ARexx

COLLABORATORS

	TITLE : ARexx	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 24, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

ARexx

1.1 ARexx.guide

Welcome to the DrawStudio ARexx manual. This gives a detailed description of the ARexx commands available in the program.

OBJECT_MOVE	Move objects on the page
OBJECT_ROTATE	Rotate objects
OBJECT_SCALE	Resizes the objects
OBJECT_SELECT	Selects the named object
OBJECT_SPECS_GET	Gets information on the selected object(s)
OBJECT_SPLIT_COMPOUND	Splits a compound object into Beziers
OBJECT_TOBACK	Move the objects to the back of the page
OBJECT_TOFRONT	Move the objects to the front of the page
OBJECT_UNGROUP	Ungroups objects into individual objects
PAGE_SPECS_GET	Gets information on the current page
PASTE	Pastes objects from the clipboard
PROJECT_CLOSE	Closes the current project
PROJECT_LOCK	Locks the project before executing ARexx commands
PROJECT_NEW	Creates a new project
PROJECT_OPEN	Opens a project from disk
PROJECT_PLACE	Imports a graphics to place on the page
PROJECT_SAVE	Saves the project to disk
PROJECT_UNLOCK	Removes the GUI lock after executing ARexx commands
QUIT	Exits the DrawStudio program
REDO	Redoes the last undone command
REDRAW_OFF	Turns off the redraw for faster ARexx execution
REDRAW_ON	Turns on the redraw and redraws the project
REQ_FILE	Opens a general purpose file requester
REQ_MESSAGE	Opens a general porpose message requester
SCRIPT	Runs a named script
UNDO	Undos the last action

1.2 ARexx.guide/Overview

```
Overview
********
```

This online manual provides detailed information on the ARexx commands available to you when using DrawStudio. ARexx has 3 main uses in DrawStudio:

1. ARexx can be used to automate repetative tasks that usually require lots of menu accesses. Scripts can be built up from many MENU commands, which is a level of programming that can be understood by even the novice programmer.

2. Complex scripts can be generated to perform effects and draw complex mathematical shapes.

3. ARexx can be used to transfer information between DrawStudio and other ARexx-aware programs.

The commands listed here show the command name as well as any input parameters and any values returned. Examples are also given showing the commands in use.

1.3 ARexx.guide/Command Templates

Command Templates
****

The parameters passed to the ARexx commands closely follow the Amiga's style guidelines. The parsing of the arguments follows the standard template format described below.

Commands are always of the form:

```
command [options]
```

The command may be something like PROJECT_OPEN or OBJECT_SCALE and the options may be filenames, numbers etc... A typical command template may look like:

```
OBJECT_ROTATE ANGLE/A,CX,CY
```

The commands and options are not case sensitive, therefore 'OBJECT_ROTATE', 'Object_Rotate' or 'object_rotate' can be used to rotate an object. The options after the command name are separated by commas, and are named (e.g. ANGLE or CX are option names). After the name, follows an optional modifier (e.g. /A or /S are modifiers) which describes what type of information the option specifies.

When using the command, the option names may be ommitted if the parameters for the command are given in the same order as the options in the template, but for clarity it is recommended that the option names be used.

The following modifiers are used:

No modifier
: If the option has no modifier, the option is expecting a string or a floating point number. Strings are lines of text with no spaces; to use a string with a space, place the string in double-quotes ("). Floating point numbers are values with a (optional) decimal place (e.g. 15.7, 9.462 or 5).

  Distance and length value can be given length specifiers. For example, you may move an object 1 inch to the left and 5 cm down with the command:

```
OBJECT_MOVE '1in' '5cm'
```

  If no units are given, points are assumed.

Multiple strings (/M)
: Many strings or floating point numbers can be specified if an option uses this modifier.

Numeric (/N)
: Numeric options allow both positive and negative integers. Decimal points are not allowed in a numeric argument.

Boolean (/S)
     Some options can be specified to "switch" that option on. By
     leaving the option out, the option is switched off.

Keyword (/K)
     A keyword option shows that the option name must be used to set
     this option.

Always (/A)
     This option must always be included in this command.

   In practice, it soon becomes very easy to interpret command templates - some examples with explanations are given below:

     OBJECT_ROTATE ANGLE/A,CX,CY

   The command 'OBJECT_ROTATE' is used to rotate the selected objects in DrawStudio. OBJECT_ROTATE requires an angle of rotation (in degrees) and an optional centre of rotation. The following are valid OBJECT_ROTATE commands:

   The following would rotate all selected objects by 22.5 degrees clockwise around their centre:

     OBJECT_ROTATE 22.5

   The following would rotate the currently selected objects by 50 degrees, anticlockwise, around the top left of the page:

     OBJECT_ROTATE 50 0 0

1.4 ARexx.guide/Return Values

Return Values
****

   The return values for the ARexx commands are specified in the same notation as the input parameters, although the types of returned values is more limited than the input parameter types. In order for results to be returned from ARexx commands, it is essential that the line:

     options results

   be placed near the start of the ARexx script.

   Commands may return either strings, numbers or arrays of either. By default, all ARexx commands return their values in a variable called "RESULT". For example, the following call to the CREATE_TEXT command would return the name of the created text object:

     CREATE_TEXT 100.0 200.0 '"Hello world"'

   If the user wishes to return the result in another variable other than RESULT, they may specify the VAR keyword. For example, the

following would perform the same action as above, only putting the result in the varible called "TEXTNAME"

```
CREATE TEXT 100.0 200.0 '"Hello world"' VAR 'FULLNAME'
```

When lengths or distances are returned, the value is always given in points.

1.5 ARexx.guide/Error Checking

Error Checking
****

DrawStudio uses the standard ARexx method of returning errors, with a further extension.

Whenever a command is executed, a variable called "RC" has its value set by ARexx. If the command executed normally, RC is set to zero. If any failure happened, RC is set to either 5 (warning), 10 (failure) or 20 (serious failure).

DrawStudio also sets the value of a further variable called "RC2", which either contains a text description of the reason for failure or a standard AmigaDos error code.

A description string is returned in RC2 if a failure occurs within the execution of a command. RC2 will be an AmigaDos error number if there is an error with the command syntax (e.g. mis-spelled command name or missing quotes).

If, for example the user was to try to use the REQ_MESSAGE command and the user cancelled the requester, the following would be returned:

```
RC = 10
RC2 = "REQ_MESSAGE; User cancelled"
```

If the user tried to open the requester with the command:

```
REQ_MASSAGE
```

the following values would be set:

```
RC = 20
RC2 = 236
```

where AmigaDos error 236 represents 'not implemented', i.e. unknown command. The default blank script template will convert the most common likely AmigaDos error codes into description strings (see the Amiga's AmigaDos manual for a full description of AmigaDos errors).

By default, the blank script template turns on automatic error checking. The line:

```
signal on error
```

tells DrawStudio to jump to the ERROR: label whenever a command fails. The blank script then puts up a requester showing the error.

The user may wish to turn off the automatic error checking to perform error checking themselves. This way, you can trap the error and continue processing sensibly afterwards.

```
     /* Turn off automatic error checking */

     signal off error

     /* Open the requester */

     OBJECT_MOVE 10.0 50.0

     /* Check for the error condition */

     if RC ~= 0 then do
  REQ_MESSAGE TEXT '"An error occurred (probably'||,
     'no objects selected)"'
  end
     else do
  REQ_MESSAGE TEXT '"Objects moved"'
     end
```

1.6 ARexx.guide/ATTRS_FILLCOLOUR_SET

```
ATTRS_FILLCOLOUR_SET
********************

Synopsis
     Allows you to change the current solid fill colour of the selected
     objects.

Parameters Template
     COLOUR/M/N

Return Template
     None

Parameters
    COLOUR/M/N
    Three RGB values plus one optionaly opacity value to set the
    solid colour of the object's fill. The RGB values are between
    0 and 255 and the opacity value is between 0 and 100.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
```

```
     string describing the problem.

Example
     The following turns the object fill of all the currently selected
     to red:

    ATTRS_FILLCOLOUR_SET 255 0 0

     The following set the selected objects' fill to 20 percent opaque
     magenta:

    ATTRS_FILLCOLOUR_SET 255 0 255 20

Known bugs
     None.

See also...
ATTRS_FILLFILL_SET
```

1.7 ARexx.guide/ATTRS_FILLFILL_SET

```
ATTRS_FILLFILL_SET
******************

Synopsis
     Allows you to change the current fill type of the selected objects.

Parameters Template
     TYPE, NAME

Return Template
     None

Parameters
    TYPE
    This allows you to change the object fill type of the selected
    objects to:

       * None; the objects become unfilled.

       * Solid; the objects are filled with a solid colour.

       * Pattern; the objects are filled with a simple, repeating
         pattern.

       * Gradient; the objects are filled with a colour gradient.

       * Bitmap; the objects are filled with a bitmap.

    In every case except "None", a name must also be provided to
    tell DrawStudio which particular fill of the chosen type
    should be used.

    NAME
```

```
    The name of the particular fill to use with the chosen type.
    For example, if you chose a solid fill, this would be the
    name of a particular colour in the colour list.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following removes an object fills from the selected objects:

    ATTRS_FILLFILL_SET "None"

     The following turns the object fill of all the currently selected
     to solid red:

    ATTRS_FILLFILL_SET "Solid" '"Pure Red"'

     This would fill all selected objects with a brick pattern, if the
     default patterns are in use:

    ATTRS_FILLFILL_SET "Pattern" "PATTERN.H"

Known bugs
     None.

See also...
ATTRS_FILLCOLOUR_SET
```

1.8 ARexx.guide/ATTRS_GET

```
ATTRS_GET
*********

Synopsis
     Returns information on the current object's attributes. Only works
     if one object is selected.

Parameters Template
     None

Return Template
     PENTYPE, FILLTYPE, PENNAME, FILLNAME, START, END, JOIN, THICKNESS

Parameters
     None.

Returns
    PENTYPE
```

The type of colouring used for the object's pen. This can be of type:

* None
* Solid
* Pattern
* Gradient
* Bitmap

FILLTYPE
The type of colouring used for the object's fill. All the above fill types are available.

PENNAME
The name of the particular colour type for the object's pen. If the pen is not filled, the PENNAME is always "None". For pattern, gradient and bitmap fills the PENNAME is the name of the appropriate fill where known, otherwise it is "Unknown". Solid colour fills also return the name of the colour where known, or

"RGB R G B O"

when the colour name is unknown ("R" = red value, "G" = green value, "B" = blue value and "O" = opacity value).

FILLNAME
The name of the particular colour type for the object's fill, as described above.

START
The name of the line start shape. These are labelled "LINESTART.A", "LINESTART.B", etc... taking the order from the order found in the linestart pop-up or the attributes requester.

END
The name of the line end shape, named as above, only as "LINEEND.A" etc...

JOIN
The name of the line joins used in Bezier objects. This can be of type:

* Mitre
* Bevel
* Round

DASH
The name of the line dash pattern. These are labelled "DASH.A", "DASH.B", etc... taking order from the order found

```
    in the dash pop-up or the attributes requester.

    THICKNESS
    The thickness of the line. The value is a floating point and
    given in points.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following removes an object fills from the selected objects:

    ATTRS_FILLFILL_SET "None"

     The following turns the object fill of all the currently selected
     to solid red:

    ATTRS_FILLFILL_SET "Solid" '"Pure Red"'

     This would fill all selected objects with a brick pattern, if the
     default patterns are in use:

    ATTRS_FILLFILL_SET "Pattern" "PATTERN.H"

Known bugs
     None.

See also...
ATTRS_FILLCOLOUR_SET
```

1.9 ARexx.guide/ATTRS_PENCOLOUR_SET

```
ATTRS_PENCOLOUR_SET
*******************

Synopsis
     Allows you to change the current solid pen colour of the selected
     objects.

Parameters Template
     COLOUR/M/N

Return Template
     None

Parameters
    COLOUR/M/N
    Three RGB values plus one optionaly opacity value to set the
    solid colour of the object's pen. The RGB values are between
    0 and 255 and the opacity value is between 0 and 100.
```

```
Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following turns the object fill of all the currently selected
     to green:

    ATTRS_PENCOLOUR_SET 0 255 0

     The following sets all selected objects' fill to 50 percent opaque
     blue:

    ATTRS_PENCOLOUR_SET 0 255 0 50

Known bugs
     None.

See also...
ATTRS_PENFILL_SET
```

1.10 ARexx.guide/ATTRS_PENFILL_SET

```
ATTRS_PENFILL_SET
*****************

Synopsis
     Allows you to change the current pen fill type of the selected
     objects.

Parameters Template
     TYPE, NAME

Return Template
     None

Parameters
    TYPE
    This allows you to change the object pen fill type of the
    selected objects to:

       * None; the objects have no visable pen.

       * Solid; the objects are use a solid colour for their pen.

       * Pattern; the objects use a simple, repeating pattern for
         their pen.

       * Gradient; the objects pen is drawn with a colour
```

```
         gradient.

       * Bitmap; the objects use a bitmap to draw the pen.

    In every case except "None", a name must also be provided to
    tell DrawStudio which particular pen of the chosen type
    should be used.

    NAME
    The name of the particular fill to use with the chosen type.
    For example, if you chose a solid pen, this would be the name
    of a particular colour in the colour list.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following removes the pen from the selected objects:

    ATTRS_PENFILL_SET "None"

     The following turns the pen of all the currently selected to solid
     yellow:

    ATTRS_PENFILL_SET "Solid" '"Pure Yellow"'

     This would draw all selected objects' pens with a checked pattern,
     if the default patterns are in use:

    ATTRS_PENFILL_SET "Pattern" "PATTERN.P"

Known bugs
     None.

See also...
ATTRS_PENCOLOUR_SET
```

1.11 ARexx.guide/ATTRS_PENSTYLE_SET

```
ATTRS_PENSTYLE_SET
******************

Synopsis
     Allows you to change the current pen start, end, join and
     thickness.

Parameters Template
     START, END, JOIN, THICKNESS
```

```
Return Template
     None

Parameters
    START
    The name of the line start to use. Allowed values are
    "LINESTART.A", "LINESTART.B", etc... or "None" for a flat
    start. If no value is given the current line's start is not
    changed.

    END
    The name of the line end to use. Allowed values are
    "LINEEND.A", "LINEEND.B", etc... or "None" for a flat start.
    If no value is given the current line's end is not changed.

    JOIN
    The name of the line join to use. Allowed values are "Miter",
    "Bevel" or "Round". If no value is given the current line's
    join is not changed.

    THICKNESS
    The thickness of the line. By default, the value is given in
    points. If no value is given, the current line's thickness
    remains unchanged.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following adds a simple arrowhead the selected objects:

    ATTRS_PENSTYLE_SET "PATTERN.B"

     The following changes the join of connected line (i.e. Bezier
     objects) to rounded joins:

    ATTRS_PENSTYLE_SET JOIN "Round"

     This would change the line thickness of all selected objects to
     5pt:

    ATTRS_PENSTYLE_SET THICKNESS 5.0

     This would change all selected lines to have arrows on both ends
     (double-arrowhead on the start) at 2mm thickness:

    ATTRS_PENSTYLE_SET "PATTERN.J" "PATTERN.C" THICKNESS "2mm"

Known bugs
     None.
```

```
See also...
ATTRS_PENCOLOUR_SET
```

1.12 ARexx.guide/ATTRS_TEXT_SET

```
ATTRS_TEXT_SET
**************

Synopsis
     Allows you to change the style of the selected text objects.

Parameters Template
     FONT, SIZE, ALIGN

Return Template
     None

Parameters
     Any combination of parameters may be supplied, allowing you to
     leave certain text parameters unchanged by not specifying them in
     the command (for example, all text objects could be converted to 12
     point text without changing their font or alignment).

    FONT
    Selects the font, by name, to be used for all the selected
    text objects. The font name must be identical to that used in
    the font requester ("Text/Font..." menu) and is case
    sensitive.

    SIZE
    The size, in points, for the text. Floating point values may
    be used to select the font size, allowing very accurate text
    sizing.

    ALIGN
    Selects the text alignment for the characters in the text
    objects.  The following alignments are allowed:

       * Left.

       * Centre.

       * Right.

    Note that the ALIGN value has no meaning to text objects with
    only one line of text (i.e. without a line break or new line).

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.
```

```
     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following sets all the selected text objects to 12 point text,
     leaving the font and alignment unchanged:

    ATTRS_TEXT_SET SIZE 12.0

     The following changes all selected text objects to 24 point, Times,
     without changing alignment:

    ATTRS_TEXT_SET "Times" 24.0

     This would change all selected text objects to 8.5 point Courier
     text with right alignment:

    ATTRS_TEXT_SET "Courier" 8.5 "Right"

Known bugs
     None.

See also...
     Nothing.
```

1.13 ARexx.guide/COPY

```
COPY
****

Synopsis
     Copies the currently selected objects into the system clipboard.
     They may then be pasted out to elsewhere in the current project,
     to another DrawStudio project or to another application that
     understands the DrawStudio format.

     See also CUT and PASTE.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
```

```
     string describing the problem.

Example
     The following copies the currently selected objects to the
     clipboard:

    COPY

Known bugs
     None.

See also...
CUT, PASTE
```

1.14 ARexx.guide/CREATE_ARC

```
CREATE_ARC
**********

Synopsis
     Creates an arc object. Allows you to specify its position,
     horizontal and vertical size as well as its start and end angles.

Parameters Template
     X/A, Y/A, RX, RY, ANGLE1/A, ANGLE2/A

Return Template
     NAME

Parameters
    X/A
    The horizontal (X) position on the page used for the centre
    of the ellipse which will form the arc. The number is a
    floating point value.

    Y/A
    The vertical (Y) position on the page used for the centre of
    the ellipse which will form the arc. The number is a floating
    point value.

    RX/A
    The horizontal (X) radius of the arc. The number is a
    floating point value.

    RY/A
    The vertical (Y) radius of the arc. The number is a floating
    point value.

    ANGLE1/A
    The starting angle for the arc, in degrees. The angle starts
    from zero degrees being vertical, and increments in a
    clockwise direction (e.g. 90 degrees is "3 o'clock" and 270
    degrees is "9 o'clock").
```

```
    ANGLE2/A
    The end angle for the arc, in degrees.

Returns
    NAME
    The name of the newly created object.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates an arc in the top-left corner of the page,
     drawing only on the page. It has a radius of 1 inch (72 points):

    CREATE_ARC 0.0 0.0 72.0 72.0 90.0 180.0

Known bugs
     None.

See also...
CREATE_ELLIPSE
```

1.15 ARexx.guide/CREATE_BEZIER

```
CREATE_BEZIER
*************

Synopsis
     Creates a Bezier object, consisting of Bezier curves and straight
     line segments. Full control is given over the creation of the
     object, allowing any valid shape of Bezier object to be created
     from ARexx.

Parameters Template
     CLOSED/S, PARAMS/M/A

Return Template
     NAME

Parameters
    CLOSED/S
    Add this switch to form a closed Bezier object; Bezier objects
    are open by default.

    PARAMS/M/A
    Parameters that describe the shape of the Bezier object
    consist of a list of numbers and characters - the numbers
    describe the location of the object points and control points
    and the characters are used to distinguish between line and
    Bezier curve sections.
```

The first 2 numbers of any Bezier object are the starting location (in points) on the page where the object is to begin.

Then follows 2 possible letters - either "L" or "B" depending on whether the following points are to describe a "L"ine section, or a "B"ezier curve.

If a line section is chosen, 2 more numbers follow which describe the coordinate value of the second point on the line. If a Bezier curve is chosen, 6 more numbers follow which describe the 2 Bezier control points and the end point of the Bezier curve (see the manual for details on how Bezier curves are described).

The object can now be ended by not specifying any more parameters, or it can be continued with another line or Bezier section by adding another "L" or "B" character and describing another section.

See the examples below to see the command in use.

Returns

NAME
The name of the newly created object.

Errors

rc = 0 if operation was successful.

rc = 10 if the operation failed for any reason, rc2 will contain a string describing the problem.

Example

The following creates a straight line from the top-left of the page to 1 inch in from the left edge and 2 inches from the top edge of the page:

```
CREATE_BEZIER 0.0 0.0 L 72.0 144.0
```

The next example joins the same two points as the above example (0,0) and (72,144) with a Bezier curve:

```
CREATE_BEZIER 0.0 0.0 B 72.0 0.0 72.0 0.0 72.0 144.0
```

To create a closed object with the above curve:

```
CREATE_BEZIER CLOSED 0.0 0.0 B 72.0 0.0 72.0 0.0 72.0 144.0
```

Or we can create an open object to look similar by drawing a curve and a line:

```
CREATE_BEZIER 0.0 0.0 B 72.0 0.0 72.0 0.0 72.0 144.0,
L 0.0 0.0
```

This creates a 2 inch (closed) square 1 inch from the top and left of the page:

```
CREATE_BEZIER CLOSED 72.0 72.0 L 216.0 72.0 L 216.0 216.0,
```

```
    L 72.0 216.0

Known bugs
     None.

See also...
CREATE_LINE, CREATE_FREEHAND
```

1.16 ARexx.guide/CREATE_ELLIPSE

```
CREATE_ELLIPSE
**************

Synopsis
     Creates an ellipse object. Allows you to specify its position,
     horizontal and vertical size.

Parameters Template
     X/A, Y/A, RX, RY

Return Template
     NAME

Parameters
    X/A
    The horizontal (X) position on the page used for the centre
    of the ellipse which will form the ellipse.

    Y/A
    The vertical (Y) position on the page used for the centre of
    the ellipse which will form the ellipse.

    RX/A
    The horizontal (X) radius of the ellipse.

    RY/A
    The vertical (Y) radius of the ellipse.

Returns
    NAME
    The name of the newly created object.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates an ellipse in the top-left corner of the
     page.  The ellipse is 2 inches wide by half and inch tall and fits
     neatly against the corner of the page:

    CREATE_ELLIPSE 72.0 18.0 72.0 18.0
```

```
Known bugs
     None.

See also...
CREATE_ARC
```

1.17 ARexx.guide/CREATE_FREEHAND

```
CREATE_FREEHAND
***************

Synopsis
     Creates a freehand object. A freehand object is simply a list of
     connected points which are joined together smoothly with lines and
     curves.

Parameters Template
     PARAMS/M/A

Return Template
     NAME

Parameters
    PARAMS/M/A
    The only parameters to the freehand are a list of coordinate
    points which specify the points on the object. The object that
    is created is a Bezier object like that created by the
    CREATE_BEZIER command.

Returns
    NAME
    The name of the newly created object.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates a squiggle on the page:

    CREATE_FREEHAND 0.0 0.0 100.0 120.0 80.0 170.0 200.0 140.0

Known bugs
     None.

See also...
CREATE_BEZIER, CREATE_LINE
```

1.18 ARexx.guide/CREATE_LINE

```
CREATE_LINE
***********

Synopsis
     Creates a line object. The object simply joins 2 coordinate points
     on the page with a straight line.

Parameters Template
     X1/A, Y1/A, X2/A, Y2/A

Return Template
     NAME

Parameters
    X1
    The starting horizontal (X) coordinate of the line.

    Y1
    The starting vertical (Y) coordinate of the line.

    X2
    The end horizontal (X) coordinate of the line.

    Y2
    The end vertical (Y) coordinate of the line.

Returns
    NAME
    The name of the newly created object.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates a straight line from the top-left of the
     page:

    CREATE_LINE 0.0 0.0 72.0 144.0

Known bugs
     None.

See also...
CREATE_FREEHAND, CREATE_BEZIER
```

1.19 ARexx.guide/CREATE_RECT

```
CREATE_RECT
```

```
***********

Synopsis
     Creates a rectangle object. The object creates a rectangle passing
     through the two coordinate points given.

Parameters Template
     X1/A, Y1/A, X2/A, Y2/A

Return Template
     NAME

Parameters
    X1
    The top-left horizontal (X) coordinate of the line.

    Y1
    The top-left vertical (Y) coordinate of the line.

    X2
    The bottom-right horizontal (X) coordinate of the line.

    Y2
    The bottom-right vertical (Y) coordinate of the line.

Returns
    NAME
    The name of the newly created object.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates a rectangle from the top-left of the page:

    CREATE_RECT 0.0 0.0 72.0 144.0

Known bugs
     None.

See also...
     Nothing.
```

1.20 ARexx.guide/CREATE_RND

```
CREATE_RND
**********

Synopsis
     Creates a rounded rectangle object. The object creates a rectangle
     with rounded corners passing through the two coordinate points
```

```
     given.

Parameters Template
     X1/A, Y1/A, X2/A, Y2/A, RADIUS

Return Template
     NAME

Parameters
    X1/A
    The top-left horizontal (X) coordinate of the line.

    Y1/A
    The top-left vertical (Y) coordinate of the line.

    X2/A
    The bottom-right horizontal (X) coordinate of the line.

    Y2/A
    The bottom-right vertical (Y) coordinate of the line.

    RADIUS
    The radius of the corners of the rectangle. If no value is
    given, the value set in the current set of project attributes
    is used.

Returns
    NAME
    The name of the newly created object.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates a rectangle from the top-left of the page,
     with the current corner radius:

    CREATE_RND 0.0 0.0 72.0 144.0

     This creates the same sized rectangle as above, only with a corner
     radius of 10 points:

    CREATE_RND 0.0 0.0 72.0 144.0 10.0

Known bugs
     None.

See also...
     Nothing.
```

1.21 ARexx.guide/CREATE_TEXT

```
CREATE_TEXT
***********

Synopsis
     Creates a text object with the given text string. The attributes
     of the text can be changed with the ATTRS_TEXT_SET command.

Parameters Template
     X/A, Y/A, STRING/A

Return Template
     NAME

Parameters
    X/A
    The horizontal (X) value of the text start.

    Y/A
    The vertical (Y) value of the text start.

    STRING/A
    The text string the use in the object. Remember, if you are
    using spaces in your text string you must enclose this
    argument in quotes.

Returns
    NAME
    The name of the newly created object.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates the text string "Hello world" 1 inch from
     the left of the page and 2 inches from the top of the page.

    CREATE_TEXT 72.0 144.0 '"Hello world"'

Known bugs
     None.

See also...
     Nothing.
```

1.22 ARexx.guide/CUT

```
CUT
***
```

```
Synopsis
     Copies the currently selected objects into the system clipboard and
     then removes the selected objects from the page. They may then be
     pasted out to elsewhere in the current project, to another
     DrawStudio project or to another application that understands the
     DrawStudio format.

     See also COPY and PASTE.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following cuts the currently selected objects to the clipboard:

    CUT

Known bugs
     None.

See also...
COPY, PASTE
```

1.23 ARexx.guide/MENU

```
MENU
****

Synopsis
     Simulates selecting a particular menu item from the menus.  Allows
     access to all the menu functions from an ARexx script.

     To select a particular menu item, you have to specify upto 3
     parameters. A menu has a "MENU" part (e.g. "Project" or "Edit"),
     and an "ITEM" part (e.g. "Copy" and "Paste" are items of the "Edit"
     menu). Some menu items display SUB-menus, and this can be
     specified (e.g. "Left", "Centre" and "Right" are sub-menus of the
     "Text/Alignment" menu item).
```

```
Parameters Template
     MENU/A, ITEM/A, SUB

Return Template
     None.

Parameters
    MENU/A
    The text of the main menu. This must match the text in the
    menu exactly (including spaces).

    ITEM/A
    The text of the menu item. This must match the text in the
    menu exactly (including spaces).

    SUB
    The text of the menu item's submenu, should the item have a
    submenu.  This must match the text in the menu exactly
    (including spaces).

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following opens the align requester:

    MENU "Object" "Align..."

     The following opens the save requester. Note the use of two sets of
     quotes because the menu item text contains spaces:

    MENU "Project" '"Save as..."'

     This sets the current text alignment to centre:

    MENU "Text" "Alignment" "Centre"

Known bugs
     None.

See also...
     Nothing.
```

1.24 ARexx.guide/OBJECT_ALIGN

```
OBJECT_ALIGN
************
```

Synopsis

Allows you to align all the selected objects to a given set of rules. Object can be aligned relative to the page or relative to themselves and can be left, right or centrally aligned both horizontally and vertically. Furthermore, the objects can also be aligned or distributed to form an even spread of objects.

Parameters Template

TOPAGE/S, ALIGN_X/S, DISTRIBUTE_X/S, ALIGN_Y/S, DISTRIBUTE_Y/S, LEFT/S, CENTRE_X/S, RIGHT/S, WIDTH/S, TOP/S, CENTRE_Y/S, BOTTOM/S, HEIGHT/S

Return Template

None.

Parameters

TOPAGE/S
By default, all objects are aligned or distributed with respect to themselves (i.e. within their own bounding box). Setting this switch will align the objects with respect to the page as a whole.

ALIGN_X/S
By default, the objects' horizontal (X) positions will remain unaltered by this command. Setting this switch, however, will force the objects horizontal positions to be aligned. How they are aligned is determined by the LEFT, CENTRE_X and RIGHT switches, described below.

DISTRIBUTE_X/S
By default, the objects' horizontal (X) positions will remain unaltered by this command. Setting this switch, however, will force the objects horizontal positions to be distributed evenly. How they are distributed is determined by the LEFT, CENTRE_X, RIGHT and WIDTH switches, described below.

ALIGN_Y/S
By default, the objects' vertical (Y) positions will remain unaltered by this command. Setting this switch, however, will force the objects vertical positions to be aligned. How they are aligned is determined by the TOP, CENTRE_Y and BOTTOM switches, described below.

DISTRIBUTE_Y/S
By default, the objects' vertical (Y) positions will remain unaltered by this command. Setting this switch, however, will force the objects vertical positions to be distributed evenly. How they are distributed is determined by the TOP, CENTRE_Y, BOTTOM and HEIGHT switches, described below.

LEFT/S
When using either the ALIGN_X or DISTRIBUTE_X switches, this switch will tell the alignment routines to use each object's left edge as the point of alignment/ distribution.

CENTRE_X/S

When using either the ALIGN_X or DISTRIBUTE_X switches, this switch will tell the alignment routines to use each object's horizontal (X) centre as the point of alignment/ distribution.

RIGHT/S
When using either the ALIGN_X or DISTRIBUTE_X switches, this switch will tell the alignment routines to use each object's right edge as the point of alignment/ distribution.

WIDTH/S
When using DISTRIBUTE_X switch, this switch will tell the alignment routines to leave an equal space between each object - i.e. distribute each object's width evenly.

TOP/S
When using either the ALIGN_Y or DISTRIBUTE_Y switches, this switch will tell the alignment routines to use each object's top edge as the point of alignment/ distribution.

CENTRE_Y/S
When using either the ALIGN_Y or DISTRIBUTE_Y switches, this switch will tell the alignment routines to use each object's vertical (Y) centre as the point of alignment/ distribution.

BOTTOM/S
When using either the ALIGN_Y or DISTRIBUTE_Y switches, this switch will tell the alignment routines to use each object's bottom edge as the point of alignment/ distribution.

HEIGHT/S
When using DISTRIBUTE_Y switch, this switch will tell the alignment routines to leave an equal space between each object - i.e. distribute each object's height evenly.

Returns
Nothing.

Errors
rc = 0 if operation was successful.

rc = 10 if the operation failed for any reason, rc2 will contain a string describing the problem.

Example
The following centres all the selected objects centrally on the page:

```
OBJECT_ALIGN TOPAGE ALIGN_X CENTRE_X
```

The following distributes all the selected object vertically on the page, whilst aligning them to the left of the page:

```
OBJECT_ALIGN TOPAGE ALIGN_X LEFT DISTRIBUTE_Y HEIGHT
```

Known bugs
None.

```
See also...
     Nothing.
```

1.25 ARexx.guide/OBJECT_CLONE

```
OBJECT_CLONE
************

Synopsis
     Creates a copy of the selected objects, offset by a user-defined
     distance.

Parameters Template
     DX, DY

Return Template
     None.

Parameters
    DX
    An optional distance to move the object horizontally. If no
    value is given, the value from the user's preferences will be
    used.

    DY
    An optional distance to move the object vertically. If no
    value is given, the value from the user's preferences will be
    used.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following clones the selected objects, offsetting them the
     default distance as set in the user preferences:

    OBJECT_CLONE

     The following clones the selected objects, offsetting them 1 inch
     to the right and 2 inches above the original objects:

    OBJECT_CLONE 72.0 -144.0

Known bugs
     None.

See also...
     Nothing.
```

1.26 ARexx.guide/OBJECT_CONVERTTOBEZIER

```
OBJECT_CONVERTTOBEZIER
**********************

Synopsis
     Converts selected objects to Bezier objects, where possible.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

     rc = 10 if some objects could not be converted to Beziers (e.g.
     bitmap objects cannot be converted to Bezier objects).

Example
     The following converts selected objects to Bezier objects:

    OBJECT_CONVERTTOBEZIER

Known bugs
     None.

See also...
     Nothing.
```

1.27 ARexx.guide/OBJECT_DELETE

```
OBJECT_DELETE
*************

Synopsis
     Deletes selected objects.

Parameters Template
```

```
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following deletes the selected objects:

    OBJECT_DELETE

Known bugs
     None.

See also...
     Nothing.
```

1.28 ARexx.guide/OBJECT_DESELECT

```
OBJECT_DESELECT
***************

Synopsis
     Deselects the named object.

Parameters Template
     NAME

Return Template
     None.

Parameters
    NAME
    The name of the object to be deselected. If no object name is
    given, all objects are deselected.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
```

```
     string describing the problem.

Example
     The following deselects all objects:

    OBJECT_DESELECT

     The following deselects the object called "TEXT.5":

    OBJECR_DESELECT "TEXT.5"

Known bugs
     None.

See also...
OBJECT_SELECT
```

1.29 ARexx.guide/OBJECT_GROUP

```
OBJECT_GROUP
************

Synopsis
     Groups the selected objects into a single, grouped, object.

Parameters Template
     None.

Return Template
     NAME

Parameters
     None.

Returns
    NAME
    The name of the group object created.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following groups all the selected objects:

    OBJECT_GROUP

Known bugs
     None.

See also...
OBJECT_UNGROUP
```

1.30 ARexx.guide/OBJECT_JOIN

```
OBJECT_JOIN
***********

Synopsis
     Joins 2 selected objects by connecting their closest points.

Parameters Template
     None.

Return Template
     NAME

Parameters
     None.

Returns
    NAME
    The name of the joined object created.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following joins 2 selected objects at their closest end points.

    OBJECT_JOIN

Known bugs
     None.

See also...
     Nothing.
```

1.31 ARexx.guide/OBJECT_MAKE_COMPOUND

```
OBJECT_MAKE_COMPOUND
********************

Synopsis
     Makes the selected objects a single, compound, object.

Parameters Template
     None.
```

```
Return Template
     NAME

Parameters
     None.

Returns
    NAME
    The name of the compound object created.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates a compound object from the selected objects:

    OBJECT_MAKE_COMPOUND

Known bugs
     None.

See also...
OBJECT_SPLIT_COMPOUND
```

1.32 ARexx.guide/OBJECT_MOVE

```
OBJECT_MOVE
***********

Synopsis
     Moves the selected objects a given distance.

Parameters Template
     X, Y

Return Template
     None.

Parameters
    X
    The distance to move the objects in the horizontal (X)
    direction. This parameter may be ommitted to avoid any
    movement in the X direction.

    Y
    The distance to move the objects in the vertical (Y)
    direction. This parameter may be ommitted to avoid any
    movement in the Y direction.

Returns
     Nothing.
```

```
Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following moves all the selected objects 1 inch to the left:

    OBJECT_MOVE -72.0

     The following moves all the selected objects 2 inchs to the right
     and 1 inch down:

    OBJECT_MOVE 144.0 72.0

Known bugs
     None.

See also...
OBJECT_SCALE, OBJECT_ROTATE
```

1.33 ARexx.guide/OBJECT_ROTATE

```
OBJECT_ROTATE
*************

Synopsis
     Rotates the selected objects a given angle.

Parameters Template
     ANGLE/A, CX, CY

Return Template
     None.

Parameters
    ANGLE/A
    The angle of rotation (in degrees) for the selected objects
    to be rotated. Positive angles are clockwise, negative angles
    are anti-clockwise.

    CX
    The horizontal (X) centre of rotation - if no value is given,
    the horizontal centre of the object is used.

    CY
    The vertical (Y) centre of rotation - if no value is given,
    the vertical centre of the object is used.

Returns
     Nothing.
```

```
Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following rotates the selected objects 45 degrees clockwise
     around the centre of the objects:

    OBJECT_ROTATE 45.0

     The following rotates the selected objects 10 degrees
     anti-clockwise around the top left of the page:

    OBJECT_ROTATE 10.0 0.0 0.0

Known bugs
     None.

See also...
OBJECT_MOVE, OBJECT_SCALE
```

1.34 ARexx.guide/OBJECT_SCALE

```
OBJECT_SCALE
************

Synopsis
     Scales (resizes) the objects by the given amount. Different
     horizontal and vertical scaling values may be given, as well as a
     defined centre of the scale.

Parameters Template
     SX, SY, CX, CY

Return Template
     None.

Parameters
    SX
    The amount to scale the object in the horizontal (X)
    direction. For example, a value of 1.0 means "no scale", 2.0
    means "double" and 0.25 means "quarter" scale. If no value is
    given, the object is not scaled in the X direction.

    SY
    The amount to scale the object in the vertical (Y) direction.
    For example, a value of 1.0 means "no scale", 2.0 means
    "double" and 0.25 means "quarter" scale. If no value is
    given, the object is not scaled in the Y direction.

    CX
    The horizontal (X) centre of the resize - if no value is
```

```
    given, the horizontal centre of the object is used.

    CY
    The vertical (Y) centre of the resize - if no value is given,
    the vertical centre of the object is used.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following halves the size of the selected objects:

    OBJECT_SCALE 0.5 0.5

     The following doubles the width and trebles the height of the
     selected objects around the top-left of the page:

    OBJECT_SCALE 2.0 3.0 0.0 0.0

Known bugs
     None.

See also...
OBJECT_MOVE, OBJECT_ROTATE
```

1.35 ARexx.guide/OBJECT_SELECT

```
OBJECT_SELECT
*************

Synopsis
     Make the named object selected. If no name is given, all the
     objects on the page are selected.

Parameters Template
     NAME

Return Template
     None.

Parameters
    NAME
    The name of the object to be selected. If the object is
    already selected, it will remain selected. If no name is
    given, all the objects on the page are selected.

Returns
     Nothing.
```

```
Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following selects all the objects on the page:

    OBJECT_SELECT

     The following selects the object called "BITMAP.7":

    OBJECT_SELECT "BITMAP.7"

Known bugs
     None.

See also...
OBJECT_DESELECT
```

1.36 ARexx.guide/OBJECT_SPECS_GET

```
OBJECT_SPECS_GET
****************

Synopsis
     Returns position information on the selected object(s).

Parameters Template
     None.

Return Template
     NUMSELECTED/N, NAME, LEFT, TOP, RIGHT, BOTTOM, WIDTH, HEIGHT,
     ROTATION, RNDRADIUS

Parameters
     None.

Returns
    NUMSELECTED/N
    The number of selected objects.

    NAME
    If one object only is selected, this is the name of the
    selected object.

    LEFT
    The location of the left edge of the object(s) bounding box.

    TOP
    The location of the top edge of the object(s) bounding box.
```

```
    RIGHT
    The location of the right edge of the object(s) bounding box.

    BOTTOM
    The location of the bottom edge of the object(s) bounding box.

    WIDTH
    The width of the selected object(s).

    HEIGHT
    The height of the selected object(s).

    ROTATION
    The angle, in degrees, through which the object has been
    rotated.

    RNDRADIUS
    The radius of the corners of a rounded rectangle. This value
    is only returned for rounded rectangle objects.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following returns information on the selected object(s) into a
     stem variable called "OBJSPECS":

    OBJECT_SPECS_GET STEM 'OBJSPECS.'

Known bugs
     None.

See also...
PAGE_SPECS_GET
```

1.37 ARexx.guide/OBJECT_SPLIT_COMPOUND

```
OBJECT_SPLIT_COMPOUND
*********************

Synopsis
     Splits any selected compound objects into their constituent Bezier
     objects.

Parameters Template
     None.

Return Template
     None.

Parameters
```

```
     None.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following splits any selected compound objects into their
     constituent Bezier objects:

    OBJECT_SPLIT_COMPOUND

Known bugs
     None.

See also...
OBJECT_MAKE_COMPOUND
```

1.38 ARexx.guide/OBJECT_TOBACK

```
OBJECT_TOBACK
*************

Synopsis
     Moves the selected objects behind other objects on the page,
     toward the back of the page. Objects may be moved all the way to
     the back of the layer or they may be "shuffled" back slowly toward
     the back.

     Note that the objects will always remain in their current layer,
     i.e. they will never change layers.

Parameters Template
     SHUFFLE/S

Return Template
     None.

Parameters
    SHUFFLE/S
    By default, the selected objects are moved all the way to the
    back of the layer using this command. By using the "SHUFFLE"
    parameter, the objects can be moved back slowly past the other
    objects in the layer to the desired depth.

Returns
     Nothing.

Errors
```

```
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following moves the selected objects to the back of their
     layer:

    OBJECT_TOBACK

     The following shuffles the selected objects back one level:

    OBJECT_TOBACK SHUFFLE

Known bugs
     None.

See also...
OBJECT_TOFRONT
```

1.39 ARexx.guide/OBJECT_TOFRONT

```
OBJECT_TOFRONT
**************

Synopsis
     Moves the selected objects infront of other objects on the page,
     toward the front of the page. Objects may be moved all the way to
     the front of the layer or they may be "shuffled" forwards slowly
     toward the front.

     Note that the objects will always remain in their current layer,
     i.e. they will never change layers.

Parameters Template
     SHUFFLE/S

Return Template
     None.

Parameters
    SHUFFLE/S
    By default, the selected objects are moved all the way to the
    front of the layer using this command. By using the "SHUFFLE"
    parameter, the objects can be moved forwards slowly past the
    other objects in the layer to the desired depth.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.
```

```
     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following moves the selected objects to the front of their
     layer:

    OBJECT_TOFRONT

     The following shuffles the selected objects forward one level:

    OBJECT_TOFRONT SHUFFLE

Known bugs
     None.

See also...
OBJECT_TOBACK
```

1.40 ARexx.guide/OBJECT_UNGROUP

```
OBJECT_UNGROUP
**************

Synopsis
     Ungroups any selected group objects into their constituent
     individual objects. Only one layer of ungrouping is performed,
     i.e. if the group objects themselves contained grouped objects,
     they must be ungrouped seperately.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following ungroups all selected group objects:

    OBJECT_UNGROUP

Known bugs
```

```
     None.

See also...
OBJECT_GROUP
```

1.41 ARexx.guide/PAGE_SPECS_GET

```
PAGE_SPECS_GET
**************

Synopsis
     Returns size and orientation information on the current page.

Parameters Template
     None.

Return Template
     WIDTH, HEIGHT, ORIENTATION

Parameters
     None.

Returns
    WIDTH
    The width, in points, of the current page.

    HEIGHT
    The height, in points, of the current page.

    ORIENTATION
    This is either "Landscape" or "Portrait" depending on whether
    the page is horizontally or vertically orientated.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following returns information on the current page into a stem
     variable called "PAGESPECS":

    PAGE_SPECS_GET STEM 'PAGESPECS.'

Known bugs
     None.

See also...
OBJECT_SPECS_GET
```

1.42 ARexx.guide/PASTE

```
PASTE
*****

Synopsis
     Pastes objects from the clipboard onto the current page. The
     objects can be pasted at a given location or at a user-chosen
     place.

Parameters Template
     X, Y

Return Template
     None.

Parameters
    X
    The horizontal (X) location for the objects to be pasted. This
    value must be given with the Y value (see below), or else an
    interactive paste will be performed where the user can click
    on the page to choose the paste location.

    Y
    The vertical (Y) location for the objects to be pasted. This
    value must be given with the X value (see above), or else an
    interactive paste will be performed where the user can click
    on the page to choose the paste location.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following pastes the objects at the top left of the page:

    PASTE 0.0 0.0

     The following pastes the objects interactively, allowing the user
     to click on the page to choose the paste location:

    PASTE

Known bugs
     This command returns immediately, even if you choose PASTE with no
     parameters (waiting for user input). There is currently no way of
     waiting for the user to perform the paste.

See also...
CUT, COPY
```

1.43 ARexx.guide/PROJECT_CLOSE

```
PROJECT_CLOSE
*************

Synopsis
     Closes the current project, removing any project locks (see
     PROJECT_LOCK).

Parameters Template
     FORCE/S

Return Template
     None.

Parameters
    FORCE/S
    Close the current project without asking the user's
    permission.  By default, if the project has been changed you
    will be asked to confirm that the project is about to be
    closed.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following closes the current project, asking permission if the
     project has been changed since the last time it was saved:

    PROJECT_CLOSE

     The following closes the current project, without asking permission
     if the project has been changed since the last time it was saved:

    PROJECT_CLOSE FORCE

Known bugs
     None.

See also...
PROJECT_OPEN, PROJECT_SAVE
```

1.44 ARexx.guide/PROJECT_LOCK

```
PROJECT_LOCK
************
```

```
Synopsis
     Locks the named project, bringing up the "busy" mouse pointer and
     blocking any input from the GUI. This command should be used before
     any other ARexx commands are executed to avoid the user
     interrupting the ARexx script from the GUI.

     The project may be unlocked at the end of the ARexx script with
     the PROJECT_UNLOCK command. Locking another project will unlock
     the current project before locking the new project.

Parameters Template
     NAME

Return Template
     None.

Parameters
    NAME
    The parameter allows you to specify the name of the project
    to be locked.  If no name, the current project will be locked
    (i.e. the last used project).

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following locks the current project:

    PROJECT_LOCK

     The following locks a project called "PROJECT.3":

    PROJECT_LOCK "PROJECT.3"

Known bugs
     None.

See also...
PROJECT_UNLOCK, PROJECT_CLOSE
```

1.45 ARexx.guide/PROJECT_NEW

```
PROJECT_NEW
***********

Synopsis
     Creates a new project, ready for editing.
```

```
     Currently, a maximum of 10 projects may be open at any one time.

Parameters Template
     None.

Return Template
     NAME

Parameters
     None.

Returns
    NAME
    The name of the newly opened project.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following creates a new project:

    PROJECT_NEW

Known bugs
     None.

See also...
PROJECT_CLOSE, PROJECT_OPEN
```

1.46 ARexx.guide/PROJECT_OPEN

```
PROJECT_OPEN
************

Synopsis
     Opens a project from disk, ready for editing.

     Currently, a maximum of 10 projects may be open at any one time.

Parameters Template
     FILE/A

Return Template
     NAME

Parameters
    FILE/A
    The filename of the project to open.

Returns
    NAME
```

```
    The name of the newly opened project.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following opens a new project from the RAM disk:

    PROJECT_OPEN "ram:penknife.dsdr"

     The following opens a project from the hard disk. Notice the use
     of double-quotes because the filename contains spaces:

    PROJECT_OPEN '"Work:Draw Files/F1Car.dsdr"'

Known bugs
     None.

See also...
PROJECT_NEW, PROJECT_CLOSE
```

1.47 ARexx.guide/PROJECT_PLACE

```
PROJECT_PLACE
*************

Synopsis
     Opens a graphic from disk, placing it on the current page.
     DrawStudio will auto-detect the format of the graphic as it loads.

Parameters Template
     FILE/A, X, Y

Return Template
     None.

Parameters
    FILE/A
    The filename of the graphic to import.

    X
    The horizontal (X) position to place the graphic. This value
    must be given along with the Y value (see below), otherwise
    the user will be required to manually place the graphic on
    the page.

    Y
    The vertical (Y) position to place the graphic. This value
    must be given along with the X value (see above), otherwise
    the user will be required to manually place the graphic on
    the page.
```

```
Returns
     Nothing

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following opens a graphic from the RAM disk, placing it in the
     top-left corner of the page:

    PROJECT_PLACE "ram:FiGlover.gif" 0.0 0.0

     The following allows you to place a graphic on the current page by
     clicking:

    PROJECT_PLACE "Work:Graphics/Pics/Cheetah.ilbm"

Known bugs
     None.

See also...
     Nothing.
```

1.48 ARexx.guide/PROJECT_SAVE

```
PROJECT_SAVE
************

Synopsis
     Saves the current project out to disk. If the filename given
     already exists, the user will be asked whether they wish to
     overwrite it unless the FORCE switch is given.

Parameters Template
     FILE/A, FORCE/S

Return Template
     None.

Parameters
    FILE/A
    The filename of the project to save.

    FORCE/S
    By specifying the FORCE switch, the user will not be warned if
    they are about to overwrite an existing file.

Returns
     Nothing
```

```
Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following saves the current project to disk:

    PROJECT_SAVE "Work:Draw/Poster.dsdr"

     The following saves the current project to RAM disk, overwriting
     the file without asking for user confirmation:

    PROJECT_SAVE "ram:artwork.dsdr" FORCE

Known bugs
     None.

See also...
PROJECT_CLOSE, PROJECT_NEW
```

1.49 ARexx.guide/PROJECT_UNLOCK

```
PROJECT_UNLOCK
**************

Synopsis
     Removes the GUI lock, added with the PROJECT_LOCK command.  This
     will allow the user to use DrawStudio's user interface. You should
     end all ARexx scripts with a PROJECT_UNLOCK to return control to
     the user.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following unlocks the currently locked project:
```

```
    PROJECT_UNLOCK

Known bugs
     None.

See also...
PROJECT_LOCK, PROJECT_CLOSE
```

1.50 ARexx.guide/QUIT

```
QUIT
****

Synopsis
     Exits the program. The user will be asked whether they wish to
     save any open projects, unless the FORCE switch is used.

Parameters Template
     FORCE/S

Return Template
     None.

Parameters
    FORCE/S
    By specifying the FORCE switch, the user will not be warned if
    they are about quit the program and there are unsaved projects
    open.

Returns
     Nothing

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following quits DrawStudio, asking the user if they wish to
     save any unsaved projects:

    QUIT

     The following quits DrawStudio without asking for user
     confirmation:

    QUIT FORCE

Known bugs
     The command does not return an error if the program does not quit.
     To determine whether the program has quit at the moment, try and
     lock or unlock a project after issuing the command and see whether
     this returns an error.
```

```
See also...
     Nothing.
```

1.51 ARexx.guide/REDO

```
REDO
****

Synopsis
     Redoes the last undone command.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following redoes the last undone command:

    REDO

Known bugs
     None.

See also...
UNDO
```

1.52 ARexx.guide/REDRAW_OFF

```
REDRAW_OFF
**********

Synopsis
     Turns the project redraw off, allowing for much faster operations
     on the selected objects as they do not have to be redraw after
     each operation. The redraw is turned back on with the REDRAW_ON
```

```
     command.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following turns the project redraw off:

    REDRAW_OFF

Known bugs
     None.

See also...
REDRAW_ON
```

1.53 ARexx.guide/REDRAW_ON

```
REDRAW_ON
*********

Synopsis
     Turns the project redraw on, after being turned off with the
     REDRAW_OFF command. The project is redraw to show the changes
     caused by the actions performed whilst the redraw was off.

     The redraw is also turned back on after a PROJECT_UNLOCK command.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing
```

```
Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following turns the project redraw on:

    REDRAW_ON

Known bugs
     None.

See also...
REDRAW_OFF, PROJECT_UNLOCK
```

1.54 ARexx.guide/REQ_FILE

```
REQ_FILE
********

Synopsis
     Opens a general purpose file requester, allowing the user to
     choose a file. The name of the file is returned to the ARexx
     script, allowing any operation to be performed on the chosen file.

Parameters Template
     TITLE, PATH, FILE, AUTOCANCEL/S

Return Template
     FILE, RESULT/N

Parameters
    TITLE
    The text to place in the top title bar of the requester.
    Usually the text should inform the user what will happen to
    the selected file (i.e. "Choose a file to load..." or "Choose
    a file to delete").

    PATH
    The default path for the file requester (e.g. "ram:" or
    "Work:Graphics").

    FILE
    The default file for the file requester.

    AUTOCANCEL/S
    By selecting this switch, the file requester will return an
    error should the user click on the "Cancel" button. By
    default, the requester will return the result of which button
    was clicked in the "RESULT" variable.
```

```
Returns
    FILE
    The full filename (including path) of the chosen file from the
    requester.

    RESULT
    The number of the button chosen ("OK" or "Cancel") to close
    the requester; this is only returned if the AUTOCANCEL
    parameter is not used (see above). "OK" returns 1, "Cancel"
    return 0.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following opens a simple file requester, stopping the script
     if the user cancels the requester:

    REQ_FILE AUTOCANCEL

     The following will delete the chosen file, with "ram:deleteme.dsdr"
     being the default:

    REQ_FILE '"Choose a file to delete..."',
    '"ram:"' '"deleteme.dsdr"' STEM req.

    if req.result == 1 then do
    /* Address the system and delete the file */

    address command delete '"'req.file'"'

    end

Known bugs
     None.

See also...
REQ_MESSAGE, MENU
```

1.55 ARexx.guide/REQ_MESSAGE

```
REQ_MESSAGE
***********

Synopsis
     Opens a general purpose message requester. The user may choose the
     message text and the number/text of the buttons used in the
     requester.

Parameters Template
     TEXT, BUTTONS, AUTOCANCEL/S
```

```
Return Template
     RESULT/N

Parameters
    TEXT/A
    The text to place in the requester.

    BUTTONS
    This parameter lets you specify the number of buttons to use
    in the requester, as well as the text to be used in the
    buttons. If this parameter is not given, the requester
    buttons will be of the form "OK/Cancel".

    To specify a different set of buttons, enter the text for the
    buttons seperated by vertical bar ("|") characters.

    For example, "OK" would produce a simple requester with one
    button and "Good|Bad|Ugly" would produce a requester with 3
    buttons.

    The button number chosen by the user (providing the AUTOCANCEL
    switch is not in use, see below) will be returned in the
    RESULT variable. The rightmost button has the value 0, with
    the buttons being numbered 1,2,3,etc... from the left.

    AUTOCANCEL/S
    By selecting this switch, the file requester will return an
    error should the user click on the "Cancel" or rightmost
    button. By default, the requester will return the result of
    which button was clicked in the "RESULT" variable.

Returns
    FILE
    The full filename (including path) of the chosen file from the
    requester.

    RESULT
    The button number chosen by the user (providing the AUTOCANCEL
    switch is not in use, see above). The rightmost button has
    the value 0, with the buttons being numbered 1,2,3,etc...
    from the left.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following opens a simple OK/Cancel requester, stopping the
     script if the user clicks on Cancel:

    REQ_MESSAGE '"Delete object ?"' AUTOCANCEL

     This will open a simple information requester:
```

```
    REQ_MESSAGE '"Hello world"' "OK"

     The following will interogate the user and respond accordingly:

    REQ_MESSAGE '"What do you think of the show so far?"',
    '"Great|OK|Poor"'

    if result == 1 then
    REQ_MESSAGE '"Thank you!"' "OK"
    else if result == 2 then
    REQ_MESSAGE '"We are trying very hard"' "OK"
    else
    REQ_MESSAGE '"I am sorry you feel that way"' "OK"

Known bugs
     None.

See also...
REQ_FILE, MENU
```

1.56 ARexx.guide/SCRIPT

```
SCRIPT
******

Synopsis
     Calls a named ARexx script, located in the DrawStudio ARexx drawer.

Parameters Template
     FILE/A

Return Template
     None.

Parameters
    FILE/A
    The name of the ARexx script to call. You do not need to give
    a filename path or extension when using the SCRIPT command.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following calls the soft shadow script:

    SCRIPT 'ShadowSoft'

Known bugs
```

```
     None.

See also...
     Nothing.
```

1.57 ARexx.guide/UNDO

```
UNDO
****

Synopsis
     Undos the last action performed on the project. The action may be
     redone with the REDO command.

Parameters Template
     None.

Return Template
     None.

Parameters
     None.

Returns
     Nothing.

Errors
     rc = 0 if operation was successful.

     rc = 10 if the operation failed for any reason, rc2 will contain a
     string describing the problem.

Example
     The following undos the last action:

    UNDO

Known bugs
     None.

See also...
REDO
```