Shockwave Flash

Introduction

PHP offers the ability to create Shockwave Flash files via Paul Haeberli's libswf module.

Note
SWF support was added in PHP 4 RC2. The libswf does not have support for Windows. The development of that library has been stopped, and the source is not available to port it to another systems. For up to date SWF support take a look at the MING functions.

Note
This extension has been moved to the » PECL repository and is no longer bundled with PHP as of PHP 5.0.0.

Installing/Configuring

Requirements

You need the libswf library to compile PHP with support for this extension. You can download libswf at » ftp://ftp.sgi.com/sgi/graphics/grafica/flash/.

Installation

Once you have libswf all you need to do is to configure *--with-swf[=DIR]* where DIR is a location containing the directories include and lib. The include directory has to contain the *swf.h* file and the lib directory has to contain the *libswf.a* file. If you unpack the libswf distribution the two files will be in one directory. Consequently you will have to copy the files to the proper location manually.

Runtime Configuration

This extension has no configuration directives defined in *php.ini*.

Resource Types

This extension has no resource types defined.

Predefined Constants

The constants below are defined by this extension, and will only be available when the extension has either been compiled into PHP or dynamically loaded at runtime.

MOD_COLOR (integer)

MOD_MATRIX (integer)

TYPE_PUSHBUTTON (integer)

TYPE_MENUBUTTON (integer)

BSHitTest (float)

BSDown (float)

BSOver (float)

BSUp (float)

OverDowntoIdle (integer)

IdletoOverDown (integer)

OutDowntoIdle (integer)

OutDowntoOverDown (integer)

OverDowntoOutDown (integer)

OverUptoOverDown (integer)

OverUptoIdle (integer)

IdletoOverUp (integer)

ButtonEnter (integer)

ButtonExit (integer)

MenuEnter (integer)

MenuExit (integer)

Examples

Basic Usage

Once you've successfully installed PHP with Shockwave Flash support you can then go about creating Shockwave files from PHP. You would be surprised at what you can do, take the following code:

Example #1 - SWF example

```
<?php
swf_openfile("test.swf", 256, 256, 30, 1, 1, 1);
swf_ortho2(-100, 100, -100, 100);
swf_defineline(1, -70, 0, 70, 0, .2);
swf_definerect(4, 60, -10, 70, 0, 0);
swf_definerect(5, -60, 0, -70, 10, 0);
swf_addcolor(0, 0, 0, 0);

swf_definefont(10, "Mod");
swf_fontsize(5);
swf_fontslant(10);
swf_definetext(11, "This be Flash wit PHP!", 1);

swf_pushmatrix();
swf_translate(-50, 80, 0);
swf_placeobject(11, 60);
swf_popmatrix();

for ($i = 0; $i < 30; $i++) {
   $p = $i/(30-1);
   swf_pushmatrix();
   swf_scale(1-($p*.9), 1, 1);
   swf_rotate(60*$p, 'z');
   swf_translate(20+20*$p, $p/1.5, 0);
   swf_rotate(270*$p,  'z');
   swf_addcolor($p, 0, $p/1.2, -$p);
   swf_placeobject(1, 50);
   swf_placeobject(4, 50);
   swf_placeobject(5, 50);
   swf_popmatrix();
   swf_showframe();
}

for ($i = 0; $i < 30; $i++) {
   swf_removeobject(50);
   if (($i%4) == 0) {
       swf_showframe();
   }
}

swf_startdoaction();
swf_actionstop();
swf_enddoaction();

swf_closefile();
```

```
?>
```

SWF Functions

swf_actiongeturl

swf_actiongeturl -- Get a URL from a Shockwave Flash movie

Description

void **swf_actiongeturl** (string `$url`, string `$target`)

Gets the URL specified by the parameter *`url`* with the given *`target`*.

Parameters

`url`
: The URL, as a string.

`target`
: The target, as a string.

Return Values

No value is returned.

swf_actiongotoframe

swf_actiongotoframe -- Play a frame and then stop

Description

void **swf_actiongotoframe** (int `$framenumber`)

The swf_actiongotoframe() function will go to the frame specified by *`framenumber`*, play it, and then stop.

Parameters

`framenumber`
: The frame number.

Return Values

No value is returned.

swf_actiongotolabel

swf_actiongotolabel -- Display a frame with the specified label

Description

void **swf_actiongotolabel** (string `$label`)

The swf_actiongotolabel() function displays the frame with the label given by the *`label`* parameter and then stops.

Parameters

`label`
: The frame label.

Return Values

No value is returned.

swf_actionnextframe

swf_actionnextframe -- Go forward one frame

Description

void **swf_actionnextframe** (void)

Go forward one frame.

Return Values

No value is returned.

See Also

- swf_actionprevframe()

swf_actionplay

swf_actionplay -- Start playing the flash movie from the current frame

Description

void **swf_actionplay** (void)

Start playing the flash movie from the current frame.

Return Values

No value is returned.

See Also

- swf_actionstop()

swf_actionprevframe

swf_actionprevframe -- Go backwards one frame

Description

void **swf_actionprevframe** (void)

Go backwards one frame.

Return Values

No value is returned.

See Also

- swf_actionnextframe()

swf_actionsettarget

swf_actionsettarget -- Set the context for actions

Description

void **swf_actionsettarget** (string `$target`)

Sets the context for all actions. You can use this to control other flash movies that are currently playing.

Parameters

`target`
The target, as a string.

Return Values

No value is returned.

swf_actionstop

swf_actionstop -- Stop playing the flash movie at the current frame

Description

void **swf_actionstop** (void)

Stop playing the flash movie at the current frame.

Return Values

No value is returned.

See Also

- swf_actionplay()

swf_actiontogglequality

swf_actiontogglequality -- Toggle between low and high quality

Description

void **swf_actiontogglequality** (void)

Toggle the flash movie between high and low quality.

Return Values

No value is returned.

swf_actionwaitforframe

swf_actionwaitforframe -- Skip actions if a frame has not been loaded

Description

void **swf_actionwaitforframe** (int `$framenumber`, int `$skipcount`)

The swf_actionwaitforframe() function will check to see if the frame, specified by the *`framenumber`* parameter has been loaded, if not it will skip the number of actions specified by the *`skipcount`* parameter. This can be useful for "Loading..." type animations.

Parameters

`framenumber`
: The frame number.

`skipcount`
: The number of actions to skip.

Return Values

No value is returned.

swf_addbuttonrecord

swf_addbuttonrecord -- Controls location, appearance and active area of the current button

Description

void **swf_addbuttonrecord** (int `$states`, int `$shapeid`, int `$depth`)

Allow you to define the specifics of using a button.

Parameters

`states`
: Defines what states the button can have, these can be any or all of the following constants: **BSHitTest**, **BSDown**, **BSOver** or **BSUp**.

`shapeid`
: The second parameter, the *`shapeid`* is the look of the button, this is usually the object id of the shape of the button.

`depth`
: This parameter is the placement of the button in the current frame.

Return Values

No value is returned.

Examples

Example #2 - swf_addbuttonrecord() example

```
<?php
swf_startButton($objid, TYPE_MENUBUTTON);
swf_addButtonRecord(BSDown|BSOver, $buttonImageId, 340);
swf_onCondition(MenuEnter);
swf_actionGetUrl("http://www.example.com", "_level1");
swf_onCondition(MenuExit);
swf_actionGetUrl("", "_level1");
swf_endButton();
?>
```

swf_addcolor

swf_addcolor -- Set the global add color to the rgba value specified

Description

void **swf_addcolor** (float `$r`, float `$g`, float `$b`, float `$a`)

Sets the global add color to the specified color. This color is then implicitly used by the swf_placeobject(), swf_modifyobject() and swf_addbuttonrecord() functions.

The color of the object will be add by the given values when the object is written to the screen.

Parameters

r
: Red value

g
: Green value

b
: Blue value

a
: Alpha value

These parameters can be either positive or negative.

Return Values

No value is returned.

swf_closefile

swf_closefile -- Close the current Shockwave Flash file

Description

void **swf_closefile** ([int `$return_file`])

Close a file that was opened by the swf_openfile() function.

Parameters

`return_file`
If set then the contents of the SWF file are returned from the function.

Return Values

No value is returned.

Examples

Example #3 - Creating a simple flash file based on user input and outputting it and saving it in a database

```
<?php

// The $text variable is submitted by the
// user

// Global variables for database
// access (used in the swf_savedata() function)
$DBHOST = "localhost";
$DBUSER = "sterling";
$DBPASS = "secret";

swf_openfile("php://stdout", 256, 256, 30, 1, 1, 1);

   swf_definefont(10, "Ligon-Bold");
       swf_fontsize(12);
       swf_fontslant(10);

   swf_definetext(11, $text, 1);

   swf_pushmatrix();
       swf_translate(-50, 80, 0);
       swf_placeobject(11, 60);
   swf_popmatrix();
```

```
   swf_showframe();

   swf_startdoaction();
       swf_actionstop();
   swf_enddoaction();

$data = swf_closefile(1);

$data ?
 swf_savedata($data) :
 die("Error could not save SWF file");

// void swf_savedata(string data)
// Save the generated file a database
// for later retrieval
function swf_savedata($data)
{
   global $DBHOST,
          $DBUSER,
          $DBPASS;

   $dbh = @mysql_connect($DBHOST, $DBUSER, $DBPASS);

   if (!$dbh) {
       die (sprintf("Error [%d]: %s",
                    mysql_errno(), mysql_error()));
   }

   $stmt = "INSERT INTO swf_files (file) VALUES ('$data')";

   $sth = @mysql_query($stmt, $dbh);

   if (!$sth) {
       die (sprintf("Error [%d]: %s",
                    mysql_errno(), mysql_error()));
   }

   @mysql_free_result($sth);
   @mysql_close($dbh);
}
?>
```

See Also

- swf_openfile()

swf_definebitmap

swf_definebitmap -- Define a bitmap

Description

void **swf_definebitmap** (int `$objid`, string `$image_name`)

The swf_definebitmap() function defines a bitmap given an image.

Parameters

`objid`
: An SWF object id.

`image_name`
: A GIF, JPEG, RGB or FI image. The image will be converted into a Flash JPEG or Flash color map format.

Return Values

No value is returned.

swf_definefont

swf_definefont -- Defines a font

Description

void **swf_definefont** (int `$fontid`, string `$fontname`)

The swf_definefont() function defines a font parameter and gives it the specified id. It then sets the font given by *`fontname`* to the current font.

Parameters

`fontid`
The id to be given to the font.

`fontname`
The font so be set as current font.

Return Values

No value is returned.

See Also

- swf_definetext()

swf_defineline

swf_defineline -- Define a line

Description

void **swf_defineline** (int `$objid`, float `$x1`, float `$y1`, float `$x2`, float `$y2`, float `$width`)

Defines a line.

Parameters

`objid`
: The object id.

`x1`
: x-coordinate of start point.

`y1`
: y-coordinate of start point.

`x2`
: x-coordinate of end point.

`y2`
: y-coordinate of end point.

`width`
: The line width.

Return Values

No value is returned.

swf_definepoly

swf_definepoly -- Define a polygon

Description

void **swf_definepoly** (int `$objid`, array `$coords`, int `$npoints`, float `$width`)

Defines a polygon given an array of x, y coordinates.

Parameters

`objid`
: The object id.

`coords`
: An array of x, y coordinates.

`npoints`
: The number of overall points that are contained in the array given by *`coords`*

`width`
: The width of the polygon's border, if set to 0.0 the polygon is filled.

Return Values

No value is returned.

swf_definerect

swf_definerect -- Define a rectangle

Description

void **swf_definerect** (int `$objid`, float `$x1`, float `$y1`, float `$x2`, float `$y2`, float `$width`)

Defines a rectangle with an upper left hand coordinate and a lower right hand coordinate.

Parameters

`objid`
: The object id.

`x1`
: x-coordinate of upper left point.

`y1`
: y-coordinate of upper left point.

`x2`
: x-coordinate of lower right point.

`y2`
: y-coordinate of lower right point.

`width`
: Width of the rectangles border, if the width is 0.0 then the rectangle is filled.

Return Values

No value is returned.

swf_definetext

swf_definetext -- Define a text string

Description

void **swf_definetext** (int `$objid`, string `$str`, int `$docenter`)

Defines a text string using the current font and font size.

Parameters

`objid`
The object id.

`str`
The text, as a string.

`docenter`
The *`docenter`* is where the word is centered, if *`docenter`* is 1, then the word is centered in x.

Return Values

No value is returned.

Examples

Example #4 - Horizontal text example

```
<?php

$Xposition = 50;
$Yposition = 50;
$Zposition = 0;
$Obj_depth = 1;

$char = 'THIS IS THE TEXT';

swf_pushmatrix();

swf_definefont (swf_nextid(), "Mod");
swf_fontsize(10);

$secondid = swf_nextid();
swf_definetext($secondid, $char, 1);
swf_translate($Xposition, $Yposition, Zposition);
```

```
swf_placeobject($secondid, $Obj_depth);
swf_popmatrix();

?>
```

See Also

- swf_definefont()

swf_endbutton

swf_endbutton -- End the definition of the current button

Description

void **swf_endbutton** (void)

The swf_endbutton() function ends the definition of the current button.

Return Values

No value is returned.

See Also

- swf_startbutton()

swf_enddoaction

swf_enddoaction -- End the current action

Description

void **swf_enddoaction** (void)

Ends the current action started by the swf_startdoaction() function.

Return Values

No value is returned.

See Also

- swf_startdoaction()

swf_endshape

swf_endshape -- Completes the definition of the current shape

Description

void **swf_endshape** (void)

The swf_endshape() completes the definition of the current shape.

Return Values

No value is returned.

See Also

- swf_startshape()

swf_endsymbol

swf_endsymbol -- End the definition of a symbol

Description

void **swf_endsymbol** (void)

Ends the definition of a symbol that was started by the swf_startsymbol() function.

Return Values

No value is returned.

See Also

- swf_startsymbol()

swf_fontsize

swf_fontsize -- Change the font size

Description

void **swf_fontsize** (float `$size`)

Changes the font size to the value given by the *`size`* parameter.

Parameters

`size`
The font size, as an integer.

Return Values

No value is returned.

swf_fontslant

swf_fontslant -- Set the font slant

Description

void **swf_fontslant** (float `$slant`)

Set the current font slant to the angle indicated by the *`slant`* parameter.

Parameters

`slant`
: Positive values create a forward slant, negative values create a negative slant.

Return Values

No value is returned.

swf_fonttracking

swf_fonttracking -- Set the current font tracking

Description

void **swf_fonttracking** (float `$tracking`)

Set the font tracking to the value specified by the *`tracking`* parameter. This function is used to increase the spacing between letters and text, positive values increase the space and negative values decrease the space between letters.

Parameters

`tracking`
The font tracking.

Return Values

No value is returned.

swf_getbitmapinfo

swf_getbitmapinfo -- Get information about a bitmap

Description

array **swf_getbitmapinfo** (int `$bitmapid`)

Returns information about a bitmap.

Parameters

`bitmapid`
: The bitmap id.

Return Values

Returns an array with the following elements:

- "size" - The size in bytes of the bitmap.
- "width" - The width in pixels of the bitmap.
- "height" - The height in pixels of the bitmap.

swf_getfontinfo

swf_getfontinfo -- Gets font information

Description

array **swf_getfontinfo** (void)

Gets information about the font by giving the height in pixels of a capital A and a lowercase x.

Return Values

Returns an associative array with the following parameters:

- Aheight - The height in pixels of a capital A.
- xheight - The height in pixels of a lowercase x.

swf_getframe

swf_getframe -- Get the frame number of the current frame

Description

int **swf_getframe** (void)

The swf_getframe() function gets the number of the current frame.

Return Values

Returns the current frame number, as an integer.

See Also

- swf_setframe()

swf_labelframe

swf_labelframe -- Label the current frame

Description

void **swf_labelframe** (string `$name`)

Labels the current frame with the given *`name`*.

Parameters

`name`
The frame label.

Return Values

No value is returned.

swf_lookat

swf_lookat -- Define a viewing transformation

Description

void **swf_lookat** (float `$view_x`, float `$view_y`, float `$view_z`, float `$reference_x`, float `$reference_y`, float `$reference_z`, float `$twist`)

Defines a viewing transformation by giving the viewing position and the coordinates of a reference point in the scene.

Parameters

`view_x`
: x-coordinate for the viewing position

`view_y`
: y-coordinate for the viewing position

`view_z`
: z-coordinate for the viewing position

`reference_x`
: x-coordinate for the reference point

`reference_y`
: y-coordinate for the reference point

`reference_z`
: z-coordinate for the reference point

`twist`
: Controls the rotation along with viewer's z axis.

Return Values

No value is returned.

Examples

Example #5 - A simple 3D-rotation around a text

```
<?php
```

```
header('Content-type: application/x-shockwave-flash');

swf_openfile("php://stdout", 320, 200, 25, 1, 1, 1);

swf_ortho(-100, 100, -100, 100, -100, 100); // create 3D coordinates

swf_definefont(0, "Pix3");
swf_addcolor(0, 0, 0, 1);
swf_fontsize(10);
swf_fonttracking(0.2);

for ($i = 0; $i < 628; $i += 8) {
   $j = $i / 100;
   swf_pushmatrix();
   swf_translate(0, 0, 0);
   swf_perspective(100, 1, 0, 10);
   swf_lookat(sin($j) * 60, 50, cos($j) * 60, 0, 0, 0, 0);

   swf_definetext (1, 'HotKey@', 0);
   swf_translate(-50,0,0);

   swf_placeobject(1,10);

   swf_definetext(2, 'example.com', 0);
   swf_translate(55, 0, 0);
   swf_placeobject(2, 11);

   swf_showframe();
   swf_removeobject(10);
   swf_removeobject(11);
   swf_popmatrix();
}

swf_closefile();
?>
```

swf_modifyobject

swf_modifyobject -- Modify an object

Description

void **swf_modifyobject** (int `$depth`, int `$how`)

Updates the position and/or color of the object at the specified *`depth`*.

Parameters

`depth`
: The depth, as an integer.

`how`
: Determines what is updated. *`how`* can either be the constant **MOD_MATRIX** or **MOD_COLOR** or it can be a combination of both. **MOD_COLOR** uses the current mulcolor (specified by the function swf_mulcolor()) and addcolor (specified by the function swf_addcolor()) to color the object. **MOD_MATRIX** uses the current matrix to position the object.

Return Values

No value is returned.

swf_mulcolor

swf_mulcolor -- Sets the global multiply color to the rgba value specified

Description

void **swf_mulcolor** (float `$r`, float `$g`, float `$b`, float `$a`)

Sets the global multiply color to the given one. This color is then implicitly used by the swf_placeobject(), swf_modifyobject() and swf_addbuttonrecord() functions.

The color of the object will be multiplied by the given color values when the object is written to the screen.

Parameters

r
: Red value

g
: Green value

b
: Blue value

a
: Alpha value

These parameters can be either positive or negative.

Return Values

No value is returned.

swf_nextid

swf_nextid -- Returns the next free object id

Description

int **swf_nextid** (void)

The swf_nextid() function returns the next available object id.

Return Values

Returns the id, as an integer.

swf_oncondition

swf_oncondition -- Describe a transition used to trigger an action list

Description

void **swf_oncondition** (int `$transition`)

The swf_oncondition() function describes a transition that will trigger an action list.

Parameters

`transition`
There are several types of possible transitions, the following are for buttons defined as TYPE_MENUBUTTON:

- IdletoOverUp
- OverUptoIdle
- OverUptoOverDown
- OverDowntoOverUp
- IdletoOverDown
- OutDowntoIdle
- MenuEnter (IdletoOverUp|IdletoOverDown)
- MenuExit (OverUptoIdle|OverDowntoIdle)

For TYPE_PUSHBUTTON there are the following options:

- IdletoOverUp
- OverUptoIdle
- OverUptoOverDown
- OverDowntoOverUp
- OverDowntoOutDown
- OutDowntoOverDown
- OutDowntoIdle
- ButtonEnter (IdletoOverUp|OutDowntoOverDown)
- ButtonExit (OverUptoIdle|OverDowntoOutDown)

Return Values

No value is returned.

swf_openfile

swf_openfile -- Open a new Shockwave Flash file

Description

void **swf_openfile** (string `$filename`, float `$width`, float `$height`, float `$framerate`, float `$r`, float `$g`, float `$b`)

Opens a new file. This must be the first function you call, otherwise your script will not work.

Parameters

`filename`
: The path to the SWF file. If you want to send your output to the screen, set this to *php://stdout*.

`width`
: The movie width

`height`
: The movie height

`framerate`
: The frame rate.

`r`
: Red value for the background.

`g`
: Green value for the background.

`b`
: Blue value for the background.

Return Values

No value is returned.

ChangeLog

Version	Description
4.0.1	Support for *php://stdout* was added.

See Also

- swf_closefile()

swf_ortho2

swf_ortho2 -- Defines 2D orthographic mapping of user coordinates onto the current viewport

Description

void **swf_ortho2** (float `$xmin`, float `$xmax`, float `$ymin`, float `$ymax`)

Defines a two dimensional orthographic mapping of user coordinates onto the current viewport, this defaults to one to one mapping of the area of the Flash movie.

If a perspective transformation is desired, swf_perspective() can be used.

Parameters

`xmin`

`xmax`

`ymin`

`ymax`

Return Values

No value is returned.

See Also

- swf_ortho()

swf_ortho

swf_ortho -- Defines an orthographic mapping of user coordinates onto the current viewport

Description

void **swf_ortho** (float `$xmin`, float `$xmax`, float `$ymin`, float `$ymax`, float `$zmin`, float `$zmax`)

Defines an 3D orthographic mapping of user coordinates onto the current viewport.

Parameters

`xmin`

`xmax`

`ymin`

`ymax`

`zmin`

`zmax`

Return Values

No value is returned.

See Also

- swf_ortho2()

swf_perspective

swf_perspective -- Define a perspective projection transformation

Description

void **swf_perspective** (float `$fovy`, float `$aspect`, float `$near`, float `$far`)

Defines a perspective projection transformation.

Note
Various distortion artifacts may appear when performing a perspective projection, this is because Flash players only have a two dimensional matrix. Some are not to pretty.

Parameters

`fovy`
: A field-of-view angle in the y direction.

`aspect`
: The aspect ratio of the viewport that is being drawn onto.

`near`
: The near clipping plane.

`far`
: The far clipping plane.

Return Values

No value is returned.

swf_placeobject

swf_placeobject -- Place an object onto the screen

Description

void **swf_placeobject** (int `$objid`, int `$depth`)

Places the object in the current frame at a specified *`depth`*.

This uses the current mulcolor (specified by swf_mulcolor()) and the current addcolor (specified by swf_addcolor()) to color the object and it uses the current matrix to position the object.

Parameters

`objid`
: The object id.

`depth`
: Must be between 1 and 65535.

Return Values

No value is returned.

swf_polarview

swf_polarview -- Define the viewer's position with polar coordinates

Description

void **swf_polarview** (float `$dist`, float `$azimuth`, float `$incidence`, float `$twist`)

The swf_polarview() function defines the viewer's position in polar coordinates.

Parameters

`dist`
: The distance between the viewpoint to the world space origin.

`azimuth`
: Defines the azimuthal angle in the x,y coordinate plane, measured in distance from the y axis.

`incidence`
: Defines the angle of incidence in the y,z plane, measured in distance from the z axis. The incidence angle is defined as the angle of the viewport relative to the z axis.

`twist`
: Specifies the amount that the viewpoint is to be rotated about the line of sight using the right hand rule.

Return Values

No value is returned.

swf_popmatrix

swf_popmatrix -- Restore a previous transformation matrix

Description

void **swf_popmatrix** (void)

Pushes the current transformation matrix back onto the stack.

Return Values

No value is returned.

See Also

- swf_pushmatrix()

swf_posround

swf_posround -- Enables or Disables the rounding of the translation when objects are placed or moved

Description

void **swf_posround** (int `$round`)

Enables or disables the rounding of the translation when objects are placed or moved, there are times when text becomes more readable because rounding has been enabled.

Parameters

`round`
: Whether to enable rounding or not, if set to the value of 1, then rounding is enabled, if set to 0 then rounding is disabled.

Return Values

No value is returned.

swf_pushmatrix

swf_pushmatrix -- Push the current transformation matrix back unto the stack

Description

void **swf_pushmatrix** (void)

Pushes the current transformation matrix back onto the stack.

Return Values

No value is returned.

See Also

- swf_popmatrix()

swf_removeobject

swf_removeobject -- Remove an object

Description

void **swf_removeobject** (int `$depth`)

Removes the last object drawn at the depth specified by *`depth`*.

Parameters

`depth`
The depth, as an integer.

Return Values

No value is returned.

swf_rotate

swf_rotate -- Rotate the current transformation

Description

void **swf_rotate** (float `$angle`, string `$axis`)

Rotates the current transformation by a given *`angle`* around the given *`axis`*.

Parameters

`angle`
: The rotation angle.

`axis`
: The axis. Valid values axis are *x* (the x axis), *y* (the y axis) or *z* (the z axis).

Return Values

No value is returned.

swf_scale

swf_scale -- Scale the current transformation

Description

void **swf_scale** (float $x, float $y, float $z)

The swf_scale() scales curve coordinates by the given value.

Parameters

x
x scale factor.

y
y scale factor.

z
z scale factor.

Return Values

No value is returned.

swf_setfont

swf_setfont -- Change the current font

Description

void **swf_setfont** (int `$fontid`)

The swf_setfont() sets the current font to the value given by the *`fontid`* parameter.

Parameters

`fontid`
The font identifier.

Return Values

No value is returned.

swf_setframe

swf_setframe -- Switch to a specified frame

Description

void **swf_setframe** (int `$framenumber`)

Changes the active frame to the specified on.

Parameters

`framenumber`
: The frame number to be set.

Return Values

No value is returned.

See Also

- swf_getframe()

swf_shapearc

swf_shapearc -- Draw a circular arc

Description

void **swf_shapearc** (float `$x`, float `$y`, float `$r`, float `$ang1`, float `$ang2`)

Draws a circular arc.

Parameters

x
: x-coordinate of the center.

y
: y-coordinate of the center.

r
: The arc radius.

ang1
: The start angle.

ang2
: The end angle.

Return Values

No value is returned.

swf_shapecurveto3

swf_shapecurveto3 -- Draw a cubic bezier curve

Description

void **swf_shapecurveto3** (float `$x1`, float `$y1`, float `$x2`, float `$y2`, float `$x3`, float `$y3`)

Draw a cubic bezier curve using the given coordinates.

The current position is then set to the *x3*, *y3* coordinate.

Parameters

x1
: x-coordinate of the first off curve control point.

y1
: y-coordinate of the first off curve control point.

x2
: x-coordinate of the second off curve control point.

y2
: y-coordinate of the second off curve control point.

x3
: x-coordinate of the endpoint.

y3
: y-coordinate of the endpoint.

Return Values

No value is returned.

See Also

- swf_shapecurveto()

swf_shapecurveto

swf_shapecurveto -- Draw a quadratic bezier curve between two points

Description

void **swf_shapecurveto** (float `$x1`, float `$y1`, float `$x2`, float `$y2`)

Draws a quadratic bezier curve from the current location, though the two given points.

The current position is then set to the point defined by the *x2* and *y2* parameters.

Parameters

x1
: x-coordinate of the first point.

y1
: y-coordinate of the first point.

x2
: x-coordinate of the second point.

y2
: y-coordinate of the second point.

Return Values

No value is returned.

See Also

- swf_shapecurveto3()

swf_shapefillbitmapclip

swf_shapefillbitmapclip -- Set current fill mode to clipped bitmap

Description

void **swf_shapefillbitmapclip** (int `$bitmapid`)

Sets the fill to bitmap clipped, empty spaces will be filled by the bitmap.

Parameters

`bitmapid`
: The bitmap id.

Return Values

No value is returned.

See Also

- swf_shapefillbitmaptile()

swf_shapefillbitmaptile

swf_shapefillbitmaptile -- Set current fill mode to tiled bitmap

Description

void **swf_shapefillbitmaptile** (int `$bitmapid`)

Sets the fill to bitmap tile, empty spaces will be filled by the bitmap.

Parameters

`bitmapid`
: The bitmap id.

Return Values

No value is returned.

See Also

- swf_shapefillbitmapclip()

swf_shapefilloff

swf_shapefilloff -- Turns off filling

Description

void **swf_shapefilloff** (void)

Turns off filling for the current shape.

Return Values

No value is returned.

swf_shapefillsolid

swf_shapefillsolid -- Set the current fill style to the specified color

Description

void **swf_shapefillsolid** (float `$r`, float `$g`, float `$b`, float `$a`)

Sets the current fill style to solid, and then sets the fill color to the given color.

Parameters

r
: Red value

g
: Green value

b
: Blue value

a
: Alpha value

Return Values

No value is returned.

swf_shapelinesolid

swf_shapelinesolid -- Set the current line style

Description

void **swf_shapelinesolid** (float `$r`, float `$g`, float `$b`, float `$a`, float `$width`)

Sets the current line style to the given color and width.

Parameters

r
: Red value

g
: Green value

b
: Blue value

a
: Alpha value

width
: The line width. If 0.0 is given then no lines are drawn.

Return Values

No value is returned.

swf_shapelineto

swf_shapelineto -- Draw a line

Description

void **swf_shapelineto** (float $x, float $y)

Draws a line to the *x* and *y* coordinates. The current position is then set to that point.

Parameters

x
x-coordinate of the target.

y
y-coordinate of the target.

Return Values

No value is returned.

swf_shapemoveto

swf_shapemoveto -- Move the current position

Description

void **swf_shapemoveto** (float $x, float $y)

Moves the current position to the given point.

Parameters

x
x-coordinate of the target.

y
y-coordinate of the target.

Return Values

No value is returned.

swf_showframe

swf_showframe -- Display the current frame

Description

void **swf_showframe** (void)

Outputs the current frame.

Return Values

No value is returned.

swf_startbutton

swf_startbutton -- Start the definition of a button

Description

void **swf_startbutton** (int `$objid`, int `$type`)

Starts the definition of a button.

Parameters

`objid`
: The object id.

`type`
: Can either be **TYPE_MENUBUTTON** or **TYPE_PUSHBUTTON**. The **TYPE_MENUBUTTON** constant allows the focus to travel from the button when the mouse is down, **TYPE_PUSHBUTTON** does not allow the focus to travel when the mouse is down.

Return Values

No value is returned.

See Also

- swf_endbutton()

swf_startdoaction

swf_startdoaction -- Start a description of an action list for the current frame

Description

void **swf_startdoaction** (void)

Starts the description of an action list for the current frame. This must be called before actions are defined for the current frame.

Return Values

No value is returned.

See Also

- swf_enddoaction()

swf_startshape

swf_startshape -- Start a complex shape

Description

void **swf_startshape** (int `$objid`)

Starts a complex shape.

Parameters

`objid`
: The object id.

Return Values

No value is returned.

See Also

- swf_endshape()

swf_startsymbol

swf_startsymbol -- Define a symbol

Description

void **swf_startsymbol** (int `$objid`)

Defines an object id as a symbol. Symbols are tiny flash movies that can be played simultaneously.

Parameters

`objid`
: The object id you want to define as a symbol.

Return Values

No value is returned.

See Also

- swf_endsymbol()

swf_textwidth

swf_textwidth -- Get the width of a string

Description

float **swf_textwidth** (string `$str`)

Gives the width of the string in pixels, using the current font and font size.

Parameters

`str`
The string.

Return Values

Returns the line width, as a float.

swf_translate

swf_translate -- Translate the current transformations

Description

void **swf_translate** (float $x, float $y, float $z)

Translates the current transformation by the given values.

Parameters

x
x value.

y
y value.

z
z value.

Return Values

No value is returned.

swf_viewport

swf_viewport -- Select an area for future drawing

Description

void **swf_viewport** (float `$xmin`, float `$xmax`, float `$ymin`, float `$ymax`)

Selects an area for future drawing for *`xmin`* to *`xmax`* and *`ymin`* to *`ymax`*, if this function is not called the area defaults to the size of the screen.

Parameters

`xmin`

`xmax`

`ymin`

`ymax`

Return Values

No value is returned.