

PGS.MD3

Copyright © 1995 Soft-Logik Publishing Corporation

COLLABORATORS

	<i>TITLE :</i> PGS.MD3		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	PGS.MD3	1
1.1	geteps	1
1.2	geterrornumber	2
1.3	geterrorstring	3
1.4	getfacingpagedisplay	3
1.5	getfile	3
1.6	getfilepath	4
1.7	getfontcache	5
1.8	getfontfamilies	5
1.9	getfontstyles	6
1.10	getgreeking	6
1.11	getgrid	7
1.12	getgriddisplay	7
1.13	getgridobject	8
1.14	getgridsnap	9
1.15	getgroup	10
1.16	getguides	11
1.17	getguide	11
1.18	getguidedisplay	12
1.19	getguidesnap	13
1.20	getinvisibledisplay	13
1.21	getline	13
1.22	getmarginguides	14
1.23	getmasterpagedesc	15
1.24	getmasterpages	15
1.25	getobject	16
1.26	getobjectlock	17
1.27	getoutlinedisplay	18
1.28	getpagedesc	18
1.29	getpagemasterpage	19

1.30	getpagename	20
1.31	getpagenumbering	20
1.32	getpath	21
1.33	getpicture	22
1.34	getpicturedisplay	23
1.35	getpolygon	24
1.36	getportname	25
1.37	getredo	25
1.38	getrefreshmode	25
1.39	getregion	26
1.40	getrotation	27
1.41	getrulerdisplay	27
1.42	getselectedobjects	28
1.43	gettextframe	29
1.44	gettextobj	29
1.45	gettextwrap	30
1.46	getscreendpi	31
1.47	getscreenname	31
1.48	getstring	32
1.49	gettextlinkdisplay	33
1.50	getundo	33
1.51	getwindowpos	33
1.52	getwindows	34
1.53	greeking	35
1.54	group	35
1.55	hidewindow	36

Chapter 1

PGS.MD3

1.1 geteps

GETEPS

External macros only!

Purpose: Gets coordinates and information for an EPS object.

Syntax: `geteps [POSITION pstem/V] [FRAME fflag/V]`
`[CONTENTOFFSET cstem/V] [CONTENTSCALE cstem/V] [ROTATION rstem/V]`
`[ABOUT rstem/V] [CONSTRAIN cflag/V] [PRINT pflag/V]`
`[FILEINFO fstem/V] [DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format: Parameter Values to enter

POSITION	gets the coordinates of the frame.
CONTENTOFFSET	gets the offset in the frame.
CONTENTSCALE	gets the scale of the object in the frame.
ROTATION	gets the rotation of the frame.
ABOUT	gets the rotation point.
CONSTRAIN	gets the proportional scale flag state.
PRINT	gets the print flag state.
FILEINFO	gets the file status of the EPS object.
DOCUMENT	is the document name.
WINDOW	is the window name.
OBJECTID	is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

POSITION:
`pstem.left` left coordinate
`pstem.top` top coordinate
`pstem.right` right coordinate
`pstem.bottom` bottom coordinate

FRAME: returns <ON|OFF>

CONTENTOFFSET:
`cstem.x` horizontal offset
`cstem.y` vertical offset

```

CONTENTSSCALE:
cstem.h      horizontal scale
cstem.v      vertical scale

ROTATION:
rstem.mode   rotate about <POINT|CENTER>
rstem.slant  slant angle
rstem.twist  twist angle

ABOUT:
rstem.x      horizontal point
rstem.y      vertical point

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

FILEINFO:
fstem.mode   <INTERNAL|EXTERNAL>
fstem.file   filepath and name

```

```

Example: geteps position coord /* will print the eps object bounding box to the ←
output console */
say 'Left: ' || coord.left
say 'Top: ' || coord.top
say 'Right: ' || coord.right
say 'Bottom: ' || coord.bottom

```

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.2 geterrornumber

GETERRORNUMBER

External macros only!

Purpose: Gets the number of the last error. Error numbers are different from the RC number, which is merely a measure of the error severity.

Syntax: geterrornumber (no parameters)

Result: The number is returned to RESULT.

```

Example: geterrornumber /* will return the error number and string */
errnum=result
geterrorstring
errname=result
say "Error #" || errnum || " occurred:"
say errname

```

See also

GETERRORSTRING

Command Format

1.3 geterrorstring

GETERRORSTRING

External macros only!

Purpose: Gets a short explanation of the last error. This is similar to the ARExx `errortext()` function, except that this command gets the last PageStream ARExx error.

Syntax: `geterrorstring` (no parameters)

Result: The message is returned to RESULT.

Example:

```
geterrornumber /* will return the error number and string */
errnum=result
geterrorstring
errname=result
say "Error #||errnum||" occurred:"
say errname
```

See also

GETERRORNUMBER
Command Format

1.4 getfacingpagedisplay

GETFACINGPAGEDISPLAY

External macros only!

Purpose: Gets the facing page display status.

Syntax: `getfacingpagedisplay` [WINDOW name/S]

Format: Parameter Values to enter
WINDOW is the window name. (Default=current)

Result: Returns the facing page display status <ON|OFF> to RESULT.

Example:

```
getfacingpagedisplay /* will print the facing page display status to ↔
the output console */
if result='ON' then say 'Facing Pages' else say 'Not Facing Pages'
```

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.5 getfile

GETFILE

External macros only!

Purpose: Opens the ASL file requester to get a filename and path from the user.

Syntax: `getfile <TITLE title/S> [LOAD | SAVE] [PATH filepath/F]
[FILE filename/F] [POSBUTTON label/S] [NEGBUTTON label/S]`

Format:

Parameter	Values to enter
TITLE	is the title of the file requester.
LOAD	opens the ASL load file requester. (Default)
SAVE	opens the ASL save file requester.
PATH	is the default file path. (Default=last)
FILE	is the default filename to open. (Default=last)
POSBUTTON	is the label for the left button. (Default=OK)
NEGBUTTON	is the label for the right button. (Default=Cancel)

Result: If NEGBUTTON is chosen, it sets RC to 10. If POSBUTTON is chosen, it sets RC to 0 and returns the full filepath and name to the RESULT variable.

Notes: The ASL file requester does not support keyboard equivalents for its buttons, so do not use underscores in the button labels.

Example: `'getfile TITLE "Save your work" save path ram: posbutton Save'
filename=RESULT
button=RC`

Command Format

1.6 getfilepath

GETFILEPATH

External macros only!

Purpose: Opens the ASL file requester to get a file path from the user.

Syntax: `getfilepath <TITLE title/S> [PATH filepath/F]
[POSBUTTON label/S] [NEGBUTTON label/S]`

Format:

Parameter	Values to enter
TITLE	is the title of the file requester.
PATH	is the default file path. (Default=last)
POSBUTTON	is the label for the left button. (Default=OK)
NEGBUTTON	is the label for the right button. (Default=Cancel)

Result: If NEGBUTTON is chosen, it sets RC to 10. If POSBUTTON is chosen, it sets RC to 0 and returns the path to the RESULT variable.

Notes: The ASL file requester does not support keyboard equivalents for its buttons, so do not use underscores in the button labels.

The actual ASL path requester is not used because it is a bad example of interface design. It does not list the files in the path at all, not even ghosted, so it is difficult for the user to know which path to select. The file requester is substituted and the selected file is ignored. This is not an oversight, but the result of a conscious decision to overcome a shortcoming in the ASL design.

```
Example: 'getfilepath title "Choose a path" path ram: posbutton Choose'
        path=RESULT
        button=RC
```

Command Format

1.7 getfontcache

GETFONTCACHE

External macros only!

Purpose: Gets the font cache size and limit.

Syntax: getfontcache <stem/V>

Format: Parameter Values to enter
 stem is a stem variable for the size and limit.

Result: stem.cachesize is the cache size
 stem.cachemaxh is the maximum height to cache

```
Example: 'getfontcache temp' /* will print the cache size to the output console ↔
        */
        say 'Cache size is ' || temp.cachesize
```

Command Format

1.8 getfontfamilies

GETFONTFAMILIES

External macros only!

Purpose: Gets the names of all the added font families.

Syntax: getfontfamilies <stem/V>

Format: Parameter Values to enter
 stem is the name of a stem variable for the family names.

Result: The number of font families is returned to RESULT.

The names of the font families are returned to stem.# where #

is a number from 0 to the number of font families less 1.

```
Example: getfontfamilies fontnames /* will print the font family names to the ↔
output console */
numfonts=result
do count=0 to numfonts-1
  say fontnames.count
end count
```

See also

```
getfontstyles
Command Format
```

1.9 getfontstyles

GETFONTSTYLES

External macros only!

Purpose: Gets the names of all the styles of an added font family.

Syntax: getfontstyles <family/S stem/V>

Format: Parameter Values to enter
 family is the name of the font family.
 stem is the name of a stem variable for the style names.

Result: The number of styles is returned to RESULT.

The names of the styles are returned to stem.# where # is a number from 0 to the number of styles less 1.

```
Example: getfontstyles Times stylenames /* will print the font style names to ↔
the output console */
numstyles=result
do count=0 to numstyles-1
  say stylenames.count
end count
```

See also

```
getfontfamilies
Command Format
```

1.10 getgreekking

GETGREEKING

External macros only!

Purpose: Gets the greeking status for a window.

Syntax: getgreekking [WINDOW name/S]

Format: Parameter Values to enter
 WINDOW is the window name. (Default=current)

Result: Returns the greeking status <ON|OFF> to RESULT.

Example: getgreeking
 say result

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.11 getgrid

GETGRID

External macros only!

Purpose: Gets the snap-to-grid settings for a master page.

Syntax: getgrid <stem/V> [MASTERPAGE name/S]

Format: Parameter Values to enter
 stem is the name of a stem variable for the information.
 MASTERPAGE is the master page name. (Default=current)

Result: stem.h horizontal grid spacing
 stem.v vertical grid spacing
 stem.x horizontal snap offset
 stem.y vertical snap offset
 stem.snap <ALL|RANGE>
 stem.rangeh horizontal range
 stem.rangev vertical range
 stem.displayh horizontal display interval
 stem.displayv vertical display interval
 stem.displayx horizontal display offset
 stem.displayy vertical display offset

Example: getgrid info

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.12 getgriddisplay

GETGRIDDISPLAY

External macros only!

Purpose: Gets the grid display status.

Syntax: getgriddisplay [DEPTH layer/V] [COLOR number/V]

[WINDOW name/S]

Format: Parameter Values to enter
 DEPTH gets the grid depth.
 COLOR gets the grid color number.
 WINDOW is the window name. (Default=current)

Result: Returns the grid display status <ON|OFF> to RESULT.

DEPTH: returns <INFRONT|INBACK>

Example: `getgriddisplay depth layer /* will print the grid status to the output ↔`
`console */`
`if result='ON' then say 'The grid is shown '||layer`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.13 getgridobject

GETGRIDOBJECT

External macros only!

Purpose: Gets coordinates and information for a grid object.

Syntax: `getgridobject [POSITION pstem/V] [POINTS cstem/V]`
`[DIVISIONS dstem] [ROTATION rstem/V] [ABOUT rstem/V]`
`[CONSTRAIN cflag/V] [PRINT pflag/V]`
`[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format: Parameter Values to enter
 POSITION gets the coordinates of the bounding box.
 POINTS gets the coordinates of the vertices of a non-rectangular grid (numbered counterclockwise)
 DIVISIONS gets the number of grid cells in each direction
 ROTATION gets the rotation of the bounding box.
 ABOUT gets the rotation point.
 CONSTRAIN gets the proportional scale flag state.
 PRINT gets the print flag state.
 DOCUMENT is the document name.
 WINDOW is the window name.
 OBJECTID is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

POSITION:
 pstem.left left coordinate
 pstem.top top coordinate
 pstem.right right coordinate
 pstem.bottom bottom coordinate

POINTS:
 cstem.x1 horizontal coordinate of point 1
 cstem.y1 vertical coordinate of point 1

```

cstem.x2      horizontal coordinate of point 2
cstem.y2      vertical coordinate of point 2
cstem.x3      horizontal coordinate of point 3
cstem.y3      vertical coordinate of point 3
cstem.x4      horizontal coordinate of point 4
cstem.y4      vertical coordinate of point 4

```

DIVISIONS:

```

dstem.h      number of horizontal divisions
dstem.v      number of vertical divisions

```

ROTATION:

```

rstem.mode   rotate about <POINT|CENTER>
rstem.slant  slant angle
rstem.twist  twist angle

```

ABOUT:

```

rstem.x      horizontal point
rstem.y      vertical point

```

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

```

Example: getgridobject position coord /* will print the grid object bounding box ←
        coordinates to the output console */
        say 'Left: ' || coord.left
        say 'Top: ' || coord.top
        say 'Right: ' || coord.right
        say 'Bottom: ' || coord.bottom

```

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.14 getgridsnap

GETGRIDSNAP

External macros only!

Purpose: Gets the grid snap status.

Syntax: getgridsnap [WINDOW name/S]

Format: Parameter Values to enter
 WINDOW is the window name. (Default=current)

Result: Returns the grid snap status <ON|OFF> to RESULT.

```

Example: getgridsnap /* will print the grid snap status to the output console */
        say 'Grid Snap: ' || result

```

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.15 getgroup

GETGROUP

External macros only!

Purpose: Gets coordinates and information for a group.

Syntax: `getgroup [POSITION pstem/V] [FRAME fflag/V]
[CONTENTOFFSET cstem/V] [CONTENTSCALE cstem/V] [ROTATION rstem/V]
[ABOUT rstem/V] [CONSTRAIN cflag/V] [PRINT pflag/V]
[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format:	Parameter	Values to enter
	POSITION	gets the coordinates of the frame.
	CONTENTOFFSET	gets the offset in the frame.
	CONTENTSCALE	gets the scale of the object in the frame.
	ROTATION	gets the rotation of the frame.
	ABOUT	gets the rotation point.
	CONSTRAIN	gets the proportional scale flag state.
	PRINT	gets the print flag state.
	DOCUMENT	is the document name.
	WINDOW	is the window name.
	OBJECTID	is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

POSITION:
 pstem.left left coordinate
 pstem.top top coordinate
 pstem.right right coordinate
 pstem.bottom bottom coordinate

FRAME: returns <ON|OFF>

CONTENTOFFSET:
 cstem.x horizontal offset
 cstem.y vertical offset

CONTENTSCALE:
 cstem.h horizontal scale
 cstem.v vertical scale

ROTATION:
 rstem.mode rotate about <POINT|CENTER>
 rstem.slant slant angle
 rstem.twist twist angle

ABOUT:
 rstem.x horizontal point
 rstem.y vertical point

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

Example: `getgroup position coord /* will print the group bounding box to the ←
output console */
say 'Left: ' || coord.left
say 'Top: ' || coord.top
say 'Right: ' || coord.right
say 'Bottom: ' || coord.bottom`

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.16 getguides

GETGUIDES

External macros only!

Purpose: Gets the horizontal or vertical ruler guides for a master page side.

Syntax: `getguides <stem/V> <VERTICAL|HORIZONTAL> [MPG name/S]`

Format: Parameter Values to enter
 stem is the name of a stem variable for the information.
 HORIZONTAL specifies horizontal guides
 VERTICAL specifies vertical guides
 MPG is the master page name. (Default=current)

Notes: The current side is the default if one is not specified.

Result: The number of guides is returned to RESULT.

The positions of the guides are returned to `stem.#` where # is a number from 0 to the number of guides less 1.

Example: `getguides info vertical
say 'There are ' || result || ' vertical guides for this master page.'`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.17 getguide

GETGUIDE

External macros only!

Purpose: Gets the snap-to-guide settings for a master page side.

Syntax: `getguide <stem/V> [MASTERPAGE name/S]`

Format: Parameter Values to enter
 stem is the name of a stem variable for the information.
 MASTERPAGE is the master page name. (Default=current)

Result: stem.snap ALL|RANGE
 stem.rangeh horizontal range
 stem.rangev vertical range

Example: `getguide info`
`if info.snap='RANGE' then do`
`say 'Snap horizontally within ' || info.rangeh`
`say 'Snap vertically within ' || info.rangev`
`end`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.18 getguidedisplay

GETGUIDEDISPLAY

External macros only!

Purpose: Gets the guide display status.

Syntax: `getguidedisplay [DEPTH layer/V] [COLOR number/V]`
`[PAGE flag/S] [RULER flag/S] [WINDOW name/S]`

Format: Parameter Values to enter
 DEPTH gets the guide depth.
 COLOR gets the guide color number.
 PAGE gets the page guide display status.
 RULER gets the ruler guide display status.
 WINDOW is the window name. (Default=current)

Result: Returns the guide display status <ON|OFF> to RESULT. If either page or ruler guides are displayed, ON will be returned.

DEPTH: returns <INFRONT|INBACK>

PAGE: returns <ON|OFF>

RULER: returns <ON|OFF>

Example: `getguidedisplay page pflag ruler rflag /* will print the guide status ←`
`to the output console */`
`if result=ON then do`
`say 'Page Guides: ' || pflag`
`say 'Ruler Guides: ' || rflag`
`end`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.19 getguidesnap

GETGUIDESNAP

External macros only!

Purpose: Gets the guide snap status.

Syntax: `getguidesnap [WINDOW name/S]`

Format: Parameter Values to enter
WINDOW is the window name. (Default=current)

Result: Returns the guide snap status <ON|OFF> to RESULT.

Example: `getguidesnap /* will print the grid snap status to the output console ←
*/
say 'Guide Snap: '||result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.20 getinvisibledisplay

GETINVISIBLEDISPLAY

External macros only!

Purpose: Gets the invisible symbol display status.

Syntax: `getinvisibledisplay [WINDOW name/S]`

Format: Parameter Values to enter
WINDOW is the window name. (Default=current)

Result: Returns the invisible display status <ON|OFF> to RESULT.

Example: `getinvisibledisplay /* will print the invisible display status to the ←
output console */
say 'Invisible Symbol Display: '||result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.21 getline

GETLINE

External macros only!

Purpose: Gets coordinates and information for a line.

Syntax: `getline [POSITION pstem/V] [ROTATION rstem/V]`
`[ABOUT rstem/V] [CONSTRAIN cflag/V] [PRINT pflag/V]`
`[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format: Parameter Values to enter

POSITION	gets the coordinates of the line.
ROTATION	gets the rotation of the line.
ABOUT	gets the rotation point.
CONSTRAIN	gets the proportional scale flag state.
PRINT	gets the print flag state.
DOCUMENT	is the document name.
WINDOW	is the window name.
OBJECTID	is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

POSITION:

<code>pstem.x1</code>	first endpoint horizontal coordinate
<code>pstem.y1</code>	first endpoint vertical coordinate
<code>pstem.x2</code>	second endpoint horizontal coordinate
<code>pstem.y2</code>	second endpoint vertical coordinate

ROTATION:

<code>rstem.mode</code>	rotate about <POINT CENTER>
<code>rstem.slant</code>	slant angle
<code>rstem.twist</code>	twist angle

ABOUT:

<code>rstem.x</code>	horizontal point
<code>rstem.y</code>	vertical point

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

Example: `getline position coord /* will print the line coordinates to the output ↵`
`console */`
`if coord.x2>coord.x1 then do`
`temp=coord.x1`
`coord.x1=coord.x2`
`coord.x2=temp`
`temp=coord.y1`
`coord.y1=coord.y2`
`coord.y2=temp`
`end`
`say 'Left Endpoint: '||coord.x1||', '||coord.y1`
`say 'Right Endpoint: '||coord.x2||', '||coord.y2`

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.22 getmarginguides

GETMARGINGUIDES

External macros only!

Purpose: Gets the margin guides for a master page.

Syntax: `getmarginguides <stem/V> [MASTERPAGE name/S]`

Format: Parameter Values to enter
 stem is the name of a stem variable for the information.
 MASTERPAGE is the master page name. (Default=current)

Result: stem.inside inside margin
 stem.outside outside margin
 stem.top top margin
 stem.bottom bottom margin

Example: `getmarginguides info /* will print the margin guides */`
`say 'Inside margin: ' || info.inside`
`say 'Outside margin: ' || info.outside`
`say 'Top margin: ' || info.top`
`say 'Bottom margin: ' || info.bottom`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.23 getmasterpagedesc

GETMASTERPAGEDESC

External macros only!

Purpose: Gets the description of a master page.

Syntax: `getmasterpagedesc [MASTERPAGE name/S]`

Format: Parameter Values to enter
 MASTERPAGE is the master page name. (Default=current)

Result: The description is returned to RESULT.

Example: `getmasterpagedesc /* will return the description of the current master` ←
`page */`
`mpagedesc=result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.24 getmasterpages

GETMASTERPAGES

External macros only!

Purpose: Gets the number of master pages in a document or chapter and their names.

Syntax: getmasterpages <stem/V> [DOCUMENT name/S | CHAPTER name/S]

Format:

Parameter	Values to enter
stem	is the name of a stem variable for the master page names.
DOCUMENT	is the document name. (Default=current)
CHAPTER	is the document/chapter name.

Result: The number of master pages in the document or chapter is returned to RESULT.

The names of the master pages are returned to stem.# where # is a number from 0 to the number of master pages less 1.

Example: getmasterpages mpagenames /* will print the master page names to the
output console */
numpages=result
do count=0 to numpages-1
say mpagenames.count
end count

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.25 getobject

GETOBJECT

External macros only!

Purpose: Gets the object ID, rotation status and the status of various flags for an object.

Syntax: getobject [TYPE type/V] [PAGENUMBER number/S]
[ROTATION rstem/V] [ABOUT rstem/V] [CONSTRAIN cflag/V]
[PRINT pflag/V] [LOCK lflag/V] [BOUNDINGBOX bstem/V]
[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]

Format:

Parameter	Values to enter
TYPE	gets the object type.
PAGENUMBER	gets the full page number path.
ROTATION	gets the object rotation.
ABOUT	gets the object rotation point.
CONSTRAIN	gets the object proportional scale flag state.
PRINT	gets the object print flag state.
LOCK	gets the object lock flag state.
BOUNDINGBOX	gets the bounding box of all selected objects.
DOCUMENT	is the document name.
WINDOW	is the window name.
OBJECTID	is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

TYPE: returns the object type number.

Drawing	2
Group	3
Compound	4
Box	5
Line	6
Ellipse	7
Grid	8
Polygon	9
Path	10
Text Frame	11
Picture	12
EPS	13
Frameless Text	14
Multiple Select	?**

ROTATION:

rstem.mode	Rotate about	<POINT CENTER>
rstem.slant	Slant angle	
rstem.twist	Twist angle	

ABOUT:

rstem.x	Horizontal point
rstem.y	Vertical point

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

LOCK: returns <ON|OFF>

BOUNDINGBOX:

bstem.left	left coordinate
bstem.top	top coordinate
bstem.right	right coordinate
bstem.bottom	bottom coordinate

Notes: BOUNDINGBOX: the bounding box includes the stroke thickness of objects, and is expanded to contain rotated objects.

```
Example: getobject constrain cflag print pflag /* will print some object ↵
         information to the output console */
         say 'The proportional scale flag is '||cflag
         say 'The print flag is '||pflag
```

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.26 getobjectlock

GETOBJECTLOCK

External macros only!

Purpose: Gets the lock status of an object.

Syntax: `getobjectlock [DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format: Parameter Values to enter
 DOCUMENT is the document name.
 WINDOW is the window name.
 OBJECTID is the number of the object. (Default=current)

Result: Returns the lock status <ON|OFF|UNKNOWN> to RESULT. UNKNOWN means that objects with conflicting lock attributes are selected.

Example: `getobjectlock window 'View.1' /* will print the lock status to the ↔
 output console */
 say 'Object lock is '||result`

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.27 getoutlinedisplay

GETOUTLINEDISPLAY

External macros only!

Purpose: Gets the text frame outline display status.

Syntax: `getoutlinedisplay [WINDOW name/S]`

Format: Parameter Values to enter
 WINDOW is the window name. (Default=current)

Result: Returns the outline display status <ON|OFF> to RESULT.

Example: `getoutlinedisplay /* will print the outline display status to the ↔
 output console */
 say 'Text Frame Outline Display: '||result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.28 getpagedesc

GETPAGEDESC

External macros only!

Purpose: Gets the description of a page.

Syntax: `getpagedesc [PAGE number/S]`

Format: Parameter Values to enter
 PAGE is the page number. (Default=current)

Result: The description is returned to RESULT.

Example: `getpagedesc /* will return the description of the current page */`
`pagedesc=result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.29 getpagemasterpage

GETPAGEMASTERPAGE

External macros only!

Purpose: Gets the master page name for a page, whether its objects are shown and whether they're shown in front or in back of objects on the page itself.

Syntax: `getpagemasterpage [MASTERPAGE name/V] [SIDE name/V]`
`[DEPTH level/V] [PAGE number/S | DOCUMENT name/S | WINDOW name/S]`

Format: Parameter Values to enter
 MASTERPAGE gets the master page name.
 SIDE gets the master page side (for double sided pages).
 DEPTH gets the object depth.
 PAGE is the page number. (Default=current)
 DOCUMENT is the document name.
 WINDOW is the window name.

Result: If the master page objects are visible on the actual page, ON will be returned to RESULT, else OFF will be returned.

MASTERPAGE: returns the name of the master page for the actual page to the specified variable.

SIDE: returns the master page side <LEFT|CENTER|RIGHT> to the specified variable.

DEPTH: returns <INFRONT|INBACK> to the specified variable.

Example: `getpagemasterpage masterpage name page 5 /* will print the master page ↔`
`for a page to the output console */`
`say 'The master page for the current page is '||name`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.30 getpagename

GETPAGENAME

External macros only!

Purpose: Gets the name of a page.

Syntax: `getpagename [PAGE number/S]`

Format: Parameter Values to enter
 PAGE is the page number. (Default=current)

Result: The name is returned to RESULT.

Example: `getpagename /* will return the name of the current page */
 pagename=result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.31 getpagenumbering

GETPAGENUMBERING

External macros only!

Purpose: Gets the page numbering system.

Syntax: `getpagenumbering <stem/V>
 [DOCUMENT name/S | CHAPTER name/S | WINDOW name/S]`

Format: Parameter Values to enter
 stem is the name of a stem variable for the information.
 DOCUMENT is the document name. (Default=current)
 CHAPTER is the document/chapter name.

Result: `stem.startmode` AUTOMATIC|AUTOEVEN|AUTOODD|CUSTOM
`stem.start` starting page number
`stem.lengthmode` AUTOMATIC|CUSTOM
`stem.length` number of pages
`stem.masterpage` name of master page to use for blank pages
`stem.format` DEFAULT|LONG|ARABIC|ROMANUPPER|ROMANLOWER|
 ALPHAUPPER|ALPHALOWER
`stem.language` DEFAULT|name
`stem.prefix` string

Example: `getpagenumbering info /* will reset the start page number to one higher ↵
 if page numbering is set to custom */
 if info.startmode='CUSTOM' then do 'setchapternumbering start custom '|| ↵
 info.start+1`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.32 getpath

GETPATH

External macros only!

Purpose: Gets coordinates and information for a path.

Syntax: `getpath [POSITION pstem/V] [FRAME fflag/V]`
`[CONTENTOFFSET cstem/V] [CONTENTSCALE cstem/V] [ROTATION rstem/V]`
`[ABOUT rstem/V] [CONSTRAIN cflag/V] [PRINT pflag/V]`
`[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format:	Parameter	Values to enter
	POSITION	gets the coordinates of the frame.
	CONTENTOFFSET	gets the offset in the frame.
	CONTENTSCALE	gets the scale of the object in the frame.
	ROTATION	gets the rotation of the frame.
	ABOUT	gets the rotation point.
	CONSTRAIN	gets the proportional scale flag state.
	PRINT	gets the print flag state.
	DOCUMENT	is the document name.
	WINDOW	is the window name.
	OBJECTID	is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

POSITION:

<code>pstem.left</code>	left coordinate
<code>pstem.top</code>	top coordinate
<code>pstem.right</code>	right coordinate
<code>pstem.bottom</code>	bottom coordinate

FRAME: returns <ON|OFF>

CONTENTOFFSET:

<code>cstem.x</code>	horizontal offset
<code>cstem.y</code>	vertical offset

CONTENTSCALE:

<code>cstem.h</code>	horizontal scale
<code>cstem.v</code>	vertical scale

ROTATION:

<code>rstem.mode</code>	rotate about <POINT CENTER>
<code>rstem.slant</code>	slant angle
<code>rstem.twist</code>	twist angle

ABOUT:

<code>rstem.x</code>	horizontal point
<code>rstem.y</code>	vertical point

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

```
Example: getpath position coord /* will print the path bounding box to the ←
output console */
say 'Left: ' || coord.left
say 'Top: ' || coord.top
say 'Right: ' || coord.right
say 'Bottom: ' || coord.bottom
```

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.33 getpicture

GETPICTURE

External macros only!

Purpose: Gets coordinates and information for a picture.

```
Syntax: getpicture [POSITION pstem/V] [FRAME fflag/V]
[CONTENTOFFSET cstem/V] [CONTENTSCALE cstem/V] [ROTATION rstem/V]
[ABOUT rstem/V] [DPI dstem/V] [CONSTRAIN cflag/V] [PRINT pflag/V]
[FILEINFO fstem/V] [DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]
```

Format:	Parameter	Values to enter
	POSITION	gets the coordinates of the frame.
	CONTENTOFFSET	gets the offset in the frame.
	CONTENTSCALE	gets the scale of the object in the frame.
	ROTATION	gets the rotation of the frame.
	ABOUT	gets the rotation point.
	CONSTRAIN	gets the proportional scale flag state.
	PRINT	gets the print flag state.
	FILEINFO	gets the file status of the picture.
	DOCUMENT	is the document name.
	WINDOW	is the window name.
	OBJECTID	is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

```
POSITION:
pstem.left    left coordinate
pstem.top     top coordinate
pstem.right   right coordinate
pstem.bottom  bottom coordinate
```

```
FRAME: returns <ON|OFF>
```

```
CONTENTOFFSET:
cstem.x       horizontal offset
cstem.y       vertical offset
```

```
CONTENTSCALE:
cstem.h       horizontal scale
cstem.v       vertical scale
```

ROTATION:
 rstem.mode rotate about <POINT|CENTER>
 rstem.slant slant angle
 rstem.twist twist angle

ABOUT:
 rstem.x horizontal point
 rstem.y vertical point

DPI:
 dstem.x horizontal resolution
 dstem.y horizontal resolution

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

FILEINFO:
 fstem.mode <INTERNAL|EXTERNAL>
 fstem.file filepath and name

Example: `getpicture position coord /* will print the picture bounding box to the ←
 output console */
 say 'Left: ' || coord.left
 say 'Top: ' || coord.top
 say 'Right: ' || coord.right
 say 'Bottom: ' || coord.bottom`

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.34 getpicturedisplay

GETPICTUREDISELAY

External macros only!

Purpose: Gets the picture display status.

Syntax: `getpicturedisplay [WINDOW name/S]`

Format: Parameter Values to enter
 WINDOW is the window name. (Default=current)

Result: Returns the picture display status <ON|OFF> to RESULT.

Example: `getpicturedisplay /* will print the picture display status to the ←
 output console */
 say 'Picure Display: ' || result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.35 getpolygon

GETPOLYGON

External macros only!

Purpose: Gets coordinates and information for a polygon.

Syntax: `getpolygon [POSITION pstem/V] [SHAPE sstem/V]`
`[ROTATION rstem/V] [ABOUT rstem/V] [CONSTRAIN cflag/V]`
`[PRINT pflag/V] [DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format:	Parameter	Values to enter
	POSITION	gets the coordinates of the polygon.
	SHAPE	gets the polygon shape, sides and angles.
	ROTATION	gets the rotation of the polygon.
	ABOUT	gets the rotation point.
	CONSTRAIN	gets the proportional scale flag state.
	PRINT	gets the print flag state.
	DOCUMENT	is the document name.
	WINDOW	is the window name.
	OBJECTID	is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

POSITION:

<code>pstem.centerx</code>	horizontal center coordinate
<code>pstem.centery</code>	vertical center coordinate
<code>pstem.radiusx</code>	horizontal radius
<code>pstem.radiusy</code>	vertical radius

SHAPE:

<code>sstem.type</code>	<NORMAL STAR PUFFY SCALLOP WAVY>
<code>sstem.sides</code>	number of sides
<code>sstem.offsetangle</code>	pre-rotation angle
<code>sstem.deflection</code>	alternate point radius
<code>sstem.deflectionangle</code>	alternate point angle

ROTATION:

<code>rstem.mode</code>	rotate about <POINT CENTER>
<code>rstem.slant</code>	slant angle
<code>rstem.twist</code>	twist angle

ABOUT:

<code>rstem.x</code>	horizontal point
<code>rstem.y</code>	vertical point

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

Example: `getpolygon position coord /* will print the polygon center coordinates ↔`
`to the output console */`

```
say 'Horizontal Center: ' || coord.centerx
say 'Vertical Center:   ' || coord.centery
```

Command Format
 Object ID numbers
 DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.36 getportname

GETPORTNAME

External macros only!

Purpose: Gets the name of PageStream's ARexx port.

Syntax: getportname (no parameters)

Result: The name is returned to RESULT.

Example: getportname /* will return the portname */
 name=result

Command Format

1.37 getredo

GETREDO

External macros only!

Purpose: Gets the type of action that can be redone.

Syntax: getredo [DOCUMENT name/S | WINDOW name/S]

Format: Parameter Values to enter
 DOCUMENT is the document name.
 WINDOW is the window name.

Result: Returns the type of action that can be redone to RESULT. For example, if the movement of an object had just been undone, it would return "Move".

Example: getredo
 say "Can redo "||result /* will print the redoable action to the output ↵
 console */

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.38 getrefreshmode

GETREFRESHMODE

External macros only!

Purpose: Gets the refresh mode of a window.

Syntax: `getrefreshmode [WINDOW name/S]`

Format: Parameter Values to enter
 WINDOW is the window name. (Default=current)

Result: Returns the refresh mode <ON|OFF|WAIT> to RESULT.

Example: `getrefreshmode window 'View.1' /* will print the refresh mode to the ↔
 output console */
 say 'Refresh Mode: ' || result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.39 getregion

GETREGION

External macros only!

Purpose: Opens a small message requester to instruct the user to click on the page to return a coordinate value to the macro. The requester has a Cancel gadget.

Syntax: `getregion <stem/V> [MESSAGE message/S]`

Format: Parameter Values to enter
 stem is the name of a stem variable for the mouse coordinates.
 MESSAGE is the message to display in the requester.
 (Max length=55)

Result: If Cancel is chosen, it sets RC to 10. If the user clicks on the page or draws a region, it sets RC to 0 and returns the coordinates to the stem variable.

```
stem.x1  start horizontal coordinate
stem.y1  start vertical coordinate
stem.x2  end horizontal coordinate
stem.y2  end vertical coordinate
```

Example: `'getregion coord message "Drag to define an area"' /* will print the ↔
 coordinates to the output console */
 button=RC
 if RC=0 then do
 say coord.x1
 say coord.y1
 say coord.x2
 say coord.y2`

end

Command Format

1.40 getrotation

GETROTATION

External macros only!

Purpose: Gets the rotation status of an object.

Syntax: `getrotation <stem/V> [DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format: Parameter Values to enter
 stem is the name of a stem variable for the information.
 DOCUMENT is the document name.
 WINDOW is the window name.
 OBJECTID is the number of the object.

Result: stem.slant
 stem.twist

Example: `getrotation amount /* will print the rotation status to the output ↔
 console */
 say 'Slant: ' || amount.slant
 say 'Twist: ' || amount.twist`

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.41 getrulerdisplay

GETRULERDISPLAY

External macros only!

Purpose: Gets the ruler display status.

Syntax: `getrulerdisplay [OFFSET ostem/V] [ZERO zstem/V] [MSYS mstem/V]
 [DIRECTION dstem/V] [WINDOW name/S]`

Format: Parameter Values to enter
 OFFSET gets the ruler offset in pixels from the top left corner of the window.
 ZERO gets the ruler zero point.
 MSYS gets the measurement system units for the rulers.
 DIRECTION gets the direction in which the rulers are measured.
 WINDOW is the window name. (Default=current)

Result: Returns the ruler display status <ON|OFF> to RESULT.

```

ostem.x    horizontal display offset
ostem.y    vertical display offset
zstem.x    horizontal zero offset
zstem.y    vertical zero offset
mstem.h    horizontal ruler measurement system
            <INCHES|CENTIMETERS|MILLIMETERS|PICAS|POINTS|
            PRINTERPICAS|PRINTERPOINTS|CICEROS|DIDOTPOINTS|FEET|
            |METERS|SAMEAS>
mstem.v    vertical ruler measurement system (same as above)
dstem.h    horizontal measurement direction <LEFT|RIGHT>
dstem.v    vertical measurement direction <UP|DOWN>

```

```

Example: getrulerdisplay msys system /* will print the ruler status and ↔
        measurement system to the output console */
        say 'The rulers are '||result
        say 'Horizontal measurement system: '||system.h
        say 'Vertical measurement system: '||system.v

```

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.42 getselectedobjects

GETSELECTEDOBJECTS

External macros only!

Purpose: Gets the ID's of all selected objects, as well as their bounding box.

```

Syntax: getselectedobjects [IDLIST idstem/V] [BOUNDINGBOX bstem/V]
        [DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]

```

```

Format: Parameter    Values to enter
IDLIST              gets the ID's of each selected object.
BOUNDINGBOX        gets the bounding box of all selected objects.
DOCUMENT            is the document name.
WINDOW              is the window name.
OBJECTID           is the number of the object. (Default=current)

```

Result: The object count is returned to RESULT.

IDLIST: The object ID's of the selected objects are returned to idstem.# where # is a number from 0 to the number of selected objects less 1.

```

BOUNDINGBOX:
bstem.left         left coordinate
bstem.top          top coordinate
bstem.right        right coordinate
bstem.bottom       bottom coordinate

```

Notes: BOUNDINGBOX: the bounding box includes the stroke thickness of

objects, and is expanded to contain rotated objects.

```
Example: getselectedobjects boundingbox coord /* will print the objects' ←
coordinates to the output console */
say 'Left:  ' || coord.left
say 'Top:   ' || coord.top
say 'Right: ' || coord.right
say 'Bottom: ' || coord.bottom
```

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.43 gettextframe

GETTEXTFRAME

External macros only!

Purpose: Gets the number of columns in a frame, and the gutter space between them.

```
Syntax: gettextframe [COLUMNS number/V] [GUTTER space/V]
[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]
```

Format: Parameter Values to enter
 COLUMNS gets the number of columns in the frame.
 GUTTER gets the space between columns in the frame.
 DOCUMENT is the document name.
 WINDOW is the window name.
 OBJECTID is the number of the object. (Default=current)

Result: Returns the object's text frame status <ON|OFF> to RESULT.

```
Example: gettextframe columns count gutter space
if result='ON' then say 'Columns: 'count', Gutter: 'space /* will print ←
the column count and gutter space to the output console */
```

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.44 gettextobj

GETTEXTOBJ

External macros only!

Purpose: Gets coordinates and information for a frameless text object.

```
Syntax: gettextobj [POSITION pstem/V] [ROTATION rstem/V] [ABOUT rstem/V]
[CONSTRAIN cflag/V] [PRINT pflag/V]
```

[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]

Format: Parameter Values to enter
 POSITION gets the coordinates of the text object.
 ROTATION gets the rotation of the text object.
 ABOUT gets the rotation point.
 CONSTRAIN gets the proportional scale flag state.
 PRINT gets the print flag state.
 DOCUMENT is the document name.
 WINDOW is the window name.
 OBJECTID is the number of the object. (Default=current)

Result: The object ID is returned to the RESULT variable.

POSITION:
 pstem.left left coordinate
 pstem.top top coordinate
 pstem.right right coordinate
 pstem.bottom bottom coordinate

ROTATION:
 rstem.mode rotate about <POINT|CENTER>
 rstem.slant slant angle
 rstem.twist twist angle

ABOUT:
 rstem.x horizontal point
 rstem.y vertical point

CONSTRAIN: returns <ON|OFF>

PRINT: returns <ON|OFF>

Example: `getttextobj position coord /* will print the text object's bounding box ↵`
`to the output console */`
`say 'Left: ' || coord.left`
`say 'Top: ' || coord.top`
`say 'Right: ' || coord.right`
`say 'Bottom: ' || coord.bottom`

Command Format
 Object ID numbers
 DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.45 gettextwrap

GETTEXTWRAP

External macros only!

Purpose: Gets the text wrap status of an object.

Syntax: `gettextwrap [REGION mode/V] [WRAP type/V] [STANDOFF stem/V]`
`[DOCUMENT name/S | WINDOW name/S | OBJECTID number/I]`

Format: Parameter Values to enter
 REGION is the wrap mode.
 WRAP is the wrap type.
 STANDOFF is the offset of the text from the object.
 DOCUMENT is the document name.
 WINDOW is the window name.
 OBJECTID is the number of the object. (Default=current)

Result: The information is returned to the variables.

REGION: returns <SHAPE|BOUNDINGBOX|FENCE>

WRAP: returns <NOWRAP|WRAPLEFT|WRAPRIGHT|WRAPJUMP|WRAPAROUND|WRAPINSIDE to the specified variable.

STANDOFF:
 stem.x horizontal offset
 stem.y vertical offset

Example: `getttextwrap region mode wrap type standoff offset /* will print the ↵`
`text wrap status to the output console */`
`if mode~='NOWRAP' then`
`say 'Text will '||type||' by '||offset.x||' horizontally, and '|| ↵`
`offset.y||' vertically.'`

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.46 getscreendpi

GETSCREENDPI

External macros only!

Purpose: Gets the display resolution of the current screen.

Syntax: `getscreendpi <dpistem/V>`

Format: Parameter Values to enter
 dpistem gets the display dpi.

Result: dpistem.x horizontal display dpi.
 dpistem.y vertical display dpi.

Example: `getscreendpi rez /* will return the screenname */`
`say rez.x`
`say rez.y`

Command Format

1.47 getscreenname

GETSCREENNAME

External macros only!

Purpose: Gets the name of the screen on which PageStream is opened.

Syntax: getscreenname (no parameters)

Result: The name is returned to RESULT.

```
Example: getscreenname /* will return the screenname */
        screen=result
```

Command Format

1.48 getstring

GETSTRING

External macros only!

Purpose: Opens a requester with one text string gadget into which the user can type a string, and two buttons for exit gadgets.

Syntax: getstring [STRING default/S] [TITLE label/S] [POSBUTTON label/S] [NEGBUTTON label/S]

Format:	Parameter	Values to enter
	STRING	is the default string for the text string gadget. (Default=blank)
	TITLE	is the label for the text string gadget (Max length=8) (Default=blank)
	POSBUTTON	is the label for the left button. (Max length=8) (Default=_Ok)
	NEGBUTTON	is the label for the right button. (Max length=8) (Default=_Cancel)

Result: If NEGBUTTON is chosen, it sets RC to 10. If POSBUTTON is chosen, it sets RC to 0 and returns the string to the RESULT variable.

Notes: If you want a different requester layout, or a, design a custom macro requester with the ALLOCAREXXREQUESTER command.

Precede the character to underscore as a bound keyboard equivalent in the label name. For example, "_Ok" would make "O" the keyboard shortcut for the "Ok" gadget.

```
Example: 'getstring string "Erase this." title "_Text" posbutton "_Yes" negbutton ←
        "_No"
        userstring=RESULT
        button=RC
```

Command Format

1.49 gettextlinkdisplay

GETTEXTLINKDISPLAY

External macros only!

Purpose: Gets the text frame link display status.

Syntax: `gettextlinkdisplay [WINDOW name/S]`

Format: Parameter Values to enter
 WINDOW is the window name. (Default=current)

Result: Returns the text link display status <ON|OFF> to RESULT.

Example: `gettextlinkdisplay /* will print the textlink display status to the
 output console */
 say 'Text Frame Link Display: ' || result`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.50 getundo

GETUNDO

External macros only!

Purpose: Gets the type of action that can be undone.

Syntax: `getundo [DOCUMENT name/S | WINDOW name/S]`

Format: Parameter Values to enter
 DOCUMENT is the document name.
 WINDOW is the window name.

Result: Returns the type of action that can be undone to RESULT. For example, if an object had just been rotated, it would return "Rotate".

Example: `getundo
 say "Can undo " || result /* will print the undoable action to the output
 console */`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.51 getwindowpos

GETWINDOWPOS

Purpose: Gets the size and position of a document window.

Syntax: `getwindowpos [AT atstem/V] [SIZE sizestem/V]
[WINDOW name/S]`

Format: Parameter Values to enter
 AT gets the window position.
 SIZE gets the window size in pixels.
 WINDOW is the window name. (Default=current)

Result: `atstem.x` horizontal window position.
`atstem.y` vertical window position.
`sizestem.w` window width
`sizestem.h` window height

Example: `getwindowpos at coords size coords /* prints the window position and
size to the output console */
say 'Left: '|coords.x
say 'Top: '|coords.y
say 'Width: '|coords.w
say 'Height: '|coords.h`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.52 getwindows

GETWINDOWS

External macros only!

Purpose: Gets the names of the open view windows for a document.

Syntax: `getwindows <stem/V> [DOCUMENT name/S]`

Format: Parameter Values to enter
 stem is the name of a stem variable for the window names.
 DOCUMENT is the document name. (Default=current)

Result: The number of open windows is returned to RESULT.

The names of the open windows are returned to `stem.#` where # is a number from 0 to the number of open windows less 1.

Example: `getwindows winnames /* will print the window names to the output
console */
numwins=result
do count=0 to numwins-1
say winnames.count
end count`

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.53 greeking

GREEKING

Purpose: Changes the text greeking status.

Syntax: greeking <ON | OFF | TOGGLE> [WINDOW name/S]

Format:	Parameter	Values to enter
	ON	toggles on the greeking option.
	OFF	toggles off the greeking option.
	TOGGLE	toggles greeking on and off.
	WINDOW	is the window name. (Default=current)

Example: greeking off

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.54 group

GROUP

Purpose: Groups selected objects into a logical group.

Syntax: group [POSITION left/D top/D right/D bottom/D]
 [CONTENTOFFSET offsetx/D offsety/D] [CONTENTSCALE scalex/P scaley/P]
 [ROTATE angle/A | SKEW slantangle/A twistangle/A | SLANT angle/A |
 TWIST angle/A] [ABOUT pointx/D pointy/D | ABOUTCENTER]
 [CONSTRAIN | FREE] [PRINT | NOPRINT] [INFRONT | INBACK | BEST]
 [DOCUMENT name/S | WINDOW name/S]

Format:	Parameter	Values to enter
	POSITION	is the coordinates of the frame. (Default=current)
	CONTENTOFFSET	is the offset in the frame. (Default=0,0)
	CONTENTSCALE	is the scale of the object in the frame. (Default=0)
	ROTATE	is the rotation angle. (Default=0)
	SKEW	is the slant and twist angle. (Default=0)
	SLANT	is the slant angle. (Default=0)
	TWIST	is the twist angle. (Default=0)
	ABOUT	is the rotation point.
	ABOUTCENTER	rotates around its center. (Default)
	CONSTRAIN	toggles on the resizing constraint.
	FREE	toggles off the resizing constraint.
	PRINT	toggles on the print flag.
	NOPRINT	toggles off the print flag.
	INFRONT	creates at the top of the stack. (Default)
	INBACK	creates at the bottom of the stack.
	BEST	creates at the optimum stack position.
	DOCUMENT	is the document name. (Default=current)
	WINDOW	is the window name. (Default=current)

Result: The identification number (handle) of the new object is returned to the RESULT variable.

Example: group
group best
group 1 1 5.23 6.24 document 'project.doc'

See also UNGROUP.

Command Format

Object ID numbers

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE

1.55 hidewindow

HIDEWINDOW

Purpose: Hides the current window, all windows for the document, or all open windows.

Syntax: hidewindow [CURRENT | ALL | EXCEPT | WINDOW name/S]

Format: Parameter Values to enter
CURRENT will hide the current window. (Default)
ALL will hide all open windows.
EXCEPT will hide all but the current window.
WINDOW will hide a specific window.

Example: hidewindow
hidewindow all

See also REVEALWINDOW

Command Format

DOCUMENT, CHAPTER, WINDOW, PAGE, MASTERPAGE, MPG, STYLETAG & ARTICLE
