OpenSpark Interactive Limited

User Manual for:

PiMZ OSControl Xtra

Version 2.2

Contents

1. Introduction

The OSControl Xtra allows you to create professional interfaces by providing a range of sophisticated interface controls within Macromedia Director 7 and above.

OSControl provides you with a set of new cast member types. OSControl members include improved versions of the standard Director controls as well as totally new controls. OSControl includes menus, popup menus, sliders, progress bars, tabs and multi-line static text fields.

The xtra automatically uses the current Operating System settings for fonts and color schemes to draw OSControl sprites. As a result your interface always looks right on both Macintosh and Windows.

Unlike native Director controls, OSControls are not drawn direct-to-stage, so the user can drag screen objects or sprites over the controls giving a more natural look and feel to the overall interface.

1.1 System Requirements

Platform	**Authoring**	**Runtime**
Windows	Director 7 and above Windows 95, 98, NT, Me, 2000, XP	Windows Director Projector Windows 95, 98, NT, Me, XP Shockwave Safe
Macintosh	Director 7 and above Mac OS 8.5 and above	Macintosh Director Projector Mac OS 8.5 and above Mac OS X compatible (carbon) Shockwave Safe

1.2 Installation

Installing the OSControl Xtra

For each copy of Director, copy the following files to the Director Xtras folder:

- **OSControl Xtra**: for playback (this Xtra should be included in projectors).
- **OSControl Options**: provides authoring functionality.

NB: If you are upgrading an installation of a version earlier than V1.1 you must delete the OSControl Options Demo file from the Director Xtras folder.

Registering the OSControl Xtra

If you have purchased a licence for the OSControl Xtra, you will receive a URL from which to download your keyfile. This is called OSCTRL.BRD. Place this in the same folder as the OSControl and OSControl Options xtras. When you launch Director, the OSControl Xtra will check for an OSCTRL.BRD file with the same file path, and use the data in that to verify that the version of Director that you are using is indeed registered.

If you are working with Director 10.1 on Macintosh, and your serial number starts "DRD100-5XXX", "DRD100-7XXX" or "DRD100-9XXX" (Macromedia Volume Package Licence, Not For Resale or Educational Licence), your OSCTRL.BRD file may fail to work. This is because of a change in the way authentication is handled on Macintosh in version 10.1. If this is the case, please contact xtras@openspark.com .

Installing the OSControl Behavior Library cast

Copy the OSControl Behavior Library cast (**OSControl.cst**) to your Director Libs folder. This cast contains example behaviors to set the properties of OSControl sprites.

Installing the View Description List Files for Director 8 and above

To use the property inspector in Director 8 and above you need to copy all the **View Description List** (VDL) files in the Props/Member folder to the Props/Member folder of your installation of Director. Note that properties set with behaviors will only show correctly in the behavior tab of the Property Inspector.

Installing the Example Movies

Copy the OSControl Example Movies to your Director Xtras folder for reference. Example movies are available for both Director 7 and 8. If you use Director 7 movies in Director 8 you should update them using the Update Movies menu option in the Xtras menu.

Please note that the example movies are protected and, if saved under a new file name, the OSControls will cease to work.

Demonstration Version

The demonstration version of the OSControl Xtra is fully functional and free, but protected. Using the demo version, if you save a project, rename a sample movie, or create a projector, all OSControl sprites will be disabled and have a diagonal red line across them. In order to create projects that contain OSControl members you must purchase the Xtra and obtain a key-file.

The Demo version of the OSControl Options Xtra is protected as follows:

- You cannot save the provided example movies under another name without disabling the OSControl members.
- Newly inserted demo OSControl members only work during one session. After you have saved and re-opened movies with demo OSControl members they will cease to function and be shown with a red line through them. These files can be restored later when you

resave them in a copy of Director with the licensed version of the OSControl Options Xtra.

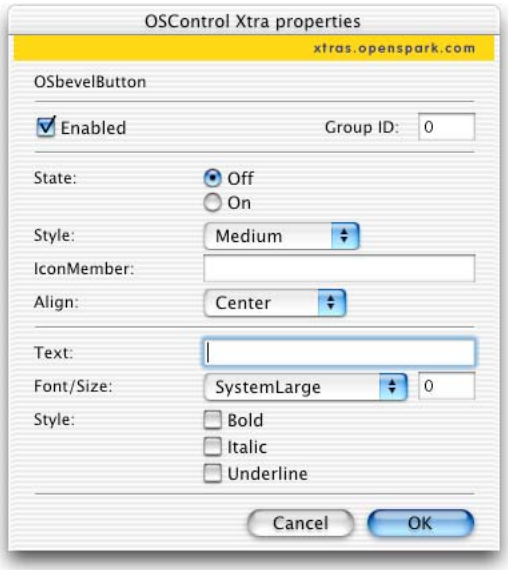
- You cannot make projectors of the example movies or movies made with the demo Xtra.
- You cannot play back the example movies or movies made with the demo Xtra in Shockwave.

NB: If you have not registered the OSControl Xtra, do NOT rename the example files. The OSControl members will cease to function after that. The same will happen if you save the example movies under another name.

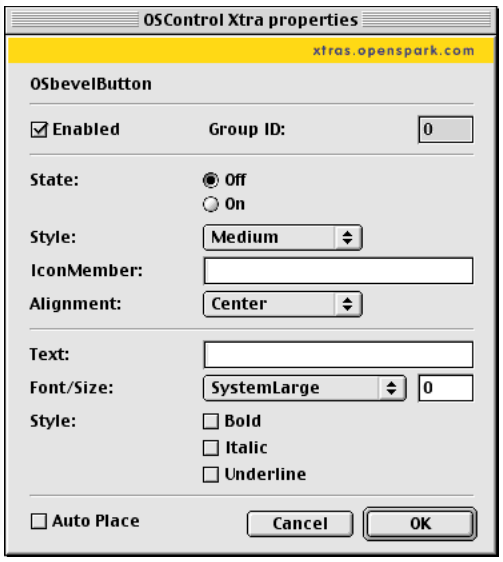
1.3 Inserting OSControl Members

After installing the OSControl Xtras you are ready to insert OSControl members into your movie. In the main Director menu bar you will now find an item called OSControl under the Insert menu.

If you select one of the listed controls, a new OSControl cast member is created and you are presented with OSControl Properties Dialog.



OSControl Xtra properties
xtras.openspark.com
OSbevelButton
Enabled [✓] Group ID: 0
State: [✓ Off] On
Style: Medium
IconMember:
Align: Center
Text:
Font/Size: SystemLarge 0
Style: Bold, Italic, Underline



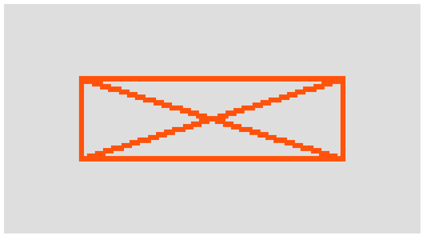
OSControl Xtra properties
xtras.openspark.com
OSbevelButton
Enabled [✓] Group ID: 0
State: [✓ Off] On
Style: Medium
IconMember:
Alignment: Center
Text:
Font/Size: SystemLarge 0
Style: Bold, Italic, Underline
Auto Place

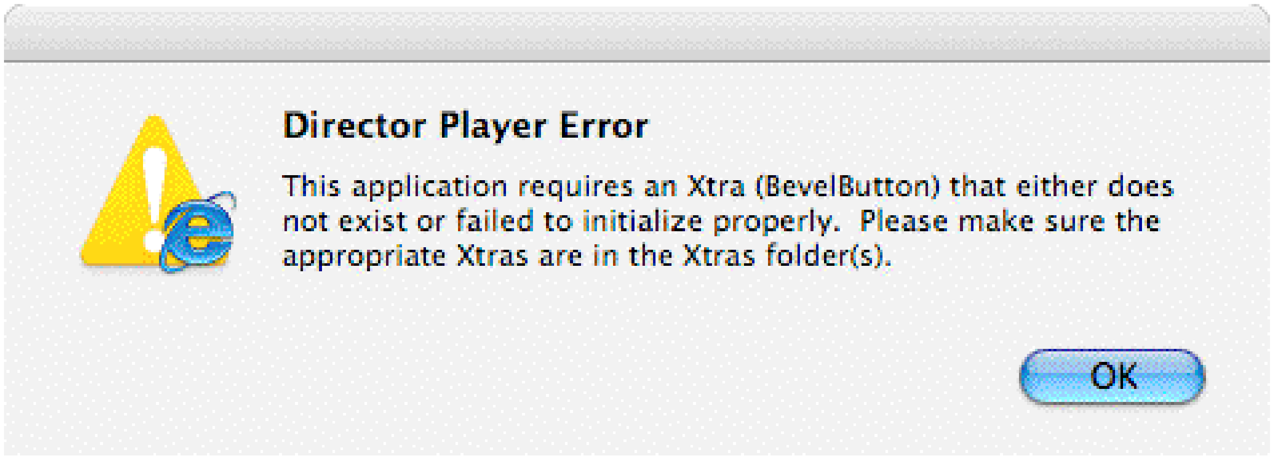
In this dialog window you can set the properties of your newly created OSControl cast member. You can also check the Auto Place checkbox should you want your newly created member placed on the stage. This will place the newly created cast member in an available sprite channel in the current frame. The sprite will only fill a single cell: the Span Duration sprite preference settings will be ignored. The Auto Place setting persists for other OSControl members that are subsequently created. There are other ways of setting OSControl properties all of which are described in the section Working with OSControl.

Of course not every OSControl member type has the same properties. Some properties are only available on certain OSControl member types. A list of all OSControl properties is given in the section OSControl Properties and Methods.

2. OSControl in Shockwave Movies

The OSControl Xtra has been registered as Shockwave-safe. This means that you can create movies for Shockwave, using the OSControl Xtra. However, you need to ensure that end-users have a copy of the OSControl Xtra installed on their machines, otherwise your OSControl sprites will appear as red rectangles containing a red diagonal cross and a warning alert will appear, for each member type, indicating that there is an Xtra missing.





Director Player Error

This application requires an Xtra (BevelButton) that either does not exist or failed to initialize properly. Please make sure the appropriate Xtras are in the Xtras folder(s).

To do this, you need to:

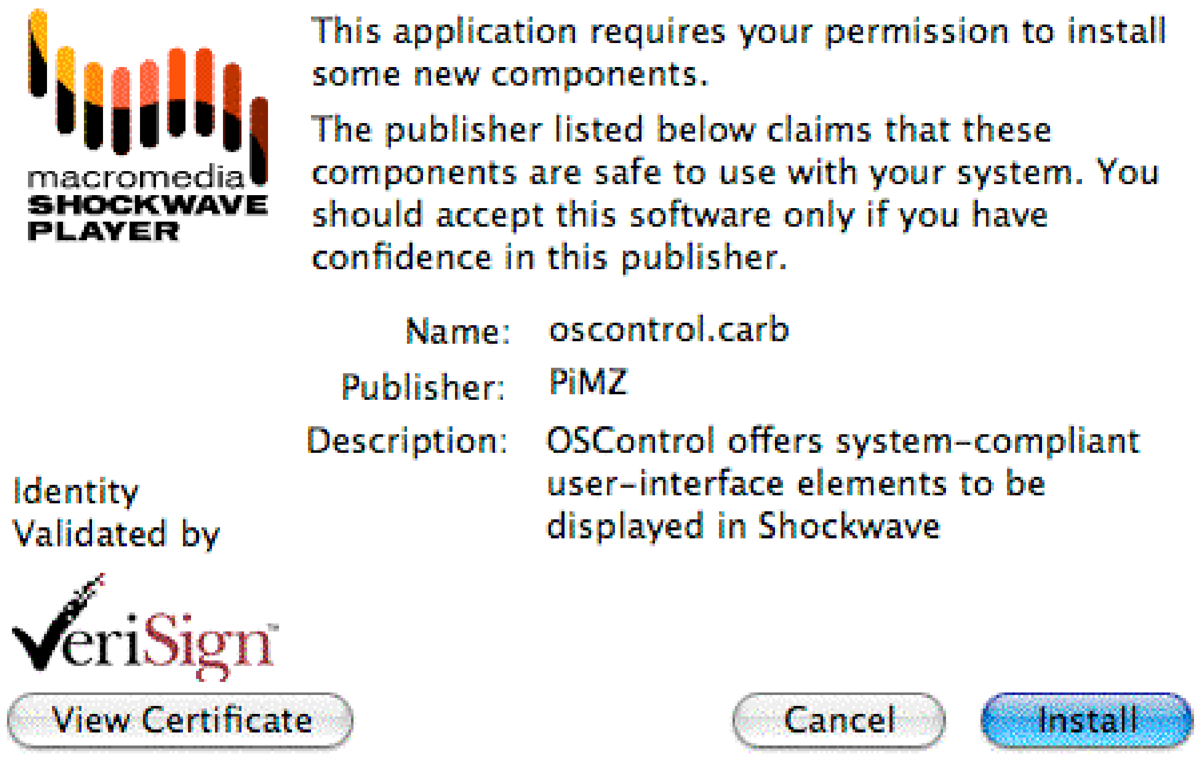
1. Provide the OSControl auto-download package on your site
2. Set your movies up so that they can access the auto-download package if needed.

You need to have an Internet connection open when you perform the second step.

Once you have done this, the first time users encounter a movie of yours containing OSControl members, they will see a dialog suggesting that they install the OSControl Xtra. There is a small chance that they may refuse to install the Xtra. In this case, your content will not appear as you intended, so you may wish to direct the user to an alternative movie.

If you have registered the Xtra you will have received information on where to find the auto-download packages.

Please Note: in previous versions of the xtra, the file for Mac OS X was called "OSControl Xtra Carbon". To ensure that the techniques described below function for both Mac OS X and Mac Classic, please ensure that the xtras for both Macintosh platforms are called "OSControl Xtra".


This application requires your permission to install some new components.

The publisher listed below claims that these components are safe to use with your system. You should accept this software only if you have confidence in this publisher.

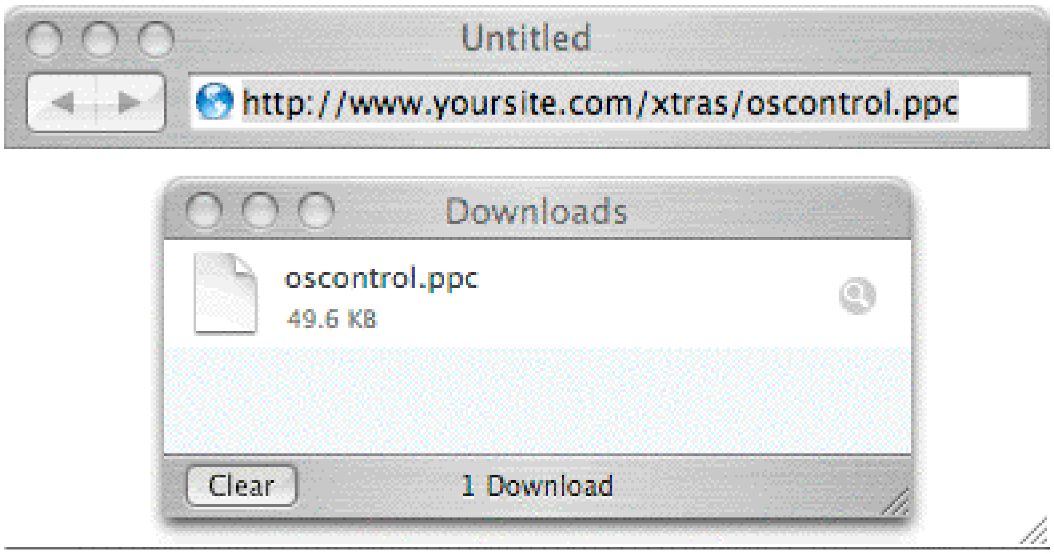
Name: oscontrol.carb
Publisher: PiMZ
Description: OSControl offers system-compliant user-interface elements to be displayed in Shockwave

2.1 Providing an auto-download package

To create movies which will automatically download the OSControl Xtra, you first need to install the auto-download package on your web site. The package consists of three files:

- oscontrol.ppc for Mac OS 9.2 and earlier
- oscontrol.carb for Mac OS X and later
- oscontrol.w32 for Windows

You should place these files together in the same folder on your website. In the example that follows, I will consider that they are in the folder <http://www.yoursite.com/xtras/>. To check that you have the correct location for the three files, try downloading each of them, using your browser.



2.2 Telling your movies where to find the download package

You now need to tell Director where to find the download package, so that Director can incorporate this information in your movies, if required. To do this, you need to edit the xtrainfo.txt file, which you will find in the same folder as your copy of Director.

In Director MX and earlier versions, the xtrainfo.txt file lives in the same folder as the Director application itself. In Director MX 2004, you need to look inside the Configurations folder.

Note: in Mac OS X, do not attempt to edit the xtrainfo.txt file with TextEdit. This Mac OS X application will use the Linux Line Feed character for line breaks, and will render the file illegible for Director. Use the Classic application SimpleText, or any other application that can save the file in a plain text format, using Carriage Return characters for the line breaks.

Open the xtrainfo.txt file, and add the following text:

```
[#namePPC:"OSControl Xtra", #nameW32:"OSControl Xtra.x32",
#package:"http://www.yoursite.com/xtras/oscontrol",
#info:"http://xtras.openspark.com/index.php?page=OSC_home"]
```

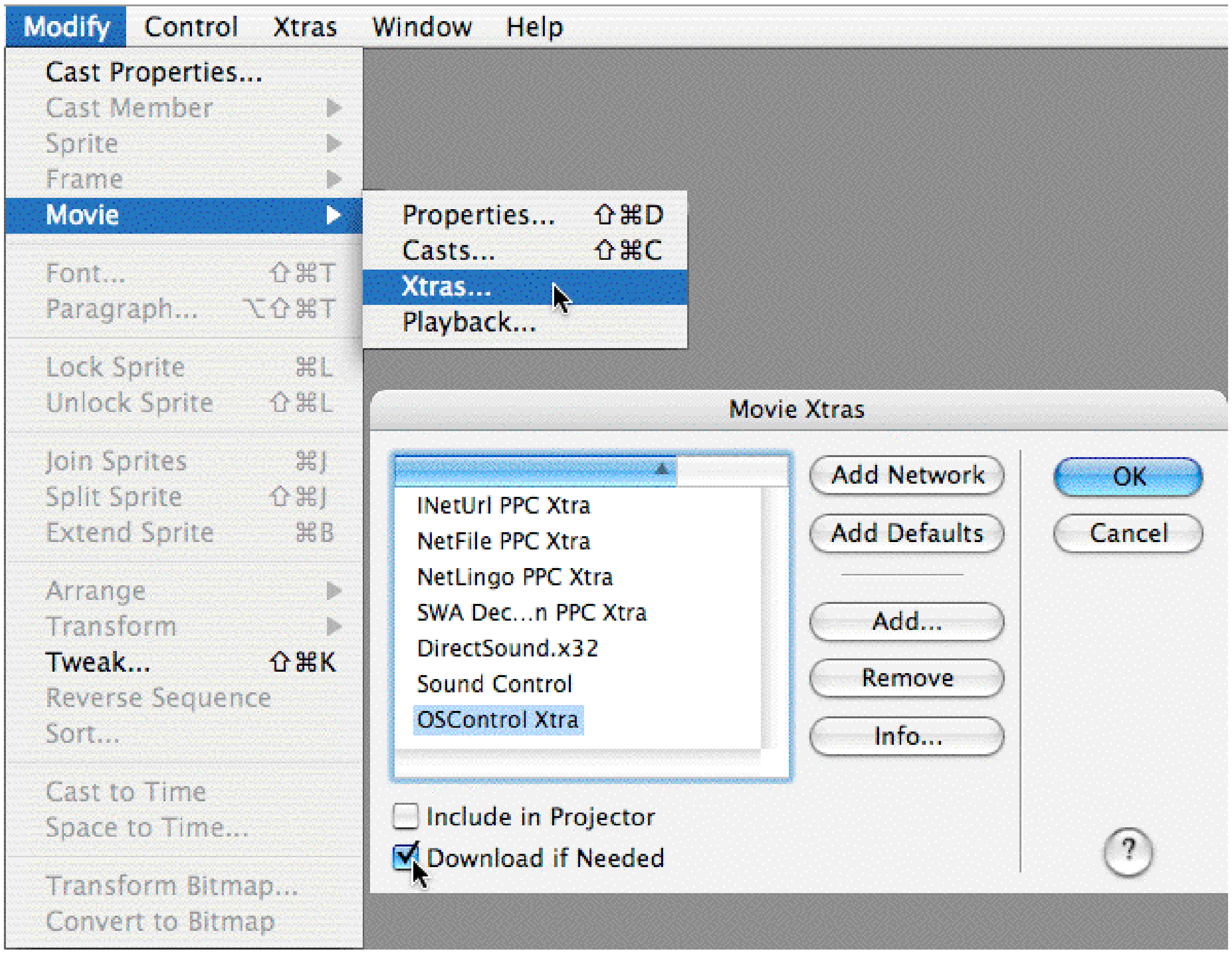
Although this list appears on three separate lines, please ensure that there are no line breaks in the list itself . Director needs to be able to convert the single line of text into a Lingo list, so any line break characters will make it unusable.

What this list tells Director is this: If a movie needs an xtra named "OSControl Xtra" (on Macintosh) or "OSControl Xtra.x32" (on Windows), it can download the necessary xtra from <http://www.yoursite.com/xtras/>. Inside that folder will be files with the name "oscontrol" followed by the extension appropriate for the platform: .ppc, .carb or .w32. You don't need to mention these extensions in the list, since Director will know which extension to look for, depending on the platform the movie is running in.

The #info entry is optional. The #namePPC and #nameW32 entries are also used to tell Director which xtras may be embedded into a projector when you publish the movie.

2.3 Setting a movie to download the OSControl Xtra automatically

Including this information in the xtrainfo.txt file is not enough to make your movies download the OSControl Xtra automatically. You need to tell each movie explicitly that it should download the xtra if needed. To do this, open the movie, then choose Modify|Movie|Xtras.

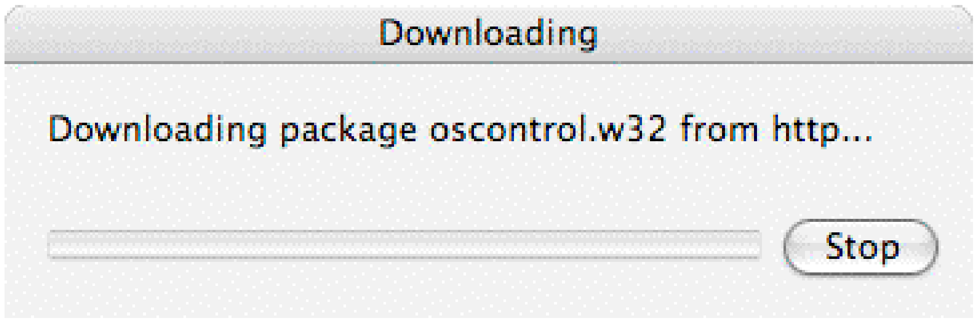


INetUrl PPC Xtra
NetFile PPC Xtra
NetLingo PPC Xtra
SWA Dec...n PPC Xtra
DirectSound.x32
Sound Control
OSControl Xtra

☐ Include in Projector
☑ Download if Needed

The OSControl Xtra will appear in the list of xtras if there is already an OSControl member in the current movie. If not, you can click on the Add... button, and select the OSControl Xtra from the list that is shown.

Select the OSControl Xtra in the Movie Xtras dialog. The Download if Needed button will probably be unchecked. Before you check it, ensure that you are connected to the Internet, as Director will need to link to <http://www.yoursite.com/xtras/> in order to check that the packages are available. If your machine is not connected to the Internet, or the package is not available at the URL you entered in the xtrainfo.txt file, then you will not be able to check the Download if Needed button. If all is well, you should see a series of brief progress windows, indicating that the package for each platform can indeed be downloaded.



Downloading package oscontrol.w32 from http...

Once this has happened, the Download if Need box will be checked. Click OK in the Movie Xtras window to dismiss it.

2.4 Checking that the auto-download works

To check that your movie will correctly trigger the auto-download process, you need:

- to have Shockwave installed for your browser
- to have removed any existing downloads of the OSControl Xtra from the Shockwave Xtras folder.

To install the latest version of Shockwave, visit the Shockwave Download Center at <http://www.macromedia.com/shockwave/download/download.cgi?>. The location of an existing download of the OSControl Xtra, depends on the platform you are working on. Here are some examples:

Windows XP:	C:\WINDOWS\system32\Macromed\Shockwave 10\Xtras\download\
Mac OS 9.2:	Mac HD:System Folder:Extensions:Macromedia:Shockwave 8:Xtras:download:
Mac OS 10.3.5:	Mac HD:users:<username>:Library:Application Support:Macromedia:Shockwave 10:Xtras:download:

Inside the download folder, you may find a folder named PiMZ. If so, the OSControl Xtra will be inside this. Move the entire folder to (say) your desktop, so that Shockwave no longer has any access to the xtra. It's a good idea to relaunch your browser, so that you can be sure that the OSControl Xtra is cleared from the browser's memory.

In the movie you were working on, add an OSControl member (if you have not already done so), and save it. Check that you have an open Internet connection, and open a browser window. In the Finder (Mac) or Explorer (Windows), locate your movie's icon, and drag it onto the browser window.

If you have followed all the preceding steps correctly, you should now see the VeriSign alert. Click Install, and the OSControl Xtra should be downloaded. Any OSControl sprites should now appear correctly in your movie.

3. OSControl Properties and Methods

Currently the OSControl Xtra provides the following platform specific controls. Those marked with an asterisk are new to version 2.0 or have improved features:

Control name	Control Description
OSbevelbutton*	Rectangular button with optional text label and image.
OSbox	Rectangular shape used to draw panes and outlines, with optional text label
OScheckbox	Checkbox control with an optional text label.
OSlittlearrows	Arrows to alter the value of a numeric entry field with the mouse.
OSmenu*	Menu control to display a selection of choices in a small area on the stage.
OSpopupmenu*	Popup menu control to display a selection of choices in a small area on the stage.
OSprogressbar	Progress bar for displaying the progress of a process or a calculation.
OSpushbutton	A standard push button.
OSradiobutton	A radio button control with an optional text label.
OSscrollbar*	A control to scroll text or images that is too large to fit in a given area.
OSslider	A slider control to select an integer value from a given range of values.
OSstatictext	Static text control for a label or other text – text not editable during playback.
OStabs*	Tab control.

The remainder of this section describes all the properties and methods of each OSControl.

3.1 Supported Sprite Properties

Many of the standard properties can be applied to sprites containing OSControl cast members but some are not supported. The following two tables list supported and unsupported sprite properties.

Supported Sprite Properties				
loc	loch	locV	rect	width
height	top	left	bottom	right
constraint	constrainH()	constrainV()	sprite…intersects	sprite…within
trails	visible	moveableSprite	tweened	

NB: rect, width, and height will have no effect with many OSControl sprites on Mac

Unsupported Sprite Properties				
ink	blend	flipH	flipV	skew

rotation	quad	backColor	bgColor	foreColor

3.2 OSbevelbutton

An OSbevelbutton is a rectangular button with text and an optional image. Examples of similar controls are the buttons in the Director toolbar and tool palette.

Properties

Property name	**Access**	**Type**	**Property Description**
alignment	R/W	Symbol	Alignment of iconMember with respect to the button text. Values: #left, #right, #center, #full. A value of #center places the icon above the button text, and may distort the icon vertically, or hide it entirely if there is not enough room for both icon and text to appear. A value of #right will show the text at its full width, and may distort the icon horizontally, or hide it entirely. The values #full and #left will show the icon at its full width and may clip or squeeze the text in order to do so. Both #right and #full place the icon to the right of the text: #right places the icon beside the text, #full places the icon fully to the right. See the Notes & Tips below for illustrations.
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
font	R/W	String	Button text font. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontColor	R	RGB	Color of the text font at runtime (means of access to system setting).
fontSize	R/W	Integer	Button text font size.
fontStyle	R/W	List of symbols	Button text style. Any combination of bold, italic and underline can be set. Values: #plain, #bold, #italic, #underline.
groupId	R/W	Integer	Group to which this control belongs.
iconMember	R/W	String/Integer	Optional reference to the bitmap member to be used as the button icon (member name or number).
selectionColor	R	RGB	Color of the selection rectangle (means of access to system setting).
state	R/W	Symbol	State of button directly linked to *value.* Values: #on, #off.
style	R/W	Symbol	Visual 3d height of the button bevel. This is ignored by Mac OS X. Values: #small, #medium or #large.
text	R/W	String	Button text. The button does not scale to accommodate the text.

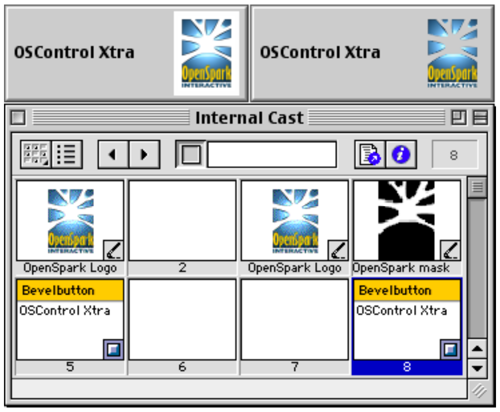
type	R	Symbol	Member only property set to #OSbevelbutton.
value	R/W	Integer	State of button directly linked to *state.* Values: 1, 0.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
click()	Sprite only method that emulates a mouse click on the control and sends mouseDown and mouseUp events to the sprite's behavior. Note: the 'me' parameter received by the behavior will point to the sprite, not the behavior instance.
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Notes & Tips

The optional *iconMember* specifies a graphic cast member that is placed on the button as an icon image. To mask out the background of the icon, you can use a 1-, 2-, 4- or 8-bit bitmap with a #grayscale palette. This must be placed in the cast member slot immediately following the icon member. The mask and the icon member are aligned with their respective registration points.



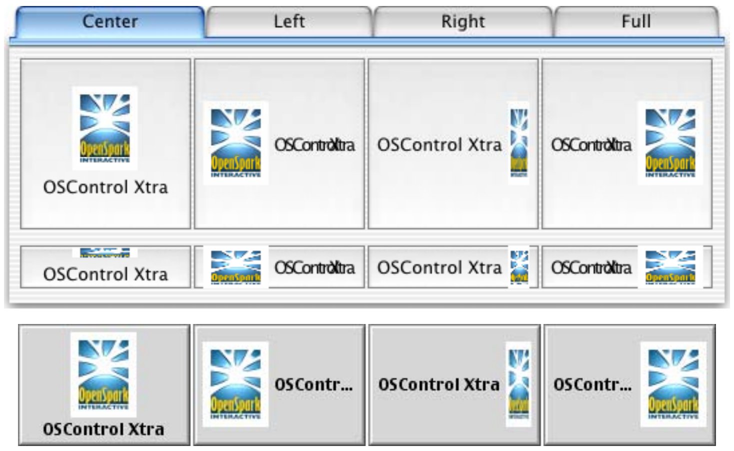
Inserting pixels to either side of the icon member changes the position on the button. Inserted pixels are outside the masked area and therefore invisible.

IconMember Linkage: Assigning an *iconMember* to a bevelbutton member is slightly different from using a Lingo call or the init_iconMember behavior intializer for a bevelbutton sprite.

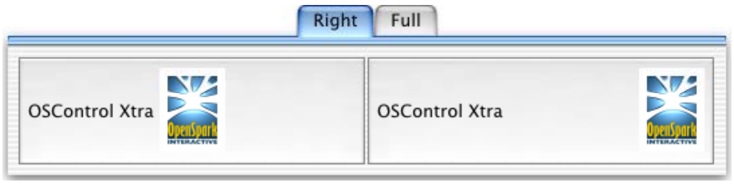
Setting the *iconMember* member property of a bevelbutton creates a strong link between the bevelbutton and the specified bitmap cast member. You can freely move, rename and modify the *iconMember* without the bevelbutton losing track of it.

If you set the *iconMember* as a sprite property either through a Lingo call or through the init_iconMember behavior initializer, the OSControl Xtra will parse your input once you have set it (or if the sprite span starts), but will not keep track of any changes that are applied to the icon member afterwards.

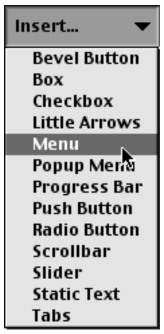
Take care to ensure that the button is big enough to contain both the text and the icon. The illustration below shows how the text and icon adjust to the button size, depending on the value of the #alignment property and the current platform.



The #full value for alignment places the icon to the far right of the button.



It is especially suitable for buttons that serve as an anchor point for popup menus. In the illustration below, the icon is a downward-pointing arrow:



3.3 OSbox

An OSbox is a rectangular shape used to draw panes and outlines. Examples of similar controls are the beveled squares in the Director Cast windows.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
filled	R/W	Boolean	Determines if the box is filled or not.
font	R/W	String	Box text font. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontColor	R	RGB	Color of the text font at runtime (means of access to system setting).
fontSize	R/W	Integer	Box text font size.
fontStyle	R/W	List of symbols	Box text style. Any combination of bold, italic and underline can be set. Values: #plain, #bold, #italic, #underline.
groupId	R/W	Integer	Group to which this control belongs.
selectionColor	R	RGB	Color of the selection rectangle (means of access to system setting).
style	R/W	Symbol	Style of the border. Plain is no border and should only be used if *filled* is set. On the Macintosh, PrimaryGroup box and SecondaryGroup box should be used appropriately to comply with Apple's HIG. On Windows, PrimaryGroup is the grouping box where SecondaryGroup is the border of a Status field. FocusRing draws a colored ring around an active input field. Values: #plain, #bevel, #primaryGroup, #secondaryGroup, #focusRing
text	R/W	String	Text of the label at the top of the box. The box outline is reduced to accommodate the label. The box does not scale to accommodate long text. Label text is only visible on boxes with *style* set to #primaryGroup or #secondaryGroup.
type	R	Symbol	Member only property set to #OSbox.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Notes & Tips

To create a backdrop using the System color scheme, use an OSbox with *style* of #plain and with the *filled* property set to TRUE in sprite 1. Stretch this sprite to fill the entire Stage.

To improve the performance of your program, set the *filled* property to FALSE wherever you have more than one box overlapping. If the *filled* property of overlapping boxes is set to TRUE, the same pixels will be drawn to the screen multiple times, unnecessarily.

3.4 OScheckbox

An OScheckbox is a checkbox control with an optional text label. Examples of similar controls can be found in the Director General Preferences Dialog.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
focusable	R/W	Boolean	Defines if the OScheckbox can have focus on a Windows machine.
font	R/W	String	Label text font. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontColor	R	RGB	Color of the text font at runtime (means of access to system setting).
fontSize	R/W	Integer	Label text font size.
fontStyle	R/W	List of symbols	Label text style. Any combination of bold, italic and underline can be set. Values: #plain, #bold, #italic, #underline.
groupId	R/W	Integer	Group to which this control belongs.
selectionColor	R	RGB	Color of the selection rectangle (means of access to system setting).
state	R/W	Symbol	The checked, unchecked or mixed (grayed check) state of the checkbox. Directly linked to *value*. Values: #off, #on, #mixed.
text	R/W	String	Text of the label for the checkbox. The checkbox does not scale to accommodate long text.
type	R	Symbol	Member only property set to #OScheckbox.
value	R/W	Integer	Checked, unchecked or mixed (grayed check) state of the checkbox. Directly linked to *state*. Values: 0 unchecked, 1 checked, 2 mixed.
version	R	String	Version number of the installed OSControl Xtra.

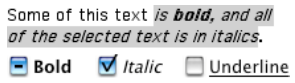
Methods

Method name	Method Description
click()	Sprite only method that sends mouseDown and mouseUp events to your sprite's behavior.
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Notes & Tips

The checkbox has a fixed size on both Macintosh and Windows. The sprite rect should be large enough to show both the checkbox and the label text. When you click a checkbox its state changes automatically.

Use a checkbox with a *state* of #mixed to indicate that the attribute associated with the checkbox is TRUE for only part of a selection. In the illustration below, the attribute Bold is only true for part of the selected text, so the Bold button is shown in a #mixed state.


Some of this text is bold, and all of the selected text is in italics.
Bold
Italic
Underline

On Windows XP, checkboxes do not appear with a #mixed state.

3.5 OSlittlearrows

OSlittlearrows are generally used to change the value of a decimal entry field with the mouse. These are analogous to the *spin* controls in Visual Basic. An example of a little arrows control can be found in Director's Text inspector.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
groupId	R/W	Integer	Group to which this control belongs.
type	R	Symbol	Member only property set to #OSlittlearrows.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
click(#up/ #down)	Sprite only method that sends mouseDown and mouseUp events to your sprite's behavior.
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Events

Event name	Event Description
update(spriteRef, partClicked)	Sent to any behaviors on the sprite when the user clicks on one of the little arrows, and repeatedly from then on until the user releases the mouse. The *partClicked* parameter can take the values *#up* or *#down*

Notes & Tips

The size of the little arrows is fixed on the Macintosh to 13 x 23 pixels. On Windows the little arrows will fill the whole sprite rectangle. So make sure the sprite rectangle is the right size for your little arrows on the Windows platform.

Clicking the little arrows control sends an #update event to your sprite behavior…

```
on update me, partClicked
```

… where the *partClicked* parameter will be either `#up` or `#down` .

3.6 OSmenu

An OSmenu control can be used to display a selection of choices in a small area on the stage. Examples of similar controls are the contextual menus found throughout Director that are displayed with right mouse clicks on Windows and Control clicks on Macs. The difference with the OSpopupmenu is that it does not have a visible part when the control is not expanded, and that on Windows there is no scrollbar. It can be layered on top of any other elements on the screen.

OSmenu members can be used to display hierarchical menus. (See section Defining the Contents of an OSmenu below for details).

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
font	R/W	String	Menu item text font. Font settings are ignored on Windows. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontSize	R/W	Integer	Menu item font size.
groupId	R/W	Integer	Group to which this control belongs.
itemList	R/W	List	Runtime property used to define the menu items.
itemString	R/W	String	Runtime property used to define the menu items.
menuLocH	R/W	Integer	Horizontal menu location offset measured from the top left of the menu sprite.
menuLocV	R/W	Integer	Vertical menu location offset measured from the top left of the menu sprite. If both *menuLocH* and *menuLocV* are 0, the menu is displayed at the mouse position.
menuMember	R/W	String/Integer	Name or number of the Field cast member that contains the menu items. (NB: for performance this member must be a #field member, not a #text member.) The menu items in this field can contain the following special characters: | Any Lingo code or handler after this character is executed when the item is chosen. None of the text between the pipe character and the end of the line will appear in the menu. * This is the selected item; when expanded this item is selected !v Places a selection arrow in front of the item (Disables the item (- Draws a separator line. On Mac OS X, separators appear as white space. <I Item in italics on Mac; ignored by Windows <B Item in bold on Mac; ignored by Windows <U Item in underline on Mac; ignored by Windows <S Item in shadow on Mac; ignored by Windows <O Item in outline on Mac; ignored by Windows
mouseButton	R/W	Symbol	Determines which mouse button activates the menu. Values: #left | #right | #none
rightMouseDown	R/W	Boolean	This property is obsolete. It is included only for backwards compatibility with previous versions of the OSControl Xtra. Please use the *mouseButton* property instead.
type	R	Symbol	Member only property set to #OSmenu.

version	R	String	Version number of the installed OSControl Xtra.

Methods

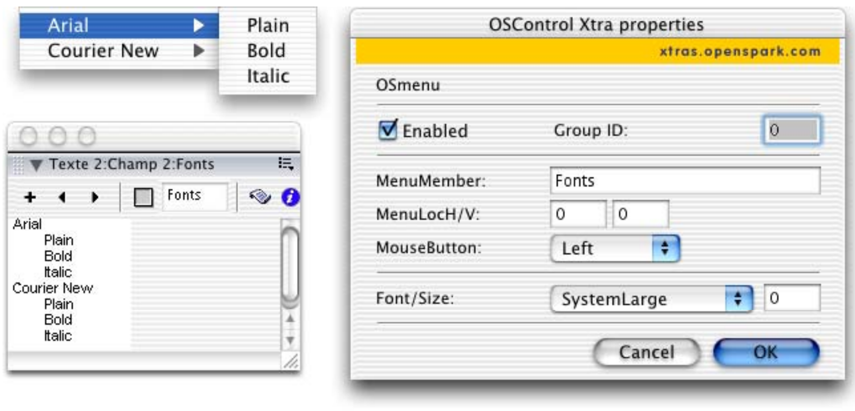
Method name	Method Description
click(optional point)	Sprite only method. Opens the OSmenu at the mouseLoc, or at the point on the Stage defined by the optional point parameter.
getItem(path)	Sprite only method. Returns a property list defining the chosen item. The *path* parameter can be an integer or a list of integers. It defines which item of which submenu is being referred to.
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
setItem(path, value)	Sprite only method. Resets the string of the item or, if you specify a number greater than the number of items, appends a new item. The *path* parameter can be an integer or a list of integers. It defines which item of which submenu is being referred to.
showProps()	Displays the properties for this control in the message window.
updateControl()	Sprite only method. Resets the menu item list. Call this method after changing the menu items field.

Events

Event name	Event Description
update(spriteRef,itemPath,itemString)	Sent to any behaviors on the sprite when the user selects an item, unless that item has a Lingo expression associated with it. The *itemPath* parameter will be an integer or a list of integers, and will describe the path to the selected item. The *itemString* parameter will be the text of the selected item.
mouseUp()	Sent to any behaviors on the sprite when the user releases the mouse without making any selection. (Note: no *mouseUpOutside* events are sent: *mouseUp* is used instead).

Defining the Contents of an OSmenu

The menu shown in the illustration below can be defined in a number of ways. The simplest is to set the *menuMember* of the OSmenu member to a field, as shown. Use tabulation characters to indicate which lines in the field are part of a submenu.



```
Arial
	Plain
	Bold
	Italic
Courier New
	Plain
	Bold
	Italic
```

[✓ Enabled] Group ID: 0

MenuMember: Fonts

MenuLocH/V: 0 0

MouseButton: Left

Font/Size: SystemLarge 0

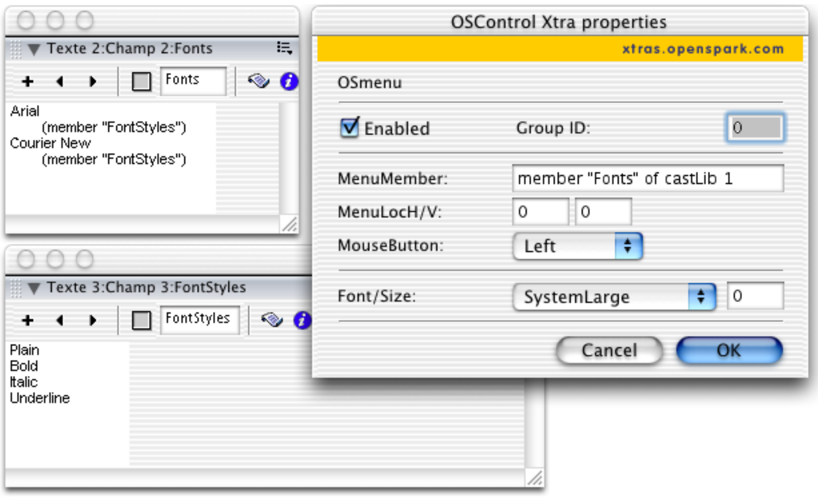
itemString

To achieve exactly the same result using Lingo only, you can set the *itemString* property of the sprite. Here's a behavior that you can drop on an OSmenu sprite. It defines the menu on the fly when the user clicks on the menu sprite, just before the menu itself appears.

```
on mouseDown(me)
  sprite(me.spriteNum).itemString = \
"Arial"&RETURN& \
TAB&"Plain"&RETURN& \
TAB&"Bold"&RETURN& \
TAB&"Italic"&RETURN& \
"Courier New"&RETURN& \
TAB&"Plain"&RETURN& \
TAB&"Bold"&RETURN& \
TAB&"Italic"
end mouseDown
```

Fields for Submenus

You can also define submenus in their own separate field, and then refer to the submenu fields in the field that holds the main definition. This is especially useful if many menu items share the same submenu. Note that the member references in the Fonts field appear in parentheses, to distinguish them from ordinary menu items.


Texte 2:Champ 2:Fonts

Fonts

Arial
	(member "FontStyles")
Courier New
	(member "FontStyles")

Texte 3:Champ 3:FontStyles

FontStyles

Plain
Bold
Italic
Underline

OSControl Xtra properties

xtras.openspark.com

OSmenu

✓ Enabled	Group ID: 0
MenuMember:	member "Fonts" of castLib 1
MenuLocH/V:	0 0
MouseButton:	Left
Font/Size:	SystemLarge 0

Menu Item Properties

The definition field can include more than just the name of the menu items. You can add meta data to each line that defines a menu item. A field with the following text would create a menu with the Italic item in italics, and disabled:

```
Plain
Bold
Italic<I(
```

The order of the meta data is not important. The <I italic tag and the (disable tag could be placed before the item name, with the same effect:

```
Plain
Bold
<I(Italic
```

Note: You can also define the Lingo to be executed when a given menu item is chosen. This meta data is treated in detail in the following section: Acting on a Menu Selection. Lingo instructions must be placed after the item name and all other meta data.

setItem()

You can add or modify individual items on the fly using the *setItem()* method. This is a sprite-only method; it does not function with members. Here is code that has the same effect on the Courier New | Italic item in the menu shown above:

```
sprite(1).setItem([2, 3], [#enabled: 0, #fontStyle: [#italic]])
```

The following lines would add an item with a submenu to the main menu:

```
tFontStyles = []
tFontStyles.append([#text: "Plain"])
tFontStyles.append([#text: "Bold", #fontStyle: [#bold]])
tFontStyles.append([#text: "Italic", #fontStyle: [#italic]])
sprite(1).setItem(3, [#text: "Times", #submenu: tFontStyles])
```

To remove the submenu, you could use:

```
sprite(1).setItem(3, [#submenu: 0])
```

Use the *getItem()* method to see a list of all the properties that you can set for a given menu item:

```
put sprite(1).getItem(3)
-- [#text: "Times", #enabled: 1, #checked: 0, #lingo: 0, #fontStyle: [#plain],
#subMenu: 0]
```

itemList

You can define an entire menu, including customized values for each item, by setting the *itemList* property for the sprite. Note that both *itemList* and *itemString* are sprite-only properties: the resulting menu display will not be saved to the member when the movie halts.

```
on mouseDown(me)
  tItemList  = [] -- will define the entire menu

  tFontStyle = [] -- will define the fontStyle submenu

  tItem      = [#text: "Plain"]
  tFontStyle.append(tItem)

  tItem      = [#text: "Bold"]
  tFontStyle.append(tItem)

  tItem      = [#text: "Italic", #fontStyle:[#italic], #enabled: 0]
  tFontStyle.append(tItem)

  -- Add main menu items, each with the same fontStyle submenu
  tItem = [#text: "Arial", #submenu: tFontStyle]
  tItemList.append(tItem)

  tItem = [#text: "Courier New", #submenu: tFontStyle]
  tItemList.append(tItem)

  -- Set the contents of the entire menu just before it appears
  sprite(me.spriteNum).itemList = tItemList
end mouseDown
```

Note that default values will be used wherever the value of a given property is not set explicitly. You only need to set the values that differ from the default.

Acting on a Menu Selection

If you are using a *menuMember* field to define the menu, you can include a Lingo instruction for each menu item. This instruction will be executed when the menu item is chosen. Suppose you use a field with the following text to create a menu:

```
Open...|goFileManager.openFile(xtra("FileIO").new().displayOpen())
(-
Quit|halt
```

If the user selects the Quit item, the movie will halt. In a projector, this will also make the application quit. The instruction does not need to be a single Lingo keyword. It can be any expression that can be evaluated by the Lingo command *do*. This includes making calls to Director Xtras, or expressions that use global variables, as the above example shows. Globals do not have to be explicitly declared.

Note: If you want to test the above *menuMember*, create a movie script named “File Manager”, with the following scriptText:

```
-- Movie Script "File Manager"

on startMovie()
```

```
  global goFileManager
  goFileManager = script("File Manager").new()
end


on OpenFile(me, aFilePath)
  put #openFile, aFilePath
end
```

Using *setItem()* to Define the Lingo to Execute

You can change the Lingo command on the fly using the *setItem()* method you saw earlier. This instruction would change the Lingo command for the Open... item in the menu we've just been looking at:

```
sprite(1).setItem(1, [#lingo: "OpenFile"])
```

update()

If you do not specify Lingo commands in your *menuMember*, selecting a menu item from this control will send update calls to your sprite behavior…

```
on update me, itemPath, itemString
```

… where *itemPath* indicates the path to the chosen item and *itemString* is the item content. If the chosen item is in the main menu, *itemPath* will be an integer. (This retains backward compatibility with previous versions of the OSControl Xtra). If an item is chosen from a submenu, *itemPath* will be a linear list. An *itemPath* of [2, 3] would indicate that the chosen item was third item in the submenu belonging to item 2 of the main menu. In the Font menu we looked at earlier, this would correspond to Courier New | Italic.

Note: The *update()* handler will not be called if a Lingo string is provided for the selected menu item.

If the user cancels the menu without selecting an item, your sprite behavior will be sent a *mouseUp* message. No *mouseUp* message is sent when an item is selected.

Notes & Tips

Items with submenus cannot be selected. The #lingo property for these items is ignored and a *mouseUp* event is sent instead of an *update* event.

The OSmenu can be set to respond to either the left or right mouse button. This is useful to make contextual and shortcut menus. Note that to use the right mouse button in Shockwave, you have to disable the SW contextual menu from within Director (in version 7: Modify > Movie > Playback, in version 8+ in the Publish Settings) or in the Shockwave object/embed tag.

The *mouseButton* property takes three values: *#left*, *#right* and *#none*. If you use *#none*, then you must explicitly send a *click()* event to the sprite to open the menu. The menu will then remain open until the user clicks the mouse.

If you use an OSmenu with fixed location, make sure you do not have the menu pop up in the same place the click occurred otherwise the *mouseUp* that happens after the initial *mouseDown* that invoked the menu would immediately close the menu.

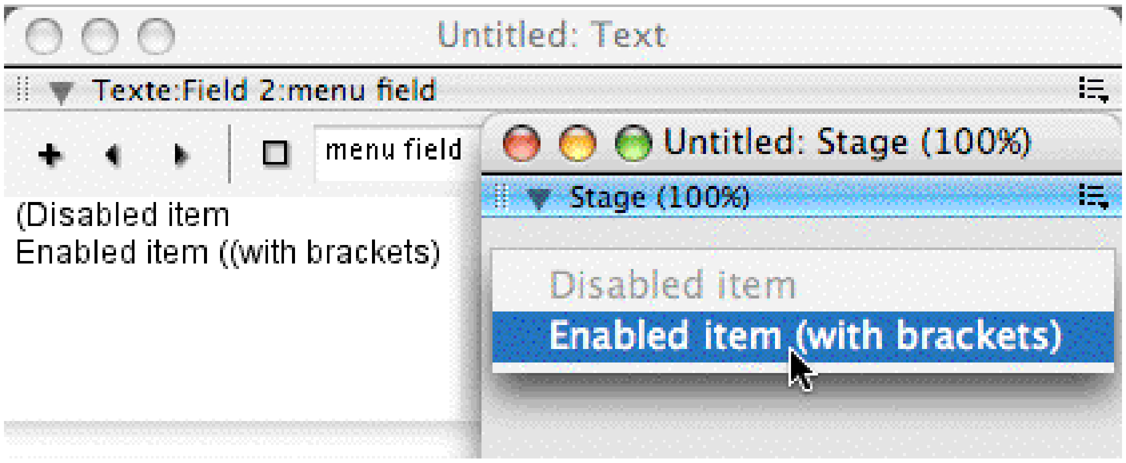
When the menu is expanded, the expanded part is drawn Direct-To-Stage and will be on top of everything else on the screen. Because the Operating System takes over at this time, animations and events inside director will halt completely. This is the normal OS behavior.

If you want a contextual (Mac) or shortcut (Windows) menu, make sure you set the *mouseButton* property to #right. These types of menus pop-up at each click, so using a left mouse button for them may prove to be inconvenient. Also you must ensure that the OSmenu sprite is on top of everything that the menu relates to, so that it catches all right mouse clicks. If the *mouseButton* is set to #right, an OSmenu sprite will pass left mouse clicks down to the appropriate sprite.

Special characters in menuMember fields

The following characters have a special meaning when used in menuMember fields or for the itemString property. Any of these special characters can be escaped by using it twice (see image below).

;	(semicolon):	menu item terminator (equal to a line ending)
(	(left bracket):	disables menu item (OSmenu only)
(-	(left bracket-hyphen):	separator (OSmenu only)
*	(asterisk):	set selection of OSpopupMenu
!v	(exclammation mark):	mark menu item (OSmenu only)
|	(pipe):	all that follows this character is considered Lingo code
<I	(angle bracket left-I)	make fontstyle of this item italic (OSmenu on Mac only)
<B	(angle bracket left-B)	make fontstyle of this item bold (OSmenu on Mac only)
<U	(angle bracket left-U)	make fontstyle of this item underlined (OSmenu on Mac only)



(Disabled item
Enabled item ((with brackets)

Platform Differences

This control is a popup menu on the Macintosh and a shortcut menu on Windows. On Macintosh you can choose any font but on Windows the font is always the systemLarge font.

On Mac OS do not set the emulateMultiButtonMouse property of your movie as this will disable right-mouse clicks on OSmenu sprites. Mac users will be able to use the standard key combination of Ctrl-click to display an OSmenu where the *mouseButton* is set to #right.

3.7 OSpopupmenu

An OSpopupmenu control can be used to quickly display a selection of choices in a small area on the stage. An example is the ink selector in the Director Score Window.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
focusable	R/W	Boolean	Defines if the popup menu can have focus on Windows.
font	R/W	String	Menu item text font. Font settings are ignored on Windows. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontColor	R	RGB	Color of the text font at runtime (means of access to system setting).
fontSize	R/W	Integer	Menu item text font size
groupId	R/W	Integer	Group to which this control belongs.
itemList	R/W	List	Runtime property
itemString	R.W	String	Runtime property
maxValue	R/W	Integer	Height of the expanded menu in pixels on Windows. No effect on Macintosh.
menuMember	R/W	String/Integer	Name or number of the Field cast member that contains the menu items. (NB: for performance this member must be a #field member, not a #text member.) The menu items in this field can contain the following special characters: | any Lingo code or handler after this character is executed when the item is chosen * this is the selected item; when expanded this item is selected !v places a selection arrow in front of the item (disables the item (- draws a separator line <I item in italics on Mac; ignored by Windows <B item in bold on Mac; ignored by Windows <U item in underline on Mac; ignored by Windows <S item in shadow on Mac; ignored by Windows <O item in outline on Mac; ignored by Windows
selectionColor	R	RGB	Color of the selection rectangle (means of access to system setting).
text	R/W	String	Text of menu title.
type	R	Symbol	Member only property set to #OSpopupmenu.
value	R/W	Integer	Selected item number. Zero if no selection.

version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
click()	Sprite only method. Opens the OSpopupmenu.
getItem(path)	Sprite only method. Returns a property list defining the chosen item. The *path* parameter should be an integer. It defines which item of the popup menu is being referred to.
setGroupProp	Sets the property of each OSControl in the group.
setItem(path, value)	Sprite only method. Resets the string of the item or, if you specify a number greater than the number of items, appends a new item.
showProps()	Displays the properties for this control in the message window.
updateControl()	Sprite only method to reset the menu item list. This must be called after changing the menu items field. You can pass a string parameter containing a new menu definition. This is more convenient than altering the field member directly because those changes are permanent and modifying a field is rather slow.

Events

Event name	Event Description
update(spriteRef,itemIndex,itemString)	Sent to any behaviors on the sprite when the user selects an item, unless that item has a Lingo expression associated with it. The *itemIndex* parameter will be an integer which indicates the position of the selected item. The *itemString* parameter will be the text of the selected item.
mouseUp()	Sent to any behaviors on the sprite when the user releases the mouse without making any selection. (Note: no *mouseUpOutside* events are sent: *mouseUp* is used instead).

Notes & Tips

To define the contents of an OSpopupmenu, you can use the techniques described above for the OSmenu, with one major difference: OSpopupmenus do not display submenus. The techniques you saw above for acting on a menu selection also apply here. Since there are no submenus, the *itemPath* parameter for the *update()* handler will always be an integer.

When the menu is expanded, the expanded part is drawn Direct-To-Stage and will be on top of everything else on the screen. Because the Operating system takes over at this time, animations and events inside director will halt completely. This is the normal OS behavior.

Special characters in menuMember fields

Please see the notes in the entry for OSmenu members.

The default event on item selection is an *update* …

```
on update me, itemIndex, itemString
```

… where *itemIndex* is the number of the item that was chosen and *itemString* is the item content.

If the user cancels the menu without selecting an item, your sprite behavior will be sent a *mouseUp* event. This is not sent if an item is successfully selected. If the selected item has its own Lingo expression to execute, *update()* will not be called.

Platform differences

Although they are quite different, the Windows Combobox and the Macintosh Popup Button are the closest matching controls for what is generally referred to as a popup-menu. You may not be able to provide a consistent UI by using this control for both platforms. For example, on Windows it is not usual to use a Combobox for navigating between different program parts in the way it is on Macintosh and in web pages. Also a Windows Listbox (the pop-up part of the Combobox) cannot contain checkmarks, disabled items and separator lines. So these will not show up on Windows when you implement these on the Mac. The individual item fontStyle codes are only available on the Mac, and ignored on Windows.

One of the most radical differences is that the Windows Combobox does not support submenus. For cross-platform consistency, the OSpopupmenu does not therefore support submenus on Macintosh either. There are two workarounds for this limitation:

- Master-Slave popup menus
- Using an OSbevelbutton in association with an OSmenu

Both of these techniques are illustrated in the demonstration movies.

The OSpopupmenu fills the whole sprite rect on both Macintosh and Windows but the height of the listbox on Windows must be set or else it might be 1 pixel high.

3.8 OSprogressbar

An OSprogressbar control is commonly used for displaying the progress of a process (e.g. downloading) or a calculation in the active window or a background window. Examples of OSprogressbar are in the Director Save Progress window.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
groupId	R/W	Integer	Group to which this control belongs.
maxValue	R/W	Integer	Scale of the progress bar with plain *style*. With *style* set to barberpole, *maxValue* has no effect.
style	R/W	Symbol	Style of the progress bar. The plain style is used when you know how long the process will take otherwise use the barberpole style. This shows a moving barber-pole pattern. NB: moving the barber-pole pattern is done using Lingo. Values: #plain, #barberpole.
type	R	Symbol	Member only property set to #OSprogressbar.
value	R/W	Integer	Value of the progress bar. For the plain *style*, this value defines how much of the bar is shown. For the barberpole *style*, a non-zero value results in the barber-pole moving. Range: 0 – *maxValue*.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Notes & Tips

There is no vertical version of the OSprogressbar.

3.9 OSpushbutton

An OSpushbutton is a normal push button. An example of a similar control is the Cancel button in the Director Save Changes dialogue.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
focusable	R/W	Boolean	Defines if the pushbutton can have focus on Windows.
font	R/W	String	Pushbutton text font. Font settings are ignored on Windows. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontColor	R	RGB	Color of the text font at runtime (means of access to system setting).
fontSize	R/W	Integer	Pushbutton text font size.
fontStyle	R/W	List of symbols	Pushbutton text style. Any combination of bold, italic and underline can be set. Values: #plain, #bold, #italic, #underline.
groupId	R/W	Integer	Group to which this control belongs.
selectionColor	R	RGB	Color of the selection rectangle (means of access to system setting).
style	R/W	Symbol	Default or plain where a default pushbutton has a focus ring on Macintosh and a slightly thicker shadow on Windows. A default pushbutton also has initial focus on Windows so you should only have one default button on a page. Values: #plain, #default.
text	R/W	String	Text for the pushbutton. The pushbutton does not scale to accommodate long text.
type	R	Symbol	Member only property set to #OSpushbutton.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
click()	Sprite only method that sends mouseDown and mouseUp events to any behaviors on the sprite.
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Notes & Tips

OSpushbuttons with a *style* of #plain always draw a few pixels smaller than the sprite rect to make room for the ring around the button, in case it is set to be a default button on Classic Mac OS. Otherwise the button would jump around if you changed the style to default. On Macintosh an OSpushbutton's height is defined in the OS; on Windows it can have any size, and will always occupy almost the complete sprite rect. A default OSpushbutton can lose its focus on Windows when another control gets focus.

3.10 OSradiobutton

An OSradiobutton is a radio button control with an optional text label. Examples of OSradiobutton-like controls can be found in the Director General Preferences Dialog.

Properties

Property name	**Access**	**Type**	**Property Description**
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
focusable	R/W	Boolean	Defines if the radio button can have focus on a Windows.
font	R/W	String	Label text font. Font settings are ignored on Windows. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontColor	R	RGB	Color of the text font at runtime (means of access to system setting).
fontSize	R/W	Integer	Label text font size.
fontStyle	R/W	List of symbols	Label text style. Any combination of bold, italic and underline can be set. Values: #plain, #bold, #italic, #underline.
groupId	R/W	Integer	Group to which this control belongs.
selectionColor	R	RGB	Color of the selection rectangle (means of access to system setting).
state	R/W	Symbol	Unselected, selected or mixed state (grayed select circle) of the radio button. The *state* is directly linked to *value*. Values: #off, #on, #mixed.
text	R/W	String	Radio button label text. The radio button does not scale to accommodate long text.
type	R	Symbol	Member only property set to #OSradiobutton
value	R/W	Integer	Unselected, selected or mixed state (grayed select circle) of the radio button. The *value* is directly linked to *state*. Values: 0 off, 1 on, 2 mixed.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
click()	Sprite only method that sends mouseDown and mouseUp events to your sprite's behavior.
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

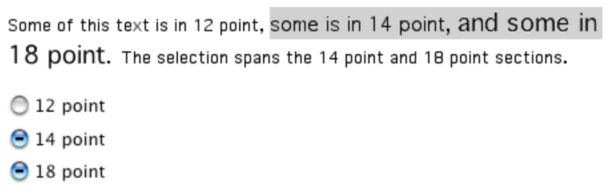
Notes & Tips

Radio buttons represent related choices, but not necessarily opposite ones. These choices are mutually exclusive; the user can only set one button to the on state at any one time.

The radio button of an OSradiobutton has a fixed size on both Macintosh and Windows but the sprite rect should be set large enough to show the radio button and the label text.

Radio buttons are designed to be grouped using *groupId.* When a radio button is pressed, its state will change automatically when the mouse is released and a message is sent to all radio buttons in the group to change their state so there is always only one radio button in a group selected. Make sure that when you enter a screen there is one, and only one, radio button selected.

There is a special case called the mixed state, which shows that a selected range has a variety of items in the on state. For example, a set of radio buttons for selecting font size might have buttons representing 14- and 18-point sizes. If a passage of text with both 14- and 18-point text were selected, both the 14 and 18 buttons would appear in the mixed state. The "one button per set" rule still applies, however; if the user selected the button marked 14, all the text in the selection would change to 14 point, and the mixed state would be cleared.


Some of this text is in 12 point, some is in 14 point, and some in 18 point. The selection spans the 14 point and 18 point sections.

NB: In Windows XP, there is no mixed-state radio button.

3.11 OSscrollbar

An OSscrollbar control is used to scroll text, images or other data that is too large to fit in a fixed space on the stage. Examples of scrollbars are found in almost all Director windows.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
groupId	R/W	Integer	Group to which this control belongs.
maxValue	R/W	Integer	Scale of the scrollbar. With the scrollbar pointer at the end of the scale *value* will be equal to *maxValue* minus *viewSize*.
orientation	R	Symbol	Runtime property: Orientation of the scrollbar in the current sprite and frame. To change the orientation in the score, alter the width or height of the sprite. The orientation will be vertical if the height of the sprite is greater than its width. Values: #vertical, #horizontal.
scrollMember	R/W	Integer/ String	Reference to the field or text member to scroll. Set this to zero if you are not using the OSscrollbar to scroll a field or text member.
type	R	Symbol	Member only property set to #OSscrollbar.
value	R/W	Integer	Position of the scrollbar pointer. Values: from 0 to (*maxValue* minus *viewSize*).
version	R	String	Version number of the installed OSControl Xtra.
viewSize	R/W	Integer	The viewSize property defines the proportional height (vertical scrollbar) or width (horizontal scrollbar) of the scrollbar pointer.

Methods

Method name	Method Description
click(#up/#down/#pageUp/#pageDown)	Sprite only method that acts like a mouseDown on the named part of the sprite.
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Events

Event name	Event Description
update(spriteRef, partClicked)	Sent to any behaviors on the sprite when the clicks on the sprite, and at regular intervals while the mouse is down if the value of the sprite changes. The parameter *partClicked* will have one of

the following values: #up/#down/#pageUp/#pageDown/#thumb.

Notes & Tips

Version 2.0 of the OSControl xtra introduces a new *scrollMember* property. This refers to the field or text member which will be scrolled by the OSscrollbar. The scrollbar will update its display automatically if the length of the text in the chosen member changes, (for instance, if the user types in additional text), or if the *scrollTop* property of the scrolled member is changed. This means that you no longer need to use any behaviors if all you want to do is scroll text. Use any expression that Director can resolve to a field or text member in the *scrollMember* property.

An OSscrollbar can be used horizontally or vertically. By default, OSscrollbar sprites appear horizontally. To change the orientation, alter the width or height of the sprite. The orientation will switch to vertical if the height of the sprite is greater than its width.

On the Macintosh, scrollbars are almost always 16 pixels wide. On Windows, the scrollbar size is either defined by the developer or by a fixed number of Dialog Units (DLUs) that can be set in the Display settings Control panel.

On the Macintosh, the scrollbar will remain 16 pixels wide irrespective of the width of the sprite's rect. On Windows the scrollbar will scale to exactly fit the sprite dimensions. So to get a common display on both platforms we recommend that you set the sprite width to 16 pixels.

Clicking this control will send *update* calls to your sprite behavior each time the *value* of the sprite changes:

```
on update me, partClicked
```

… where *partClicked* can be `#thumb, #up, #down, #pageUp or #pageDown.`

3.12 OSslider

An OSslider control is used to make a selection from a limited range of integer values. Examples of sliders can be found in the Director Paint Preferences Dialog.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
focusable	R/W	Boolean	Defines if the slider can have focus on Windows.
groupId	R/W	Integer	Group to which this control belongs.
maxValue	R/W	Integer	Scale of the slider. With the slider pointer at the end of the scale the *value* will be equal to *maxValue*.
orientation	R	Symbol	Runtime property: Orientation of the scrollbar in the current sprite and frame. To change the orientation in the score, alter the width or height of the sprite. The orientation will be vertical if the height of the sprite is greater than its width. Values: #vertical, #horizontal.
style	R/W	Symbol	Pointer style of the slider. Set to #plain gives a square pointer but set to #up and #down gives an arrow pointer. When *orientation* is vertical, #up will point left and #down will point right. Tick marks are moved to match the style.
tickMarks	R/W	Integer	Number of tick-marks that are shown on the slider. If the number is too large (more than the width or height of the slider divided by 3.5), tick-marks are not displayed. You can hide the tick marks by making the rect narrower or setting the number of tick-marks to less than 2.
type	R	Symbol	Member only property set to #OSslider.
value	R/W	Integer	Current position of the slider pointer. Values: 0 completely down or left, *maxValue* completely up or right.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Events

Event name	Event Description
update()	Sent to any behaviors on the sprite when the clicks on the sprite, and at regular intervals while the mouse is down if the value of the sprite changes.

Notes & Tips

An OSslider can be used horizontally and vertically. By default, OSslider sprites appear horizontally. To change the *orientation,* alter the width or height of the sprite. The *orientation* will switch to vertical if the height of the sprite is greater than its width.

Clicking this control will send *update* calls to your sprite behavior each time the *value* of the sprite changes:

```
on update me
```

3.13 OSstatictext

An OSstatictext is a non-editable text control that can be used for labels or other fixed text. The text is not anti-aliased and cannot have multiple styles or other formatting. The advantage over a normal Director field is that you can set the font to the specific system font that is defined on the playback machine. Examples of text controls can be found in almost all Director windows.

Properties

Property name	Access	Type	Property Description
alignment	R/W	Symbol	Alignment of the text within the rect of the sprite. Values: #left, #right, #center.
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
font	R/W	String	Text font. Font settings are ignored on Windows. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontColor	R	RGB	Color of the text font at runtime (means of access to system setting).
fontSize	R/W	Integer	Text font size.
fontStyle	R/W	List of symbols	Text style. Any combination of bold, italic and underline can be set. Values: #plain, #bold, #italic, #underline.
groupId	R/W	Integer	Group to which this control belongs.
selectionColor	R	RGB	Color of the selection rectangle (means of access to system setting).
text	R/W	String	Text for the box. The box does not scale to accommodate long text.
type	R	Symbol	Member only property set to #OSstatictext
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
showProps()	Displays the properties for this control in the message window.

Notes & Tips

An OSstatictext can hold up to 64 K of text. If you change the text of the OSstatictext through Lingo it will, unlike Director field members, update after the first updateStage.

An OSstatictext member will wrap its text within its rect width on both Macintosh and Windows. Make sure the text fits in the sprite height on both platforms.

3.14 OStabs

OStabs is a tab control.

Properties

Property name	Access	Type	Property Description
enabled	R/W	Boolean	Enables interaction; disabled controls are grayed out.
font	R/W	String	Text font. Font settings are ignored on Windows. See section 3.9 Text Appearance for tips on the choice of font and fontSize.
fontSize	R/W	Integer	Text font
groupId	R/W	Integer	Group to which this control belongs.
itemList	R/W	List	Runtime Property
itemString	R/W	String	Runtime property
tabMember	R/W	String/Integer	Name or number of the Field cast member that contains the menu items. (NB: for performance this member must be a #field member, not a #text member.) The tab headers in this field can contain the following special characters: | Any Lingo code or handler after this character is executed when the tab is chosen. * The first header name preceded by an asterisk will be the default selected tab. (Disables the tab.
type	R	Symbol	Member only property set to #OStabs.
value	R/W	Integer	Index of selected tab.
version	R	String	Version number of the installed OSControl Xtra.

Methods

Method name	Method Description
getItem(path)	Sprite only method
setGroupProp(#property, value)	Sets the property of each OSControl in the group.
setItem(path, value)	Sprite only method. Resets the string of the item or, if you specify a number greater than the number of items, appends a new item.
showProps()	Displays the properties for this control in the message window.
updateControl	Sprite only method

Events

Event name	Event Description
update(spriteRef, tabIndex,tabString)	Sent to any behaviors on the sprite when the user selects a tab, unless that tab has a Lingo expression associated with it. The *tabIndex* parameter will be an integer indicating which tab was selected. The *tabString* parameter will be the text of the selected tab.

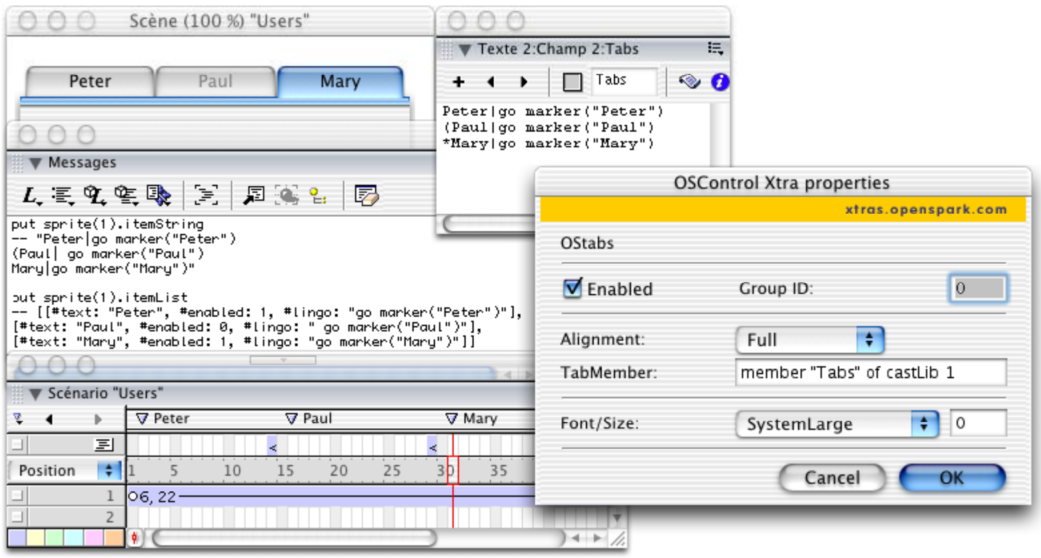
Notes & Tips

The OStabs control works in a very similar way to the OSpopupmenu.

You can define the tab headers using a *tabMember* field. Place an asterisk before a header name to indicate the tab to select by default, or use a (character to disable a tab. You can also define a Lingo instruction to execute by placing the instruction after a pipe character. In the example illustrated below, the tabs are defined by a field with the following text:

```
Peter|go marker("Peter")
(Paul|go marker("Paul")
*Mary|go marker("Mary")
```

The asterisk on the last line ensures that the tab "Mary" appears selected by default, while the (character on the second line disables the tab "Paul". Neither the asterisk nor the (character appears in the tab header, nor does any text that appears after the pipe:



```
Peter|go marker("Peter")
(Paul|go marker("Paul")
*Mary|go marker("Mary")
```

```
put sprite(1).itemString
-- "Peter|go marker("Peter")
(Paul| go marker("Paul")
Mary|go marker("Mary")"

put sprite(1).itemList
-- [[#text: "Peter", #enabled: 1, #lingo: "go marker("Peter")"],
[#text: "Paul", #enabled: 0, #lingo: " go marker("Paul")"],
[#text: "Mary", #enabled: 1, #lingo: "go marker("Mary")"]]
```

OSControl Xtra properties

OStabs

- Enabled: ✓
- Group ID: 0
- Alignment: Full
- TabMember: member "Tabs" of castLib 1
- Font/Size: SystemLarge 0

Thanks to the Lingo after the pipe character, clicking on a tab makes the playback head jump to the appropriate marker.

itemList, *itemString* and *setItem*

As with the OSmenu and the OSpopupmenu, you can use the *itemList* and *itemString* properties of the sprite to modify the headers and their actions on the fly, for the entire sprite. If you want to change the properties of a single tab, you can use the setItem() method. The following instruction enables the tab "Paul", and changes its name to "*Paul":

```
sprite(1).setItem(2, [#enabled: 1, #text: "*Paul"])
```

Note that the asterisk does appear in the tab header, since the *setItem()* method has no effect on which tab is selected. To select the tab "Paul" using Lingo, you would need to set the *value* property:

```
sprite(1).value = 2
```

Tab 2 will be selected even if it is disabled. A tab that is disabled has a different appearance when it is selected than an enabled tab.

Special characters in menuMember fields

Please see the notes in the entry for OSmenu members.

4. Working with OSControl

This section provides some further explanation of how the various controls are used and provides a number of valuable tips.

4.1 Updating from previous versions

initProps() – No Longer Needed

The sprite method *initProps(sprite)* in previous versions of OSControl is now no longer needed. It addressed an issue where Director will not reset the OSControl sprite properties across spans. A new transparent means of dealing with this has been included in the new version of OSControl and *initProps()* is no longer required. In the new version of OSControl this method is ignored so code written for older version can be run against the new Xtra.

updateControl() for OSmenu, Ospopupmenu and OStabs members

In version 1.x, you could pass a string as the parameter to an *updateControl()* call to an OSmenu or an OSpopupmenu. For backwards compatibility, this technique has been retained in version 2.0. However, two new properties have been introduced to simplify the process with the new hierarchical OSmenu and OStabs controls.

We recommend that you use the following technique in version 2.0:

```
anOSmenuSprite.itemString      = stringMenuDefinition
anOSmenuSprite.itemList        = listMenuDefinition
anOSpopupmenuSprite.itemString = stringMenuDefinition
anOSpopupmenuSprite.itemList   = listMenuDefinition
anOStabsSprite.itemString      = stringTabsDefinition
anOStabsSprite.itemList        = listTabsDefinition
```

For details of what is meant by `string...Definition` and `list...Definition`, see the entries for the appropriate controls in section 2 above.

For the new OStabs members, string parameters to the *updateControl()* method will be ignored. You can continue to use *updateControl()* with no parameters to restore the menu or tab definition stored in the field associated with the member or sprite.

4.2 Updating the Stage

OSControl sprites are designed to be used in conjunction with sprite behaviors. Unlike the standard Director controls, OSControl sprites do not automatically redraw themselves when modified. As a result, the display of OSControl sprites may appear to lag after a *mouseDown*, *mouseUp*, *mouseUpOutside* or *update* event. This will be especially noticeable at low frame rates (below about 10 frames per second). To deal with this issue, you can attach a behavior to your OSControl sprites that issues an explicit *updateStage* call on *mouseDown*, *mouseUp*, *mouseUpOutside* and *update*.

Note that issuing an *updateStage* call will make filmLoop sprites advance by one frame. This may result in the filmLoop sprites appearing out of synch with other sprites. If the tempo of the movie is 10 frames per second or greater, you may prefer to allow Director to update the stage automatically. The following behavior may prove a useful compromise:

```
on mouseDown
  if the frameTempo < 10 then
    updateStage
  end if
end mouseDown

on mouseUp
  if the frameTempo < 10 then
    updateStage
  end if
end mouseUp

on mouseUpOutside
  if the frameTempo < 10 then
    updateStage
  end if
end mouseUpOutside

on update
  if the frameTempo < 10 then
    updateStage
  end if
end update
```

4.3 Setting Properties

Setting Sprite Properties and Setting Member Properties

There is an important difference between setting the properties of a member (set the xproperty of member myOSControl) and of a sprite (set the xproperty of sprite y). If you change properties at member level, either through the OSControl Properties Dialog or through Lingo, changes will be saved in the file and persist. Also all instances of those members used throughout your movie will be changed.

If you set the properties of an OSControl at sprite level, these changes will only have effect during that sprite span. So if you go to another section of your project, and later come back, the properties of the OSControl will revert to the member property settings.

Using *getPropertyDescriptionList*

Another way to change properties is to use the on getPropertyDescriptionList() handler in a behavior. Those properties will be saved in the score and will also be persistent. So you can have one button member that is used in different sprites in your movie, and every button has a different button text, because the text property of the sprite is set by the behavior attached to it. In order to accentuate the special character of these properties that are set in the GPDL handler, you have to prefix them with the string init_. They do not need to be declared as properties at the top of your script, since they are only used when the behavior is attached. This method is unique to this Xtra and can be very powerful.

Example: Adding this behavior to an OSControl Push Button will change the text of the OSpushbutton instance, but will not affect the other properties that are set at member level.

```
on getPropertyDescriptionList me
      return [ \
#init_Text: [ \
  #comment: "Button Text":, \
  #format:  #string, \
  #default: "Push Button" \
  ] \
]
end getPropertyDescriptionList
```

For further examples of behaviors setting member properties, see the OSControl Demo movie, and the behaviors in the OSControl section of the Library palette.

Important note for users of Director 7
Because of a severe memory leak caused by a bug in Director 7 (which we unfortunately could not work around) we had to disable this function of the OSControl Xtra. The *init_* behavior initializers do not work anymore in version 1.2 and up of the OSControl Xtra. In order to make previously written behaviors compatible with this change we have devised the following trick.

Example:

```
on beginSprite me
```

```
  -- INIT PROPS IN DIRECTOR 7
  global version
  if version < 8 then
    -- init behavior initializers on beginSprite because of
    -- Director 7 scriptList memory leak (see faq for more info)
    sprite(pMySprite).style   = me.init_Style
    sprite(pMySprite).enabled = me.init_Enabled
    sprite(pMySprite).groupID = me.init_GroupID
    sprite(pMySprite).text    = me.init_Text
  end if

  -- rest of beginSprite handler omitted
end beginSprite
```

In Director 8 and up, there is yet another way to change the properties of both OSControl sprites and members. If you select a sprite containing an OSControl member in the Score window, or an OSControl member in the Cast window, and open Director's Property Inspector panel, you will see an extra tab with an Xtra icon. Here you can set all the properties that the selected member supports. In order for this to function the text files in the folder Props should be copied to the Props/Member folder in your Director Application Folder. See the installation section for more information.

4.4 Putting Properties

To find out which properties are associated with a particular OSControl member or sprite, use ShowProps:

```
sprite(x).showProps()
```

or

```
member(x).showProps()
```

This outputs all the objects properties and their values to the Message Window. Please note that for readability, the display of itemString and itemList properties is limited to their first characters.

4.5 GroupID

Every OSControl member or sprite has a *groupID* property. Setting a group of OSControls to the same (non-zero) *groupID* allows you to address them and/or their sprites using one call:

```
setGroupProp(#property, value)
```

This very powerful messaging mechanism allows you, for example, to disable quickly an entire group of controls by addressing only one of the group's members:

```
sprite(x).setGroupProp(#enabled, FALSE)
```

The *groupID* is also used for OSradioButton controls to automatically ensure only one within the group is set. In this case, only the sprites with the same groupID are affected, not the members.

4.6 Click

At times, you may want to simulate a click on a control. You can achieve this by calling click on the sprite instance:

```
sprite(x).click()
```

The OSControl will behave just as if it were briefly clicked by the mouse and will send *mouseDown* and *mouseUp* messages to the behaviors on the sprite. On the Macintosh, you will actually see the control being pressed for a moment, on Windows you will not (this complies with the standards on each platform). Multipart controls can have an optional part parameter (e.g. #pageUp or #down).

Known issue: If you have a dual monitor system, and you send a click message to an OSpopupmenu or OSmenu sprite, the menu may not appear on the correct position if the current mouse location is not in the monitor where the control is.

4.7 Focus

For OSpushbutton sprites with their style set to #default, the keyboardFocusSprite will automatically simulate a click (see above) when you hit the return key on your keyboard (just like any other default pushbutton reacts in other applications). When such a click is sent to an OSControl sprite, it also generates *mouseDown* and *mouseUp* messages.

On Windows, the situation is less clear because some, but not all OSControls also support focus. In the Windows GUI, a control that has focus can be manipulated through the keyboard. When an OSControl has focus, it is the keyboardFocusSprite (see Director help for details) and behaves just like Directors own focusable sprites (e.g. editable text fields). Hitting the tab key on your keyboard cycles through the focusable sprites, hitting RETURN or ENTER while an OSControl has focus generates a click event. If you do not want your OSControls to be focusable, you can set their focusable property to false. Remember focus only applies to Windows.

Technical note on focusable sprites and behavior initializers: If you set the focusable property of a sprite through behavior initializers, Director may not mark the sprite as the keyboardFocusSprite immediately after the sprite span starts. It is not until the first exitFrame that Director will make this sprite the keyboardFocusSprite (unless there are other, lower-numbered focusable sprites). You can get around this problem by explicitly setting the keyboardFocusSprite to your sprite in your beginSprite handler.

4.8 Cross-platform Differences

Creating an interface that looks and feels right for the platform on which it is run requires more than just having the right look for the controls. Each platform has its own set of norms for interface layout, control dimensions and how different controls are used. You need to be aware of these issues when designing your interface, and you need to test your interface on each of your target platforms. This section describes a number of design issues that you need to keep in mind.

Overview of Platform Differences

An OSscrollbar has a maximum width on Macintosh of 16 pixels but on Windows an OSscrollbar will fill the whole sprite rect.

An OSmenu control is a popup menu on the Macintosh and a shortcut menu on Windows. On Macintosh you can choose any font but on Windows the font is always the SystemLarge font.

OSpopupmenu: although they are quite different, the Windows Combobox and the Macintosh Popup Button are the closest matching controls for what is generally referred to as a popup-menu. You may not be able to provide a consistent UI by using this control for both platforms. On Windows for example, it is not very common to use a Combobox for navigating between different program parts, while on the Macintosh and in web pages it is. What is also important to keep in mind is that a Windows Listbox (the pop-up part of the Combobox) cannot contain checkmarks, disabled items and separator lines. So these will not show up on Windows when you implement these on the Mac. The individual item font style codes are only available on the Mac, and ignored on Windows.

On Macintosh an OSpushbutton's height is defined within the operating system; on Windows it can have any size, and will always occupy almost the complete sprite rect.

A default style OSpushbutton can lose its default state (=focus) on Windows when another control gets focus.

Differences in text appearance are treated separately below.

Testing the OS version on Macintosh

Because the OSControl Xtra requires minimal Mac OS 8.5 to play back on the Macintosh, we wrote the following code to check for the current system version. It uses the free Mac version of the Buddy API Xtra which can be found on http://www.buddyapi.com. Ideally this code should be placed in the prepareMovie handler of a stub projector that then branches to your project, or, if the minimum system version is not met, to another movie that informs the user about the incompatibility of its system version with the OSControl Xtra.

```
on prepareMovie
```

```
  -- code to test version number of Mac OS so you can alert users
  -- who still use Mac OS < 8.5

  if the platform contains Mac then
    goodversion = 1
    xtraList = string(the XtraList)

    if xtraList contains "buddy" then
      macVer = string(baVersion(mac))

      if macVer < 8.5 then --running pre Mac OS 8.0 system
        goodversion = 0
      end if

      if NOT goodVersion then
         -- alert user or branch to warning movie.
         -- alert "This movie requires the OSControl Xtra that
         -- is not compatible with your system version."&RETURN&
         -- "Upgrade to Mac OS 8.5 or higher and try again."
         -- halt --stop execution
         go movie alertMovie

      else -- branch to right movie

         go movie myproject

      end if
    end if
  end if
end prepareMovie
```

4.9 Text Appearance

System Fonts

For a truly platform-specific look, you can choose to use SystemLarge and SystemSmall as the font for an OSControl. In this case, the font used will depend on System settings which may be customized by the end user. If you set the fontSize of the SystemLarge and SystemSmall fonts to 0, the size of the font will also depend on the user's customized settings.

On most platforms, you can overrule the user's customized settings by setting the fontSize to a value other than 0. Mac OS X behaves slightly differently: the fontSize of the SystemLarge and SystemSmall fonts is always set to suit the user's customized settings, regardless of the value for fontSize that you may have set.

If the movie is running on Mac OS X or if you have explicitly set the fontSize to 0, the text displayed in an OSControl sprite may be too large for the sprite. The sprite will not be resized to fit the text.

Oversized Text

The way the oversized text appears depends on the platform and the version of Director. In Director MX on Mac OS X, for example, letters will be moved closer together and may overlap. In Director 8.5 in Mac Classic, letters that cannot be shown will be clipped and replaced with three dots… In Windows XP, the text will simply be clipped at the borders of the sprite.

Missing Fonts

If you choose a font which is not installed on the target machine, it will be replaced by one the system fonts. If you want to use a specific font with your OSControl sprites, you should embed it in your movie, by creating a #font member.

Font members are installed at runtime onto the end-users machine. The fonts become available to other applications while the Director movie is running. On certain Windows machines with tight security settings, the current user may not have sufficient privileges for the embedded font to be installed. In this case, any text in that font may not appear at all.

5. Feedback

We are interested in your view on the quality and functionality of OSControl.

Please contact us by email support@openspark.com and let us know your views, especially:

- If a control does not behave as you would expect;
- You discover a bug or want to suggest improvements, or;
- You have suggestions and ideas for future versions.

Many thanks for your interest in the OSControl Xtra and for any feedback.