ACTIVATE_COMMAND or MSG_GEN_GUP_INTERACTION_COMMAND.

IC_DISMISS

Dismisses interaction, making it non-visible. Will always cause window to come down, even overriding the user's preference, such as having pinned the window.

IC_INTERACTION_COMPLETE

Notification to the GenInteraction that the user has completed one interaction. The specific UI must then decide whether the interaction should stay around to allow the user multiple interactions, or whether it should come down.

Motif will dismiss the interaction if it is modal, or if HINT_INTERACTION_SINGLE_USAGE is set.

OPEN LOOK dismisses unpinned interactions. This command is automatically sent as a side effect to the interaction, via MSG_GEN_GUP_INTERACTION_COMMAND, when a trigger with GA_SIGNAL_INTERACTION_COMPLETE set is activated and doesn't send another

MSG_GEN_GUP_INTERACTION_COMMAND as a result of being activated. Has no effect if the interaction has already been

dismissed. This command is special in that a button may not be created for the sole purpose of activating this command. Because of this, `IC_INTERACTION_COMPLETE` may not be used with `ATTR_GEN_TRIGGER_INTERACTION_COMMAND` or `MSG_GEN_INTERACTION_ACTIVATE_COMMAND`. It can be used with `MSG_GEN_GUP_INTERACTION_COMMAND`.

<code>IC_APPLY</code>	Standard response for <code>GIT_PROPERTIES</code> . Applies properties. Causes <code>MSG_GEN_APPLY</code> to be sent to the UI gadgets under the <code>GenInteraction</code> .
<code>IC_RESET</code>	Standard response for <code>GIT_PROPERTIES</code> . Resets properties. Causes <code>MSG_GEN_RESET</code> to be sent to the UI gadgets under the <code>GenInteraction</code> .
<code>IC_OK</code>	Standard response for <code>GIT_NOTIFICATION</code> .
<code>IC_YES</code>	Standard response for <code>GIT_AFFIRMATION</code> .
<code>IC_NO</code>	Standard response for <code>GIT_AFFIRMATION</code> .
<code>IC_STOP</code>	Standard response for <code>GIT_PROGRESS</code> .
<code>IC_EXIT</code>	Special interaction command used to indicate that this <code>GenTrigger</code> exits the application. Motif has a “Exit” item in the “File” menu. This should only be used with <code>ATTR_GEN_TRIGGER_INTERACTION_COMMAND</code> , not with <code>MSG_GEN_GUP_INTERACTION_COMMAND</code> or <code>MSG_GEN_INTERACTION_ACTIVATE_COMMAND</code> . It is only supported for <code>GenTriggers</code> under <code>GIV_POPUP</code> <code>GenInteractions</code> .
<code>IC_HELP</code>	Special interaction command used to indicate that this <code>GenTrigger</code> brings up help. This should only be used with <code>ATTR_GEN_TRIGGER_INTERACTION_COMMAND</code> .

Library: **Objects/gInterC.def**

Library: **netware.def**



■ JCFeatures

```
JCFeatures          record
    JCF_LEFT        :1
    JCF_RIGHT        :1
    JCF_CENTER       :1
    JCF_FULL         :1
JCFeatures          end
```

Library: **Objects/Text/tCtclC.def**

■ JCToolboxFeatures

```
JCToolboxFeatures   record
    JCTF_LEFT        :1
    JCTF_RIGHT        :1
    JCTF_CENTER       :1
    JCTF_FULL         :1
JCToolboxFeatures   end
```

Library: **Objects/Text/tCtrlC.def**

■ JobStatus

```
JobStatus           struct
    ; DO NOT CHANGE THE ORDER OF THESE FIRST FOUR ITEMS
    JS_fname         char 13 dup (?)
    JS_parent        char FILE_LONGNAME_LENGTH+1 dup (?)
    JS_documentName  char FILE_LONGNAME_LENGTH+1 dup (?)
    JS_numPages      word
    JS_time          SpoolTimeStruct <>
    JS_printing      byte
JobStatus           ends
```

This structure is returned by the **SpoolJobsInfo** library call.

JS_fname stores the standard DOS (8.3) spool filename.

JS_parent stores the parent application's name.

JS_documentName stores the document name.

JS_numPages stores the number of pages in the document.

JS_time stores the time spooled.

JS_printing stores the status of printing in progress. (TRUE if we are printing.)

Library: **spool.def**

Reference book

■ Justification

```
Justification      etype byte
  J_LEFT          enum Justification
  J_RIGHT         enum Justification
  J_CENTER        enum Justification
  J_FULL          enum Justification
```

Library: **graphics.def**

■ KbdGrab

```
KbdGrab          struct
  KG_OD          optr
  KG_unused      word
KbdGrab          ends
```

Library:

■ KbdGrab

```
KbdGrab          struct
  KG_OD          optr
  KG_unused      word
KbdGrab          ends
```

Library: **Objects/uiInputC.def**

■ KbdReturnFlags

```
KbdReturnFlags    record
  KRF_PREVENT_PASS_THROUGH :1
  KRF_UNUSED         :15
KbdReturnFlags    end
```

Library: **uiInputC.def**

■ KeyboardOverride

```
KeyboardOverride  etype word
  KO_NO_KEYBOARD  enum KeyboardOverride
  KO_KEYBOARD_REQUIRED enum KeyboardOverride
  KO_KEYBOARD_EMBEDDED enum KeyboardOverride
```



KO_NO_KEYBOARD

This forces the window to act as if none of the child objects accept text input - no floating keyboard will be made available.

KO_KEYBOARD_REQUIRED

This forces the window to act as if a child object required text input, so a floating keyboard will be brought on screen.

KO_KEYBOARD_EMBEDDED

If this is present, it means that the application is providing an keyboard directly inside the box - no floating keyboard is needed.

Library: **genC.def**

■ KeyboardShortcut

```
KeyboardShortcut    record
  KS_PHYSICAL       :1          ;TRUE: match key, not character
  KS_ALT             :1          ;TRUE: <ALT> must be pressed
  KS_CTRL            :1          ;TRUE: <CTRL> must be pressed
  KS_SHIFT           :1          ;TRUE: <SHIFT> must be pressed
  KS_CHAR            Chars:12    ;character itself (Char or VChar)
KeyboardShortcut    end
```

Library: **input.def**

■ KeyboardType

```
KeyboardType        etype byte, 1, 1
  KT_NOT_EXTD        enum KeyboardType ;84-key PC/AT
  KT_EXTD             enum KeyboardType ;102-key PC/AT, PS/2
  KT_BOTH             enum KeyboardType ;does both (U.S. only)
```

Library: **localize.def**

■ KeyMapType

```
KeyMapType      etype word, 1, 1
KEYMAP_US_EXTD  enum KeyMapType
KEYMAP_US       enum KeyMapType
KEYMAP_UK_EXTD  enum KeyMapType
KEYMAP_UK       enum KeyMapType
KEYMAP_GERMANY_EXTD enum KeyMapType
KEYMAP_GERMANY  enum KeyMapType
KEYMAP_SPAIN_EXTD enum KeyMapType
KEYMAP_SPAIN    enum KeyMapType
KEYMAP_DENMARK_EXTD enum KeyMapType
KEYMAP_DENMARK  enum KeyMapType
KEYMAP_BELGIUM_EXTD enum KeyMapType
KEYMAP_BELGIUM  enum KeyMapType
KEYMAP_CANADA_EXTD enum KeyMapType
KEYMAP_CANADA   enum KeyMapType
KEYMAP_ITALY_EXTD enum KeyMapType
KEYMAP_ITALY    enum KeyMapType
KEYMAP_LATIN_AMERICA_EXTD enum KeyMapType
KEYMAP_LATIN_AMERICA enum KeyMapType
KEYMAP_NETHERLANDS enum KeyMapType
KEYMAP_NETHERLANDS_EXTD enum KeyMapType
KEYMAP_NORWAY_EXTD enum KeyMapType
KEYMAP_NORWAY    enum KeyMapType
KEYMAP_PORTUGAL_EXTD enum KeyMapType
KEYMAP_PORTUGAL  enum KeyMapType
KEYMAP_SWEDEN_EXTD enum KeyMapType
KEYMAP_SWEDEN    enum KeyMapType
KEYMAP_SWISS_FRENCH_EXTD enum KeyMapType
KEYMAP_SWISS_FRENCH enum KeyMapType
KEYMAP_SWISS_GERMAN_EXTD enum KeyMapType
KEYMAP_SWISS_GERMAN enum KeyMapType
KEYMAP_FRANCE_EXTD enum KeyMapType
KEYMAP_FRANCE    enum KeyMapType
```

Library: **localize.def**

■ LanguageDialect

```
LanguageDialect record
:8
LD_DEFAULT      :1
LD_ISE_BRITISH  :1
LD_IZE_BRITISH  :1
LD_AUSTRALIAN   :1
```

```

LD_FINANCIAL      :1
LD_LEGAL          :1
LD_MEDICAL        :1
LD_SCIENCE        :1
LanguageDialect   end

```

Library: **sllang.def**

■ LargeMouseData

```

LargeMouseData    struct
; LMD_location must be first entry
LMD_location      PointDWFixed
LMD_buttonInfo    byte
LMD_uiFunctionsActive  UIFunctionsActive
LargeMouseData    ends

```

LMD_location stores the mouse position in <32 bit integer>.<16 bit fraction> format.

LMD_buttonInfo stores **ButtonInfo**.

LMD_uiFunctionsActive stores additional data normally passed as part of mouse event in **bp**. The data normally provided by the bit UIFA_IN is *not* provided by GEOS for LARGE mouse events. The reason for this is that with small mouse events, **VisCallChildUnderPoint** can lock down each child, look at its bounds, & set UIFA_IN correctly. This is not possible with large objects, as the bounds information, if at all existent, is private to that object & not known by the Vis library. The bit will be unchanged from the state it holds going into the VisContentClass handler for the mouse event.

Library: **Objects/uiInput.def**

■ LASCFeatures

```

LASCFeatures      record
LASCF_SINGLE      :1
LASCF_ONE_AND_A_HALF :1
LASCF_DOUBLE      :1
LASCF_TRIPLE      :1
LASCF_CUSTOM      :1
LASCFeatures      end

```

Library: **Objects/Text/tCtrlC.def**

■ LASCToolboxFeatures

```

LASCToolboxFeatures      record
    LASCTF_SINGLE         :1
    LASCTF_ONE_AND_A_HALF :1
    LASCTF_DOUBLE         :1
    LASCTF_TRIPLE         :1
LASCToolboxFeatures      end

```

Library: **Objects/Text/tCtrlC.def**

■ LayerPriority

```

LayerPriority      etype byte
    LAYER_PRIO_MODAL          enum LayerPriority, 6 ; For system-modal dialog
                                ; boxes, when layer is on
                                ; screen
    LAYER_PRIO_ON_TOP         enum LayerPriority, 8 ; For "screen-floating"
                                ; boxes
    LAYER_PRIO_STD            enum LayerPriority, 12 ; Standard layer priority
    LAYER_PRIO_ON_BOTTOM      enum LayerPriority, 14 ; Window stays on bottom

```

Library: **win.def**

■ LibraryCallType

```

LibraryCallType      etype word
    LCT_ATTACH         enum LibraryCallType
    LCT_DETACH         enum LibraryCallType
    LCT_NEW_CLIENT      enum LibraryCallType
    LCT_NEW_CLIENT_THREAD enum LibraryCallType
    LCT_CLIENT_THREAD_EXIT enum LibraryCallType
    LCT_CLIENT_EXIT     enum LibraryCallType

```

Library: **library.def**



■ LineAttr

```
LineAttr      struct
    LA_colorFlag    ColorFlag CF_INDEX ; RGB or INDEX
    LA_color        RGBValue <0,0,0>  ; RGB values or index
    LA_mask         SystemDrawMask     ; draw mask
    LA_mapMode      ColorMapMode       ; color map mode
    LA_end          LineEnd            ; end type
    LA_join         LineJoin           ; join type
    LA_style        LineStyle          ; style type
    LA_width        WWFixed            ; line width
LineAttr      ends
```

This structure is used with **GrSetLineAttr**.

Library: **graphics.def**

■ LineEnd

```
LineEnd      etype byte
    LE_BUTTCAP    enum LineEnd ; but cap
    LE_ROUND CAP  enum LineEnd ; round cap
    LE_SQUARECAP  enum LineEnd ; square cap
```

Library: **graphics.def**

■ LineFlags

```
LineFlags      record
    LF_STARTS_PARAGRAPH      :1
    LF_ENDS_PARAGRAPH        :1
    LF_ENDS_IN_CR            :1
    LF_ENDS_IN_COLUMN_BREAK  :1
    LF_ENDS_IN_SECTION_BREAK :1
    LF_ENDS_IN_NULL          :1
    LF_NEEDS_DRAW            :1
    LF_NEEDS_CALC            :1
    LF_ENDS_IN_AUTO_HYPHEN   :1
    LF_ENDS_IN_OPTIONAL_HYPHEN :1
    LF_INTERACTS_ABOVE       :1
    LF_INTERACTS_BELOW       :1
    LF_LAST_CHAR_EXTENDS_RIGHT :1
    LF_LAST_CHAR_KERNED      :1
    LF_CONTAINS_EXTENDED_STYLE :1
LineFlags      end
```

LF_STARTS_PARAGRAPH
Set if line starts a paragraph.

LF_ENDS_PARAGRAPH

Set if line ends a paragraph.

LF_ENDS_IN_CR

Set if field ends in CR.

LF_ENDS_IN_COLUMN_BREAK

Set if line ends in a column break.

LF_ENDS_IN_SECTION_BREAK

Set if line ends in a section break.

LF_ENDS_IN_NULL

Set if line ends in NULL, last one in document.

LF_NEEDS_DRAW

Set if line needs redrawing.

LF_NEEDS_CALC

Set if line needs calculating.

LF_ENDS_IN_AUTO_HYPHEN

Set if line ends in a generated hyphen.

LF_ENDS_IN_OPTIONAL_HYPHEN

Set if line ends in an optional hyphen.

Sometimes characters in a line will extend outside the top and bottom bounds of the line. We mark these lines with these bits.

LF_INTERACTS_ABOVE

Set if line interacts with line above it.

LF_INTERACTS_BELOW

Set if line interacts with line below it.

When doing an optimized redraw of a line we draw the last field in the line if the field got longer. If the field got shorter we just clear from beyond the right edge of the field. There are a few situations where we can't really do this:

- Current last character on line extended to the right of its font box. (Italic characters are a good example of this).
- The last character on the line was negatively kerned before we made the modification and is that is no longer the case (this character was removed).

We flag these two cases separately.

LF_LAST_CHAR_EXTENDS_RIGHT

Set if the last character on the line extends to the right of its font box.



LF_LAST_CHAR_KERNED

Set if the last character on the line is kerned. The only time we use this is to copy it into the next field.

Set by the application if the line contains styles which are not supported directly by the kernel. This allows applications to optimize line redraw by skipping over code which may attempt to draw attributes which don't exist for the line.

LF_CONTAINS_EXTENDED_STYLE

Set if the line contains a non-kernel supported style.

Library: **text.def**

■ LineInfo

```
LineInfo      struct
  LI_flags    LineFlags
  LI_hgt      WBFixed
  LI_blo      WBFixed
  LI_adjustment word
  LI_count    WordAndAHalf
  LI_spacePad WBFixed
  LI_lineEnd  word
  LI_firstField FieldInfo
LineInfo      ends
```

LI_flags stores miscellaneous line flags.

LI_hgt stores the height of the line (in points).

LI_blo stores the baseline offset (in points).

LI_adjustment stores the adjustment for justification.

LI_count stores the number of characters in the line. This is the sum of the field counts.

LI_spacePad stores the amount to pad last field to get full justification.

LI_lineEnd stores the rounded end-of-line position which indicates the end of the last non-white-space character.

LI_firstField stores the first field of the line. (At least one field is always present.)

Library: **text.def**

■ LineJoin

```
LineJoin          etype byte
    LJ_MITERED      enum LineJoin; miter join
    LJ_ROUND        enum LineJoin; round join
    LJ_BEVELED      enum LineJoin; beveled join

    LAST_LINE_JOIN_TYPE = LJ_BEVELED
```

Library: **graphics.def**

■ LineStyle

```
LineStyle          etype byte
    LS_SOLID        enum LineStyle ; _____ (solid)
    LS_DASHED       enum LineStyle ; _ _ _ _ _ _ (dashed)
    LS_DOTTED       enum LineStyle ; . . . . . (dotted)
    LS_DASHDOT      enum LineStyle ; _ . _ . _ . (dash-dot)
    LS_DASHDDOT     enum LineStyle ; _ . . _ . . (dash-double-dot)
    LS_CUSTOM       enum LineStyle
```

Library: **graphics.def**

■ LinkPart

```
LinkPart          struct
    LP_next        optr
LinkPart          ends
```

The low bit of the optr is clear to indicate a sibling optr; this bit is set to indicate that the optr links a parent. (If 0, then object is not in a composite.)

Library: **Objects/metaC.def**

■ LMemBlockHeader

```
LMemBlockHeader    struct
    LMBH_handle     hptr
    LMBH_offset     nptr.word
    LMBH_flags      LocalMemoryFlags <>
    LMBH_lmemType    LMemType LMEM_TYPE_GENERAL
    LMBH_blockSize  word
    LMBH_nHandles   word
    LMBH_freeList    lptr
    LMBH_totalFree   word
LMemBlockHeader    ends
```

This structure is found at the beginning of every block which contains an LMem heap. You can examine any of the fields (after having locked the block)



but you should not change any of these fields yourself; they are managed by the LMem routines.

LMBH_handle stores the handle of this block.

LMBH_offset stores the offset from the beginning of the block to the beginning of the heap.

LMBH_flags stores the **LocalMemoryFlags** which describe the state of the local memory block.

LMBH_lmemType stores the type of LMem heap in use in this block.

LMBH_blockSize stores the total size of the block. This size may change in either direction as a result of chunk allocation and heap compaction.

LMBH_nHandles stores the number of handles available in the chunk handle table. Not all of these chunks are necessarily allocated as owned or free chunks. The table grows automatically when necessary.

LMBH_freeList stores the chunk handle of the first free chunk in the linked list of free chunks.

LMBH_totalFree stores the total amount of free space in the LMem heap.

Library: **lmem.def**

■ LMemType

```
LMemType          etype word
LMEM_TYPE_GENERAL  enum LMemType
LMEM_TYPE_WINDOW   enum LMemType
LMEM_TYPE_OBJ_BLOCK enum LMemType
LMEM_TYPE_GSTATE   enum LMemType
LMEM_TYPE_FONT_BLK enum LMemType
LMEM_TYPE_GSTRING   enum LMemType
LMEM_TYPE_DB_ITEMS enum LMemType
```

Library: **lmem.def**

■ LocalCmpStringsDosToGeosFlags

```
LocalCmpStringsDosToGeosFlags record
                                     :6
    LCSDTG_NO_CONVERT_STRING_2      :1
    LCSDTGF_NO_CONVERT_STRING_1     :1
LocalCmpStringsDosToGeosFlags end
```

Library: **localize.def**

Reference book

LocalDistanceFlags

```

LocalDistanceFlags record
    LDF_FULL_NAMES          :1
    LDF_PRINT_PLURAL_IF_NEEDED :1
                                :10
    LDF_PASSING_DECIMAL_PLACES :1 ; Internal
    LDF_DECIMAL_PLACES       :1 ; Internal
LocalDistanceFlags end

```

Library: **localize.def**

LocalMemoryFlags

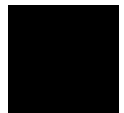
```

LocalMemoryFlags record
    LMF_HAS_FLAGS           :1      ;True if block has a flags block
    LMF_IN_RESOURCE         :1      ;True if block is just loaded from resource
    LMF_DETACHABLE          :1      ;True if block is detachable
    LMF_DUPLICATED          :1      ;True if block created by
                                    ;ObjDuplicateResource
    LMF_RELOCATED           :1      ;True if block is being relocated
    LMF_AUTO_FREE           :1      ;Indicates that block may be freed when
                                    ;in-use count hits 0.
    LMF_IN_LMEM_ALLOC       :1      ;EC ONLY -- In the middle of an LMemAlloc,
                                    ; do not try to do a ECLMemValidateHeap.
                                    ;INTERNAL FLAG -- DO NOT MODIFY
    LMF_IS_VM               :1      ;True if block is stored in VM file and
                                    ;should be marked dirty whenever a chunk
                                    ; is marked dirty.
    LMF_NO_HANDLES          :1      ;Block does not have handles (malloc like)
    LMF_NO_ENLARGE          :1      ;Do not enlarge block to try to alloc
    LMF_RETURN_ERRORS       :1      ;Return errors when allocation impossible
                                :1
                                :1
    LMF_DEATH_COUNT         :3      ;Means nothing if 0, else is # of death
                                    ;methods left which must hit
                                    ;BlockDeathCommon before it will destroy
                                    ;the block. Used by MSG_FREE_DUPLICATE &
                                    ;MSG_REMOVE_BLOCK

LocalMemoryFlags end

```

Library: **lmem.def**



■ MakeRectVisibleFlags

```
MakeRectVisibleFlags    record
                        :8
    MRVF_ALWAYS_SCROLL  :1
    MRVF_USE_MARGIN_FROM_TOP_LEFT :1
                        :6
MakeRectVisibleFlags    end
```

MRVF_ALWAYS_SCROLL

Set if we always want to do the scrolling, even if the object is already partly onscreen. Mostly only useful if an object is already barely onscreen and we want to center it.

MRVF_USE_MARGIN_FROM_TOP_LEFT

Ignore current placement of the object; margins are always calculated from the top or left edge of the view, regardless of the original position of the rectangle.

Library: **Objects/gViewC.def**

■ MakeRectVisibleMargin

```
MakeRectVisibleMargin    etype word
    MRVM_0_PERCENT        enum MakeRectVisibleMargin, 0
    MRVM_25_PERCENT       enum MakeRectVisibleMargin, 0ffffh/4
    MRVM_50_PERCENT       enum MakeRectVisibleMargin, 0ffffh/2
    MRVM_75_PERCENT       enum MakeRectVisibleMargin, 0ffffh*3/4
    MRVM_100_PERCENT      enum MakeRectVisibleMargin, 0ffffh
```

How far to bring the rectangle onscreen. See comments for each constant. If you need to get more a precise percentage, multiply your percentage by 0ffffh and use that rather than one of these constants.

MRVM_0_PERCENT

Scroll the view just far enough to get the rectangle barely onscreen. If the rectangle is larger than the view, brings as much as possible onscreen. If MRVF_USE_MARGIN_FROM_TOP_LEFT is set, always brings the object to the top or left edge of the screen.

MRVM_25_PERCENT

MRVM_50_PERCENT

Centers the object onscreen.

MRVM_75_PERCENT

MRVM_100_PERCENT

Scrolls the rectangle all the way to the opposite edge of the screen from whence it came. Probably only useful if always

using margin from top left, in order to bring something to the bottom edge of the screen.

Library: **Objects/gViewC.def**

■ **MakeRectVisibleParams**

```
MakeRectVisibleParams      struct
  MRVP_bounds              RectDWord
  MRVP_xMargin             MakeRectVisibleMargin
  MRVP_xFlags              MakeRectVisibleFlags
  MRVP_yMargin             MakeRectVisibleMargin
  MRVP_yFlags              MakeRectVisibleFlags
MakeRectVisibleParams      ends
```

MRVP_bounds stores the bounds of the **Rectangle** to make visible. (This rectangle must be less than 65535 points high or wide.)

MRVP_xMargin stores how far to bring the **Rectangle** on screen.

Library: **Objects/gViewC.def**

■ **ManufacturerID**

```
ManufacturerID             etype word
  MANUFACTURER_ID_GEOWORKS          enum ManufacturerID
```

Library: **geode.def**

■ **MapListBlockHeader**

```
MapListBlockHeader        struct
  MLBH_base                LMemBlockHeader
  MLBH_numDestFields       word
  MLBH_chunk1              word
MapListBlockHeader        ends
```

Library: **impex.def**

■ **MCFeatures**

```
MCFeatures                 record
  MCF_LEFT_MARGIN          :1
  MCF_PARA_MARGIN          :1
  MCF_RIGHT_MARGIN         :1
MCFeatures                 end
```

Library: **Objects/Text/tCtrlC.def**



■ MCToolboxFeatures

```
MCToolboxFeatures    record
MCToolboxFeatures    end
```

Library: **Objects/Text/tCtrlC.def**

■ MeasurementType

```
MeasurementType      etype byte
    MEASURE_US                enum MeasurementType
    MEASURE_METRIC            enum MeasurementType
```

Library: **localize.def**

■ MediaType

```
MediaType            etype byte, 0
    MEDIA_NONEXISTENT      enum MediaType; used as error value
    MEDIA_160K             enum MediaType
    MEDIA_180K             enum MediaType
    MEDIA_320K             enum MediaType
    MEDIA_360K             enum MediaType
    MEDIA_720K             enum MediaType
    MEDIA_1M2              enum MediaType
    MEDIA_1M44             enum MediaType
    MEDIA_2M88             enum MediaType
    MEDIA_FIXED_DISK       enum MediaType
    MEDIA_CUSTOM           enum MediaType
    MEDIA_SRAM             enum MediaType
    MEDIA_ATA              enum MediaType
    MEDIA_FLASH            enum MediaType
```

Library: **drive.def**

■ MemGetInfoType

```
MemGetInfoType       etype word, 0, 2
    MGIT_SIZE             enum MemGetInfoType
    MGIT_FLAGS_AND_LOCK_COUNT  enum MemGetInfoType
    MGIT_OWNER_OR_VM_FILE_HANDLE  enum MemGetInfoType
    MGIT_ADDRESS          enum MemGetInfoType
    MGIT_OTHER_INFO       enum MemGetInfoType
    MGIT_EXEC_THREAD      enum MemGetInfoType
```

Library: **heap.def**

■ MenuSepFlags

```
MenuSepFlags      record
  MSF_SEP          :1
  MSF_USABLE       :1
  MSF_FROM_CHILD   :1
                  :5
MenuSepFlags      end
```

MSF_SEP When recursing to lower objects in the menu, this is set when there is at least one usable object between this object and the separator drawn above it. When un-recursing (moving up the menu), this is set when there is at least one usable object between this object and the separator drawn below it.

MSF_USABLE When recursing to lower objects in the menu, this is set when an object has at least one previous sibling which is **GS_USABLE**.

MSF_FROM_CHILD Set when message is sent from a child to its visible parent, so the parent can distinguish from the case where it is called from its previous sibling or parent.

Library: **Objects/visC.def**

■ MessageError

```
MessageError      etype word
  MESSAGE_NO_ERROR      enum MessageError
  MESSAGE_NO_HANDLES    enum MessageError ; short on handles and
                                      ; MF_CAN_DISCARD_IF_DESPERATE
                                      ; was passed
```

Library: **object.def**

■ MessageFlags

```
MessageFlags      record
  MF_CALL          :1
  MF_FORCE_QUEUE    :1
  MF_STACK          :1
                  :1
  MF_CHECK_DUPLICATE :1
  MF_CHECK_LAST_ONLY :1
  MF_REPLACE        :1
```



```

MF_CUSTOM                :1
MF_FIXUP_DS              :1
MF_FIXUP_ES              :1
MF_DISCARD_IF_NO_MATCH  :1
MF_MATCH_ALL             :1
MF_INSERT_AT_FRONT       :1
MF_CAN_DISCARD_IF_DESPERATE:1
MF_RECORD                :1
                          :1

MessageFlags             end

```

Library: **object.def**

■ MetaAlterFTVMCExclFlags

```

MetaAlterFTVMCExclFlags  record
    MAEF_NOT_HERE          :1
    MAEF_SYS_EXCL          :1
    MAEF_APP_EXCL          :1
    MAEF_GRAB              :1
    MAEF_FOCUS             :1
    MAEF_TARGET            :1
    MAEF_MODEL             :1
                          :6
    MAEF_MODAL             :1
    MAEF_OD_IS_WINDOW      :1
    MAEF_OD_IS_MENU_RELATED :1
MetaAlterFTVMCExclFlags  end

```

MAEF_NOT_HERE

Overrides all other flags! Set if this request should *not* be honored here, but instead sent on up the hierarchy with this bit cleared. This bit exists for two reasons:

- 1) So that nodes can tell the difference between messages coming up from objects below & those requests which it has made for itself, which should be handled by the next node up.
- 2) Thus allowing MSG_META_MUP_ALTER_FTVMC_EXCL to be sent to the object making the request itself, thereby allowing nodes the freedom to direct the message in directions other than the visual hierarchy, if the next node is not in that direction.

MAEF_SYS_EXCL

Not passed, but this bit as stored in a HierarchicalGrab structure indicates whether the object has a system-wide exclusive.

Reference book

- MAEF_APP_EXCL**
Not passed, but this bit as stored in a HierarchicalGrab structure indicates whether the object has an applications-wide exclusive.
- MAEF_GRAB**
Set to force grab exclusive, clear to release it.
- MAEF_FOCUS**
Set to grab/release focus.
- MAEF_TARGET**
Set to grab/release target.
- MAEF_MODEL**
Set to grab/release model.
- MAEF_MODAL**
Meaningful for focus grab only—set if object requesting grab is a modal dialog, or a derivative window that happens to have the same focus node above it as the modal dialog (such as a popup menu). If this bit is clear, but the application/field/system etc. is in a modal state, the requesting object's optr will be saved away, but not granted the focus, until the current modal state within that focus node has been completed.
- MAEF_OD_IS_WINDOW**
Meaningful for focus grab only—whether object is a windowed object or not.
- MAEF_OD_IS_MENU_RELATED**
Meaningful for focus grab only—whether object is a specific UI menu-related object

Library: **uiInputC.def**

MetaBase

```
MetaBase      struct
  MB_class    fptr.ClassStruct ; Instance's class
MetaBase      ends
```

This base structure is defined so Esp can build on it for all other classes.

Library: **Objects/metaC.def**

■ **MinIncrementType**

```
MinIncrementType    union
    MIT_US           MinUSMeasure
    MIT_METRIC       MinMetricMeasure
    MIT_POINT        MinPointMeasure
    MIT_PICA         MinPicaMeasure
MinIncrementType    end
```

Library: **ruler.def**

■ **MinMetricMeasure**

```
MinMetricMeasure    etype byte, 0
    MMM_MILLIMETER   enum    MinMetricMeasure
    MMM_HALF_CENTIMETER   enum    MinMetricMeasure
    MMM_CENTIMETER   enum    MinMetricMeasure
```

Library:

■ **MinPicaMeasure**

```
MinPicaMeasure      etype byte, 0
    MPM_PICA         enum    MinPicaMeasure
    MPM_INCH         enum    MinPicaMeasure
```

Library: **ruler.def**

■ **MinPointMeasure**

```
MinPointMeasure     etype byte, 0
    MPM_25_POINT     enum    MinPointMeasure
    MPM_50_POINT     enum    MinPointMeasure
    MPM_100_POINT    enum    MinPointMeasure
```

Library: **ruler.def**

■ **MinUSMeasure**

```
MinUSMeasure        etype byte, 0
    MUSM_EIGHTH_INCH   enum    MinUSMeasure
    MUSM_QUARTER_INCH  enum    MinUSMeasure
    MUSM_HALF_INCH     enum    MinUSMeasure
    MUSM_ONE_INCH      enum    MinUSMeasure
```

Library: **ruler.def**

■ MixMode

```
MixMode  etype byte
MM_CLEAR    enum MixMode ; dest <- 0
MM_COPY     enum MixMode ; dest <- src
MM_NOP      enum MixMode ; dest <- dest
MM_AND      enum MixMode ; dest <- src AND dest
MM_INVERT   enum MixMode ; dest <- NOT dest
MM_XOR      enum MixMode ; dest <- src XOR dest
MM_SET      enum MixMode ; dest <- 1
MM_OR       enum MixMode ; dest <- src OR dest

LAST_MIX_MODE= MM_OR      ; last legal draw mode
```

Library: **graphics.def**

■ MonikerGroupEntry

```
MonikerGroupEntry      struct
MGE_type               VisMonikerListEntryType
MGE_group              word
MonikerGroupEntry      ends
```

Library: **token.def**

■ MonikerMessageParams

```
MonikerMessageParams    struct
MMP_xInset              word
MMP_yInset              word
MMP_xMaximum            word
MMP_yMaximum            word
MMP_gState              hptr.GState
MMP_textHeight          word
MMP_visMoniker          lpPtr.VisMoniker
MMP_monikerFlags        DrawMonikerFlags
MonikerMessageParams    ends
```

MMP_xInset stores the horizontal inset to the start of where to draw the moniker if top or bottom justifying.

MMP_yInset stores the vertical inset to the start of where to draw the moniker, if left or right justifying.



MMP_xMaximum and *MMP_yMaximum* store the maximum size of the moniker. If *VMF_CLIP_TO_MAXIMUM_WIDTH* is set in the *MMP_monikerFlags*, the moniker will be clipped to that width.

MMP_gState stores the gstate to use when drawing the moniker. (This gstate is typically passed into *MSG_VIS_DRAW*).

MMP_textHeight stores the height of the system text, which speeds up many moniker operations. If we happen to know the height of the system text, we should pass it here for speed, or else pass 0.

MMP_visMoniker stores the visual moniker itself. This moniker must be in the same block as the object.

MMP_monikerFlags stores justification information and miscellaneous flags used when drawing the moniker.

Library: **Objects/visC.def**

■ MouseGrab

```
MouseGrab      struct
  MG_OD        optr
  MG_gWin      hptr
MouseGrab      ends
```

This structure is similar to an ordinary “grab” except it additionally stores the window handle that mouse data should be translated into before sending.

Library: **Objects/uiInputC.def**

■ MouseReturnFlags

```
MouseReturnFlags  record
  MRF_PROCESSED      :1
  MRF_REPLAY         :1
  MRF_PREVENT_PASS_THROUGH :1
  MRF_SET_POINTER_IMAGE :1
  MRF_CLEAR_POINTER_IMAGE :1
                      :7
  MRF_INK_RETURN_VALUE InkReturnValue:4
MouseReturnFlags  end
```

MRF_PROCESSED

To be set by any non-window objects which have had mouse events passed on down to them. Used by base window to determine if window background was hit, as opposed to any of its children. This should be returned set by any object finding the mouse within its bounds.

MRF_REPLAY

Will cause event to be played through implied grab if the active grab has gone from a valid grab to no grab, in the MSG_META_BUTTON routine which is returning this flag set. Normally used when a gadget releases the grab because the ptr is out of its range, & it wishes to have the event replayed to the implied grab. Note: in a pre passive button handler, this can be returned to cause the event to be re-sent to the pre-passive list.

MRF_PREVENT_PASS_THROUGH

Set by pre-passive button routines only, if event should NOT be passed through to active/implied mouse grab. Any grab in the pre-passive list may set this bit, & the effect will occur.

MRF_SET_POINTER_IMAGE

Causes the PIL_GADGET level cursor to be changed to cx:dx.

MRF_CLEAR_POINTER_IMAGE

Causes the PIL_GADGET level cursor to be reset to the default.

MRF_INK_RETURN_VALUE

This field is only filled in by handlers for MSG_META_QUERY_IF_PRESS_IS_INK.

Library: **uiInputC.def**



 **Reference book**

■ NameArrayAddFlags

```
NameArrayAddFlags  record
    NAAF_SET_DATA_ON_REPLACE  :1
                                :15
NameArrayAddFlags  end
```

NAAF_SET_DATA_ON_REPLACE

If replacing an existing name set, set data for the name to the data passed.

Library: **chunkarr.def**

■ NameArrayElement

```
NameArrayElement  struct
    NAE_meta        RefElementHeader  ;standard ElementArray header
    NAE_data         label byte        ;data
NameArrayElement  ends
```

The name itself immediately follows the byte of data.

Library: **chunkarr.def**

■ NameArrayHeader

```
NameArrayHeader    struct
    NAH_meta         ElementArrayHeader
    NAH_dataSize      word
NameArrayHeader    ends
```

This structure must be at the front of every name array. Since name arrays are special kinds of element arrays, the **NameArrayHeader** must itself begin with an **ElementArrayHeader**.

NAH_meta stores this element array header.

NAH_dataSize stores the size of the data section of each element in the name array.

Library: **chunkarr.def**



■ NameArrayMaxElement

```
NameArrayMaxElement      struct
    NAME_meta             RefElementHeader
    NAME_data             byte NAME_ARRAY_MAX_DATA_SIZE dup (?)
    NAME_name             char NAME_ARRAY_MAX_NAME_SIZE dup (?)
NameArrayMaxElement      ends
```

Library: **chunkarr.def**

■ NavigateCommonFlags

```
NavigateCommonFlags      record
    NCF_IS_COMPOSITE      :1
    NCF_IS_FOCUSABLE      :1
    NCF_IS_MENU_RELATED   :1
    NCF_IS_ROOT_NODE      :1
                          :4
NavigateCommonFlags      end
```

NCF_IS_COMPOSITE

Set this if calling from a composite object (is subclass of VisCompClass).

NCF_IS_FOCUSABLE

Set this if specific UI will allow this object to get the focus. If NCF_IS_COMPOSITE is also set, means the object itself will get the focus first, followed by its children, in the navigation order.

NCF_IS_MENU_RELATED

Set if this object is menu-related (menu button, or icon in header area).

NCF_IS_ROOT_NODE

Set if this node is the root level of the tree.

Library: **Objects/visC.def**

■ NavigateCommonParams

```
NavigateCommonParams      struct
    NCP_object            optr
    NCP_navFlags          NavigateFlags
    NCP_navCommonFlags    NavigateCommonFlags
    NCP_genericData       lptr
NavigateCommonParams      ends
```

NCP_object stores the object that originated the message. When returned, this entry stores the final recipient of the message.

NCP_genericData stores the chunk handle of generic instance data for the calling object, or zero if not there is no generic data. (This is used for checking for navigation hints on the object.)

Library: **Objects/visC.def**

■ NavigateFlags

```
NavigateFlags      record
; reply flags
NF_COMPLETED_CIRCUIT      :1
NF_REACHED_ROOT           :1
                           :6
; command flags (MUST BE IN LOWER BYTE)
                           :1      ;reserved for future use
                           :1      ;(lines up with VTF_IS_COMPOSITE)
                           :1      ;reserved for future use (lines up with
                           :1      ; GS_ENABLED and NCF_FOCUSABLE)
NF_NAV_MENU_BAR           :1
                           :1      ;reserved for future use (lines up with
                           :1      ;VTF_IS_WIN_GROUP and NCF_IS_ROOT)
NF_INITIATE_QUERY         :1
NF_SKIP_NODE              :1
NF_TRAVEL_CIRCUIT         :1
NF_BACKTRACK_AFTER_TRAVELING :1
NavigateFlags      end
```

NF_COMPLETED_CIRCUIT

This is set when this message is received at a node and the originating node (in **^lcx:dx**) matches the recipient node, meaning we have travelled the entire circuit. Recursion ends at this point. This is useful for preventing infinite looping in cases where the circuit consists of 1 object, and is handy for error checking after objects have been added/removed from the WIN_GROUP.

NF_REACHED_ROOT

This might be useful for error checking. Is set as this message is received by the root (WIN_GROUP object, and is passed on to the first child.

NF_NAV_MENU_BAR

Set to navigate through items considered part of the menu bar area: menu buttons, icons in the header area, etc. Clear this flag to navigate through control-items in the window area.

NF_INITIATE_QUERY

When a MSG_SPEC_NAVIGATE_TO_NEXT_FIELD or MSG_SPEC_NAVIGATE_TO_NEXT_FIELD handler sends this



query to an object, this flag is set, since the `cx:dx` passed are the object, and don't want to get confused and think that we have already travelled the entire circuit. This flag is reset before the message is passed onto the next node in the circuit.

NF_SKIP_NODE

Set to tell the recipient to forward the message on to the next object in the circuit. if reset, and is composite, will forward message to first child. If leaf node, will forward to next sibling. If is last sibling, will forward to parent passing NF_SKIP_NODE set so that parent will forward to sibling or parent.

NF_TRAVEL_CIRCUIT

Set to force message to be passed through the entire navigation circuit, back to the originating object. This is useful for error checking, and for the "navigate to previous" case.

Library: **Objects/visC.def**

■ NCCFlagsUnion

```
NCCFlagsUnion      union
  NCCFU_char        VisTextCharAttrFlags
  NCCFU_paragraph   VisTextParaAttrFlags
  NCCFU_border      VisTextParaAttrBorderFlags
NCCFlagsUnion      ends
```

Library: **Objects/Text/tCtrlC.def**

■ NoteType

```
NoteType etype byte, 0, 2
  NT_INK      enum NoteType
  NT_TEXT     enum NoteType
```

Library: **pen.def**

■ NotificationType

```
NotificationType  struct
  NT_manuf        ManufacturerID
  NT_type         word
NotificationType  ends
```

This structure defines a basic GCN notification type that can be passed with MSG_META_NOTIFY and MSG_META_NOTIFY_WITH_DATA_BLOCK.

Library: **Objects/metaC.def**

Reference book

■ NotifyColorChange

```
NotifyColorChange  struct
    NCC_color      ColorQuad
    NCC_grayScreen  SystemDrawMask
    NCC_pattern     GraphicPattern
    NCC_flags       NCCFlagsUnion
                    ;VTCAF_MULTIPLE_COLORS
                    ;VTCAF_MULTIPLE_GRAY_SCREEN
                    ;VTCAF_MULTIPLE_PATTERNS
                    ; -or-
                    ;VTCAF_MULTIPLE_BG_COLORS
                    ;VTCAF_MULTIPLE_BG_GRAY_SCREEN
                    ;VTCAF_MULTIPLE_BG_PATTERNS
                    ; -or-
                    ;VTPAF_MULTIPLE_BG_COLORS
                    ;VTPAF_MULTIPLE_BG_GRAY_SCREEN
                    ;VTPAF_MULTIPLE_BG_PATTERNS
                    ; -or-
                    ;VTPABF_MULTIPLE_BORDER_COLORS
                    ;VTPABF_MULTIPLE_BORDER_GRAY_SCREEN
                    ;VTPABF_MULTIPLE_BORDER_PATTERNS
NotifyColorChange  ends
```

Library: **Objects/Text/tCtrlC.def**

■ NotifyDisplayChange

```
NotifyDisplayChange  struct
    NDC_displayNum   word
    NDC_name         char MAX_DISPLAY_NAME_SIZE dup (?)
    NDC_overlapping  BooleanByte
NotifyDisplayChange  ends
```

Library: **Objects/gDCtrlC.def**

■ NotifyDisplayListChange

```
NotifyDisplayListChange  struct
    NDLC_counter     word
    NDLC_group       optr
NotifyDisplayListChange  ends
```

Library: **Objects/gDCtrlC.def**



■ NotifyDocumentChange

```
NotifyDocumentChange      struct
  NDC_attrs                GenDocumentAttrs
  NDC_type                 GenDocumentType
  NDC_fileHandle           hptr
  NDC_emptyExists          BooleanByte
  NDC_defaultExists        BooleanByte
NotifyDocumentChange      ends
```

Library: **Objects/gDocCtrl.def**

■ NotifyEnabledFlags

```
NotifyEnabledFlags record
  NEF_STATE_CHANGING      :1      ;this is the object whose state is changing
                           :7
NotifyEnabledFlags      end
```

Library: **Objects/genC.def**

■ NotifyFloatFormatChange

```
NotifyFloatFormatChange   struc
  NFFC_vmFileHan          word
  NFFC_vmBlkHan           word
  NFFC_format              word
  NFFC_count              word
NotifyFloatFormatChange   ends
```

Library: **math.def**

■ NotifyFocusWindowKbdStatus

```
NotifyFocusWindowKbdStatus      struct
  NFWKS_needsFloatingKbd      word
  NFWKS_kbdPosition           Point<>
  NFWKS_focusWindow           optr
  NFWKS_sysModal              word
```

NFWKS_needsFloatingKbd

If non-zero, the window needs a floating keyboard - otherwise, it already has an embedded keyboard.

NFWKS_kbdPosition

The position at which to display the keyboard

NFWKS_focusWindow

The OD of the window with the focus. This is used by embedded keyboards, so they can enable themselves when their parent window has the focus.

NFWKS_sysModal

Either zero if window is not sys-modal or 0xffff if it is

Library:

■ NotifyFontAttrChange

```
NotifyFontAttrChange    struct
    NFAC_fontWeight      byte
    NFAC_fontWeightDiffs byte
    NFAC_fontWidth       byte
    NFAC_fontWidthDiffs  byte
    NFAC_trackKerning     word
    NFAC_trackKerningDiffs byte
NotifyFontAttrChange    ends
```

Library: **Objects/Text/tCtrlC.def**

■ NotifyFontChange

```
NotifyFontChange    struct
    NFC_fontID      FontID
    NFC_diffs       byte
NotifyFontChange    ends
```

Library: **Objects/Text/tCtrlC.def**

■ NotifyGenControlStatusChange

```
NotifyGenControlStatusChange    struct
    NGCS_controller      optr
    NGCS_statusChange     GenControlStatusChange
NotifyGenControlStatusChange    ends
```

NGCS_controller stores the optr of the GenControl object itself. This optr may be used to send messages or fetch information from the GenControl (typically to add or remove features).

NGCS_statusChange stores the type of status change that the GenControl object has undergone.

Library: **Objects/gCtrlC.def**



■ NotifyInkHasTarget

```
NotifyInkHasTarget      struct
    NIHT_optr           optr
NotifyInkHasTarget      ends
```

This structure is sent to objects requesting GWNT_INK_HAS_TARGET notification.

Library: **pen.def**

■ NotifyJustificationChange

```
NotifyJustificationChange      struct
    NJC_justification          Justification
    NJC_diffs                  byte
    NJC_useGeneral              byte      ;if non-zero then use "general" in place
                                         ;of "full" justification.

NotifyJustificationChange      ends
```

Library: **Objects/Text/tCtrlC.def**

■ NotifyPageInfoChange

```
NotifyPageInfoChange      struct
    NPIC_width              word
    NPIC_height              word
    NPIC_rightMargin         word      ; 13.3 (8* actual value)
    NPIC_leftMargin          word      ; 13.3 (8* actual value)
    NPIC_topMargin           word      ; 13.3 (8* actual value)
    NPIC_bottomMargin        word      ; 13.3 (8* actual value)
NotifyPageInfoChange      ends
```

Library: **pageInfo.def**

■ NotifyPageStateChange

```
NotifyPageStateChange      struct
    NPSC_firstPage          word      ;first page
    NPSC_lastPage           word      ;last page
    NPSC_currentPage        word      ;current page
NotifyPageStateChange      ends
```

Library: **Objects/gPageCC.def**

■ NotifyPointSizeChange

```
NotifyPointSizeChange    struct
    NPSC_pointSize  WWFixed
    NPSC_diffs      byte
NotifyPointSizeChange    ends
```

Library: **Objects/Text/tCtrlC.def**

■ NotifySearchReplaceEnableChange

```
NotifySearchReplaceEnableChange    struct
    NSREC_flags      SearchReplaceEnableFlags
NotifySearchReplaceEnableChange    ends
```

Library: **Objects/Text/tCtrlC.def**

■ NotifySelectStateChange

```
NotifySelectStateChange    struct
    NSSC_selectionType      SelectionDataType
    NSSC_clipboardableSelection  BooleanByte
    NSSC_selectAllAvailable    BooleanByte
    NSSC_deleteableSelection    BooleanByte
    NSSC_pasteable             BooleanByte
NotifySelectStateChange    ends
```

NSSC_selectionType determines if a text object has the target and a selection.

NSSC_clipboardableSelection stores BB_TRUE if a selection that can be copied to the clipboard exist.

NSSC_selectAllAvailable stores BB_TRUE if “select all” is allowed.

NSSC_deleteableSelection stores BB_TRUE if a selection that can be deleted exists.

NSSC_pasteable stores BB_TRUE if the current clipboard is “pasteable.”

Library: **Objects/gEditCC.def**



■ NotifyTextStyleChange

```
NotifyTextStyleChange    struct
    NTSC_styles           TextStyle
    NTSC_indeterminates   TextStyle
NotifyTextStyleChange    ends
```

Library: **Objects/Text/tCtrlC.def**

■ NotifyUndoStateChange

```
NotifyUndoStateChange    struct
    NUSC_undoTitle   optr
    NUSC_undoType     UndoDescription
NotifyUndoStateChange    ends
```

NUSC_undoTitle stores the title of the current undo item or 0:0 if there is no operation to undo.

Library: **Objects/gEditCC.def**

■ NotifyViewStateChange

```
NotifyViewStateChange    struct
    NVSC_origin           PointDWordFixed
    NVSC_docBounds        RectDWord
    NVSC_increment        PointDWord
    NVSC_scaleFactor      PointWWFixed
    NVSC_color            ColorQuad
    NVSC_attrs            GenViewAttrs
    NVSC_horizAttrs       GenViewDimensionAttrs
    NVSC_vertAttrs        GenViewDimensionAttrs
    NVSC_inkType           GenViewInkType
    NVSC_contentSize      XYSIZE
    NVSC_contentScreenSize XYSIZE
    NVSC_originRelative    PointDWord
    NVSC_documentSize      PointDWord
NotifyViewStateChange    ends
```

Library: **Objects/gViewCC.def**

■ NumberFormatFlags

```
NumberFormatFlags    record
                        :7
    NFF_LEADING_ZERO    :1
NumberFormatFlags    end
```

Library: **localize.def**

■ NumberType

```
NumberType            etype byte, 0, 1
    NT_VALUE           enum NumberType ; It's just a number
    NT_BOOLEAN         enum NumberType ; It's a boolean
    NT_DATE_TIME       enum NumberType ; It's a date/time
```

Library: **parse.def**

■ ObjChunkFlags

```
ObjChunkFlags        record
                        :3
    OCF_VARDATA_RELOC    :1
    OCF_DIRTY            :1
    OCF_IGNORE_DIRTY    :1
    OCF_IN_RESOURCE      :1
    OCF_IS_OBJECT        :1
ObjChunkFlags        end
```

Library: **object.def**

■ ObjCompCallType

```
ObjCompCallType      etype word, 0, 2
    OCCT_SAVE_PARAMS_TEST_ABORT    enum ObjCompCallType
    OCCT_SAVE_PARAMS_DONT_TEST_ABORT    enum ObjCompCallType
    OCCT_DONT_SAVE_PARAMS_TEST_ABORT    enum ObjCompCallType
    OCCT_DONT_SAVE_PARAMS_DONT_TEST_ABORT    enum ObjCompCallType
    OCCT_ABORT_AFTER_FIRST    enum ObjCompCallType
    OCCT_COUNT_CHILDREN    enum ObjCompCallType
```

Library: **Objects/metaC.def**



■ ObjectTransform

```
ObjectTransform      struct
    OT_center        PointDWFxed
    OT_width          WWFixed
    OT_height         WWFixed
    OT_parentWidth    WWFixed
    OT_parentHeight   WWFixed
    OT_transform      GrObjTransMatrix
ObjectTransform      ends
```

OT_center stores the center of the object in the parent's coordinate system.

OT_width stores the width of object in the object's coordinate system

OT_height stores the height of object in the object's coordinate system.

OT_parentWidth stores the width of the object in the parent's coordinate system. This width includes line width and can therefore be used for invalidation, etc.

OT_parentHeight stores the height of the object in the parent's coordinate system. This height includes line height, etc. and can therefore be used for invalidation, etc.

Library: **grobj.def**

■ ObjFlushInputQueueNextStop

```
ObjFlushInputQueueNextStop      etype word, 0, 2
OFIQNS_INPUT_MANAGER             enum ObjFlushInputQueueNextStop
OFIQNS_SYSTEM_INPUT_OBJ          enum ObjFlushInputQueueNextStop
OFIQNS_INPUT_OBJ_OF_OWNING_GEODE enum ObjFlushInputQueueNextStop
OFIQNS_PROCESS_OF_OWNING_GEODE   enum ObjFlushInputQueueNextStop
OFIQNS_DISPATCH                  enum ObjFlushInputQueueNextStop
```

OFIQNS_INPUT_MANAGER

MF_FORCE_QUEUE message to the kernel's input manager thread passing OFIQNS_INPUT_OBJ_OF_OWNING_GEODE.

OFIQNS_SYSTEM_INPUT_OBJ

MF_FORCE_QUEUE message to the System input object (usually the GenSystem object), passing OFIQNS_INPUT_OBJ_OF_OWNING_GEODE.

OFIQNS_INPUT_OBJ_OF_OWNING_GEODE

MF_FORCE_QUEUE message next to the InputObj of the geode owning the block that the object is in, passing OFIQNS_PROCESS_OF_OWNING_GEODE.

OFIQNS_PROCESS_OF_OWNING_GEODE
MF_FORCE_QUEUE message next to the process of the geode
owning the block that the object is in, passing
OFIQNS_DISPATCH.

OFIQNS_DISPATCH
Queues are flushed, so FORCE_QUEUE dispatch passed Event.

Library: **Objects/metaC.def**

■ ObjLMemBlockHeader

```
ObjLMemBlockHeader    struct
  OLMBH_header         LMemBlockHeader <>
  OLMBH_inUseCount     word
  OLMBH_interactibleCount word
  OLMBH_output         optr
  OLMBH_resourceSize   word
ObjLMemBlockHeader    ends
```

This structure is the standard object block (resource) header that begins every object block. Since Object blocks are types of LMem blocks, the first entry in the **ObjLMemBlockHeader** must contain a **LMemBlockHeader**.

OLMBH_header stores the LMem block header.

OLMBH_inUseCount stores the “in use” count for this block. If this count is not zero, then the block may not safely be freed.

OLMBH_interactibleCount stores the “interactable” count for this block, which prevents the block from being swapped. If not zero, then one or more objects in the block are either visible to the user or about to be activated by the user (e.g. via keyboard shortcut).

OLMBH_output stores the optr of the object that will be notified about changes in the resource status, such as in-use counts changing to or from zero. Messages may be sent to this optr using the **TravelOption** TO_OBJ_BLOCK_OUTPUT.

OLMBH_resourceSize stores the size of the object block.

Library: **object.def**



■ ObjRelocation

```
ObjRelocation      struct
    OR_type        ObjRelocationType    ; Type of relocation
    OR_offset      word                 ; Offset to relocation
ObjRelocation      ends
```

This is the structure of an object relocation table entry (for instance data).

Library: **object.def**

■ ObjRelocationID

```
ObjRelocationID    record
    RID_SOURCE      ObjRelocationSource:4
    RID_INDEX       :12
ObjRelocationID    end
```

Library: **object.def**

■ ObjRelocationSource

```
ObjRelocationSource    etype byte
    ORS_NULL           enum ObjRelocationSource
    ORS_OWNING_GEODE   enum ObjRelocationSource
    ORS_KERNEL         enum ObjRelocationSource
    ORS_LIBRARY         enum ObjRelocationSource
    ORS_CURRENT_BLOCK  enum ObjRelocationSource
    ORS_VM_HANDLE      enum ObjRelocationSource
    ORS_OWNING_GEODE_ENTRY_POINT enum ObjRelocationSource
    ORS_NON_STATE_VM   enum ObjRelocationSource
    ORS_UNKNOWN_BLOCK  enum ObjRelocationSource
    ORS_EXTERNAL       enum ObjRelocationSource
```

ORS_NULL
ObjRelocation to zero.

ORS_OWNING_GEODE
Resource of geode. index is resource ID

ORS_KERNEL
Kernel entry point

ORS_LIBRARY
Index is imported library number.

ORS_CURRENT_BLOCK
Handle of block.

ORS_VM_HANDLE
Reloc to handle of block saved on the saved list—index is a VM
id

ORS_OWNING_GEODE_ENTRY_POINT
Entry point of owner.

ORS_NON_STATE_VM
Index is a VM index, vm file handle stored in block header

ORS_UNKNOWN_BLOCK
Index is a handle>>4

ORS_EXTERNAL
Internal.

Library: **object.def**

■ ObjRelocationType

ObjRelocationType etype byte

RELOC_END_OF_LIST	enum ObjRelocationType
RELOC_HANDLE	enum ObjRelocationType;resource ID to handle
RELOC_SEGMENT	enum ObjRelocationType;resource ID to segment
RELOC_ENTRY_POINT	enum ObjRelocationType;resource ID/entry #

RELOC_END_OF_LIST
Relocation from resource ID to handle. The target contains the
resource identification (described below)

RELOC_HANDLE
Relocation from resource ID to segment. The target contains
the resource identification (described below)

RELOC_SEGMENT
Relocation from resource ID/entry point number to a far
pointer. The low word of the target contains the resource
identification and the high word of the target contains the
entry point number.

RELOC_ENTRY_POINT
Resource ID/entry number.

Library: **object.def**



■ OldErrorCheckingFlags

```
OldErrorCheckingFlags    record
    OECF_REGION:1        ;Region checking
    OECF_HEAP_FREE_BLOCKS:1 ;Ensure that all free blocks are 0xcccc
    OECF_LMEM_INTERNAL:1  ;Internal lmem checking
    OECF_LMEM_FREE_AREAS:1 ;Ensure that all free areas are 0xcccc
    OECF_LMEM_OBJECT:1    ;Consistency checks on objects in lmem chunks
    OECF_BLOCK_CHECKSUM:1 ;Checksum on a particular block
    OECF_GRAPHICS:1       ;Misc graphics stuff
    OECF_SEGMENT:1        ;Extensive segment checking
    OECF_NORMAL:1         ;Misc kernel error checking
    OECF_VMEM:1           ;VM file consistency
    OECF_APP:1            ;Application error checking (if implemented
                          ;by applications)
    OECF_LMEM_MOVE:1      ;Force lmem blocks to move whenever possible
    OECF_UNLOCK_MOVE:1   ;Force unlocked blocks to move
    OECF_VMEM_DISCARD:1   ;Force clean VM blocks to be discarded
    OECF_ANAL_VMEM:1      ;Extensive VM error checking
    OECF_TEXT:1
OldErrorCheckingFlags    end
```

Library: **ec.def**

■ OperatorStackElement

```
OperatorStackElement    struct
    OSE_type              EvalStackOperatorType ; Type of the operator
    OSE_data              EvalStackOperatorData ; The associated data
OperatorStackElement    ends
```

Library: **parse.def**

■ OperatorType

```

OperatorType      etype byte, 0, 1
  OP_RANGE_SEPARATOR      enum OperatorType
  OP_NEGATION              enum OperatorType
  OP_PERCENT               enum OperatorType
  OP_EXPONENTIATION        enum OperatorType
  OP_MULTIPLICATION        enum OperatorType
  OP_DIVISION              enum OperatorType
  OP_MODULO                enum OperatorType
  OP_ADDITION              enum OperatorType
  OP_SUBTRACTION           enum OperatorType
  OP_EQUAL                 enum OperatorType
  OP_NOT_EQUAL             enum OperatorType
  OP_LESS_THAN             enum OperatorType
  OP_GREATER_THAN          enum OperatorType
  OP_LESS_THAN_OR_EQUAL    enum OperatorType
  OP_GREATER_THAN_OR_EQUAL enum OperatorType
;
;
OP_STRING_CONCAT          enum OperatorType
OP_RANGE_INTERSECTION      enum OperatorType
;
; The following are graphic versions of existing operators. For example,
; OP_NOT_EQUAL_GRAPHIC is the same as OP_NOT_EQUAL, but shows up on screen
; as an equals sign with a line through it.
;
OP_NOT_EQUAL_GRAPHIC      enum OperatorType
OP_DIVISION_GRAPHIC       enum OperatorType
OP_LESS_THAN_OR_EQUAL_GRAPHIC enum OperatorType
OP_GREATER_THAN_OR_EQUAL_GRAPHIC enum OperatorType
;
; The following are included here because it's convenient. They represent
; an as yet undecided operator. The scanner has seen it and recognized it.
; The parser will decide which operator it is.
;
; The problem is that a single lexical token can correspond to two different
; operations, depending on the context:
; Percent Operator:
;   9%   <- Divides the value to the left by 100
; Modulo Operator:
;   9%2  <- Performs the operation "9 MOD 2"
; Negation Operator:
;   -5   <- Negates the value to the right
; Subtraction Operator:
;   3-5  <- Performs the operation "3 MINUS 5"
;
; Since the scanner has no idea what is coming next in the input stream it
; can only return that the operator is undecided.
;

```



```
; These operators should always be the last in the list since none of the
; tables depend on them.
;
OP_PERCENT_MODULO          enum OperatorType
OP_SUBTRACTION_NEGATION    enum OperatorType
```

Library: **parse.def**

■ OriginChangedParams

```
OriginChangedParams      struct
  OCP_origin  PointDWord  ;new origin
  OCP_window  lpstr Window ;window of view
OriginChangedParams      ends
```

Library: **Objects/gViewC.def**

■ PACFeatures

```
PACFeatures              record
  PACF_WORD_WRAP          :1
  PACF_COLUMN_BREAK_BEFORE :1
  PACF_KEEP_PARA_WITH_NEXT :1
  PACF_KEEP_PARA_TOGETHER  :1
  PACF_KEEP_LINES          :1
PACFeatures              end
```

Library: **Objects/Text/tCtrlC.def**

■ PACToolboxFeatures

```
PACToolboxFeatures record
PACToolboxFeatures end
```

Library: **Object/Text/tCtrlC.def**

■ PageEndCommand

```
PageEndCommand          etype  byte
  PEC_FORM_FEED          enum   PageEndCommand ; 0 = form feed
  PEC_NO_FORM_FEED       enum   PageEndCommand ; 1 = no form feed
```

Library: **graphics.def**

■ PageLayout

```

PageLayout          union
  PL_paper           PageLayoutPaper
  PL_envelope        PageLayoutEnvelope
  PL_label           PageLayoutLabel
PageLayout          end

```

Library: **spool.def**

■ PageLayoutEnvelope

```

PageLayoutEnvelope record
  :12
  PLE_ORIENTATION EnvelopeOrientation:1
  PLE_TYPE         PageType:3           ; PT_ENVELOPE
PageLayoutEnvelope end

```

Library: **spool.def**

■ PageLayoutLabel

```

PageLayoutLabel    record
  :1
  PLL_ROWS          :6           ; labels down
  PLL_COLUMNS       :6           ; labels across
  PLL_TYPE          PageType:3   ; PT_LABEL
PageLayoutLabel    end

```

Library: **spool.def**

■ PageLayoutPaper

```

PageLayoutPaper    record
  :12
  PLP_ORIENTATION   PaperOrientation:1
  PLP_TYPE          PageType:3       ; PT_PAPER
PageLayoutPaper    end

```

Library: **spool.def**



■ PageSetupInfo

```

PageSetupInfo      struct
    PSI_meta        VMChainLink
    PSI_pageSize     XYSIZE      ; In pixels (points)
    PSI_layout       PageLayout
    PSI_numColumns   word
    PSI_columnSpacing word        ; In points * 8
    PSI_ruleWidth    word        ; In pixels (points)
    ;
    ; The margins are relative to the edges of the page, and are in points * 8
    ;
    PSI_leftMargin   word
    PSI_rightMargin  word
    PSI_topMargin    word
    PSI_bottomMargin word
PageSetupInfo      ends

```

Library: **Objects/vTextC.def**

■ PageSizeControlAttrs

```

PageSizeControlAttrs record
    ; EXTERNAL
    PZCA_ACT_LIKE_GADGET :1
    PZCA_PAPER_SIZE      :1
    PZCA_INITIALIZE      :1
    :5
    ; INTERNAL
    PZCA_NEW_PAGE_TYPE   :1 ; INTERNAL
    PZCA_SWAP_WIDTH_HEIGHT :1 ; INTERNAL
    PZCA_SIZE_LIST_INITIALIZED :1 ; INTERNAL
    PZCA_IGNORE_UPDATE   :1 ; INTERNAL
    PZCA_PORTRAIT_VALID  :1
    PZCA_LANDSCAPE_VALID :1
    :2
PageSizeControlAttrs end

```

PZCA_ACT_LIKE_GADGET

Tells the `PageSizeControl` object to act like any other generic gadget, where one uses messages to set/get the state of the object.

PZCA_PAPER_SIZE

Tell the `PageSizeControl` to display paper, and not document, sizes. Most applications will *not* want this set.

PZCA_INITIALIZE

Initialize with the default system values. The flag PZCA_ACT_LIKE_GADGET must be set if you want to use this attribute.

Library: **spool.def**

■ PageSizeControlChanges

```

PageSizeControlChanges    struct
    PSCC_destination      optr    ; destination for message
    PSCC_message          word    ; message to be sent
;
;      Pass:      SS:BP      = PageSizeReport
;                DX         = size PageSizeReport
;      Returns: Nothing
;                AX, CX, DX, BP - may destroy
;
PageSizeControlChanges    ends

```

Library: **spool.def**

■ PageSizeControlFeatures

```

PageSizeControlFeatures    record
    PSIZECF_MARGINS        :1      ; not part of default features
    PSIZECF_CUSTOM_SIZE    :1
    PSIZECF_LAYOUT         :1
    PSIZECF_SIZE_LIST      :1
    PSIZECF_PAGE_TYPE      :1
PageSizeControlFeatures    end

```

Library: **spool.def**

■ PageSizeControlMaxDimensions

```

PageSizeControlMaxDimensions    struct
    PZCMD_width           dword    ; maximum width
    PZCMD_height          dword    ; maximum height
PageSizeControlMaxDimensions    ends

```

Library: **spool.def**



■ PageSizeControlToolboxFeatures

```

PageSizeControlToolboxFeatures    record
    PSIZECTF_DIALOG_BOX           :1
PageSizeControlToolboxFeatures    end

```

Library: **spool.def**

■ PageSizeReport

```

PageSizeReport    struct
    PSR_width      dword           ; width of the page
    PSR_height     dword           ; height of the page
    PSR_layout     PageLayout      ; layout options
    PSR_margins    PCMarginParams  ; document margins
PageSizeReport    ends

```

Library: **spool.def**

■ PageType

```

PageType          etype word, 0, 2
    PT_PAPER       enum PageType
    PT_ENVELOPE    enum PageType
    PT_LABEL       enum PageType

```

Library: **print.def**

■ Palette

```

Palette          struct
    P_entries     word 16          ; Number of 3-byte entries in palette.
Palette          ends

```

This structure stores the custom palettes allocated by **GrCreatePalette** and associated with windows.

Library: **color.def**

■ PaperOrientation

```

PaperOrientation  etype byte, 0, 1
    PO_PORTRAIT   enum PaperOrientation
    PO_LANDSCAPE  enum PaperOrientation

```

Library: **print.def**

■ ParserFlags

```

ParserFlags      record
;
; These are initialized by the parser. They are initialized to zero.
;
PF_HAS_LOOKAHEAD      :1      ; The next token to get is the look-ahead
                           ; token.
PF_CONTAINS_DISPLAY_FUNC :1      ; Set: This expression contains a function
                           ; which should be evaluated when the
                           ; result of the expression is displayed.

PF_OPERATORS          :1      ; Set: Allow operators.
PF_NUMBERS             :1      ; Set: Allow numbers.
PF_CELLS               :1      ; Set: Allow cell references.
PF_FUNCTIONS           :1      ; Set: Allow functions.
PF_NAMES               :1      ; Set: Allow names.
PF_NEW_NAMES           :1      ; Set: Allow new names (app only).
ParserFlags          end

```

Library: **parse.def**



■ ParserParameters

```
ParserParameters    struct
;
; Applications should initialize these fields before calling ParseString
;
PP_common            CommonParameters <>
;
; Possible callbacks:
; CT_FUNCTION_TO_TOKEN
; CT_NAME_TO_TOKEN
;
PP_parserBufferSize  word           ; Size of the buffer
;
; Fields below this point are initialized by ParseString
;
PP_flags             ParserFlags    ; Parsing flags
PP_textPtr           fptr.char      ; Pointer to text
PP_currentToken       ScannerToken   ; Current token
PP_lookAheadToken     ScannerToken   ; Look ahead token

PP_error             ParserScannerEvaluatorError
PP_tokenStart         word           ; Offset to start of token
PP_tokenEnd           word           ; Offset to end of token
ParserParameters    ends
```

The parser allocates this structure on the stack when performing parsing operations such as **ParseString**.

Library: **parse.def**

■ ParserScannerEvaluatorError

```

ParserScannerEvaluatorError    etype byte, 0, 1
;
; Scanner errors
;
PSEE_BAD_NUMBER                enum ParserScannerEvaluatorError
PSEE_BAD_CELL_REFERENCE        enum ParserScannerEvaluatorError
PSEE_NO_CLOSE_QUOTE            enum ParserScannerEvaluatorError
PSEE_COLUMN_TOO_LARGE          enum ParserScannerEvaluatorError
PSEE_ROW_TOO_LARGE             enum ParserScannerEvaluatorError
PSEE_ILLEGAL_TOKEN             enum ParserScannerEvaluatorError
;
; Parser errors
;
PSEE_GENERAL                   enum ParserScannerEvaluatorError
PSEE_TOO_MANY_TOKENS          enum ParserScannerEvaluatorError
PSEE_EXPECTED_OPEN_PAREN       enum ParserScannerEvaluatorError
PSEE_EXPECTED_CLOSE_PAREN      enum ParserScannerEvaluatorError
PSEE_BAD_EXPRESSION            enum ParserScannerEvaluatorError
PSEE_EXPECTED_END_OF_EXPRESSION enum ParserScannerEvaluatorError
PSEE_MISSING_CLOSE_PAREN       enum ParserScannerEvaluatorError
PSEE_UNKNOWN_IDENTIFIER        enum ParserScannerEvaluatorError
PSEE_NOT_ENOUGH_NAME_SPACE     enum ParserScannerEvaluatorError
;
; Serious evaluator errors
;
PSEE_OUT_OF_STACK_SPACE        enum ParserScannerEvaluatorError
PSEE_NESTING_TOO_DEEP          enum ParserScannerEvaluatorError
;
; Evaluator errors that are returned as the result of formulas.
; These are returned on the argument stack.
;
PSEE_ROW_OUT_OF_RANGE          enum ParserScannerEvaluatorError
PSEE_COLUMN_OUT_OF_RANGE       enum ParserScannerEvaluatorError
PSEE_FUNCTION_NO_LONGER_EXISTS enum ParserScannerEvaluatorError
PSEE_BAD_ARG_COUNT             enum ParserScannerEvaluatorError
PSEE_WRONG_TYPE                enum ParserScannerEvaluatorError
PSEE_DIVIDE_BY_ZERO            enum ParserScannerEvaluatorError
PSEE_UNDEFINED_NAME            enum ParserScannerEvaluatorError
PSEE_CIRCULAR_REF              enum ParserScannerEvaluatorError
PSEE_CIRCULAR_DEP              enum ParserScannerEvaluatorError
PSEE_CIRC_NAME_REF             enum ParserScannerEvaluatorError
PSEE_NUMBER_OUT_OF_RANGE       enum ParserScannerEvaluatorError
PSEE_GEN_ERR                   enum ParserScannerEvaluatorError
PSEE_NA                        enum ParserScannerEvaluatorError
;
; Dependency errors
;
PSEE_TOO_MANY_DEPENDENCIES     enum ParserScannerEvaluatorError

```



```

;
; Applications can define errors too, they start here.
;
PSEE_FIRST_APPLICATION_ERROR          enum ParserScannerEvaluatorError, 0xc0
;
; !!! NOTE !!!
; These PSEE_ errors map directly to the floating point errors
; Any change in the float library errors require corresponding
; changes here.
;
PSEE_FLOAT_POS_INFINITY               enum ParserScannerEvaluatorError, 250
PSEE_FLOAT_NEG_INFINITY               enum ParserScannerEvaluatorError
PSEE_FLOAT_GEN_ERR                    enum ParserScannerEvaluatorError

```

Library: **parse.def**

■ ParserToken

```

ParserToken          struct
    PT_type           ParserTokenType    ; Type of token data
    PT_data            ParserTokenData    ; The data itself
ParserToken          ends

```

Library: **parse.def**

■ ParserTokenCellData

```

ParserTokenCellData  struct
    PTC cellRef       CellReference <>
ParserTokenCellData  ends

```

Library: **parse.def**

■ ParserTokenData

```

ParserTokenData      union
    PTD_number        ParserTokenNumberData
    PTD_string         ParserTokenStringData
    PTD_name           ParserTokenNameData
    PTD_cell           ParserTokenCellData
    PTD_function       ParserTokenFunctionData
    PTD_operator       ParserTokenOperatorData
ParserTokenData      end

```

Library: **parse.def**

Reference book

■ ParserTokenFunctionData

```
ParserTokenFunctionData  struct
    PTFD_functionID      word    ; Identifier for the function
ParserTokenFunctionData  ends
```

Library:

■ ParserTokenNameData

```
ParserTokenNameData      struct
    PTND_name            word    ; The token describing the name
ParserTokenNameData      ends
```

Library: **parse.def**

■ ParserTokenNumberData

```
ParserTokenNumberData    struct
    PTND_value           FloatNum <> ; 8 byte constant
ParserTokenNumberData    ends
```

Library: **parse.def**

■ ParserTokenOperatorData

```
ParserTokenOperatorData  struct
    PTOD_operatorID      OperatorType ; The operator ID.
ParserTokenOperatorData  ends
```

Library: **parse.def**

■ ParserTokenStringData

```
ParserTokenStringData    struct
    PTSD_length          word    ; Length of the string
ParserTokenStringData    ends
```

Library: **parse.def**



■ ParserTokenType

```
ParserTokenType      etype byte, 0, 1
  PARSER_TOKEN_NUMBER      enum ParserTokenType
  PARSER_TOKEN_STRING      enum ParserTokenType
  PARSER_TOKEN_CELL        enum ParserTokenType
  PARSER_TOKEN_END_OF_EXPRESSION  enum ParserTokenType
  PARSER_TOKEN_OPEN_PAREN  enum ParserTokenType
  PARSER_TOKEN_CLOSE_PAREN enum ParserTokenType
  PARSER_TOKEN_NAME        enum ParserTokenType
;
; All the items above are in common with the ScannerTokenType list.
; You can add or delete items below this point without changing
; the other table.
;
  PARSER_TOKEN_FUNCTION    enum ParserTokenType
  PARSER_TOKEN_CLOSE_FUNCTION  enum ParserTokenType
  PARSER_TOKEN_ARG_END      enum ParserTokenType
  PARSER_TOKEN_OPERATOR     enum ParserTokenType
```

Library: **parse.def**

■ PASCFeatures

```
PASCFeatures      record
  PASCF_SPACE_ON_TOP      :1
  PASCF_SPACE_ON_BOTTOM  :1
PASCFeatures      end
```

Library: **Objects/Text/tCtrlC.def**

■ PASCToolboxFeatures

```
PASCToolboxFeatures      record
PASCToolboxFeatures      end
```

Library: **Objects/Text/tCtrlC.def**

■ PathCombineType

```
PathCombineType      etype word
  PCT_NULL            enum PathCombineType; destroy current
  PCT_REPLACE         enum PathCombineType; replace current
  PCT_UNION           enum PathCombineType; add to current
  PCT_INTERSECTION    enum PathCombineType; intersect with curr
```

Library: **graphics.def**

■ PathCompareType

PathCompareType etype byte
 PCT_EQUAL enum PathCompareType
 PCT_SUBDIR enum PathCompareType
 PCT_UNRELATED enum PathCompareType
 PCT_ERROR enum PathCompareType

- PCT_EQUAL The 2 paths are equal.
- PCT_SUBDIR Path 2 is a subdirectory of path 1.
- PCT_UNRELATED Either the 2 paths are unrelated, or path 1 is a subdirectory of path 2.
- PCT_ERROR Some error occurred while parsing one of the paths (i.e., one of the paths was not found, etc.).

Library: **file.def**

■ PatternType

PatternType etype byte
 PT_SOLID enum PatternType
 PT_SYSTEM_HATCH enum PatternType
 PT_SYSTEM_BITMAP enum PatternType
 PT_USER_HATCH enum PatternType
 PT_USER_BITMAP enum PatternType
 PT_CUSTOM_HATCH enum PatternType
 PT_CUSTOM_BITMAP enum PatternType

- PT_SOLID Solid pattern. Passed in AH: Nothing.
- PT_SYSTEM_HATCH System-defined hatch pattern. Passed in AH: SystemHatch.
- PT_SYSTEM_BITMAP System-defined tiled bitmap. Passed in AH: SystemBitmap.
- PT_USER_HATCH User-defined hatch pattern. Passed in AH: 0-255.

PT_USER_BITMAP

User-defined tiled bitmap. Passed in AH: 0-255.

PT_CUSTOM_HATCH

Application-custom hatch pattern. Passed in AH: Nothing.

PT_CUSTOM_BITMAP

Appl-custom tiled bitmap.

Library: **graphics.def**

■ PComInitFlags

```
PComInitFlags record
  PCCIF_NOTIFY_OUTPUT      :1 ; notify caller of output
  PCCIF_NOTIFY_EXIT        :1 ; notify caller of remote exit
                             :14
PComInitFlags end
```

Library: **pccom.def**

■ PComReturnType

```
PComReturnType etype byte
  PCCRT_NO_ERROR            enum PComReturnType
  PCCRT_CANNOT_LOAD_SERIAL_DRIVER enum PComReturnType
  PCCRT_CANNOT_CREATE_THREAD enum PComReturnType
  PCCRT_CANNOT_ALLOC_STREAM enum PComReturnType
  PCCRT_ALREADY_INITIALIZED enum PComReturnType
```

Library: **pccom.def**

■ PDocSizeParams

```
PDocSizeParams struct
  PCDSP_width  dword (?) ; width of the document
  PCDSP_height dword (?) ; height of the document
PDocSizeParams ends
```

Library: **spool.def**

■ PCMarginParams

```
PCMarginParams      struct
    PCMP_left        word (?)          ; left margin
    PCMP_top          word (?)          ; top margin
    PCMP_right        word (?)          ; right margin
    PCMP_bottom       word (?)          ; bottom margin
PCMarginParams      ends
```

Library: **print.def**

■ PCProgressType

```
PCProgressType      etype word, 0, 2
    PCPT_PAGE         enum PCProgressType ; change page number
    PCPT_PERCENT      enum PCProgressType ; change percent done
    PCPT_TEXT         enum PCProgressType ; change text message
```

Library: **spool.def**

■ PenInputDisplayType

```
PenInputDisplayType etype word
    PIDT_KEYBOARD      enum PenInputDisplayType
    PIDT_CHAR_TABLE    enum PenInputDisplayType
    PIDT_CHAR_TABLE_SYMBOLS enum PenInputDisplayType
    PIDT_CHAR_TABLE_INTERNATIONAL enum PenInputDisplayType
    PIDT_CHAR_TABLE_MATH enum PenInputDisplayType
    PIDT_CHAR_TABLE_CUSTOM enum PenInputDisplayType
    PIDT_HWR_ENTRY_AREA enum PenInputDisplayType
```

Library: **Objects/gPenICC.def**

■ PLInit

```
PLInit      struct
    PLI_align      byte
    PLI_message     word
    PLI_point       PointDWFxed
    PLI_instructions PriorityListInstructions
    PLI_maxElements word
    PLI_class        fptr.ClassStruct
PLInit      ends
```

Library: **grobj.def**



■ Point

```
Point      struct
    P_x    sword
    P_y    sword
Point      ends
```

Library: **graphics.def**

■ PointDWordFixed

```
PointDWordFixed      struct
    PDF_x             DWordFixed
    PDF_y             DWordFixed
PointDWordFixed      ends
```

This structure stores a point (in graphic coordinates) where each coordinate is in terms of a **DWordFixed** value.

Library: **graphics.def**

■ PointDWord

```
PointDWord      struct
    PD_x         sdword
    PD_y         sdword
PointDWord      ends
```

This structure stores a point (in graphic coordinates) where each coordinate is in terms of a signed dword value.

Library: **graphics.def**

■ PointerDef

```
PointerDef      struct
    PD_width     PointerDefWidth
    PD_height    byte
    PD_hotX      sbyte
    PD_hotY      sbyte
PointerDef      ends
```

This structure defines a mouse pointer.

PD_width and *PD_height* store the width and height of the cursor, in points.

PD_hotX and *PD_hotY* store the horizontal and vertical offset to the “hot spot” of the pointer. These offsets are relative to the upper left corner of the pointer.

Library: **graphics.def**

■ PointerDefWidth

```
PointerDefWidth record
    PDW_ALWAYS_SHOW_PTR:1
    ; flag, set in BUSY cursors & any other cursors that should be shown
    ; for status's sake, regardless of whether the ptr image is normally
    ; hidden or not (such as in keyboard-only or ink-only systems).
    ; Interpreted, implemented by Input Manager -- Video drivers should
    ; ignore.
    PDW_WIDTH:7
    ; width of cursor
PointerDefWidth end
```

Library: **graphics.def**

■ PointerModes

```
PointerModes          record
    PM_HANDLES_RESIZE          :1
    PM_HANDLES_ROTATE          :1
    PM_POINTER_IS_ACTION_OBJECT :1
PointerModes          end
```

Library: **grobj.def**

■ PointWBFixed

```
PointWBFixed          struct
    PWBf_x              WBFixed
    PWBf_y              WBFixed
PointWBFixed          ends
```

This structure stores a point (in graphic coordinates) where each coordinate is in terms of a **WBFixed** value.

Library: **graphics.def**

■ PointWWFixed

```
PointWWFixed          struct
    PF_x                WWFixed
    PF_y                WWFixed
PointWWFixed          ends
```

This structure stores a point (in graphic coordinates) where each coordinate is in terms of a **WWFixed** value.

Library: **graphics.def**



■ PrefAttributes

```
PrefAttributes      record
    PA_REBOOT_IF_CHANGED      :1
    PA_LOAD_IF_USABLE         :1
    PA_SAVE_IF_USABLE         :1
    PA_SAVE_IF_ENABLED        :1
    PA_SAVE_IF_CHANGED        :1
                                :3
PrefAttributes      end
```

PA_REBOOT_IF_CHANGED

This bit signals that changes in the state of this object, or this object's children require a system reboot to take effect. In a PrefDialogClass object, if this bit is set, then all children will be queried for reboot status (using MSG_PREF_GET_REBOOT_INFO), and if a reboot is required, PrefMgr will be notified.

PA_LOAD_IF_USABLE

Load options only if this object is usable (this is ON by default).

PA_SAVE_IF_USABLE

Save options only if this object is usable (this is ON by default).

PA_SAVE_IF_ENABLED

Save options only if this object is enabled.

PA_SAVE_IF_CHANGED

Save options only if this object has changed.

Library: **config.def**

■ PrefDialogChangeType

```
PrefDialogChangeType      etype word, 0
    PDCT_OPEN              enum PrefDialogChangeType
    PDCT_CLOSE              enum PrefDialogChangeType
    PDCT_DESTROY            enum PrefDialogChangeType
    PDCT_RESTART            enum PrefDialogChangeType
    PDCT_SHUTDOWN           enum PrefDialogChangeType
```

Library: **config.def**

■ PrefEnableData

```
PrefEnableData    struct
    PED_item      word
    PED_lptr      lptr
    PED_flags     PrefEnableFlags
PrefEnableData    ends
```

PED_item stores the identifier of the item that controls enabling/disabling of the object. If the identifier is GIGS_NONE, then the action will be performed if no items are selected.

PED_lptr stores a pointer to the object that is to be enabled or disabled.

Library: **config.def**

■ PrefEnableFlags

```
PrefEnableFlags    record
    PEF_DISABLE_IF_SELECTED    :1
    PEF_DISABLE_IF_NONE       :1
                                :6
PrefEnableFlags    end
```

PEF_DISABLE_IF_SELECTED

If set, disable the object if the associated item is selected, otherwise do the opposite.

PEF_DISABLE_IF_NONE

If this flag is set, then the *PED_item* field is ignored. Instead, the item group will disable the specified object if no items are selected—or if there are no items in the list.

Library: **config.def**

■ PrefInitFileFlags

```
PrefInitFileFlags  record
    PIFF_USE_ITEM_STRINGS      :1
    PIFF_USE_ITEM_MONIKERS     :1
    PIFF_APPEND_TO_KEY         :1
                                :5
PrefInitFileFlags  end
```

PIFF_USE_ITEM_STRINGS

If set, then the item group's children must be of class **PrefStringItemClass**, and their strings will be used to interact with the init file.



PIFF_USE_ITEM_MONIKERS

If set, then the monikers of the items are used to interact with the init file.

PIFF_APPEND_TO_KEY

If set, the strings in this list will be added to strings that may already exist for this key.

Library: **config.def**

■ PrefInteractionAttrs

```
PrefInteractionAttrs record
  PIA_LOAD_OPTIONS_ON_INITIATE:1
  PIA_SAVE_OPTIONS_ON_APPLY:1
:6
PrefInteractionAttrs      end
```

PIA_LOAD_OPTIONS_ON_INITIATE

If set, then the dialog will send MSG_PREF_INIT, followed by MSG_META_LOAD_OPTIONS to itself when it receives a MSG_GEN_INTERACTION_INITIATE.

PIA_SAVE_OPTIONS_ON_APPLY

This flag is normally OFF, to allow non-dialog prefInteraction objects to reside inside other interactions without duplicate SAVE_OPTIONS messages being sent. This flag is normally on for objects of **PrefDialogClass**.

Library: **config.def**

■ PrefItemGroupStringVars

```
PrefItemGroupStringVars struct
  PIGSV_endPtr      nptr.char
  PIGSV_selections  word  PREF_ITEM_GROUP_MAX_SELECTIONS dup (?)
  PIGSV_numSelections word
  PIGSV_buffer      char  PREF_ITEM_GROUP_STRING_BUFFER_SIZE dup (?)
```

Library: **config.def**

■ PrefMgrFeatures

```
PrefMgrFeatures      record
    PMF_HARDWARE      :1
    PMF_SYSTEM         :1
    PMF_NETWORK        :1
    PMF_USER           :1
                      :12
PrefMgrFeatures      end
```

PMF_HARDWARE

These settings are for a user who has permissions to actually change the configuration of the workstation. In a network environment where users log in to different machines at different times, normal users would be prevented from changing the mouse & video drivers, etc.

PMF_SYSTEM

These changes are more complex & potentially more damaging than the basic “user” changes, therefore, some users may be prevented from using these settings.

PMF_NETWORK

Network settings -- generally only the system administrator will see these settings.

PMF_USER

These are basic user changes -- normally this flag works the opposite way of the other flags -- by default all UI is “on”, but if the user flag is off, then some basic UI might go away. This would allow a network administrator, for example, to look at a scaled-down, “network-only” prefmgr.

Library: **config.def**



■ PrefModuleEntryType

```

PrefModuleEntryType      etype word, 0, 1
    PMET_FETCH_UI        enum PrefModuleEntryType
    ;
    ; Return the OD of the top-most object in the UI tree.
    ; All UI *must* be in the same segment as this object. The UI tree
    ; will be duplicated by the application, and added to the app's
    ; generic tree.
    ;
    ; Pass: nothing
    ; Return: ^ldx:ax - OD of root of UI tree.
    ; Destroy: nothing
    ;
    PMET_GET_MODULE_INFO  enum PrefModuleEntryType
    ; Return information about this module that will be used to determine
    ; whether to display it on-screen, what it will look like, etc.
    ;
    ; Pass: ds:si - pointer to a PrefModuleInfo buffer to be filled in
    ; Return: -- buffer filled in
    ; Destroy: ax, bx
    ;

```

Entry points for the Preferences Modules. Each module (library) must have the following routines exported in its .gp file, in the following order, as the first routines exported from that library.

Library: **config.def**

■ PrefModuleInfo

```
PrefModuleInfo struct
    PMI_requiredFeatures      PrefMgrFeatures <>
    ; Features that MUST be set for this module to appear in PrefMgr
    PMI_prohibitedFeatures    PrefMgrFeatures <>
    ; Features that MUST NOT be set for this module to appear.
    PMI_minLevel              UIInterfaceLevel      0
    ; Minimum user level required for this module to appear.
    ; Currently not implemented by PrefMgr -- should be set to zero by
    ; module.
    PMI_maxLevel              UIInterfaceLevel      UIInterfaceLevel
    ; Minimum user level required for this module to appear.
    ; Currently not implemented by PrefMgr. Should be set to
    ; the maximum value by module.
    PMI_monikerList           optr
    ; Moniker list in a shared, lmem, read-only resource that will be
    ; used as the module's trigger
    PMI_monikerToken          GeodeToken
    ; A unique GeodeToken that will be used to enter the module's
    ; moniker list into the token database. This is done so that the
    ; module is only called with PMET_GET_MODULE_INFO the first time
    ; it is encountered by PrefMgr -- all subsequent times, the
    ; necessary information is cached.
PrefModuleInfo ends
```

Library: **config.def**

■ PrefTimeDateControlFeatures

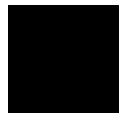
```
PrefTimeDateControlFeatures      record
    PTDCF_DATE:1
    PTDCF_TIME:1
PrefTimeDateControlFeatures      end
```

Library: **config.def**

■ PrefTimeDateControlScreen

```
PrefTimeDateControlScreen        etype    word, 0, 2
    PTDCS_TIME_DATE              enum      PrefTimeDateControlScreen
```

Library: **config.def**



■ PrefTocExtraEntry

```
PrefTocExtraEntry  struct
    PTEE_item      lptr.char
    PTEE_driver    lptr.char
    PTEE_info      word
PrefTocExtraEntry  ends
```

PTEE_item stores the lptr of the item name. For device lists, this is the device. For others, this is the filename.

PTEE_driver stores the driver name (for device lists only).

PTEE_info stores an extra word of information.

Library: **config.def**

■ PrefTriggerAction

```
PrefTriggerAction      struct
    PTA_message         word
    PTA_dest            optr
PrefTriggerAction      ends
```

Library: **config.def**

■ PrefVMMapBlock

```
PrefVMMapBlock struct
    PVMMB_root fptr
PrefVMMapBlock ends
```

Map block for a VM file this object can handle.

PVMMB_root is the root of the object tree contained in the VM file. Segment portion is VM block handle

Library: **config.def**

■ PrintControlAttrs

```
PrintControlAttrs    record
    :1
    PCA_SEE_IF_DOC_WILL_FIT    :1    ; check if document will fit on paper
    PCA_MARK_APP_BUSY          :1    ; mark busy while application is printing
    PCA_VERIFY_PRINT           :1    ; indicate we want to verify before
                                ; printing
    PCA_SHOW_PROGRESS          :1    ; show the print progress dialog box
    PCA_PROGRESS_PERCENT       :1    ; show progress by percentage completed
    PCA_PROGRESS_PAGE          :1    ; show progress by page completed
    PCA_FORCE_ROTATION         :1    ; Force rotation of output
    PCA_COPY_CONTROLS          :1    ; Copy controls are available
    PCA_PAGE_CONTROLS          :1    ; Page range controls are available
    PCA_QUALITY_CONTROLS       :1    ; Print quality controls are available
    PCA_USES_DIALOG_BOX        :1    ; A print dialog box should appear
    PCA_GRAPHICS_MODE          :1    ; Supports graphics mode output
    PCA_TEXT_MODE              :1    ; Supports text mode output
    PCA_DEFAULT_QUALITY        PrintQualityEnum:2 ; default print quality
PrintControlAttrs    end
```

Library: **spool.def**

■ PrintControlFeatures

```
PrintControlFeatures    record
    PRINTCF_PRINT_TRIGGER      :1    ; wants a print trigger
    PRINTCF_FAX_TRIGGER        :1    ; wants a fax trigger
PrintControlFeatures    end
```

Library: **spool.def**

■ PrintControlStatus

```
PrintControlStatus    etype word
    PCS_PRINT_BOX_VISIBLE      enum PrintControlStatus; Print DB is on screen
    PCS_PRINT_BOX_NOT_VISIBLE  enum PrintControlStatus; Print DB not on screen
```

Library: **spool.def**



■ PrintControlToolboxFeatures

```
PrintControlToolboxFeatures  record
    PRINTCTF_PRINT_TRIGGER    :1      ; wants a print tool trigger
    PRINTCTF_FAX_TRIGGER      :1      ; wants a fax tool trigger
PrintControlToolboxFeatures  end
```

Library: **spool.def**

■ PrinterDriverType

```
PrinterDriverType  etype byte, 0
    PDT_PRINTER      enum PrinterDriverType
    PDT_PLOTTER       enum PrinterDriverType
    PDT_FACSIMILE     enum PrinterDriverType
    PDT_CAMERA        enum PrinterDriverType
    PDT_OTHER         enum PrinterDriverType
    PDT_ALL           equ -1          ; all printers of all types
```

Library: **print.def**

■ PrinterMode

```
PrinterMode  etype byte, 0, 2
    PM_GRAPHICS_LOW_RES      enum PrinterMode ; lowest quality...fastest...
    PM_GRAPHICS_MED_RES     enum PrinterMode ; medium quality...slower...
    PM_GRAPHICS_HI_RES      enum PrinterMode ; best quality...slowest...
    PM_TEXT_DRAFT           enum PrinterMode ; fastest ascii output
    PM_TEXT_NLQ            enum PrinterMode ; best quality ascii output

    PM_FIRST_TEXT_MODE      equ PM_TEXT_DRAFT; equate to make the code easier
```

Library: **print.def**

■ PrinterOutputModes

```
PrinterOutputModes record
    POM_unused          :3      ; leave these bits alone!!!
    POM_GRAPHICS_LOW    :1      ; Graphics mode low quality available
    POM_GRAPHICS_MEDIUM :1      ; Graphics mode medium quality available
    POM_GRAPHICS_HIGH   :1      ; Graphics mode high quality available
    POM_TEXT_DRAFT      :1      ; Character mode draft quality available
    POM_TEXT_NLQ        :1      ; Character mode NLQ quality available
PrinterOutputModes end
```

Library: **spool.def**

■ PrintQualityEnum

```
PrintQualityenum etype byte, 0, 1
    PQT_HIGH      enum PrintQualityenum ; default to high
    PQT_MEDIUM    enum PrintQualityenum ; default to medium
    PQT_LOW       enum PrintQualityenum ; default to low
```

Library: **spool.def**

■ PrintStatusFlags

```
PrintStatusFlags record
    PSF_FAX_AVAILABLE :1      ; set if a fax driver is available
                                :3
    PSF_ABORT         :1      ; user wants to abort printing
    PSF_RECEIVED_COMPLETED :1 ; MSG_PC_PRINTING_COMPLETED received
    PSF_RECEIVED_NAME  :1      ; MSG_PC_SET_DOC_NAME received
    PSF_VERIFIED       :1      ; PSG_PC_VERIFY_? received
PrintStatusFlags end
```

Library: **spool.def**



■ PriorityList

```
PriorityList      struct
  PL_message      word
  PL_point         PointDWFxed
  PL_instructions  PriorityListInstructions
  PL_class         fptr.ClassStruct
  PL_numElements  word
  PL_maxElements  word
  PL_list         lptr
PriorityList      ends
```

PL_message stores the method to send to each object.

PL_point stores the document coordinates to be examined by the object.

PL_instructions stores the instructions for the processing of objects.

PL_class stores the class of object to be used (based on the specific instructions passed in *PL_instructions*).

PL_numElements stores the number of elements currently in the list.

PL_maxElements stores the maximum number of elements allowed in the list.

PL_list stores the chunk handle of the list chunk array.

Library: **grobj.def**

■ PriorityListElement

```
PriorityListElement  struct
  PLE_od             optr
  PLE_priority       byte
  PLE_other          byte
PriorityListElement  ends
```

Library: **grobj.def**

■ PriorityListInstructions

```
PriorityListInstructions record
    PLI_CHECK_SELECTION_HANDLE_BOUNDS      :1
    PLI_ONLY_PROCESS_SELECTED              :1
    PLI_ONLY_PROCESS_CLASS                  :1
    PLI_ONLY_INSERT_CLASS                   :1
    PLI_STOP_AT_FIRST_HIGH                  :1
    PLI_ONLY_INSERT_HIGH                    :1
    PLI_DONT_INSERT_OBJECTS_WITH_SELECTION_LOCK :1
PriorityListInstructions end
```

PLI_CHECK_SELECTION_HANDLE_BOUNDS

Do trivial reject check on bounds of objects that include the selection handles, instead of the normal *parent* bounds which are only guaranteed to surround the image.

PLI_ONLY_PROCESS_SELECTED

Only send messages to objects that are in selection list.

PLI_ONLY_PROCESS_CLASS

Only send messages to objects of class.

PLI_ONLY_INSERT_CLASS

Only insert items of class into priority list.

PLI_STOP_AT_FIRST_HIGH

Stop processing children after an object evaluates high.

PLI_ONLY_INSERT_HIGH

Only insert objects that evaluate high.

PLI_DONT_INSERT_OBJECTS_WITH_SELECTION_LOCK

Don't insert objects that have their selection lock set. These objects must return EPN_SELECTION_LOCK_SET.

Library: **grobj.def**



■ ProcessCallRoutineParams

```

ProcessCallRoutineParams      struct
    PCRP_address              fptr
    PCRP_dataAX               word
    PCRP_dataBX               word
    PCRP_dataCX               word
    PCRP_dataDX               word
    PCRP_dataSI               word
    PCRP_dataDI               word
ProcessCallRoutineParams      ends

```

Library: **processC.def**

■ ProtocolNumber

```

ProtocolNumber                struct
    PN_major                  word
    PN_minor                  word
ProtocolNumber                ends

```

This structure defines the protocol level of a file, geode, or document.

PN_major represents the most significant compatibility comparisons. If the major protocol is different between two items, they are incompatible.

PN_minor represents less significant differences. If minor protocols are different, the two items may or may not be compatible.

Library: **geode.def**

■ PSCFeatures

```

PSCFeatures                   record
    PSCF_9                    :1
    PSCF_10                   :1
    PSCF_12                   :1
    PSCF_14                   :1
    PSCF_18                   :1
    PSCF_24                   :1
    PSCF_36                   :1
    PSCF_54                   :1

```

```

PSCF_72          :1
PSCF_SMALLER     :1
PSCF_LARGER      :1
PSCF_CUSTOM_SIZE:1
PSCFeatures      end

```

Library: **Objects/Text/tCtrlC.def**

■ PSCToolboxFeatures

```

PSCToolboxFeatures record
  PSCTF_9          :1
  PSCTF_10         :1
  PSCTF_12         :1
  PSCTF_14         :1
  PSCTF_18         :1
  PSCTF_24         :1
  PSCTF_36         :1
  PSCTF_54         :1
  PSCTF_72         :1
  PSCTF_SMALLER    :1
  PSCTF_LARGER     :1
PSCToolboxFeatures end

```

Library: **Objects/Text/tCtrlC.def**

■ PtrImageLevel

```

PtrImageLevel      etype word, 0
  PIL_SYSTEM        enum PtrImageLevel
  PIL_1             enum PtrImageLevel
  PIL_FLOW          enum PtrImageLevel
  PIL_3             enum PtrImageLevel
  PIL_GEODE         enum PtrImageLevel
  PIL_5             enum PtrImageLevel
  PIL_GADGET        enum PtrImageLevel
  PIL_7             enum PtrImageLevel
  PIL_WINDOW        enum PtrImageLevel
  PIL_DEFAULT       enum PtrImageLevel

```

Enumerated type for ptr image level. Each level may specify a particular mouse ptr image, or specify that the level does not currently have an image. The first (starting at lowest enumerated value) level which has a ptr image declared will be used at any given moment in time.

PIL_SYSTEM



PIL_1	
PIL_FLOW	Used for UI quick transfer move/copy cursors.
PIL_3	
PIL_GEODE	Overriding status of geode. Used to indicate busy state, modal state for applications. Applications and the UI should use WinSetGeodePtrImage to set the image.
PIL_5	
PIL_GADGET	Gadget level. Used for visible gadgets within windows, such as text object, control points, etc. Applications and the UI should use WinSetGeodePtrImage to set the image.
PIL_7	
PIL_WINDOW	Window level. Background cursor to use. Applications and the UI should use WinSetGeodePtrImage to set the image.
PIL_DEFAULT	Default pointer.

Library: **win.def**

■ PtrImageValue

```
PtrImageValue      etype word
    PIV_NONE          enum PtrImageValue, 0
    PIV_VIDEO_DRIVER_DEFAULT  enum PtrImageValue
    PIV_UPDATE        enum PtrImageValue
```

Enumerated type passed as an alternative to an optr to a PointerDef.

PIV_NONE
No ptr image requested for level. Important: Keep the value of this enumerated constant to 0, so that a null optr results in no ptr image being requested.

PIV_VIDEO_DRIVER_DEFAULT
Use video driver default ptr image.

PIV_UPDATE
Re-write current highest priority ptr image to the video driver. Useful when changing PointerDef at previous optr, changing screens.

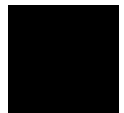
Library: **win.def**

Reference book

■ QuickSortParameters

```
QuickSortParameters      struct
    QSP_compareCallback  fptr    ; Compare two elements
    ;
    ;      Pass:
    ;          ds:si      - first array element
    ;          es:di      - second array element
    ;          bx         - parameter passed to ArrayQuickSort
    ;      Return:
    ;          flags set so caller can jl, je, or jg
    ;          according as first element is less than,
    ;          equal to, or greater than the second.
    ;      Destroyed:
    ;          ax, bx, cx, dx, di, si
    ;
    QSP_lockCallback      fptr    ; Lock an element (segment = 0, for none)
    ;
    ;      Pass:
    ;          ds:si      - array element to lock
    ;          bx         - parameter passed to ArrayQuickSort
    ;      Return:
    ;          nothing
    ;      Destroyed:
    ;          nothing
    ;
    QSP_unlockCallback    fptr    ; Unlock an element (segment = 0, for none)
    ;
    ;      Pass:
    ;          ds:si      - array element to unlock
    ;          bx         - parameter passed to ArrayQuickSort
    ;      Return:
    ;          nothing
    ;      Destroyed:
    ;          nothing
    ;
    QSP_insertLimit       word
    QSP_medianLimit       word
    ;
    ; These are used internally by the quicksort algorithm and should not
    ; be set by the caller.
    ;
    QSP_nLesser           word
    QSP_nGreater          word
    align                 word
QuickSortParameters      ends
```

QSP_insertLimit stores the size of the list below which we choose to use insertion sort and not quicksort.



QSP_medianLimit stores the size of list below which we no longer choose the median but instead use the first key.

QSP_nLesser and *QSP_nGreater* store the number of elements in the lesser and greater list parts. This is used internally by the quicksort algorithm.

Library: **chunkarr.def**

■ QuitLevel

```
QuitLevel          etype word
QL_BEFORE_UI       enum QuitLevel
QL_UI              enum QuitLevel
QL_AFTER_UI        enum QuitLevel
QL_DETACH          enum QuitLevel
QL_AFTER_DETACH    enum QuitLevel
```

QL_BEFORE_UI

Quit message sent out before MSG_META_QUIT messages are sent to the items on the active list.

QL_UI

Default handler for this method sends MSG_META_QUIT to the application object for the process.

QL_AFTER_UI

Quit message sent out before MSG_META_DETACH is sent to the process.

QL_DETACH

Default handler for this method sends MSG_META_DETACH to the process.

QL_AFTER_DETACH

Default handler for this method doesn't really do anything. Why is it still around? You'll understand when you're older.

Library: **Objects/metaC.def**

■ RandomGenInitFlags

```
RandomGenInitFlags record
  RGIF_USE_SEED      :1
  RGIF_GENERATE_SEED :1
                    :6
RandomGenInitFlags end
```

Library: **math.def**

RangeEnumFlags

```

RangeEnumFlags      record
    REF_ALL_CELLS      :1      ; Set: callback for all cells.
    REF_NO_LOCK        :1      ; Set: callback will lock/unlock cells.
    REF_ROW_FLAGS      :1      ; Set: get row flags when calling back
    REF_MATCH_ROW_FLAGS :1      ; Set: callback for cells w/matching row
                                ; flags
    ;
    ; These next ones are returned from the callback routine.
    ;
    REF_CELL_ALLOCATED :1      ; Set: The callback routine allocated the
                                ; cell for which the callback occurred.
    REF_CELL_FREED     :1      ; Set: The callback routine freed the cell
                                ; for which the callback occurred.
    REF_OTHER_ALLOC_OR_FREE :1 ; Set: The callback routine may have
                                ; allocated or freed a cell other than the
                                ; one for which the callback occurred.
    REF_ROW_FLAGS_MODIFIED :1 ; Set: The callback routine changed
                                ; the row flags
RangeEnumFlags      end

```

Library: **cell.def**

RangeEnumParams

```

RangeEnumParams      struct
    REP_callback      fptr.far      ;routine to call
    REP_bounds        Rectangle      ;cells to enumerate
    ;
    ; Stuff below this point is filled in by RangeEnum() and can be
    ; used by the callback.
    ;
    REP_rowFlags      word            ;current row flags
    ;
    ; Stuff below this point is setup by RangeEnum(). Applications don't
    ; need to worry about it.
    ;
    REP_cfp           fptr.CellFunctionParameters
    ;
    ; Stuff below this point is only used with REF_MATCH_ROW_FLAGS
    ;
    REP_matchFlags     word            ;flags to match
    REP_flagRow        word            ;row we locked for flags
RangeEnumParams      ends

```

This structure is used most often by the **RangeEnum** routine. In this case, the structure defines how the enumeration will occur.



Occasionally, however, an empty **RangeEnumParams** structure may be passed along with the **CellGetExtent** routine. In that case, the routine will fill in the *REP_bounds* entry.

REP_callback stores the routine to perform the enumeration.

REP_bounds stores the range of the cells to perform the enumeration on.

REP_rowFlags stores the “row flags” associated with a specific cell. During the enumeration, this entry holds the row flags of the most recently processed cell. These flags (4 bits in total) are able to provide custom information about cells.

REP_cfp and *REP_matchFlags* are set up automatically by **RangeEnum**. (*REP_cfp* stores the address of the **CellFunctionParameters** structure.)

Library: **cell.def**

■ RangeInsertParams

```
RangeInsertParams  struct
    RIP_bounds      Rectangle
    RIP_delta       Point
    RIP_cfp         dword
RangeInsertParams  ends
```

This structure is used by the **RangeInsert** routine, which shifts a group of cells from one position to another.

RIP_bounds stores the range of cells to shift.

RIP_delta stores the Point value which specifies the horizontal and vertical distance to shift the group of cells.

RIP_cfp stores the address of the **CellFunctionParameters** structure; this entry is set up automatically by the routine.

Library: **cell.def**

■ RangeSortCellExistsFlags

```
RangeSortCellExistsFlags  record
                                :6      ; Unused bits
    RSCEF_SECOND_CELL_EXISTS  :1      ; Set: Second cell exists
    RSCEF_FIRST_CELL_EXISTS   :1      ; Set: First cell exists
RangeSortCellExistsFlags  end
```

Library: **cell.def**

■ RangeSortError

```
RangeSortError      etype word, 0, 1
    RSE_NO_ERROR      enum RangeSortError
    RSE_UNABLE_TO_ALLOC  enum RangeSortError

    RSE_NO_ERROR
        No error, the sort was successful.

    RSE_UNABLE_TO_ALLOC
        The sorting code was unable to allocate the temporary block it
        needs to sort the data.
```

Library: **cell.def**

■ RangeSortFlags

```
RangeSortFlags      record
    RSF_SORT_ROWS      :1 ; Set: Sort rows
    RSF_SORT_ASCENDING  :1 ; Set: Sort in ascending order
    RSF_IGNORE_CASE     :1 ; Set: Ignore case in string compares
                        :4
    RSF_IGNORE_SPACES   :1 ; Set: Ignore spaces & punctuation
                        ; NOTE: this is not supported directly
                        ; by the Cell library, but is placed
                        ; here for the convenience of apps.

RangeSortFlags      end
```

Library: **cell.def**



■ RangeSortParams

```

RangeSortParams    struct
    RSP_range        Rectangle
    RSP_active        Point
    RSP_callback      dword    ; Comparison routine
    ;
    ; Callback is defined as:
    ;     PASS:    ds:si    = Pointer to first cells data
    ;             es:di    = Pointer to second cells data
    ;             ax       = RangeSortCellExistsFlags
    ;             ss:bx    = Parameters passed to RangeSort
    ;     RETURN:  Flags set for comparison of the two cells
    ;     DESTROYED: cx, dx, di, si, bp
    ;
    RSP_flags        RangeSortFlags
    align            word
    ;
    ; The rest is used in RangeSort and should be ignored.
    ;
    RSP_cfp          dword
    RSP_sourceChunk   word
    RSP_destChunk     word
    RSP_base          word
    RSP_lockedEntry   dword
    RSP_cachedFlags   byte
    align            word
RangeSortParams    ends

```

RSP_range stores the rectangular range to conduct the sort on.

RSP_active stores the cell in the row/column that the current sort is acting upon.

RSP_flags stores the **RangeSortFlags** used in the sorting operation.

RSP_cfp stores the **CellFunctionParameters**.

RSP_sourceChunk stores the source row chunk for swapping.

RSP_destChunk stores the destination chunk for swapping.

RSP_base stores the base row/column for sort block.

RSP_lockedEntry stores the entry that is currently locked. If the segment is -1, nothing is locked.

RSP_cachedFlags stores the flags (carry set if cell exists).

Library: **cell.def**

■ RecalcSizeArgs

```
RecalcSizeArgs      record
    RSA_CHOOSE_OWN_SIZE      :1
    RSA_SUGGESTED_SIZE      :15
RecalcSizeArgs      end
```

RSA_CHOOSE_OWN_SIZE

Geometry manager wants object to choose its own size, typically objects will use current size if any, or choose an initial size, is coming up for the first time.

RSA_SUGGESTED_SIZE

Suggested size to use, if above bit is not set.

Library: **Objects/visC.def**

■ Rectangle

```
Rectangle      struct
    R_left      sword
    R_top       sword
    R_right     sword
    R_bottom    sword
Rectangle      ends
```

This is the standard structure for a rectangle in GEOS.

Library: **graphics.def**

■ RectDWFixed

```
RectDWFixed      struct
    RDWF_left     DWFixed
    RDWF_top      DWFixed
    RDWF_right    DWFixed
    RDWF_bottom   DWFixed
RectDWFixed      ends
```

Library: **bitmap.def**



■ RectDWord

```
RectDWord      struct
    RD_left    sdword
    RD_top     sdword
    RD_right   sdword
    RD_bottom  sdword
RectDWord      ends
```

This is the standard structure for a rectangle in extended coordinates.

Library: **graphics.def**

■ RectRegion

```
RectRegion      struct
    RR_y1M1    word
    RR_eo1     word EOREGREC
    RR_y2      word
    RR_x1      word
    RR_x2      word
    RR_eo2     word EOREGREC
    RR_eo3     word EOREGREC
RectRegion      ends
```

This structure stores a rectangular region.

Library: **graphics.def**

■ RectWWFixed

```
RectWWFixed      struct
    RWWF_left  WWFixed
    RWWF_top   WWFixed
    RWWF_right WWFixed
    RWWF_bottom WWFixed
RectWWFixed      ends
```

Library: **grobj.def**

■ RefElementHeader

```
RefElementHeader struct
    REH_refCount WordAndAHalf
RefElementHeader ends
```

This structure defines a header for an element of a fixed size.

Library: **chunkarr.def**

■ Region

```
Region    struct
    R_data    word
Region    ends
EOREGREC    =    8000h
```

This structure stores a region of a graphics coordinate space. A region is of arbitrary length and consists of a series of word-length values.

Regions are described in terms of a rectangular array (thus the similarity to bitmaps). Instead of specifying an on/off value for each pixel, however, regions assume that the area will be fairly undetailed and that the data structure can thus be treated in the manner of a sparse array. Only the cells in which the color value of a row changes are recorded. The tricky part here is keeping in mind that when figuring out whether or not a row is the same as a previous row; the system works its way up from the bottom, so that you should compare each row with the row beneath it to determine whether it needs an entry.

The easiest region to describe is the null region, which is a special case described by a single word with the value EOREGREC (which stands for *End Of REGION RECO*rd value). Describing a non-null region requires several numbers.

The first four words of the region (starting with R_data) give the (rectangular) bounds of the region. What follows is a series of one or more numbers. Each word-length entry describes a row, specifying which pixels of that row are part of the region. The only rows which need to be described are those which are different

Library: **graphics.def**

■ RegionFillRule

```
RegionFillRule    etype byte
RFR_ODD_EVEN    enum RegionFillRule ; 0 = odd-even rule
RFR_WINDING    enum RegionFillRule ; 1 = winding rule
```

Library: **graphics.def**



■ ReleaseNumber

```
ReleaseNumber      struct
  RN_major          word
  RN_minor          word
  RN_change         word
  RN_engineering    word
ReleaseNumber      ends
```

This structure stores the release number (version) of a file, document or geode.

RN_major stores the major release number. Different major release numbers signify a considerable version difference.

RN_minor stores the minor release version. Different minor release numbers typically mean a minor version “tweaking” of the system.

RN_change and *RN_engineering* are internal release numbers that should not need to be examined or changed.

Library: **geode.def**

■ ReplaceAllFromOffsetFlags

```
ReplaceAllFromOffsetFlags  record
  RAFOF_CONTINUING_REPLACE  :1
  RAFOF_HAS_UNDO            :1
                           :14
ReplaceAllFromOffsetFlags  end
```

RAFOF_CONTINUING_REPLACE

Set if this message was generated to continue a replace all sent to a different object.

RAFOF_HAS_UNDO

Set if this action will be undoable - mainly used for error-checking, to ensure that all objects involved in a ReplaceAll are undoable.

Library: **Objects/vTextC.def**

■ ReplaceAllFromOffsetStruct

```
ReplaceAllFromOffsetStruct    struct
    RAFOS_data                hptr.SearchReplaceStruct
    RAFOS_startOffset         dword
    RAFOS_flags                ReplaceAllFromOffsetFlags
ReplaceAllFromOffsetStruct    ends
```

RAFOS_data stores a pointer to the replace-all data. The data is organized in the following format:

```
SearchReplaceStruct<>
    data                Null-Terminated Search string
    data                Null-Terminated Replace string
```

Note: The sender of this message is responsible for freeing the passed **SearchReplaceStruct**.

RAFOS_startOffset stores the offset in this object to begin the replace all operation.

RAFOS_flags stores flags that are set internally by the text object; do not set them.

Library: **Objects/vTextC.def**

■ ReplaceAllInRangeStruct

```
ReplaceAllInRangeStruct    struct
    RAIRS_data                hptr.SearchReplaceStruct
    RAIRS_range                VisTextRange <>
ReplaceAllInRangeStruct    ends
```

RAIRS_data stores a pointer to the replace-all data. The data is organized in the following format:

```
SearchReplaceStruct<>
    data                Null-Terminated Search string
    data                Null-Terminated Replace string
```

Note: The sender of this message is responsible for freeing the passed **SearchReplaceStruct**.

Library: **Objects/vTextC.def**



■ ReplaceItemMonikerFlags

```
ReplaceItemMonikerFlags  record
    RIMF_NOT_ENABLED      :1
                           :15
ReplaceItemMonikerFlags  end
```

RIMF_NOT_ENABLED

Ensure that the item we are setting the moniker for will be disabled while visible.

Library: **Objects/gDListC.def**

■ ReplaceItemMonikerFrame

```
ReplaceItemMonikerFrame  struct
    RIMF_source            dword
    RIMF_sourceType        VisMonikerSourceType
    even
    RIMF_dataType          VisMonikerDataType
    even
    RIMF_length            word
    RIMF_width             word
    RIMF_height            word
    RIMF_itemFlags         ReplaceItemMonikerFlags
    RIMF_item              word
ReplaceItemMonikerFrame  ends
```

RIMF_source stores a pointer to the source moniker. This pointer may be an optr, hpctr, or fpctr, depending on the RIMF_sourceType.

RIMF_sourceType defines the type of pointer contained in RIMF_source.

RIMF_dataType defines whether the source is a VisMoniker, text string, graphics string, or GeodeToken.

RIMF_length stores the size (in bytes) of the source. (This value is not used for VMST_OPTR.) If the data type is VMDT_TEXT and *RIMF_length* is 0, the text is assumed to be null-terminated. If the data type is VMDT_GSTRING and *RIMF_length* is 0, the length of the gstring is computed by scanning the gstring.

RIMF_width stores the width of the graphics string if the data type is VMDT_GSTRING. If 0, the width of the gstring is computed by scanning the gstring.

RIMF_height stores the height of the graphics string if the data type is VMDT_GSTRING. If 0, the height of the gstring is computed by scanning the gstring.

RIMF_itemFlags stores the item flags for the function.

RIMF_item stores the position of the item whose moniker you're setting.

Library: **Objects/gDListC.def**

■ ReplaceVisMonikerFrame

```
ReplaceVisMonikerFrame    struct
    RVMF_source            dword
    RVMF_sourceType        VisMonikerSourceType
    even
    RVMF_dataType          VisMonikerDataType
    even
    RVMF_length            word
    RVMF_width             word
    RVMF_height            word
    RVMF_updateMode        VisUpdateMode
    even
ReplaceVisMonikerFrame    ends
```

This structure is used by MSG_GEN_REPLACE_VIS_MONIKER to store the parameters used in that operation.

RVMF_source stores a pointer to the source moniker. This pointer may be an optr, hptr, or fptr, depending on the *RVMF_sourceType*.

RVMF_sourceType defines the type of pointer contained in *RVMF_source*.

RVMF_dataType defines whether the source is a VisMoniker, text string, graphics string, or GeodeToken.

RVMF_length stores the size (in bytes) of the source. (This value is not used for VMST_OPTR.) If the data type is VMDT_TEXT and *RVMF_length* is 0, the text is assumed to be null-terminated. If the data type is VMDT_GSTRING and *RVMF_length* is 0, the length of the gstring is computed by scanning the gstring.

RVMF_width stores the width of the graphics string if the data type is VMDT_GSTRING. If 0, the width of the gstring is computed by scanning the gstring.

RVMF_height stores the height of the graphics string if the data type is VMDT_GSTRING. If 0, the height of the gstring is computed by scanning the gstring.

RVMF_updateMode stores the visual update mode to define when to show the new moniker.

Library: **Objects/genC.def**



■ **ReplaceWithGraphicParams**

```
ReplaceWithGraphicParams struct
  RWGP_range      VisTextRange      ;range to replace
  RWGP_pasteFrame word              ;ptr to frame if quick paste.
                                      ;0 otherwise.
  RWGP_sourceFile word              ;source vm file
  RWGP_graphic    VisTextGraphic
ReplaceWithGraphicParams ends
```

Library: **Objects/vTextC.def**

■ **ReplaceWithHWRData**

```
ReplaceWithHWRData struct
  RWHWRD_range  VisTextRange <>
  RWHWRD_context HWRContext <>
ReplaceWithHWRData ends
```

RWHWRD_range stores the range to replace. This range may start beyond the end of the text. If so, spaces should be appended to the text.

RWHWRD_context stores the context in which the hand writing recognition data was added.

Library: **Objects/gPenICC.def**

■ **RequestedViewArea**

```
RequestedViewArea  etype byte
  RVA_NO_AREA_CHOICE      enum RequestedViewArea ;no choice made here
  RVA_X_SCROLLER_AREA     enum RequestedViewArea ;put with x scroller
  RVA_Y_SCROLLER_AREA     enum RequestedViewArea ;put with y scroller
  RVA_LEFT_AREA           enum RequestedViewArea ;put in left area
  RVA_TOP_AREA            enum RequestedViewArea ;put in top area
  RVA_RIGHT_AREA          enum RequestedViewArea ;put in right area
  RVA_BOTTOM_AREA         enum RequestedViewArea ;put in bottom area
```

Library: **Objects/genC.def**

■ RGBDelta

```

RGBDelta          struct
  RGBD_red        sbyte
  RGBD_green      sbyte
  RGBD_blue       sbyte
RGBDelta          ends

```

Library: **color.def**

■ RGBValue

```

RGBValue          struct
  RGB_red         byte
  RGB_green       byte
  RGB_blue        byte
RGBValue          ends

```

Library: **color.def**

■ RGCFeatures

```

RGCFeatures       record
  RGCF_GRID_SPACING      :1
  RGCF_SNAP_TO_GRID      :1
  RGCF_SHOW_GRID         :1
RGCFeatures       end

```

Library: **ruler.def**

■ RowBlockList

```

RowBlockList      struct
  RBL_blocks       nptr N_ROW_BLOCKS dup (0)
RowBlockList      ends

```

Library: **cell.def**



■ RSCCFeatures

```
RSCCFeatures      record
  RSCCF_SHOW_VERTICAL      :1
  RSCCF_SHOW_HORIZONTAL    :1
  RSCCF_SHOW_RULERS        :1
RSCCFeatures      end
```

Library: **ruler.def**

■ RSCCToolboxFeatures

```
RSCCToolboxFeatures      record
RSCCToolboxFeatures      end
```

Library: **ruler.def**

■ RTCFeatures

```
RTCFeatures      record
  RTCF_DEFAULT      :1
  RTCF_SPREADSHEET :1
  RTCF_INCHES       :1
  RTCF_CENTIMETERS :1
  RTCF_POINTS       :1
  RTCF_PICAS        :1
RTCFeatures      end
```

Library: **ruler.def**

■ RTCToolboxFeatures

```
RTCToolboxFeatures record
  RTCTF_DEFAULT      :1
  RTCTF_SPREADSHEET :1
  RTCTF_INCHES       :1
  RTCTF_CENTIMETERS :1
  RTCTF_POINTS       :1
  RTCTF_PICAS        :1
RTCToolboxFeatures end
```

Library: **ruler.def**

■ RulerGridNotificationBlock

```
RulerGridNotificationBlock    struct
  RGNB_gridSpacing            WWFixed
  RGNB_gridOptions            GridOptions
RulerGridNotificationBlock    ends
```

Library: **ruler.def**

■ RulerGuideControlFeatures

```
RulerGuideControlFeatures    record
  RGCF_HV                     :1
  RGCF_LIST                    :1
  RGCF_POSITION                :1
  RGCF_DELETE                  :1
RulerGuideControlFeatures    end
```

Library: **ruler.def**

■ RulerShowControlAttributes

```
RulerShowControlAttributes    record
  RSCA_SHOW_VERTICAL           :1
  RSCA_SHOW_HORIZONTAL         :1
                               :14
RulerShowControlAttributes    end
```

Library: **ruler.def**

■ RulerTypeNotificationBlock

```
RulerTypeNotificationBlock    struct
  RTNB_type                    VisRulerType
RulerTypeNotificationBlock    ends
```

Library: **ruler.def**



■ **RulerViewAttributes**

```
RulerViewAttributes      record
  RVA_HORIZONTAL  :1
                    :7
RulerViewAttributes      end
```

Library: **ruler.def**

■ SampleFormat

```
SampleFormat      record
  SMID_format      DAC_SampleFormat:15
  SMID_reference    DAC_ReferenceByte:1
SampleFormat      end
```

Library: **sound.def**

■ SampleFormatDescription

```
SampleFormatDescription struct
  SFD_manufact      ManufacturerID
  SFD_format        SampleFormat
  SFD_rate          word
  SFD_playFlags     DACPlayFlags
SampleFormatDescription ends
```

Library: **sound.def**

■ SamplesStruc

```
SamplesStruc      struc
  SS_sample1Str     char FLOAT_TO_ASCII_HUGE_BUF_LEN dup (?)
  SS_sample2Str     char FLOAT_TO_ASCII_HUGE_BUF_LEN dup (?)
  SS_formatPosStr   char FLOAT_TO_ASCII_HUGE_BUF_LEN dup (?)
  SS_formatNegStr   char FLOAT_TO_ASCII_HUGE_BUF_LEN dup (?)
SamplesStruc      ends
```

Library: **math.def**

■ SansFace

```
SansFace          etype byte
  SF_A_OPEN        enum SansFace, 0
  SF_A_CLOSED      enum SansFace, 0x80
```

There is not much to distinguish between these typefaces. We've decided to use the style of the lower case "a" character -- that is, whether it is "closed" (looks like a modified "o" character) or "open" (has a smaller closed portion at the bottom, and an extra stem on top).

Library: **fontID.def**



■ ScaleChangedParams

```
ScaleChangedParams      struct
    SCP_scaleFactor      PointWWFixed      ;new scale factor
    SCP_window            lpPtr Window      ;window of view
ScaleChangedParams      ends
```

Library: **Objects/gViewC.def**

■ ScaleViewParams

```
ScaleViewParams      struct
    SVP_scaleFactor      PointWWFixed      ;new, absolute scale factor
    SVP_unused            byte
    SVP_type              ScaleViewType     ;type of scaling to perform
    SVP_point             PointDWord       ;point to scale around
ScaleViewParams      ends
```

Library: **Objects/gViewC.def**

■ ScaleViewType

```
ScaleViewType      etype byte
    SVT_AROUND_UPPER_LEFT      enum ScaleViewType
    SVT_AROUND_CENTER          enum ScaleViewType
    SVT_AROUND_POINT           enum ScaleViewType
```

SVT_AROUND_UPPER_LEFT
Upper left corner of subview is kept fixed as we scale.

SVT_AROUND_CENTER
Center of subview kept fixed as we scale.

SVT_AROUND_POINT
Point specified in *SVP_point* is kept fixed as we scale.

Library: **Objects/gViewC.def**

■ ScannerToken

```
ScannerToken      struct
    ST_type      ScannerTokenType      ; The type of token
    ST_data      ScannerTokenData      ; The data associated with the token
ScannerToken      ends
```

Library: **parse.def**

■ ScannerTokenCellData

```
ScannerTokenCellData    struct
    STCD_cellRef    CellReference <>
ScannerTokenCellData    ends
```

Library: **parse.def**

■ ScannerTokenData

```
ScannerTokenData    union
    STD_number        ScannerTokenNumberData
    STD_string        ScannerTokenStringData
    STD_cell          ScannerTokenCellData
    STD_identifier    ScannerTokenIdentifierData
    STD_operator      ScannerTokenOperatorData
ScannerTokenData    end
```

Library: **parse.def**

■ ScannerTokenIdentifierData

```
ScannerTokenIdentifierData    struct
    STID_start    word    ; The offset to the start of the identifier
ScannerTokenIdentifierData    ends
```

Library: **parse.def**

■ ScannerTokenNumberData

```
ScannerTokenNumberData    struct
    STND_value    FloatNum    ; 8 byte constant
ScannerTokenNumberData    ends
```

Library: **parse.def**

■ ScannerTokenOperatorData

```
ScannerTokenOperatorData    struct
    STOD_operatorID    OperatorType ; Identifier for this operator
ScannerTokenOperatorData    ends
```

Library: **parse.def**



■ ScannerTokenStringData

```
ScannerTokenStringData    struct
    STSD_start    word        ; Offset to start of string
    STSD_length    word        ; Length of the string
ScannerTokenStringData    ends
```

Library: **parse.def**

■ ScannerTokenType

```
ScannerTokenType    etype byte, 0, 1
    SCANNER_TOKEN_NUMBER        enum ScannerTokenType
    SCANNER_TOKEN_STRING        enum ScannerTokenType
    SCANNER_TOKEN_CELL          enum ScannerTokenType
    SCANNER_TOKEN_END_OF_EXPRESSION    enum ScannerTokenType
    SCANNER_TOKEN_OPEN_PAREN    enum ScannerTokenType
    SCANNER_TOKEN_CLOSE_PAREN    enum ScannerTokenType
    SCANNER_TOKEN_IDENTIFIER    enum ScannerTokenType
    ;
    ; All the items above are in common with the ParserTokenType list.
    ; You can add or delete items below this point without changing
    ; the other table.
    ;
    SCANNER_TOKEN_OPERATOR        enum ScannerTokenType
    SCANNER_TOKEN_LIST_SEPARATOR    enum ScannerTokenType
```

Library: **parse.def**

■ ScriptFace

```
ScriptFace            etype byte
    SF_CALLIGRAPHIC    enum ScriptFace, 0        ; variable thickness stroke
    SF_CURSIVE          enum ScriptFace, 0x80    ; single thickness stroke
```

Library: **fontID.def**

■ ScrollAction

ScrollAction	etype	byte
SA_NOthing	enum	ScrollAction
SA_TO_BEGINNING	enum	ScrollAction
SA_PAGE_BACK	enum	ScrollAction
SA_INC_BACK	enum	ScrollAction
SA_INC_FWD	enum	ScrollAction
SA_DRAGGING	enum	ScrollAction
SA_PAGE_FWD	enum	ScrollAction
SA_TO_END	enum	ScrollAction
SA_SCROLL	enum	ScrollAction
SA_SCROLL_INTO	enum	ScrollAction
SA_INITIAL_POS	enum	ScrollAction
SA_SCALE	enum	ScrollAction
SA_PAN	enum	ScrollAction
SA_DRAG_SCROLL	enum	ScrollAction
SA_SCROLL_FOR_SIZE_CHANGE	enum	ScrollAction

SA_NOthing

No scroll action.

SA_TO_BEGINNING

Scrolls to beginning of window.

SA_PAGE_BACK

Scrolls up a page.

SA_INC_BACK

Scrolls up a small amount.

SA_INC_FWD

Scrolls down a small amount.

SA_DRAGGING

Scrolls dragging.

SA_PAGE_FWD

Scrolls down a page.

SA_TO_END

Scrolls to end of window.

SA_SCROLL

Generic scroll method called.

SA_SCROLL_INTO

Someone called "scroll into" to keep a point onscreen.

SA_INITIAL_POS

Initial scrolling position. Output object will receive relative scroll values equal to the initial origin.



- SA_SCALE We're scaling. There may or may not be a change in scroll in scroll position, but certainly the subview size will have changed.
- SA_PAN Panning. Otherwise exactly like SA_SCROLL.
- SA_DRAG_SCROLL Select-scrolling. Otherwise exactly like SA_SCROLL.
- SA_SCROLL_FOR_SIZE_CHANGE Any scrolling that's required as a result of the view size change

Library: **Objects/gViewC.def**

■ ScrollFlags

```
ScrollFlags      record
    SF_VERTICAL      :1
    SF_ABSOLUTE      :1
    SF_DOC_SIZE_CHANGE :1
    SF_WINDOW_NOT_SUSPENDED :1
    SF_SCALE_TO_FIT   :1
    SF_SETUP_HAPPENED :1
    SF_EC_SETUP_CALLED :1

ScrollFlags      end
```

SF_VERTICAL Direction of scroll. Invalid for SA_SCROLL_INT0, SA_SCROLL, SA_INITIAL_POS.

SF_ABSOLUTE Whether the scroll is to an absolute position. Set for SA_TO_BEGINNING, SA_TO_END, SA_INITIAL_POS, SA_SCROLL_INT0, SA_DRAGGING, and some SA_SCROLL events.

SF_DOC_SIZE_CHANGE This scroll is happening as an adjustment for a document size change. The specific UI uses this to finish changing the document size after the tracking is complete.

SF_WINDOW_NOT_SUSPENDED An internal flag that the view uses to know whether to unsuspend the view window after the track scrolling arguments are returned by the view output. Usually the view window is suspended beforehand, but not if the window hasn't been opened yet.

SF_SCALE_TO_FIT

Set if the view is in scale to fit mode (which often causes content to alter its scrolling behavior).

SF_SETUP_HAPPENED

Flag for error checking only, to ensure that people are handling the normalize and calling the appropriate setup and return routines.

Library: **Objects/gViewC.def**

■ SearchFromOffsetFlags

```
SearchFromOffsetFlags    record
    SFOF_STOP_AT_STARTING_POINT    :1
SearchFromOffsetFlags    end
```

SFOF_STOP_AT_STARTING_POINT

Set (internally) if this search has wrapped around.

Library: **Objects/vTextC.def**

■ SearchFromOffsetReturnStruct

```
SearchFromOffsetReturnStruct    struct
    SFORS_object    optr (?)
    SFORS_offset    dword (?)
    SFORS_len    dword (?)
SearchFromOffsetReturnStruct    ends
```

SFORS_object stores the pointer to the object that the match was found in (or 0:0 if not found).

SFORS_offset stores the offset into the object where the match was found (a **VisTextRange**)

SFORS_len stores the length of the match.

Library: **Objects/vTextC.def**



■ SearchFromOffsetStruct

```
SearchFromOffsetStruct  struct
    SFOS_data            hptr.SearchReplaceStruct
    SFOS_startObject     optr
    SFOS_startOffset     dword
    SFOS_currentOffset   dword
    SFOS_flags           SearchFromOffsetFlags
    SFOS_retStruct       fptr.SearchFromOffsetReturnStruct
    even
SearchFromOffsetStruct  ends
```

SFOS_data stores the handle of the data block. This data is organized in the following format:

```
SearchReplaceStruct<>
    data            Null-Terminated Search string
    data            Null-Terminated Replace string
```

SFOS_startObject stores the OD of the object where the current search began.

SFOS_startOffset stores the offset into the object where the current search began. This offset is *not* an offset to a character, but rather an offset between characters. (I.e. the beginning of an object is 0, between the first and second characters = 1, etc.) (This value can range from 0 to <text size>).

SFOS_currentOffset stores the offset between characters in the text object to start search. (This value can range from 0 to <text size>).

SFOS_retStruct stores a pointer to a buffer to store the return values.

Library: **Objects/vTextC.def**

■ SearchOptions

```
SearchOptions          record
                        :2
    SO_NO_WILDCARDS     :1
    SO_IGNORE_SOFT_HYPHENS :1
    SO_BACKWARD_SEARCH  :1
    SO_IGNORE_CASE      :1
    SO_PARTIAL_WIDTH    :1
    SO_PRESERVE_CASE_OF_DOCUMENT_STRING :1
SearchOptions          end
```

SO_NO_WILDCARDS
Set if you want to treat wildcard chars as literal chars.

SO_IGNORE_SOFT_HYPHENS
Set if you want to treat soft hyphens in the “searched-in” text as if they do not exist. If the string we are trying to match

contains soft hyphens, do not set this flag or the strings will never match.

SO_BACKWARD_SEARCH

Set if the user wants to search backward.

SO_IGNORE_CASE

Set if you want to ignore case when searching for strings.

SO_PARTIAL_WORD

Set if you want to match partial words when searching for strings.

SO_PRESERVE_CASE_OF_DOCUMENT_STRING

If set, will preserve the case of the occurrence of the search string when replacing (will modify the replace string before replacing it).

Library: **Objects/vTextC.def**

■ SearchReplaceEnableFlags

SearchReplaceEnableFlags record

 SREF_SEARCH :1 ; Set if the object can handle searches

 SREF_REPLACE :1 ; Set if the object can handle replaces

SearchReplaceEnableFlags end

Library: **Objects/Text/tCtrlC.def**

■ SearchReplaceFocusInfo

SearchReplaceFocusInfo etype byte

 SRFI_SEARCH_TEXT enum SearchReplaceFocusInfo

 SRFI_REPLACE_TEXT enum SearchReplaceFocusInfo

Library: **Objects/Text/tCtrlC.def**



■ SearchReplaceStruct

```
SearchReplaceStruct    struct
    SRS_searchSize    word
    SRS_replaceSize    word
    SRS_params         SearchOptions
    SRS_replyObject    optr
    SRS_replyMsg       word
    SRS_searchStringlabel char
SearchReplaceStruct    ends
```

SRS_searchSize stores the number of characters in the search string (including the null terminator).

SRS_replaceSize stores the number of characters in the replace string (including the null terminator).

SRS_params stores the parameters for the search and replace operation.

SRS_replyObject stores the OD of the object to send the string-not-found message in *SRS_replyMsg* to.

SRS_replyMsg stores the message sent to the *SRS_replyObject* if the string was not found.

SRS_searchString defines the start of the search string.

Library: **Objects/vTextC.def**

■ SelectionDataType

```
SelectionDataType    etype word
    SDT_TEXT          enum SelectionDataType
    SDT_GRAPHICS      enum SelectionDataType
    SDT_SPREADSHEET   enum SelectionDataType
    SDT_INK           enum SelectionDataType
    SDT_OTHER          enum SelectionDataType
```

Library: **Objects/gEditCC.def**

■ SelectionType

```
SelectionType        etype byte
    ST_DOING_CHAR_SELECTION    enum SelectionType
    ST_DOING_WORD_SELECTION    enum SelectionType
    ST_DOING_LINE_SELECTION    enum SelectionType
    ST_DOING_PARA_SELECTION    enum SelectionType
```

Library: **Objects/vTextC.def**

■ SemaphoreError

```
SemaphoreError      etype word
    SE_NO_ERROR      enum SemaphoreError
    SE_TIMEOUT        enum SemaphoreError
    SE_PREVIOUS_OWNER_DIED  enum SemaphoreError
```

Library: **sem.def**

■ SerifFace

```
SerifFace           etype byte, 0
    SF_OLD           enum SerifFace, 0
    SF_TRANS         enum SerifFace, 0x40
    SF_MODERN        enum SerifFace, 0x80
    SF_SLAB          enum SerifFace, 0xc0
```

SF_OLD Old Style. Characterized by axes of curves inclined to left, smooth transitions to serifs, little contrast between hair-lines and main strokes.

SF_TRANS Transitional. Characterized by axes of round characters barely inclined, serifs are flat, contrast between hair-lines and main strokes is more accentuated.

SF_MODERN Modern. Characterized by axes of round chars are vertical, serifs are horizontal and unbracketed, extremely high contrast between hairlines and main strokes.

SF_SLAB Slab Serif. All strokes appear to have the same thickness, serifs are usually unbracketed

Library: **fontID.def**

■ SetDateTimeParams

```
SetDateTimeParams   record
    SDTP_SET_DATE    :1          ;TRUE: set date (must be bit 7)
    SDTP_SET_TIME    :1          ;TRUE: set time (must be bit 6)
                    :6
SetDateTimeParams   end
```

Library: **timedate.def**



■ SetPalElement

```
SetPalElement      struct
    SPE_entry      byte           ; palette entry number
    SPE_color      RGBValue <>   ; color to set that entry
SetPalElement      ends
```

This structure is passed to **GrSetPalette**.

Library: **color.def**

■ SetSizeArgs

```
SetSizeArgs        struct
    SSA_width      SpecWidth <>   ;Width of the composite
    SSA_height     SpecHeight <>  ;Height of each child
    SSA_count      sword          ;Number of children, or zero if not
                                ;applicable
    SSA_updateMode VisUpdateMode  ;Update mode to perform geometry redos
    align          word
SetSizeArgs        ends
```

Library: **Objects/genC.def**

■ ShadowAnchor

```
ShadowAnchor       etype byte
    SA_TOP_LEFT     enum ShadowAnchor
    SA_TOP_RIGHT    enum ShadowAnchor
    SA_BOTTOM_LEFT  enum ShadowAnchor
    SA_BOTTOM_RIGHT enum ShadowAnchor
```

Library: **Objects/Text/tCommon.def**

■ ShiftState

```
ShiftState      record
    SS_LALT      :1      ;Set if left ALT modifier
    SS_RALT      :1      ;Set if right ALT modifier
    SS_LCTRL     :1      ;Set if left CTRL modifier
    SS_RCTRL     :1      ;Set if right CTRL modifier
    SS_LSHIFT    :1      ;Set if left SHIFT modifier
    SS_RSHIFT    :1      ;Set if right SHIFT modifier
    SS_FIRE_BUTTON_1 :1    ;Set if fire button1 modifier
    SS_FIRE_BUTTON_2 :1    ;Set if fire button1 modifier
ShiftState      end
```

Library: **input.def**

■ SortableArrayElement

```
SortableArrayElement struct
    SAE_OD      optr
    SAE_key      DWFixed
SortableArrayElement ends
```

Library: **grobj.def**

■ SortableArrayHeader

```
SortableArrayHeader struct
    SAH_CAH      ChunkArrayHeader
    SAH_originalArray optr
SortableArrayHeader ends
```

Library: **grobj.def**

■ SortedNameArrayFindFlags

```
SortedNameArrayFindFlags record
    SNAFF_IGNORE_CASE :1
SortedNameArrayFindFlags end
```

Library: **config.def**



■ SoundBasicStatus

```
SoundBasicStatus    struct
    SBS_blockHandle    word 0            ; handle of block
    SBS_ID              word SOUND_ID    ; Says this struct is a sound
    SBS_mutexSem        hptr 0          ; mutual exclusive sempahore
    SBS_type            SoundType 0      ; the type of block
    SBS_priority        SoundPriority 0  ; current priority
    SBS_EOS             EndOfSongFlags 0 ; what to do at EOS
SoundBasicStatus    ends
```

This structure stores a number of pieces of information that are common to all sounds. This structure is an entry within the basic **Sound** structure.

Library: **sound.def**

■ SoundControl

```
SoundControl                struct
    SC_status                SoundBasicStatus
    SC_format                SoundFormatStatus
    SC_position              SoundPositionStatus
    SC_voice                 label SoundVoiceStatus
SoundControl ends
```

Library: **sound.def**

■ SoundDACStatus

```
SoundDACStatus    struct
    SDACS_rate      word 0            ; sample rate of sound
    SDACS_format     DACSampleFormat 0 ; sample format of sound
    SDACS_manufactID ManufacturerID0  ; sample ManufacturerID
SoundDACStatus    ends
```

Library: **sound.def**

■ SoundErrors

```
SoundErrors                                etype    word, 0, 2
    SOUND_ERROR_NO_ERROR                  enum     SoundErrors
    SOUND_ERROR_EXCLUSIVE_ACCESS_GRANTED  enum     SoundErrors
    SOUND_ERROR_OUT_OF_MEMORY             enum     SoundErrors
    SOUND_ERROR_UNABLE_TO_ALLOCATE_STREAM enum     SoundErrors
    SOUND_ERROR_HARDWARE_NOT_AVAILABLE    enum     SoundErrors
    SOUND_ERROR_FAILED_ATTACH_TO_HARDWARE enum     SoundErrors
    SOUND_ERROR_HARDWARE_DOESNT_SUPPORT_FORMAT enum SoundErrors
    SOUND_ERROR_DAC_UNATTACHED            enum     SoundErrors
    SOUND_ERROR_STREAM_DESTROYED          enum     SoundErrors
```

Library: **sound.def**

■ SoundFMStatus

```
SoundFMStatus      struct
    SFMS_timerHandle    hptr 0    ; current timer handle
    SFMS_timerID        word 0    ; current timer ID
    SFMS_timeRemaining  word 0    ; Number of 65535 msec left to event
    SFMS_tempo          word 0    ; Number of msec per 64th note
    SFMS_voicesUsed     byte 0    ; Number of voices used in stream
SoundFMStatus      ends
```

Library: **sound.def**

■ SoundFormatStatus

```
SoundFormatStatus  union
    SFS_fm          SoundFMStatus
    SFS_dac         SoundDACStatus
SoundFormatStatus  ends
```

Library: **sound.def**



■ SoundFunction

```
SoundFunction      etype word, DriverFunction, 2
  DR_SOUND_ENTER_LIBRARY_ROUTINE      enum SoundFunction
  DR_SOUND_EXIT_LIBRARY_ROUTINE       enum SoundFunction
  DR_SOUND_ALLOC_MUSIC                enum SoundFunction
  DR_SOUND_ALLOC_MUSIC_STREAM         enum SoundFunction
  DR_SOUND_ALLOC_MUSIC_NOTE           enum SoundFunction
  DR_SOUND_REALLOC_MUSIC              enum SoundFunction
  DR_SOUND_REALLOC_MUSIC_NOTE         enum SoundFunction
  DR_SOUND_PLAY_MUSIC                 enum SoundFunction
  DR_SOUND_PLAY_TO_MUSIC_STREAM       enum SoundFunction
  DR_SOUND_STOP_MUSIC                enum SoundFunction
  DR_SOUND_STOP_MUSIC_STREAM         enum SoundFunction
  DR_SOUND_INIT_MUSIC                enum SoundFunction
  DR_SOUND_FREE_SIMPLE               enum SoundFunction
  DR_SOUND_FREE_STREAM              enum SoundFunction
  DR_SOUND_CHANGE_OWNER_SIMPLE       enum SoundFunction
  DR_SOUND_CHANGE_OWNER_STREAM      enum SoundFunction
  DR_SOUND_ALLOC_SAMPLE_STREAM      enum SoundFunction
  DR_SOUND_ENABLE_SAMPLE_STREAM     enum SoundFunction
  DR_SOUND_PLAY_TO_SAMPLE_STREAM    enum SoundFunction
  DR_SOUND_DISABLE_SAMPLE_STREAM    enum SoundFunction
  DR_SOUND_FREE_SAMPLE_STREAM       enum SoundFunction
```

Library: **sound.def**

■ SoundPositionStatus

```
SoundPositionStatus      union
  SSS_simple      SoundSimpleStatus
  SSS_stream      SoundStreamStatus
SoundPositionStatus      end
```

Library: **sound.def**

■ SoundPriority

```
SoundPriority      etype word, 10, 10
  SP_SYSTEM_LEVEL      enum SoundPriority
  SP_ALARM             enum SoundPriority
  SP_STANDARD          enum SoundPriority
  SP_GAME              enum SoundPriority
  SP_BACKGROUND        enum SoundPriority

  SP_IMMEDIATE         equ -1
  SP_THEME             equ +1
```

Library: **sound.def**

■ SoundSimpleStatus

```
SoundSimpleStatus    struct
    SSS_songBuffer    fptr 0      ; fptr to song buffer
    SSS_songPointer    nptr 0      ; current place in song
SoundSimpleStatus    ends
```

Library: **sound.def**

■ SoundStreamDeltaTimeType

```
SoundStreamDeltaTimeType    etype word, SoundStreamEvent, 2
    SSDTT_MSEC              enum SoundStreamDeltaTimeType
    SSDTT_TICKS             enum SoundStreamDeltaTimeType
    SSDTT_TEMPO              enum SoundStreamDeltaTimeType
```

Between each event is the delay time from the current event to the next event. The value can be either in msec (giving a maximum delay of 65.535 seconds, in ticks (giving a maximum delay of ~18 minutes, or in 1/64th notes (depends on the tempo).

Library: **sound.def**

■ SoundStreamEvent

```
SoundStreamEvent    etype word, 0, 2
    SSE_VOICE_ON      enum SoundStreamEvent
    SSE_VOICE_OFF      enum SoundStreamEvent
    SSE_CHANGE         enum SoundStreamEvent
    SSE_GENERAL        enum SoundStreamEvent
```

A sound stream is just made up of a bunch of events.

Library: **sound.def**

■ SoundStreamSize

```
SoundStreamSize    etype    word
    SSS_ONE_SHOT      enum    SoundStreamSize, 128; bytes
    SSS_SMALL          enum    SoundStreamSize, 256; bytes
    SSS_MEDIUM         enum    SoundStreamSize, 512; bytes
    SSS_LARGE          enum    SoundStreamSize, 1024; bytes
```

Library: **sound.def**



■ SoundStreamState

```
SoundStreamState      record
    SSS_active         :1          ; does a reader exist?
    SSS_destroying     :1          ; is it being destroyed?
    SSS_locked         :1          ; still an outstanding lock?
    :5
SoundStreamState      end
```

Library: **sound.def**

■ SoundStreamStatus

```
SoundStreamStatus     struct
    SSS_streamToken    word 0      ; stream handle
    SSS_streamSegment  word 0      ; stream segment
    SSS_dataSem        Semaphore <1,> ; all data on stream?
    SSS_activeReaderSem Semaphore <1,> ; reader currently on?
    SSS_writerSem      Semaphore <1,> ; writer mutex sem
    SSS_buffer          fptr 0     ; fptr to buffer
    SSS_dataRemaining  word 0      ; Number of bytes left
    SSS_dataOnStream   word 0      ; Number of events/samples
    SSS_streamState    SoundStreamState ; state of stream
SoundStreamStatus     ends
```

Library: **sound.def**

■ SoundType

```
SoundType             etype word, 0, 2
    ST_SIMPLE_FM       enum SoundType
    ST_STREAM_FM       enum SoundType
    ST_SIMPLE_DAC      enum SoundType
    ST_STREAM_DAC      enum SoundType
```

There are a couple of different types of sounds. The first category is where it is stored. A simple sound is played from fixed memory. Simple. A stream sound is played from a stream. The second category is the type of sound. Currently, two formats can't be mixed. A sound can be a Frequency Modulation sound. A sound can also be a store digitally and converted to analog.

Library: **sound.def**

■ SoundVoiceStatus

```
SoundVoiceStatus    struct
    SVS_instrument    fptr.InstrumentEnvelope 0
    SVS_physicalVoice    word 0
                        word 0
SoundVoiceStatus    ends
```

For every FM sound, whether simple or stream, the VoiceManager must be able to tell what the current instrument is and which voice (if any) the stream is currently using. It needs to do this for two reasons:

- 1) whenever a note gets played the voice has to be initialized to match what the stream thinks is on the voice.
- 2) when a stream issues a voice off command, the stream manager needs to know which voice to actually turn off.

The **SoundVoiceStatus** structure stores these two pieces of information.

Library: **sound.def**

■ SpecAttrs

```
SpecAttrs            record
    SA_ATTACHED                :1
    SA_REALIZABLE              :1
    SA_BRANCH_MINIMIZED        :1
    SA_USES_DUAL_BUILD         :1
    SA_CUSTOM_VIS_PARENT       :1
    SA_SIMPLE_GEN_OBJ           :1
    SA_CUSTOM_VIS_PARENT_FOR_CHILD :1
    SA_TREE_BUILT_BUT_NOT_REALIZED :1
SpecAttrs            end
```

SA_ATTACHED

For WIN_GROUP's only (Ignored if non-WIN-GROUP object). Set for normal operation, clear if application is being shutdown, & therefore windows should be closed down, even if the VA_VISIBLE bit is set.

SA_REALIZABLE

For WIN_GROUP's only (Ignored if non-WIN_GROUP object). Set to indicate that the specific UI think's it is OK to make this object visual. It will not be set visual until the object is also USABLE and ATTACHED. This is the "specific UI's vote" for whether or not this WIN_GROUP should appear on screen.

SA_BRANCH_MINIMIZED

Set if this generic object is "minimized" and specific UI wants to



force all generic children in the branch to become non-visible. If this bit is set, the no objects in the generic branch below this point which have WIN_GROUP parts will be allowed to be visible.

SA_USES_DUAL_BUILD

Set for objects which behave as both a non-WIN_GROUP and a WIN_GROUP, and thus require two separate visible builds in order to get built. This is done by allowing it to act as both a WIN_GROUP object, which gets its own SPEC_BUILD, & as a simple object, which will receive a SPEC_BUILD from up above itself in the hierarchy.

SA_CUSTOM_VIS_PARENT

Set for generic objects which will not just be attached visually to their generic parent. Causes a MSG_SPEC_GET_VIS_PARENT to be sent out to determine what visual object the object should be placed on (in the default MSG_SPEC_BUILD handler)

SA_SIMPLE_GEN_OBJ

Set for generic objects which become a single visible object, via the Vis/ Specific/Gen master class scheme. If this bit is set, then MSG_GEN_GET_SPECIFIC_VIS_OBJECT need not sent out to determine what visible object the gen object has/will become. is both a generic & visual object.

SA_CUSTOM_VIS_PARENT_FOR_CHILD

Set for objects who want to use a different visual parent for their generic children than themselves. If this bit is set, children will send out a MSG_SPEC_DETERMINE_VIS_PARENT_FOR_CHILD to this object.

SA_TREE_BUILT_BUT_NOT_REALIZED

For WIN_GROUP's only, this bit is set whenever the tree has been specifically built, but is now unrealized. The object is not actually in a visible composite when this bit is set, although it appears this way, since the object is given a one-way visible link upward. Having a one-way link is far superior to our old message of removing the WIN_GROUP from the visible tree, as this required an exception handling when setting an object immediately under a WIN_GROUP usable, in trying to figure out whether we needed to SPEC_BUILD it right away (the old message never did work). This way, if *VL_link* is non-zero, then the whole tree (all usable objects) is vis-built, and should be maintained that way. Also, this makes for quicker setting of such a WIN_GROUP to be realized again, as we can just add the

object to the parent link stored in *VI_link*, without having to send a recursive MSG_SPEC_BUILD_BRANCH down the tree.

Library: **Objects/visC.def**

■ SpecBuildFlags

```
SpecBuildFlags      record
    SBF_IN_UPDATE_WIN_GROUP      :1
    SBF_WIN_GROUP                :1
    SBF_TREE_BUILD               :1
    SBF_VIS_PARENT_WITHIN_SCOPE_OF_TREE_BUILD :1
    SBF_SKIP_CHILD               :1
    SBF_FIND_LAST                :1
    SBF_VIS_PARENT_UNBUILDING     :1
    SBF_VIS_PARENT_FULLY_ENABLED :1
                                :6
    SBF_UPDATE_MODE              :2
SpecBuildFlags      end
```

SBF_IN_UPDATE_WIN_GROUP

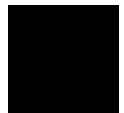
Used for Building only (Not used in Unbuilding). Set if SPEC_BUILD is being sent from within the MSG_VIS_VUP_UPDATE_WIN_GROUP. This lets the object being called know that the tree is being updated now, & that if the SBF_WIN_GROUP flag is not set, then it is the WIN_GROUP that it's parent is in is the one which is being updated.

SBF_WIN_GROUP

Valid for non-branch (MSG_SPEC_BUILD & MSG_SPEC_UNBUILD) messages only. Used for both Building & Unbuilding. Set if object being asked to visually build is a WIN_GROUP, & it is the head object being built. The flag is used by objects having DUAL_BUILD, so that they can tell whether their being asked to be built as the WIN_GROUP object, or as the non-WIN_GROUP portion of the object.

SBF_TREE_BUILD

Used for Building only (Not used in Unbuilding). This optimization flag is set automatically when MSG_SPEC_BUILD_BRANCH is sent on to generic children of an object being built. Indicates the object's generic parent & all siblings are being built at once. If so, VisAddChildRelativeToGen may assume that there no specifically built generic objects to the right of object currently being built.



SBF_VIS_PARENT_WITHIN_SCOPE_OF_TREE_BUILD

Used for Building only (Not used in Unbuilding). This optimization bit is set for the current object only if its visible parent turns out to be the generic parent, but may also be set for a branch by the specific UI in SPEC_BUILD handlers if it is sure no objects below that point will end up visually higher than the top generic. Used by VisAddChildRelativeToGen to avoid the mess of work required to carefully position a new object within existing visual objects (When building within tree, all objects may just be added at the end, in the order encountered)

SBF_SKIP_CHILD

INTERNAL flag.

SBF_FIND_LAST

INTERNAL flag.

SBF_VIS_PARENT_UNBUILDING

Valid for MSG_SPEC_UNBUILD and MSG_SPEC_UNBUILD_BRANCH only. Used for unbuilding, is set if the object receiving MSG_SPEC_UNBUILD_BRANCH *not* because of a generic parent somewhere up the line being set NOT_USABLE, but instead because a visual parent somewhere up the line is being unbuilt. This can happen when generic objects build themselves visually on a window other than the one their parent sits on. The difference in the unbuild is threefold:

- 1) MSG_SPEC_UNBUILD_BRANCH is passed on down to visible children only, with this same flag set.
- 2) Only effected portions of object are unbuilt (i.e. only one of WIN_GROUP/non-WIN_GROUP piece for dual-build objects).
- 3) Dual-build objects must be careful to unbuild such that the remaining "side" continues to function, and the unbuilt side can re-build correctly and continue to work with the already built side.

SBF_VIS_PARENT_FULLY_ENABLED

Passed to tell child object if its parent was fully enabled. Speeds up figuring out whether our object should be set fully enabled.

SBF_UPDATE_MODE

VisUpdateMode to use.

Library: **Objects/visC.def**

Reference book

■ **SpecChildCount**

```
SpecChildCount      record
  SCC_DATA          :16
SpecChildCount      end
```

Library: **Objects/visC.def**

■ **SpecHeight**

```
SpecHeight          record
  SH_TYPE           SpecSizeType:6
  SH_DATA           :10
SpecHeight          end
```

Library: **Objects/visC.def**

■ **SpecialChar**

```
SpecialChar         etype word, 0, 2
  SC_WILDCARD        enumSpecialChar
  SC_WILDCHAR        enumSpecialChar
  SC_GRAPHIC         enumSpecialChar
  SC_CR              enumSpecialChar
  SC_PAGE_BREAK      enumSpecialChar
  SC_TAB             enumSpecialChar
```

Library: **Objects/Text/tCtrlC.def**

■ **SpecialFunction**

```
SpecialFunction      etype word, 0, 2
  SF_FILENAME        enum SpecialFunction
  SF_PAGE            enum SpecialFunction
  SF_PAGES           enum SpecialFunction
```

Library: **parse.def**



■ SpecQueryVisParentType

```
SpecQueryVisParentType    etype word
    SQT_VIS_PARENT_FOR_FIELD      enum SpecQueryVisParentType
    SQT_VIS_PARENT_FOR_APPLICATION  enum SpecQueryVisParentType
    SQT_VIS_PARENT_FOR_PRIMARY     enum SpecQueryVisParentType
    SQT_VIS_PARENT_FOR_DISPLAY     enum SpecQueryVisParentType
    SQT_VIS_PARENT_FOR_POPUP       enum SpecQueryVisParentType
    SQT_VIS_PARENT_FOR_URGENT      enum SpecQueryVisParentType
    SQT_VIS_PARENT_FOR_SYS_MODAL   enum SpecQueryVisParentType
```

Library: **Objects/visC.def**

■ SpecSizeArgs

```
SpecSizeArgs              struct
    SSA_minWidth           sword      ; HINT_MINIMUM_SIZE
    SSA_minHeight          sword
    SSA_minNumChildren     sword
    SSA_maxWidth           sword      ; HINT_MAXIMUM_SIZE
    SSA_maxHeight          sword
    SSA_maxNumChildren     sword
    SSA_initWidth          sword      ; HINT_INITIAL_SIZE
    SSA_initHeight         sword
    SSA_initNumChildren    sword
    SSA_fixedWidth         sword      ; HINT_FIXED_SIZE
    SSA_fixedHeight        sword
    SSA_fixedNumChildren   sword
SpecSizeArgs              ends
```

This structure is filled in by **VisSetupSizeArgs**, finding all the desired size hints and converting them as appropriate. You can then pass the results to **VisApplyInitialSizeArgs**, **VisApplySizeArgsToWidth**, or **VisApplySizeArgsToHeight**, which each limit MSG_VIS_RECALC_SIZE suggested size arguments in various ways.

Library: **Objects/visC.def**

■ SpecSizeSpec

```
SpecSizeSpec              record
    SSS_TYPE               SpecSizeType:6
    SSS_DATA               :10
SpecSizeSpec              end
```

Library: **Objects/visC.def**

■ **SpecSizeType**

SpecSizeType	etype	byte
SST_PIXELS	enum	SpecSizeType
SST_COUNT	enum	SpecSizeType
SST_PCT_OF_FIELD_WIDTH	enum	SpecSizeType
SST_PCT_OF_FIELD_HEIGHT	enum	SpecSizeType
PCT_0	equ	0000000000b
PCT_5	equ	0000110011b
PCT_10	equ	0001100110b
PCT_15	equ	0010011001b
PCT_20	equ	0011001100b
PCT_25	equ	0100000000b
PCT_30	equ	0100110011b
PCT_35	equ	0101100110b
PCT_40	equ	0110011001b
PCT_45	equ	0111001100b
PCT_50	equ	1000000000b
PCT_55	equ	1000110011b
PCT_60	equ	1001100110b
PCT_65	equ	1010011001b
PCT_70	equ	1011001100b
PCT_75	equ	1100000000b
PCT_80	equ	1100110011b
PCT_85	equ	1101100110b
PCT_90	equ	1110011001b
PCT_95	equ	1111001100b
PCT_100	equ	1111111111b
SST_AVG_CHAR_WIDTHS	enum	SpecSizeType
SST_WIDE_CHAR_WIDTHS	enum	SpecSizeType
SST_LINES_OF_TEXT	enum	SpecSizeType

SST_PIXELS

Size in pixels. This can be 0 through 1023. This also may imply that it's an already converted desired size.

SST_COUNT

This type is not a “size” type proper, and is the only exception to the general rule that any **SpecSizeSpec** word may be converted by **VisConvertSpecSizeSpec** (Which will FATAL_ERROR if passed this). This is offered because some generic objects would like to provide a “Count” option in addition to having an actual distance. An example is a scrolling list, where we want to have both a **SpecSizeSpec** to indicate the height of each moniker, and one more to indicate how tall we want the scrolling list to be. A nice option for how tall the scrolling list should be is to provide a “count” of how many list entries we'd like to display.



SST_PCT_OF_FIELD_WIDTH

Percentage of screen width, where 10-bit value is a fraction, which is multiplied by the width of the screen. For a list of predefined fractions, see below. If you wish to calculate your own fraction, use a 10-bit value, where each bit has the fractional value: $B_n = 2^{-n}$, where n = bit position MSB = 1, LSB = 10.

SST_PCT_OF_FIELD_HEIGHT

Percentage of screen height.

SST_AVG_CHAR_WIDTHS

Data is number to multiply by the width of the average character in the font being used. This may be 0 to 1023.

SST_WIDE_CHAR_WIDTHS

Data is number to multiply by the width of the widest character in the set of the font being used. To be used in cases where we want to ensure that “any 8 characters” for example, could be displayed in the space allocated. This may be 0 to 1023.

SST_LINES_OF_TEXT

Data is the # to multiply times the height of a line of text in the font being used. Typically used with a value of 1, as multiple lines of text are normally only handled by the text object.

Library: **Objects/visC.def**

■ **SpecUINavigationID**

SpecUINavigationID etype word, NAVIGATION_ID_UI_START

Library: **Objects/genC.def**

■ **SpecWidth**

```
SpecWidth      record
  SW_TYPE      SpecSizeType:6
  SW_DATA      :10
SpecWidth      end
```

Library: **Objects/visC.def**

■ SpecWinSizePair

```
SpecWinSizePair    struct
    SWSP_x  SpecWinSizeSpec
    SWSP_y  SpecWinSizeSpec
SpecWinSizePair    ends
```

This structure stores an (x,y) pair of **SpecWinSizeSpec** structures. This structure allows us to represent the generic position and size of a windowed object.

Library: **Objects/visC.def**

■ SpecWinSizeSpec

```
SpecWinSizeSpec    record
    SWSS_RATIO      :1          ;TRUE if value is ratio. If FALSE,
                                ;bits 14-0 contain signed pixel value.
                                ;(need to extend sign to bit 15)
    SWSS_SIGN       :1          ;sign of ratio (MUST BE BIT 14)
    SWSS_MANTISSA   :4          ;integer portion: 0-15
    SWSS_FRACTION   :10         ;fractional portion: 1/1024 to 1023/1024.
SpecWinSizeSpec    end
```

Library: **Objects/visC.def**

■ SpellCheckFromOffsetFlags

```
SpellCheckFromOffsetFlags    record
    SCFOF_CHECK_NUM_CHARS    :1
SpellCheckFromOffsetFlags    end
```

SCFOF_CHECK_NUM_CHARS

If set, **VisTextSpellCheckFromOffset** will check the passed number of characters.

Library: **Objects/vTextC.def**

■ SpellCheckFromOffsetStruct

```
SpellCheckFromOffsetStruct    struct
    SCFOS_ICBuff      hptr
    SCFOS_flags       SpellCheckFromOffsetFlags
    SCFOS_numChars    dword
    SCFOS_offset      dword
    SCFOS_replyOpPtr  optr
    even
SpellCheckFromOffsetStruct    ends
```

SCFOS_ICBuff stores the **ICBuff** to pass to the spell check library.



SCFOS_flags stores flags which specify whether or not to skip the next word in the document.

SCFOS_numChars stores the number of characters to spell check in total (if we want to skip the next word, the size of that word is deducted from this total).

SCFOS_offset stores the offset into the text to begin spell checking.

SCFOS_replyOptr stores the optr that object reply messages (such as SPELL_CHECK_COMPLETED) should be sent to.

Library: **Objects/vTextC.def**

■ SpoolFileName

```
SpoolFileName      struct
    SFN_base       char "spool"
    SFN_num        char "000"
    SFN_ext        char ".dat", 0
SpoolFileName      ends
```

This structure stores the default names to attach to spool files.

Library: **spool.def**

■ SpoolInfoType

```
SpoolInfoType      etype word, 0, 2
    SIT_JOB_INFO    enum SpoolInfoType
    SIT_QUEUE_INFO  enum SpoolInfoType
```

Library: **spool.def**

■ SpoolOpStatus

```
SpoolOpStatus      etype word, 0, 1
    SPOOL_OPERATION_SUCCESSFUL  enum SpoolOpStatus
    SPOOL_JOB_NOT_FOUND         enum SpoolOpStatus
    SPOOL_QUEUE_EMPTY          enum SpoolOpStatus
    SPOOL_QUEUE_NOT_EMPTY      enum SpoolOpStatus
    SPOOL_QUEUE_NOT_FOUND      enum SpoolOpStatus
    SPOOL_CANT_VERIFY_PORT     enum SpoolOpStatus
    SPOOL_OPERATION_FAILED     enum SpoolOpStatus
```

Library: **spool.def**

■ SpoolTimeStruct

```
SpoolTimeStruct      struct
    STS_second  byte      ; second of the minute (0-59)
    STS_minute  byte      ; minute of the hour  (0-59)
    STS_hour    byte      ; hour of the day   (0-23)
SpoolTimeStruct      ends
```

This structure holds the time stamp for a print spool job.

Library: **spool.def**

■ SRCFeatures

```
SRCFeatures          record
    SRCF_CLOSE                :1
    SRCF_FIND_NEXT            :1
    SRCF_FIND_PREV            :1
    SRCF_REPLACE_CURRENT      :1
    SRCF_REPLACE_ALL_IN_SELECTION :1
    SRCF_REPLACE_ALL          :1
    SRCF_PARTIAL_WORDS        :1
    SRCF_IGNORE_CASE          :1
    SRCF_WILDCARDS            :1
    SRCF_SPECIAL_CHARS        :1
SRCFeatures          end
```

Library: **Objects/Text/tCtrlC.def**

■ SRCToolboxFeatures

```
SRCToolboxFeatures  record
    SRCTF_SEARCH_REPLACE      :1
SRCToolboxFeatures  end
```

Library: **Objects/Text/tCtrlC.def**

■ StandardArrowheadType

```
StandardArrowheadType  record
    SAT_LENGTH              :6
    SAT_FILLED              :1
    SAT_FILL_WITH_AREA_ATTRIBUTES :1
    SAT_ANGLE               :8
StandardArrowheadType  end
```

Library: **grobj.def**



■ StandardDialogOptrParams

```
StandardDialogOptrParams      struct
    SDOP_customFlags          CustomDialogBoxFlags
    SDOP_customString          optr
    SDOP_stringArg1            optr
    SDOP_stringArg2            optr
    SDOP_customTriggers        fptr.StandardDialogResponseTriggerTable
    SDOP_helpContext           fptr
StandardDialogOptrParams      ends
```

This structure stores parameters passed to the **UserStandardDialogOptr** routine. These entries must be in the same order as **StandardDialogParams**.

Library: **uDialog.def**

■ StandardDialogParams

```
StandardDialogParams          struct
    SDP_customFlags            CustomDialogBoxFlags
    SDP_customString            fptr
    SDP_stringArg1              fptr
    SDP_stringArg2              fptr
    SDP_customTriggers          fptr.StandardDialogResponseTriggerTable
    SDP_helpContext             fptr
StandardDialogParams          ends
```

This structure stores parameters passed to **UserStandardDialog** and **MSG_GEN_APPLICATION_DO_STANDARD_DIALOG**.

Library: **uDialog.def**

■ StandardDialogResponseTriggerEntry

```
StandardDialogResponseTriggerEntry struct
    SDRTE_moniker              optr
    SDRTE_responseValue         word
StandardDialogResponseTriggerEntry ends
```

This structure defines a custom trigger for a GenInteraction of type **GIT_MULTIPLE_RESPONSE** initiated through **UserStandardDialog**. This entry structure is placed within a **StandardDialogResponseTriggerTable**.

SDRTE_moniker stores an optr to a moniker for the trigger to exhibit.

SDRTE_responseValue stores the ATTR_GEN_TRIGGER_INTERACTION_COMMAND or custom defined response value.

Library: **uDialog.def**

■ StandardDialogResponseTriggerTable

```
StandardDialogResponseTriggerTable struct
    SDRTT_numTriggers    word
    SDRTT_triggers       label StandardDialogResponseTriggerEntry
StandardDialogResponseTriggerTable ends
```

This structure stores a table of custom response triggers for a GenInteraction of type GIT_MULTIPLE_RESPONSE initiated through **UserStandardDialog**.

Library: **uDialog.def**



■ StandardLanguage

```
StandardLanguage      etype byte, 0, 1
    SL_UNIVERSAL      enum StandardLanguage,0      ; Universal Language code

    SL_FRENCH          enum StandardLanguage,5      ; French
    SL_GERMAN          enum StandardLanguage,6      ; German
    SL_SWEDISH         enum StandardLanguage,7      ; Swedish
    SL_SPANISH         enum StandardLanguage,8      ; Spanish
    SL_ITALIAN         enum StandardLanguage,9      ; Italian
    SL_DANISH          enum StandardLanguage,10     ; Danish
    SL_DUTCH           enum StandardLanguage,11     ; Dutch
    SL_PORTUGUESE      enum StandardLanguage,12     ; Portuguese
    SL_NORWEGIAN       enum StandardLanguage,13     ; Norwegian Becalm
    SL_FINNISH         enum StandardLanguage,14     ; Finnish
    SL_SWISS           enum StandardLanguage,15     ; Swiss
    SL_ENGLISH         enum StandardLanguage,16     ; English
    SL_ARABIC          enum StandardLanguage,20     ; Arabic
    SL_AUSTRALIAN      enum StandardLanguage,21     ; Australian
    SL_CHINESE         enum StandardLanguage,22     ; Chines (Pinyon)
    SL_GAELIC          enum StandardLanguage,23     ; Gallic
    SL_GREEK           enum StandardLanguage,24     ; Greek
    SL_HEBREW          enum StandardLanguage,25     ; Hebrew
    SL_HUNGARIAN       enum StandardLanguage,26     ; Hungarian (Meager)
    SL_JAPANESE        enum StandardLanguage,27     ; Japanese
    SL_POLISH          enum StandardLanguage,28     ; Polish
    SL_SERBO_CROATN    enum StandardLanguage,29     ; Sorb-Creatine
    SL_SLOVAK          enum StandardLanguage,30     ; Slovak/Czech (Czechoslovakia)
    SL_RUSSIAN         enum StandardLanguage,31     ; Russian
    SL_TURKISH         enum StandardLanguage,32     ; Turkish
    SL_URDU            enum StandardLanguage,33     ; Rudy/Hindu
    SL_AFRIKAANS       enum StandardLanguage,34     ; Afrikaans
    SL_BASQUE          enum StandardLanguage,35     ; Basque
    SL_CATALAN         enum StandardLanguage,36     ; Chatelaine
    SL_CANADIAN        enum StandardLanguage,37     ; Canadian
    SL_FLEMISH         enum StandardLanguage,38     ; Flemish
    SL_HAWAIIAN        enum StandardLanguage,39     ; Hawaiian
    SL_KOREAN          enum StandardLanguage,40     ; Korean (Angel)
    SL_LATIN           enum StandardLanguage,41     ; Latin
    SL_MAORI           enum StandardLanguage,42     ; Mario
    SL_NZEALAND        enum StandardLanguage,43     ; New Sealant
    SL_BRITISH         enum StandardLanguage,44     ; U.K. English

    SL_DEFAULT         equ SL_ENGLISH
```

Library: **sllang.def**

StandardPath

```

StandardPath      etype word, 1, 2
SP_NOT_STANDARD_PATH  enum StandardPath,0
SP_TOP              enum StandardPath,1
SP_APPLICATION      enum StandardPath
SP_DOCUMENT         enum StandardPath
SP_SYSTEM           enum StandardPath
SP_PRIVATE_DATA     enum StandardPath
SP_STATE            enum StandardPath
SP_FONT             enum StandardPath
SP_SPOOL            enum StandardPath
SP_SYS_APPLICATION  enum StandardPath
SP_USER_DATA        enum StandardPath
SP_MOUSE_DRIVERS    enum StandardPath
SP_PRINTER_DRIVERS  enum StandardPath
SP_FILE_SYSTEM_DRIVERS  enum StandardPath
SP_VIDEO_DRIVERS    enum StandardPath
SP_SWAP_DRIVERS     enum StandardPath
SP_KEYBOARD_DRIVERS enum StandardPath
SP_FONT_DRIVERS     enum StandardPath
SP_IMPORT_EXPORT_DRIVERS  enum StandardPath
SP_TASK_SWITCH_DRIVERS  enum StandardPath
SP_HELP_FILES       enum StandardPath
SP_TEMPLATE         enum StandardPath
SP_POWER_DRIVERS    enum StandardPath
SP_DOS_ROOM         enum StandardPath
SP_HWR              enum StandardPath
SP_WASTE_BASKET     enum StandardPath
SP_BACKUP           enum StandardPath
SP_PAGER_DRIVERS    enum StandardPath

```

SP_TEMP_FILES Sequ SP_WASTE_BASKET

SP_NOT_STANDARD_PATH
Not a standard path.

SP_TOP Top level directory. (Location of GEOS.EXE and GEOS.INI files.)
Generally C:/GEOWORKS.

SP_APPLICATION
Application directory. (Location of all applications.) Default is
WORLD.

SP_DOCUMENT
Document directory. (Location of all application datafiles.)
Default is DOCUMENT.

SP_SYSTEM System directory. (Location of drivers, libraries, TOKEN.DB.)
Default is SYSTEM.

SP_PRIVATE_DATA	Private data. Default is PRIVDATA.
SP_STATE	State directory. (Location of state files.) Default is PRIVDATA/STATE.
SP_FONT	Font directory. (Location of all fonts.) Default is USERDATA/FONT.
SP_SPOOL	Spool directory. (Location of application spool files.) Default is PRIVDATA/SPOOL.
SP_SYS_APPLICATION	Secondary application directory. (Location of GCM apps, welcome, applications that should not be launched by the user.) Default is SYSTEM/SYSAPPL.
SP_USER_DATA	Public data. Default is USERDATA
SP_MOUSE_DRIVERS	Mouse drivers. Default is SYSTEM/MOUSE
SP_PRINTER_DRIVERS	Printer drivers. Default is SYSTEM/PRINTER.
SP_FILE_SYSTEM_DRIVERS	File system drivers. Default is SYSTEM/FS.
SP_VIDEO_DRIVERS	Video drivers. Default is SYSTEM/VIDEO.
SP_SWAP_DRIVERS	Swap drivers. Default is SYSTEM/SWAP.
SP_KEYBOARD_DRIVERS	Keyboard drivers. Default is SYSTEM/KBD.
SP_FONT_DRIVERS	Font drivers. Default is SYSTEM/FONT.
SP_IMPORT_EXPORT_DRIVERS	Import/export libraries. Default is SYSTEM/IMPEX.
SP_TASK_SWITCH_DRIVERS	Task-switching drivers. Default is SYSTEM/TASK.
SP_HELP_FILES	Help files. Default is USERDATA/HELP.

SP_TEMPLATE
Template files. Default is USERDATA/TEMPLATE.

SP_POWER_DRIVERS
Power-management drivers. Default is SYSTEM/POWER.

SP_DOS_ROOM
Where DOS Launchers default to and where Welcome looks to give user a list of buttons. Default is DOSROOM.

SP_HWR
HandWritingRecognition drivers. Default is SYSTEM/HWR.

SP_WASTE_BASKET
This is where discarded files go. Default is PRIVDATA/WASTE.

SP_BACKUP
This is where backup files go. Default is PRIVDATA/BACKUP.

Library: **file.def**

■ StandardPathByte

```
StandardPathByte    record
    SPB_SP            StandardPath:8
StandardPathByte    end
```

Library: **file.def**

■ StandardSoundType

```
StandardSoundType    etype word
    SST_ERROR          enum StandardSoundType
    SST_WARNING         enum StandardSoundType
    SST_NOTIFY         enum StandardSoundType
    SST_NO_INPUT       enum StandardSoundType
    SST_KEY_CLICK      enum StandardSoundType
    SST_ALARM          enum StandardSoundType
    SST_CUSTOM_SOUND   equ 0xffffd
    SST_CUSTOM_BUFFER  equ 0xffffe
    SST_CUSTOM_NOTE    equ 0xfffff
```

SST_ERROR
Sound produced when an Error box comes up.

SST_WARNING
General warning beep sound.

SST_NOTIFY
General notify beep.



SST_NO_INPUT

Sound produced when the users keystrokes/mouse presses are not going anywhere (if the user clicks off a modal dialog box, or clicks on the field or something).

SST_KEY_CLICK

Sound produced when the keyboard is pressed, or when the user clicks on a floating keyboard.

SST_CUSTOM_SOUND

Allows applications to play a custom sound handle and does all the checking for sound being off, etc. This is not a part of the enumerated type to simplify error checking later.

SST_CUSTOM_BUFFER

Allows applications to play a custom sound buffer and does all the checking for sound being off, etc. This is not a part of the enumerated type to simplify error checking later.

SST_CUSTOM_NOTE

Allows applications to play a custom note and does all the checking for sound being off, etc. This is not a part of the enumerated type to simplify error checking later.

All sounds are given the following defaults:

TEMPO = 1 msec per 64th note
PRIORITY = SYSTEM_IMMEDIATE.

Library: **ui.def**

■ StartUndoChainStruct

```
StartUndoChainStruct    struct
    SUCS_owner          optr
    SUCS_title          optr
StartUndoChainStruct    ends
```

SUCS_owner stores the owner of this action.

SUCS_title stores the null-terminated title of this action. If null, then the title of the undo action will be the title passed with the next MSG_GEN_PROCESS_UNDO_START_CHAIN.

Library: **Objects/gProcC.def**

■ SubscriptPosition

```
SubscriptPosition    etype byte
    SBP_CHEMICAL      enum SubscriptPosition, 30
    SBP_DENOMINATOR   enum SubscriptPosition, 0
    SBP_DEFAULT       enum SubscriptPosition, 50
```

Library: **font.def**

■ SubscriptSize

```
SubscriptSize        etype byte
    SBS_CHEMICAL      enum SubscriptSize, 65
    SBS_DENOMINATOR   enum SubscriptSize, 60
    SBS_DEFAULT       enum SubscriptSize, 50
```

Library: **font.def**

■ SuperscriptPosition

```
SuperscriptPosition  etype byte
    SPP_DISPLAY       enum SuperscriptPosition, 50
    SPP_FOOTNOTE      enum SuperscriptPosition, 40
    SPP_ALPHA         enum SuperscriptPosition, 45
    SPP_NUMERATOR     enum SuperscriptPosition, 50
    SPP_DEFAULT       enum SuperscriptPosition, 50
```

Library: **font.def**

■ SuperscriptSize

```
SuperscriptSize      etype byte
    SPS_DISPLAY       enum SuperscriptSize, 55
    SPS_FOOTNOTE      enum SuperscriptSize, 65
    SPS_ALPHA         enum SuperscriptSize, 75
    SPS_NUMERATOR     enum SuperscriptSize, 60
    SPS_DEFAULT       enum SuperscriptSize, 50
```

Library: **font.def**



■ SysConfigFlags

```

SysConfigFlags      record
    SCF_UNDER_SWAT   :1,          ; Non-zero if kernel started by Swat stub
    SCF_2ND_IC       :1,          ; Non-zero if second 8259 present
    SCF_RTC          :1,          ; Non-zero if real-time clock around
    SCF_COPROC       :1,          ; Non-zero if math coprocessor present
    SCF_RESTARTED    :1,          ; Non-zero if restarted from our tsr
    SCF_CRASHED      :1,          ; Non-zero if we crashed the last time we ran
    SCF_MCA          :1,          ; Non-zero if we're on a Micro Channel machine
    SCF_LOGGING      :1,          ; Non-zero if we're writing log messages
SysConfigFlags      end

```

Library: **system.def**

■ SysDrawMask

```

SysDrawMask      record
    SDM_INVERSE    :1,          ; bit 7: 0 for mask as is
                                ; 1 for inverse of mask
    SDM_MASK       SystemDrawMask:7 ; bits 6-0: draw mask number
                                ; 0x7f to set custom mask
SysDrawMask      end

```

Library: **graphics.def**

■ SysGetInfoType

```

SysGetInfoType    etype word, 0, 2
    SGIT_TOTAL_HANDLES      enum SysGetInfoType
    SGIT_HEAP_SIZE          enum SysGetInfoType
    SGIT_LARGEST_FREE_BLOCK enum SysGetInfoType
    SGIT_TOTAL_COUNT        enum SysGetInfoType
    SGIT_NUMBER_OF_VOLUMES  enum SysGetInfoType
    SGIT_TOTAL_GEODES       enum SysGetInfoType
    SGIT_NUMBER_OF_PROCESSES enum SysGetInfoType
    SGIT_NUMBER_OF_LIBRARIES enum SysGetInfoType
    SGIT_NUMBER_OF_DRIVERS  enum SysGetInfoType
    SGIT_CPU_SPEED          enum SysGetInfoType
    SGIT_SYSTEM_DISK        enum SysGetInfoType
    SGIT_UI_PROCESS         enum SysGetInfoType

```

Library: **sysstats.def**

■ SysInitialTextMode

```

SysInitialTextMode      etype    byte, 0
    SITM_UNKNOWN          enum    SysInitialTextMode, 0
    SITM_TEXT_80_25_16_COLOR      enum    SysInitialTextMode, 3
    SITM_TEXT_80_25_MONO          enum    SysInitialTextMode, 7

```

Library: **system.def**

■ SysMachineType

```

SysMachineType          etype byte, 0
    SMT_UNKNOWN          enum SysMachineType
    SMT_PC               enum SysMachineType
    SMT_PC_CONV          enum SysMachineType
    SMT_PC_JR            enum SysMachineType
    SMT_PC_XT            enum SysMachineType
    SMT_PC_XT_286        enum SysMachineType
    SMT_PC_AT            enum SysMachineType
    SMT_PS2_30           enum SysMachineType
    SMT_PS2_50           enum SysMachineType
    SMT_PS2_60           enum SysMachineType
    SMT_PS2_80           enum SysMachineType
    SMT_PS1              enum SysMachineType

```

Library: **system.def**

■ SysNotifyFlags

```

SysNotifyFlags          record
    SNF_RETRY            :1,      ; Retry the operation.
    SNF_EXIT             :1,      ; Shutdown the system.
    SNF_ABORT            :1,      ; Abort the operation.
    SNF_CONTINUE         :1,      ; Continue when done. This is different from
                                ; SNF_RETRY as it implies the notification is
                                ; not for a real error, but just to notify
                                ; the user of something.
    SNF_REBOOT           :1,      ; Hard exit -- dirty shutdown followed by
                                ; reload/restart of GEOS
    SNF_BIZARRE          :1,      ; Indicates notice is unexpected and user
                                ; should be directed to the trouble-shooting
                                ; guide.
                                :10
SysNotifyFlags          end

```

Library: **system.def**



■ SysProcessorType

```
SysProcessorType    etype byte, 0
    SPT_8088        enum SysProcessorType
    SPT_8086        enum SysProcessorType, SPT_8088
    SPT_80186       enum SysProcessorType
    SPT_80286       enum SysProcessorType
    SPT_80386       enum SysProcessorType
    SPT_80486       enum SysProcessorType
```

Library: **system.def**

■ SysShutdownType

```
SysShutdownType    etype word
    SST_CLEAN       enum SysShutdownType
    SST_CLEAN_FORCEDenum SysShutdownType
    SST_DIRTY       enum SysShutdownType
    SST_PANIC       enum SysShutdownType
    SST_REBOOT      enum SysShutdownType
    SST_RESTART     enum SysShutdownType
    SST_FINAL       enum SysShutdownType
    SST_SUSPEND     enum SysShutdownType
    SST_CONFIRM_STARTenum SysShutdownType
    SST_CONFIRM_END enum SysShutdownType
```

Note: SysNotify depends on these things increasing in severity as the number increases. Place any new modes in the proper order.

SST_CLEAN

Shut down applications cleanly, allowing ones that wish to abort the shutdown to do so.

MSG_META_CONFIRM_SHUTDOWN is sent out via the MANUFACTURER_ID_GEOWORKS: GCNSLT_SHUTDOWN_CONTROL list.

Pass:

^|cx:dx= object to notify when everything's been confirmed; or 0:0 to simply notify the UI in the standard fashion (via MSG_META_DETACH).

bp= message to send it. When the message is sent, cx will be 0 if the shutdown request has been denied; non-zero if the shutdown may proceed.

Return: carry set if another shutdown is already in-progress

SST_CLEAN_FORCED

Shut down applications cleanly, but do not send out MSG_META_CONFIRM_SHUTDOWN.

Reference book

Pass:nothing.

Return:nothing.

SST_DIRTY

Do not shut down applications. Attempt to exit device drivers and close all open files, however.

Pass:

ds:si= reason for the shutdown (null-terminated string).

si = -1 if no reason to give the user.

Return:doesn't.

SST_PANIC

Do not shut down applications. Do not close files. Only exit device drivers marked with GA_SYSTEM. This can be really bad for the system and should be used only in dire straits.

Pass:nothing.

Return:doesn't.

SST_REBOOT

Like SST_DIRTY, but warm-boots the machine, rather than just exiting to DOS.

Pass:nothing.

Return:doesn't.

SST_RESTART

Like SST_CLEAN_FORCED, but reload the system, rather than exiting to DOS.

Pass:nothing.

Return:only if couldn't set up for restart (e.g. loader.exe wasn't found).

SST_FINAL

Perform the final phase of an SST_CLEAN or SST_CLEAN_FORCED shutdown.

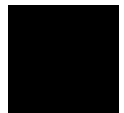
Pass:

ds:si= reason for shutdown (**si** = -1 if no reason to give).

Return:doesn't.

SST_SUSPEND

Suspend system operation in preparation for switching to a new DOS task. Broadcasts MSG_META_CONFIRM_SHUTDOWN



through the system's MANUFACTURER_ID_GEOWORKS:
GCNSLT_SHUTDOWN_CONTROL list.

Pass:

^l**cx:dx**= object to notify when everything's been confirmed
bp= message to send it. When the message is sent, **cx** will be 0
if the shutdown request has been denied; non-zero if the
shutdown may proceed.

Return:carry set if another shutdown is already in-progress.

SST_CONFIRM_START

Called by the recipient of a MSG_META_CONFIRM_SHUTDOWN
so there's some order to the way confirmation boxes are
presented to the user. Only one thread may be confirming the
shutdown at a time. The caller will block until it is given
permission. **SysShutdown** will return carry set to indicate
that some other thread has already canceled the shutdown and
the caller should *not* put up its confirmation box. It need not
call **SysShutdown** again.

Pass:nothing.

Return:carry set if some other object has already denied the
shutdownrequest. Caller should do nothing further.

SST_CONFIRM_END

Finishes the handling of a MSG_META_CONFIRM_SHUTDOWN.

Pass: **cx**= 0 to deny the shutdown.
= non-zero to allow the shutdown.

Return: nothing.

Library: **system.def**

■ SysSimpleGraphicsMode

SysSimpleGraphicsMode	etype	byte, 0
SSGM_NONE	enum	SysSimpleGraphicsMode, 0
SSGM_VGA	enum	SysSimpleGraphicsMode, 1
SSGM_EGA	enum	SysSimpleGraphicsMode, 2
SSGM_MCGA	enum	SysSimpleGraphicsMode, 3
SSGM_HGC	enum	SysSimpleGraphicsMode, 4
SSGM_CGA	enum	SysSimpleGraphicsMode, 5
SSGM_SPECIAL	enum	SysSimpleGraphicsMode, 6
SSGM_SVGA_VESA	enum	SysSimpleGraphicsMode, 7

Library: **system.def**

■ SysStats

```
SysStats      struct
    SS_idleCount    dword
    SS_swapOuts     SysSwapInfo
    SS_swapIns      SysSwapInfo
    SS_contextSwitches word
    SS_interrupts   word
    SS_runQueue     word
SysStats      ends
```

This structure is returned by **SysStatistics** and stores the current performance statistics of GEOS.

SS_idleCount stores the number of idle ticks during the last second.

SS_swapOuts stores the outward-bound swapping activity (**SysSwapInfo**).

SS_swapIns stores the inward-bound swapping activity (**SysSwapInfo**).

SS_contextSwitches stores the number of context switches that occurred during the last second.

SS_interrupts stores the number of interrupts that occurred during the last second.

SS_runQueue stores the number of runnable threads at the end of the last second.

Library: **sysstats.def**

■ SysSwapInfo

```
SysSwapInfo    struct
    SSI_paragraphs word
    SSI_blocks     word
SysSwapInfo    ends
```

This structure stores the current swap activity of the system. This swap information is used in the **SysStats** structure.

SSI_paragraph stores the number of “paragraphs” swapped.

SSI_blocks stores the number of blocks swapped.

Library: **sysstats.def**



■ **SystemAttrs**

```
SystemAttrs      record
  SA_NOT          :1          ;Any following set bits must be OFF for hints
                               ;to be included, rather than on.
  SA_TINY          :1          ;If set, screen must be either horizontally or
                               ;vertically tiny for hints to be included.
  SA_HORIZONTALLY_TINY:1      ;If set, screen must be horizontally tiny for
                               ;hints to be included.
  SA_VERTICALLY_TINY:1        ;If set, screen must be vertically tiny for
                               ;hints to be included.
  SA_COLOR         :1          ;If set, must be a color screen for hints to
                               ;be included.
  SA_PEN_BASED     :1          ;If set, system must be pen based for hints
                               ;to be included.
  SA_KEYBOARD_ONLY :1          ;If set, system must be set keyboard-only for
                               ;hints to be included.
  SA_NO_KEYBOARD   :1          ;If set, system must be set no-keyboard for
                               ;hints to be included.
                               :8
SystemAttrs      end
```

Library: **genC.def**

■ **SystemBitmap**

```
SystemBitmap      etype byte
```

Library: **graphics.def**

■ SystemDrawMask

```
SystemDrawMask      etype byte
SDM_TILE            enum SystemDrawMask ; tile pattern
SDM_SHADED_BAR      enum SystemDrawMask ; shaded bar
SDM_HORIZONTAL      enum SystemDrawMask ; horizontal lines
SDM_VERTICAL        enum SystemDrawMask ; vertical lines
SDM_DIAG_NE         enum SystemDrawMask ; diagonal lines going up to NorthEast
SDM_DIAG_NW         enum SystemDrawMask ; diagonal lines going up to NorthWest
SDM_GRID            enum SystemDrawMask ; checkerboard
SDM_BIG_GRID        enum SystemDrawMask ; larger checkerboard
SDM_BRICK            enum SystemDrawMask ; brick wall
SDM_SLANT_BRICK     enum SystemDrawMask ; slanted brick wall

SDM_0               enum SystemDrawMask, 89      ; all zeroes
SDM_12_5            enum SystemDrawMask, 81
SDM_25              enum SystemDrawMask, 73
SDM_37_5            enum SystemDrawMask, 65
SDM_50              enum SystemDrawMask, 57
SDM_62_5            enum SystemDrawMask, 49
SDM_75              enum SystemDrawMask, 41
SDM_87_5            enum SystemDrawMask, 33
SDM_100             enum SystemDrawMask, 25      ; all ones
SDM_CUSTOM          enum SystemDrawMask, 0x7f    ; setting a custom mask
SET_CUSTOM_PATTERN=SDM_CUSTOM
```

Library: **graphics.def**

■ SystemHatch

```
SystemHatch         etype byte
SH_VERTICAL          enum SystemHatch ; vertical lines
SH_HORIZONTAL        enum SystemHatch ; horizontal lines
SH_45_DEGREE         enum SystemHatch ; lines at 45 degrees
SH_135_DEGREE        enum SystemHatch ; lines at 135 degrees
SH_BRICK             enum SystemHatch ; basic brick
SH_SLANTED_BRICK     enum SystemHatch ; basic brick, slanted
```

Library: **graphics.def**



■ **SystemVMID**

```

SystemVMID      etype word, 0xff00
    SVMID_RANGE_DBASE      equ 0xff00      ;Reserved for DB code

    DB_MAP_ID              enum SystemVMID  ; ID for DB map block
    DB_GROUP_ID            enum SystemVMID  ; ID for new DB group
    DB_ITEM_BLOCK_ID       enum SystemVMID  ; ID for new DB item block
    SVMID_HA_DIR_ID        enum SystemVMID  ; ID for HugeArray dir blocks
    SVMID_HA_BLOCK_ID      enum SystemVMID  ; ID for HugeArray data blocks

```

Library: **vm.def**

■ Tab

```

Tab          struct
  T_position  word          ; Position of tab (pixels * 8)
  T_attr      TabAttributes ; Tab attributes.
  T_grayScreen SysDrawMask  ; Gray screen for tab lines
  T_lineWidth byte          ; Width of line before (after) tab
                          ; 0 = none, units are pixels * 8
  T_lineSpacing byte        ; Space between tab and line
                          ; 0 = none, units are pixels * 8
  T_anchor    word          ; Anchor character.
Tab          ends

      Library:  text.def

```

■ TabAttributes

```

TabAttributes      record
                  :3
  TA_LEADER        TabLeader:3
  TA_TYPE          TabType:2
TabAttributes      end

      Library:  text.def

```

■ TabLeader

```

TabLeader          etype byte
  TL_NONE          enum TabLeader
  TL_DOT           enum TabLeader
  TL_LINE          enum TabLeader
  TL_BULLET        enum TabLeader

      Library:  text.def

```

■ TabReference

```

TabReference      record
  TR_TYPE          TabReferenceType:1 ; Type of reference.
  TR_REF_NUMBER    :7                ; Reference number
TabReference      end

      Library:  text.def

```



■ TabReferenceType

```
TabReferenceType    etype byte
    TRT_RULER      enum TabReferenceType    ; Reference is into the ruler.
    TRT_OTHER      enum TabReferenceType
```

Library: **text.def**

■ TabType

```
TabType    etype byte
    TT_LEFT      enum TabType
    TT_CENTER    enum TabType
    TT_RIGHT     enum TabType
    TT_ANCHORED  enum TabType
```

Library: **text.def**

■ TargetLevel

```
TargetLevel          etype word
    TL_TARGET         enum TargetLevel, 0
    TL_CONTENT        enum TargetLevel
    TL_GENERIC_OBJECTS enum TargetLevel, 1000
    TL_GEN_SYSTEM     enum TargetLevel
    TL_GEN_FIELD      enum TargetLevel
    TL_GEN_APPLICATION enum TargetLevel
    TL_GEN_PRIMARY    enum TargetLevel
    TL_GEN_DISPLAY_CTRL enum TargetLevel
    TL_GEN_DISPLAY    enum TargetLevel
    TL_GEN_VIEW       enum TargetLevel
    ;
```

```
    ; Place PC/GEOS library extensions here
    TL_LIBRARY_LEVELSenum TargetLevel, 2000
```

```
    ; EXPORTED FOR INDIVIDUAL APPLICATIONS
    TL_APPLICATION_OBJECTSenum TargetLevel, 3000
```

TL_TARGET Final target object. (Currently, just text objects, such as VisText, GenTextDisplay or GenText).

TL_CONTENT Content within view (generic, visual, or whatever).

TL_GENERIC_OBJECTS

TL_GEN_SYSTEM The system object itself.

TL_GEN_FIELD
Field within system.

TL_GEN_APPLICATION
Application within field.

TL_GEN_PRIMARY
Primary within application.

TL_GEN_DISPLAY_CTRL
Display control within primary.

TL_GEN_DISPLAY
Display within display control.

TL_GEN_VIEW
View within display.

Library: **Objects/genC.def**

■ **TargetReference**

```
TargetReference      struct
  TR_object    optr      ; OD of node/leaf in target hierarchy
  TR_class     fptr      ; class of above object
TargetReference      ends
```

Library: **Objects/gViewC.def**

■ **TCCFeatures**

```
TCCFeatures          record
  TCCF_CHARACTER    :1
  TCCF_WORD         :1
  TCCF_LINE         :1
  TCCF_PARAGRAPH    :1
  TCCF_RECALC       :1
TCCFeatures          end
```

Library: **Objects/Text/tCtrlC.def**

■ **TCCToolboxFeatures**

```
TCCToolboxFeatures  record
TCCToolboxFeatures  end
```

Library: **Objects/Text/tCtrlC.def**



■ TCFeatures

```
TCFeatures          record
    TCF_LIST        :1
    TCF_POSITION    :1
    TCF_GRAY_SCREEN :1
    TCF_TYPE        :1
    TCF_LEADER      :1
    TCF_LINE        :1
    TCF_CLEAR       :1
    TCF_CLEAR_ALL   :1
TCFeatures          end
```

Library: **Objects/Text/tCtrlC.def**

■ TCToolboxFeatures

```
TCToolboxFeatures  record
TCToolboxFeatures  end
```

Library: **Objects/Text/tCtrlC.def**

■ TempGenControlInstance

```
TempGenControlInstance  struct
    TGCI_interactableFlags  GenControlInteractableFlags
    TGCI_childBlock         hpPtr
    TGCI_toolBlock          hpPtr
    TGCI_toolParent         optr
    TGCI_features           word
    TGCI_toolboxFeatures    word
    TGCI_activeNotificationList  GCNListType
    TGCI_upToDate           GenControlInteractableFlags
TempGenControlInstance  ends
```

TGCI_interactableFlags holds the current status of various portions of the controller (the entire controller object itself, its associated toolbox, or its associated “normal” UI). These flags define which portions of the controller are interactable by the user. These bits may be changed by the default handlers for MSG_GEN_CONTROL_NOTIFY_INTERACTABLE and MSG_GEN_CONTROL_NOTIFY_NOT_INTERACTABLE only. If any bits become set, the controller adds itself to the notification list, so that it will be able to update the interactable areas. The controller then remains on the lists until all bits become clear.

TGCI_toolParent stores the object passed to the MSG_GEN_CONTROL_GENERATE_TOOLBOX_UI method. This object is the object that the tools were added to at that point.

TGCI_upToDate holds the status of the **GenControlInteractableFlags** in *TGCI_interactableFlags* at the point that the last notification update came in. This defines what portions of the controller's UI were up to date at that last notification period. This way, if some part of the UI becomes non-interactable, then interactable again before another update, we can detect this scenario and avoid a redundant update.

Library: **Objects/gCtrlC.def**

■ TempGenToolControlInstance

```
TempGenToolControlInstance    struct
    TGTCI_curController      optr
    TGTCI_features           word
    TGTCI_required           word
    TGTCI_allowed            word
    TGTCI_curToolGroup       optr
    TGTCI_toolGroupVisible   byte
TempGenToolControlInstance    ends
```

TGTCI_curController stores the optr of the controller whose tool options and placement location are currently being displayed for editing by the user.

TGTCI_features stores the mask of currently active features.

TGTCI_required stores the mask of features which must always be active, i.e. can't be "hidden" by the user.

TGTCI_allowed stores the mask of features which controller and application together will allow the user to access. Bits set here but not in "*TGTCI_features*" will appear in the "hidden" list.

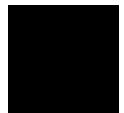
TGTCI_curToolGroup stores the currently selected tool group.

TGTCI_toolGroupVisible stores a non-zero value if a tool group list is visible. If visible, all tool groups are highlighted, and the current one "selected" to bring it to the attention of the user.

Library: **Objects/gToolCC.def**

■ TempImportExportData

```
TempImportExportData    struct
    TIED_formatUI        optr    ; OD of duplicated format UI
    TIED_formatLibrary   hptr    ; handle of library for above
TempImportExportData    ends
```



Library: **impex.def**

■ **TempMetaGCNData**

```
TempMetaGCNData    struct
    TMGCND_listOfLists    lptr.GCNListOfListsHeader
    TMGCND_flags          TempMetaGCNFlags
TempMetaGCNData    ends
```

TMGCND_listOfLists stores the chunk handle holding the GCN list of lists.

Library: **Objects/metaC.def**

■ **TempMetaGCNFlags**

```
TempMetaGCNFlags    record
    TMGCNF_RELOCATED    :1    ; set if relocated
                        :7
TempMetaGCNFlags    end
```

Library: **Objects/metaC.def**

■ **TempPrintCtrlInstance**

```
TempPrintCtrlInstance    struct
    TPCI_currentSummons    optr    ; currently active summons
    TPCI_progressBox        optr    ; OD of progress dialog box
    TPCI_jobParHandle        hptr    ; memory handle to JobParameters
    TPCI_fileHandle        word    ; file handle (if printing)
    TPCI_gstringHandle        word    ; gstring handle (if printing)
    TPCI_printBlockHan        word    ; the printer block handle
    TPCI_attrs                PrintControlAttrs
    TPCI_status                PrintStatusFlags
    TPCI_holdUpCompletionCount    byte; Number of things not wanting the message
                                ; stored in TEMP_PRINT_COMPLETION_EVENT to
                                ; be sent out just yet.
TempPrintCtrlInstance    ends
```

Library: **spool.def**

■ **TestRectReturnType**

```
TestRectReturnType    etype byte
    TRRT_OUT            enum TestRectReturnType
    TRRT_PARTIAL        enum TestRectReturnType
    TRRT_IN              enum TestRectReturnType
```

Library: **graphics.def**

■ TextArrayType

```
TextArrayType      etype byte
    TAT_CHAR_ATTRS      enum TextArrayType
    TAT_PARA_ATTRS      enum TextArrayType
    TAT_GRAPHICS        enum TextArrayType
    TAT_TYPES            enum TextArrayType
```

Library: **Objects/vTextC.def**

■ TextAttr

```
TextAttr          struct
    TA_color       ColorQuad      ; RGB values or index
    TA_mask        SystemDrawMask ; draw mask
    TA_pattern     GraphicPattern ; pattern
    TA_styleSet    TextStyle      ; text style bits to set
    TA_styleClear  TextStyle      ; text style bits to clear
    TA_modeSet     TextMode       ; text mode bits to set
    TA_modeClear   TextMode       ; text mode bits to clear
    TA_spacePad    WBFixed        ; space padding
    TA_font        FontID         ; typeface
    TA_size        WBFixed        ; point size
    TA_trackKern   sword          ; track kerning
    TA_fontWeight  FontWeight     ; weight of font
    TA_fontWidth   FontWidth      ; width of font
    align         word
TextAttr          ends
```

This structure is used with **GrSetTextAttr** and **GrDrawTextField**.

Library: **graphics.def**

■ TextClipboardOption

```
TextClipboardOption etype word
    TCO_COPY          enum TextClipboardOption
    TCO_RETURN_TRANSFER_FORMAT enum TextClipboardOption
    TCO_RETURN_TRANSFER_ITEM  enum TextClipboardOption
    TCO_RETURN_NOTHING enum TextClipboardOption
```

Library: **Objects/vTextC.def**



■ **TextColors**

```
TextColors      struc
    TC_unselectedColor    byte
    TC_selectedColor      byte
TextColors      ends
```

Library: **genC.def**

■ **TextElementArrayHeader**

```
TextElementArrayHeader  struct
    TEAH_meta      ElementArrayHeader
    TEAH_arrayType  TextArrayType
    TEAH_unused     byte
TextElementArrayHeader  ends
```

Library: **Objects/vTextC.def**

■ **TextFocusFlags**

```
TextFocusFlags record      ;Record passed in BP
    TFF_EDITABLE_TEXT_OBJECT_HAS_FOCUS      :1
                                           ; Set if an editable text object has the focus
    TFF_OBJECT_RUN_BY_UI_THREAD              :1
                                           ; Set if the object is run by the UI thread
                                           :14
TextFocusFlags end
```

Library: **vTextC.def**

■ **TextGuardianFlags**

```
TextGuardianFlags  record
    TGF_ENFORCE_DESIRED_MIN_HEIGHT          :1
    TGF_ENFORCE_DESIRED_MAX_HEIGHT          :1
    TGF_DISABLE_ENFORCED_DESIRED_MAX_HEIGHT_WHILE_EDITING:1
    TGF_ENFORCE_MIN_DISPLAY_SIZE            :1
    TGF_SHRINK_WIDTH_TO_MIN_AFTER_EDIT      :1
TextGuardianFlags  end
```

TGF_ENFORCE_DESIRED_MIN_HEIGHT
If true then text object will not shrink below the
desiredMinHeight while it is being edited or when some
attribute changes.

TGF_ENFORCE_DESIRED_MAX_HEIGHT

If true then text object will not expand above the desiredMaxHeight while it is being edited or when some attribute changes.

TGF_DISABLE_ENFORCED_DESIRED_MAX_HEIGHT_WHILE_EDITING

If true, text object can grow beyond desiredMaxHeight during editing, but when the object stops being edited it will shrink back to desiredMaxHeight. This flag is meaningless if TGF_ENFORCE_DESIRED_MAX_HEIGHT is not set.

TGF_ENFORCE_MIN_DISPLAY_SIZE

If true, then during resize don't allow text object to become shorter than is necessary to display all the text.

TGF_SHRINK_WIDTH_TO_MIN_AFTER_EDIT

If true then when the text object loses the edit grab shrink the width to minimum needed to hold the text. Used when user clicks and releases in the same spot to create.

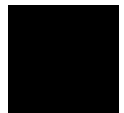
Library: **grobj.def**

■ TextLargeRunArrayHeader

```
TextLargeRunArrayHeader  struct
    TLRAH_meta             HugeArrayDirectory
    TLRAH_elementVMBlock   word    ; Element block (or null)
TextLargeRunArrayHeader  ends
```

This structure stores a generic array of runs in the large text format.

Library: **Objects/vTextC.def**



■ TextMetricStyles

```
TextMetricStyles    struct
    TMS_styleCallback    fptr.far    ;style callback routine
    ;
    ;    PASS:    ss:bx    = TOC_vars
    ;            di      = Offset into the field
    ;            ds      = Segment address of old text pointer
    ;    RETURN: TMS_textAttr set
    ;            ds:si    = Pointer to the text
    ;            cx      = Number of characters in this style
    ;    DESTROYED:    nothing
    ;
    TMS_graphicCallback    fptr.far    ;graphic callback routine
    ;
    ;    PASS:    ss:bx    = L1CL_vars
    ;            di      = Offset into field
    ;            ds      = Segment address of text pointer
    ;    RETURN: cx      = Height of the graphic of graphic
    ;                    at current position
    ;            dx      = Width of the graphic
    ;    DESTROYED:    nothing
    ;
    TMS_fieldStart        dword
    TMS_sizeSoFar        WBFixed
    TMS_lastCharWidth    WBFixed
    TMS_textAttr        TextAttr
    TMS_fontHandle        hptr.FontBuf
    TMS_trackKernValue    BBFixed
    TMS_flags            TMSFlags
    TMS_gstateHandle    hptr.GState
    TMS_gstateSegment    word
    TMS_styleHeight    WBFixed
    TMS_styleBaseline    WBFixed
TextMetricStyles    ends
```

Library: **text.def**

■ TextMode

```
TextMode          record
    TM_DRAW_CONTROL_CHARS    :1      ; Does the following mapping when drawing
                                   ; text:
                                   ; C_SPACE      -> C_CNTR_DOT
                                   ; C_NONBRKSPACE -> C_CNTR_DOT
                                   ; C_CR         -> C_PARAGRAPH
                                   ; C_TAB        -> C_LOGICAL_NOT
                                   ;
    TM_TRACK_KERN            :1      ;internal only - not settable
    TM_PAIR_KERN             :1      ;internal only - not settable
    TM_PAD_SPACES            :1      ;internal only - not settable
    TM_DRAW_BASE             :1
    TM_DRAW_BOTTOM           :1
    TM_DRAW_ACCENT           :1
    TM_DRAW_OPTIONAL_HYPHENS :1
TextMode          end
```

Library: **graphics.def**

■ TextReference

```
TextReference      struct
    TR_type        TextReferenceType
    TR_ref         TextReferenceUnion
TextReference      ends
```

Library: **Objects/vTextC.def**

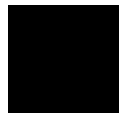
■ TextReferenceBlock

```
TextReferenceBlock struct
    TRB_handle      hptr.char
TextReferenceBlock ends
```

This structure corresponds to a **TextReferenceType** of TRT_BLOCK. It is used with MSG_VIS_TEXT_REPLACE_TEXT and MSG_VIS_TEXT_GET_TEXT_RANGE to reference text used by those messages.

TRB_handle stores the handle of the text buffer. No entries need to be filled in to allocate a destination buffer. The heap allocation request will be made with the HAF_NO_ERR flag. If VTGRF_RESIZE is passed then either the passed block or the allocated block will be resized to accommodate the text.

Library: **Object/vTextC.def**



■ TextReferenceBlockChunk

```
TextReferenceBlockChunk    struct
    TRBC_ref    optr.char
TextReferenceBlockChunk    ends
```

This structure corresponds to a **TextReferenceType** of TRT_OPTR. It is used with MSG_VIS_TEXT_REPLACE_TEXT and MSG_VIS_TEXT_GET_TEXT_RANGE to reference text used by those messages.

TRBC_ref stores the optr to a text buffer (a group of character)s. The handle field of *TRBC_ref* must be filled in.

It is assumed that the LMem heap will be able to accommodate this allocation. The caller is responsible for ensuring that this is the case.

If VTGRF_RESIZE is passed then either the passed block or the allocated block will be resized to accommodate the text.

Library: **Objects/vTextC.def**

■ TextReferenceDBItem

```
TextReferenceDBItem        struct
    TRDBI_file    hptr
    TRDBI_item    word
    TRDBI_group    word
TextReferenceDBItem        ends
```

This structure corresponds to a **TextReferenceType** of TRT_DB_ITEM. It is used with MSG_VIS_TEXT_REPLACE_TEXT and MSG_VIS_TEXT_GET_TEXT_RANGE to reference text used by those messages.

TRDBI_file stores the VM file associated with this DB item.

TRDBI_item stores the DB item itself.

TRDBI_group stores the DB group the item belongs to.

Both *TRDBI_file* and *TRDBI_group* must be filled in if you want a VM block to be allocated.

If the *TRDBI_group* field is set to DB_UNGROUPED then the item will be allocated ungrouped. *TRDBI_group* will hold the group in which the item was allocated on return.

If VTGRF_RESIZE is passed then either the passed block or the allocated block will be resized to accommodate the text.

Library: **Objects/vTextC.def**

■ TextReferenceHugeArray

```
TextReferenceHugeArray    struct
    TRHA_file             hptr
    TRHA_array            word
TextReferenceHugeArray    ends
```

This structure corresponds to a **TextReferenceType** of TRT_HUGE_ARRAY. It is used with MSG_VIS_TEXT_REPLACE_TEXT and MSG_VIS_TEXT_GET_TEXT_RANGE to reference text used by those messages.

TRHA_file stores the VM file associated with this huge array.

TRHA_array stores the Huge Array.

The *TRHA_file* field must be set if you want a huge-array to be allocated.

If VTGRF_RESIZE is passed then either the passed block or the allocated block will be resized to accommodate the text.

Library: **Objects/vTextC.def**

■ TextReferencePointer

```
TextReferencePointer      struct
    TRP_pointer           fptr.char
TextReferencePointer      ends
```

This structure corresponds to a **TextReferenceType** of TRT_POINTER. It is used with MSG_VIS_TEXT_REPLACE_TEXT and MSG_VIS_TEXT_GET_TEXT_RANGE to reference text used by those messages.

TRP_pointer stores the pointer to the text. This field must be filled in.

VTGRF_RESIZE has no meaning with this sort of reference.

VTGRF_ALLOCATE and VTGRF_ALLOCATE_ALWAYS are not valid flags to pass with this type of text reference.

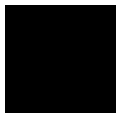
This reference is the safest way to copy text out of a text object. Since the caller allocates the block it can also handle errors in the allocation.

Library: **Objects/vTextC.def**

■ TextReferenceSegmentChunk

```
TextReferenceSegmentChunk struct
    TRSC_chunk            word
    TRSC_segment          word
TextReferenceSegmentChunk ends
```

This structure corresponds to a **TextReferenceType** of TRT_SEGMENT_CHUNK. It is used with MSG_VIS_TEXT_REPLACE_TEXT and MSG_VIS_TEXT_GET_TEXT_RANGE to reference text used by those messages.



TRSC_segment stores the segment address of the text chunk. *TRSC_chunk* stores the chunk offset to the text.

It is assumed that the LMem heap will be able to accommodate this allocation. The caller is responsible for ensuring that this is the case.

If VTGRF_RESIZE is passed then either the passed block or the allocated block will be resized to accommodate the text.

Library: **Objects/vTextC.def**

■ TextReferenceType

TextReferenceType etype word, 0, 2

TRT_POINTER	enum TextReferenceType
TRT_SEGMENT_CHUNK	enum TextReferenceType
TRT_OPTR	enum TextReferenceType
TRT_BLOCK	enum TextReferenceType
TRT_VM_BLOCK	enum TextReferenceType
TRT_DB_ITEM	enum TextReferenceType
TRT_HUGE_ARRAY	enum TextReferenceType

Library: **Objects/vTextC.def**

■ TextReferenceUnion

```
TextReferenceUnion union
    TRU_pointer      TextReferencePointer
    TRU_segChunk     TextReferenceSegmentChunk
    TRU_blockChunk   TextReferenceBlockChunk
    TRU_block        TextReferenceBlock
    TRU_vmBlock      TextReferenceVMBlock
    TRU_dbItem       TextReferenceDBItem
    TRU_hugeArray    TextReferenceHugeArray
TextReferenceUnion end
```

Library: **Objects/vTextC.def**

■ TextReferenceVMBlock

```
TextReferenceVMBlock struct
    TRVMB_file      hpPtr
    TRVMB_block     word
TextReferenceVMBlock ends
```

This structure corresponds to a **TextReferenceType** of TRT_VM_BLOCK. It is used with MSG_VIS_TEXT_REPLACE_TEXT and MSG_VIS_TEXT_GET_TEXT_RANGE to reference text used by those messages.

TRVMB_file stores the VM file associated with this VM block. This entry must be filled in if you want a VM block to be allocated.

TRVMB_block stores the VM block itself.

If VTGRF_RESIZE is passed then either the passed block or the allocated block will be resized to accommodate the text.

Library: **Objects/vTextC.def**

■ TextRulerAction

```
TextRulerAction      etype byte
    TRA_NULL          enum TextRulerAction
    TRA_MOVE_TAB      enum TextRulerAction
    TRA_COPY_TAB      enum TextRulerAction
    TRA_MOVE_MARGIN   enum TextRulerAction
```

Library: **Objects/Text/tCtrlC.def**

■ TextRulerControlAttributes

```
TextRulerControlAttributes  record
    TRCA_ROUND               :1
    TRCA_IGNORE_ORIGIN       :1
                                :14
TextRulerControlAttributes  end
```

Library: **Objects/Text/tCtrlC.def**

■ TextRulerFlags

```
TextRulerFlags      record
                                :3
    TRF_ALWAYS_MOVE_BOTH_MARGINS :1
    TRF_ROUND_COORDINATES        :1
    TRF_OBJECT_SELECTED          :1
    TRF_SELECTING                 :1
    TRF_DRAGGING                 :1
TextRulerFlags      end
```

Library: **Objects/Text/tCtrlC.def**



■ TextRunArrayElement

```
TextRunArrayElement    struct
    TRAE_position      WordAndAHalf <>    ; Position for start of run
    TRAE_token         word                ; Token for run
TextRunArrayElement    ends
```

This structure stores an element in an array of text runs.

Library: **Objects/vTextC.def**

■ TextRunArrayHeader

```
TextRunArrayHeader    struct
    TRAH_meta          ChunkArrayHeader
    TRAH_elementVMBlock word                ; Element block
    TRAH_elmentArray   lptr                ; ChunkHandle of element array
TextRunArrayHeader    ends
```

This structure stores the header of an array of runs (for non-LARGE text objects).

Library: **Objects/vTextC.def**

■ TextSearchInHugeArrayFrame

```
TextSearchInHugeArrayFrame    struct
    TSIHAF_str1Size      dword (?)
    TSIHAF_curOffset     dword (?)
    TSIHAF_endOffset     dword (?)
    TSIHAF_searchFlags   SearchOptions
    TSIHAF_hugeArrayVMFile  hptr
    TSIHAF_hugeArrayVMBlock hptr
    even
TextSearchInHugeArrayFrame    ends
```

TSIHAF_str1Size stores the total length of the string to search (str1).

TSIHAF_curOffset stores the offset (from the start of str1) to the first character to check.

TSIHAF_endOffset stores the offset (from the start of str1) to the last character to check. The text search will only match words that start at less than or equal to the character position in *TSIHAF_endOffset*. To check to the start of a string (backward searches only) pass 0:0. To check to the end of a string (forward searches only) pass *TSIHAF_str1Size*-1.

TSIHAF_hugeArrayVMFile and *TSIHAF_hugeArrayVMBlock* store the file and block handles for the huge array we will be extracting text from.

Library: **Objects/vTextC.def**

■ TextStyle

```

TextStyle          record
    :1              ; Do not use this bit.
    TS_OUTLINE      :1
    TS_BOLD          :1
    TS_ITALIC        :1
    TS_SUPERSCRIPT   :1
    TS_SUBSCRIPT     :1
    TS_STRIKE_THRU   :1
    TS_UNDERLINE     :1
TextStyle          end

```

Library: **graphics.def**

■ TextStyleElementHeader

```

TextStyleElementHeader struct
    TSEH_meta      NameArrayElement
    TSEH_baseStyle word
    TSEH_flags      StyleElementFlags
    TSEH_reserved   byte 6 dup (?)
    TSEH_privateData TextStylePrivateData
    TSEH_charAttrToken word
    TSEH_paraAttrToken word
    TSEH_name       label char
TextStyleElementHeader ends

```

Library: **Objects/vTextC.def**

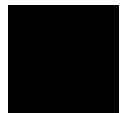
■ TextStyleFlags

```

TextStyleFlags      record
    TSF_APPLY_TO_SELECTION_ONLY :1
    TSF_POINT_SIZE_RELATIVE     :1
    TSF_MARGINS_RELATIVE        :1
    TSF_LEADING_RELATIVE        :1
                                :12
TextStyleFlags      end

```

Library: **Objects/vTextC.def**



■ TextStylePrivateData

```
TextStylePrivateData    struct
    TSPD_flags          TextStyleFlags
    TSPD_unused         byte 2 dup (0)
TextStylePrivateData    ends
```

Library: **Objects/vTextC.def**

■ TextTransferBlockHeader

```
TextTransferBlockHeader struct
    TTBH_meta            VMChainTree
    TTBH_reservedOther   word 20 dup (0)

    TTBH_firstVM         label word
    TTBH_text            dword      ;huge array ID
    TTBH_charAttrRuns    dword      ;huge array ID
    TTBH_paraAttrRuns    dword      ;huge array ID
    TTBH_typeRuns        dword      ;huge array ID
    TTBH_graphicRuns     dword      ;huge array ID

    TTBH_firstLMem       label word
    TTBH_charAttrElements dword      ;VM block handle
    TTBH_paraAttrElements dword      ;VM block handle
    TTBH_typeElements    dword      ;VM block handle
    TTBH_graphicElements dword      ;VM block handle
    TTBH_styles           dword      ;VM block handle
    TTBH_names            dword      ;VM block handle
    TTBH_pageSetup        dword      ;VM block handle
    TTBH_lastLMem        label word

    TTBH_reservedVM      dword 10 dup (0)
TextTransferBlockHeader ends
```

Library: **Objects/vTextC.def**

■ TFStyleRun

```
TFStyleRun             struct
    TFSR_count          word      ? ; character count
    TFSR_attr            TextAttr <> ; text attributes
TFStyleRun             ends
```

Library: **gstring.def**

■ THCFeatures

```

THCFeatures      record
    THCF_FOLLOW_HYPERLINK      :1
    THCF_SET_KYPERLINK        :1
    THCF_SET_CONTEXT           :1
    THCF_DEFINE_FILE           :1
    THCF_DEFINE_CONTEXT        :1
THCFeatures      end

```

Library: **Objects/Text/tCtrlC.def**

■ THCToolboxFeatures

```

THCToolboxFeatures record
THCToolboxFeatures end

```

Library: **Objects/Text/tCtrlC.def**

■ ThreadException

```

ThreadException  etype word, 0, 4
    TE_DIVIDE_BY_ZERO      enum ThreadException
    TE_OVERFLOW            enum ThreadException
    TE_BOUND               enum ThreadException
    TE_FPU_EXCEPTION       enum ThreadException
    TE_SINGLE_STEP         enum ThreadException
    TE_BREAKPOINT          enum ThreadException

```

Library: **thread.def**

■ ThreadGetInfoType

```

ThreadGetInfoType etype word, 0, 2
    TGIT_PRIORITY_AND_USAGE enum ThreadGetInfoType
    TGIT_THREAD_HANDLE      enum ThreadGetInfoType
    TGIT_QUEUE_HANDLE       enum ThreadGetInfoType

```

Library: **thread.def**



■ **ThreadModifyFlags**

```
ThreadModifyFlags    record
    TMF_BASE_PRIO    :1
    TMF_ZERO_USAGE    :1
                    :6
ThreadModifyFlags    end
```

Library: **thread.def**

■ **ThreadPriority**

```
ThreadPriority        etype byte
    PRIORITY_TIME_CRITICAL    enum ThreadPriority, 0
    PRIORITY_HIGH             enum ThreadPriority, 64    ;IM
    PRIORITY_UI               enum ThreadPriority, 96    ;UI
    PRIORITY_FOCUS            enum ThreadPriority, 128   ;FOCUS
    PRIORITY_STANDARD         enum ThreadPriority, 160   ;STD
    PRIORITY_LOW              enum ThreadPriority, 192   ;BACKGROUND
    PRIORITY_LOWEST           enum ThreadPriority, 255   ;Used by kernel
```

Library: **thread.def**

■ **ThreePointArcParams**

```
ThreePointArcParams    struct
    TPAP_close          ArcCloseType    ; how the arc should be closed
    TPAP_point1         PointWWFixed    ; Point #1 (start of arc)
    TPAP_point2         PointWWFixed    ; Point #2 (a non-terminal point on the
                                        ; arc)
    TPAP_point3         PointWWFixed    ; Point #3 (end of arc)
ThreePointArcParams    ends
```

Library: **graphics.def**

■ **ThreePointArcToParams**

```
ThreePointArcToParams    struct
    TPATP_close         ArcCloseType    ; how the arc should be closed
    TPATP_point2        PointWWFixed    ; Point #2 (a non-terminal point on the
                                        ; arc)
    TPATP_point3        PointWWFixed    ; Point #3 (end of arc)
ThreePointArcToParams    ends
```

Library: **graphics.def**

■ ThreePointRelArcToParams

```
ThreePointRelArcToParams      struct
    TPRATP_close      ArcCloseType      ; how the arc should be closed
    TPRATP_delta2      PointWWFixed      ; delta to Point #2
    TPRATP_delta3      PointWWFixed      ; delta to Point #3
ThreePointRelArcToParams      ends
```

Library: **graphics.def**

■ TimerType

```
TimerType      etype word, 0, 2
    TIMER_ROUTINE_ONE_SHOT      enum TimerType
    TIMER_ROUTINE_CONTINUAL      enum TimerType
    TIMER_EVENT_ONE_SHOT      enum TimerType
    TIMER_EVENT_CONTINUAL      enum TimerType
    TIMER_MS_ROUTINE_ONE_SHOT      enum TimerType
    TIMER_EVENT_REAL_TIME      enum TimerType
```

Library: **timer.def**

■ TMSFlags

```
TMSFlags      record
    TMSF_IS_BREAK_CHARACTER      :1      ;TRUE: Last char was a break character.
    TMSF_IS_OPTIONAL_HYPHEN      :1      ;TRUE: Break is an optional hyphen.
    TMSF_PAD_SPACES      :1      ;TRUE: AddWidth should pad spaces.
    TMSF_UPDATE_SIZE_ONLY      :1      ;TRUE: AddWidth only updates size.
    TMSF_OPT_HYPHENS      :1      ;TRUE: deal with optional hyphens.
    TMSF_NEGATIVE_KERNING      :1      ;TRUE: last char on line was negatively
                                ;kerned.
    TMSF_EXTENDS_ABOVE      :1      ;TRUE: last char on line has tall accent.
    TMSF_EXTENDS_BELOW      :1      ;TRUE: last char on line has large
                                ;descender.
    TMSF_STYLE_CHANGED      :1      ;TRUE: style changed, update line height.
                                :7      ;Yes, I want an entire word...
TMSFlags      end
```

Library: **text.def**



■ TocCategoryStruct

```
TocCategoryStruct  struct
    TCS_tokenChars  TokenChars <>
    TCS_files       dbptr <>          ; file name array
    TCS_devices     dbptr <>
TocCategoryStruct  ends
```

This is the element structure for each element in the categories array.

TCS_devices stores the device name array (if and only if
TCF_EXTENDED_DEVICE_DRIVERS is set).

Library: **config.def**

■ TocDeviceStruct

```
TocDeviceStruct    struct
    TDS_driver      word           ; element in driver array
    TDS_info        word           ; extra word of info (depends on device type).
    TDS_name        label char
TocDeviceStruct    ends
```

Library: **config.def**

■ TocDiskStruct

```
TocDiskStruct      struct
    TDSS_volumeName VolumeName
    TDSS_mediaType  MediaType
    TDSS_name       label char
TocDiskStruct      ends
```

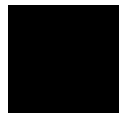
Library: **config.def**

■ TOC_ext

```

TOC_ext struct
;
; Entries that are passed
;
TOCE_areaToFill          sword
TOCE_hyphenCallback      dword
;PASS:      ss:bp        = pointer to TOC_vars structure on stack.
;           di           = Offset to the position where we would split the
;                       word
;           TOCI_lastWordStart =
;                       Offset in the text where the word to break starts
;           TOCI_lastWordPos =
;                       Position (distance from left edge of the field)
;                       where the word to break starts
;
;RETURN:      TOCI_suggestedHyphen =
;            The offset to the character to break the word at.;
Zero to break at the start of the word.
;           TOCI_suggestedHyphenPos =
;            The position (distance from left edge of the
;            field) where the hyphen starts.
;           TOCE_hyphenWidth =
;            Width of the hyphen that was placed at the end of
;            the line.
;
;DESTROYED: nothing
;
TOCE_tabCallback          dword
;PASS:      ds:si        = pointer to text
;           ss:bp        = TOC_vars
;           ss:bx        = LICL_vars
;
;RETURN:      carry set if there is no tabstop within the margins.
;           TOCE_areaToFill set correctly.
;
;DESTROYED: nothing
;
TOCE_heightCallback       dword
;PASSED:      ss:bp      = TOC_vars
;           ax.bl       = Line height for new characters (WBFixed)
;
;RETURN:      nothing
;
;DESTROYED: nothing
;
TOCE_passBack            word
TOCE_anchorChar          word
;

```



```

; Entries that are passed and returned
;
TOCE_flags                TOCFlags
TOCE_lineHeight           WBFixed
TOCE_lineBLO              WBFixed
TOCE_lineFlags            LineFlags
;
; Entries that are returned
;
TOCE_otherFlags           TOCOtherFlags
TOCE_nSpaces              sword
TOCE_nExtraSpaces         sword
TOCE_widthToAnchor        sword
TOCE_nChars               sword
TOCE_break                sword
TOCE_hyphenWidth          WBFixed
TOCE_fieldWidth           WBFixed
TOCE_justWidth            sword
TOC_ext  ends

```

This structure contains some fields which are passed to **GrTextObjCalc** by the application along with some fields which are returned.

TOCE_areaToFill stores the width of the area we are trying to fit the field to.

TOCE_hyphenCallback stores the address of the callback routine to perform automatic hyphenation. (The callback's parameters are listed in the structure display.)

TOCE_tabCallback stores the address of the callback routine to call when a TAB character is encountered. (The callback's parameters are listed in the structure display.)

TOCE_heightCallback stores the address of the callback routine to call when the line height changes. (The callback's parameters are listed in the structure display.)

TOCE_passBack stores a custom word of data to allow applications to pass data to their callbacks.

TOCE_anchorChar stores the anchor character to look for if the current field is associated with an anchored tab-stop.

TOCE_flags stores the **TOCFlags** that are both passed and returned.

TOCE_lineHeight should store the current value of the line height (at the time this stack frame is passed in). If a line would grow taller as a result of adding the new field, then this value is returned to reflect the new height.

TOCE_lineBLO stores the current value of the lines baseline-offset (at the time this stack frame is passed in). A line height is determined by its ascent and descent. To compute these values we need the baseline.

TOCE_lineFlags stores the **LineFlags** for the current line based on the previous calculations and the current calculations.

TOCE_otherFlags stores some optimization flags which decide whether an optimized redraw is possible after a text change.

TOCE_nSpaces stores the number of spaces in the line which can be padded for full justification.

TOCE_nExtraSpaces stores the number of spaces after the word-break. These spaces are on the line but shouldn't be considered for full justification.

TOCE_widthToAnchor stores the amount of the field which falls before the anchor character.

TOCE_nChars stores the number of characters in the field which fit in the area.

TOCE_break stores the position of the break in the text.

TOCE_hyphenWidth stores the width of the generated hyphen at the end of the line.

TOCE_fieldWidth stores the width of the field up to the word-break.

TOCE_justWidth stores the amount of the field which should be considered for justification. This value doesn't include the spaces at the end of the line.

Library: **text.def**

■ TocFileStruct

```
TocFileStruct      struct
    TFS_sourceDisk  word           ; Disk token
    TFS_release     ReleaseNumber <>
    TFS_name        label char
TocFileStruct      ends
```

Library: **config.def**



■ TOCFlags

```

TOCFlags          record
    TOCF_NO_WORD_WRAP      :1      ; PASS: Set - word-wrap should be done
    TOCF_AUTO_HYPHENATE    :1      ; PASS: Set - attempt auto hyphenation

    TOCF_FOUND_ANCHOR      :1      ; RET: Set - an anchor character was found
    TOCF_IS_HARD_HYPHEN    :1      ; RET: Set - break is a hard or opt hyphen
    TOCF_FOUND_BREAK       :1      ; RET: Set - an auto-hyphen position was
                                ; found
    TOCF_LINE_TERMINATED   :1      ; RET: Set - last field on line
    TOCF_ONE_TAB_TOO_LARGE :1      ; RET: Set - tab couldn't be handled
    TOCF_OPT_HYPHEN_TOO_WIDE :1    ; RET: Set - optional hyphen too wide to
                                ; fit

TOCFlags          end

```

Library: **text.def**

■ TOC_int

```

TOC_int  struct
    TOCI_style          TextMetricStyles
    TOCI_currentHgt      WBFixed    ; Height of the field
    TOCI_currentBlo      WBFixed    ; Baseline of the field
    TOCI_lastWordStart   word       ; Position in text of last word start.
    TOCI_lastWordPos     WBFixed    ; Position of last word start.
    TOCI_lastWordEndPos  word       ; Position of last word end.
    TOCI_lastHyphen      word       ; Position of last usable hyphen.
    TOCI_lastHyphenPos   WBFixed    ; Position of last soft/hard hyphen.
    TOCI_tallCharHeightPos word     ; Position of tall character with an
                                ; important height value.
    TOCI_tallCharHeight  WBFixed    ; Height of tall char.
    TOCI_tallCharBaselinePos word    ; Position of tall character with an
                                ; important baseline value.
    TOCI_tallCharBaseline WBFixed    ; Baseline of tall char.
    TOCI_suggestedHyphen word       ; Position of suggested hyphen in text
    TOCI_suggestedHyphenPos WBFixed  ; Position of suggested hyphen on line
    align               word

TOC_int  ends

```

This structure stores fields which are internal to **GrTextObjCalc**. All fields are initialized and used inside **GrTextObjCalc**.

Library: **text.def**

■ TocMap

```
TocMap    struct
    TM_disks      dbptr
    TM_categoriesdbptr
TocMap    ends
```

This structure is the map item of the TOC file.

Library: **config.def**

■ TOCOtherFlags

```
TOCOtherFlags    record
    TOCOF_IS_FIRST_FIELD      :1 ; PASS: Set - This is the first field on the
                                ; line
    TOCOF_PREV_CHAR_KERNED    :1 ; HACK added by jim 4/27/92 so kernel will
                                ; make
    TOCOF_LAST_BREAK_KERNED   :1 ; HACK added by jim 4/27/92 so kernel will
                                ; make
                                :5
TOCOtherFlags    end
```

Library: **text.def**



■ TocUpdateCategoryFlags

```
TocUpdateCategoryFlags record
    TUCF_EXTENDED_DEVICE_DRIVERS:1
        ; Files being enumerated are assumed to be extended device drivers.

    TUCF_CUSTOM_FILES:1
        ; The TUCP_fileArrayElementSize field will be
        ; used when creating the files array. Otherwise, each element
        ; of the files array will be of size TocFileStruct. NOTE: If
        ; this flag is used, the data structure used for each file
        ; element MUST contain TocFileStruct as its first element.

    TUCF_ADD_CALLBACK:1
        ; TUCP_addCallback contains a fptr to a callback
        ; routine that will be called when a file is added to the
        ; files array.

    TUCF_DIRECTORY_NOT_FOUND:1
        ; Don't actually scan the directory, because it doesn't exist.
        ; Just create the category, and leave it empty.

:12
TocUpdateCategoryFlags end
```

Library: **config.def**

■ TocUpdateCategoryParams

```
TocUpdateCategoryParams struct
    TUCP_flags                TocUpdateCategoryFlags
    TUCP_tokenChars            TokenChars
    TUCP_fileArrayElementSize byte
    TUCP_addCallback           fptr.far
; CALLBACK:
;     PASS:    ds:si - filename to add
;              di - VM handle of SortedNameArray
;              (pass to TocSortedNameArrayAdd)
;
;     RETURN:  carry CLEAR if new element added,
;              ax - element number
;
;              carry SET if add aborted
;
;     CAN DESTROY: bx,cx,dx
;
    align    word
TocUpdateCategoryParams ends
```

Library:

■ TOC_vars

```
TOC_vars                struct
    TOCV_int             TOC_int
    TOCV_ext             TOC_ext
    align                word
TOC_vars                ends
```

This structure is passed to **GrTextObjCalc** and consists of external parameters (*TOC_ext*) and internal variables (*TOC_int*).

Library: **text.def**

■ ToggleState

```
ToggleState            record
    TS_CAPSLOCK         :1
    TS_NUMLOCK          :1
    TS_SCROLLLOCK       :1
ToggleState            end
```



Library:

■ TokenDBItem

```
TokenDBItem      struct
    TDBI_group    word
    TDBI_item     word
TokenDBItem      ends
```

This structure defines the identifier for a token database item.

Library: **token.def**

■ TokenEntry

```
TokenEntry      struct
    TE_type      TokenIndexType
    TE_token      GeodeToken <>
    TE_monikerList TokenDBItem <>
    TE_flags      TokenFlags
    TE_release    ReleaseNumber <>
    TE_protocol   ProtocolNumber <>
TokenEntry      ends
```

This structure stores a token entry, which is used in the index (map item) of the token database.

TE_type specifies the type of index entry that this token entry corresponds to.

TE_token specifies the **GeodeToken** for this file.

TE_monikerList stores the list of monikers for this token. This entry points to a chunk containing the item numbers of the chunks of the token.

TE_flags stores the **TokenFlags** of the token, which contain the token's relocation status.

TE_release stores the **ReleaseNumber** of the token database.

TE_protocol stores the **ProtocolNumber** of the token database.

Library: **token.def**

■ TokenError

TokenError	etyp	word, 1
BAD_PROTOCOL_IN_SHARED_TOKEN_DATABASE_FILE	enum	TokenError
ERROR_OPENING_SHARED_TOKEN_DATABASE_FILE	enum	TokenError
ERROR_OPENING_LOCAL_TOKEN_DATABASE_FILE	enum	TokenError

Library: **token.def**

■ TokenFlags

```
TokenFlags      record
    TF_NEED_RELOCATION    :1
    TF_UNUSED           :15
TokenFlags      end
```

Library: **token.def**

■ TokenMonikerInfo

```
TokenMonikerInfo      struct
    TMI_moniker      TokenDBItem <>
    TMI_fileFlag      word           ; 0 if token is in shared
                                      ; token DB file
                                      ; non-0 if it's in local file
TokenMonikerInfo      ends
```

The **TokenMonikerInfo** structure is used by applications which call **TokenLookupMoniker**, store the information returned, and later use it to call **TokenLockTokenMoniker**.

Library: **token.def**

■ TokenRangeFlags

```
TokenRangeFlags record
    TRF_ONLY_GSTRING      :1
    TRF_ONLY_PASSED_MANUFID :1
                                      :14
TokenRangeFlags end
```

Library: **token.def**

■ ToolboxInfo

```
ToolboxInfo      struct
    TI_object      optr
    TI_name         optr
ToolboxInfo      ends
```

TI_object stores the optr of the GenInteraction under which tools may be placed (This optr is *unrelocated*! Use the UN_OPTR macro in assembly).

TI_name stores the null-terminated string name for the above tool location (This optr is also unrelocated. Use UN_OPTR macro in assembly).

Library: **Objects/gToolCC.def**



■ ToolGroupHighlightType

```
ToolGroupHighlightType    etype byte
    TGHT_INACTIVE_HIGHLIGHT    enum ToolGroupHighlightType
    TGHT_ACTIVE_HIGHLIGHT      enum ToolGroupHighlightType
    TGHT_NO_HIGHLIGHT          enum ToolGroupHighlightType
```

Library: **Objects/gToolGC.def**

■ ToolGroupInfo

```
ToolGroupInfo            struct
    TGI_object            optr
ToolGroupInfo            ends
```

TGI_object stores the GenToolGroup that this tool control will operate on.
(This optr is *unrelocated*! Use the UN_OPTR macro in assembly).

Library: **Objects/gToolCC.def**

■ TrackScrollingParams

```
TrackScrollingParams      struct
    TSP_action            ScrollAction
    TSP_flags              ScrollFlags          ;scroll flags
    TSP_caller            optr                  ;object to return args to
    ;
    ; Only one set of these are sent via a MSG_META_CONTENT_TRACK_SCROLLING. The
    ; relative values (xChange, yChange) are sent on the relative scrolls --
    ; SA_SCROLL, SA_INC_FWD, SA_INC_BACK, SA_PAGE_FWD, SA_PAGE_BACK, SA_PAN. The
    ; absolute values (newOriginX, newOriginY) are sent on the absolute scrolls.
    ; To play it safe, your handler should call GenSetupTrackingArgs, which will
    ; fill in all of these.
    ;
    TSP_change            PointDWord           ;proposed change
    TSP_newOrigin          PointDWord           ;proposed new origin
    ;
    ; These arguments are NOT sent via MSG_META_CONTENT_TRACK_SCROLLING. If you
    ; want to have these (and you probably will), your handler should call
    ; GenSetupTrackingArgs, which will fill in all of these.
    ;
    TSP_oldOrigin          PointDWord           ;old origin
    TSP_viewWidth          sword               ;view width
    TSP_viewHeight         sword               ;view height
TrackScrollingParams      ends
```

TSP_action stores the action taking place. Drags don't require the return message; in fact, return methods will be ignored for drags.

Library: **Objects/gViewC.def**

Reference book

■ TransferFileHeader

```
TransferFileHeader    struct
    TFH_normalItem    word        ; VM block handle of normal transfer item
TransferFileHeader    ends
```

This structure defines the map block of the transfer VM file, this is saved out in the UI's transfer VM file when the system is shutdown. The VM block handles must be valid handles for this VM transfer file.

Library: **Objects/clipbrd.def**

■ TransFlags

```
TransFlags            record
    TF_INV_VALID      :1
    TF_ROTATED        :1
    TF_SCALED         :1
    TF_TRANSLATED     :1
TransFlags            end
```

Library: **tmatrix.def**

■ TransMatrix

```
TransMatrix           struct
    TM_e11            WWFixed <0,1>
    TM_e12            WWFixed <0,0>
    TM_e21            WWFixed <0,0>
    TM_e22            WWFixed <0,1>
    TM_e31            DWFixed <0,0>
    TM_e32            DWFixed <0,0>
TransMatrix           ends
```

This structure stores the transformation matrix used within the GEOS graphics system. This matrix has six variable elements. (The last column of the 3x3 transformation matrix is the identity column [0 0 1].)

This **TransMatrix** is initially set to the identity matrix.

TM_e11 stores the value (32 bit **WWFixed**) at row 1, column 1.

TM_e12 stores the value (32 bit **WWFixed**) at row 1, column 2.

TM_e21 stores the value (32 bit **WWFixed**) at row 2, column 1.

TM_e22 stores the value (32 bit **WWFixed**) at row 2, column 2.

TM_e31 stores the value (48 bit **DWFixed**) at row 3, column 1.



TM_e32 stores the value (48 bit **DWFixed**) at row 3, column 2.

Library: **graphics.def**

■ TravelingObjectReference

```
TravelingObjectReference struct
    TIR_travelingObject    optr
    TIR_parent             lptr
    TIR_compChildFlags     CompChildFlags
TravelingObjectReference ends
```

TIR_travelingObject stores the optr of the object which should be kept moving to the top GenDisplay. This optr should be stored in unrelocated for. (e.g. in assembly):

UN_OPTR TUIToolbox3

TIR_parent stores the optr of the object within the GenDisplay under which the traveling object should be added.

TIR_compChildFlags stores the **CompChildFlags** to use when adding the traveling object below the parent.

Library: **Objects/gDispC.def**

■ TravelOption

```
TravelOption          etype word, 0
    TO_NULL           enum TravelOption
    TO_SELF           enum TravelOption
    TO_OBJ_BLOCK_OUTPUT enum TravelOption
    TO_PROCESS        enum TravelOption
```

TO_NULL No object to deliver message to, the event should be destroyed.

TO_SELF No additional UI behavior requested -- allow MetaClass handler to dispatch event if possible, else destroy it.

TO_OBJ_BLOCK_OUTPUT
Sends event to Object Block output, if any, otherwise destroys the event.

TO_PROCESS
Sends event to the process owning the UI block.

Library: **Objects/metaC.def**

■ **TRCCFeatures**

```
TRCCFeatures      record
  TRCCF_ROUND      :1
  TRCCF_IGNORE_ORIGIN :1
TRCCFeatures      end
```

Library: **Objects/Text/tCtrlC.def**

■ **TRCCToolboxFeatures**

```
TRCCToolboxFeatures  record
TRCCToolboxFeatures  end
```

Library: **Objects/Text/tCtrlC.def**

■ **TSCFeatures**

```
TSCFeatures      record
  TSCF_PLAIN      :1
  TSCF_BOLD       :1
  TSCF_ITALIC     :1
  TSCF_UNDERLINE  :1
  TSCF_STRIKE_THRU :1
  TSCF_SUBSCRIPT  :1
  TSCF_SUPERSCRIPT :1
  TSCF_BOXED      :1
  TSCF_BUTTON     :1
  TSCF_INDEX      :1
  TSCF_ALL_CAP    :1
  TSCF_SMALL_CAP  :1
TSCFeatures      end
```

Library: **Objects/Text/tCtrlC.def**



■ **TSCToolboxFeatures**

```

TSCToolboxFeatures  record
    TSCTF_PLAIN           :1
    TSCTF_BOLD            :1
    TSCTF_ITALIC          :1
    TSCTF_UNDERLINE       :1
    TSCTF_STRIKE_THRU     :1
    TSCTF_SUBSCRIPT        :1
    TSCTF_SUPERSCRIPT      :1
    TSCTF_BOXED           :1
    TSCTF_BUTTON          :1
    TSCTF_INDEX           :1
    TSCTF_ALL_CAP         :1
    TSCTF_SMALL_CAP       :1
TSCToolboxFeatures  end

```

Library: **Objects/Text/tCtrlC.def**

■ **TVTNCPIData**

```

TVTNCPIData  struct
    TVTNCPID_handle  word
    TVTNCPID_id      word
TVTNCPIData  ends

```

Library: **vTextC.def**

■ **UChar**

```

UChar  etype byte
    UC_NULL           enum UChar, 0x0  ;NULL
    UC_QUICK_COPY     enum UChar, 0x1  ;unnecessary -- should remove!
    UC_BUTTON_EVENT   enum UChar, 0x2  ;send on a button event

```

Library: **uiInputC.def**

■ **UIButtonFlags**

```

UIButtonFlags  record
    UIBF_NO_KEYBOARD           :1
    UIBF_CLICK_TO_TYPE        :1
    UIBF_SELECT_ALWAYS_RAISES :1
    UIBF_SELECT_DISPLAYS_MENU :1

```

```

UIBF_KEYBOARD_ONLY      :1
UIBF_CLICK_GOES_THRU    :1
UIBF_SPECIFIC_UI_COMPATIBLE :1
UIBF_BLINKING_CURSOR    :1
UIButtonFlags           end

```

UIBF_NO_KEYBOARD

Set if working in no-keyboard mode (i.e. pen system). Can be used by UI and applications to provide extensions to easy simplify usage. This is exclusive of UIBF_KEYBOARD_ONLY below.

UIBF_CLICK_TO_TYPE

Determines which FOCUS model to use:

If true: “explicit focus” - must press mouse button over window to give window keyboard focus.

If false: “pointer focus” or “real estate model” - window underneath mouse pointer is automatically given keyboard focus, after a delay in some UIs.

UIBF_SELECT_ALWAYS_RAISES

Set if the SELECT function always raises the window underneath the pointer to the front, whether in the visible region of the window, or inside a view that is inside the window. If false, the SELECT function within a view will not bring that window to the front.

UIBF_SELECT_DISPLAYS_MENU

Set if SELECT and FEATURES buttons are swapped so that SELECT opens a menu, while FEATURES executes the default menu item.

UIBF_KEYBOARD_ONLY

Set if working in keyboard only mode. Can be used by UI and applications to provide extensions to easy simplify keyboard usage. This is exclusive of UIBF_NO_KEYBOARD above.

UIBF_CLICK_GOES_THRU

Applies only in “explicit focus” model - otherwise known as “click to type.” Set if mouse press event which brings window to front should also be sent onto gadget.

UIBF_SPECIFIC_UI_COMPATIBLE

Set if specific UI should run in compatibility mode.

UIBF_BLINKING_CURSOR

Set if the text cursor should blink.



Library: **Objects/uiInputC.def**

■ UIExpressOptions

```
UIExpressOptions    record
                    :4
    UIEO_RETURN_TO_DEFAULT_LAUNCHER:1; Set to have a "Return to <default
                                ; launcher>" button in the Express Menu
    UIEO_GEOS_TASKS_LIST      :1      ; Set for list of currently running GEOS
                                ; applications
    UIEO_DESK_ACCESSORY_LIST  :1      ; Set for list of desk accessories
                                ; (applications in World/Desk Accessories
                                ; directory)
    UIEO_MAIN_APPS_LIST      :1      ; Set for list of applications in World
                                ; directory
    UIEO_OTHER_APPS_LIST     :1      ; Set for hierarchial list of applications
                                ; in subdirectories below World directory.
    UIEO_CONTROL_PANEL       :1      ; Set for control panel area.
    UIEO_DOS_TASKS_LIST      :1      ; Set for list of available DOS tasks.
    UIEO_UTILITIES_PANEL     :1      ; Set for utilities panel area.
    UIEO_EXIT_TO_DOS         :1      ; Set for Exit to DOS trigger.
    UIEO_POSITION            UIExpressPositions:3
                                ; Position of Express menu.

UIExpressOptions    end
```

Library: **ui.def**

■ UIExpressPositions

```
UIExpressPositions  etype word
    UIEP_NONE        enum UIExpressPositions
    UIEP_TOP_PRIMARY  enum UIExpressPositions
    UIEP_LOWER_LEFT   enum UIExpressPositions
```

Library: **ui.def**

■ UIFunctionsActive

```

UIFunctionsActive  record
    UIFA_SELECT      :1          ; Basic mouse function
    UIFA_MOVE_COPY   :1          ; Direct action (move/copy, "quick transfer" if
                                ; between applications)
    UIFA_FEATURES     :1          ; Popup menu, special UI capabilities.
    UIFA_CONSTRAIN    :1          ; Set if modifier(s) designated as "constrain"
                                ; are pressed. This flag will change with the
                                ; state of the modifier. Note that it may
                                ; generally NOT be used when the target object
                                ; can infer a meaning to "Extend" or "Toggle"
                                ; selection. (i.e. should only be used w/SELECT
                                ; function on things like object control points).

    UIFA_PREF_A       :1
    UIFA_PREF_B       :1
    UIFA_PREF_C       :1
    ; User "preferences" Meaning varies with active function. NOTE:
    ; 1) requests followed by (D) are updated every event holding this
    ; info (Dynamic)
    ;
    ;
    ; SELECT:           A          B          C
    ;                   Toggle     Extend
    ;
    ; MOVE_COPY:       Move(D)    Copy(D)
    ;
    ; FEATURES:        Popup      Pan
    ;                   menu       View
    ;
    ;
    UIFA_IN            :1          ; Set if point (cx, dx) is inside the visual
                                ; bounds of the object
UIFunctionsActive  end

```

Library: **Objects/uiInput.def**

■ UIHelpOptions

```

UIHelpOptions      record
                                :15
    UIHO_HIDE_HELP_BUTTONS :1
UIHelpOptions      end

```

UIHO_HIDE_HELP_BUTTONS

Set to not add help buttons to various dialog boxes. Usually used on small screen devices where screen space is at a premium, or on a device that has a dedicated help button already.

Default interpretation: false (i.e., help buttons appear).

Library: **ui.def**



■ UIInterfaceLevel

UIInterfaceLevel	etype	word
UIIL_INTRODUCTORY	enum	UIInterfaceLevel
UIIL_BEGINNING	enum	UIInterfaceLevel
UIIL_INTERMEDIATE	enum	UIInterfaceLevel
UIIL_ADVANCED	enum	UIInterfaceLevel
UIIL_GURU	enum	UIInterfaceLevel

UIIL_INTRODUCTORY

This level is designed for the first-time user, or those who just use computers infrequently. Complex models & all but the most basic features are shunned in favor of metaphors & functionality that is easy to grasp. Ease of learning and the absence of anything that isn't immediately obvious are the most important considerations of interfaces presented at this level, with a focus towards providing a pleasant experience. Ease of quick results is very important, however prefabricated & limited in scope (remember PrintShop?).

Default behavior:

In this mode, the UI protects the user from the concepts of "running" applications and "open" documents by letting them just switch to whatever application and document they wish to use. The UI takes care of managing the running status of applications and open status of documents transparently, in the background. Applications generally come up maximized, and have no window controls whatsoever (the exception here is desk accessories, which float on top and may be moved and dismissed by the user). Only one document at a time may be worked on, and it generally appears in a display which is permanently maximized. Applications where the user typically uses only one data file (Address book) won't have a "File" menu at all. Applications designed for creation of new files will have only "New..." & "Switch to..." options, & will automatically fetch and place files in a single directory. Keyboard shortcuts and mnemonic navigation are turned off (excepting keyboard only systems) Application menu structures in general are kept to a minimum, and advanced features are kept completely hidden, not even accessible through the "Options" menu.

UIIL_BEGINNING

This level is designed for those who feel comfortable with the basic operation of their computer, how it works, what modules exist within it, etc. and wish to gain access to more of its capability, or need to solve a particular problem or need for

something other than a canned solution. This level adds in a number; of useful features to UI-provided menus, and results in applications offering “options” to turn on all but the most advanced or short-cut oriented capabilities within them. Ease of Learning remains the most important aspect of the interface, and probability of successful usage the most important goal. We’re trying to get the user to be able to actually *accomplish* things here, all on their own, with a low risk of failure. Time to task completion is not an issue so long as the user is able to figure out what the model is, how to use it, and is able to actually complete whatever it is they’re trying to do. This may be accomplished via more verbose, or scripted dialogs, as opposed to the “set everything at once” type of dialogs seen at higher levels. The computer should detect abnormal or dangerous situations & help the user to avoid costly mistakes. Dangerous actions should be undoable. Options that trade performance against safety or recoverability will be tilted towards recoverability.

Default behavior:

In this mode, the document control adds “New..”, “Switch to...”, “Quick Backup”, and variety of other capabilities. Applications offer a way to access most of their features. The application launch & document models remain “transparent”. applications continue to run full-screen. Files created by the user remain in one directory.

UIIL_INTERMEDIATE

This level is designed for people familiar with the capabilities of the software, & who are now willing to learn a few things that might not otherwise be obvious in order to speed up their ability to get things done.

Default behavior:

Here we introduce the user to the concepts of “running applications” and “open documents”, and add in easy to understand “power” features that makes it easier to get things done. The user must open and close applications and documents to manage accessibility and performance. Application windows start out overlapping (except on machines w/small screens). Adds window min/max/restore capability, pinned menus. Systems with both mice and keyboards get keyboard accelerators and mnemonic navigation. The system allows only one instance of any given application to be running,



but allows multiple documents to be open within that application.

UIIL_ADVANCED

This level is designed for the people who use their computer day in, day out, and know GEOS like the back of their hand. All the bells and whistles available are offered here, though still organized intelligently with the degree of accessibility set by the user -- a technical writer may live in his Word Processor, for instance, but venture into other applications only infrequently—they shouldn't all look like the cockpits of 727's. A reduction in the number of steps necessary to complete common tasks, & the speed in which this can be done becomes very important. The key phrases here are "powerful", "well designed", and "intelligent".

Default behavior:

The UI offers the possibility of multiple instances of a given application, dialog-clarified. The notion of hierarchical storage of document files is introduced.

UIIL_GURU Same as "UIIL_ADVANCED" level, but minus protective warning dialogs that might be annoying to someone who never makes mistakes. Options that trade performance against safety or recoverability are tilted towards performance.

Library: **ui.def**

■ UIInterfaceOptions

```
UIInterfaceOptions record
    UIIO_OPTIONS_MENU           :1
    UIIO_DISABLE_POPOUTS       :1
    UIIO_ALLOW_INITIALLY_HIDDEN_MENU_BARS :1
                                :13
UIInterfaceOptions end
```

UIIO_OPTIONS_MENU

Set if the options menu should exist.

UIIO_DISABLE_POPOUTS

True to not allow GIV_POPOUT GenInteractions to pop in and out. False to allow pop in and pop out behavior.

Library: **ui.def**

■ UILaunchModel

UILaunchModel	etype	word
UILM_TRANSPARENT	enum	UILaunchModel
UILM_SINGLE_INSTANCE	enum	UILaunchModel
UILM_MULTIPLE_INSTANCES	enum	UILaunchModel
UILM_GURU	enum	UILaunchModel

UILM_TRANSPARENT

“Transparent” application launch mode is one in which the user doesn’t have to understand the concepts of a “running app”, as the system takes care of launching and shutting down applications in the background to manage memory effectively.

-> Express menu is really just a “startup” or “switch to” menu

-> applications are shut down in background, and reloaded when switched to

-> Single instance limit on any given application.

-> Application windows are full screen (except for those marked as Desk Accessories, which float on top, and would not be managed transparently, i.e. would have to stay in memory until exited).

-> Minimize/Maximize/Restore/Close features of full-screen main Primary windows removed-

> “Exit” item is eliminated from File menu.

-> Default mode for UIIL_INTRODUCTORY.

The following levels are all user-controlled, meaning that the user has to understand the concept of a running application, & must manage how many applications are running themselves.

-> Express menu allows switching between currently running applications.

-> By default, application windows are not maximized on launch, and are movable and resizable. This could be “fine tuned” by using window options above, however.

UILM_SINGLE_INSTANCE

This mode allows only a single instance of any one given application to be running at a time.

-> Single instance limit on given application.



-> If application or document is double-clicked on, and an instance of the application is already running, that instance would be brought to the top (and any document opened/switched to within it, depending on the doc model).

-> Default mode for UIIL_INTERMEDIATE.

UILM_MULTIPLE_INSTANCES

-> If application or document double-clicked on, and an instance of the application is already running, a dialog would come up asking if one of the already running applications should be used, or whether a new instance should be created.

-> Default mode for UIIL_ADVANCED.

UILM_GURU

-> Like GEOS V1.2 -- the system does nothing to protect the user, so double-clicking on an application just launches another instance.

-> Default mode for UIIL_GURU.

Library: **ui.def**

■ UILaunchOptions

```
UILaunchOptions record
  UILO_DESK_ACCESSORIES      :1 ;TRUE if the desk accessory mode is
                                ; supported (default = TRUE)
  UILO_CLOSABLE_APPS         :1 ;Set if all apps should be closable.
                                ; This allows the user to close apps
                                ; even when in transparent mode.
                                :14
UILaunchOptions end
```

Library: **ui.def**

■ UIWindowOptions

```
UIWindowOptions    record
    UIWO_MAXIMIZE_ON_STARTUP                :1
    UIWO_COMBINE_HEADER_AND_MENU_IN_MAXIMIZED_WINDOWS :1
    UIWO_PRIMARY_MIN_MAX_RESTORE_CONTROLS    :1
    UIWO_WINDOW_MENU                        :1
    UIWO_PINNABLE_MENUS                    :1
    UIWO_KBD_NAVIGATION                     :1
    UIWO_POPOUT_MENU_BAR                   :1
UIWindowOptions    end
```

UIWO_MAXIMIZE_ON_STARTUP

If set, applications by default would come up maximized. (applications marked as desk accessories would override this behavior).

Default interpretation under Motif: True if running on a small screen (less than 512 pixels in *x*, or 320 pixels in *y*), or if on keyboard-only machine, or if InterfaceLevel < UIIL_INTERMEDIATE, or if LaunchMode = UILM_TRANSPARENT.

UIWO_COMBINE_HEADER_AND_MENU_IN_MAXIMIZED_WINDOWS

This is a screen space saving measure—if set, the header and menu areas of maximized windows is combined, such that only the window gadgetry, window menu and menus are left, i.e. the title string is eliminated. Default interpretation under Motif: True if running on a small screen (less than 512 pixels in *x*, or 320 pixels in *y*).

UIWO_PRIMARY_MIN_MAX_RESTORE_CONTROLS

If false, window gadgetry and menu items for minimizing, maximizing, and restoring would disappear from primary windows. Default interpretation under Motif: Always false if LaunchMode = UILM_TRANSPARENT, else true if InterfaceLevel >= UIIL_INTERMEDIATE.

UIWO_WINDOW_MENU

If true, a window menu for keyboard control of min/max/restore/move/resize features will be provided. If false, only a “close” icon will appear in this space. Default interpretation: True if keyboardOnly = true.

UIWO_PINNABLE_MENUS

True to allow “pinnable” menus. Default interpretation under Motif: True if InterfaceLevel >= UIIL_INTERMEDIATE.



UIWO_KBD_NAVIGATION

True to allow keyboard accelerators, keyboard navigation
 Default interpretation under Motif: True if keyboard-only
 machine or InterfaceLevel >= UIIL_INTERMEDIATE.

UIWO_POPOUT_MENU_BAR

True to allow menu bar to pop-out into a dialog box. This should
 only be allowed in very specific situations, because the specific
 UI will not always provide gadgetry to restore the menu bar if
 the dialog is closed. Default interpretation under Motif: True if
 running on a small screen (less than 512 pixels in *x*, or 320
 pixels in *y*).

Library: **ui.def**

■ UIWindowOptionsInteger

```
UIWindowOptionsInteger    record
  UIWOI_MASK              UIWindowOptions:8
  UIWOI_OPTIONS           UIWindowOptions:8
UIWindowOptionsInteger    end
```

UIWOI_MASK

Mask of which **UIWindowOptions** in *UIWOI_value* have
 meaning. (If zero, user has made no preference for that specific
 option, and the default behavior should be used).

UIWOI_OPTIONS

Actual **UIWindowOptions** to use (if mask bit above is set for
 any given bit).

Library: **ui.def**

■ UndoActionDataFlags

```
UndoActionDataFlags      struct
  UADF_flags              dword
  UADF_extraFlags         word
UndoActionDataFlags      ends
```

Library: **Objects/gProcC.def**

■ UndoActionDataOptr

```
UndoActionDataOptr       struct
  UADO_optr              optr
UndoActionDataOptr       ends
```

Library: **Objects/gProcC.def**

■ UndoActionDataPtr

```
UndoActionDataPtr  struct
    UADP_ptr        fptr
    UADP_size        word
UndoActionDataPtr  ends
```

Library: **Objects/gProcC.def**

■ UndoActionDataType

```
UndoActionDataType  etype word, 0, 2
    UADT_FLAGS        enum UndoActionDataType
    UADT_PTR           enum UndoActionDataType
    UADT_VM_CHAIN      enum UndoActionDataType
    UADT_OPTR          enum UndoActionDataType
```

UADT_FLAGS

The passed data is of type **UndoActionFlags**.

UADT_PTR

The passed data is of type **UndoActionDataPtr**.

UADT_VM_CHAIN

The passed data is of type **UndoActionVMChain**.

UADT_OPTR

This is not a valid type to pass to MSG_GEN_PROCESS_ADD_ACTION; it is used by the undo code when playing back an action of type UADT_PTR.

If MSG_GEN_PROCESS_UNDO_ADD_ACTION is called with an action of type UADT_PTR, if the action is played back, the data will be returned to the object with type UADT_OPTR (via MSG_META_UNDO). Calling MSG_GEN_PROCESS_ADD_ACTION can cause previously-sent actions to move, so the optr should be re-dereferenced after sending this message.

Library: **Objects/gProcC.def**



■ UndoActionDataUnion

```
UndoActionDataUnion      union
    UADU_flags            UndoActionDataFlags
    UADU_ptr              UndoActionDataPtr
    UADU_vmChain          UndoActionDataVMChain
    UADU_optr             UndoActionDataOptr
UndoActionDataUnion      ends
```

Library: **Object/gProcC.def**

■ UndoActionDataVMChain

```
UndoActionDataVMChain    struct
    UADVMC_vmChain        dword
    UADVMC_file           hptr
UndoActionDataVMChain    ends
```

This structure is filled in by the undo code for MSG_META_UNDO. VM Chains passed to MSG_GEN_PROCESS_UNDO_ADD_ACTION should lie in the undo file (which can be obtained by sending MSG_GEN_PROCESS_UNDO_GET_FILE).

Library: **Objects/gProcC.def**

■ UndoActionStruct

```
UndoActionStruct         struct
    UAS_dataType          UndoActionDataType
    UAS_data              UndoActionDataUnion
    UAS_appType           dword
UndoActionStruct         ends
```

UAS_dataType stores the type of data passed in **UndoActionDataUnion**.

UAS_data stores the data to be stored with the action.

UAS_appType stores two extra words of data to be sent with MSG_META_CLIPBOARD_UNDO that indicate the type of action we are undoing.

Library: **Objects/gProcC.def**

■ UndoDescription

```
UndoDescription          etype byte
    UD_UNDO               enum UndoDescription
    UD_REDO               enum UndoDescription
    UD_NOT_UNDOABLE       enum UndoDescription
```

UD_UNDO Passed in **NotifyUndoStateChange** if there is an active undo chain.

UD_REDO Passed in **NotifyUndoStateChange** if there is an active *redo* chain.

UD_NOT_UNDOABLE
 Passed in **NotifyUndoStateChange** if the last action was not undoable. Must pass 0:0 as title.

Library: **Objects/gEditCC.def**

■ UpdateUIDataBlk

```
UpdateUIDataBlk      struct
  UIDB_formatDataVMFileHan   word
  UIDB_formatDataVMBlkHan    word
  UIDB_curFormatToken        FormatIdType        ;Current format token
UpdateUIDataBlk      ends
```

Library: **math.def**

■ UpdateWindowFlags

```
UpdateWindowFlags    record
  UWF_ATTACHING        :1
  UWF_DETACHING        :1
  UWF_RESTORING_FROM_STATE :1
  UWF_FROM_WINDOWS_LIST :1
UpdateWindowFlags    end
```

UWF_ATTACHING
 Set if MSG_META_UPDATE_WINDOW is being sent because application is attaching.

UWF_DETACHING
 Set if MSG_META_UPDATE_WINDOW is being sent because application is detaching.

UWF_RESTORING_FROM_STATE
 Set if application is restoring from state (will only be set if UWF_ATTACHING is also set, i.e. application is attaching).

UWF_FROM_WINDOWS_LIST
 Set if MSG_META_UPDATE_WINDOW is sent to this object because this object was on the GenApplication's GAGCNLT_WINDOWS GCN list, and not from a subsequent



“build-on-demand” request. (This will only be set if UWF_ATTACHING is also set, i.e. application is attaching)

Library: **Objects/metaC.def**

■ UserDoDialogStruct

```
UserDoDialogStruct      struct
    UDDS_callingThread   hptr
    UDDS_semaphore       hptr
    UDDS_response        word
    UDDS_complete        word
    UDDS_boxRunByCurrentThreadword
    UDDS_dialog          optr
    UDDS_queue           hptr
UserDoDialogStruct      ends
```

This structure is passed to MSG_GEN_INTERACTION_INITIATE_BLOCKING_THREAD_ON_RESPONSE.

UDDS_callingThread stores the handle of the thread that is waiting for this dialog to come down. If 0, the thread has no event queue, and is instead blocking on *UDDS_semaphore*.

UDDS_semaphore stores the handle of a the semaphore that a non-event-driven thread is blocking on. See note above.

UDDS_response stores the response value returned by the dialog to **UserDoDialog**. It is most often a value from the enumerated type **InteractionCommand**.

UDDS_complete is set non-zero upon dialog completion. *UDDS_reponse* should be set before *UDDS_complete* is set non-zero.

UDDS_boxRunByCurrentThread is set non-zero if the box is run by the current thread.

UDDS_dialog stores the optr of the dialog that is currently up. This optr is needed internally by **UserDoDialog** for loop dispatching mode, to help ascertain which events should be dispatched and which saved off.

UDDS_queue stores the backed up queue of events arriving for the thread, but not dispatched due to the determination that they weren't relevant to the dialog's operation. These are reinserted into the queue upon completion of the dialog.

Library: **Objects/gInterC.def**

■ **UtilAsciiToHexError**

```
UtilAsciiToHexError      etype word
    UATH_NON_NUMERIC_DIGIT_IN_STRING  enum UtilAsciiToHexError
    UATH_CONVERT_OVERFLOW             enum UtilAsciiToHexError
```

Library: **system.def**

■ **UtilHexToAsciiFlags**

```
UtilHexToAsciiFlags      record
                                :11
    UHTAF_SBCS_STRING      :1
    UHTAF_THOUSANDS_SEPARATORS :1
        UHTAF_SIGNED_VALUE :1
    UHTAF_INCLUDE_LEADING_ZEROS :1
    UHTAF_NULL_TERMINATE   :1
UtilHexToAsciiFlags      end
```

Library: **system.def**



 **Reference book**

■ VarDataCHandler

```
VarDataCHandler    struct
    VDCH_dataType  word
    VDCH_handler   fptr.far
VarDataCHandler    ends
```

This structure is used as an entry in a class' vardata handler table. Usually, several of these structures will make up a table. For each entry within this table, specific vardata routines will call the *VDCH_handler* routine with the *VDCH_dataType*.

Library: **object.def**

■ VarDataEntry

```
VarDataEntry       struct
    VDE_dataType   word
    VDE_entrySize  word
    VDE_extraData  label byte
VarDataEntry       ends
```

This structure stores a vardata entry within an object.

VDE_dataType stores the vardata data type, a unique identifier.

VDE_entrySize stores the size of the vardata if it contains extra data; otherwise, this field is left null.

VDE_extraData marks the start of the extra data within the vardata entry.

Library: **object.def**

■ VarDataFlags

```
VarDataFlags       record
                                :14
    VDF_EXTRA_DATA    :1      ; set if data entry has extra data
    VDF_SAVE_TO_STATE :1      ; set if this data entry should be
                                ; saved to state file
VarDataFlags       end
```

Library: **object.def**



■ VarDataHandler

```
VarDataHandler      struct
    VDH_dataType    word          ; data type for this handler
    VDH_handler      nptr.far     ; handler routine
VarDataHandler      ends
```

This structure defines a handler in a **VarDataHandlerTable**.

Library: **object.def**

■ VarGeoData

```
VarGeoData          struct
    VGD_lineWidth   word
    VGD_centerOffset word
    VGD_secondWidth word
VarGeoData          ends
```

Library: **Objects/visC.def**

■ VarObjRelocation

```
VarObjRelocation    struct
    VOR_type        VarObjRelocationType
    VOR_offset      word
VarObjRelocation    ends
```

Library: **object.def**

■ VarObjRelocationType

```
VarObjRelocationType record
    VORT_DATA_TYPE    :14      ; high 14 bits of VarData type constant.
    VORT_RELOC_TYPE    ObjRelocationType:2
VarObjRelocationType end
```

Library: **object.def**

■ VChar

VChar etype byte

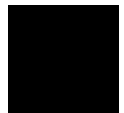
VC_NULL	enum VChar, 0x0 ;NULL
VC_CTRL_A	enum VChar, 0x1 ;<ctrl>-A
VC_CTRL_B	enum VChar, 0x2 ;<ctrl>-B
VC_CTRL_C	enum VChar, 0x3 ;<ctrl>-C
VC_CTRL_D	enum VChar, 0x4 ;<ctrl>-D
VC_CTRL_E	enum VChar, 0x5 ;<ctrl>-E
VC_CTRL_F	enum VChar, 0x6 ;<ctrl>-F
VC_CTRL_G	enum VChar, 0x7 ;<ctrl>-G
VC_CTRL_H	enum VChar, 0x8 ;<ctrl>-H
VC_CTRL_I	enum VChar, 0x9 ;<ctrl>-I
VC_CTRL_J	enum VChar, 0xa ;<ctrl>-J
VC_CTRL_K	enum VChar, 0xb ;<ctrl>-K
VC_CTRL_L	enum VChar, 0xc ;<ctrl>-L
VC_CTRL_M	enum VChar, 0xd ;<ctrl>-M
VC_CTRL_N	enum VChar, 0xe ;<ctrl>-N
VC_CTRL_O	enum VChar, 0xf ;<ctrl>-O
VC_CTRL_P	enum VChar, 0x10 ;<ctrl>-P
VC_CTRL_Q	enum VChar, 0x11 ;<ctrl>-Q
VC_CTRL_R	enum VChar, 0x12 ;<ctrl>-R
VC_CTRL_S	enum VChar, 0x13 ;<ctrl>-S
VC_CTRL_T	enum VChar, 0x14 ;<ctrl>-T
VC_CTRL_U	enum VChar, 0x15 ;<ctrl>-U
VC_CTRL_V	enum VChar, 0x16 ;<ctrl>-V
VC_CTRL_W	enum VChar, 0x17 ;<ctrl>-W
VC_CTRL_X	enum VChar, 0x18 ;<ctrl>-X
VC_CTRL_Y	enum VChar, 0x19 ;<ctrl>-Y
VC_CTRL_Z	enum VChar, 0x1a ;<ctrl>-Z
VC_ESCAPE	enum VChar, 0x1b ;ESC

; Extended keyboard codes -- those normally in ASCII ctrl set
; CTRL <key> sequences pressed by user will also be mapped here.

VC_BACKSPACE	= VC_CTRL_H
VC_TAB	= VC_CTRL_I
VC_LF	= VC_CTRL_J
VC_ENTER	= VC_CTRL_M
VC_BLANK	enum VChar, 0x20 ;space

; Numeric keypad keys

VC_NUMPAD_ENTER	enum VChar, 0xd ;* only on PS/2 keyboards
VC_NUMPAD_DIV	enum VChar, '/' ;* only on PS/2 keyboards
VC_NUMPAD_MULT	enum VChar, '*'
VC_NUMPAD_PLUS	enum VChar, '+'
VC_NUMPAD_MINUS	enum VChar, '-'
VC_NUMPAD_PERIOD	enum VChar, '.'
VC_NUMPAD_0	enum VChar, '0'



```

VC_NUMPAD_1          enum VChar, '1'
VC_NUMPAD_2          enum VChar, '2'
VC_NUMPAD_3          enum VChar, '3'
VC_NUMPAD_4          enum VChar, '4'
VC_NUMPAD_5          enum VChar, '5'
VC_NUMPAD_6          enum VChar, '6'
VC_NUMPAD_7          enum VChar, '7'
VC_NUMPAD_8          enum VChar, '8'
VC_NUMPAD_9          enum VChar, '9'

; Extended keyboard codes -- non-ASCII

VC_F1                enum VChar, 0x80 ; Function keys
VC_F2                enum VChar, 0x81
VC_F3                enum VChar, 0x82
VC_F4                enum VChar, 0x83
VC_F5                enum VChar, 0x84
VC_F6                enum VChar, 0x85
VC_F7                enum VChar, 0x86
VC_F8                enum VChar, 0x87
VC_F9                enum VChar, 0x88
VC_F10               enum VChar, 0x89
VC_F11               enum VChar, 0x8a ; * only on PS/2 keyboards
VC_F12               enum VChar, 0x8b ; * only on PS/2 keyboards
VC_F13               enum VChar, 0x8c ; * non-standard key
VC_F14               enum VChar, 0x8d ; * non-standard key
VC_F15               enum VChar, 0x8e ; * non-standard key
VC_F16               enum VChar, 0x8f ; * non-standard key

VC_UP                enum VChar, 0x90 ; Cursor keys
VC_DOWN              enum VChar, 0x91
VC_RIGHT             enum VChar, 0x92
VC_LEFT              enum VChar, 0x93
VC_HOME              enum VChar, 0x94 ; Scroll commands
VC_END               enum VChar, 0x95
VC_PREVIOUS          enum VChar, 0x96
VC_NEXT              enum VChar, 0x97
VC_INS               enum VChar, 0x98 ; INS
VC_DEL               enum VChar, 0x9a ; DEL

VC_PRINTSCREEN        enum VChar, 0x9b ; * from <shift>-NUMPAD_MULT
                        ; Appears as key only on PS/2
VC_PAUSE              enum VChar, 0x9c ; * from <ctrl>-NUMLOCK
                        ; Appears as key only on PS/2
VC_BREAK              enum VChar, 0x9e ; * from <ctrl>- or <alt>-combo
                        ; with various keys
VC_SYSTEMRESET        enum VChar, 0x9f ; <ctrl>-<alt>-<del> combo

; Joystick control keys (0xa0 - 0xa9)

```

```

VC_JOYSTICK_0          enum VChar, 0xa0 ; joystick 0 degrees
VC_JOYSTICK_45         enum VChar, 0xa1 ; joystick 45 degrees
VC_JOYSTICK_90         enum VChar, 0xa2 ; joystick 90 degrees
VC_JOYSTICK_135        enum VChar, 0xa3 ; joystick 135 degrees
VC_JOYSTICK_180        enum VChar, 0xa4 ; joystick 180 degrees
VC_JOYSTICK_225        enum VChar, 0xa5 ; joystick 225 degrees
VC_JOYSTICK_270        enum VChar, 0xa6 ; joystick 270 degrees
VC_JOYSTICK_315        enum VChar, 0xa7 ; joystick 315 degrees
VC_FIRE_BUTTON_1       enum VChar, 0xa8 ; fire button #1
VC_FIRE_BUTTON_2       enum VChar, 0xa9 ; fire button #2

; Shift Keys (0xe0 - 0xe7)

VC_LALT                enum VChar, 0xe0
VC_RALT                enum VChar, 0xe1
VC_LCTRL               enum VChar, 0xe2
VC_RCTRL               enum VChar, 0xe3
VC_LSHIFT              enum VChar, 0xe4
VC_RSHIFT              enum VChar, 0xe5
VC_SYSREQ              enum VChar, 0xe6 ; * Not on base PC keyboard.
                                   ; On PS/2 keyboards, is
                                   ; accessed via ALT PRINTSCREEN

VC_ALT_GR              enum VChar, 0xe7

; Toggle state keys (0xe8 - 0xef)

VC_CAPSLOCK            enum VChar, 0xe8
VC_NUMLOCK             enum VChar, 0xe9
VC_SCROLLLOCK          enum VChar, 0xea

; Extended state keys (0xf0 - 0xf7)

; Invalid key

VC_INVALID_KEY         enum VChar, 0xff

```

The previous represent the low byte of the character value only when the high byte is CS_CONTROL.

Library: **input.def**



■ VCR_param

```
VCR_param          struct
    VCR_routine     dword
    VCR_BP_param    word
    VCR_DX_param    word
    VCR_CX_param    word
VCR_param          ends
```

This structure stores stack parameters used in MSG_VIS_CALL_ROUTINE.

Library: **Objects/visC.def**

■ ViewCommandType

```
ViewCommandType    etype word
    VCT_ZOOM_IN      enum ViewCommandType;no other data
    VCT_ZOOM_OUT     enum ViewCommandType;no other data
    VCT_SET_SCALE    enum ViewCommandType;data is scale %
```

Library: **ui.def**

■ ViewSize

```
ViewSize           etype word, 8000h
    VS_TYPICAL      enum ViewSize   ;choose size typical of the specific UI
    VS_SMALL        enum ViewSize   ;choose a small size
    VS_LARGE        enum ViewSize   ;choose a large size
```

Library: **Objects/gViewC.def**

■ ViewTargetInfo

```
ViewTargetInfo      struct
    VTI_target       TargetReference ; Final target object within content
    VTI_content      TargetReference ; The content object itself
ViewTargetInfo      ends
```

Library: **Objects/gViewC.def**

■ VisAddRectFlags

```
VisAddRectFlags    record
    VARF_NOT_IF_ALREADY_INVALID    :1
    VARF_ONLY_REDRAW_MARGINS      :1
    VARF_UPDATE_WILL_HAPPEN       :1
                                   :5
VisAddRectFlags    end
```

VARF_NOT_IF_ALREADY_INVALID

Don't invalidate the rectangle if any node going up to the win group has its image or window marked invalid.

VARF_ONLY_REDRAW_MARGINS

This flag indicates that the object is invalidating old bounds, and can optimize invalidation if desired by splitting the message into four, one for each margin, in some cases.

VARF_UPDATE_WILL_HAPPEN

The caller knows of an impending update, so adding the rectangle to the update region rather than invalidating is a reasonable (and fast) thing to do.

Library: **Objects/visC.def**

■ VisAddRectParams

```
VisAddRectParams    struct
    VARP_bounds    Rectangle    ;rect to invalidate
    VARP_flags     VisAddRectFlags
    VARP_unused    byte         ;word align
VisAddRectParams    ends
```

Library: **Objects/visC.def**

■ VisAttrs

```
VisAttrs record
    VA_VISIBLE                :1
    VA_FULLY_ENABLED          :1
    VA_MANAGED                :1
    VA_DRAWABLE               :1
```



```

VA_DETECTABLE          :1
VA_BRANCH_NOT_MINIMIZABLE :1
VA_OLD_BOUNDS_SAVED    :1
VA_REALIZED            :1
VisAttrs end

```

VA_VISIBLE This attribute is for WIN_GROUP's only. (Ignored if non-WIN-GROUP object) Set if object may be visually built out, meaning that it is allowed to be linked visually into a composite, and if that composite is realized, then it would be made visible, too.

VA_FULLY_ENABLED
Flag to tell whether a vis object is enabled or not. If cleared visual objects typically don't allow clicks and are drawn in a 50% pattern, even if they're not generic. Set by MSG_SPEC_BUILD, MSG_SPEC_NOTIFY_ENABLED, and MSG_SPEC_NOTIFY_NOT_ENABLED in generic objects.

VA_MANAGED
Set if object is managed, that is, space is reserved for it in the composite via the geometry manager. Set if no space should be allocated for it. A message will allow this status to change, & if the window on which this object is placed is realized, then we must have the geometry manager redo the geometry.

VA_DRAWABLE
Set if object is drawn, set if invisible. A message will allow setting of this flag. If the window on which this object resides is realized when this happens, the bounding box of the object will be invalidated on that window.

VA_DETECTABLE
Set if object might respond to mouse, ptr, kbd, etc. data. set if composite shouldn't bother to send such data onto the child. This bit will only be tested when a composite is passing a message down to its children. Display only objects should have this bit clear. The message allowing changing of this bit will not change any grab in progress on the object. Note that a window composite may not have this bit clear. Basically, we can't avoid an implied grab to a window.

VA_BRANCH_NOT_MINIMIZABLE
For Generic objects only (Would be a SpecAttrs if room). Used to keep modal windows up on screen even if they are generic children of a primary which is minimized.

VA_OLD_BOUNDS_SAVED

(Would be in optFlags or geoFlags if room). Flag to keep track of whether old bounds have been saved for the object by the geometry manager for use by the invalidation mechanism. Bounds are kept in variable data type VVDT_OLD_BOUNDS.

VA_REALIZED

Set by default VisOpen and VisClose messages to indicate if object is realized (visible within a window) onscreen or not. Is also used to ensure that all objects receive a VisOpen, even if the visible part of the tree has just been added to a branch that is already realized, & then updated—this should all be done in one operation, without delaying the update—and the top object of the branch to be realized should be marked as “WINDOW_INVALID”, even if it is a non-windowed object, as the MSG_UPDATE_WINDOWS will follow the path bits & figure out that the object needs to be sent a MSG_VIS_OPEN. May not be set by MSG_VIS_SET_ATTR.

Library: **Objects/visC.def**

■ VisCompGeoAttrs

```
VisCompGeoAttrs    record
    VCGA_ORIENT_CHILDREN_VERTICALLY    :1
    VCGA_INCLUDE_ENDS_IN_CHILD_SPACING :1
    VCGA_ALLOW_CHILDREN_TO_WRAP        :1
    VCGA_ONE_PASS_OPTIMIZATION         :1
    VCGA_CUSTOM_MANAGE_CHILDREN        :1
    VCGA_HAS_MINIMUM_SIZE               :1
    VCGA_WRAP_AFTER_CHILD_COUNT        :1
    VCGA_ONLY_DRAW_IN_MARGINS          :1
VisCompGeoAttrs    end
```

VCGA_ORIENT_CHILDREN_VERTICALLY

Place the composite's children vertically, rather than horizontally.

VCGA_INCLUDE_ENDS_IN_CHILD_SPACING

When used with full justification, divides the spacing up so that there is as much space allocated before the first child and after the last child as there are between the children. An example of this is a motif reply bar. When this is clear, there is no space allocated at the ends of the composite.

VCGA_ALLOW_CHILDREN_TO_WRAP

Allows children to wrap if their combined lengths will not allow them to fit inside the bounds of this object's parent. The composite will keep within the bounds of its parent and wrap



the children as necessary. When this is clear, the children will force the composite to be as big as needed to fit the children on one line (unless, of course, the CAN_TRUNCATE_WIDTH_TO_FIT_PARENT flags are set.)

VCGA_ONE_PASS_OPTIMIZATION

This is an optimization which speeds up the geometry manager, only making one pass of sizing the children and using the sum of the sizes as the size of the composite. In order to use this flag, you must be sure that the children don't want to wrap, and are always one size, regardless of the size of the parent, such as buttons in a horizontal composite.

VCGA_CUSTOM_MANAGE_CHILDREN

Don't use the geometry manager to manage the children. This allows you to set up the sizes and positions of the children without the need of the geometry manager. If this flag is set, the composite will be default return its current size when asked to calculate its size, like a simple non-composite object.

VCGA_HAS_MINIMUM_SIZE

Geometry manager will send a MSG_VIS_COMP_GET_MIN_SIZE to this object if this flag is set, and always make the composite at least as big as that.

VCGA_WRAP_AFTER_CHILD_COUNT

Used in conjunction with VCGA_ALLOW_CHILDREN_TO_WRAP. If set, composite will wrap after a certain number of children, the number being obtained from a MSG_VIS_COMP_GET_CHILD_WRAP_COUNT.

VCGA_ONLY_DRAWS_IN_MARGINS

This flag can be set by a composite to optimize invalidation. If set, a composite whose image is invalid will only have its margins invalidated. Any visual child below it will have to have its image invalid in order to get invalidated. To get proper invalidations, the composite cannot draw anything inside its margins (that isn't the color of the background).

Library: **Objects/vCompC.def**

■ VisCompGeoDimensionAttrs

```
VisCompGeoDimensionAttrs record
  VCGDA_WIDTH_JUSTIFICATION      WidthJustification:2
  VCGDA_EXPAND_WIDTH_TO_FIT_PARENT :1
  VCGDA_DIVIDE_WIDTH_EQUALLY      :1
  VCGDA_HEIGHT_JUSTIFICATION      HeightJustification:2
  VCGDA_EXPAND_HEIGHT_TO_FIT_PARENT:1
  VCGDA_DIVIDE_HEIGHT_EQUALLY      :1
VisCompGeoDimensionAttrs end
```

VCGDA_WIDTH_JUSTIFICATION

Horizontal justifications for placing the children. Note that horizontal full justification is only meaningful if the composite is oriented horizontally.

VCGDA_EXPAND_WIDTH_TO_FIT_PARENT

Composite will try to expand to fill the available width of the parent. By default, a composite will only be as wide as its children require.

VCGDA_DIVIDE_WIDTH_EQUALLY

Will attempt to divide width equally among its manageable children if oriented horizontally. Does not guarantee that the children can cooperate (the size can only be suggested).

VCGDA_HEIGHT_JUSTIFICATION

Vertical justifications for placing the children. Note that vertical full justification is only meaningful if the composite is oriented vertically.

VCGDA_EXPAND_HEIGHT_TO_FIT_PARENT

Composite will try to expand to fill the available height of the parent. By default, a composite will only be as tall as its children require.

VCGDA_DIVIDE_HEIGHT_EQUALLY

Will attempt to divide height equally among its manageable children if oriented horizontally. Does not guarantee that the children can cooperate (the size can only be suggested).

Library: **Objects/vCompC.def**



■ VisCompSpacingMarginsInfo

```
VisCompSpacingMarginsInfo    record
    VCSMI_USE_THIS_INFO      :1
    VCSMI_LEFT_MARGIN        :3
    VCSMI_TOP_MARGIN         :3
    VCSMI_RIGHT_MARGIN       :3
    VCSMI_BOTTOM_MARGIN      :3
    VCSMI_CHILD_SPACING      :3
VisCompSpacingMarginsInfo    end
```

VCSMI_USE_THIS_INFO
VisCompCalcNewSize uses this info for the composite's spacing and margins. If zero, will send MSG_VIS_COMP_GET_CHILD_SPACING and MSG_VIS_COMP_GET_MARGINS to get the information it needs.

VCSMI_LEFT_MARGIN

VCSMI_TOP_MARGIN

VCSMI_RIGHT_MARGIN

VCSMI_BOTTOM_MARGIN
Margins to use when doing geometry, rather than sending a MSG_VIS_COMP_GET_MARGINS.

VCSMI_CHILD_SPACING
Spacing (both between children and between wrapped lines) to use in lieu of MSG_VIS_COMP_GET_CHILD_SPACING.

Library: **Objects/vCompC.def**

■ VisContentAttrs

```
VisContentAttrs    record
    VCNA_SAMW_WIDTH_AS_VIEW      :1
    VCNA_SAME_HEIGHT_AS_VIEW    :1
    VCNA_LARGE_DOCUMENT_MODEL    :1
    VCNA_WINDOW_COORDINATE_MOUSE_EVENTS :1
    VCNA_ACTIVE_MOUSE_GRAB_REQUIRES_LARGE_EVENTS :1
    VCNA_VIEW_DOC_BOUNDS_SET_MANUALLY :1
    VCNA_VIEW_DOES_NOT_WIN_SCROLL :1
VisContentAttrs    end
```

VCNA_SAME_WIDTH_AS_VIEW
Set if the content's width should just follow the subview's window width, if possible. You want to set this if the view is not supposed to be horizontally scrollable.

VCNA_SAME_HEIGHT_AS_VIEW

Set if the content's height should just follow the subview's window height, if possible. You want to set this if the view is not supposed to be vertically scrollable.

VCNA_LARGE_DOCUMENT_MODEL

Set if using a large document model, in which this object will be a large (32-bit) VisContent. Effects:

Bounds are larger than the graphics space under this model, so the 16-bit Visible bounds of this object are meaningless. The application must initialize the GenView, or use MSG_GEN_VIEW_SET_DOC_SIZE, to set the view's document size.

If no active mouse grab: Incoming mouse events are converted into 32-bit LARGE mouse events & sent to the VisContent, where the default handler will send them on to the first visible child, thereby providing correct behavior for the single-child case. Applications having multiple visible layers (32-bit children of VisContents) must intercept these messages & direct them to the correct layer.

MSG_VIS_DRAW is by default sent on to the first visible child, thereby providing correct behavior for the single-child case. Applications having multiple visible layers (32-bit children of VisContents) must intercept this message & direct it to the correct layer(s).

VCNA_WINDOW_COORDINATE_MOUSE_EVENTS

Required if VCNA_LARGE is set. Support for this bit requires a VisContentClass object. Set if the GenView associated with this content has been set up to send mouse events in window coordinates, instead of document coordinates (For either 32-bit support capability or fractional mouse position capability). Indicates that the VisContent will need to convert the coordinates to document coordinates before sending them on. This is done via the equation:

$$\text{Doc Coords} = (\text{Win Coords} / \text{Scale Factor}) + \text{Doc Origin}$$

VCNA_ACTIVE_MOUSE_GRAB_REQUIRES_LARGE_EVENTS

This bit only used by VisContentClass objects. Set/cleared by MSG_VIS_VUP_ALTER_INPUT_FLOW routine to indicate whether the current mouse grab wishes to receive LARGE mouse events in place of the standard ones. Note that this mechanism may be used even when the bit VCNA_LARGE is not set.



VCNA_VIEW_DOC_BOUNDS_SET_MANUALLY

Not often used, this will prevent the content from automatically sending off its size to the view on geometry updates. The view's document bounds must be set some other way. Setting VCNA_LARGE_DOCUMENT_MODEL will also cause this behavior.

VCNA_VIEW_DOES_NOT_WIN_SCROLL

Set to indicate that the view does not actually scroll its window, it just sends origin messages to the content when the user interacts with the scrollbar. Visual invalidation will use this flag to invalidate the correct region of the content. Should be set whenever ATTR_GEN_VIEW_DO_NOT_WIN_SCROLL is set in the view.

Library: **Objects/vCntC.def**

■ VisGeoAttrs

```
VisGeoAttrs      record
;
; Geometry state flags
;
VGA_GEOMETRY_CALCULATED      :1
VGA_NO_SIZE_HINTS            :1
;
; Miscellaneous flags
;
VGA_NOTIFY_GEOMETRY_INVALID   :1
VGA_DONT_CENTER               :1
VGA_USE_VIS_SET_POSITION      :1
VGA_USE_VIS_CENTER            :1
VGA_ONLY_RECALC_SIZE_WHEN_INVALID :1
VGA_ALWAYS_RECALC_SIZE        :1
VisGeoAttrs      end
```

VGA_GEOMETRY_CALCULATED

Set in an after the first time an object's geometry has been calculated. This is used by the specific UI size hint handlers to figure out whether an initial size hint should be applied to an object or not. This is set at the time an object's size and position has been completely determined. It can be cleared if need be.

VGA_NO_SIZE_HINTS

Specific attribute only: if set, we have checked to see if the object has one or more of HINT_INITIAL_SIZE, HINT_MINIMUM_SIZE, HINT_MAXIMUM_SIZE, HINT_FIXED_SIZE set, and it doesn't. We clear this flag if one of the desired size methods are called.

Reference book

VGA_NOTIFY_GEOMETRY_VALID

If set, geometry manager will notify object when its geometry messages have all been finished and its geometry is valid.

VGA_DONT_CENTER

Allows an object to individually override the parent composite's centering along its width. Will appear on the top (of a horizontal composite, left edge if vertical) instead.

VGA_USE_VIS_SET_POSITION

All objects that don't use the default Vis or VisComp handlers for MSG_VIS_SET_POSITION and MSG_VIS_POSITION_BRANCH should set this flag. It's an optimization that allows static calls to the geometry manager.

VGA_USE_VIS_CENTER

If set, geometry manager uses standard vis or visComp center message to calculate the object's center.

VGA_ONLY_RECALC_SIZE_WHEN_INVALID

Set this if your object wants its message called the first time after its geometry is invalid, and then always return the current size. Example: buttons in a horizontal composite.

VGA_ALWAYS_RECALC_SIZE

If set, doesn't do optimizations to calculate the size of this object. May be needed for composites that expand to fit and center their children to match their parent, or some other obscure cases where the size might change from one call to another.

Library: **Objects/visC.def**

■ VisInputFlowGrabFlags

```
VisInputFlowGrabFlags    record
    VIFGF_NOT_HERE       :1
                          :1
    VIFGF_FORCE           :1
    VIFGF_GRAB            :1
    VIFGF_KBD             :1
    VIFGF_MOUSE           :1
    VIFGF_LARGE           :1
    VIFGF_PTR             :1
VisInputFlowGrabFlags    end
```

VIFGF_NOT_HERE

This flag overrides all other flags! Set if this request should not



be honored here, but instead sent on up the hierarchy with this bit cleared. This bit exists for two reasons:

- 1) So that nodes can tell the difference between messages coming up from objects below & those requests which it has made for itself, which should be handled by the next node up.
- 2) Thus allowing MSG_VIS_VUP_ALTER_INPUT_FLOW to be sent to the object making the request itself, thereby allowing nodes the freedom to direct the message in directions other than the visual hierarchy, if the next node is not in that direction.

VIFGF_FORCE

If VIFGF_GRAB is set and GrabType = VIFGT_ACTIVE, set to force grab away from current owner, clear if we should leave any current owner alone.

VIFGF_GRAB

Set to grab, clear to release.

Note 1: object must be passed in release case as well as grab. Release will not occur unless object matches.

Note 2: Only one obj may have the active grab at any one time, whereas any number of objects may add themselves to a passive list.

VIFGF_KBD

Set to grab/release kbd (keyboard).

VIFGF_MOUSE

Set to grab/release mouse.

VIFGF_LARGE

If VIFGF_MOUSE and VIFGF_GRAB: LARGE mouse events requested

VIFGF_PTR

If VIFGF_MOUSE, set if ptr events need to be sent.

Library: **Objects/visC.def**

■ VisInputFlowGrabType

VisInputFlowGrabType	etype byte
VIFGT_ACTIVE	enum VisInputFlowGrabType
VIFGT_PRE_PASSIVE	enum VisInputFlowGrabType
VIFGT_POST_PASSIVE	enum VisInputFlowGrabType

Library: **Objects/visC.def**

Reference book

■ VisLargeTextAttrs

```
VisLargeTextAttrs  record
    VLTA_EXACT_HEIGHT      :1
                           :15
VisLargeTextAttrs  end
```

Library: **Objects/vLTextC.def**



■ VisLargeTextDisplayModes

```

VisLargeTextDisplayModes  etype word
    VLTDM_PAGE              enum VisLargeTextDisplayModes
    ;
    ; In page mode the values stored in the region array are used.
    ;
    VLTDM_CONDENSED         enum VisLargeTextDisplayModes
    ;
    ; In condensed mode all text regions are put vertically one after the
    ; other. Calculated fields are:
    ; - VLTRAE_spatialPosition
    ;     The x position is taken from vardata. The y position is the
    ;     sum of the region heights (VLTRAE_size.XYS_height) for all
    ;     preceding regions plus the offset stored in vardata plus the
    ;     page spacing stored in vardata.
    ;
    VLTDM_GALLEY            enum VisLargeTextDisplayModes
    ;
    ; In galley mode all text regions are put vertically one after the
    ; other as in condensed mode, except that the computed heights are
    ; used (so that the regions are jammed right next to each other).
    ; Calculated fields are:
    ; - VLTRAE_spatialPosition
    ;     The x position is taken from vardata. The y position is the
    ;     sum of the region calculated heights (VLTRAE_calcHeight) for all
    ;     preceding regions plus the offset stored in vardata plus the
    ;     page spacing stored in vardata.
    ; - VLTRAE_size.XYS_height
    ;     Taken from VLTRAE_calcHeight when being used to clear
    ;
    VLTDM_DRAFT_WITH_STYLES  enum VisLargeTextDisplayModes
    VLTDM_DRAFT_WITHOUT_STYLESenum VisLargeTextDisplayModes
    ;
    ; In draft mode all text regions are forced to a standard size and are
    ; then put one after the other as in galley mode. Calculated fields
    ; are:
    ; - VLTRAE_spatialPosition
    ;     Same as galley mode
    ; - VLTRAE_size
    ;     Taken from VTDMMD_draftRegionSize
    ; - VLTRAE_region
    ;     Always 0 (rectangular region)

```

Library: **Objects/vLTextC.def**

■ VisLargeTextFlags

```

VisLargeTextFlags  record
    VLTF_HEIGHT_NOTIFY_PENDING :1
                                :15
VisLargeTextFlags  end

```

Library: **Objects/vLTextC.def**

■ VisLargeTextRegionArrayElement

```

VisLargeTextRegionArrayElement  struct
    VLTRAE_charCount      dword      ;# characters in region
    VLTRAE_lineCount      dword      ;# lines in region
    VLTRAE_section        word       ;section number
    VLTRAE_spatialPosition  PointDWord ;position (in 32 bit space)
    VLTRAE_size            XYSIZE     ;region size
    VLTRAE_calcHeight      WBFixed    ;computed height of text
    VLTRAE_region          dword      ;db item containing region or
                                ;0 for rectangular
    VLTRAE_flags           VisLargeTextRegionFlags
    VLTRAE_reserved        byte 3 dup (?)
VisLargeTextRegionArrayElement  ends

```

Library: **Objects/vLTextC.def**

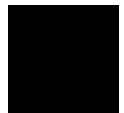
■ VisLargeTextRegionFlags

```

VisLargeTextRegionFlags  record
    VLTRF_ENDED_BY_COLUMN_BREAK :1
    VLTRF_EMPTY                :1
                                :14
VisLargeTextRegionFlags  end

```

Library: **Objects/vLTextC.def**



■ VisMoniker

```
VisMoniker      struct
    VM_type      VisMonikerType <>
    VM_width      word
    VM_data      label VisMonikerText
VisMoniker      ends
```

This structure defines a visual moniker. Individual monikers can be combined into a list using a **VisMonikerList** structure.

VM_type stores the type of vis moniker. The low byte of the record determines whether this is a moniker or a list of monikers.

VM_width stores the cached width of the moniker. This value will be calculated if this entry is null and if VMLET_GSTRING bit is *not* set. (The cached height is kept with the gstring.)

VM_data stores the start of the visual moniker data. If VMLET_GSTRING bit set in *VM_type* is set then a **VisMonikerGString** structure starts here. Otherwise a **VisMonikerText** structure starts here.

Library: **Objects/visC.def**

■ VisMonikerCachedWidth

```
VisMonikerCachedWidth  record
    VMCW_HINTED          :1 ;If set then low 15 bits are cache info
    VMCW_BERKELEY_9      :7 ;Cached width for Berkeley 9
    VMCW_BERKELEY_10     :8 ;Cached width for Berkeley 10
VisMonikerCachedWidth  end
```

Library: **visC.def**

■ VisMonikerDataType

```
VisMonikerDataType  etype byte
    VMDT_NULL          enum VisMonikerDataType
    VMDT_VIS_MONIKER    enum VisMonikerDataType
    VMDT_TEXT           enum VisMonikerDataType
    VMDT_GSTRING        enum VisMonikerDataType
    VMDT_TOKEN          enum VisMonikerDataType
```

VMDT_NULL

Indicates that there is no source.

MSG_GEN_REPLACE_VIS_MONIKER will just free current vis moniker. Not valid for MSG_VIS_CREATE_VIS_MONIKER and MSG_GEN_CREATE_VIS_MONIKER. *CVMF_source*,

RVMF_source, *CVMF_length*, *RVMF_length*, *CVMF_width*, *CVMF_height*, *RVMF_width*, and *RVMF_height* are unused.

VMDT_VIS_MONIKER

Indicates that source is a complete VisMoniker structure. *CVMF_length* and *RVMF_length* indicate the size of the complete VisMoniker structure. *CVMF_width*, *CVMF_height*, *RVMF_width*, and *RVMF_height* are unused.

VMDT_TEXT

Indicates that source is a text string. If null-terminated, *CVMF_length* and *RVMF_length* should be zero. Otherwise, *CVMF_length* and *RVMF_length* indicate the length of the text string. A **VisMoniker** structure will be created for the text string. *CVMF_width*, *CVMF_height*, *RVMF_width*, and *RVMF_height* are unused.

VMDT_GSTRING

Indicates that source is a graphics string. If *CVMF_length* and *RVMF_length* are 0, the gstring length will be determined by scanning the graphics string for GR_END_STRING. Otherwise, *CVMF_length* and *RVMF_length* indicate the length of the graphics string. *CVMF_width*, *CVMF_height*, *RVMF_width* and *RVMF_height* indicate the width and height of the graphics string. If either is zero, the width and height will be computed by examining the string. A **VisMoniker** structure will be created for the graphics string.

VMDT_TOKEN

Indicates that source is a **GeodeToken**. *CVMF_length*, *RVMF_length*, *CVMF_width*, *CVMF_height*, *RVMF_width*, and *RVMF_height* are unused. The destination object must be usable to use this data type because the specific UI must decide which moniker to choose from the moniker in the Token Database.

Library: **Objects/visC.def**

■ VisMonikerGString

```
VisMonikerGString struct
    VMGS_height      word           ;cached gstring height
    VMGS_gstring     label byte     ;start of gstring
VisMonikerGString ends
```

This structure defines the data at *VM_data* within a visual moniker if the visual moniker is a graphics string.

Library: **Objects/visC.def**



■ VisMonikerListEntry

```
VisMonikerListEntry    struct
    VMLE_type          VisMonikerListEntryType <>
    VMLE_moniker       optr
VisMonikerListEntry    ends
```

This structure is used for elements in a **VisMonikerList**. The list consists of any number of these elements inside a chunk.

VMLE_type stores the type of moniker. This type is used during a moniker search to find a desired moniker.

VMLE_moniker stores the optr of the moniker.

Library: **Objects/visC.def**

■ VisMonikerListEntryType

```
VisMonikerListEntryType  record
    :2
    VMLET_GS_SIZE         DisplaySize:2
    VMLET_STYLE           VMStyle:4
    ;
    ; bits below must match VisMonikerType
    ;
    VMLET_MONIKER_LIST    :1
    VMLET_GSTRING         :1
    VMLET_GS_ASPECT_RATIO DisplayAspectRatio:2
    VMLET_GS_COLOR        DisplayClass:4
VisMonikerListEntryType  end
```

VMLET_GS_SIZE
If is a GString, size of moniker.

VMLET_STYLE
Style of this moniker

VMLET_MONIKER_LIST
The UIC compiler always sets this if flag, indicating that this record is within a VisMonikerListElement, not the actual VisMoniker itself.

VMLET_GSTRING
TRUE if this moniker is a graphics string VisMonikerGString).
If false, this moniker is text (VisMonikerText).

VMLET_GS_ASPECT_RATIO
If is a GString, aspect ratio of GString.

VMLET_GS_COLOR

If is a GString, color requirements of GString.

Library: **Objects/visC.def**

■ VisMonikerSearchFlags

```
VisMonikerSearchFlags    record
    VMSF_STYLE            VMStyle:4
                        :1
    VMSF_COPY_CHUNK       :1
    VMSF_REPLACE_LIST     :1
    VMSF_GSTRING          :1
                        :8      ; Internal use only
VisMonikerSearchFlags    end
```

VMSF_STYLE

Preferred style of moniker

VMSF_COPY_CHUNK

True to copy the VisMoniker chunk into the specified object block, if the search is successful, and the moniker is not in that block already.

VMSF_REPLACE_LIST

True to replace to VisMonikerList chunk with the VisMoniker, if the search is successful. The idea is that the chunk handle for the list now points to the moniker.

VMSF_GSTRING

True if a gstring moniker is expected (i.e. a VisMonikerGString), false if a text moniker is expected (i.e. a VisMonikerText).

Library: **Objects/visC.def**

■ VisMonikerSourceType

```
VisMonikerSourceType    etype byte
    VMST_FPTR            enum VisMonikerSourceType
    VMST_OPTR            enum VisMonikerSourceType
    VMST_HPTR            enum VisMonikerSourceType
```

VMST_FPTR Indicates source is referenced by a fptr. *CVMF_source* and *RVMF_source* fields are a fptr.

VMST_OPTR Indicates source is referenced by a optr. *CVMF_source* and *RVMF_source* fields are an optr.



VMST_HPTR Indicates source is referenced by a hptr and offset.
CVMF_source and *RVMF_source* fields are a hptr and offset
 within the block.

Library: **Objects/visC.def**

■ VisMonikerText

```
VisMonikerText    struct
    VMT_mnemonicOffset    byte        ;offset to mnemonic, -1 if none
    VMT_text              label byte  ;start of null-terminated text
VisMonikerText    ends
```

This structure defines the data at *VM_data* within a **VisMoniker** for text monikers.

Library: **Objects/visC.def**

■ VisMonikerType

```
VisMonikerType    record
    VMT_MONIKER_LIST      :1
    VMT_GSTRING            :1
    VMT_GS_ASPECT_RATIO   DisplayAspectRatio:2
    VMT_GS_COLOR           DisplayClass:4
VisMonikerType    end
```

VMT_MONIKER_LIST

The UIC compiler always clears this flag, indicating that this record is within a VisMoniker.

VMT_GSTRING

True if this moniker is a graphics string (VisMonikerGString).
 If false, this moniker is text (VisMonikerText).

VMT_GS_ASPECT_RATIO

If is a GString, aspect ratio of moniker.

VMT_GS_COLOR

Color requirements of GString.

Library: **Objects/visC.def**

■ VisMouseGrab

```
VisMouseGrab      struct
    VMG_object     optr
    VMG_gWin       hptr.Window <>
    VMG_translation PointDWord
    VMG_flags      VisInputFlowGrabFlags
    VMG_unused     byte
VisMouseGrab      ends
```

This structure stores data associated with an object requesting a mouse grab. This structure is filled in by the handler for MSG_VIS_VUP_ALTER_INPUT_FLOW.

VMG_object stores the optr of the object having the mouse grab.

VMG_gWin stores the handle of the window that the object having grab resides in, or 0 if in same window as the **VisContent**.

VMG_translation stores the 32 bit translation to use for the grab, set by the handler for MSG_VIS_VUP_ALTER_INPUT_FLOW. This 32-bit translation, passed in *VAIFD_translation* in the above message, is set by large visible objects subclassing the message. Mouse event positions are adjusted by this amount before being sent out.

Library: **Objects/vCntC.def**

■ VisOptFlags

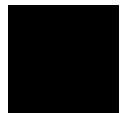
```
VisOptFlags      record
    VOF_GEOMETRY_INVALID      :1
    VOF_GEO_UPDATE_PATH      :1
    VOF_IMAGE_INVALID        :1
    VOF_IMAGE_UPDATE_PATH    :1
    VOF_WINDOW_INVALID       :1
    VOF_WINDOW_UPDATE_PATH   :1
    VOF_UPDATE_PENDING       :1
    VOF_UPDATING             :1
VisOptFlags      end
```

VOF_GEOMETRY_INVALID

Set by MSG_VIS_MARK_INVALID, which, if followed by a MSG_VIS_VUP_UPDATE_WIN_GROUP will insure that changes in the object bounds will be reflected in the window.

VOF_GEO_UPDATE_PATH

Set by MSG_VIS_MARK_INVALID to leave a trail to objects that have invalid geometry, for UPDATE_GEOMETRY to follow.



VOF_IMAGE_INVALID

Set by MSG_VIS_MARK_INVALID, which, if followed by a MSG_VIS_VUP_UPDATE_WIN_GROUP will insure that visual changes in the object will be reflected in the window.

VOF_IMAGE_UPDATE_PATH

Set by MSG_VIS_MARK_INVALID to leave a trail to objects that have invalid geometry, for UPDATE_WINDOWS_AND_IMAGE to follow.

VOF_WINDOW_INVALID

Set by MSG_VIS_MARK_INVALID, which, if followed by a MSG_VIS_VUP_UPDATE_WIN_GROUP will insure that changes in the window's view will be reflected in the window. (valid for windows only)

VOF_WINDOW_UPDATE_PATH

Set by MSG_VIS_MARK_INVALID to leave a trail to windows that have invalid views, for UPDATE_WINDOWS_AND_IMAGE to follow. (valid for composites only).

VOF_UPDATE_PENDING

Used for Group windows only, indicates that a MSG_VIS_UPDATE_WIN_GROUP is still in the UI event queue for this window, and hasn't arrived yet.

VOF_UPDATING

Set while updating visible branch, so we can give an error if we get into a nested update situation. Also may be useful for knowing how to update an object.

Library: **Objects/visC.def**

■ **VisRulerAttributes**

```
VisRulerAttributes record
  VRA_IGNORE_ORIGIN      :1
  VRA_SHOW_GUIDES        :1
  VRA_SHOW_GRID          :1
  VRA_SHOW_MOUSE         :1
  VRA_HORIZONTAL         :1
  VRA_MASTER              :1
                          :2
VisRulerAttributes end
```

Library: **ruler.def**

Reference book

■ **VisRulerConstrainStrategy**

```

VisRulerConstrainStrategy      record
    VRCS_OVERRIDE                :1
                                :1
    VRCS_SET_REFERENCE           :1
    VRCS_SNAP_TO_GRID_X_ABSOLUTE :1
    VRCS_SNAP_TO_GRID_Y_ABSOLUTE :1
    VRCS_SNAP_TO_GRID_X_RELATIVE :1
    VRCS_SNAP_TO_GRID_Y_RELATIVE :1
    VRCS_SNAP_TO_GUIDES_X        :1
    VRCS_SNAP_TO_GUIDES_Y        :1
    VRCS_CONSTRAIN_TO_HORIZONTAL_AXIS :1
    VRCS_CONSTRAIN_TO_VERTICAL_AXIS :1
    VRCS_CONSTRAIN_TO_UNITY_SLOPE_AXIS :1
    VRCS_CONSTRAIN_TO_NEGATIVE_UNITY_SLOPE_AXIS :1
    VRCS_CONSTRAIN_TO_VECTOR      :1
    VRCS_CONSTRAIN_TO_VECTOR_REFLECTION :1
    VRCS_INTERNAL                 :1
VisRulerConstrainStrategy      end

```

Library: **ruler.def**

■ **VisRulerNotifyGuideChangeBlockHeader**

```

VisRulerNotifyGuideChangeBlockHeader struct
    VRNGCBH_header      LMemBlockHeader
    VRNGCBH_vertGuideArray  word
    VRNGCBH_horizGuideArray word
VisRulerNotifyGuideChangeBlockHeader ends

```

Library: **ruler.def**

■ **VisRulerType**

```

VisRulerType      etype byte, 0
    VRT_INCHES      enum VisRulerType
    VRT_CENTIMETERS enum VisRulerType
    VRT_POINTS       enum VisRulerType
    VRT_PICAS        enum VisRulerType
    VRT_CUSTOM        enum VisRulerType, 0xfd;custom ruler definition
    VRT_NONE          enum VisRulerType, 0xfe;no rulers
    VRT_DEFAULT       enum VisRulerType, 0xff;use system default

```

Library: **ruler.def**



■ **VisTextAddNameParams**

```
VisTextAddNameParams    struct
    VTANP_name          fptr.char          ; pointer to name
    VTANP_size           word              ; length of name (0 if null-terminated)
    VTANP_flags          NameArrayAddFlags
    VTANP_data           VisTextNameData
VisTextAddNameParams    ends
```

Library: **Objects/vTextC.def**

■ **VisTextCachedRunInfo**

```
VisTextCachedRunInfo    struct
    VTCRI_lastCharAttrRun    dword
    VTCRI_lastParaAttrRun    dword
    VTCRI_lastTypeRun        dword
    VTCRI_lastGraphicRun     dword
VisTextCachedRunInfo    ends
```

Library: **Objects/vTextC.def**

■ **VisTextCachedUndoInfo**

```
VisTextCachedUndoInfo    struct
    VTCUI_vmChain          dword
    VTCUI_file             hptr
VisTextCachedUndoInfo    ends
```

Library: **Objects/vTextC.def**

■ **VisTextCharAttr**

```
VisTextCharAttr          struct
    VTCA_meta              StyleSheetElementHeader
    VTCA_fontID             FontID
    VTCA_pointSize          WBFixed
    VTCA_textStyles         TextStyle
    VTCA_color              ColorQuad
    VTCA_trackKerning        sword
    VTCA_fontWeight          byte
    VTCA_fontWidth           byte
    VTCA_extendedStyles     VisTextExtendedStyles
    VTCA_grayScreen          SystemDrawMask      ; foreground gray screen
    VTCA_pattern             GraphicPattern      ; Foreground pattern
```

```

VTCA_bgColor          ColorQuad          ; Background color
VTCA_bgGrayScreen     SystemDrawMask     ; Background gray screen
VTCA_bgPattern        GraphicPattern     ; Background pattern
VTCA_reserved         byte 7 dup (0)
VisTextCharAttr      ends

```

Library: **Objects/Text/tCommon.def**

■ VisTextCharAttrDiffs

```

VisTextCharAttrDiffs  struct
    VTCAD_diffs        VisTextCharAttrFlags
    VTCAD_extendedStyles VisTextExtendedStyles
    VTCAD_textStyles   TextStyle
    even
VisTextCharAttrDiffs  ends

```

Library: **Objects/vTextC.def**

■ VisTextCharAttrFlags

```

VisTextCharAttrFlags  record
    VTCAF_MULTIPLE_FONT_IDS      :1      ;Set if more than one font
    VTCAF_MULTIPLE_POINT_SIZES   :1      ;Set if more than one point size
    VTCAF_MULTIPLE_COLORS        :1      ;Set if more than one color
    VTCAF_MULTIPLE_GRAY_SCREEN   :1      ;Set if more than one gray screen
    VTCAF_MULTIPLE_PATTERNS      :1      ;Set if more than one hatch
    VTCAF_MULTIPLE_TRACK_KERNINGS :1      ;Set if more than
                                         ; one track kerning
    VTCAF_MULTIPLE_FONT_WEIGHTS  :1      ;Set if more than one font weight
    VTCAF_MULTIPLE_FONT_WIDTHS   :1      ;Set if more than one font width
    VTCAF_MULTIPLE_BG_COLORS     :1      ;Set if more than one bg color
    VTCAF_MULTIPLE_BG_GRAY_SCREEN :1      ;Set if more than one bg gray
    screen
    VTCAF_MULTIPLE_BG_PATTERNS   :1      ;Set if more than one bg hatch
    VTCAF_MULTIPLE_STYLES        :1      ;Set if more than one (ssheet)
    style
                                         :4
VisTextCharAttrFlags  end

```

Library: **Objects/vTextC.def**



■ VisTextClearAllTabsParams

```
VisTextClearAllTabsParams    struct
    VTCATP_range    VisTextRange
VisTextClearAllTabsParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextClearTabParams

```
VisTextClearTabParams    struct
    VTCTP_range    VisTextRange
    VTCTP_position    word           ; In units of points * 8
VisTextClearTabParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextContextType

```
VisTextContextType    etype byte
    VTCT_TEXT          enum VisTextContextType
    VTCT_CATEGORY       enum VisTextContextType
    VTCT_QUESTION       enum VisTextContextType
    VTCT_ANSWER         enum VisTextContextType
    VTCT_DEFINITION     enum VisTextContextType
    VTCT_FILE           enum VisTextContextType, 255
```

Library: **Objects/vTextC.def**

■ VisTextCursorPositionChange

```
VisTextCursorPositionChange    struct
    VTCPC_lineNumber    dword
    VTCPC_rowNumber      dword
VisTextCursorPositionChange    ends
```

Library: **Objects/vTextC.def**

■ **VisTextCustomFilterData**

```
VisTextCustomFilterData  struct
    VTCFD_startOfRange    word
    VTCFD_endOfRange      word
VisTextCustomFilterData  ends
```

Library: **Objects/vTextC.def**

■ **VisTextDefaultCharAttr**

```
VisTextDefaultCharAttr  record
    VTDCA_UNDERLINE  :1
    VTDCA_BOLD       :1
    VTDCA_ITALIC     :1
                    :1
    VTDCA_COLOR       Color:4
    VTDCA_SIZE        VisTextDefaultSize:3
    VTDCA_FONT        VisTextDefaultFont:5
VisTextDefaultCharAttr  end
```

Library: **Objects/Text/tCommon.def**

■ **VisTextDefaultDefaultTab**

```
VisTextDefaultDefaultTab  etype byte
    VTDDT_NONE              enum VisTextDefaultDefaultTab
    VTDDT_HALF_INCH        enum VisTextDefaultDefaultTab
    VTDDT_INCH              enum VisTextDefaultDefaultTab
    VTDDT_CENTIMETER        enum VisTextDefaultDefaultTab
```

Library: **Objects/Text/tCommon.def**



■ VisTextDefaultFont

```
VisTextDefaultFont  etype byte
    VTDF_BERKELEY          enum VisTextDefaultFont;Bitmap font
    VTDF_CHICAGO           enum VisTextDefaultFont;Bitmap font
    VTDF_BISON             enum VisTextDefaultFont;Bitmap font
    VTDF_WINDOWS           enum VisTextDefaultFont;Bitmap font
    VTDF_LED               enum VisTextDefaultFont;Bitmap font
    VTDF_ROMA              enum VisTextDefaultFont;Bitmap font
    VTDF_UNIVERSITY        enum VisTextDefaultFont;Bitmap font

    VTDF_URW_ROMAN         enum VisTextDefaultFont;Nimbus-Q font
    VTDF_URW_SANS          enum VisTextDefaultFont;Nimbus-Q font
    VTDF_URW_MONO          enum VisTextDefaultFont;Nimbus-Q font
    VTDF_URW_SYMBOLPS      enum VisTextDefaultFont;Nimbus-Q font
    VTDF_CENTURY_SCHOOLBOOK enum VisTextDefaultFont;Nimbus-Q font
```

Library: **Objects/Text/tCommon.def**

■ VisTextDefaultParaAttr

```
VisTextDefaultParaAttr  record
    VTDPA_JUSTIFICATION    Justification:2
    VTDPA_DEFAULT_TABS     VisTextDefaultDefaultTab:2
    VTDPA_LEFT_MARGIN      :4                ;In units of half inches
    VTDPA_PARA_MARGIN       :4                ;In units of half inches
    VTDPA_RIGHT_MARGIN     :4                ;In units of half inches -- 0 means
                                         ;VIS_TEXT_MAX_PARA_ATTR_SIZE

VisTextDefaultParaAttr  end
```

Library: **Objects/Text/tCommon.def**

■ VisTextDefaultSize

```
VisTextDefaultSize  etype byte
    VTDS_8             enum VisTextDefaultSize
    VTDS_9             enum VisTextDefaultSize
    VTDS_10            enum VisTextDefaultSize
    VTDS_12            enum VisTextDefaultSize
    VTDS_14            enum VisTextDefaultSize
    VTDS_18            enum VisTextDefaultSize
    VTDS_24            enum VisTextDefaultSize
    VTDS_36            enum VisTextDefaultSize
```

Library: **Objects/Text/tCommon.def**

■ **VisTextDropCapInfo**

```
VisTextDropCapInfo  record
    VTDCI_CHAR_COUNT      :4 = 1-1      ; # characters for drop cap charAttr
    VTDCI_LINE_COUNT      :4 = 3-1      ; # lines for drop cap
    VTDCI_POSITION        :4 = 1-1      ; 0 is full drop cap
                                ; lineCount-1 is full tall cap
                                :4
VisTextDropCapInfo  end
```

Library: **Objects/Text/tCommon.def**

■ **VisTextExtendedFilterType**

```
VisTextExtendedFilterType  etype byte
    VTEFT_REPLACE_PARAMS    enum VisTextExtendedFilterType
    VTEFT_CHARACTER_LEVELER_LEVEL  enum VisTextExtendedFilterType
    VTEFT_BEFORE_AFTER      enum VisTextExtendedFilterType

    VTEFT_REPLACE_PARAMS
        This causes MSG_VIS_TEXT_FILTER_VIA_REPLACE_PARAMS
        to be sent.

    VTEFT_CHARACTER_LEVELER_LEVEL
        This causes MSG_VIS_TEXT_FILTER_VIA_CHARACTER to be
        sent.

    VTEFT_BEFORE_AFTER
        This causes MSG_VIS_TEXT_FILTER_VIA_BEFORE_AFTER to
        be sent.
```

Library: **Objects/vTextC.def**

■ **VisTextExtendedStyles**

```
VisTextExtendedStyles  record
    VTES_BOXED              :1
    VTES_BUTTON              :1
    VTES_INDEX              :1      ; text should be indexed
    VTES_ALL_CAP             :1
    VTES_SMALL_CAP          :1
    VTES_HIDDEN              :1
    VTES_CHANGE_BAR         :1
    VTES_BACKGROUND_COLOR   :1
                                :8
VisTextExtendedStyles  end
```

Library: **Objects/Text/tCommon.def**



■ VisTextFeatures

```

VisTextFeatures      record
    VTF_NO_WORD_WRAPPING      :1      ; Set: no word-wrapping is desired.
    VTF_AUTO_HYPHENATE        :1      ; Set: if we want to auto hyphenate.
    VTF_ALLOW_SMART_QUOTES     :1      ; Set: allows smart quotes if they
                                      ; are enabled.
    VTF_ALLOW_UNDO            :1      ; Set: allows undo in this object
    VTF_SHOW_HIDDEN_TEXT      :1      ; Set: Show text marked as hidden
                                      ; *** Not implemented ***
    VTF_OUTLINE_MODE          :1      ; Set: show text in outline mode
                                      ; *** Not implemented ***
    VTF_DONT_SHOW_SOFT_PAGE_BREAKS:1      ; Set: don't display soft (non
                                      ; C_PAGE_BREAK) page breaks
                                      ; *** Not implemented ***
    VTF_DONT_SHOW_GRAPHICS    :1      ; Draw graphics as gray rectangles
                                      ; *** Not implemented ***
    VTF_TRANSPARENT           :1      ; Set: don't use wash color on DRAW
    VTF_USE_50_PCT_TEXT_MASK  :1      ; Set: force 50% draw mask for
                                      ; drawing, ignoring char attr runs.
                                      ; Used by specific GenText objects.
                                      :6
VisTextFeatures      end

```

Library: **Objects/vTextC.def**

■ VisTextFilters

```

VisTextFilters      record
    VTF_NO_SPACES             :1      ;no spaces allowed
    VTF_NO_TABS               :1      ;no tabs
    VTF_UPCASE_CHARS          :1      ;make uppercase
    VTF_FILTER_CLASS          VisTextFilterClass:5;filter classes
                                      ;(keep in low bits!)
VisTextFilters      end

```

Library: **Objects/vTextC.def**

■ VisTextFilterClass

```
VisTextFilterClass  etype byte
    VTFC_NO_FILTER          enum VisTextFilterClass;no filter
    VTFC_ALPHA              enum VisTextFilterClass;alpha chars only
    VTFC_NUMERIC            enum VisTextFilterClass;numeric only
    VTFC_SIGNED_NUMERIC     enum VisTextFilterClass;signed numeric
    VTFC_SIGNED_DECIMAL     enum VisTextFilterClass;numeric, with decimal
    VTFC_FLOAT_DECIMAL     enum VisTextFilterClass;numeric,decimal,e,E
    VTFC_ALPHA_NUMERIC     enum VisTextFilterClass;alphanumeric
    VTFC_FILENAMES         enum VisTextFilterClass;legal PCGEOS filenames
    VTFC_DOS_FILENAMES     enum VisTextFilterClass;legal DOS filenames
    VTFC_DOS_PATH          enum VisTextFilterClass;legal DOS path
    VTFC_DATE              enum VisTextFilterClass;legal date
    VTFC_TIME              enum VisTextFilterClass;legal time
    VTFC_DASHED_ALPHA_NUMERIC enum VisTextFilterClass;alphanumeric plus '-'
    VTFC_NORMAL_ASCII      enum VisTextFilterClass;normal ascii chars
    VTFC_DOS_VOLUME_NAMES  enum VisTextFilterClass;legal DOS volume names
    VTFC_DOS_CHARACTER_SET enum VisTextFilterClass;DOS character set
    VTFC_ALLOW_COLUMN_BREAKS enum VisTextFilterClass;Allow column-breaks
```

Library: **Objects/vTextC.def**

■ VisTextFindNameParams

```
VisTextFindNameParams  struct
    VTFNP_name          fptr.char          ; pointer to name to find
    VTFNP_size          word                ; length of name (0 if null-terminated)
    VTFNP_data          fptr.VisTextNameData; buffer for data (0 if none)
VisTextFindNameParams  ends
```

This structure is passed with MSG_VIS_TEXT_FIND_NAME.

Library: **Objects/vTextC.def**

■ VisTextFollowHyperlinkParams

```
VisTextFollowHyperlinkParams  struct
    VTFHLP_range        VisTextRange        ; range of characters in the selection
VisTextFollowHyperlinkParams  ends
```

Library: **Objects/vTextC.def**



■ **VisTextGenerateNotifyParams**

```
VisTextGenerateNotifyParams  struct
    VTGNP_notificationTypes  VisTextNotificationFlags
    VTGNP_sendFlags          VisTextNotifySendFlags
    VTGNP_notificationBlocks  hptr 16 dup (?)
VisTextGenerateNotifyParams  ends
```

Library: **Objects/vTextC.def**

■ **VisTextGetAttrFlags**

```
VisTextGetAttrFlags          record
    VTGAF_MERGE_WITH_PASSED  :1          ;If set then merge the attributes for
                                         ;this object with the passed attributes
                                         :15
VisTextGetAttrFlags          end
```

Library: **Objects/vTextC.def**

■ **VisTextGetAttrParams**

```
VisTextGetAttrParams          struct
    VTGAP_range               VisTextRange
    VTGAP_attr                fptr          ; attribute structure
    VTGAP_return              fptr          ; diff structure
    VTGAP_flags               VisTextGetAttrFlags
VisTextGetAttrParams          ends
```

Library: **Objects/vTextC.def**

■ **VisTextGetGraphicAtPositionParams**

```
VisTextGetGraphicAtPositionParams  struct
    VTGGAPP_position          dword
    VTGGAPP_retPtr            fptr.VisTextGraphic
VisTextGetGraphicAtPositionParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextGetLineInfoParameters

```
VisTextGetLineInfoParameters struct
    VTGLIP_buffer    fptr.LineInfo    ;pointer to buffer to store results
    VTGLIP_bsize     word             ; size of buffer
    VTGLIP_line      dword            ; specific line that we're interested in
VisTextGetLineInfoParameters ends
```

This structure is passed with MSG_VIS_TEXT_GET_LINE_INFO. That method fills in the buffer specified by *VTGLIP_buffer* with a **LineInfo** structure followed by a variable number of **FieldInfo** structures.

Library: **Objects/vTextC.def**

■ VisTextGetLineOffsetAndFlagsParameters

```
VisTextGetLineOffsetAndFlagsParameters struct
    VTGLOAFP_line    dword            ; line to get information about
    ;
    ; The following entries are filled in by the handler for
    ; MSG_VIS_TEXT_GET_LINE_OFFSET_AND_FLAGS.
    ;
    VTGLOAFP_offset   dword            ; offset to line start
    VTGLOAFP_flags    LineFlags        ; LineFlags
VisTextGetLineOffsetAndFlagsParameters ends
```

Library: **Objects/vTextC.def**

■ VisTextGetRunBoundsParams

```
VisTextGetRunBoundsParams struct
    VTGRBP_position  dword            ; Position to check for run around

    VTGRBP_type      word             ; Run offset

    VTGRBP_retVal    fptr.VisTextRange ; Ptr to VisTextRange to fill
    ; in with the bounds of the run
VisTextGetRunBoundsParams ends
```

Library: **Objects/vTextC.def**



■ VisTextGetTextRangeFlags

```
VisTextGetTextRangeFlags  record
    VTGTRF_ALLOCATE        :1
    VTGTRF_ALLOCATE_ALWAYS :1
    VTGTRF_RESIZE_DEST     :1
                           :5
VisTextGetTextRangeFlags  end
```

VTGTRF_ALLOCATE

If set, requests that the destination be allocated. Otherwise, use destination provided

VTGTRF_ALLOCATE_ALWAYS

If set, asks that destination be allocated even if there is no text to copy.

VTGTRF_RESIZE_DEST

If set, will resize the destination (if possible) so that it is just large enough to hold the text and no larger.

Library: **Objects/vTextC.def**

■ VisTextGetTextRangeParameters

```
VisTextGetTextRangeParameters  struct
    VTGTRP_range                VisTextRange                ; range to get
    VTGTRP_textReference        TextReference                ; Reference to the text
    VTGTRP_flags                VisTextGetTextRangeFlags
    align                       word
VisTextGetTextRangeParameters  ends
```

Library: **Objects/vTextC.def**

■ VisTextGraphic

```
VisTextGraphic      struct
  VTG_meta          RefElementHeader      ; basic element header
  VTG_vmChain       dword
  VTG_size          XYSizes var
  VTG_type          VisTextGraphicType
  VTG_flags         VisTextGraphicFlags
  VTG_reserved      byte 4 dup (?)
  VTG_data          VisTextGraphicData
VisTextGraphic      ends
```

This structure defines a text graphic element.

VTG_vmChain stores a dword value to pass to VMChain routines. If only the low word is zero, then the high word is a VM handle. If both words are non-zero, the dword specifies a DB item; the high word specifies the DB group, the low word specifies the DB item itself. If the high word is zero, then the low word is an **LMemChunk**. If both words are zero, then there is no data.

VTG_size stores the size of the graphic. If this value is zero, then the graphic's size is determined dynamically.

Library: **Objects/vTextC.def**

■ VisTextGraphicData

```
VisTextGraphicData  union
  VTGD_gstring       VisTextGraphicGString
  VTGD_variable      VisTextGraphicVariable
  VTGD_opaque        VisTextGraphicOpaque
VisTextGraphicData  end
```

Library: **Objects/vTextC.def**

■ VisTextGraphicFlags

```
VisTextGraphicFlags record
  VTGF_DRAW_FROM_BASELINE  :1      ;If set then draw from baseline else
                                ;draw from top
  VTGF_HANDLES_POINTER     :1      ;Graphic can deal with pointer messages
                                :14
VisTextGraphicFlags      end
```

Library: **Objects/vTextC.def**



■ VisTextGraphicGString

```
VisTextGraphicGString    struct
    VTGG_tmatrix    TransMatrix
    VTGG_drawOffset    XYOffset
VisTextGraphicGString    ends
```

Library: **Objects/vTextC.def**

■ VisTextGraphicType

```
VisTextGraphicType    etype byte
    VTGT_GSTRING    enum VisTextGraphicType
    VTGT_VARIABLE    enum VisTextGraphicType
```

Library: **Objects/vTextC.def**

■ VisTextGraphicVariable

```
VisTextGraphicVariable    struct
    VTGV_manufacturerID    ManufacturerID
    VTGV_type    VisTextVariableType
    VTGV_privateData    byte (VIS_TEXT_GRAPHIC_OPAQUE_SIZE-4) dup (?)
VisTextGraphicVariable    ends
```

Library: **Objects/vTextC.def**

■ VisTextHWRFlags

```
VisTextHWRFlags    record
    VTHWRF_NO_CONTEXT    :1
    VTHWRF_USE_PASSED_CONTEXT    :1
                                :14
VisTextHWRFlags    end
```

VTHWRF_NO_CONTEXT

This is sent when the ink is being quick-copied to the object, or in other cases where the user did not draw the ink on top of the object, and so the position of the object is not useful information for the recognizer.

Library:

■ VisTextHyphenationInfo

```
VisTextHyphenationInfo    record
    VTHI_HYPHEN_MAX_LINES      :4 = 3-1
    VTHI_HYPHEN_SHORTEST_WORD  :4 = 5-1
    VTHI_HYPHEN_SHORTEST_PREFIX :4 = 3-1
    VTHI_HYPHEN_SHORTEST_SUFFIX :4 = 3-1
VisTextHyphenationInfo    end
```

Library: **Objects/Text/tCommon.def**

■ VisTextIntFlags

```
VisTextIntFlags    record
    VTIF_HAS_LINES      :1      ;Object has valid line structures.
    VTIF_SUSPENDED      :1      ;Set if calculation suspended
    VTIF_UPDATE_PENDING :1      ;Update is about to be delivered.
    VTIF_ACTIVE_SEARCH_SPELL ActiveSearchSpellType:2
                                ;Set if a search/spell session is in
                                ;progress.
    VTIF_HILITED        :1      ;Set: We have drawn the hilite.
    VTIF_ADJUST_TYPE    AdjustType:2
                                ; How to adjust the selection.
VisTextIntFlags    end
```

Library: **Objects/vTextC.def**

■ VisTextIntSelFlags

```
VisTextIntSelFlags    record
    VTISF_IS_TARGET      :1      ; Set if the object is the target.
    VTISF_IS_FOCUS      :1      ; Set if the object is the focus.
    VTISF_CURSOR_ON      :1      ; Set if the cursor is drawn.
    VTISF_CURSOR_ENABLED :1      ; Set if the cursor is enabled.
    VTISF_DOING_SELECTION :1      ; Set if we are doing some selection.
                                ; (Basically if the mouse is down).
    VTISF_DOING_DRAG_SELECTION:1      ; Set if we have positioned the cursor.
                                ; (also doubles as flag that indicates
                                ; we are doing quick-transfer feedback)
    VTISF_SELECTION_TYPE SelectionType:2
VisTextIntSelFlags    end
```

Library: **Objects/vTextC.def**



■ **VisTextKeepInfo**

```
VisTextKeepInfo    record
    VTKI_TOP_LINES      :4      ; # lines at start of PP to keep together
    VTKI_BOTTOM_LINES   :4      ; # lines at end of PP to keep together
VisTextKeepInfo    end
```

Library: **Objects/Text/tCommon.def**

■ VisTextKeyFunction

VisTextKeyFunction etype word, 0, 6

VTKF_FORWARD_LINE	enum	VisTextKeyFunction
VTKF_BACKWARD_LINE	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_FORWARD_LINE	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_BACKWARD_LINE	enum	VisTextKeyFunction
VTKF_FORWARD_CHAR	enum	VisTextKeyFunction
VTKF_BACKWARD_CHAR	enum	VisTextKeyFunction
VTKF_FORWARD_WORD	enum	VisTextKeyFunction
VTKF_BACKWARD_WORD	enum	VisTextKeyFunction
VTKF_FORWARD_PARAGRAPH	enum	VisTextKeyFunction
VTKF_BACKWARD_PARAGRAPH	enum	VisTextKeyFunction
VTKF_START_OF_LINE	enum	VisTextKeyFunction
VTKF_END_OF_LINE	enum	VisTextKeyFunction
VTKF_START_OF_TEXT	enum	VisTextKeyFunction
VTKF_END_OF_TEXT	enum	VisTextKeyFunction
VTKF_SELECT_WORD	enum	VisTextKeyFunction

```

;=====
; None of the following entries are supported:
;   VTKF_SELECT_LINE
;   VTKF_SELECT_PARAGRAPH
;   VTKF_SELECT_OBJECT
;

```

VTKF_SELECT_LINE	enum	VisTextKeyFunction
VTKF_SELECT_PARAGRAPH	enum	VisTextKeyFunction
VTKF_SELECT_OBJECT	enum	VisTextKeyFunction

```

;=====

```

VTKF_SELECT_ADJUST_FORWARD_CHAR	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_BACKWARD_CHAR	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_FORWARD_WORD	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_BACKWARD_WORD	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_FORWARD_PARAGRAPH	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_BACKWARD_PARAGRAPH	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_TO_START	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_TO_END	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_START_OF_LINE	enum	VisTextKeyFunction
VTKF_SELECT_ADJUST_END_OF_LINE	enum	VisTextKeyFunction
VTKF_DELETE_BACKWARD_CHAR	enum	VisTextKeyFunction
VTKF_DELETE_BACKWARD_WORD	enum	VisTextKeyFunction
VTKF_DELETE_BACKWARD_LINE	enum	VisTextKeyFunction
VTKF_DELETE_BACKWARD_PARAGRAPH	enum	VisTextKeyFunction
VTKF_DELETE_TO_START	enum	VisTextKeyFunction
VTKF_DELETE_CHAR	enum	VisTextKeyFunction
VTKF_DELETE_WORD	enum	VisTextKeyFunction
VTKF_DELETE_LINE	enum	VisTextKeyFunction
VTKF_DELETE_PARAGRAPH	enum	VisTextKeyFunction
VTKF_DELETE_TO_END	enum	VisTextKeyFunction
VTKF_DELETE_EVERYTHING	enum	VisTextKeyFunction



VTKF_DESELECT	enum VisTextKeyFunction
VTKF_TOGGLE_OVERSTRIKE_MODE	enum VisTextKeyFunction
VTKF_TOGGLE_SMART_QUOTES	enum VisTextKeyFunction

Library: **Objects/vTextC.def**

■ VisTextLoadFromDBWithStyleParams

```
VisTextLoadFromDBWithStyleParams struct
    VTLFDBWSP_params      fptr.StyleSheetParams
    VTLFDBWSP_dbItem      dword           ; DB item to load from
    VTLFDBWSP_file        hptr           ; file handle (or 0)
VisTextLoadFromDBWithStyleParams ends
```

Library: **Objects/vTextC.def**

■ VisTextMaxParaAttr

```
VisTextMaxParaAttr struct
    VTMPA_paraAttr  VisTextParaAttr
    VTMPA_tabs      Tab VIS_TEXT_MAX_TABS dup (<>)
VisTextMaxParaAttr ends
```

Library: **Objects/Text/tCommon.def**

■ VisTextMinimumDimensionsParameters

```
VisTextMinimumDimensionsParameters struc
    VTMDP_height  WBFixed
    VTMDP_width   WBFixed
VisTextMinimumDimensionsParameters ends
```

Library: **Objects/vTextC.def**

■ VisTextMoveTabParams

```
VisTextMoveTabParams      struct
    VTMTTP_range           VisTextRange
    VTMTTP_destPosition     word           ; in units of points * 8
    VTMTTP_sourcePosition   word           ; in units of points * 8
VisTextMoveTabParams      ends
```

Library: **Objects/vTextC.def**

■ VisTextNameArrayElement

```
VisTextNameArrayElement  struct
    VTNAE_meta      NameArrayElement
    VTNAE_data      VisTextNameData
VisTextNameArrayElement  ends
```

Library: **Objects/vTextC.def**

■ VisTextNameCommonParams

```
VisTextNameCommonParams  struct
    VTNCP_data      VisTextNameData
    VTNCP_index     word           ; index of name
    VTNCP_object    optr          ; optr of text or list object
VisTextNameCommonParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextNameData

```
VisTextNameData          struct
    VTND_type          VisTextNameType
    VTND_contextType   VisTextContextType
    VTND_file          word           ; token of file
    VTND_helpText      DBGroupAndItem ; help text
VisTextNameData          ends
```

Library: **Objects/vTextC.def**

■ VisTextNameType

```
VisTextNameType          etype byte
    VTNT_CONTEXT          enum VisTextNameType
    VTNT_FILE              enum VisTextNameType
```

Library: **Objects/vTextC.def**

■ VisTextNotificationFlags

```
VisTextNotificationFlags  record
    VTNF_SELECT_STATE      :1
    VTNF_CHAR_ATTR         :1
    VTNF_PARA_ATTR         :1
    VTNF_TYPE              :1
```



```

VTNF_SELECTION      :1
VTNF_COUNT          :1
VTNF_STYLE_SHEET    :1
VTNF_STYLE          :1
VTNF_SEARCH_ENABLE  :1
VTNF_SPELL_ENABLE   :1
VTNF_NAME           :1
VTNF_CURSOR_POSITION :1
                   :4
VisTextNotificationFlags end

```

Library: **Objects/vTextC.def**

■ VisTextNotifyCharAttrChange

```

VisTextNotifyCharAttrChange  struct
    VTNCAC_charAttr          VisTextCharAttr
    VTNCAC_charAttrToken     word
    VTNCAC_charAttrDiffs     VisTextCharAttrDiffs
VisTextNotifyCharAttrChange  ends

```

Library: **Objects/vTextC.def**

■ VisTextNotifyCountChange

```

VisTextNotifyCountChange  struct
    VTNCC_charCount         dword
    VTNCC_wordCount         dword
    VTNCC_lineCount         dword
    VTNCC_paraCount         dword
VisTextNotifyCountChange  ends

```

Library: **Objects/vTextC.def**

■ VisTextNotifyNameChange

```

VisTextNotifyNameChange  struct
    VTNNC_count             word
VisTextNotifyNameChange  ends

```

Library: **Objects/vTextC.def**

■ VisTextNotifyParaAttrChange

```
VisTextNotifyParaAttrChange  struct
    VTNPAC_paraAttr           VisTextMaxParaAttr
    VTNPAC_paraAttrToken      word
    VTNPAC_paraAttrDiffs      VisTextParaAttrDiffs
    VTNPAC_regionOffset       sdword
    VTNPAC_regionWidth        sword
    VTNPAC_selectedTab        word
VisTextNotifyParaAttrChange  ends
```

Library: **Objects/vTextC.def**

■ VisTextNotifySelectionChange

```
VisTextNotifySelectionChange  struct
    VTNSC_selectStart         dword
    VTNSC_selectEnd           dword
    VTNSC_lineNumber          dword
    VTNSC_lineStart           dword
    VTNSC_region              word
    VTNSC_regionStartLine     dword
    VTNSC_regionStartOffset   dword
VisTextNotifySelectionChange  ends
```

Library: **Objects/vTextC.def**

■ VisTextNotifySendFlags

```
VisTextNotifySendFlags      record
    VTNSF_UPDATE_APP_TARGET_GCN_LISTS  :1
    VTNSF_NULL_STATUS                  :1
    VTNSF_STRUCTURE_INITIALIZED         :1
    VTNSF_SEND_AFTER_GENERATION         :1
    VTNSF_SEND_ONLY                     :1
    VTNSF_RELAYED_TO_LIKE_TEXT_OBJECTS  :1
                                         :10
VisTextNotifySendFlags      end
```

VTNSF_UPDATE_APP_TARGET_GCN_LISTS

Set if pertinent Application Target GCN Lists should be updated with changes in status.

VTNSF_NULL_STATUS

Send notification of null status, for all notification types (used only to notify GCN Lists of loss of eligibility to update, i.e. lost



target). The text output will always be sent only meaningful info.

VTNSF_STRUCTURE_INITIALIZED
Set if the rest of the **VisTextGenerateNotifyParams** structure is initialized.

VTNSF_SEND_AFTER_GENERATION
Set to send the notifications after generating them.

VTNSF_SEND_ONLY
Set to send the notifications *only*.

VTNSF_RELAYED_TO_LIKE_TEXT_OBJECTS
Set if the message has been registered with the object responsible for relaying the message to multiple text objects.

Library: **Objects/vTextC.def**

■ VisTextNotifyTypeChange

```
VisTextNotifyTypeChange struct
    VTNTC_type      VisTextType
    VTNTC_typeToken word
    VTNTC_typeDiffs VisTextTypeDiffs
    VTNTC_index     VisTextType
VisTextNotifyTypeChange ends
```

Library: **Objects/vTextC.def**

■ VisTextNumberType

```
VisTextNumberType etype word
    VTNT_NUMBER          enum VisTextNumberType
    VTNT_LETTER_UPPER_A  enum VisTextNumberType
    VTNT_LETTER_LOWER_A  enum VisTextNumberType
    VTNT_ROMAN_NUMERAL_UPPER  enum VisTextNumberType
    VTNT_ROMAN_NUMERAL_LOWER  enum VisTextNumberType
```

Library: **Objects/Text/tCommon.def**

■ VisTextParaAttr

```
VisTextParaAttr struct
    VTPA_meta      StyleSheetElementHeader
    VTPA_borderFlags VisTextParaBorderFlags ; border type
    VTPA_borderColor ColorQuad ; color for borders
    VTPA_attributes VisTextParaAttrAttributes ; other attributes
    VTPA_leftMargin word
```

Reference book

```

VTPA_rightMargin      word
VTPA_paraMargin      word
VTPA_lineSpacing      BBFixed <>           ; 1.0 is default
VTPA_leading          word                 ; 13.3 is default
VTPA_spaceOnTop       word                 ; 0.0 is default
VTPA_spaceOnBottom   word                 ; 0.0 is default
VTPA_bgColor          ColorQuad
VTPA_numberOfTabs     byte
VTPA_borderWidth      byte                 ; in points * 8
VTPA_borderSpacing    byte                 ; in points * 8
VTPA_borderShadow     byte                 ; in points * 8
VTPA_borderGrayScreen SystemDrawMask
VTPA_bgGrayScreen     SystemDrawMask
VTPA_borderPattern    GraphicPattern
VTPA_defaultTabs      word                 ; spacing for default tabs
VTPA_startingParaNumber word
VTPA_prependChars     char 4 dup (0)
VTPA_hyphenationInfo  VisTextHyphenationInfo
VTPA_keepInfo         VisTextKeepInfo
VTPA_dropCapInfo      VisTextDropCapInfo
VTPA_nextStyle        word
VTPA_language         StandardLanguage
VTPA_reserved         byte 15 dup (0)
VTPA_tabList          label byte
VisTextParaAttr      ends

```

Library: **Objects/Text/tCommon.def**

■ VisTextParaAttrAttributes

```

VisTextParaAttrAttributes  record
  VTPAA_JUSTIFICATION      Justification:2
  VTPAA_KEEP_PARA_WITH_NEXT :1
  VTPAA_KEEP_PARA_TOGETHER :1           ;Don't break up paragraph
  VTPAA_ALLOW_AUTO_HYPHENATION :1       ;Use VisTextHyphenationInfo
  VTPAA_DISABLE_WORD_WRAP  :1
  VTPAA_COLUMN_BREAK_BEFORE :1
  VTPAA_PARA_NUMBER_TYPE   VisTextNumberType:3
  VTPAA_DROP_CAP           :1           ;Use VisTextDropCapInfo
  VTPAA_KEEP_LINES         :1           ;Use VisTextKeepInfo
  :4
VisTextParaAttrAttributes  end

```

Library: **Objects/Text/tCommon.def**



■ VisTextParaAttrBorderFlags

```
VisTextParaAttrBorderFlags    record
    VTPABF_MULTIPLE_BORDER_LEFT      :1      ;Match with VTPBF_LEFT
    VTPABF_MULTIPLE_BORDER_TOP       :1      ;Match with VTPBF_TOP
    VTPABF_MULTIPLE_BORDER_RIGHT     :1      ;Match with VTPBF_RIGHT
    VTPABF_MULTIPLE_BORDER_BOTTOM    :1      ;Match with VTPBF_BOTTOM
    VTPABF_MULTIPLE_BORDER_DOUBLES   :1      ;Match with VTPBF_DOUBLE
    VTPABF_MULTIPLE_BORDER_DRAW_INNERS :1      ;Match with VTPBF_DRAW_INNER
    VTPABF_MULTIPLE_BORDER_ANCHORS   :1
    VTPABF_MULTIPLE_BORDER_WIDTHS    :1
    VTPABF_MULTIPLE_BORDER_SPACINGS  :1
    VTPABF_MULTIPLE_BORDER_SHADOWS   :1
    VTPABF_MULTIPLE_BORDER_COLORS    :1
    VTPABF_MULTIPLE_BORDER_GRAY_SCREEN :1
    VTPABF_MULTIPLE_BORDER_PATTERNS  :1
                                     :3
VisTextParaAttrBorderFlags    end
```

Library: **Objects/vTextC.def**

■ VisTextParaAttrDiffs

```
VisTextParaAttrDiffs          struct
    VTPAD_diffs                VisTextParaAttrFlags
    VTPAD_diffs2               VisTextParaAttrFlags2
    VTPAD_borderDiffs          VisTextParaAttrBorderFlags
    VTPAD_attributes            VisTextParaAttrAttributes
    VTPAD_hyphenationInfo      VisTextHyphenationInfo
    VTPAD_keepInfo             VisTextKeepInfo
    VTPAD_dropCapInfo          VisTextDropCapInfo
    even
VisTextParaAttrDiffs          ends
```

Library: **Objects/vTextC.def**

■ VisTextParaAttrFlags

```
VisTextParaAttrFlags          record
    VTPAF_MULTIPLE_LEFT_MARGINS      :1
    VTPAF_MULTIPLE_RIGHT_MARGINS     :1
    VTPAF_MULTIPLE_PARA_MARGINS      :1
    VTPAF_MULTIPLE_LINE_SPACINGS     :1
    VTPAF_MULTIPLE_DEFAULT_TABS      :1
    VTPAF_MULTIPLE_TOP_SPACING       :1
    VTPAF_MULTIPLE_BOTTOM_SPACING    :1
    VTPAF_MULTIPLE_LEADINGS          :1
```

```

VTPAF_MULTIPLE_BG_COLORS          :1
VTPAF_MULTIPLE_BG_GRAY_SCREEN    :1
VTPAF_MULTIPLE_BG_PATTERNS       :1
VTPAF_MULTIPLE_TAB_LISTS         :1
VTPAF_MULTIPLE_STYLES            :1
VTPAF_MULTIPLE_PREPEND_CHARS     :1
VTPAF_MULTIPLE_STARTING_PARA_NUMBERS :1
VTPAF_MULTIPLE_NEXT_STYLES       :1
VisTextParaAttrFlags             end

```

Library: **Objects/vTextC.def**

■ VisTextParaAttrFlags2

```

VisTextParaAttrFlags2             record
    VTPAF2_MULTIPLE_LANGUAGES     :1
                                   :15
VisTextParaAttrFlags2             end

```

Library: **Objects/vTextC.def**

■ VisTextParaBorderFlags

```

VisTextParaBorderFlags            record
    VTPBF_LEFT                    :1          ;Set if a border on the left
    VTPBF_TOP                     :1          ;Set if a border on the top
    VTPBF_RIGHT                   :1          ;Set if a border on the right
    VTPBF_BOTTOM                  :1          ;Set if a border on the bottom
    VTPBF_DOUBLE                  :1          ;Draw two line border
    VTPBF_DRAW_INNER_LINES        :1          ;Draw lines between bordered paragraphs
    VTPBF_SHADOW                  :1          ;Set to use shadow
                                   :7
    VTPBF_ANCHOR                  ShadowAnchor:2
VisTextParaBorderFlags            end

```

Library: **tCommon.def**

■ VisTextRange

```

VisTextRange                      struct
    VTR_start                     dword      ; start of range
    VTR_end                       dword      ; end of range
VisTextRange                      ends

```

Library: **Objects/Text/tCommon.def**



■ VisTextRangeContext

```
VisTextRangeContext    record
    VTRC_PARAGRAPH_CHANGE    :1    ;Change done on paragraph level.
    VTRC_CHAR_ATTR_CHANGE    :1    ; Used for a charAttr change (include
                                   ; last CR, don't include next CR).
    VTRC_PARA_ATTR_BORDER_CHANGE :1    ;Used for a paraAttr change.
                                   ;including a border.
                                   :13
VisTextRangeContext    end
```

Library: **Objects/vTextC.def**

■ VisTextReplaceFlags

```
VisTextReplaceFlags    record
    VTRF_FILTER          :1    ;Set to filter replacement
    VTRF_KEYBOARD_INPUT  :1    ;Set if data is coming from the
                                   ; keyboard input
    VTRF_USER_MODIFICATION :1    ;Set if replace is due to a user
                                   ;action
    VTRF_UNDO            :1    ;Set if replace is due to an undo
    VTRF_DO_NOT_SEND_CONTEXT_UPDATE :1    ;Set if this is part of a
                                   ; multi-part replace, and so the
                                   ; text object should not send a
                                   ; context update (used internally
                                   ; to the text object only)
                                   :11
VisTextReplaceFlags    end
```

Library: **Objects/vTextC.def**

■ VisTextReplaceParameters

```
VisTextReplaceParameters struct
    VTRP_range            VisTextRange
    VTRP_insCount          dword    ; number of characters to
                                   ; insert
    VTRP_textReference     TextReference ; reference to text to insert
    VTRP_flags             VisTextReplaceFlags
    align                  word
VisTextReplaceParameters ends
```

This structure is passed with MSG_VIS_TEXT_REPLACE.

Library: **Objects/vTextC.def**

■ VisTextReplaceWithHWRParams

```
VisTextReplaceWithHWRParams    struct
    VTRWHWRP_range             VisTextRange
    VTRWHWRP_flags             VisTextHWRFlags
    VTRWHWRP_ink               hpPtr.InkHeader
    VTRWHWRP_context           HWRContext
VisTextReplaceWithHWRParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextSaveDBFlags

```
VisTextSaveDBFlags    record
    VTSDBF_TEXT                :1            ;set if text is saved
                                      ; (0 means null text)
    VTSDBF_CHAR_ATTR           VisTextSaveType:2
    VTSDBF_PARA_ATTR           VisTextSaveType:2
    VTSDBF_TYPE                VisTextSaveType:2
    VTSDBF_GRAPHIC             VisTextSaveType:2
    VTSDBF_STYLE               :1
    VTSDBF_REGION              :1            ;not currently implemented
    VTSDBF_NAME                :1
                                      :4
VisTextSaveDBFlags    end
```

Library: **Objects/vTextC.def**

■ VisTextSaveStyleSheetParams

```
VisTextSaveStyleSheetParams    struct
    VTSSSP_common              StyleSheetParams
    VTSSSP_graphicsElements    word          ;VM block of graphics elements
    VTSSSP_treeBlock           word
    VTSSSP_graphicTreeOffset   word          ;offset in treeBlock
VisTextSaveStyleSheetParams    ends
```

Library: **Objects/vTextC.def**



■ VisTextSaveToDBWithStylesParams

```
VisTextSaveToDBWithStylesParams    struct
    VTSTDBWSP_params                fptr.VisTextSaveStyleSheetParams
    VTSTDBWSP_dbItem                dword
    VTSTDBWSP_flags                 VisTextSaveDBFlags
    VTSTDBWSP_xferFile              word
VisTextSaveToDBWithStylesParams    ends

    VTSTDBWSP_dbItem stores the DB item to save the text to.
```

Library: **Objects/vTextC.def**

■ VisTextSaveType

```
VisTextSaveType    etype byte
    VTST_NONE                enum VisTextSaveType;nothing saved
    VTST_SINGLE_CHUNK        enum VisTextSaveType;single attr structure
    VTST_RUNS_ONLY           enum VisTextSaveType
    VTST_RUNS_AND_ELEMENTS   enum VisTextSaveType
```

Library: **Objects/vTextC.def**

■ VisTextSetBorderBitsParams

```
VisTextSetBorderBitsParams    struct
    VTSBBP_range              VisTextRange
    VTSBBP_bitsToSet          VisTextParaBorderFlags
    VTSBBP_bitsToClear        VisTextParaBorderFlags
VisTextSetBorderBitsParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextSetBorderWidthParams

```
VisTextSetBorderWidthParams    struct
    VTSBWP_range              VisTextRange
    VTSBWP_width              byte
    even
VisTextSetBorderWidthParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextSetCharAttrByDefaultParams

```
VisTextSetCharAttrByDefaultParams struct
    VTSCABDP_range      VisTextRange
    VTSCABDP_charAttr    VisTextDefaultCharAttr
VisTextSetCharAttrByDefaultParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetCharAttrByTokenParams

```
VisTextSetCharAttrByTokenParams struct
    VTSCABTP_range      VisTextRange
    VTSCABTP_charAttr    word
VisTextSetCharAttrByTokenParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetCharAttrParams

```
VisTextSetCharAttrParams struct
    VTSCAP_range      VisTextRange
    VTSCAP_charAttr    fptr.VisTextCharAttr
VisTextSetCharAttrParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetColorParams

```
VisTextSetColorParams struct
    VTSCP_range      VisTextRange
    VTSCP_color      ColorQuad
VisTextSetColorParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetContextFlags

```
VisTextSetContextFlags record
    :7
    VTCF_TOKEN :1 ;TRUE: context and hyperlink are tokens
VisTextSetContextFlags end
```

Library: **Objects/vTextC.def**



■ VisTextSetContextParams

```
VisTextSetContextParams struct
    VTSCXP_range    VisTextRange
    VTSCXP_context  word
    VTSCXP_flags    VisTextSetContextFlags
    even
VisTextSetContextParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetDefaultTabsParams

```
VisTextSetDefaultTabsParams struct
    VTSDTP_range    VisTextRange
    VTSDTP_defaultTabs word
VisTextSetDefaultTabsParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetDropCapPParams

```
VisTextSetDropCapPParams struct
    VTSDCP_range    VisTextRange
    VTSDCP_bitsToSet word
    VTSDCP_bitsToClear word
VisTextSetDropCapPParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetFontIDParams

```
VisTextSetFontIDParams struct
    VTSFIDP_range    VisTextRange
    VTSFIDP_fontID    FontID
VisTextSetFontIDParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetFontWeightParams

```
VisTextSetFontWeightParams    struct
    VTSFWP_range              VisTextRange
    VTSFWP_fontWeight         byte
    even
VisTextSetFontWeightParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextSetFontWidthParams

```
VisTextSetFontWidthParams    struct
    VTSFWIP_range             VisTextRange
    VTSFWIP_fontWidth         byte
    even
VisTextSetFontWidthParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextSetGrayScreenParams

```
VisTextSetGrayScreenParams    struct
    VTSGSP_range              VisTextRange
    VTSGSP_grayScreen         SystemDrawMask
    even
VisTextSetGrayScreenParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextSetHyperlinkParams

```
VisTextSetHyperlinkParams    struct
    VTSHLP_range              VisTextRange
    VTSHLP_context            word
    VTSHLP_file               word
    VTSHLP_flags              VisTextSetContextFlags
    even
VisTextSetHyperlinkParams    ends
```

Library: **Objects/vTextC.def**



■ VisTextSetHyphenationPParams

```
VisTextSetHyphenationPParams  struct
    VTSHP_range                VisTextRange
    VTSHP_bitsToSet            VisTextHyphenationInfo
    VTSHP_bitsToClear          VisTextHyphenationInfo
VisTextSetHyphenationPParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetKeepPParams

```
VisTextSetKeepPParams        struct
    VTSKP_range               VisTextRange
    VTSKP_bitsToSet           word
    VTSKP_bitsToClear         word
VisTextSetKeepPParams        ends
```

Library: **Objects/vTextC.def**

■ VisTextSetLargerPointSizeParams

```
VisTextSetLargerPointSizeParams  struct
    VTSLPSP_range             VisTextRange
    VTSLPSP_maximumSize       word
VisTextSetLargerPointSizeParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetLeadingParams

```
VisTextSetLeadingParams  struct
    VTSLP_range          VisTextRange
    VTSLP_leading         word
VisTextSetLeadingParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetLineSpacingParams

```
VisTextSetLineSpacingParams  struct
    VTSLSP_range              VisTextRange
    VTSLSP_lineSpacing        BBFixed
VisTextSetLineSpacingParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetMarginParams

```
VisTextSetMarginParams  struct
    VTSMPL_range            VisTextRange
    VTSMPL_position         word
VisTextSetMarginParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetParaAttrAttributesParams

```
VisTextSetParaAttrAttributesParams struct
    VTSPAAP_range            VisTextRange
    VTSPAAP_bitsToSet        VisTextParaAttrAttributes
    VTSPAAP_bitsToClear      VisTextParaAttrAttributes
VisTextSetParaAttrAttributesParams ends
```

Library: **Objects/vTextC.def**

■ VisTextSetParaAttrByDefaultParams

```
VisTextSetParaAttrByDefaultParams struct
    VTSPABDP_range            VisTextRange
    VTSPABDP_paraAttr         VisTextDefaultParaAttr
VisTextSetParaAttrByDefaultParams ends
```

Library: **Objects/vTextC.def**



■ VisTextSetParaAttrByTokenParams

```
VisTextSetParaAttrByTokenParams  struct
    VTSPABTP_range                VisTextRange
    VTSPABTP_paraAttr             word
VisTextSetParaAttrByTokenParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetParaAttrParams

```
VisTextSetParaAttrParams  struct
    VTSPAP_range             VisTextRange
    VTSPAP_paraAttr          fptr.VisTextParaAttr
VisTextSetParaAttrParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetParagraphNumberParams

```
VisTextSetParagraphNumberParams  struct
    VTSPNP_range                VisTextRange
    VTSPNP_startingParaNumber    word
VisTextSetParagraphNumberParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetPatternParams

```
VisTextSetPatternParams  struct
    VTSHAP_range           VisTextRange
    VTSHAP_hatch           GraphicPattern
VisTextSetPatternParams  ends
```

Library: **Objects/vTextC.def**

■ VisTextSetPointSizeParams

```
VisTextSetPointSizeParams  struct
    VTSPSP_range            VisTextRange
    VTSPSP_pointSize        WWFixed
VisTextSetPointSizeParams  ends
```

Library: **Objects/vTextC.def**

■ **VisTextSetPrependCharsParams**

```
VisTextSetPrependCharsParams struct
    VTSPCP_range    VisTextRange
    VTSPCP_chars    char 4 dup (0)
VisTextSetPrependCharsParams ends
```

Library: **Objects/vTextC.def**

■ **VisTextSetSmallerPointSizeParams**

```
VisTextSetSmallerPointSizeParams struct
    VTSSPSP_range    VisTextRange
    VTSSPSP_minimumSize word
VisTextSetSmallerPointSizeParams ends
```

Library: **Objects/vTextC.def**

■ **VisTextSpaceOnTBParams**

```
VisTextSpaceOnTBParams struct
    VTSSOTBP_range    VisTextRange
    VTSSOTBP_spacing  BBFixed
VisTextSpaceOnTBParams ends
```

Library: **Objects/vTextC.def**

■ **VisTextSetTabParams**

```
VisTextSetTabParams struct
    VTSTP_range    VisTextRange
    VTSTP_tab      Tab
VisTextSetTabParams ends
```

Library: **Objects/vTextC.def**



■ VisTextSetTextStyleParams

```
VisTextSetTextStyleParams    struct
    VTSTSP_range              VisTextRange
    VTSTSP_styleBitsToSet     word
    VTSTSP_styleBitsToClear   word
    VTSTSP_extendedBitsToSet  word
    VTSTSP_extendedBitsToClearword
VisTextSetTextStyleParams    ends
```

Library: **Objects/vTextC.def**

■ VisTextSetTrackKerningParams

```
VisTextSetTrackKerningParams struct
    VTSTKP_range              VisTextRange
    VTSTKP_trackKerning       BBFixed
    even
VisTextSetTrackKerningParams ends
```

Library: **Objects/vTextC.def**

■ VisTextShowSelectionArgs

```
VisTextShowSelectionArgs    struct
    VTSSA_params              MakeRectVisibleParams
    VTSSA_flags               VisTextShowSelectionFlags
VisTextShowSelectionArgs    ends
```

Library: **Objects/vTextC.def**

■ VisTextShowSelectionFlags

```
VisTextShowSelectionFlags    record
    VTSSF_DRAGGING            :1
                                :15
VisTextShowSelectionFlags    end
```

Library: **Objects/vTextC.def**

■ **VisTextStates**

```

VisTextStates      record
    VTS_EDITABLE           :1 ; Set: text is editable.
    VTS_SELECTABLE        :1 ; Set: text is selectable.
    VTS_TARGETABLE        :1 ; Set: object is targetable.
    VTS_ONE_LINE           :1 ; Set: object is limited to one
                           ; line.
    VTS_SUBCLASS_VIRT_PHYS_TRANSLATION :1 ; Set: send virtual to physical
                           ; charAttr/paraAttr translation
                           ; messages to self (for subclass)
    VTS_OVERSTRIKE_MODE    :1 ; Set: Overstrike mode (not
                           ; insert mode)
    VTS_USER_MODIFIED      :1 ; Set: text has changed.
                           :1
VisTextStates      end

```

Library: **Objects/vTextC.def**

■ **VisTextStorageFlags**

```

VisTextStorageFlags      record
    VTSF_LARGE            :1
    VTSF_MULTIPLE_CHAR_ATTRS :1
    VTSF_MULTIPLE_PARA_ATTRS :1
    VTSF_TYPES            :1
    VTSF_GRAPHICS         :1
    VTSF_DEFAULT_CHAR_ATTR :1
    VTSF_DEFAULT_PARA_ATTR :1
    VTSF_STYLES           :1
VisTextStorageFlags      end

```

VTSF_LARGE

If set: this object uses the large storage format and the bits below are unused. If clear: this object uses the model storage format and the bits below are used.

VTSF_MULTIPLE_CHAR_ATTRS

If set: *VTI_charAttrRuns* = check handle of charAttr runs. If not set:

if (*VTTF_defaultCharAttr*)

VTI_charAttrRuns is a **VisTextDefaultCharAttrs**.

else

VTI_charAttrRuns = chunk handle of charAttr.



VTSF_MULTIPLE_PARA_ATTRS

If set: *VTI_paraAttrRuns* = check handle of paraAttr runs. If not set:

if (*VTI_paraAttrRuns* != 0)
VTI_paraAttrRuns = chunk handle of paraAttr.
else
use default paraAttr.

Library: **Objects/Text/tCommon.def**

■ **VisTextSubstAttrTokenParams**

```
VisTextSubstAttrTokenParams  struct
    VTSATP_oldToken           word
    VTSATP_newToken           word
    VTSATP_runOffset          word
    VTSATP_updateRefFlag      word
    VTSATP_relayedToLikeTextObjects word
    VTSATP_recalcFlag         fptr.word
VisTextSubstAttrTokenParams  ends
```

Library: **Objects/vTextC.def**

■ **VisTextSuspendData**

```
VisTextSuspendData  struct
    VTSD_count                word
    VTSD_recalcRange          VisTextRange    ; range to recalculate
    VTSD_selectRange          VisTextRange    ; range to select
    VTSD_notifications        word
    VTSD_needsRecalc          BooleanByte
VisTextSuspendData  ends
```

Library: **Objects/vTextC.def**

 ■ **VisTextType**

```

VisTextType      struct
    VTT_meta      RefElementHeader
    VTT_hyperlinkName  word      ; name array element (-1 if none)
    VTT_hyperlinkFile   word      ; name array element (-1 if none)
    VTT_context        word      ; name array element (-1 if none)
    VTT_unused         byte 1 dup (0)
VisTextType      ends
  
```

Library: Objects/vTextC.def

 ■ **VisTextTypeDiffs**

```

VisTextTypeDiffs  record
    VTTD_MULTIPLE_HYPERLINKS  :1
    VTTD_MULTIPLE_CONTEXTS    :1
                                :14
VisTextTypeDiffs  end
  
```

Library: Objects/vTextC.def



■ VisTextVariableType

```

VisTextVariableType      etype word
VTVT_PAGE_NUMBER          enum VisTextVariableType
    ; private data: first word is VisTextNumberType
    ; vm chain: unused
VTVT_PAGE_NUMBER_IN_SECTION  enum VisTextVariableType
    ; private data: first word is VisTextNumberType
    ; vm chain: unused
VTVT_NUMBER_OF_PAGES        enum VisTextVariableType
    ; private data: first word is VisTextNumberType
    ; vm chain: unused
VTVT_NUMBER_OF_PAGES_IN_SECTION  enum VisTextVariableType
    ; private data: first word is VisTextNumberType
    ; vm chain: unused
VTVT_SECTION_NUMBER         enum VisTextVariableType
    ; private data: first word is VisTextNumberType
    ; vm chain: unused
VTVT_NUMBER_OF_SECTIONS     enum VisTextVariableType
    ; private data: first word is VisTextNumberType
    ; vm chain: unused
VTVT_CREATION_DATE_TIME     enum VisTextVariableType
    ; private data: first word is DateTimeFormat
    ; vm chain: unused
    ; available only for large text objects
VTVT_MODIFICATION_DATE_TIME  enum VisTextVariableType
    ; private data: first word is DateTimeFormat
    ; vm chain: unused
    ; available only for large text objects
VTVT_CURRENT_DATE_TIME      enum VisTextVariableType
    ; private data: first word is DateTimeFormat
    ; vm chain: unused
    ; available only for large text objects
VTVT_STORED_DATE_TIME        enum VisTextVariableType
    ; private data: first word is DateTimeFormat, 2nd is FileDate, 3d
    ; is FileTime
    ; vm chain: unused

```

Library: **geoworks.def**

■ VisTypeFlags

```

VisTypeFlags      record
    VTF_IS_COMPOSITE      :1
    VTF_IS_WINDOW         :1
    VTF_IS_PORTAL         :1
    VTF_IS_WIN_GROUP      :1

```

```

VTF_IS_CONTENT           :1
VTF_IS_INPUT_NODE       :1
VTF_IS_GEN               :1
VTF_CHILDREN_OUTSIDE_PORTAL_WIN :1
VisTypeFlags             end

```

VTF_IS_COMPOSITE

Set if object is a VisCompClass and therefore can have children (although, of course, a composite may at times have no children).

VTF_IS_WINDOW

Set if IS_COMPOSITE and creates a window with the window system in order to display itself and children in. If set, then the assumption is made that the window is the size of *VI_bounds* and therefore messages like MSG_VIS_DRAW and MSG_META_BUTTON that traverse all children skip children with this bit set. Also, the routine that returns the window handle that a visible object sits on will return this object's *VCI_window* if this bit is set. Note that this flag differs subtly from the VTF_IS_PORTAL flag described below.

VTF_IS_PORTAL

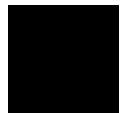
Set if object has its own window, which is stored elsewhere. Any visible children appear in that window. Object still may have portions which appear in its parents window. An example is the display control object, which manages several child windows inside its own window area. Its border is drawn in the parent window, and its own window is then inset one pixel from its bounds.

The flag has several effects:

- * it causes UPDATE_WINDOWS and CLOSE_WIN messages to be sent to the object.
- * the optimizations made for VTF_IS_WINDOW are not done. This object gets a MSG_VIS_DRAW and a MSG_META_BUTTON from its parent.
- * only one of the flags VTF_IS_WINDOW and VTF_IS_PORTAL can be set at a time.

VTF_IS_WIN_GROUP

Set for top visible object in a visible branch, which makes that branch a realizable entity. Visual updates happen on whole WIN_GROUP's. VTF_IS_WINDOW and VTF_IS_COMPOSITE must be set.



VTF_IS_CONTENT

Set if the object is basically the output descriptor of another window object. VTF_IS_WINDOW and VTF_IS_WIN_GROUP must also be set. Has a few subtle differences from a win group, one being that a Win is expected to be stuffed in by the “parent” object.

VTF_IS_INPUT_NODE

Set if this object controls input flow for either Kbd or Mouse, such as **VisContentClass**.

MSG_VIS_VUP_ALTER_INPUT_FLOW's are sent directly to objects having this bit set, unless there is a need for them actually to VUP up through each object (as is the case for mouse grabs in a 32-bit content model).

VTF_IS_GEN

Set if object has a Generic master part. This flag must be set for the object to handle “SpecClass” messages such as MSG_SPEC_BUILD. For optimization reasons, there is no SpecClass subclassed off of VisClass, but one can think of it that way.

VTF_CHILDREN_OUTSIDE_PORTAL_WIN

Only if VTF_IS_PORTAL is set, means that visible children lie in the portal's parent window areas, rather than in the window created by the portal object itself, thus they keep the portal's parent window in their instance data. An example of this is the pane, whose visual children lie around the outside of the pane's own created window. A display control, by contrast, has its visual children reside inside its window and thus would not have this flag set.

Library: **Objects/visC.def**

■ VisUpdateImageFlags

```
VisUpdateImageFlags      record
    VUIF_ALREADY_INVALIDATED :1
    VUIF_SEND_TO_ALL_CHILDREN :1
    VUIF_JUST_OPENED         :1
                                :5
VisUpdateImageFlags      end
```

VUIF_ALREADY_INVALIDATED

Set if we no longer need to invalidate things, until we hit a window at some point. If a VisComp object's image is invalidated, it will sometimes set this flag before broadcasting

the message to its children so they'll know not to invalidate themselves.

VUIF_SEND_TO_ALL_CHILDREN

Set if we need to send the invalidation message to all children, regardless of what the path bit is. This is for cases where a composite object is invalid, but only invalidates its margins to minimize invalidation, and then only children whose geometry is invalid will be invalidated further.

VUIF_JUST_OPENED

Internal flag.

Library: **Objects/visC.def**

■ **VisUpdateMode**

VisUpdateMode etype byte

VUM_MANUAL	enum VisUpdateMode	;don't update.
VUM_NOW	enum VisUpdateMode	;update NOW.
VUM_DELAYED_VIA_UI_QUEUE	enum VisUpdateMode	;delayed until UI queue ;empty
VUM_DELAYED_VIA_APP_QUEUE	enum VisUpdateMode	;delayed until APP queue ;empty

Library: **Objects/visC.def**



■ VisUpwardQueryType

```

VisUpwardQueryType  etype word
    SPEC_VIS_QUERY_START      equ 2000  ; offset to first specific UI query type
    APP_VIS_QUERY_START       equ 4000  ; offset to first app UI query type

    VUQ_DISPLAY_SCHEME        enum VisUpwardQueryType
    ; Context:                May be used when drawing part of a visible tree
    ; Source:                  Any VisClass object
    ; Destination:            Typically handled by the field object
    ; Interception:           It would be unusual to intercept this, as long as there is
    ;                          a single video mode per field.
    ;
    ; Pass:                    cx          - VUQ_DISPLAY_SCHEME
    ; Return:                  carry       - set if VUP message found routine to process
    ;                          request
    ;
    ;      ax, cx, dx, bp      - display scheme
    ;
    VUQ_VIDEO_DRIVER          enum VisUpwardQueryType
    ;
    ; Used to fetch the handle of the video driver which is in
    ; use for this location in the visible tree.
    ;
    ; Context:                Might be used when drawing part of a visible tree
    ; Source:                  Any VisClass object
    ; Destination:            Typically handled by the field object
    ; Interception:           It would be unusual to intercept this, as long as there is
    ;                          a single video mode per field.
    ;
    ; Pass:                    cx          - VUQ_VIDEO_DRIVER
    ; Return:                  carry       - set if VUP message found routine to process
    ;                          request
    ;      ax                  - handle of video driver
    ;      cx, dx, bp          - destroyed

```

Library: **Objects/visC.def**

■ VisWardMouseEventType

```

VisWardMouseEventType  etype byte, 0
    VWMET_SMALL          enum VisWardMouseEventType
    VWMET_LARGE          enum VisWardMouseEventType

```

Library: **grobj.def**

■ VisWardToolActiveStatus

```
VisWardToolActiveStatus  etype byte, 0
    VWTAS_ACTIVE          enum VisWardToolActiveStatus
    VWTAS_INACTIVE        enum VisWardToolActiveStatus
```

Library: **grobj.def**

■ VMAccessFlags

```
VMAccessFlags            record
    VMAF_FORCE_READ_ONLY          :1
    VMAF_FORCE_READ_WRITE         :1
    VMAF_ALLOW_SHARED_MEMORY      :1
    VMAF_FORCE_DENY_WRITE        :1
    VMAF_DISALLOW_SHARED_MULTIPLE :1
    VMAF_USE_BLOCK_LEVEL_SYNCHRONIZATION :1
    VMAF_FORCE_SHARED_MULTIPLE    :1
    <internal>                    :1
VMAccessFlags            end
```

VMAF_FORCE_READ_ONLY

If set then force the file to be opened read only, even if the default would be to open the file read/write.

VMAF_FORCE_READ_WRITE

If set then force the file to be opened read-write, even if the default would be to open the file read-only.

VMAF_ALLOW_SHARED_MEMORY

If set then use shared memory locally (unless otherwise impossible).

VMAF_FORCE_DENY_WRITE

If set then open file deny write.

VMAF_DISALLOW_SHARED_MULTIPLE

If set then files with the SHARED_MULTIPLE attribute cannot be opened.

VMAF_USE_BLOCK_LEVEL_SYNCHRONIZATION

If set then block the block level synchronization of the VM code is assumed to be sufficient and the {Start/End}Exclusive mechanism is not used. This is primarily intended for system software.

VMAF_FORCE_SHARED_MULTIPLE

If set, the file is opened as if it had the SHARED_MULTIPLE attribute even if it didn't. This is useful for data VM files that



need to always be opened as if SHARED_MULTIPLE were set, even when they're first created. Without this, there's a nasty race condition following the creation where the creator has to mark the file SHARED_MULTIPLE, close it, and reopen it again.

Library: **vm.def**

■ VMAttributes

```
VMAttributes      record
    VMA_SYNC_UPDATE      :1
    VMA_BACKUP            :1
    VMA_OBJECT_RELOC     :1
    VMA_NOTIFY_DIRTY     :1
    VMA_NO_DISCARD_IF_IN_USE :1
    VMA_COMPACT_OBJ_BLOCK :1
    VMA_SINGLE_THREAD_ACCESS :1
VMAttributes      end
```

VMA_SYNC_UPDATE

Allow synchronous updates only. Tells the system that it should not do asynchronous updates of the VM file. Clean VM blocks may always be discarded. Asynchronous updates are active by default

VMA_BACKUP

Maintain a backup copy of all data. The file can be returned to its backup state by calling **VMRevert**. The current state is made the backup by calling **VMSave**.

VMA_OBJECT_RELOC

Use the built-in object relocation routines

VMA_NOTIFY_DIRTY

Notify all processes that have the file open the first time a block becomes dirty after a **VMOpen**, **VMUpdate**, **VMSave**, or **VMRevert**.

VMA_NO_DISCARD_IF_IN_USE

Do not discard LMem blocks of type LMEM_TYPE_OBJ_BLOCK if *OLMBH_inUseCount* is non-zero

VMA_COMPACT_OBJ_BLOCK

If set, do a compaction when doing a unreloc before write (object blocks only)—allows generic objects in a VM file.

VMA_SINGLE_THREAD_ACCESS

If set then only a single thread will access the file, allowing optimizations in **VMLock**.

Library: **vm.def**

■ VMChainLink

```
VMChainLink      struct
  VMCL_next      word
VMChainLink      ends
```

Library: **vm.def**

■ VMChainTree

```
VMChainTree      struct
  VMCT_meta      VMChainLink
  VMCT_offset     nptr      ;offset to first chain
  VMCT_count      word      ;number of chains
VMChainTree      ends
```

Library: **vm.def**

■ VMLinkAndGrObjRelocation

```
VMLinkAndGrObjRelocation struct
  VMLAGOR_link      VMChainLink
  VMLAGOR_relocation GrObjEntryPointRelocation
VMLinkAndGrObjRelocation ends
```

Library: **grobj.def**

■ VMOpenType

```
VMOpenType      etype byte
  VMO_OPEN      enum VMOpenType; Open existing
  VMO_TEMP_FILE enum VMOpenType; Create temp file -- name is
                  ; directory
  VMO_CREATE     enum VMOpenType; Create or open existing
  VMO_CREATE_ONLY enum VMOpenType; Create, give error if already
                  ; exists
  VMO_CREATE_TRUNCATE enum VMOpenType ; Create, truncate any existing
                  ;file
```

Library: **vm.def**



■ VMOperation

VMOperation	etype	word
VMO_READ	enum	VMOperation ;default state -- allows ;readers not to modify the file
VMO_INTERNAL	enum	VMOperation
VMO_SAVE	enum	VMOperation
VMO_SAVE_AS	enum	VMOperation
VMO_REVERT	enum	VMOperation
VMO_UPDATE	enum	VMOperation
VMO_WRITE	enum	VMOperation ;for apps that don't want ;their own special codes
VMO_FIRST_APP_CODE	enum	VMOperation, 0x8000

Library: **vm.def**

■ VMRelocType

VMRelocType	etype	word
VMRT_UNRELOCATE_BEFORE_WRITE	enum	VMRelocType
VMRT_RELOCATE_AFTER_READ	enum	VMRelocType
VMRT_RELOCATE_AFTER_WRITE	enum	VMRelocType
VMRT_RELOCATE_FROM_RESOURCE	enum	VMRelocType
VMRT_UNRELOCATE_FROM_RESOURCE	enum	VMRelocType

Library: **vm.def**

■ VMStartExclusiveReturnValue

VMStartExclusiveReturnValue	etype	word
VMSErv_NO_CHANGES	enum	VMStartExclusiveReturnValue
VMSErv_CHANGES	enum	VMStartExclusiveReturnValue
VMSErv_TIMEOUT	enum	VMStartExclusiveReturnValue

Library: **vm.def**

■ **VMStatus**

```

VMStatus          etype word, 256
  VM_OPEN_OK_READ_ONLY          enum VMStatus
  VM_OPEN_OK_TEMPLATE           enum VMStatus
  VM_OPEN_OK_READ_WRITE_NOT_SHARED enum VMStatus
  VM_OPEN_OK_READ_WRITE_SINGLE  enum VMStatus
  VM_OPEN_OK_READ_WRITE_MULTIPLE enum VMStatus
  VM_OPEN_OK_BLOCK_LEVEL        enum VMStatus
  VM_CREATE_OK                  enum VMStatus

;          VM error codes

VM_FILE_EXISTS          enum VMStatus
VM_FILE_NOT_FOUND       enum VMStatus
VM_SHARING_DENIED       enum VMStatus
VM_OPEN_INVALID_VM_FILE enum VMStatus
VM_CANNOT_CREATE        enum VMStatus
VM_TRUNCATE_FAILED      enum VMStatus
VM_WRITE_PROTECTED      enum VMStatus
VM_CANNOT_OPEN_SHARED_MULTIPLE enum VMStatus
VM_FILE_FORMAT_MISMATCH enum VMStatus

;          VMUpdate status codes

VM_UPDATE_NOTHING_DIRTY          enum VMStatus
VM_UPDATE_INSUFFICIENT_DISK_SPACE enum VMStatus
VM_UPDATE_BLOCK_WAS_LOCKED       enum VMStatus

```

Library: **vm.def**

■ **VMStyle**

```

VMStyle  etype byte
  VMS_TEXT          enum VMStyle ; normal text moniker
  VMS_ABBREV_TEXT   enum VMStyle ; abbreviated text moniker i.e. a
                                ; short textual description rather
                                ; than the full title. Used for
                                ; name under icon of an iconified
                                ; primary.

  VMS_GRAPHIC_TEXT  enum VMStyle ; textual gstring
  VMS_ICON           enum VMStyle ; normal gstring moniker
  VMS_TOOL           enum VMStyle ; moniker for a tool, normally
                                ; smaller than a standard moniker

```

Library: **Objects/visC.def**



■ VupAlterInputFlowData

```
VupAlterInputFlowData    struct
    VAIFD_flags            VisInputFlowGrabFlags
    VAIFD_grabType         VisInputFlowGrabType
    VAIFD_object           optr
    VAIFD_gWin             hptr.Window
    VAIFD_translation      PointDWord
VupAlterInputFlowData    ends
```

VAIFD_gWin stores the window that the grabbing object is in (for mouse grabs only).

VAIFD_translation stores any additional 32-bit translation that should be applied to all mouse data (for mouse grabs only).

Library: **Objects/visC.def**

■ Warnings

```
Warnings                etype word, 0
```

Library: **ec.def**

■ WBFixed

```
WBFixed struct
    WBF_frac    byte    ;8 bits fraction
    WBF_int     word    ;16 bits integer
WBFixed ends
```

Library: **geos.def**

■ WidthJustification

```
WidthJustification etype byte
    WJ_LEFT_JUSTIFY_CHILDREN          enum WidthJustification
    WJ_RIGHT_JUSTIFY_CHILDREN         enum WidthJustification
    WJ_CENTER_CHILDREN_HORIZONTALLY  enum WidthJustification
    WJ_FULL_JUSTIFY_CHILDREN_HORIZONTALLY enum WidthJustification
```

Library: **Objects/vCompC.def**

■ WildCard

```
WildCard etype byte
    WC_MATCH_SINGLE_CHAR      enum WildCard, 0x10
    WC_MATCH_MULTIPLE_CHARS   enum WildCard, 0x11
    WC_MATCH_WHITESPACE_CHAR  enum WildCard, 0x12
```

Library: **Objects/vTextC.def**

■ WinColorFlags

```
WinColorFlags      record
    WCF_RGB          :1
    WCF_TRANSPARENT  :1
    WCF_PLAIN        :1
                    :2
    WCF_MAP_MODE     :3
WinColorFlags      end
```

WCF_RGB Set if using RGB colors, clear for indexed.

WCF_TRANSPARENT

Indicates window does not have a background color, & that owner must draw entire contents of window.

WCF_PLAIN Indicates window is one color only and therefore the window system may perform all draw operations for it. (No MSG_META_EXPOSED's are sent)

WCF_MAP_MODE

Graphics color mapping mode.

Library: **win.def**

■ WinConstrainType

```
WinConstrainType etype byte
    WCT_NONE              enum WinConstrainType
    WCT_KEEP_PARTIALLY_VISIBLE  enum WinConstrainType
    WCT_KEEP_VISIBLE      enum WinConstrainType
    WCT_KEEP_VISIBLE_WITH_MARGIN  enum WinConstrainType
```

WCT_NONE

Do not constrain window to parent. Allow complete clipping of window area by parent window.

WCT_KEEP_PARTIALLY_VISIBLE

Ensure that this window is at least partially visible within its parent at all times. In Motif, this means make sure the title bar is accessible.



WCT_KEEP_VISIBLE

Ensure that this window is completely visible within its parent at all times.

WCT_KEEP_VISIBLE_WITH_MARGIN

Library: **Objects/visC.def**

■ WinError

```
WinError          etype word, 0, 1
WE_COORD_OVERFLOW enum WinError    ; 16-bit coordinate overflow
WE_WINDOW_CLOSING enum WinError    ; window is closing
WE_GSTRING_PASSED enum WinError    ; gstring handle passed
```

Library: **win.def**

■ WinInfoType

```
WinInfoType       etype word, 0, 2
WIT_PRIVATE_DATA  enum WinInfoType
WIT_COLOR         enum WinInfoType
WIT_INPUT_OBJ     enum WinInfoType
WIT_EXPOSURE_OBJ enum WinInfoType
WIT_STRATEGY      enum WinInfoType
WIT_FLAGS         enum WinInfoType
WIT_LAYER_ID      enum WinInfoType
WIT_PARENT_WIN    enum WinInfoType
WIT_FIRST_CHILD_WIN enum WinInfoType
WIT_LAST_CHILD_WIN enum WinInfoType
WIT_PREV_SIBLING_WIN enum WinInfoType
WIT_NEXT_SIBLING_WIN enum WinInfoType
WIT_PRIORITY      enum WinInfoType
```

Library: **win.def**

■ WinInvalFlag

```
WinInvalFlag      etype byte, 0, 1
WIF_INVALIDATE    enum WinInvalFlag ; -invalidate the win
WIF_DONT_INVALIDATE enum WinInvalFlag ; -don't
```

Library: **win.def**

■ WinPassFlags

```
WinPassFlags      record
    WPF_CREATE_GSTATE      :1
    WPF_ROOT               :1
    WPF_SAVE_UNDER         :1
    WPF_INIT_EXCLUDED      :1
    WPF_PLACE_BEHIND       :1
    WPF_PLACE_LAYER_BEHIND :1
    WPF_LAYER              :1
    WPF_ABS                :1
    WPF_PRIORITY           WinPriorityData:8
WinPassFlags      end
```

The flags are listed below. Following the description of each flag, the list of routines which respect the flag are listed in parentheses.

WPF_CREATE_GSTATE

Set if a gstate should be created along with window (**WinOpen**).

WPF_ROOT Set if creating a root window (**WinOpen**).

WPF_SAVE_UNDER

Set if window should be created w/save under (**WinOpen**).

WPF_INIT_EXCLUDED

Init as being the head of a branch which is excluded from being an implied window, and therefore won't receive MSG_META_UNIV_ENTER, MSG_META_VIS_ENTER messages. (**WinOpen**).

WPF_PLACE_BEHIND

Indicates window should be placed behind other windows in its priority group. If clear, then window will be placed in front. (**WinOpen**, **WinChangePriority**).

WPF_PLACE_LAYER_BEHIND

Indicates whether layer should be placed behind other layers within its priority group. If clear, then layer will be placed in front. (**WinOpen**, **WinChangePriority**).

WPF_LAYER Set if operation applies to all windows having layerID (**WinChangePriority**).

WPF_ABS Whether size/offset passed is absolute or relative to current (**WinScroll**, **WinMove**, **WinResize**).

Library: **win.def**



■ WinPositionType

```
WinPositionType      etype byte
    WPT_AT_RATIO      enum WinPositionType
    WPT_STAGGER        enum WinPositionType
    WPT_CENTER         enum WinPositionType
    WPT_TILED          enum WinPositionType
    WPT_AT_MOUSE_POSITION enum WinPositionType
    WPT_AS_REQUIRED    enum WinPositionType
    WPT_AT_SPECIFIC_POSITION enum WinPositionType
```

WPT_AT_RATIO

Place this window at the specified position relative to the parent window. The position information is initially placed in the *R_left* and *R_top* fields of the *VI_bounds* for the object, as the object is initialized. During building, this information is converted from a ratio to actual coordinates.

WPT_STAGGER

Stagger this window down and to the right of previously staggered windows on this parent object.

WPT_CENTER

Center this window on the parent window.

WPT_TILED

Tile this window with its siblings.

WPT_AT_MOUSE_POSITION

Place the top-left corner of this window where the mouse pointer is. If the system has no mouse, the window is centered on the parent window.

WPT_AS_REQUIRED

Reserved for specific-UI use.

WPT_AT_SPECIFIC_POSITION

Reserved for specific-UI use.

Library: **Objects/visC.def**

■ WinPosSizeFlags

```
WinPosSizeFlags      record
    WPSF_PERSIST              :1
    WPSF_HINT_FOR_ICON        :1
    WPSF_NEVER_SAVE_STATE     :1
    WPSF_SHRINK_DESIRED_SIZE_TO_FIT_IN_PARENT :1
```

```

WPSF_CONSTRAIN_TYPE
WPSF_POSITION_TYPE
WPSF_SIZE_TYPE
WinPosSizeFlags      end

```

: 4
WinConstrainType: 2
WinPositionType: 3
WinSizeType: 3

WPSF_PERSIST
True for window to maintain its state (position, size, staggered slot #) when closed or detached. If false, the window will revert back to the specified position and size preferences (see below) when the window is re-opened. Note: could nuke this by adding HINT_DONT_PERSIST hint.

WPSF_HINT_FOR_ICON
True if this record is part of a hint for a GenPrimary or GenDisplay, and the hint is intended for the icon object. Note: could nuke this by creating separate hint.

WPSF_NEVER_SAVE_STATE
True for objects that never should have state saved when closed, such as menus. Overrides persist.

WPSF_SHRINK_DESIRED_SIZE_TO_FIT_IN_PARENT
Can be set true in objects where WPSF_SIZE_TYPE = WST_AS_DESIRED. After geometry has determined DESIRED size of window, if right side or bottom portion of window is not visible in parent window, this window will be resized to fit.

WPSF_CONSTRAIN_TYPE
Which constraint algorithm to use (keep inside, etc.).

WPSF_POSITION_TYPE
If window has not been moved/resized, this field indicates what position algorithm should be used.

WPSF_SIZE_TYPE
If false or window has not been moved/resized, this field indicates what sizing algorithm should be used.

Library: **Objects/visC.def**

■ WinPriority

```

WinPriority      etype byte
WIN_PRIO_POPUP      enum WinPriority, 4
WIN_PRIO_MODAL      enum WinPriority, 6
WIN_PRIO_COMMAND      enum WinPriority, 10
WIN_PRIO_STD      enum WinPriority, 12
WIN_PRIO_ON_BOTTOM      enum WinPriority, 14

```



WIN_PRIO_POPUP
Stay-up mode or drag mode, temporary popup menus.

WIN_PRIO_MODAL
For modal dialog boxes.

WIN_PRIO_ON_TOP
For misc which is supposed to appear “on top” of rest of application.

WIN_PRIO_COMMAND
For Command windows, non-modal dialogs, torn-off menus.

WIN_PRIO_STD
Standard window priority.

WIN_PRIO_ON_BOTTOM
Window stays on bottom.

Library: **win.def**

■ WinPriorityData

```
WinPriorityData    record
  WPD_LAYER      LayerPriority:4
  WPD_WIN        WinPriority:4      ; priority value for window.
WinPriorityData    end
```

Library: **win.def**

■ WinPtrFlags

```
WinPtrFlags        record
  WPF_PTR_IN_UNIV      :1      ; pointer in universe of window. (RAW)
                           ; this is not synchronous with the UI
                           ; thread
  WPF_PTR_IN_VIS       :1      ; pointer is in visible region of window
                           ; (RAW) this is NOT synchronous with the
                           ; UI thread
                           :6
WinPtrFlags        end
```

Library: **win.def**

■ WinRegFlags

```
WinRegFlags      record
    WRF_DELAYED_WASH      :1
    WRF_DELAYED_V         :1
    WRF_SIBLING_VALID     :1
    WRF_EXPOSE_PENDING    :1
    WRF_CLOSED            :1
    WRF_INVALID_TREE      :1
                                :2
WinRegFlags      end
```

WRF_DELAYED_WASH

Set if window has WRF_DELAYED_V set and the windowing system has delayed doing a was as a result. Will cause the fill to be done when the window block is V'd.

WRF_DELAYED_V

Set if window should not be V'd until validation operation is complete. Used to insure that no two V'd windows ever have overlapping *W_maskReg's* at any one instant.

WRF_SIBLING_VALID

Set if parent window's *W_childReg* contains running sum of regions of windows to the left of this one in the tree (*W_siblingReg*).

WRF_EXPOSE_PENDING

Means that a MSG_META_EXPOSED has been sent out, and neither **GrBeginUpdate** nor **WinUpdateAck** has been called yet.

WRF_CLOSED

Set if this window has been closed, but not yet freed.

WRF_INVALID_TREE

Set if this window is being invalidated from **WinInvalTree** and may need to redraw in its background color even if the entire window is already invalidated.

Library: **win.def**

■ WinSizeType

```
WinSizeType      etype byte
    WST_AS_RATIO_OF_PARENT      enum WinSizeType
    WST_AS_RATIO_OF_FIELD       enum WinSizeType
    WST_AS_DESIRED               enum WinSizeType
    WST_EXTEND_TO_BOTTOM_RIGHT  enum WinSizeType
    WST_EXTEND_NEAR_BOTTOM_RIGHT enum WinSizeType
```



WST_AS_RATIO_OF_PARENT

This can be used to open a window a specific size. The size information is initially placed in the *R_right* and *R_bottom* fields of the *VI_bounds* of the object, as the object is initialized. During building, this info is converted from a ratio to actually pixel-distance.

WST_AS_RATIO_OF_FIELD

WST_AS_DESIRED

Size the window according to its contents.

WST_EXTEND_TO_BOTTOM_RIGHT

This means size the window so that its bottom right corner is at the same position on the screen as the bottom right corner of the parent window.

WST_EXTEND_NEAR_BOTTOM_RIGHT

This means size the window so that its bottom right corner is a fixed margin away from the bottom right corner of the parent window. The margin is determined by the specific UI.

Library: **Objects/visC.def**

■ WordAndAHalf

```
WordAndAHalf      struct
    WAAH_low       word
    WAAH_high      byte
WordAndAHalf      ends
```

Library: **geos.def**

■ WWFixed

```
WWFixed struct
    WWF_frac       word           ;16 bits fraction
    WWF_int        word           ;16 bits integer
WWFixed ends
```

Library: **geos.def**

■ **XYOffset**

```
XYOffset      struct
  XYO_x      sword
  XYO_y      sword
XYOffset      ends
```

Library: **graphics.def**

■ **XYSize**

```
XYSize      struct
  XYS_width  word
  XYS_height word
XYSize      ends
```

Library: **graphics.def**



■ **Reference book**