Tutorials

Top Previous Next

This chapter gives worked examples of how you can use SWiSHmax.

- Creating Your First SWiSH Movie takes you through the basic steps of creating a simple Movie. It also covers sound, interactivity, preloaders and preparing your Movie for the web
- Using Actions to Control your Movie is a short example of how to use Events and Actions. The Creating Your First SWiSH Movie tutorial also has topics covering this area
- How to make an Animated Button explains how to make a button that animates when you move the mouse over the button
- Scripting Tutorials are a series of tutorials designed to help you become familiar with the scripting language. You should also refer to Scripting for an introduction to basic scripting concepts
- Using SWiSH with other programs shows you how you can use your .swf Movie with other applications, namely Flash, Dreamweaver, FrontPage and Powerpoint

Creating Your First SWiSH Movie

Top Previous Next

This section is broken into step-by-step tutorials that will take you through the process of creating a basic animated web site. The final web site is saved as "first.swi" in the Samples Menu, you may want to view it before beginning this tutorial to get an idea of what you will be creating.

Each section is a stand-alone tutorial which will help you create the individual components of the website. The tutorials are:

- Creating your first SWiSH Movie
- Importing and Working with Images and Vector Graphics
- Adding Music and Sounds
- Creating Buttons
- Creating a Menu
- Creating a Sprite
- Adding Interactivity to your Movie
- Creating a Preloader
- Exporting your Movie and getting it ready for the Web.

Creating Your First SWiSH Movie

Top Previous Next

This is a step-by-step tutorial on how to create your first SWiSH Movie. This tutorial illustrates some of the Text Effects seen in the "first.swi" file in the File | Samples | Tutorials Menu.

1. Start the SWiSH application

2. From the File Menu select Samples | Tutorials | "first.swi"

3. From the File Menu select Test | In Player

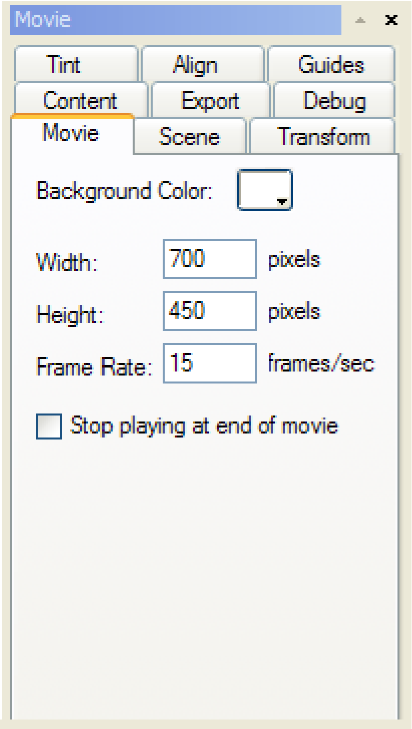
When you are ready to begin creating your Movie, you can close the player window or minimize it to view the Movie as a reference while you work on the following tutorials.

1. From the File Menu select New

2. Define your Movie's properties by select the Movie Panel and changing the Width to 700, Height to 450, and

Frame Rate to 15. Make sure that the Background color is set to white

The 'Movie' Panel should look like this:



Background Color:

Width: 700 pixels

Height: 450 pixels

Frame Rate: 15 frames/sec

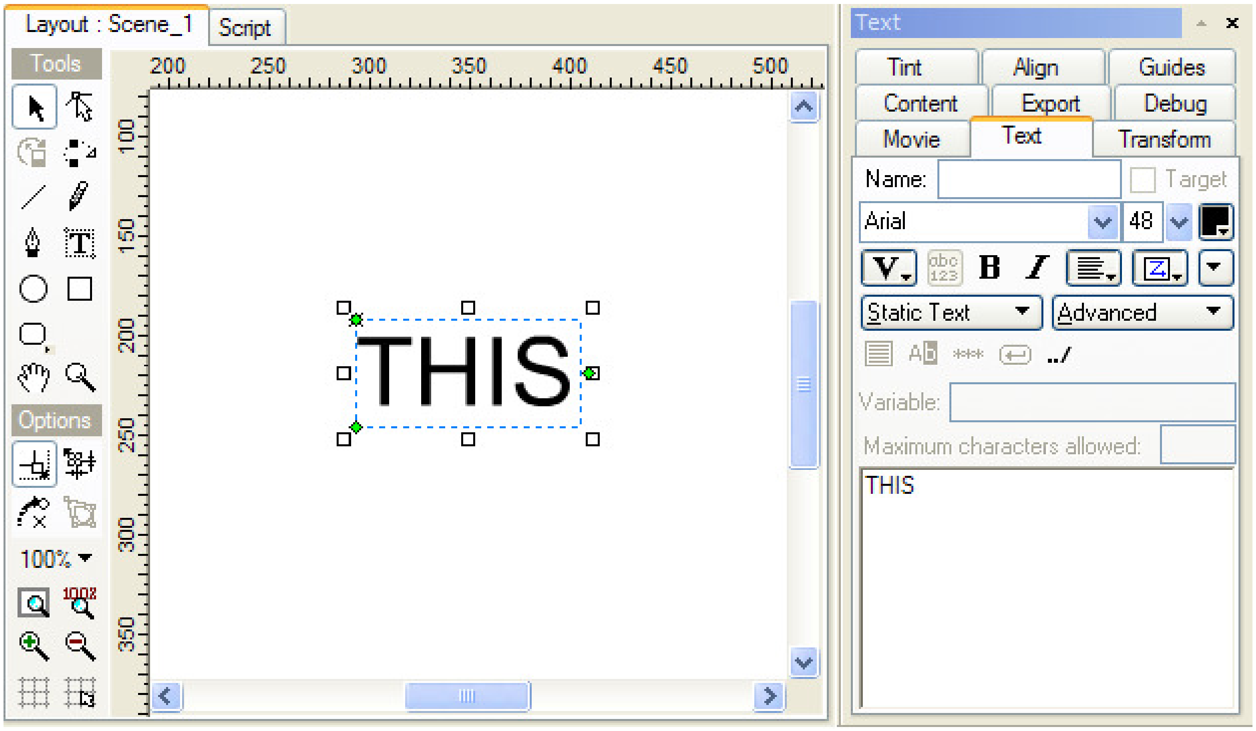
□ Stop playing at end of movie

3. In the Zoom Controls of the Layout Panel, click on the Zoom 100% button. This will center the stage area within the 'Layout' Panel

4. Ensure Scene 1 is selected in the Outline Panel, and press the Insert Text button on the Insert Toolbar. You should see the word 'Text' appear in the center of the stage in the Layout Panel

5. Select the Text Panel and change the word 'Text' to the word 'THIS' in the text window. Change the font size to 48 point from the drop-down font size Menu, or by or entering 48 in the font size box of the 'Text' Panel

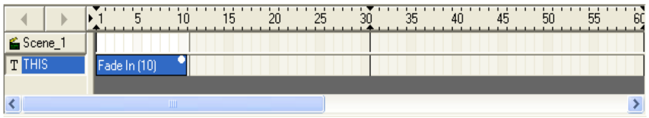
The 'Layout' Panel should look like this:



Note: The Anchor Point (small cross) is in the middle of the Text Object. This is defined by the last used setting of the Transform Panel. The anchor point should be set to 'Origin', otherwise the following Effects will appear different to those shown in the sample

6. In the Timeline Panel, click on Frame 1 in the row for 'THIS'. Press the Add Effect ▾ 'Add Effect' button, which is on the left of the 'Timeline' Panel. Select Fade | Fade In from the drop-down Menu. Double click on any Frame of the Fade In Effect in the Timeline, the Fade In Settings dialog box will appear. From the 'Fade In Settings' dialog box, press the button labeled "More Options >>" then uncheck the 'Continue from previous Effect' checkbox to access the Start At tab. From the Start At tab, select X Scale | Scale Factor, and enter the value 20 in the % edit box and press enter. Select the Motion tab, and select X Scale | Resize to 100%. Press the 'Close' button, at the bottom right of the dialog, to close the dialog box

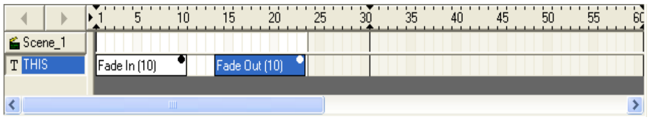
   The Timeline should look like this:

7. Press the ▶ Play Movie button on the Control Toolbar. You should see the word 'THIS' increase in scale as it fades in. This will continue to loop. Press the ■ Stop button

8. Click on Frame 15 in the row for 'THIS'. Press the Add Effect ▾ 'Add Effect' button, and select Fade | Fade Out from the Menu.

   Now the Timeline should look like this:

9. Press the ▶ 'Play Movie' button. You should see the word 'THIS' fade in, and then fade out. Press the ■ 'Stop' button

10. Right-click on the Fade Out Effect in the Timeline, and select Properties from the context Menu to display the 'Fade Out Settings' dialog box. Change the Duration from 10 to 4

Note: When you are finsihed, click on the 'Close' button at the bottom right or the **[X]** button at the top right to hide the effect properties dialog. You can do this whenever you want to hide the effect settings dialog if it gets in the way of seeing the other panels

11. Press the ▶ 'Play Movie' button again. You should now see the word 'THIS' fade out faster than before, because you have shortened its duration. Press the ■ 'Stop' button

12. Press the T 'Insert Text' button. The word 'Text' will appear directly over your first word on the stage, it will also appear in the Timeline and the Outline Panel. In the Text Panel, change the word 'Text' to the word 'IS', the font size should be set to 48 point

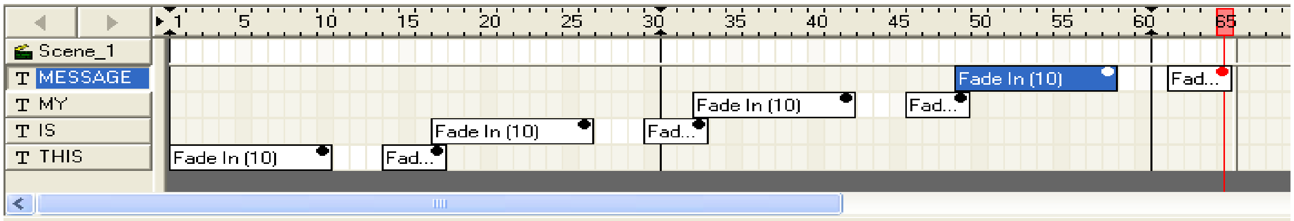
13. Press the T 'Insert Text' button to create a third word. Change this word to 'MY' and leave the font size at 48 point

14. Press the T 'Insert Text' button again to create a fourth word, change this word to 'MESSAGE' and leave the font size at 48 point

15. From the Timeline, click on the Fade In Effect you created for the word 'THIS' and while holding the CTRL key down, click on the Fade Out Effect next to it. Both the Fade In and the Fade Out Effects should be highlighted. Right-click and select Copy Effect from the context Menu. You may need to resize or scroll the Timeline so you can see the rows you want to work on

16. On the Timeline, right-click at Frame 17 next to the word 'IS' and select Paste Effect. Paste these Effects at Frame 35 next to the word 'MY' and at Frame 52 next to the word 'MESSAGE'

The Timeline should now look like this:



17. Click on the Fade Out Effect for the word 'MESSAGE' to select the Effect, then Right-click on it and select Properties from the context Menu. The 'Fade Out Settings' dialog box should appear. Select the Motion Tab, and select X Scale | Resize to Scale and enter the value 400 in the % edit box, press enter and the click on the 'Close' button

18. Press the ▶ 'Play Movie' button on the Toolbar. You should now see the sequence of words 'THIS', 'IS', 'MY', fade in while increasing in scale and then fade out. The Word 'Message' will fade in while increasing in scale and then fade out while increasing in scale. Press the ■ 'Stop' button

19. Now press the 💾 Save Movie button on the Toolbar. You should now see the 'Save As' dialog box. Type 'myfirst' into the File name edit box and press Save. Your Movie has now been saved as a .swi file and can be opened at a later time

You are ready to continue with the Importing and Working with Images and Vector Graphics tutorial.

Importing and Working with Images and Vector Graphics

Top Previous Next

This is a step-by-step tutorial on importing and working with bitmap images and vector graphics. This tutorial illustrates some of the features seen in the "first.swi" file in the File | Samples | Tutorials Menu.

Note:

- You can import bitmap images in .jpg, .jif, .jpeg, .gif, .png, .bmp, and .dib formats
- You can import vector graphics in .wmf and .emf formats, or as a static .swf file

Images

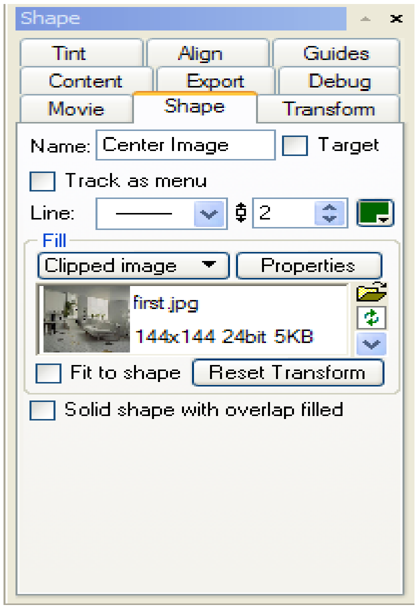
1. Open the Movie you began in the last tutorial "myfirst.swi"

2. Press the 'Insert Image' button on the Toolbar. From the 'Open' dialog box, select JPEG from the 'Files Types' list, then navigate to the Samples folder within your SWiSHmax application folder, select "first.jpg", and press Open

3. The image should be displayed in the center of the stage

4. Select the Shape Panel and edit the name of the image Object to be "Center_Image". Select the solid line from the drop-down Menu. Change the width of the line from 1 to 2pt. Press the color swatch next to the 'Line' width selection box to display the color palette. Select a dark green from the palette.

The 'Shape' Panel should look like this:



Name: Center Image □ Target

□ Track as menu

Line: ——— 2 🟩

Fill

Clipped image

first.jpg
144x144 24bit 5KB

□ Fit to shape

□ Solid shape with overlap filled

5. Press the 'Properties' button located next to the 'Clipped image' button to display the 'Image Properties' dialog box. Within the 'Image' dialog box, check the 'Recompress JPEG' checkbox and change the value from 84% (the quality of the imported image) to 10%.

The 'Image Properties' dialog should like like this:

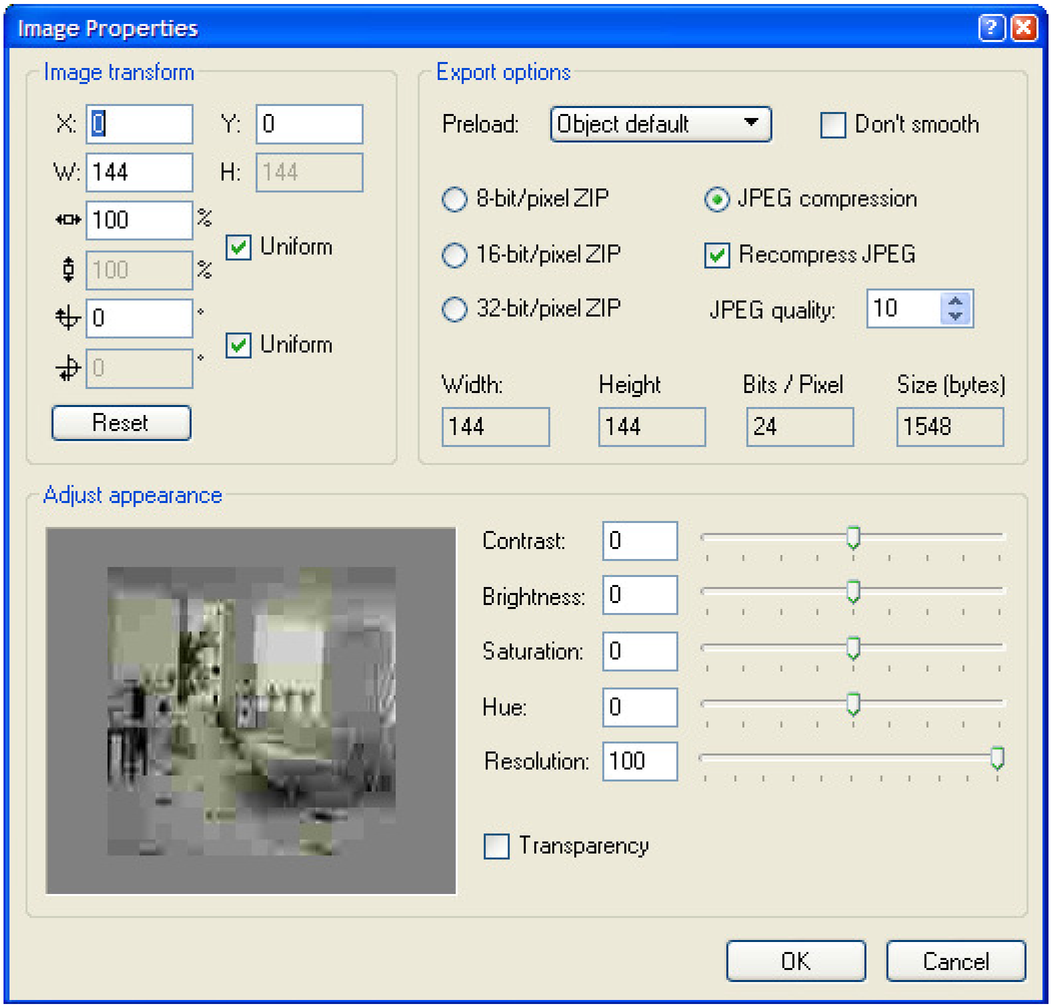


Image Properties

Image transform

X: 0 Y: 0

W: 144 H: 144

100 % ☑ Uniform

100 %

0 ° ☑ Uniform

0 °

Reset

Export options

Preload: Object default □ Don't smooth

○ 8-bit/pixel ZIP ● JPEG compression

○ 16-bit/pixel ZIP ☑ Recompress JPEG

○ 32-bit/pixel ZIP JPEG quality: 10

Width:	Height	Bits / Pixel	Size (bytes)
144	144	24	1548

Adjust appearance

Contrast: 0

Brightness: 0

Saturation: 0

Hue: 0

Resolution: 100

□ Transparency

OK Cancel

Press OK

6. From the File Menu, select Test | In Player. The image should display in the center of the stage, in front of the text and appear distorted.

7. Press the 'Send to Back' button from the Standard Toolbar

8. Press the Shape (image) 'Properties' button again and change the value of the JPEG quality from 10% to 50%, then Press OK

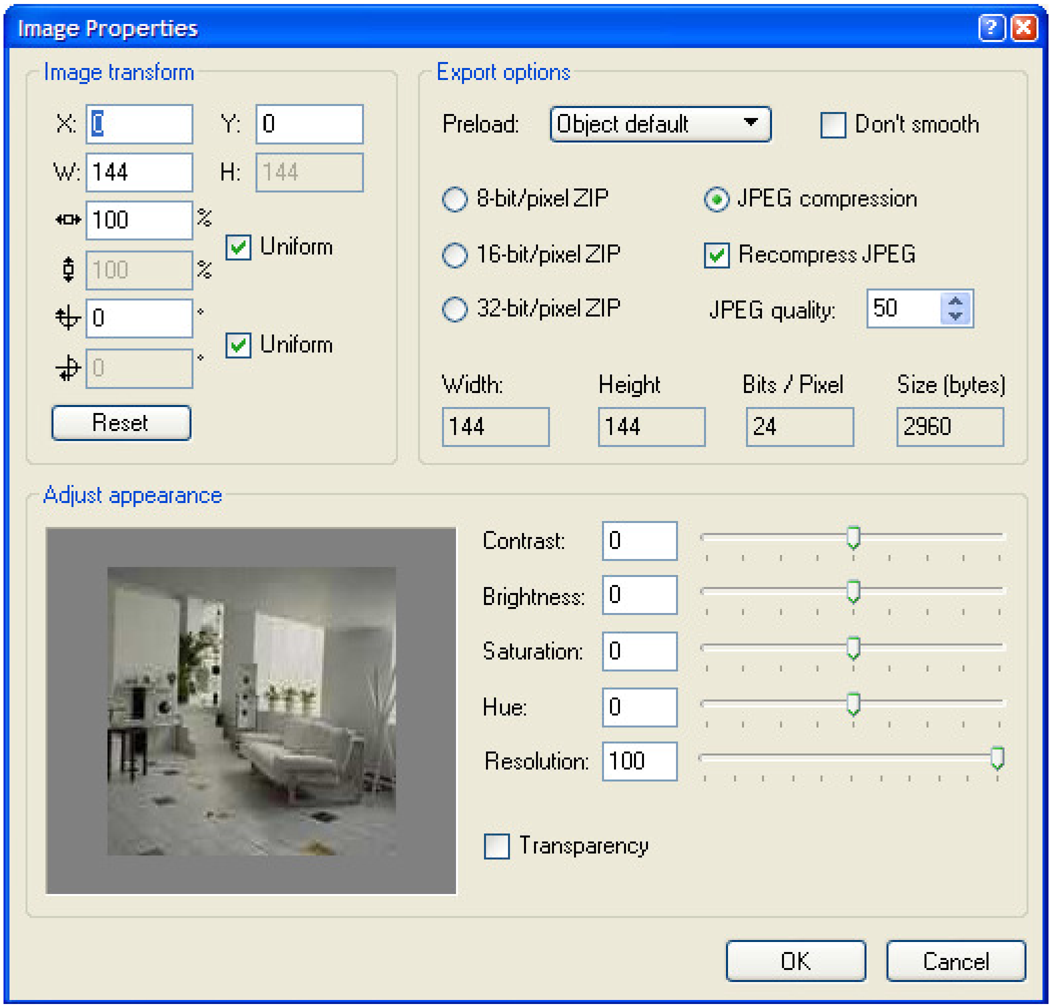


Image Properties

Image transform

X: 0 Y: 0

W: 144 H: 144

100 % ☑ Uniform

100 %

0 ° ☑ Uniform

0 °

Reset

Export options

Preload: Object default ☐ Don't smooth

○ 8-bit/pixel ZIP ● JPEG compression

○ 16-bit/pixel ZIP ☑ Recompress JPEG

○ 32-bit/pixel ZIP JPEG quality: 50

Width:	Height	Bits / Pixel	Size (bytes)
144	144	24	2960

Adjust appearance

Contrast: 0

Brightness: 0

Saturation: 0

Hue: 0

Resolution: 100

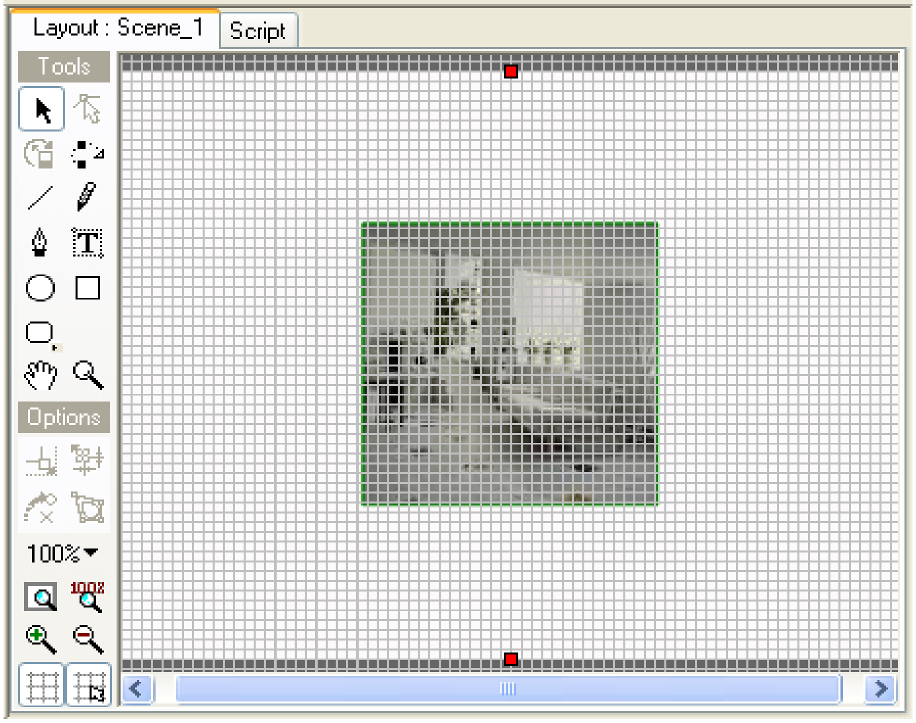
☐ Transparency

9. From the File Menu, select Test | In Player again. This time the image should appear clear and behind the text

10. Select the "Center_Image" Object from the left column of the Timeline, then right-click and select Copy Object from the context Menu. Press the 'Paste' button to paste a second image into your Scene. The copy will appear directly over the original

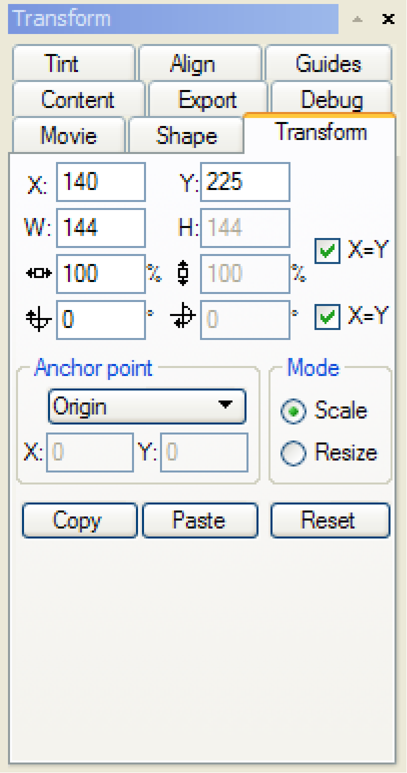
11. To help with aligning this image, select the Guides Panel, and check the 'Snap' and 'Show' in the Grid section. A Grid will appear in the Layout Panel. Change the horizontal and vertical values of the Grid in both the 'Horiz' and 'Vert' edit boxes to 5

The 'Layout' Panel should look something like this:



12. Select the Transform Panel to view the image's X and Y placement properties. With the image selected, drag the image over towards the left side of the window, and place the image at X:140 Y:225

The 'Transform' Panel should look like this:



X: 140 Y: 225
W: 144 H: 144
100 % 100 % ☑ X=Y
0 ° 0 ° ☑ X=Y
Anchor point: Origin X: 0 Y: 0
Mode: ● Scale ○ Resize

13. Rename this image object to "Left_Image" in the Shape Panel

14. With the "Left_Image" selected, press the ⤓ 'Send to Back' button from the Toolbar, and then press the ↑ 'Bring Forward' button. This image should be in the second-from-the-bottom row in the Timeline, just above Center Image.

15. Press the 📋 'Paste' button to create another copy of the image. Drag this one over to the right side of the window and place the image at X: 560, Y: 225

16. Rename this "Right Image" in the Shape Panel

17. Press the ⤓ 'Send to Back' button

18. In the Timeline Panel in the LeftIImage row, click on Frame 70. Press the Add Effect ▾ 'Add Effect' button and select Fade | Fade In from the Menu. Open the 'Fade in settings' by double clicking on the last Frame of the Effect in the Timeline. Change this Effect's duration from 10 to 4 Frames, and press Close to accept these changes.

19. Right-click on this Effect and select Copy Effect from the context Menu

20. In the Timeline Panel in the Center_Image row, right-click on Frame 72 and select Paste Effect

21. In the Timeline Panel in the Right_Image row, right-click on Frame 74 and select Paste Effect

22. Preview your Movie in the player by selecting File | Test | In Player. This time you will see the message and then, the images will fade in starting with the left image first and ending with the right image

23. Click on Frame 88 in the Left_Image row. Press the Add Effect ▾ 'Add Effect' button and select Move from the Menu. Display the 'Move settings' by double clicking on any frame of the the effect in the timeline. From the Motion tab select X Position | Move to X and change the value from 0 to 350. Change the Effect's duration from 10 to 5 Frames, and press Close to accept the changes

24. Click on Frame 93 in the Center_Image row. Press the Add Effect ▾ 'Add Effect' button and select Move from the Menu. From the Motion tab, in the 'Move settings' dialog box, select X Position | Move to X and change the value from 0 to 560. Change the Effect's duration from 10 to 5 Frames, and press Close to accept these changes

25. Click on Frame 98 in the Right_Image row. Press the Add Effect ▾ 'Add Effect' button and select Place from the Menu. Display the 'Place settings' dialog box and Select X Position | Jump to X, change the value from 0 to 140, and press Close to accept the changes

26. Click on Frame 85 of any row in the Timeline and press the ⚑ Preview Frame button on the Control Toolbar. Drag the Play Head right and then left across the Timeline to preview the animation. "Left_Image" should slide over to the center covering "Right_Image"; then "Center_Image" should slide right to cover "Right_Image"; finally, "Right_Image" will appear on the left

27. Click on Frame 105 in the Left_Image row. Press the Add Effect ▾ 'Add Effect' button and select Fade Out from the Menu. Display the 'Fade settings' by double clicking on any frame of the the effect in the timeline. Change the duration of this effect from 10 to 5 frames. Go in to the Motion settings (press "More Options >>" if you cannot see this tab), and change the X Scale option to "Resize to Scale" and leave the value set at 0%. Change the 'X Position' setting to "Move to X" and enter a value of "350". Press Close to accept these changes.

28. Right-Click on the Fade Out effect at Frame 105 for the Left_Image row, and select Copy Effect from the menu. Right-Click on Frame 106 for the Center_Image row and select Paste Effect. Right-Click on Frame 106 for the Right_Image row, and select Paste Effect.

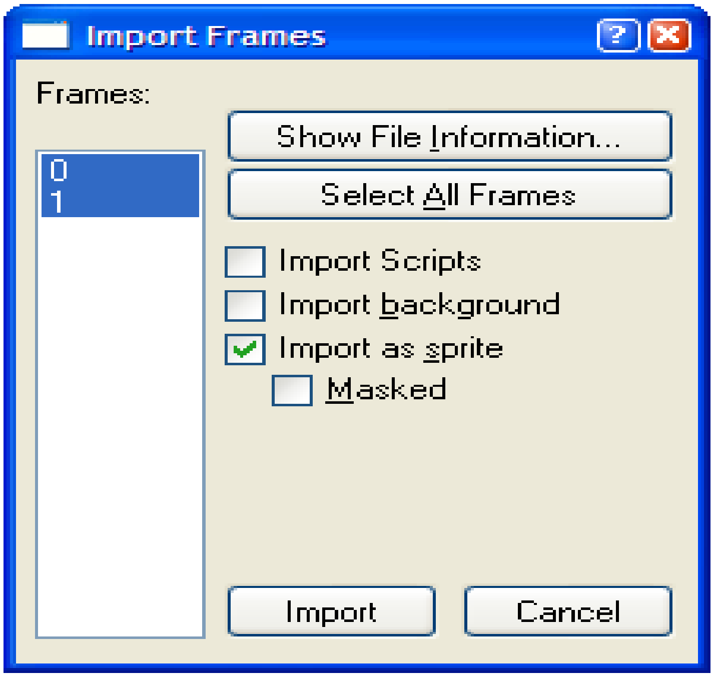
29. Click on Frame 105 of any row in the Timeline and press the ⚑ Preview Frame button on the Control Toolbar. Drag the Play Head right and then left across the Timeline to preview the animation. "Left_Image" should

slide over to the center covering "Right_Image"; then "Center_Image" should slide right to cover "Right_Image"; finally, "Right_Image" will appear on the left

30. Save your Movie

Vector Graphics

1. Press the 'Insert Content' button from the Toolbar. From the 'Open' dialog box, select Flash Player Movie (*.swf) from the 'Files of Type' Menu . From the Samples Folder, select "logo.swf" and press Open. From the 'Import' dialog box press the "Select All" button at the bottom-left, check the 'Animated Sprite' checkbox and uncheck the 'Masked' checkbox as well as the "Include background rectangle" checkbox and press Import. Note that the composite image is within a Sprite, and that the Sprite has been named "logo"



□ Import Scripts
□ Import background
☑ Import as sprite
□ Masked

2. The logo Sprite should appear in the center of the stage.

Note: If the Sprite is shown in a different location other than the center of the stage, you can center it by doing the following: Open the Transform panel; Make sure that the Anchor point of the sprite is set to 'Center'; Change the X and Y settings to X=350, Y=225.

3. After importing the logo.swf as a Sprite, it will be expanded in the Outline panel. To close this Sprite (and exit its timeline), either click on the (+) next to the name of the Sprite in the Outline panel, or click on the "Scene_1" button at the top-left of the Timeline panel (marked with a folder icon to the left of it).

4. Click at Frame 110 on the Timeline in the row for logo. Press the Add Effect ▾ 'Add Effect' button, and select Fade | Fade In from the Menu. Change the Effect's duration in the 'Fade In Settings' dialog box from 10 to 4, and from the Motion tab select X Scale | Resize to Scale and type 60 in the % edit box. Leave the 'X=Y' box checked. Press Close to accept the changes

5. Click at Frame 120 on the Timeline in the row for the logo. Press the Add Effect ▾ 'Add Effect' button, and select Fade | Fade Out from the Menu. In the 'Fade Out settings' dialog box, change the Effect's duration from 10 to 3, and press Close

6. Press the ▶ Play Movie button to preview your Movie. After your message has played and your images have finished moving, the logo should fade in quickly and then fade out quickly. If the 'Stop playing at end of movie'

check box in the Movie Panel is not checked, your Movie will continue to loop. Press the ■ Stop button. For the remainder of the tutorial, ensure that the 'Stop playing at end of movie' check box is checked

7. Press the ▶ Play Movie button to preview your Movie. This time your Movie will play through once. Press the ■ 'Stop' button

8. Save your Movie

You are ready to continue with the Adding Music and Sounds tutorial.

Adding Music and Sounds

Top Previous Next

This is a step-by-step tutorial for adding music and sounds to your Movie. This tutorial illustrates some of the features seen in the "first.swi" file in the File | Samples | Tutorials Menu.

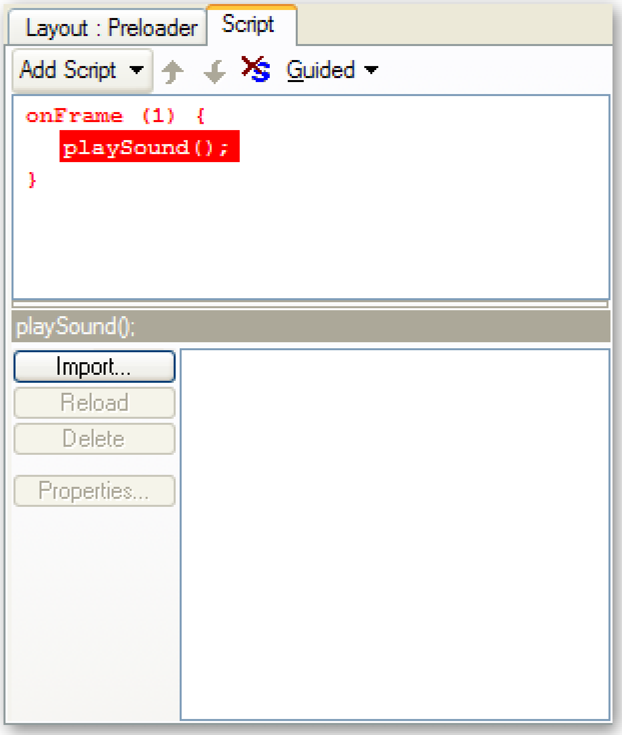
With SWiSH you can import both .wav and .mp3 formats. You can control the overall volume of sounds and apply simple fade Effects. A sound can be played when a Movie reaches a selected Frame or in response to a Mouse Event. For a more detailed explanation on using sounds in your Movie please read Play Sound. This tutorial will be limited to adding sounds to the Timeline in response to a Frame Event; adding sounds in response to Mouse Events will be discussed in the Creating Buttons tutorial.

As sound files can significantly increase the overall file size of the Movie, it is best to select short tunes that can be looped as background music for a Movie that will be playing over the Web.

Note: If you are having problems importing .wav files, try downloading and installing the latest Windows Media Player from http://www.microsoft.com/windows/mediaplayer/en/default.asp

Adding Background Music

1. Open "myfirst.swi" saved from the previous tutorial. Right-click on Frame 1 of the Scene row. From the context Menu select Sound | Play Sound. A S Script Action marker will appear at that Frame on the Timeline

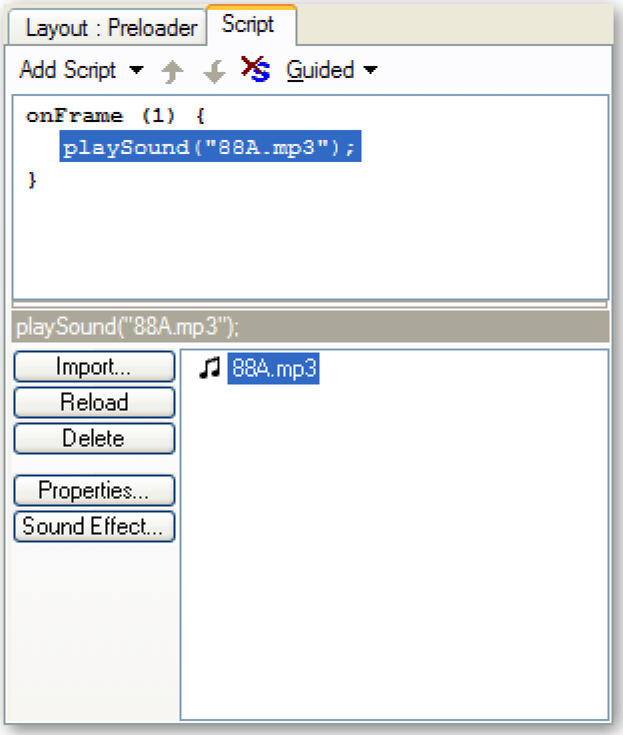


```
onFrame (1) {
    playSound();
}
```

2. From the Actions Panel you should see the Play Sound Event highlighted in the 'Add Script' window. Note that the Action playSound is in red. This is because the statement is currently incomplete (it does not contain a sound to play). Press the 'Import' button to view the 'Import Sound' dialog box. From the Samples folder, select "88A.mp3" and press Open

Note: The "Files of Type input box" should be set to show MP3 or All Files. If not set to one of these settings, you will not be able to see the "88A.mp3" file

3. Left-click on the sound "88A.mp3" to select that as the sound to be played. The selected sound is now shown as the parameter to the playSound() Action. Note that the script marker changes to 🔈 in the Scene Timeline and the Action script turns black as it is now syntactically correct

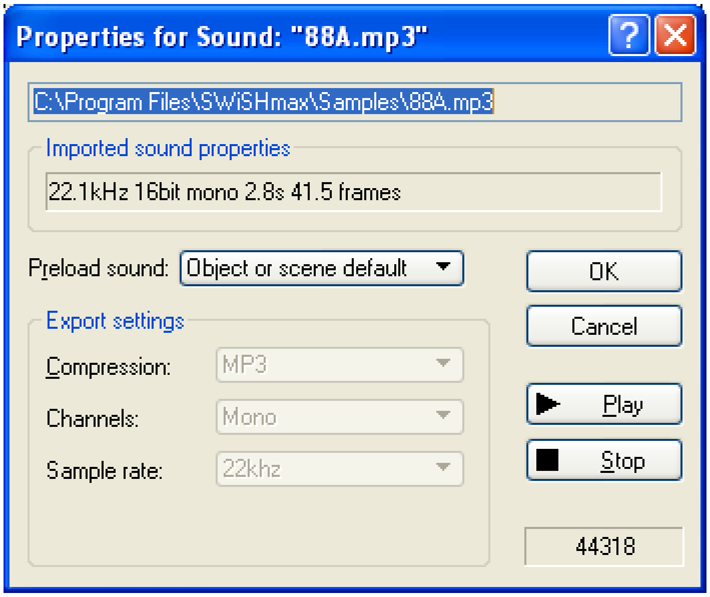


```
onFrame (1) {
    playSound("88A.mp3");
}
```

playSound("88A.mp3");

4. Select the 'Layout' Panel and Press the ▶ Play Movie button on the 'Control' Toolbar. The tune will play once and then stop. Press the ■ 'Stop' button

5. From the Script Panel, below the 'Import' Button, press the Properties... 'Properties...' button to display the 'Properties for Sound' dialog box. This dialog box, displays the file's sample rate and the length of 1x loop in seconds and Frames. Note that the quoted length (41.5 Frames) is not long enough to allow sound to be played through the entire Movie. Close this window by either pressing "Ok", "Cancel", or the **[X]** button at the top-right.



Properties for Sound: "88A.mp3"

C:\Program Files\SWiSHmax\Samples\88A.mp3

Imported sound properties

22.1kHz 16bit mono 2.8s 41.5 frames

Preload sound: Object or scene default

Export settings

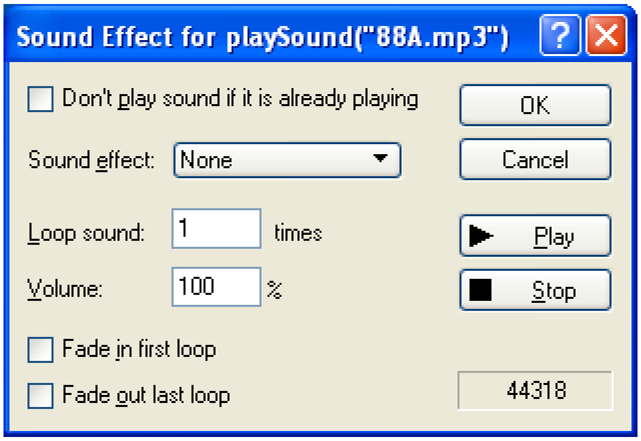
Compression: MP3

Channels: Mono

Sample rate: 22khz

44318

6. From the Script Panel, below the 'Import' Button, press the Sound Effect... 'Sound Effect...' button to display the 'Sound Effect Settings' dialog box



Sound Effect for playSound("88A.mp3")

☐ Don't play sound if it is already playing

Sound effect: None

Loop sound: 1 times

Volume: 100 %

☐ Fade in first loop

☐ Fade out last loop

7. In the 'Loop sound' edit box, enter 15 (this will ensure that the music will continue throughout the introduction). You can leave the Volume setting at 100%, and select the option 'Don't play sound if it is already playing' feature checked. Press OK to accept the changes. This will change the parameters of the playSound() Action to: "`playSound("88A.mp3",True,100,15);`"
8. Select the 'Layout' Panel and Press the ▶ 'Play Movie' button on the Toolbar. The tune will start as your Movie starts and will continue to play long after your animation has stopped

Sound Effects

You can add a Sound Effect in synchrony with the appearance of the logo.

1. Right-click on Frame 110 of the Scene row. From the context Menu select Sound | Play Sound. From the Script Panel press the 'Import' button and from the Samples folder, select "FX1001.wav" and press Open

Note: The "Files of Type input box" should be set to show Wave Sounds or All Files. If not set to one of these settings, you will not be able to see the "FX1001.wav" file

2. Press the Properties... button to display the 'Properties' dialog box. By default, the Compression for .wav files is set to MP3; leave this setting, as this will produce the smallest file size, compressing your .wav files up to 1/10 of their original size. Leave the Preload sound set to Object or Scene default, press OK

3. Click on the Sound Effect... button, as you do not want this sound to loop, leave the 'Loop sound' edit box value set to 1. Leave the Volume value set to 100%. Select Sound effect | Pan left to right. Press OK to accept these changes. The playSound action should change to: "`playSound ("FX1001.WAV",False,100,1,False,False,'Pan Left to Right');`"

4. Select the 'Layout' Panel and Press the ▶ 'Play Movie' button on the Toolbar. The sound will begin playing when your logo appears. It will begin playing in the left speaker and end playing in the right speaker. Press the ■ 'Stop' button

Stopping Sounds

1. Right-click on Frame 111 and select Sound | Stop Sound from the context Menu. Select "88A.mp3" and press Enter on your keyboard

Note:

- If you need or want to stop your background music in the middle of a loop, rather than having it stop abruptly, it may be helpful to the flow of the Movie if you stop the music while another sound is playing, as you have done in this example
- There may be times when you will want to stop all sounds playing. This can be done within the Timeline by

right-clicking on the Frame where you would like the sounds to stop and selecting Stop All Sounds from the context Menu

2. Select the 'Layout' Panel and Press the ▶ 'Play Movie' button on the Toolbar. Your background music will end just as "FX1001.wav" begins. Press the ■ 'Stop' button

3. Your introduction is complete. You may view it in your browser by pressing the 'Test in Browser' button

4. Save your Movie

You are ready to continue with the Creating Buttons tutorial.

Creating Buttons

Top Previous Next

This is a step-by-step tutorial for creating a simple button. This tutorial illustrates some of the features seen in the "first.swi" file in the File | Samples | Tutorials Menu.

The Button Object makes it easy to add mouseover Effects to your Movie without interrupting the Movie's main Timeline.

You can create an empty button by pressing the Insert Button on the Insert Toolbar, or you can choose an existing Object and convert it into a button by selecting Modify | Convert | Convert to Button or by pressing the 'Convert to Button' icon from the Grouping Toolbar. The 'Grouping' Toolbar can be can be displayed by selecting 'View | Toolbars | Grouping Toolbar'.

Note: In most circumstances, it is best create your text or shape first and then use the 'Convert to Button' option (remove any Actions attached to a shape before converting it into a button). This will save you the time of adding the text or shape for each state; you need only to go into the Button Object's 'Outline' tree to make your adjustments to the different states

To Create a Button

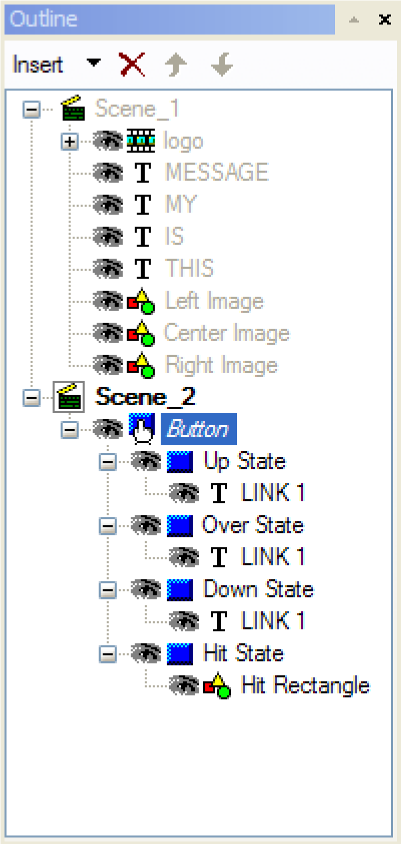
1. Open "myfirst.swi" saved from the previous tutorial. Your introduction has ended and the visitor will now enter your main site. This is a good time to create a new Scene

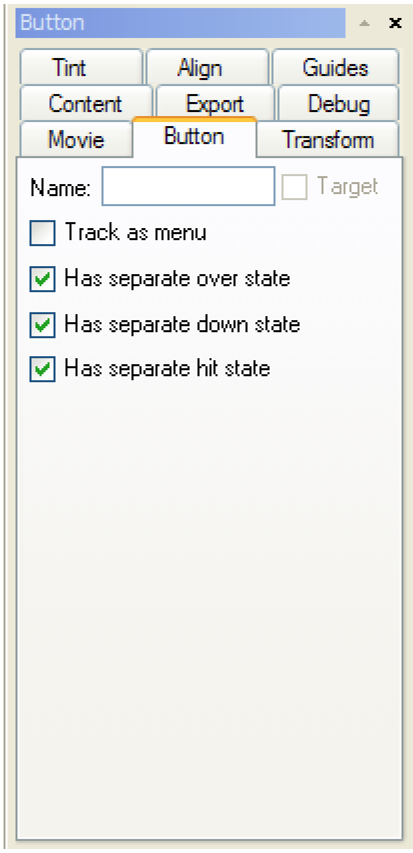
2. From the Main Menu, select Insert | Scene and a new Scene, named 'Scene_2', will be created. Check that Scene_2 appears in the Scene row on the Timeline Panel

3. Press the 'Insert Text' button on the Insert Toolbar. Change the word "Text" to "LINK 1", and the Font size to 12

4. Press the 'Convert to Button' button from the 'Grouping' Toolbar (if you do not have the grouping toolbar visible, you can use the Main Menu to select Modify | Convert | Convert to Button). From the 'Button Object' Panel select the 'Has separate over state', 'Has separate down state', and the 'Has separate hit state' boxes. A Button icon will appear in the Outline Panel. Click on the (+) icon next to the Button in the Outline panel to display its 'Object' Tree, with four child buttons indicating the button's four states

The 'Outline' and 'Button Object' Panels should look like this:





Name: □ Target

□ Track as menu

☑ Has separate over state

☑ Has separate down state

☑ Has separate hit state

The Up State represents the button's appearance when the mouse pointer is not over it. The Over State represents the button's appearance when the mouse pointer is over it. The Down State represents the button's appearance when the mouse is clicked. The Hit State represents the button's active region

5 From the button's 'Object' tree select LINK 1 from the button's Up State, and from the Text Panel, change the color from black to a dark green

6. Select LINK 1 from the button's Over State, and from the 'Text' Panel, change the color to a medium gray

7. Select LINK 1 from the button's Down State and change the color to a dark gray

8. Select the "Hit Rectangle" from the button's Hit State. You should see a black rectangle. This rectangle represents the button's active region - the area that will respond to the mouse cursor. This state is not visible when the Movie is playing

9. Press the Play Timeline button on the 'Control' Toolbar, and test your button. Your button should change to a gray when the mouse pointer is over it and to a dark gray when you press the button. Press the ■ 'Stop Movie' button to end your preview

10. Save your Movie

Adding Sounds to your Buttons

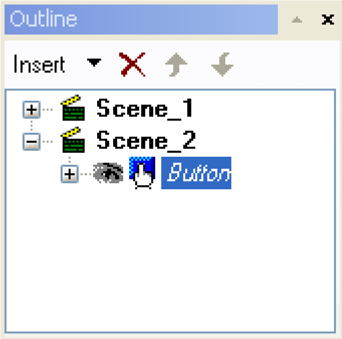
1. Continue from the section above, or alternatively you can open the file "myfirst.swi".

2. From the Outline Panel, select Scene_2

3. Select your 'LINK 1' button from the Outline Panel. Click on the minus (-) symbol to the left of the button icon, so that the button's states are not visible

Note: Make sure that none of the button's states are selected when you add your Actions, as you cannot add Actions to the button's state. Objects inside button states are not allowed to have Actions and SWiSH will ignore them

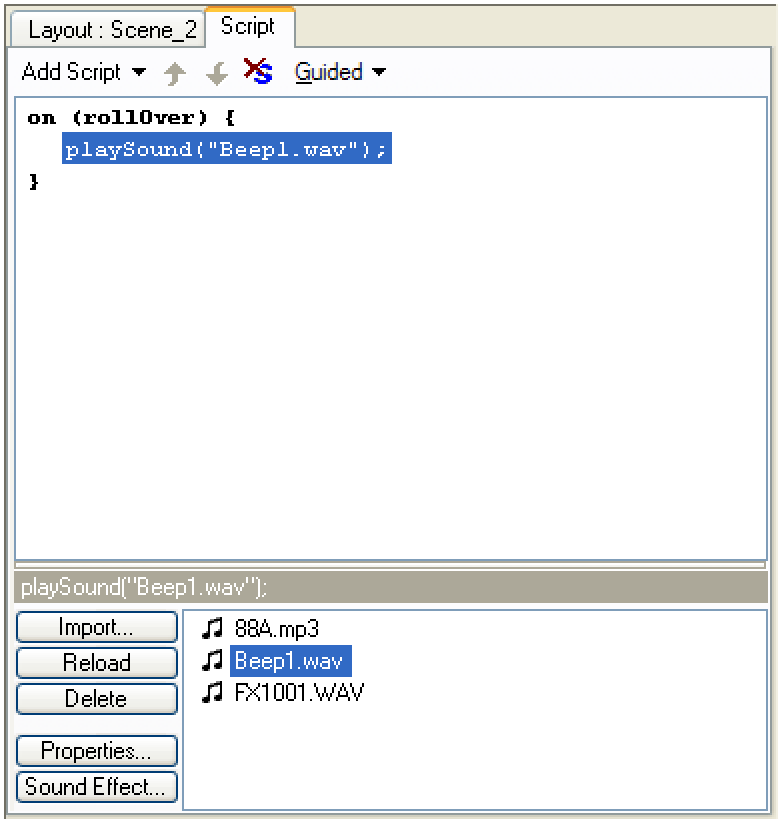
The 'Outline' Panel should look like this:



4. Select the Script Panel and press the 'Add Script' button and select Events | Button | on (rollOver) from the drop-down Menu. The On (Roll Over) Event will appear and be highlighted in the 'Script' window

5. Press the 'Add Script' button (or right click on the 'on (rollOver)' function) and select Sound | Play Sound from the drop-down Menu. Press the 'Import...' button and the 'Import Sound' dialog box will appear. Navigate to the Samples folder within your SWiSH application folder, select "Beep1.wav" and press Open. The filename "Beep1.wav" will appear at the bottom in the audio content list. Click on the filename within that list to select it for the playSound action.

The 'Script' Panel should look like this:



```
on (rollOver) {
    playSound("Beep1.wav");
}
```

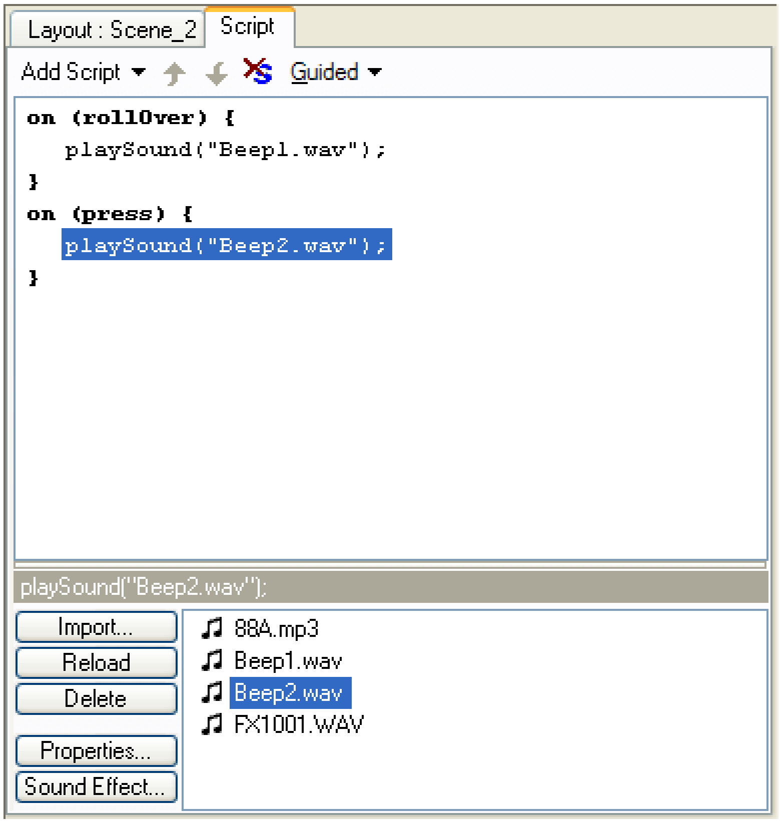
playSound("Beep1.wav");

6. Select the 'Layout' Panel tab and press the 'Play Timeline' button to test your button. When your mouse

pointer rolls over the button, you should hear the sound. Press the ■ 'Stop Movie' button

7. From the Script Panel, press the 'Add Script' button and select Events | Button | on (press)

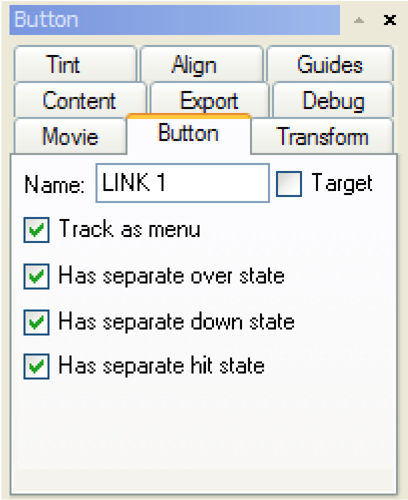
8. Press the Add Script ▾ 'Add Script' button and select Sound | Play Sound from the drop-down Menu. Press the 'Import...' button, and select "Beep2.wav" from the Samples folder. Import, then select Beep2.wav so it is the parameter of the playSound() Action. The 'Script' Panel should be as shown below:



```
on (rollOver) {
   playSound("Beep1.wav");
}
on (press) {
   playSound("Beep2.wav");
}
```

playSound("Beep2.wav");

9. Select the Button Panel. In the 'Name' text entry box, type in LINK 1, and check the Track as Menu box



Name: LINK 1 □ Target

☑ Track as menu

☑ Has separate over state

☑ Has separate down state

☑ Has separate hit state

10. Go back in to the 'Layout' Panel and press the 'Play Timeline' button to test your button. When your mouse pointer rolls over the button, you should hear the first sound you imported, and when you click the button, you should hear a second sound. Press the ■ 'Stop Movie' button to end your preview

11. Save your Movie

You are ready to continue with the Creating a Menu tutorial.

Creating a Menu

Top Previous Next

This is a step-by-step tutorial for creating a Menu. This tutorial illustrates some of the features seen in the "first.swi" file in the File | Samples | Tutorials Menu.

1. Open file "myfirst.swi" saved from the previous tutorial.

2. Check that Scene_2 appears in the Scene row on the Timeline Panel. If Scene_2 does not appear in the 'Timeline' Panel, select Scene_2 from the Outline Panel

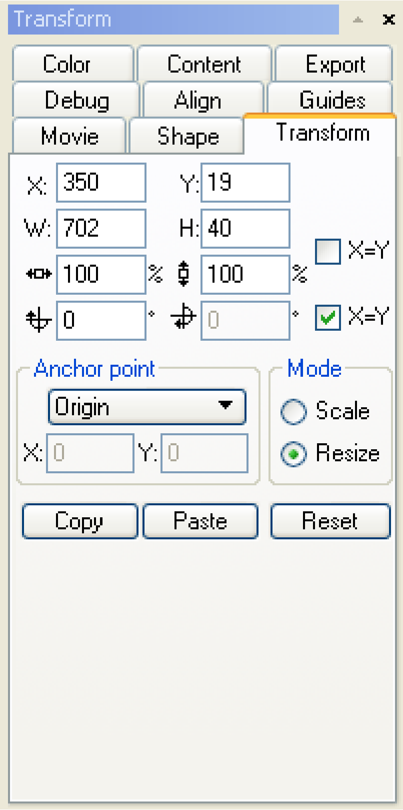
3. Press the 'Fit in Window' Zoom button to see the complete stage

Note: You can also Zoom In or Zoom out on the stage at any time by pressing the Zoom In button or the Zoom Out button

4. Select the Rectangle/Square Tool from the Toolbox. Click near the top left corner of the stage and drag the mouse down and right to create a rectangle

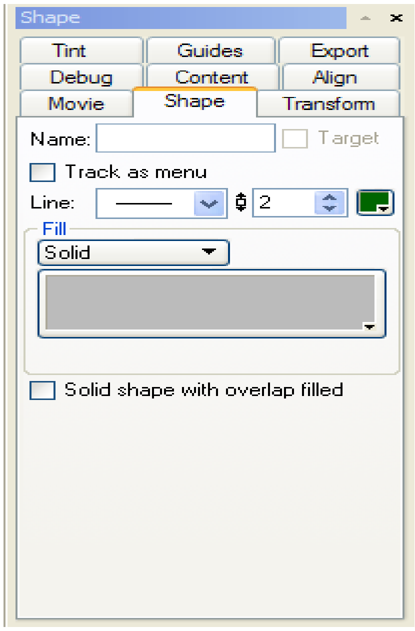
5. Select the resize tool option, if not already selected. This allows the Object dimensions to be altered in the Transform Panel without rescaling

6. Select the Transform Panel and change the width of this shape to W: 702 and the height of this shape to H: 40 (The 'X=Y' boxes should not be checked). At the top of this Panel there are 'X' and 'Y' co-ordinates indicating the Object's alignment. As the Movie's width is 702, the center of the object's X location should be 350. Change the value in the 'X:' box to 350, if necessary. As the object's height is 40, the center of its Y location should be 19. Change the value in the 'Y:' box to 19, if it is another value. This will ensure that there is no gap around the top or sides of the shape when played.The 'Transform' Panel should be as shown below:



X: 350 Y: 19
W: 702 H: 40
100 % 100 % □ X=Y
0 ° 0 ° ☑ X=Y
Anchor point: Origin X: 0 Y: 0
Mode: ○ Scale ● Resize

7. Select the Shape Object Panel and select a solid Line and set its width to 2 pixels. Change the Line color to a dark green and the Fill color to a light gray and uncheck the 'Track as Menu' option if it has been checked. The 'Shape Object' Panel should appear as shown below:



Name: □ Target
□ Track as menu
Line: 2
Fill: Solid
□ Solid shape with overlap filled

7. While the rectangle shape is selected, press 📄 to copy, and 📋 to paste your Object. It may paste directly over the original shape (depending on original location of the shape). Click on the new shape and drag it towards the bottom of the stage. Check the rectangle's X and Y position in the Transform Panel, and change to X: 350, Y: 431, if necessary

8.

Select the □ Rectangle/Square Tool from the Toolbox. Click over the left corner of the rectangle at the top of the stage, and drag the mouse down and right to create a rectangle that will completely cover both rectangles

9. Select the Transform Panel and change the width of this shape to W: 700 and the height to H: 450 (the 'X=Y' boxes should not be checked). Center the Object on the stage by changing the value in the 'X:' box to 350, and the value in the 'Y:' box to 225

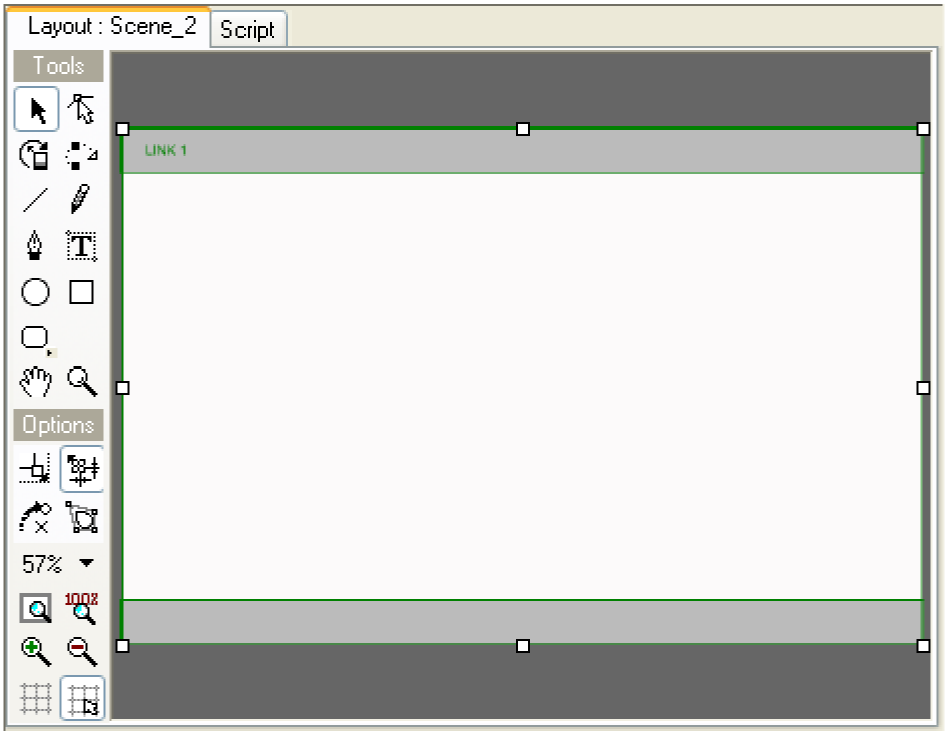
10. Select the Shape Panel and select a solid line border, and change the Line color to a dark green and the Fill to None

11. From the File Menu, select Test | Test in Player to check that your rectangles are placed correctly. Your introduction will play first and this Scene will follow. Close the 'Player' window

12. Select Scene_2 from the Outline Panel

13. Select the 'LINK 1' button and press the 'Bring to Front' button on the Standard Toolbar, then drag it over towards the top-left corner of the stage, placing it over the gray rectangle where desired

The 'Layout' Panel should look like this:



Before making a Menu, you may want to consider whether you want your buttons to look more or less the same and whether you want them to have similar behaviors. If you want your buttons to look uniform, it can save a lot of time if you create one button and then copy and paste as many of those buttons as needed, and then go into the different states to change the text. This way, the buttons will be consistent in size and color. If you want the buttons to have similar behaviors as well, you should add any sounds and perhaps some or all of their Actions before copying and pasting. You can always go into the different states and change the specific Goto Scene or URL references.

For this tutorial, all buttons will have the same sounds attached to the On(rollover), and on(Press) Events, but the other Actions will be different. As these sounds have already been attached, you can begin copying and pasting the 7 additional buttons we will be using

14. From the Outline Panel make sure that your button's states are not visible, clicking the minus (-) symbol to the

left of the main button's icon, if necessary

15. With the button selected, press the Control+C keys to copy your button, then select Scene_2 from the Timeline Panel

Note: It is important that the Scene (and not the button) is selected before you paste, as a button cannot be pasted inside of another button

16. Press 📋 to paste the copy of your button. Repeat this process until there are four buttons shown in the Outline column

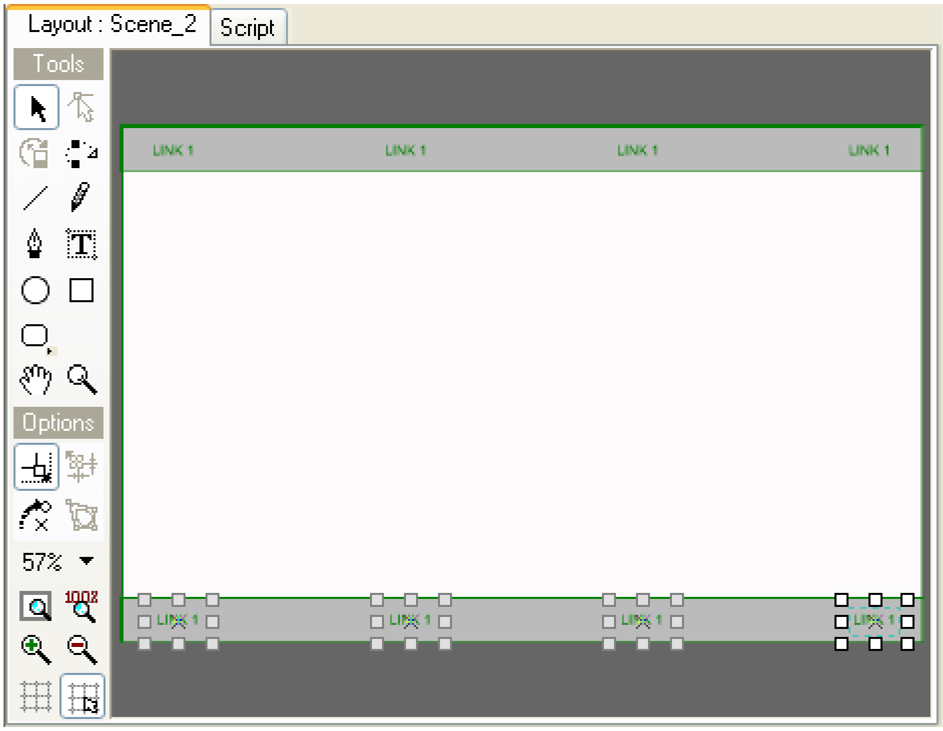
17. Select and move the last copied button, so that it is in the top-right-hand corner of the top rectangle. The other two buttons will be aligned to be evenly placed between this and the first button

18. To align the buttons horizontally, select all four buttons in the 'Outline' Panel by holding the CTRL key down and pressing the left mouse button. Once the four buttons are selected, use the Menu option Modify | Align | Relative to All Selected. Then use the Menu option Modify | Align | Distribute | Center (Horizontally). This will distribute the four buttons betwen the first and the last pasted buttons

19. To Align the buttons vertically, use the Menu option Modify | Align | Top, to align them to the top-most button

20. Use the copy 📄 button to re-copy the four aligned buttons then press the paste 📋 button to re-paste the 4 aligned buttons. Move these buttons as a group to the bottom rectangle

The 'Layout' Panel should look something like this:



21. From the File Menu, select Test | Test in Player to check that your buttons are placed correctly

22. When you are happy with the placement of your buttons, select the button to the right of the original 'LINK 1' button, and from the Button Panel change the name from "LINK 1" to "LINK 2"

23. From the Outline Panel, click on the plus (+) symbol to access "LINK 2" button's states. Change the text for

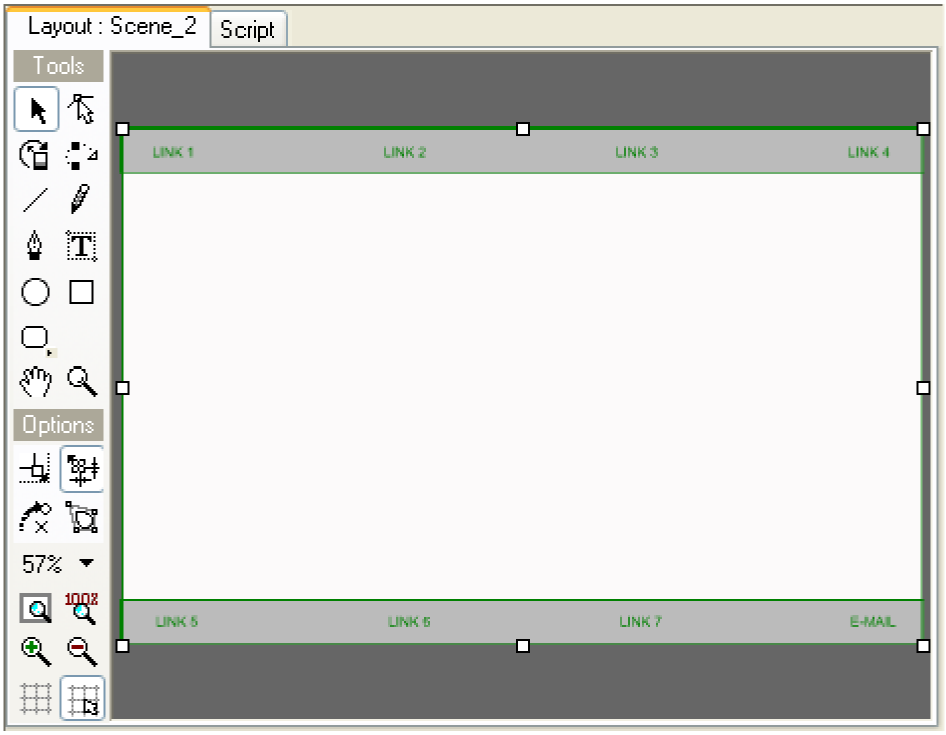
the Up, Over, and Down, states from "LINK 1" to "LINK 2"

24. Repeat steps 22 and 23, for the all but the button on the bottom right; increasing the number of each button by one, ie. LINK 3, LINK 4, LINK 5, LINK 6, LINK 7

25. For the bottom-right button, change the name from "LINK 1" to "E-MAIL".

Note: When you copy and paste a button then change the original text, the Hit Rectangle's size is not automatically updated. In many cases the Hit Rectangle will not be the same size as the text it is supposed to cover. To assure the hit rectangle is the same size as the text, select the text "E-MAIL" from the Up, Over, or Down state; Use the Transform Panel to copy the Width and Height of the text (W: and H:), then apply those settings to the 'Hit Rectangle' from the Hit State.

The 'Layout' Panel should now look something like this:



26. From the File Menu, select Test | Test in Player to check that your buttons are placed correctly. Position the mouse over the button and click on the button to ensure that the text was altered for each of the button states

27. When you are satisfied with your Menu, save your Movie

You are ready to continue with the Creating a Sprite tutorial.

Creating a Sprite

Top Previous Next

This is a step-by-step tutorial for creating a Sprite. This tutorial illustrates some of the features seen in the "first.swi" file in the File | Samples | Tutorials Menu.

A Sprite is a Movie within a Movie. Perhaps the greatest advantage of using Sprites in your Movies is that their Timeline runs independently of the main Movie's Timeline, as you will see in the "pop_up" Sprite you will be creating. Another advantage of using Sprites in your Movies is that you can easily reuse them throughout your Movie, saving time and with only a slight increase to the file size of the Movie.

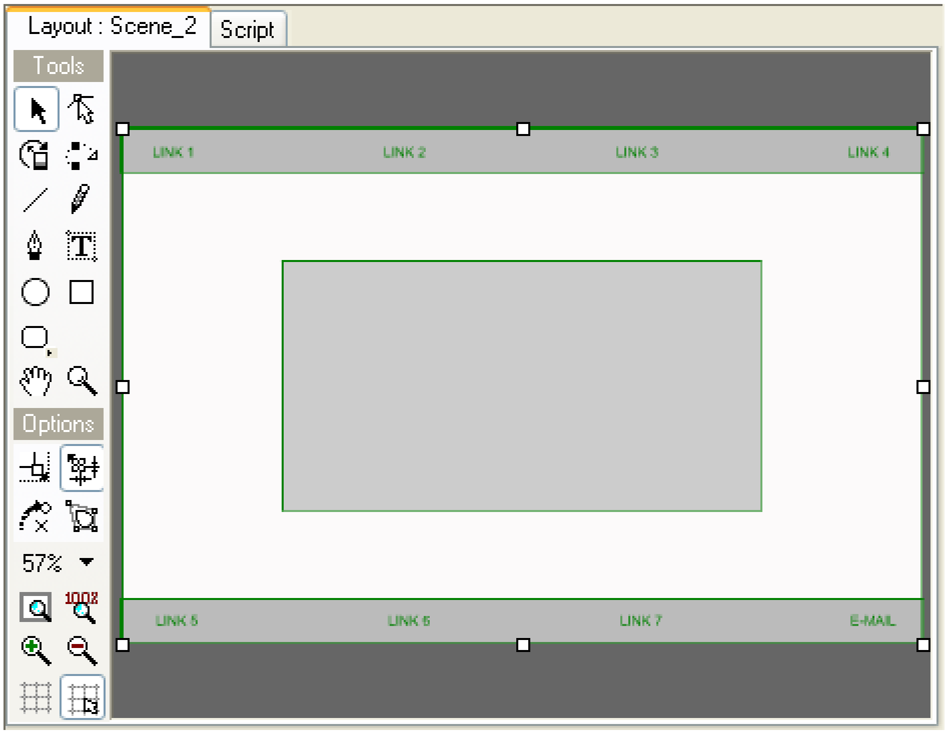
1. Open "myfirst.swi" saved from the previous tutorial.

2. Check that Scene_2 appears in the Scene row on the Timeline Panel. If Scene_2 does not appear in the 'Timeline Panel', select Scene_2 from the Outline Panel

3. Select the □ Rectangle/Square Tool from the Toolbox. Create a rectangle approximately 1/2 the height and width of the stage. From the Transform Panel, change the Width to 420 and the Height to 220, and center on the stage

4. From the Shape Panel select the solid line and set the line's width to 2. Change the line's color to a dark green. Select a solid fill in a light gray. Set the shape name to "window"

The Layout Panel should look something like this:



5. Select the new rectangle then from the Modify Menu, select Convert | Convert to Sprite. A Sprite icon will appear in the Timeline. Select the Sprite Panel and type "pop_up" in the Name edit box. Check the 'Stop Playing At End of Sprite' box

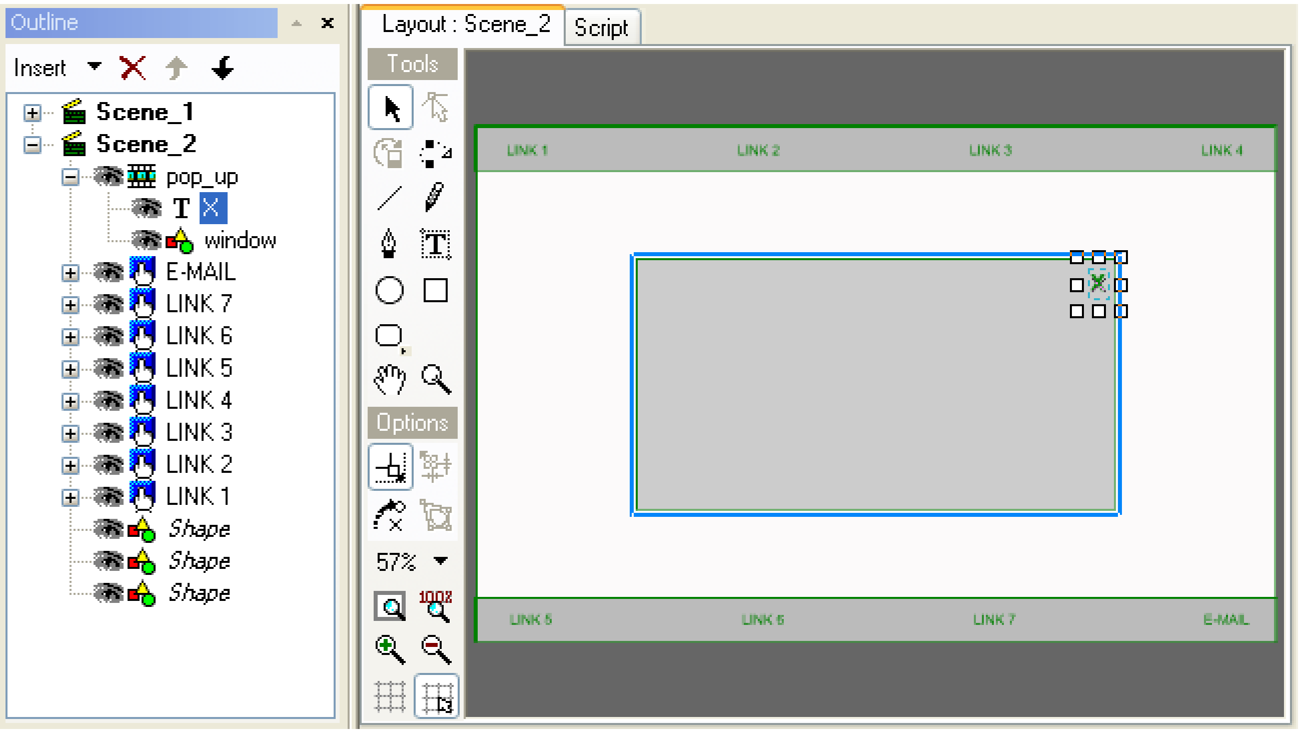
6. Select the Outline Panel and click on the plus (+) symbol next to the "pop_up" Sprite to display the Sprite's 'Object' tree

Note: In this view, the rectangle you have just created is visible in the 'Layout' Panel. In the 'Timeline' Panel, only the "pop_up" Sprite and the rectangle within this Sprite are present. Also note that the Sprite Object has its own Timeline. You can toggle this view at anytime by clicking on the plus (+) or minus (-) symbol in the 'Sprite Object' tree

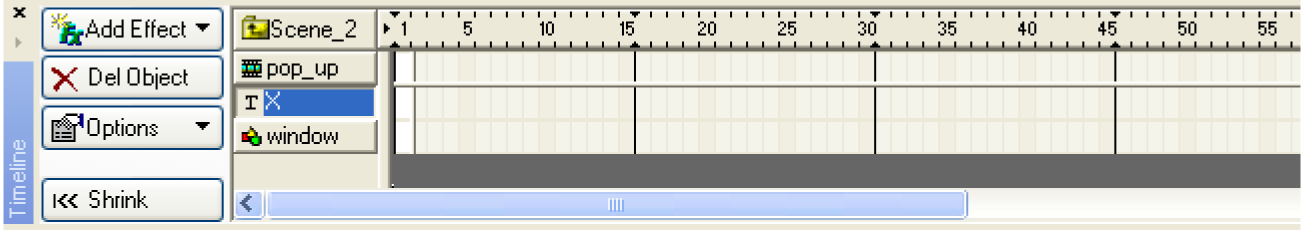
7. Make sure you are viewing the Sprite's timeline by pressing the (+) next to the "pop_up" Sprite in the Outline panel, and with the "pop_up" Sprite selected, press the 'Insert Text' button. From the Text Object Panel, change this text to the letter X, change the font to Arial and the size to 20, and change the color to a dark green.

8. Drag the X Text Object over to the top-right corner of the rectangle

The Outline and Layout Panel should look something like this:



The "pop_up" Sprite Object's Timeline should look like this:



9. Select the window object from the Timeline Panel. Click on Frame 2 and press the Add Effect ▼ 'Add Effect' button, select Fade | Fade In from the Menu. Double click on any Frame of the Fade In Effect in the Timeline. From the 'Fade In Settings' dialog box, change the Effect's duration value from 10 to 3 Frames, and press the 'Close' button to accept the new value

10. Select the X Object row from the 'Timeline' Panel and click on Frame 4. Press the Add Effect ▼ 'Add Effect' button, and select Place from the Menu

11. From the Insert Menu, select Content. In the 'Open' dialog box, from the 'Files of Type' drop-down Menu, select Plain Text (*.txt) and select "first.txt" from the Samples folder

12. From the Text Panel, change the font size to 11, and the paragraph alignment to ≡ left. Name this new Text Object "link 1" in the 'Text Object' Panel

13. On the Timeline Panel, click on Frame 4 in the link 1 Object row, press the Add Effect ▼ 'Add Effect' button and select Place from the Menu

14.

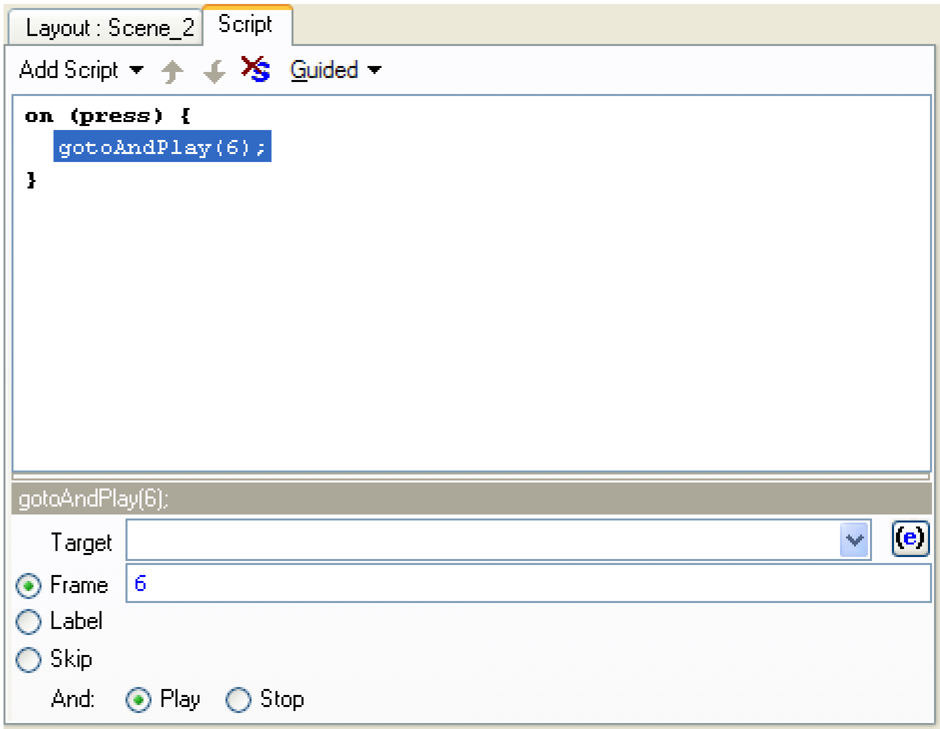
Click on Frame 5 in the "pop_up" Sprite Object row. The Add Effect ▾ 'Add Effect' button will change to an Add Script ▾ 'Add Script' button. Press the Add Script ▾ 'Add Script' button and select Movie Control | Stop from the Menu

15. Press the 'Play Timeline' button on the Control Toolbar. You should see your 'window' fade in quickly, and the 'X' and your imported text will show. Press the ■ 'Stop Movie' button on the Control Toolbar

16. Select the window Shape Object from the Timeline Panel and click on Frame 7. Press the Add Effect ▾ 'Add Effect' button, and select Fade | Fade Out from the Menu. In the 'Fade Out Settings' dialog box change the Effect's duration value from 10 Frames to 3. Press Close to accept the new value
17. Select the X Text Object from the Timeline Panel and click on Frame 7 in the in the X Object row. Press the Add Effect ▾ 'Add Effect' button and select Remove from the Menu

18. Select the link 1 Text Object from the Timeline Panel and click on Frame 7 in the in the link 1 Object row. Press the Add Effect ▾ 'Add Effect' button and select Remove from the Menu
19. Select the X Text Object from the Timeline Panel, then select the Script Panel. From the 'Script' Panel, press the Add Script ▾ 'Add Script' button and select the Menu options Events | Button | on (press)

20. Press the Add Script ▾ 'Add Script' button and use the Menu options: Movie Control | gotoAndPlay | gotoAndPlay(FRAME);. Leave the 'Target' box blank (implies this Sprite) and enter 6 into the Frame field. The 'Script' Panel should now look like the one below:



```
on (press) {
  gotoAndPlay(6);
}
```

gotoAndPlay(6);

Target

● Frame 6

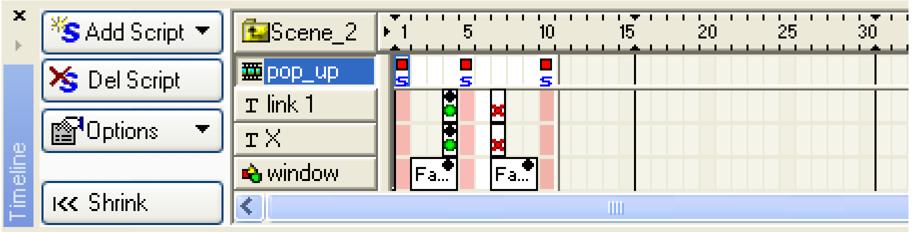
○ Label

○ Skip

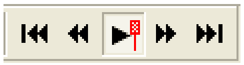
And: ● Play ○ Stop

21. On the 'Timeline' Panel, click on Frame 10 in the "pop_up" Sprite row. Press the Add Script ▾ 'Add Script' button and select Stop from the Menu
22. Click on Frame 1 in the "pop_up" Sprite row. Press the Add Script ▾ 'Add Script' button and select Stop from the Menu. This Action will hide the Sprite until it is played by the main Movie. This will be applied in the next tutorial.

The "pop_up" Sprite's Timeline should look like this:



23. To preview this Sprite from within the Sprite's Timeline, you will need to be in Preview Frame mode. Press the ▶ 'Preview Frame' button on the Control Toolbar. Click on Frame 1 and drag the Play Head right across the Timeline to preview the animation, or use the arrow controls on the 'Control' Toolbar either side of the ▶ 'Preview Frame' button

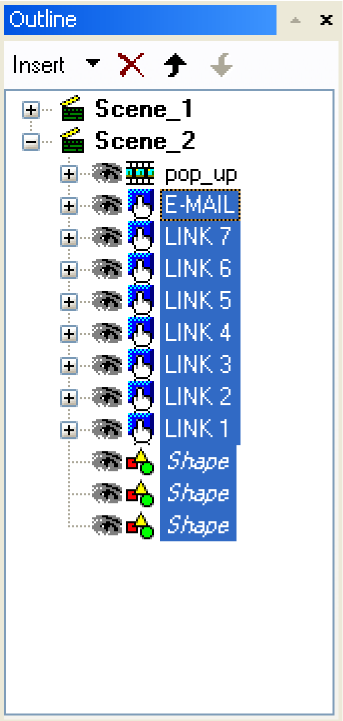


Advancing one Frame should show the window Object starting the fade in. Advancing to Frame 3 will show the link 1 Text Object and the X Object appear, and at Frame 6 the window Object should start to fade out as the link 1 Text Object and X Object disappear

Grouping a Sprite

1. The "pop_up" Sprite should be the topmost object listed under Scene_2 in the Outline Panel. Press the (-) button next to the name of the sprite in the Outline panel to close it. All other Objects in this Scene belong to your Menu and should appear from the second row and down in the Outline Panel. Select the Object in the second row of the Outline Panel (which should be the "E-MAIL" button), press and hold the Shift key and select the rectangle shape on the bottom row. As pressing and holding the Shift key allows you to select multiple Objects, these two Objects and the Objects in the rows between them should all be highlighted.

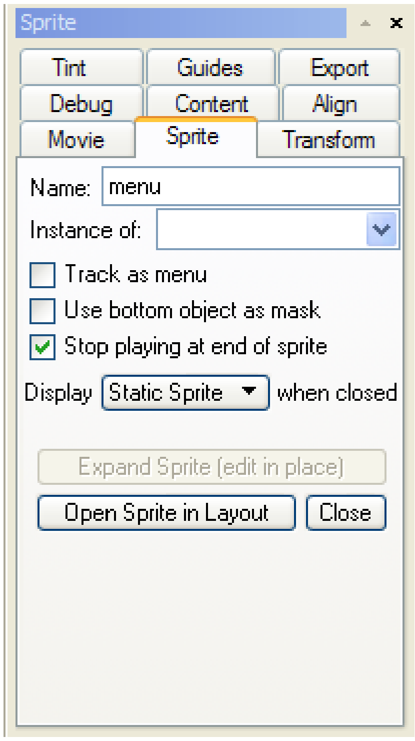
The Outline Panel should look something like this:



2. With these Objects selected, select Group | Group as a Sprite from the Modify Menu. This will create a Sprite Object consisting of every Object in your Menu

Note: If a number of Objects are selected and you select Convert | Convert to Sprite from the Modify Menu, you will create a separate Sprite for each Object selected

3. Select the Sprite Panel and type the word "menu" in the 'Name' edit box.Check the 'Stop playing at end of Sprite' box



Name: menu

Instance of:

□ Track as menu

□ Use bottom object as mask

☑ Stop playing at end of sprite

Display Static Sprite when closed

This Sprite will be controlled from the Movie's main Timeline, and our purpose for grouping and converting this Menu into a Sprite is twofold. Firstly, it will allow you to have this Menu fade in (rather than suddenly appearing after your intro/splash page) as a Sprite, so that you need only apply the Fade In Effect to the Sprite Object rather than applying this Effect to each Object within it. Secondly, this Sprite can be quickly copied and pasted and used later on in the Movie

4. Select Scene_2 from the top row of 'Timeline' Panel and select the Menu Sprite Object. Select Frame 1 and press the Add Effect ▾ 'Add Effect' button, select Fade In from the Menu.

5. Press the ▶ 'Play Movie' button on the Control Toolbar. Your Menu will fade in after your introduction and the Movie will stop. The "pop_up" Sprite will not show (we will use this Sprite later). Press the ■ 'Stop Movie' button on the Control Toolbar

6. Save your Movie

You are ready to continue with the Adding Interactivity to your Movie tutorial.

Adding Interactivity to Your Movie

Top Previous Next

This is a tutorial includes step-by-step instructions to creating links in your Movie. This tutorial illustrates some of the features seen in the "first.swi" file in the File | Samples | Tutorials Menu.

Linking to a Sprite and using the gotoAndPlay() Action

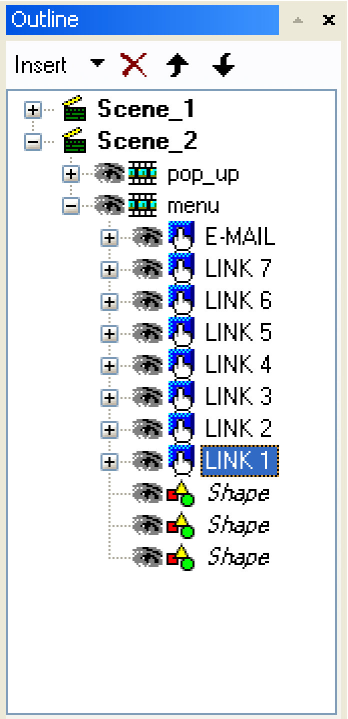
1. Open "myfirst.swi" saved from the previous tutorial.

2. Check that Scene_2 appears in the Scene row on the Timeline Panel. If Scene_2 does not appear in the 'Timeline' Panel, select Scene_2 from the Outline Panel. In the 'Scene' Object Panel, check the 'Stop playing at end of scene' checkbox

3. From the Outline Panel click on the plus (+) symbol next to the Menu Sprite Object and select the 'LINK 1' button. Make sure that the plus (+) symbol appears next to this button's icon, and that this button's states are not visible

Note: Make sure that none of the button's states are selected when you add your Actions, as you cannot add Actions to the button's state. Objects inside button states are not allowed to have Actions and SWiSH will ignore them

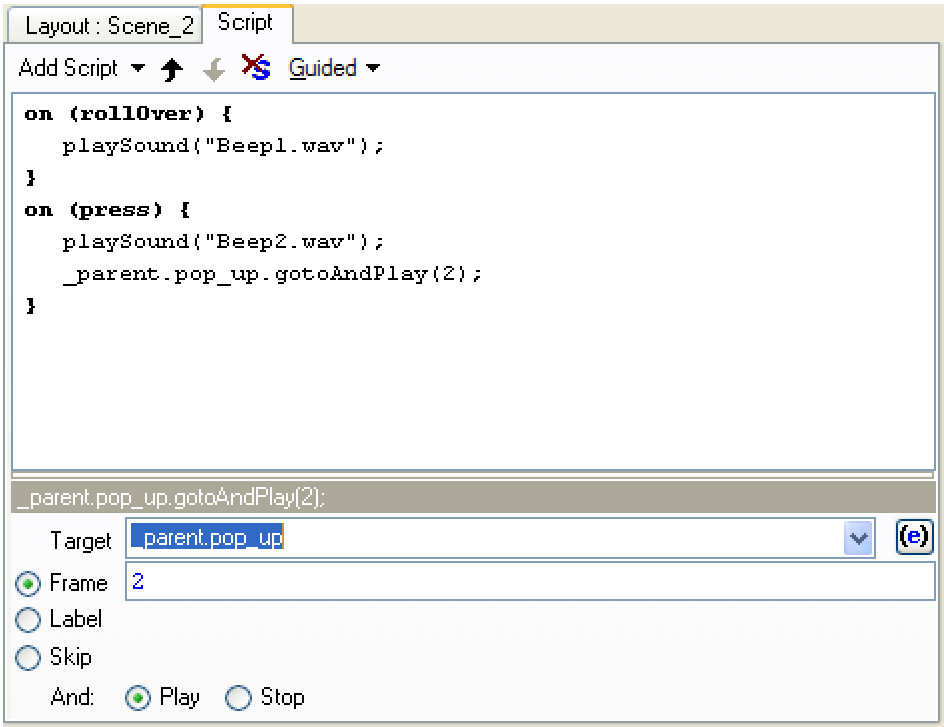
The Outline Panel should look like this:



The gotoAndPlay() Action is used to communicate with a Sprite and the Movie's Main Timeline. In the following example, we will use the gotoAndPlay() Action to call the pop-up Sprite

4. From the Script Panel, select the playSound("beep2.wav") Action in the on(Press) Event function. Press the Add Script 'Add Script' button, and select Movie Control | gotoAndPlay | gotoAndPlay(FRAME) from the Menu. Press the arrow to the right of the 'Target' name box and select "_parent.pop_up" from the Menu. Change the value from 1 to 2 in the 'Frame' box. Leave the 'Play' box checked.

The 'Script' Panel should look like this:



```
on (rollOver) {
   playSound("Beep1.wav");
}
on (press) {
   playSound("Beep2.wav");
   _parent.pop_up.gotoAndPlay(2);
}
```

_parent.pop_up.gotoAndPlay(2);

Target _parent.pop_up

● Frame 2

○ Label

○ Skip

And: ● Play ○ Stop

5. Select the Layout tab and Test your Action by pressing the ▶ 'Play' button to review the entire Movie. When the Movie ends, press the 'LINK 1' button. This should cause the text contained within the pop-up Sprite to be displayed. As this Scene will not loop, you can retest by pressing the X Text Object and the 'LINK 1' button again. Press the ■ 'Stop Movie' button

Note: The X Text Object can accept Actions, and behaves like a button without the different states

6. Select the pop-up Sprite from the Timeline then press the 'Copy' button. Select Scene_2 from the Timeline then press the 'Paste' button

Note:

- If you do not select Scene_2 on your 'Timeline' or 'Outline' Panels before pasting, you will paste the copy of your Sprite inside the original
- By copying and pasting your "pop_up" Sprite, you can save time using it as a Template. This can be done before you add your text, or you can change your text later using the original text as a guide for positioning your new text, then deleting the original

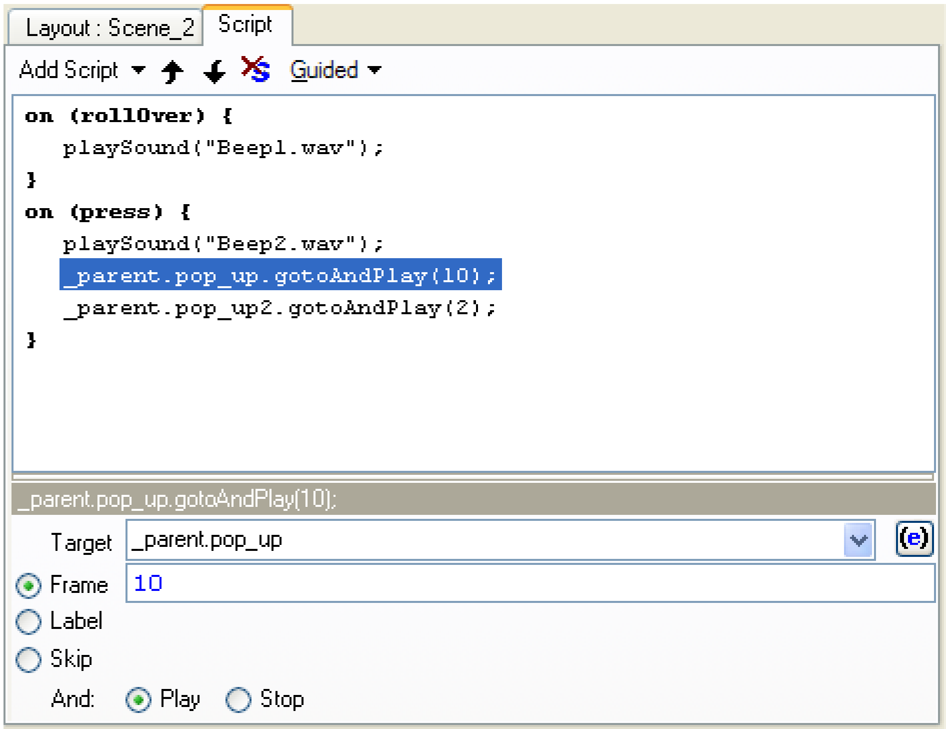
7. Select the Sprite Panel and in the 'Name' edit box. Change the name from "Copy_of_pop_up" to "pop_up2"

Note: Giving your Sprites unique names makes it easy to specify each one as an individual target which gives you more control over the actions you assign to each.

8. From the 'Outline' Panel click on the plus (+) symbol next to the Menu Sprite Object and select the 'LINK 2' button. Make sure that the plus (+) symbol appears next to this button's icon, and that this button's states are not visible

9. From the Script Panel, select the playSound("beep2.wav") Action in the on(Press) Event function. Press the Add Script ▼ 'Add Script' button, and select Movie Control | gotoAndPlay | gotoAndPlay(FRAME) from the Menu. Press the arrow to the right of the 'Target' name box and select "_parent.pop_up2" from the Menu.

Change the value from 1 to 2 in the 'Frame' box. Leave the 'Play' box checked. In a similar way, add the line "_parent.pop_up.gotoAndPlay(10);" above the current line. This will cause the LINK 1 dialog box (if displayed) to close. Your script should now look like this:



```
on (rollOver) {
   playSound("Beep1.wav");
}
on (press) {
   playSound("Beep2.wav");
   _parent.pop_up.gotoAndPlay(10);
   _parent.pop_up2.gotoAndPlay(2);
}
```

_parent.pop_up.gotoAndPlay(10);

Target _parent.pop_up

● Frame 10

○ Label

○ Skip

And: ● Play ○ Stop

10. From the 'Outline' Panel click on the plus (+) symbol next to the pop_up2 Sprite to access the Sprite's child objects. Select the link 1 Text Object, and from the 'Text' Panel, select all of the text within the text edit box and type "This is different text to show that the 'LINK 2' button is linked to a different Sprite." You may want to split this text onto a number of lines. Name the text "LINK 2"

11. Test your scripts for the 'LINK 1' and 'LINK 2' buttons by selecting Scene_2 in the 'Outline' Panel and pressing the 'Play Timeline' button and then pressing the 'LINK 1' and 'LINK 2' buttons. Press the ■ 'Stop Movie' button

12. Save your Movie

Linking to another Scene

1. Select Scene_2 from the Outline Panel. From the Edit Menu select Copy Scene and then Edit Menu | Paste Scene. A new Scene will be created. Click on the Scene name to rename the new Scene to "Scene_3"

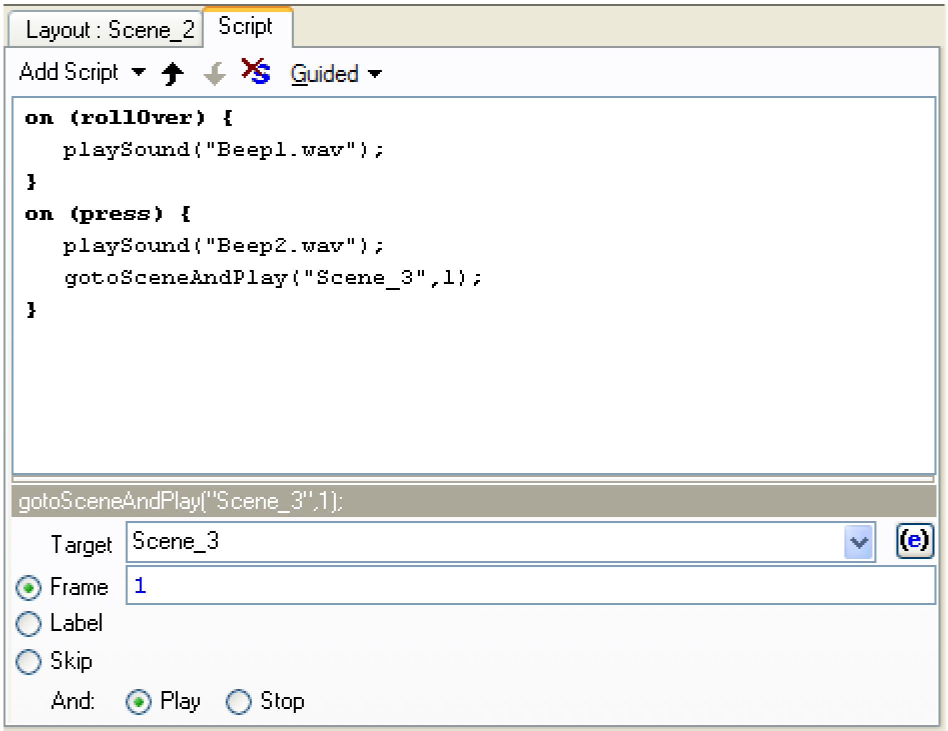
2. From the Scene Panel, change the background color to a light green by clicking on the color swatch and selecting the color from the Color Palette. Note that you should select an alpha value greater than 0%, otherwise the colour will not be visible

3. Select Scene_2 from the Outline Panel and Left-Click on the (+) symbol to expand this Scene.

4. From the 'Outline' Panel click on the plus (+) symbol next to the Menu Sprite and select the 'LINK 3' button. Make sure that the plus (+) symbol appears next to this button's icon, and that this button's states are not visible

5. From the 'Script' Panel select Play Sound "beep2.wav" under the On (Press) Event. Press the Add Script ▾ 'Add Script' button, and select Movie Control | gotoAndPlay | gotoAndPlay(FRAME) from the Menu. Press the arrow to the right of the 'Target' name box and select Scene_3 from the Menu. Enter 1 into the 'Frame' field.

Leave the 'Play' box checked. Your script should be as shown below:



```
on (rollOver) {
   playSound("Beep1.wav");
}
on (press) {
   playSound("Beep2.wav");
   gotoSceneAndPlay("Scene_3",1);
}
```

gotoSceneAndPlay("Scene_3",1);

Target Scene_3

● Frame 1

○ Label

○ Skip

And: ● Play ○ Stop

6. Test your Action for the 'LINK 3' button by selecting File | Test | Test in Player and pressing the 'LINK 3' button. Close the Player window

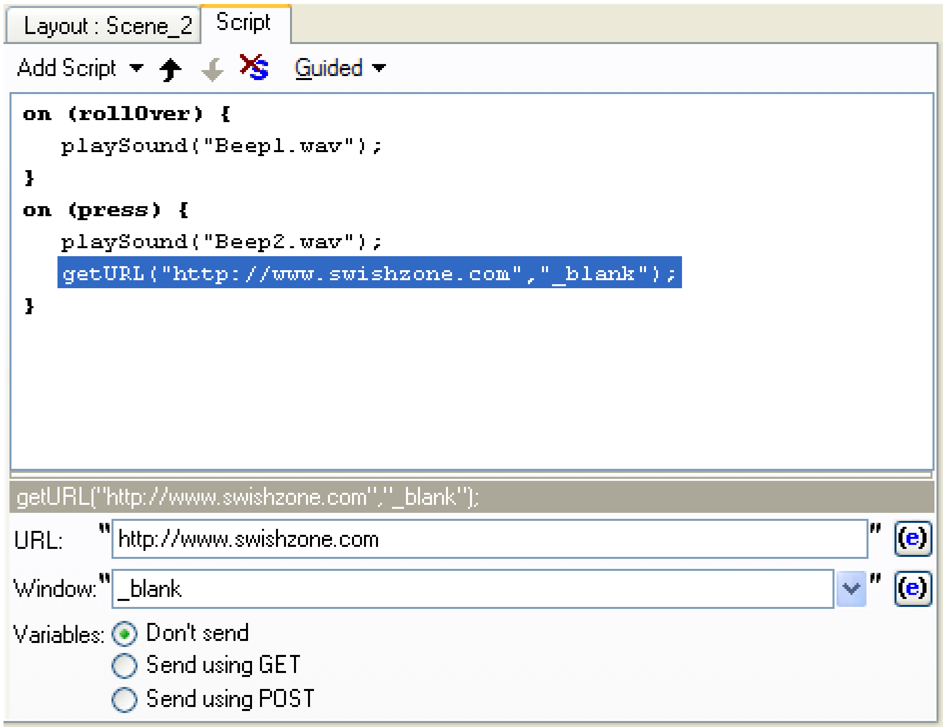
Note: If the movie went straight to Scene_3 without giving you the chance to test 'LINK 3' - make sure you have checked the option "Stop playing at end of Scene" on the Scene panel for Scene_2.

7. Save your Movie

Linking to another Web page

1. From the 'Outline' Panel click on the plus (+) symbol next to the Menu Sprite in Scene_2 to access the Sprite's child objects. Select the 'LINK 4' button. Make sure that the plus (+) symbol appears next to this button's icon, and that this button's states are not visible

2. From the 'Actions' Panel select Play Sound "beep2.wav" under the On (Press) Event. Press the Add Script 'Add Script' button, and select Browser/Network | Get URL from the Menu. In the 'URL' edit box, type "http://www.swishzone.com". For the 'Window' box select _blank from the pull-down Menu. Your script should be as shown below:



```
on (rollOver) {
   playSound("Beep1.wav");
}
on (press) {
   playSound("Beep2.wav");
   getURL("http://www.swishzone.com","_blank");
}
```

getURL("http://www.swishzone.com","_blank");

URL: "http://www.swishzone.com"

Window: "_blank"

Variables: ● Don't send
○ Send using GET
○ Send using POST

Note: When playing Movies within SWiSH, you can use the Tools Menu | Preferences | Player, and select the option at the bottom to have the player launch a browser for testing getURL() actions.

3. Test your Action for the 'LINK 4' button by selecting File | Test | In Browser and pressing the 'LINK 4' button.

If you are online, you will be brought to the SWiSH web site. This page will open in a new window

4. Save your Movie

Linking to the Visitor's E-mail Client

1. From the 'Outline' Panel click on the plus (+) symbol next to the Menu Sprite in Scene_2 to access the Sprite's child objects. Select the 'E-MAIL' button. Make sure that the plus (+) symbol appears next to the button's icon, and that the button's states are not visible

2. From the 'Actions' Panel select Play Sound "beep2.wav" under the On (Press) Event. Press the Add Script ▾ 'Add Action' button, and select Browser/Network | Mailto from the Menu. In the 'To' edit box, type your e-mail address e.g., me@myaccount.com

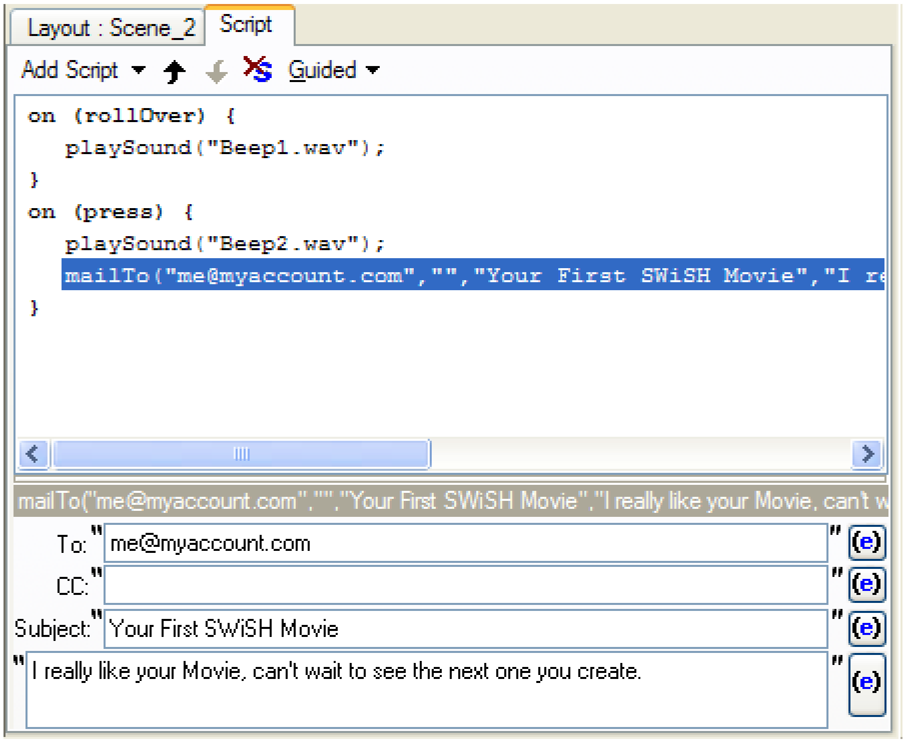
3. Leave the 'CC' field blank (this can be used to copy emails to other recipients)

4. In the 'Subject' edit box, type "Your First SWiSH Movie"

5. The window below the 'Subject' edit box is the message body area. Type "I really like your Movie, can't wait to see the next one you create." in the 'Field' edit box

When the Mailto Action is executed, the Flash Player will start the default e-mail client. The fields you have specified will be already filled in. The user can edit any of these or fill in any that you may leave blank before sending the message.

6. The Script panel should appear similar to this:



```
on (rollOver) {
   playSound("Beep1.wav");
}
on (press) {
   playSound("Beep2.wav");
   mailTo("me@myaccount.com","","Your First SWiSH Movie","I re
}
```

mailTo("me@myaccount.com","","Your First SWiSH Movie","I really like your Movie, can't w

To: "me@myaccount.com"

CC: ""

Subject: "Your First SWiSH Movie"

"I really like your Movie, can't wait to see the next one you create."

7. Save your Movie

You are ready to continue with the Creating a Preloader tutorial.

Creating a Preloader

Top Previous Next

This is a step-by-step tutorial for creating a Preloader. This tutorial illustrates some of the features seen in the "first.swi" file in the File | Samples | Tutorials Menu.

A Preloader is a looping animation that plays while the rest of the Movie loads into your visitor's browser.

1. Open "myfirst.swi" saved from the previous tutorial.

2. Press Insert Scene on the Insert Toolbar

3. Press the 'Bring to Front' button on the Standard Toolbar

4. Select the Scene Panel, and change the name to "Preloader" in the 'Scene Name' edit box

5. Press Insert Text on the Insert Toolbar

6. Select the Text Panel and change the word "Text" to the word "LOADING" in the 'text' window. Change the font size to 20, and the color to a dark green

7. In the Timeline Panel, click on Frame 2 in the Loading Text Object's row. Press the Add Effect ▾ 'Add Effect' button and choose Fade | Fade In from the Menu

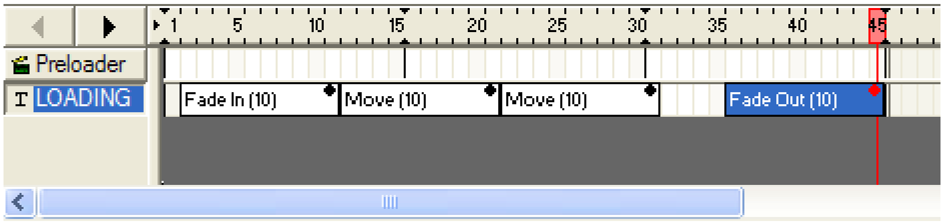
8. In the Timeline Panel, click on Frame 12 in the Loading Text Object's row. Press the Add Effect ▾ 'Add Effect' button and select Move from the Menu. Right-click on the Effect in the Timeline and select properties. From the Motion Tab in the Move Settings dialog box, select Color | Fade to Color, choose a medium gray from the Color Palette, and leave the value set to 100 in the '%' edit box. Press Close to accept the changes

9. In the Timeline Panel, click on Frame 22 in the Loading Text Object's row. Press the Add Effect ▾ 'Add Effect' button again and select Move from the Menu. From the Motion Tab in the Move Settings dialog, select Color | Fade to Color. Choose any color you wish from the Color Palette and change the value to 0 in the 'Fade to Color %' edit box

Note: Entering the 0% value in the 'Fade to Color %' edit box will cause the color to fade to the original dark green again. This can be very useful when you want to display an image or shape in its original color after having changed the color value to tint it. 100% means replace the original color completely with the new color, 0% means no change to the original color

10. Click on Frame 36 in the Loading Text Object's row. Press the Add Effect ▾ 'Add Effect' button and choose Fade Out from the Menu

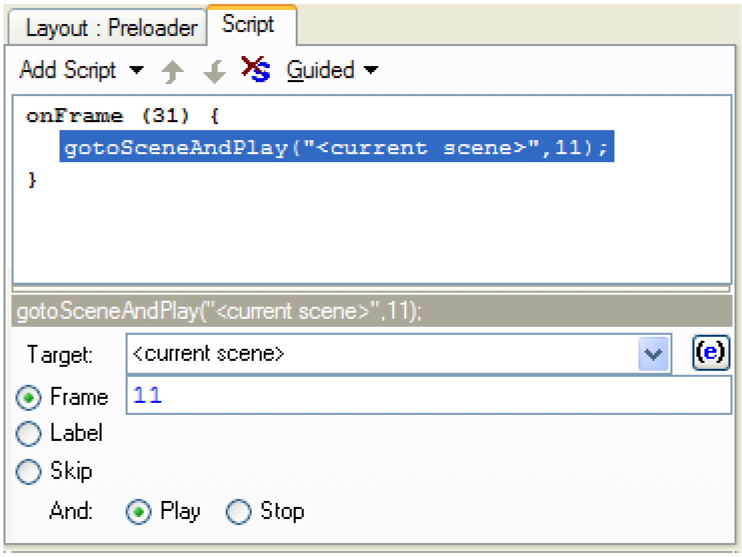
The 'Timeline' Panel should now look like this:



11. Press the 'Play Timeline' button. Your text should fade in then fade to gray and back to green again; then your text will fade out. Press the ■ 'Stop Movie' button

12. Select the Preloader Scene row from the Timeline. Right-click on Frame 31 and select Movie Control | gotoAndPlay | gotoAndPlay(FRAME) from the context Menu. A place marker for the Go to Frame Action will appear on the Timeline, and the script will appear in the window of the Script Panel. From the 'Script' Panel, change the value in the 'Frame' edit box from 1 to 11

The 'Script' Panel will look like this:



```
onFrame (31) {
    gotoSceneAndPlay("<current scene>",11);
}
```

gotoSceneAndPlay("<current scene>",11);

Target: <current scene>

● Frame 11

○ Label

○ Skip

And: ● Play ○ Stop

13. Press the ⚑ 'Play Timeline' button. Your text should fade in then fade to gray and back to green again. Your text will continue fading from gray to green and from green to gray until you press the ■ 'Stop Movie' button. Press the ■ 'Stop Movie' button

14. Press the ▶ 'Play Movie' button. Your "LOADING" text will continue to loop - it will not fade out and your introduction will not begin. Press the ■ 'Stop Movie' button

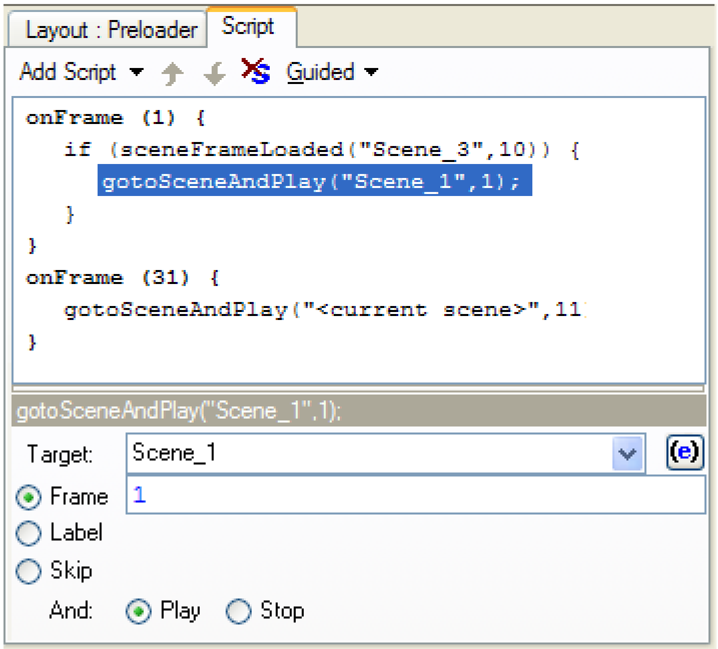
Note:

- You have created the looping animation for your Preloader. Now you need to instruct your Movie to loop this animation until your Movie, or a portion of it, has preloaded into the viewer's browser cache. The instructions that the Player needs can be added to your Movie with the If Frame Loaded and the Go to Frame etc Scripts
- If your Movie is in the viewers cache, there is no need to have them view this looping animation at all. You may instruct your Movie to go to Scene_1 rather than playing the Preloader Scene in the first Frame of the Preloader Scene as in this example

15. Select the Preloader Scene row from the Timeline. Right-click on Frame 2 and select Conditional | If Frame Loaded from the context Menu. This section will be added to and highlighted in the 'Script' Panel window. This Action will be highlighted. Change the value from 1 to 10 in the 'Frame' edit box. From the Target drop-down Menu, select Scene_3

16. Press the 'Add Script' button and select Movie Control | gotoAndPlay | gotoAndPlay(FRAME) from the Menu. Leave the Frame value at 1. Select Scene_1 from the Target drop-down Menu

The 'Script' Panel should look like this:



```
onFrame (1) {
   if (sceneFrameLoaded("Scene_3",10)) {
      gotoSceneAndPlay("Scene_1",1);
   }
}
onFrame (31) {
   gotoSceneAndPlay("<current scene>",11
}
```

gotoSceneAndPlay("Scene_1",1);

Target: Scene_1

● Frame 1

○ Label

○ Skip

And: ● Play ○ Stop

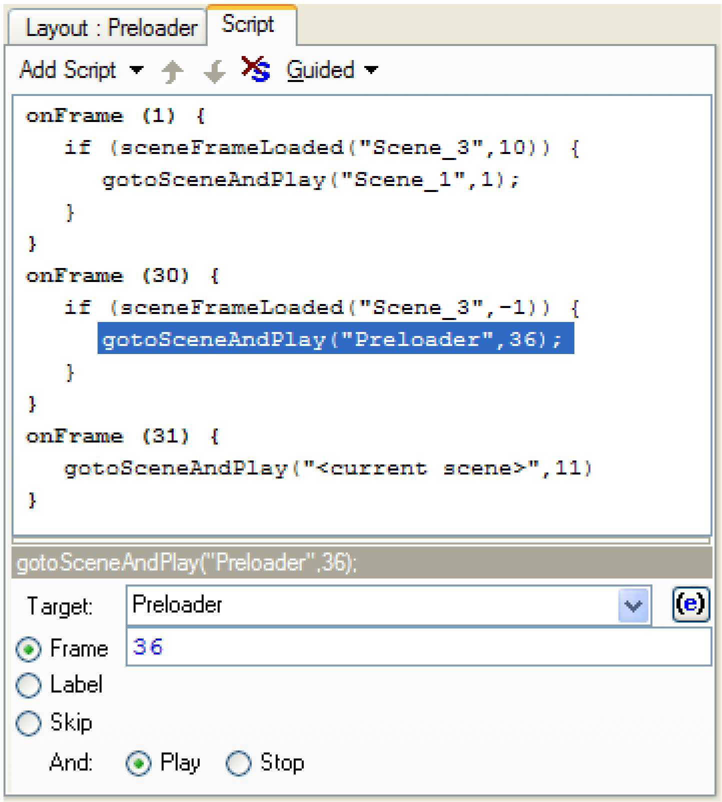
You need to add one more Action for your Preloader to function properly

17. Right-click on Frame 30 in the Preloader Scene row and select Conditional | If Frame Loaded from the Menu. Change the value to -1 in the 'Frame' edit box, and select Scene_3 from the Target drop-down Menu. Press Add Script again and select Movie Control | gotoAndPlay | gotoAndPlay(FRAME) from the Menu. Change the value from 1 to 36 in the 'Frame' edit box. Leave the Scene set to Preloader

Note: The -1 value refers to the last Frame of a Scene. You are instructing the player to download the entire

Movie before playing

The 'Script' Panel should look like this:



```
onFrame (1) {
   if (sceneFrameLoaded("Scene_3",10)) {
      gotoSceneAndPlay("Scene_1",1);
   }
}
onFrame (30) {
   if (sceneFrameLoaded("Scene_3",-1)) {
      gotoSceneAndPlay("Preloader",36);
   }
}
onFrame (31) {
   gotoSceneAndPlay("<current scene>",11)
}
```

gotoSceneAndPlay("Preloader",36);

Target: Preloader

● Frame 36

○ Label

○ Skip

And: ● Play ○ Stop

Note: When using a gotoAndPlay(FRAME) action and chosing a Scene from the Target drop-down menu, you can select a Scene by name e.g., Scene_3 as in this example, or choose a relative Scene name such as _LAST_SCENE_ from the Menu

18. Press the ▶ 'Play Movie' button. You will not see the Preloader Scene; as this is a local file your entire Movie is loaded instantaneously. Press the ■ 'Stop Movie' button

19. Save your Movie

You are now ready to continue with the Exporting your movie and Getting it Ready for the Web tutorial.

Exporting your Movie and Getting it Ready for the Web

Top Previous Next

This is a step-by-step tutorial for exporting your Movie.

This tutorial is saved as "first.swi" in the File | Samples | Tutorials Menu.

1. Open "myfirst.swi" saved from the previous tutorial.

2. From the File Menu, select Export | HTML + SWF. From the 'Publish SWiSH Movie' dialog box, locate the

folder you wish to save your files to, and press Save. This will export your Movie in the .swf file format (which the Flash Player will read) and the HTML page that your Movie will be embedded in.

Note: SWiSHmax will ask you if you want to edit the HTML file that was just exported. For this example, simply select the 'No' option. The purpose of this option is so that you can manually edit the HTML code tags in your default HTML editor (it will open the file in Windows Notepad if no other default editor is set). You can edit the title of the webpage, add META tags, etc.

3. Upload the .swf and .htm files to your server. Your Movie is now ready to play on the Web

Note : You can find more information on the options available when publishing your Movies in the Export Panel section of this manual. You will also find that the section Using SWiSH with Other Programs covers the Export options and parameters in some detail, particularly in the tutorial Using SWiSH with Dreamweaver

To view your Movie as it will be seen on the Web, locate and open "myfirst.html". This will play your Movie in your default browser.

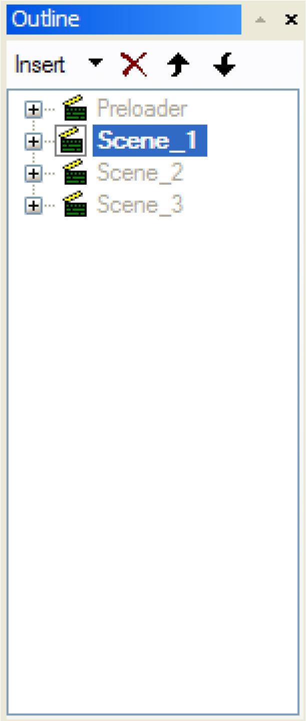
Using Actions to Control Your Movie

Top Previous Next

This is a step-by-step tutorial on how to add Actions to your SWiSH Movie. This tutorial is saved as "Actions.swi" in the File | Samples | Tutorials Menu.

1. Open "first.swi" from the File | Samples | Tutorials Menu

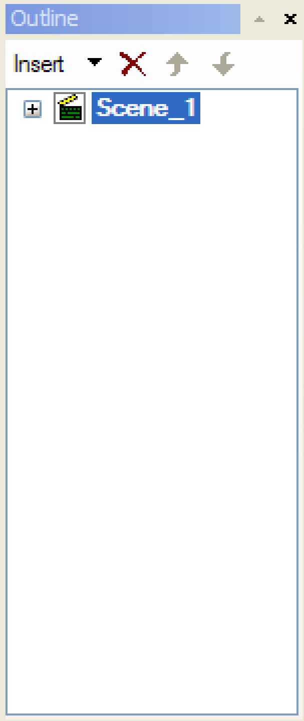
2. On the 'Outline' Panel, select Scene_1, click on the (-) symbol next it to close the Scene, and copy to the Clipboard by holding down the Control while pressing the C key (Ctrl+C) on your keyboard



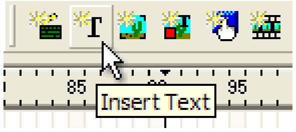
3. Select File | New and close "first.swi" without saving changes. Adjust your new Movie stage size to match "first.swi" by selecting the Movie Panel and changing the Width to 700, Height to 450

4. In the new Movie, delete the default Scene_1 from the Outline panel. Paste in Scene_1 from the Clipboard by selecting Edit | Paste from the Main Menu or CTRL+V on your keyboard. This should only leave one instance of

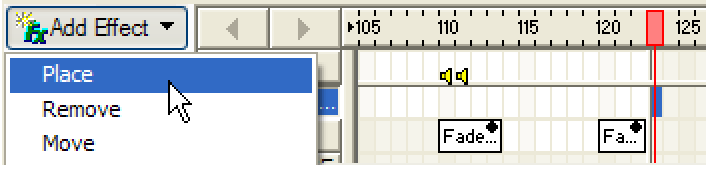
Scene_1 displayed with a (+) next to it as shown in the image below.



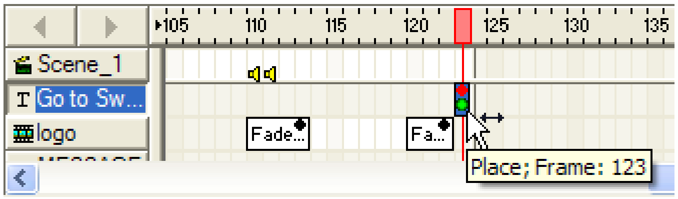
5. Add a new Text Object by pressing the 'Insert Text' button or by selecting Insert | Text from the Main Menu. Change the size to 36, and change the text to read "Go to Swishzone"



6. On the 'Timeline' Panel click on Frame 123 in the Timeline row labeled 'Go to Swishzone'. Press the Add Effect ▾ 'Add Effect' button and add a Place Effect.

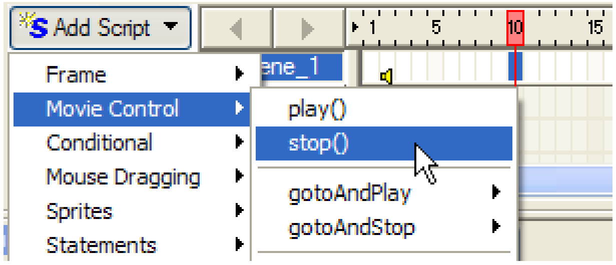


You will notice that a Place Effect marker 🚦 is now visible at Frame 123



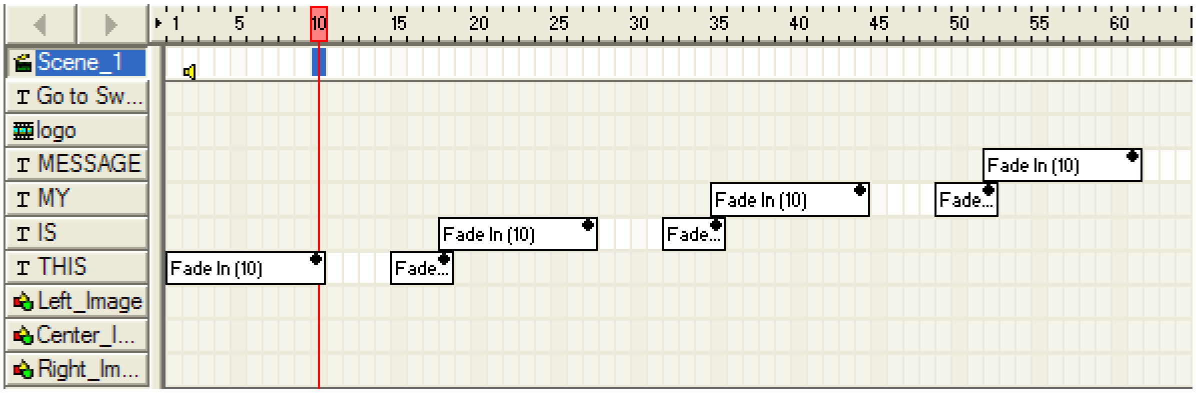
7. Select the 'Timeline' Panel tab and select Frame 10 in the row labeled Scene_1 ('Scene' is always the top row in the Timeline). The 'Add Effect' button should now be replaced with an 'Add Script' button. Press the 'Add

Script' button and select Movie Control | Stop



Note: If the 'Add Script' button does not appear - make sure you have selected a frame from the row just below the ruler indicating the frame numbers.

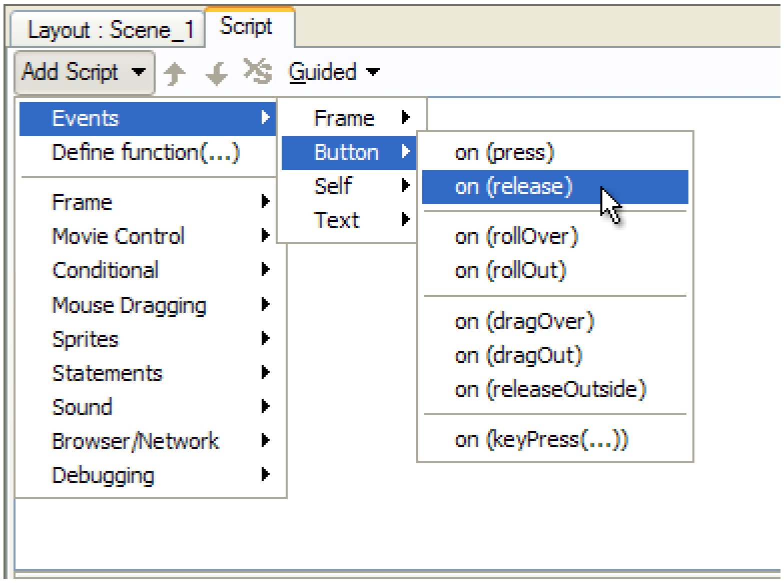
The Timeline should now look like this:



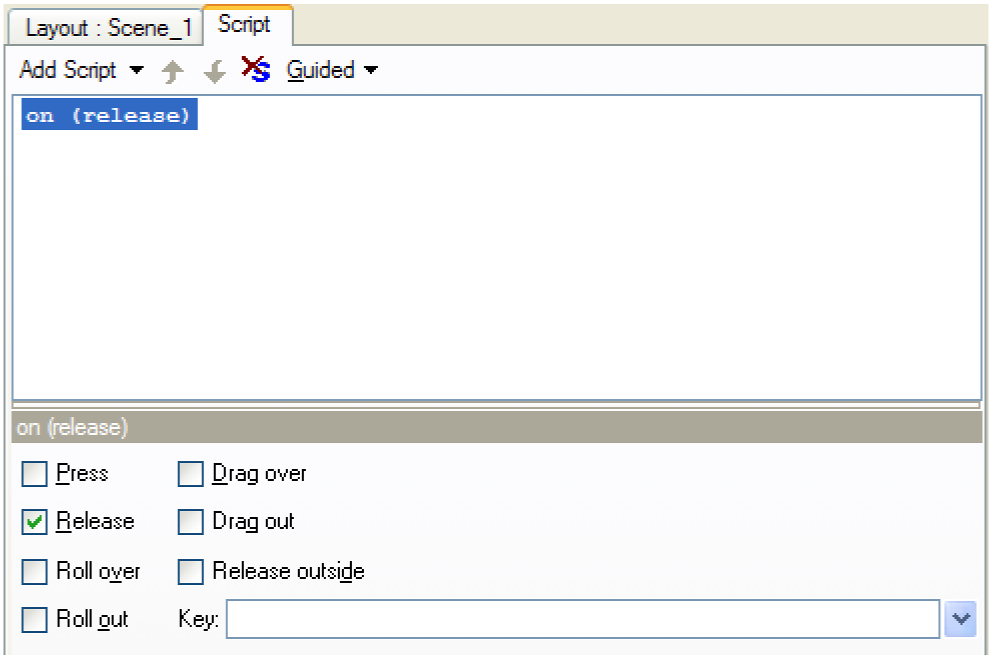
Note: When you move your mouse over the Action marker you should see a ToolTip indicating the Stop Action

8. Now select the "THIS" Text Object from the 'Outline Panel' then click on the Script Panel to open it

9. Press the 'Add Script' button and choose Events | Button | On Release from the Menu.



The 'Script Panel' should look like this

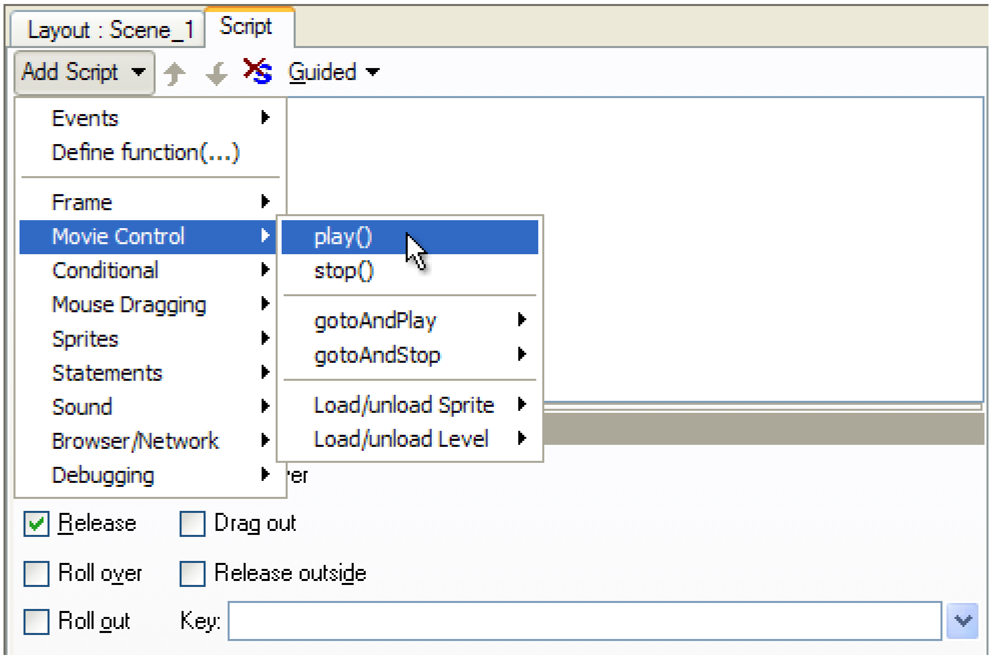


```
on (release)
```

on (release)

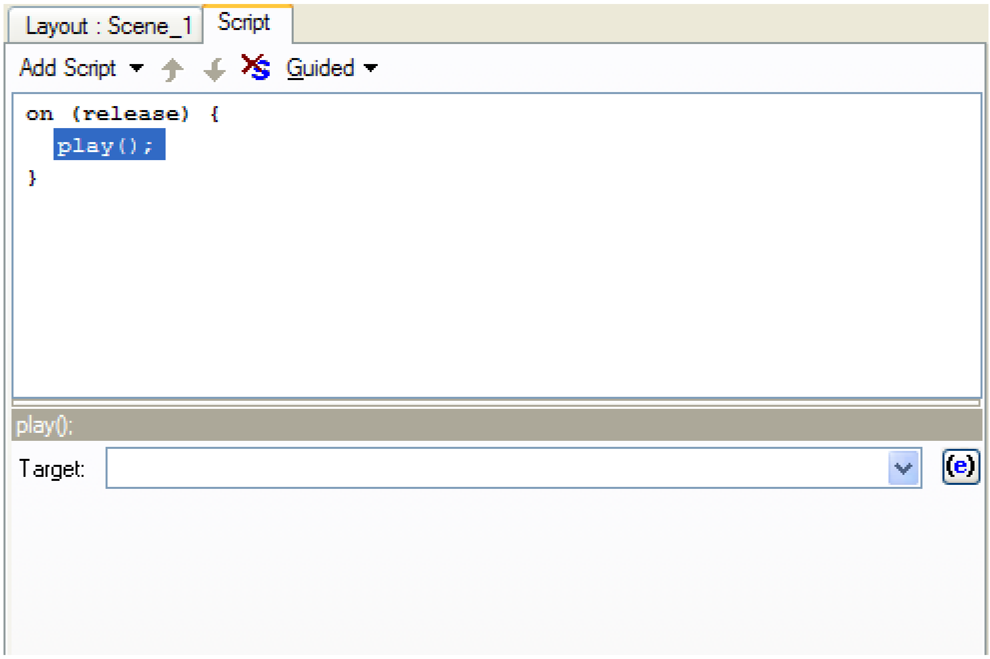
☐ Press ☐ Drag over
☑ Release ☐ Drag out
☐ Roll over ☐ Release outside
☐ Roll out Key:

10. Press 'Add Script' again and select Movie Control | Play from the menu



☑ Release ☐ Drag out
☐ Roll over ☐ Release outside
☐ Roll out Key:

The 'Script Panel' should now look like this:

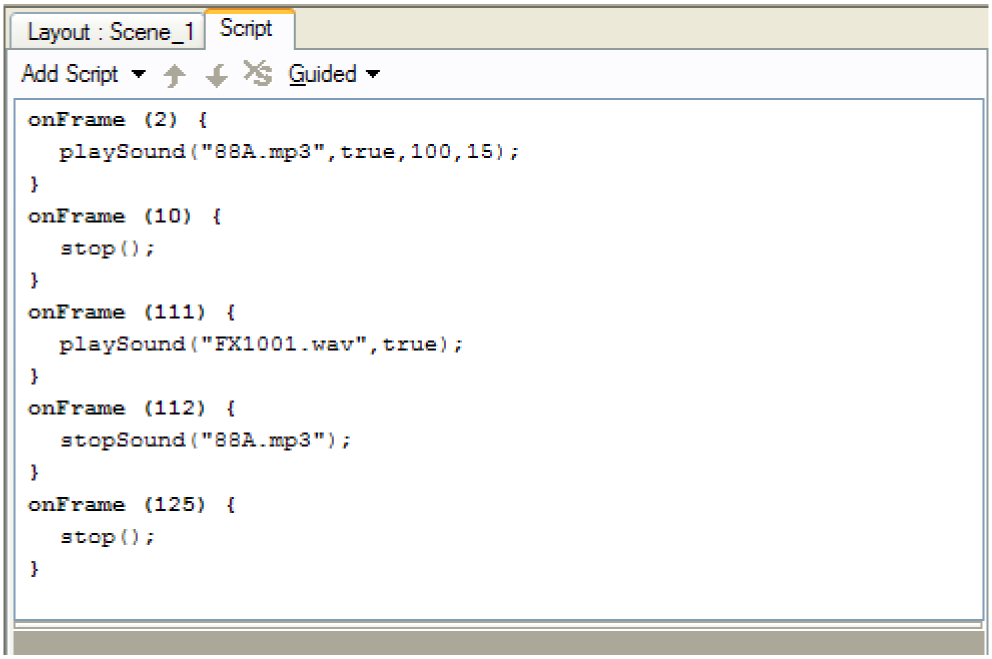


```
on (release) {
   play();
}
```

11. Press ▶ Play on the Control Toolbar. The THIS Text Object will fade in and grow onto the screen, then the Movie will stop because of the Stop Action at Frame 10. Now move your mouse over the THIS Text Object and click and release the mouse button. When the mouse is released the Movie continues playing because of the Play Action that is triggered On Release. Press ■ Stop on the Toolbar

12. Now go back to the Timeline Panel and select Frame 125 in the row labeled Scene_1

13. Press the 'Add Script' button and select Movie Control | Stop from the menu.
    The 'Script' panel should now look something like this:



```
onFrame (2) {
   playSound("88A.mp3",true,100,15);
}
onFrame (10) {
   stop();
}
onFrame (111) {
   playSound("FX1001.wav",true);
}
onFrame (112) {
   stopSound("88A.mp3");
}
onFrame (125) {
   stop();
}
```

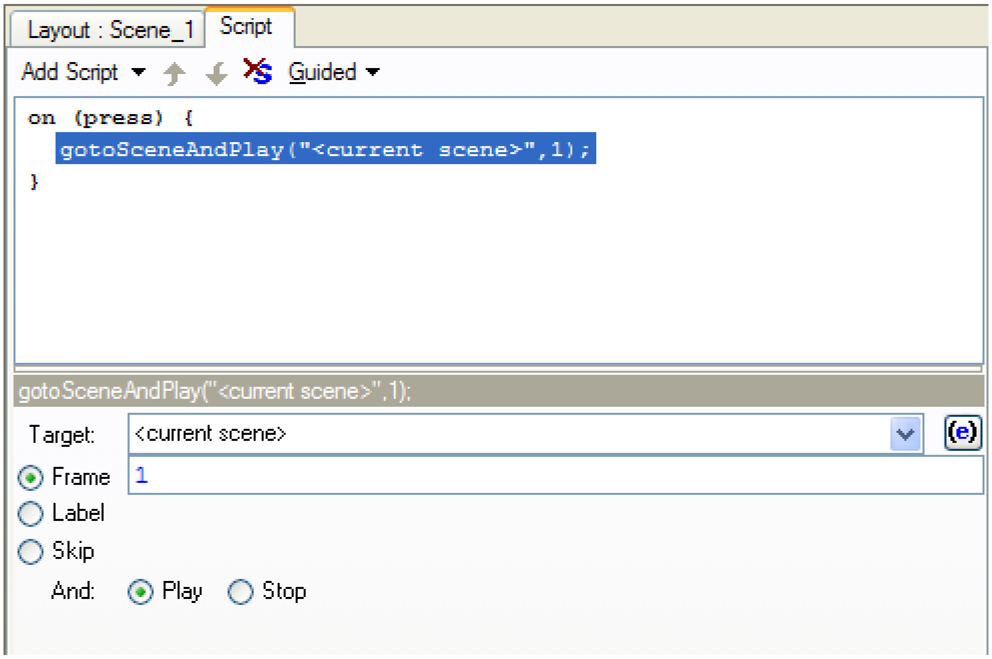
14. On the 'Timeline' Panel, select the "Go to Swishzone" Object and click on the Script Panel

15. Press the 'Add Script' button and select Events | Button | OnPress from the menu

16. Press the 'Add Script' button again and select Movie Control | gotoAndPlay | gotoAndPlay(FRAME) from the menu. You should now see the Goto Frame Properties on the Script Panel

17. The current Scene and Frame Number 1 are selected by default in the Goto Frame properties.

The 'Script Panel' should now look like this:



```
on (press) {
    gotoSceneAndPlay("<current scene>",1);
}
```

gotoSceneAndPlay("<current scene>",1);

Target: <current scene>

● Frame 1

○ Label

○ Skip

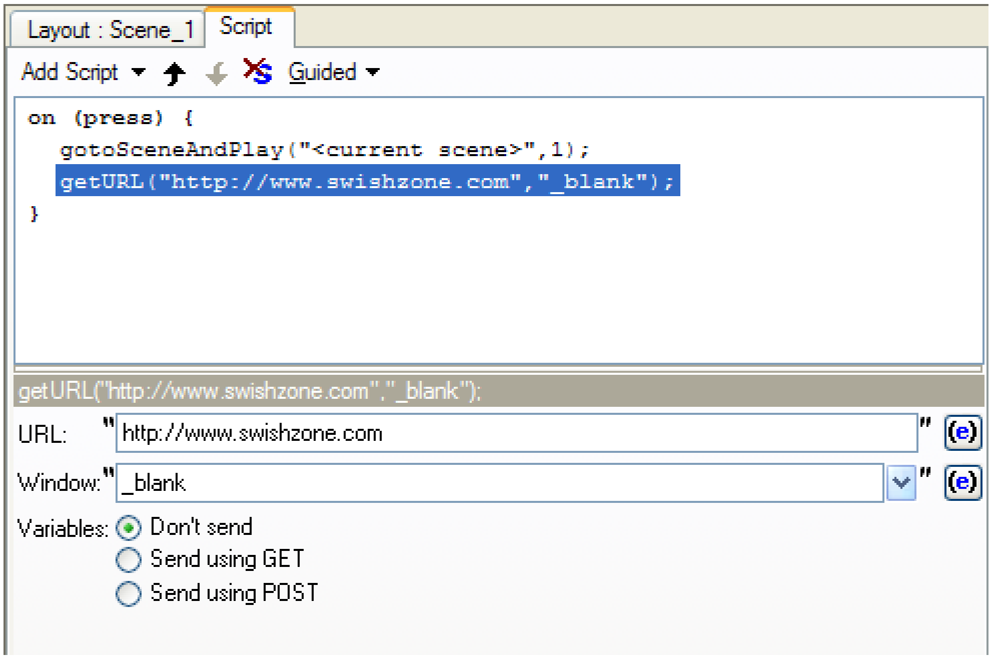
And: ● Play ○ Stop

Note: The 'gotoAndPlay' dialog box will default to Frame 1 of the current Scene. If you wish to direct the Play Head to a different Frame or Scene, enter the Frame number and use the Target drop-down menu to select a different Scene.

18. Press the 'Add Script' button again and select Browser/Network | getURL from the menu. You should now see the getURL properties on the right side of the 'Script' Panel

19. Enter "http://www.swishzone.com" into the URL text box and select "_blank" from the Target drop-down menu (using "_blank" will open a new browser window).

The 'Script Panel' should now look like this:



Layout : Scene_1 | Script

Add Script ▾ | Guided ▾

```
on (press) {
   gotoSceneAndPlay("<current scene>",1);
   getURL("http://www.swishzone.com","_blank");
}
```

getURL("http://www.swishzone.com","_blank");

URL: "http://www.swishzone.com" (e)

Window: "_blank" (e)

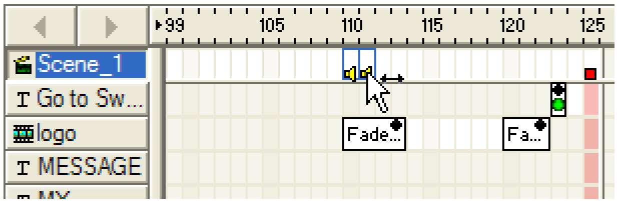
Variables: ● Don't send
○ Send using GET
○ Send using POST

20. You will be able to test most of your Actions by pressing ▶ Play on the Control Toolbar. The "THIS" Text Object will fade in and grow onto the screen, then the Movie will stop. Click and release the mouse button over this Object. The Movie will continue playing and run the rest of your animation until "Go to Swishzone" appears and then it will stop. Now move the mouse over the Go to Swishzone Object and press the mouse button. The Movie will restart at Frame 0

21. To test the getURL Action, select File | Test | In Browser. Your default browser will open and your Movie will start and then stop when it reaches the first Stop Action you applied at Frame 10, click on the THIS Object to carry on playing your Movie. When it reaches "Go to Swishzone" it will stop (at the Stop Action you applied at Frame 122). Click on the Go to Swishzone Object. Your Movie will go back to the start and, at the same time, a new browser window is launched to display the URL http://www.swishzone.com

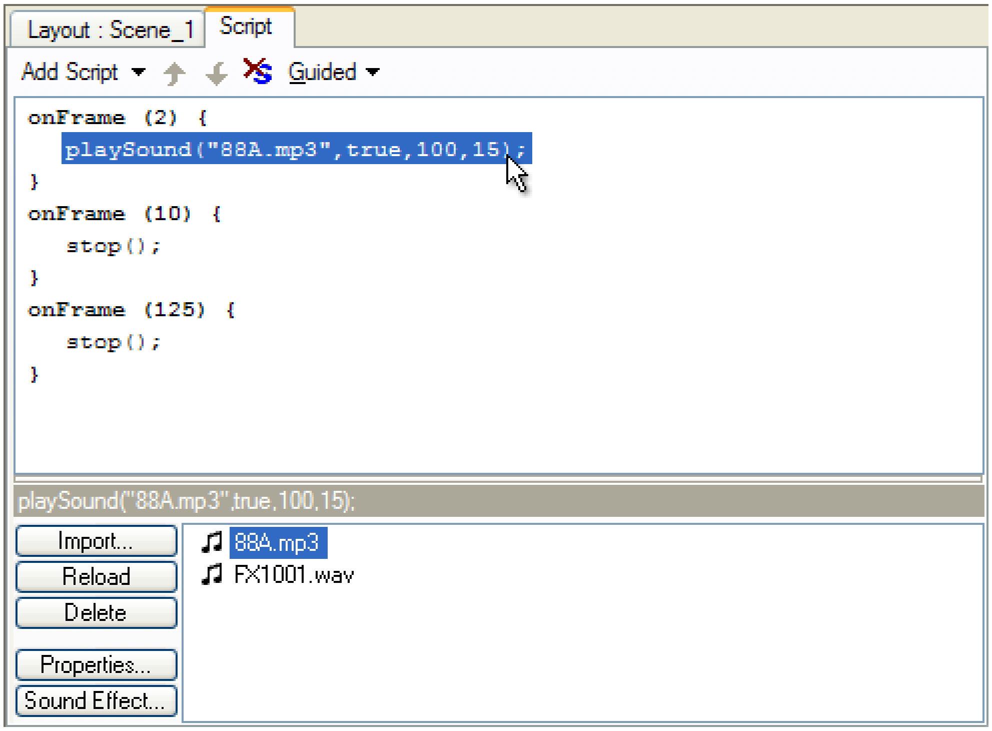
Note: You can use the Tools Menu | Preferences | Player and select the option at the bottom to launch a browser when testing the getURL action from within SWiSHmax. Otherwise, you can use Test in Browser/Player.

22. You will have noticed that there are other Actions already applied in this Scene. On the Scene line on the 'Timeline' Panel, select the Script marker in Frame 110, hold down the CTRL key while you also select the marker in Frame 111 and then press the Delete key on your keyboard



23. On the Timeline in the Scene line, double-click on the Script marker on Frame 2 (the playSound action). This will open the 'Script' Panel where you can then select Play Sound to open the 'Import' dialog box

Your 'Script' Panel should look like this:

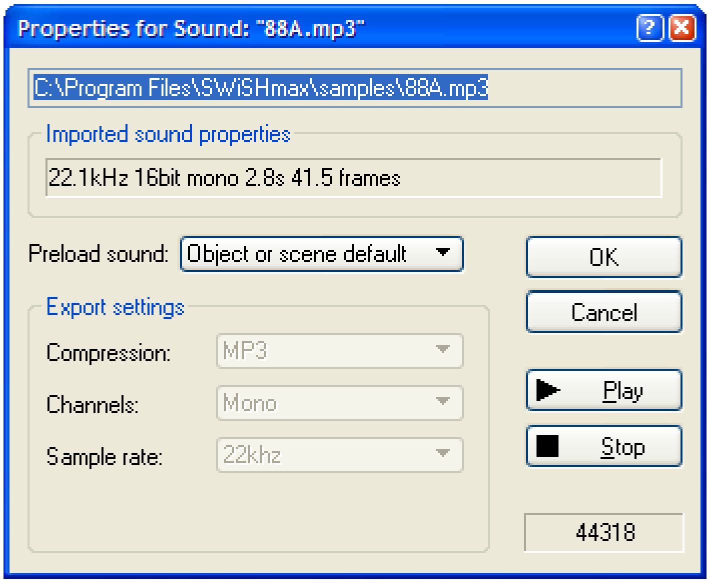


```
onFrame (2) {
   playSound("88A.mp3",true,100,15);
}
onFrame (10) {
   stop();
}
onFrame (125) {
   stop();
}
```

playSound("88A.mp3",true,100,15);

- 88A.mp3
- FX1001.wav

24. Import a wave (.wav) file from your system and you will find that it is automatically added to the Play Sound Action list at the bottom of the Script panel. By double-clicking on the sound in the Content list or by pressing the 'Properties' button you will open the 'Properties' dialog box, where you can change the type and level of compression and the Preload options.



Properties for Sound: "88A.mp3"

C:\Program Files\SWiSHmax\samples\88A.mp3

Imported sound properties

22.1kHz 16bit mono 2.8s 41.5 frames

Preload sound: Object or scene default

Export settings

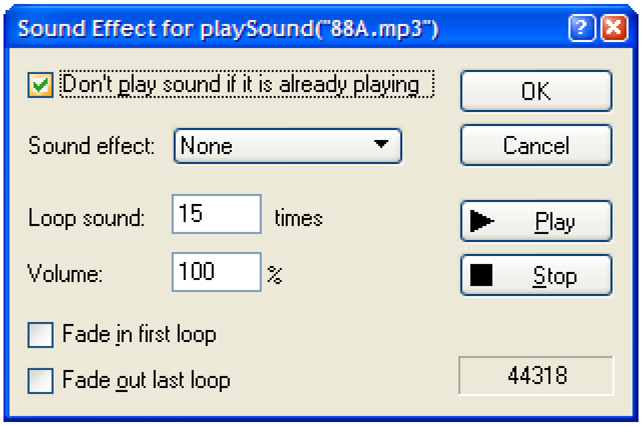
Compression: MP3

Channels: Mono

Sample rate: 22khz

44318

By pressing the 'Sound Effect' button in the Content list at the bottom of the Script panel, you can change the number of times the audio file will be played, add effects such as Pan Left/Right, and set the volume level of the audio file.



Sound Effect for playSound("88A.mp3")

☑ Don't play sound if it is already playing

Sound effect: None

Loop sound: 15 times

Volume: 100 %

□ Fade in first loop

□ Fade out last loop

44318

Note: Selecting either "Fade In" or "Fade Out" from the Sound Effect drop-down menu will fade the sound across the total number of loops set. The 'Fade in first loop" and "Fade out last loop" options will fade the sound across only one loop of the audio (the rest of the loops will play at full volume).

25. Select File | Test in Player to test your Movie

26. Save your Movie

How to Make an Animated Button

Top Previous Next

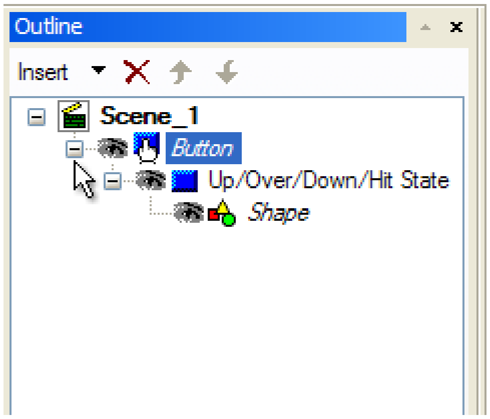
This is a step-by-step tutorial on how to create a simple animated button using a Sprite. The .swi file for this tutorial is "buttondemo.swi" in the File | Samples | Tutorials Menu.

Buttons can be created using the 'Insert Button' icon on the 'Insert' Toolbar or by converting an existing object into a button by selecting Modify | Convert to Button on the main Menu. This second option is the method illustrated in this tutorial.

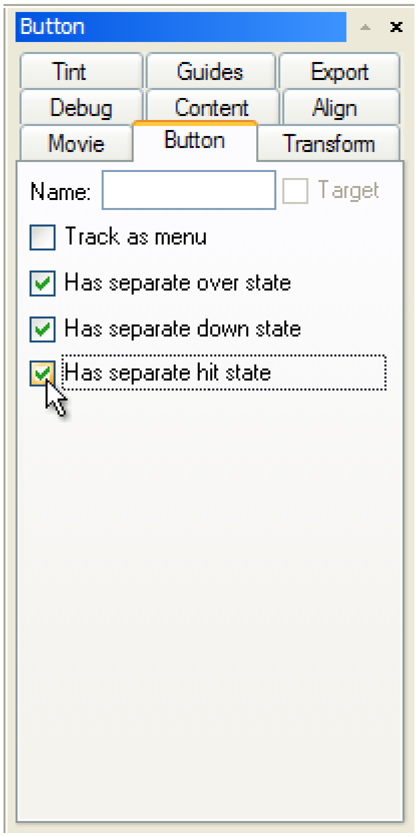
1. Start a New empty movie (File Menu | New). Create a red circle using the 'Ellipse' vector tool from the Toolbar on the 'Layout' Panel (hold down the Shift key on your keyboard while you drag out the shape to make a perfect circle)

2. On the Main Menu bar, select Modify | Convert to Button

3. After pressing the (+) icon next to the Button object, the 'Outline' Panel should look like this:



4. Open the 'Button' Panel and check the options for 'Has separate ... state' (over/down/hit)



Button

Name:

□ Target

□ Track as menu

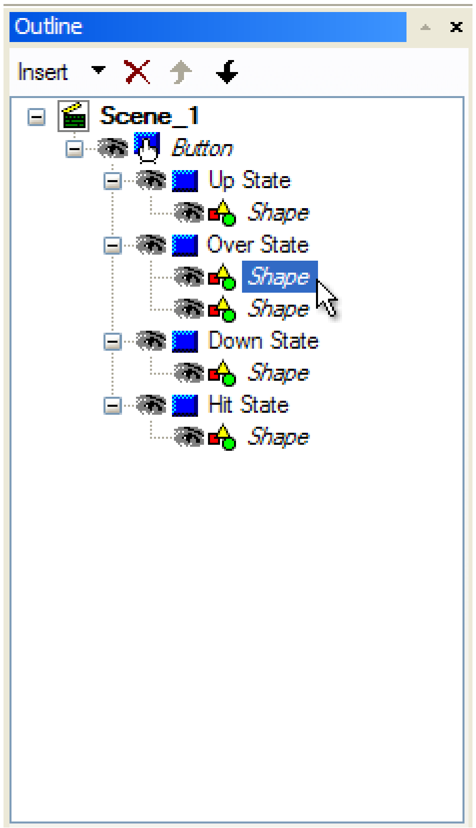
☑ Has separate over state

☑ Has separate down state

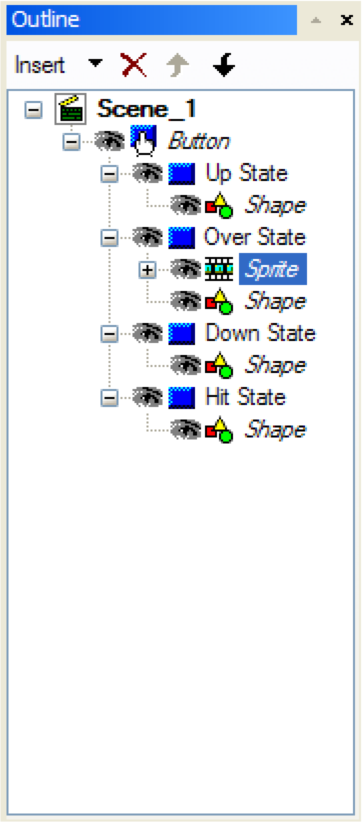
☑ Has separate hit state

5. Select the ellipse in the Over State and press Control+C to copy, and Control+V to paste a duplicate into this Over State

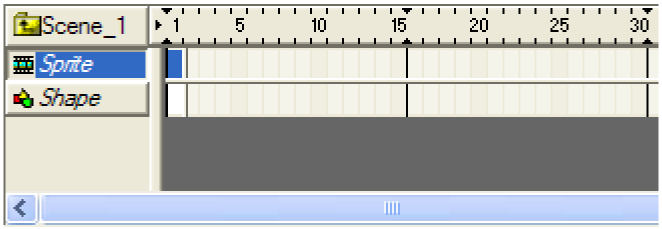
Your 'Outline' Panel should now look like this:



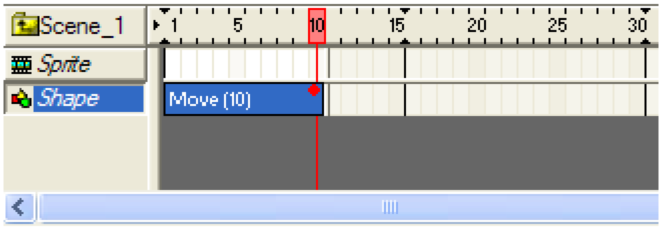
6. Open the 'Shape' Panel and select "None" for the Fill type. Give this shape a Line Width of "1" with a Line Color of Black, and then right-click on the Ellipse in the 'Outline' Panel and select Convert | Convert to Sprite. Your 'Outline' Panel should now look like this:



7. Double-clicking on the Sprite in the 'Outline' Panel will open the 'Timeline' Panel showing the Timeline for the Sprite Object. Your Timeline should look like this:



8. On the Timeline, double-click on Frame 10 in the 'Ellipse' object row. This will insert a Move Effect. Your Timeline should look like this:



9. Double-click on the Keyframe at the end of the Move Effect. This will open the 'Move Settings' dialog box. In this dialog box, select Scale | Resize to Scale | 300% and then Alpha | To Transparent. Press the 'Play Scene' button on your 'Control' Toolbar then move your mouse over the Ellipse to preview your Sprite

10. On the 'Timeline' Panel, select the Scene 1 row and press 'Play' to see how the mouseover works to this point. Move the mouse over your red circle. You should see a black circle radiating out from the red circle and fading out.

Note: If you want the black circle to continually radiate from the red circle, open the Sprite panel and uncheck the option for 'Stop Playing at End of Sprite'

11. Open your 'Outline' Panel to finish editing your button. Select the ellipse in the Over State, then open the 'Shape' Panel and change the color to a bright green. Then go back to the 'Outline' Panel and select the ellipse in the Down State, changing the color to blue in the 'Shape' Panel

12. You are finished. Press the 'Play' button on the 'Control' Toolbar to test your button

13. Save your Movie as "mybutton.swi"

Creating an Auto Run CD

Top Previous Next

Using the File | Export | EXE (projector)... option, it is possible to save your Movie as a .exe file. When the .exe file is run, your Movie will be displayed without any need to for internet connection or browser programs. This format is ideal for CD distribution.

To make your CD start automatically, it is necessary to create an autorun.inf file. This file can be made in a text editor such as notepad.

The following discussion assumes your exported Movie is called "mymovie.exe". You can name your Movie anything, but you should edit the autorun.inf file appropriately.

Open notepad and type the following:

```
[AutoRun]
open=mymovie.exe
icon=mymovie.exe,0
```

Save the file as autorun.inf - If using notepad, you will have to select All files for the Save as type option.

If you have a CD burner and appropriate software installed on your system, you can then create the autostart CD by saving the autorun.inf and mymovie.exe to the root directory of the CD.

You must create the CD using Joliet or ISO9660 file systems. Do not use the Universal Disk Format file system. Universal Disk Format is used by Roxio Direct CD.

Note: Although CDs created using the ISO9660 file system can be read on Apple and other computers, the compiled projector file will not run as it is designed for the Windows operating system and Intel processors. You could save exported .htm and .swf files to the CD, which would allow your Movie to be viewed using a browser with the Flash Plugin installed. This would allow your Movie to be viewed on non-Windows systems

When the completed CD is inserted into the CD drive, the Movie should start playing automatically.

Scripting Tutorials

Top Previous Next

The following tutorials can be loaded from the Samples area (File | Samples | Tutorials | filename.swi).

.swi File	Tutorial Name	Purpose
helloworld.swi	Hello World	Demonstrate the use of the trace command and Debug Panel
flowcontrol.swi	Variables and Flow Control	Demonstrate the use of variables, if and while statements
button.swi	Button	Demonstrates the on(release) Button Event
button1.swi	Button with rollOver	Demonstrate the on(rollOver), on(rollOut) Button Events. Demonstrate the _xscale and _yscale properties
calculator.swi	Calculator	Introduces the Text Object with Input and Dynamic text properties. Introduces the Math Object
properties.swi	Object Properties	Introduce the Object properties: _X, _Y, _xscale, _yscale, _width, _height, _alpha, _rotation, _name and _visible
dragging.swi	Mouse Dragging	Introduces startDragUnlocked(), startDragLocked() and stopDrag() methods. Introduces onSelfEvent(press) and onSelfEvent(release) Events to start and stop dragging
droptarget.swi	Drop Target / Target	Demonstrates the use of the _target and _droptarget properties
collide.swi	Collision Detection	Introduces isNearTarget() and isNearThis() methods

game.swi	Simple Gaming	Introduces User defined functions, use of onLoad() Event Handling function to initialise variables and the use of the onEnterFrame() Event Handling function to animate Objects
mymoviecontrol.swi	Movie Control	Demonstrates how to control timelines playing in Sprites and how Sprites can control the main movie timeline using various methods of the gotoAndPlay() action.
welcome.swi	Input and Dynamic Text	Demonstrates how users can add text to the movie to be used at a later time in the same movie. Introduces methods for user input through Input Text fields and storing that information as a variable to be used in dynamic content
maxarray.swi	Arrays and Random	Demonstrates how to use Arrays to display a random message. Introduces methods for Arrays and Math.randomInt()
txtmessage.swi	Data Transfer - text	Demonstrates how data can be transferred to and from .htm, .asp, .php or text files that are resident on a server. Introduces the methods loadVariables() and shows how you can load an external .txt file into your movie at runtime.

It is recommended that Guided mode be selected in the Script Panel for the entry of the following tutorials. The tutorials assume that you are familiar with the SWiSH drawing tools. If you are not familiar, please refer to these tutorials first.

Hello World

Top Previous Next

Description

This is a step-by-step tutorial for creating a Movie displaying the words "Hello World" in the Debug Panel.

Aim

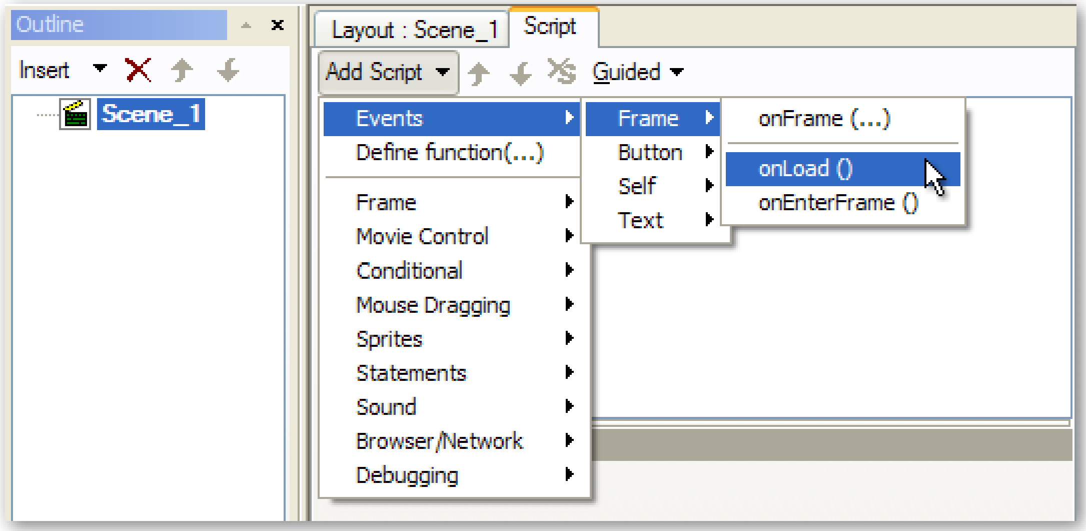
This Movie introduces the method of entering scripts, explains use of the onLoad() Event and demonstrates the use of the trace command and Debug Panel.

.swi file

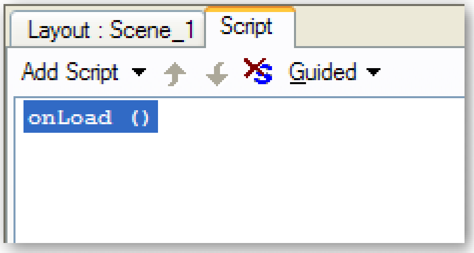
"helloworld.swi"

1. Start with a new project (File | New) and save the file as "myhelloworld.swi"

2. Select the 'Script' Panel, then enter an onLoad() Event for Scene 1, as shown below.

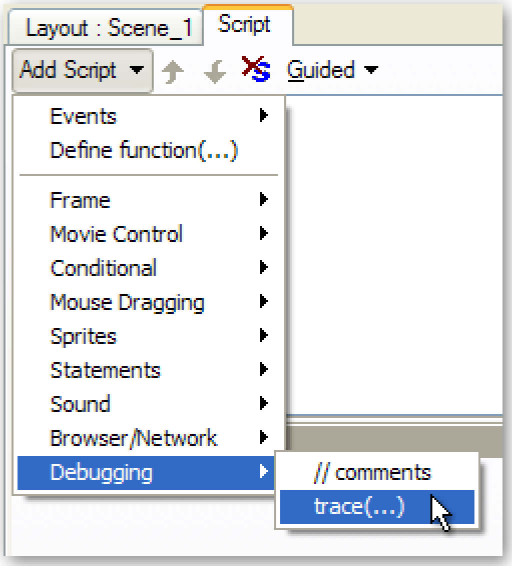


The 'Script' Panel should now look like this:



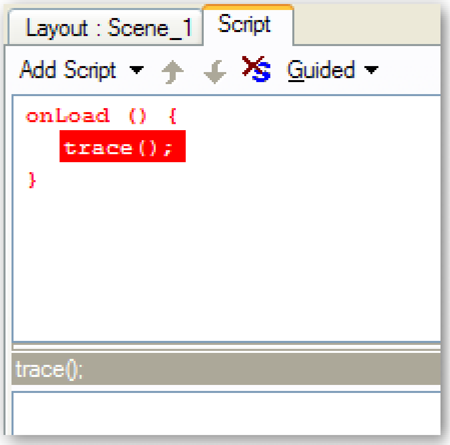
```
onLoad ()
```

3. Add a trace command to the onLoad() Event by selecting the onLoad() function, as shown above. Press the 'Add Script' button then select
Debugging | trace(...) from the menu as shown below:



Note: It is possible to append a Scripting command by right-clicking on the command you want the new command to appear after, then selecting the Add Script Menu item. Sections of script can also be duplicated and moved using the Cut and Paste Menu options

The 'Script' Panel should now look like this.



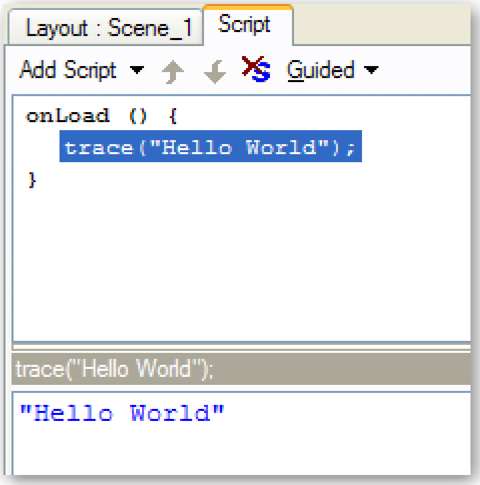
```
onLoad () {
    trace();
}
```

trace();

Note: The function and trace statement are shown in red. This is because the trace statement is currently incomplete

4. Complete the trace statement by adding the string "Hello World" (including the quotation marks). Select the trace statement in the top half of the 'Script' Panel. The bottom half of the Panel should have the heading trace(); as shown in the image above. Click the bottom section of the Panel (the section with the "trace();" heading) and enter the text "Hello World". Once the second quotation mark has been entered, the text should turn blue. Click anywhere outside of the Panel you have just entered the text into. This will cause the script to be re-validated. The entire script function should turn black indicating that it contains syntactically correct script.

   The Panels should now look like this:

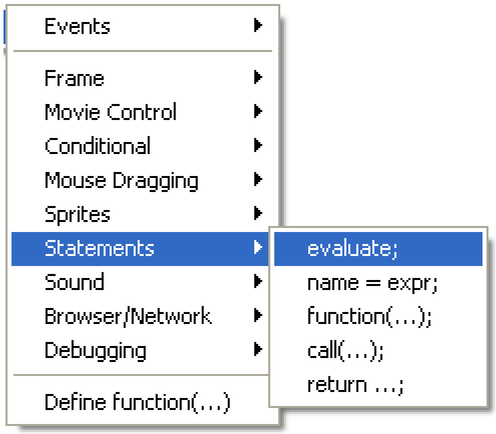


```
onLoad () {
    trace("Hello World");
}
```

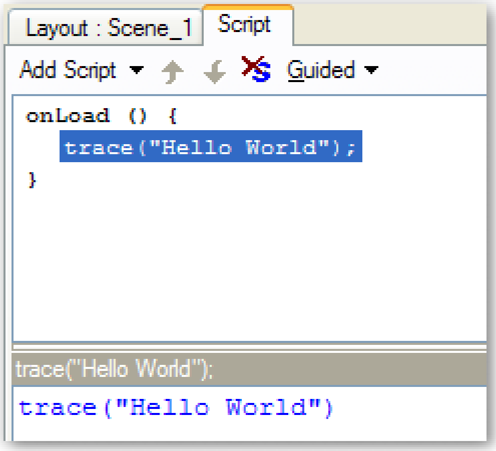
trace("Hello World");

```
"Hello World"
```

Note: If you are confident in the use of command syntax, you can enter commands directly using the Statements | evaluate; Menu option, as shown below



This will result in the following window:



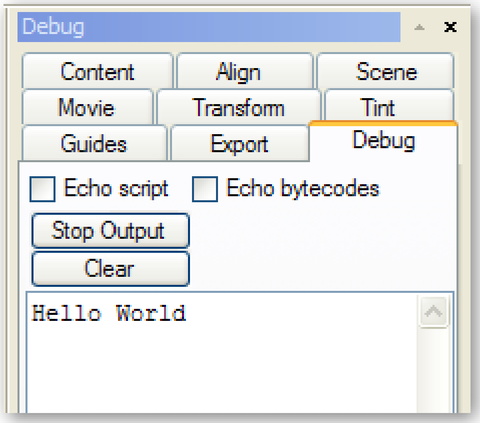
```
onLoad () {
    trace("Hello World");
}
```

trace("Hello World");

```
trace("Hello World")
```

The semi colon ';' will be added for you

5. Select the Debug tab amongst the panels displayed on the right hand side. Check your script by pressing the 'Play' button to run your Movie. The 'Debug' Panel should show the following:



```
Hello World
```

6. Save your Movie

Analysis
When the 'Play' button is pressed, the Movie loads. This causes the script in the onLoad() Event Handling Function to be executed.
The trace statement containing the string "Hello World" is then executed, causing the string to be displayed in the Debug Panel.

Variables and Flow Control

Top Previous Next

Description

This step-by-step tutorial develops the previous tutorial by introducing variable calculations and flow control. After the Movie displays the words "Hello World" in the Debug Panel, it will calculate the results of y = x * x in the Debug Panel, where x has the values of 0 to 10.
When the result y is greater than 50, it will add the additional comment "wow that is big".

Aim

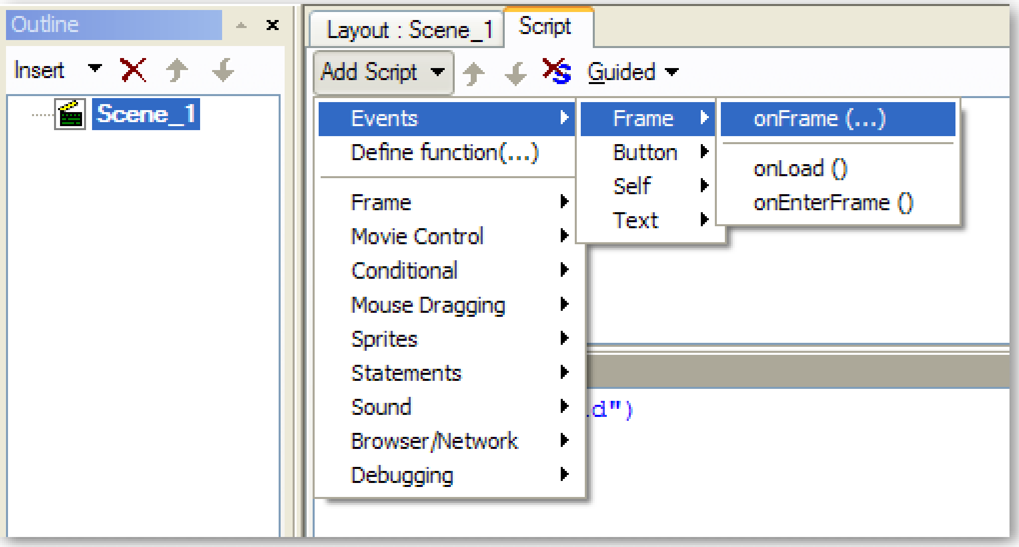
This Movie introduces the onFrame() Event, demonstrates the use of the trace command and Debug Panel and demonstrates the use of variables, as well as the if and while conditional statements.

.swi file

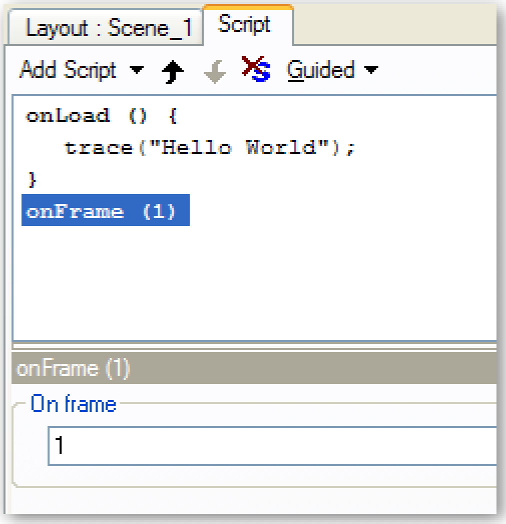
"flowcontrol.swi"

1. Load the .swi file "helloworld.swi". Ensure that the 'Play' button is not pressed and save the file as "myflowcontrol.swi".

2. Enter an onFrame Event to Scene_1. This Event will occur when the Movie reaches the specified Frame.



As Frame 1 is the currently selected Frame in the Timeline, the onFrame Event will be for Frame 1 of Scene_1. After inserting the Event, the 'Script' Panel should look like this:



```
onLoad () {
   trace("Hello World");
}
onFrame (1)
```

onFrame (1)

On frame: 1

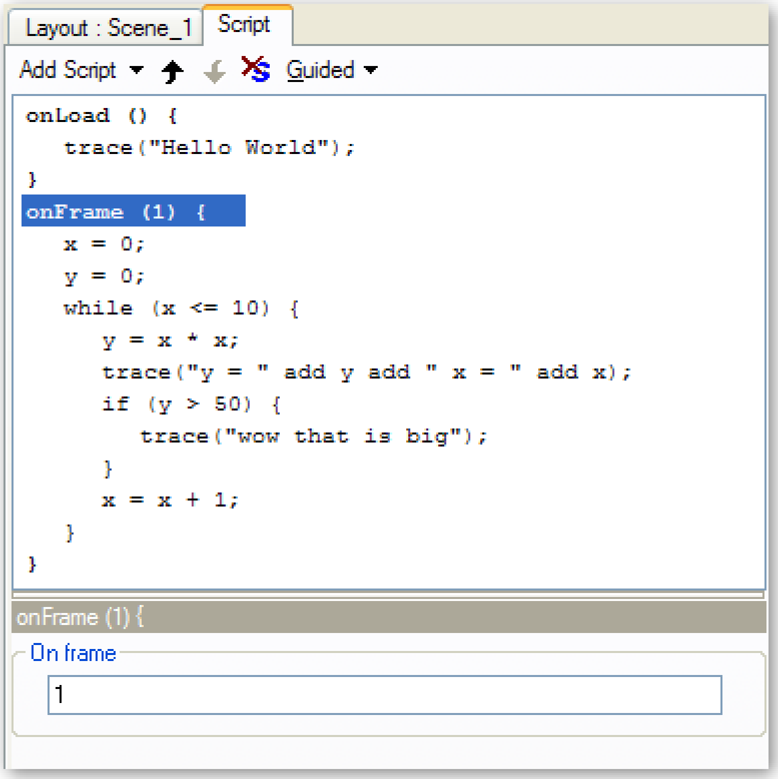
Note:

- The Timeline now indicates via an 's' that a Scripting Event will occur in Frame 1 of Scene 1. If needed, this onFrame Event can be moved to a different Frame by altering the number in the bottom half of the 'Scripting' Panel. The Event can also be moved to a different Frame by dragging the 's' in the Timeline
- The onLoad() event was entered in the previous tutorial (Hello World).

2. Add the following statements to the onFrame (1) Event. Note that script can also be added by selecting the area where you want the script to be placed and then pressing the right mouse button to access the Add Script option. The Menu commands used to enter the script are shown under the **Comments** heading

Statement	**Comments.**
x = 0;	Enter using Add Script | Statements | name = expr; Type "x" into the Name field (this field defines a property or variable). Then add the value "0" (zero - with no quotes) into the value field below the Operator. In this case the Target: option can be left blank. This option is used to define the name of the Object that contains the property or variable. Leaving this value blank is the same as entering the value this
y = 0;	Enter using Add Script | Statements | name = expr; Another way to enter this statement is to copy and paste the x=0 statement, then change to parameters to make this statement y = 0;
while (x <= 10) {	Enter using Add Script | Conditional | while (...) { Enter x <= 10 into the lower dialog box
y = x * x;	Enter using Add Script | Statements | name = expr; or copy and modify previous statement
trace("y = " add y add " x = " add x);	Enter using Add Script | Debugging | trace(...) Note that the add command is used to join multiple strings
if (y > 50) {	Enter using Add Script | Conditional | if (...) {
trace("wow that is big");	Enter using Add Script | Debugging | trace(...) This command is only executed if the value of y is greater than 50
}	Entered as part of the if statement. Shows the end of the if statement
x = x + 1;	Click on the ending bracket for the IF statement, and enter this script using Add Script | Statements | name = expr; If this statement is missing, x will never increase and loop will continue indefinitely. Note that x++ or x += 1 statements could have also been used
}	Entered as part of the while statement. Shows the end of the while loop

Once you have entered the above script commands, your 'Script' Panel should look like this:



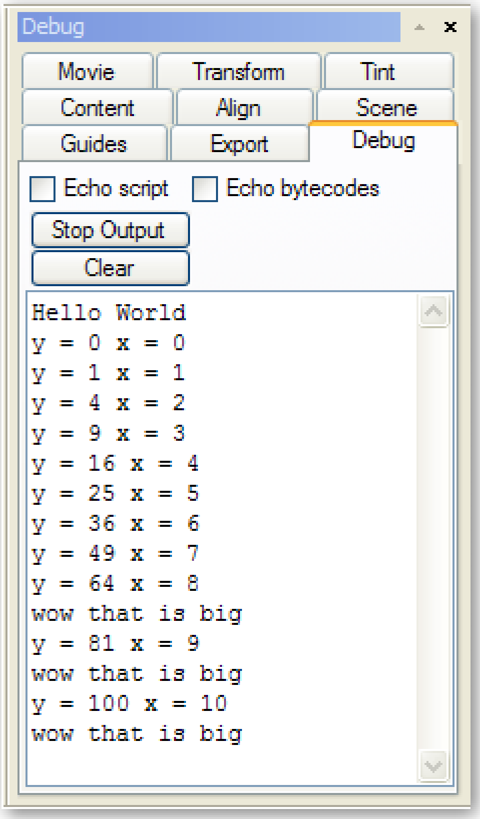
```
onLoad () {
   trace("Hello World");
}
onFrame (1) {
   x = 0;
   y = 0;
   while (x <= 10) {
      y = x * x;
      trace("y = " add y add " x = " add x);
      if (y > 50) {
         trace("wow that is big");
      }
      x = x + 1;
   }
}
```

onFrame (1) {

On frame

1

3. To check your script, select the Debug tab among the Panels displayed on the right hand side and press the 'Play' button to run your Movie. The 'Debug' Panel should show the following:



```
Hello World
y = 0 x = 0
y = 1 x = 1
y = 4 x = 2
y = 9 x = 3
y = 16 x = 4
y = 25 x = 5
y = 36 x = 6
y = 49 x = 7
y = 64 x = 8
wow that is big
y = 81 x = 9
wow that is big
y = 100 x = 10
wow that is big
```

Analysis

When the 'Play' button is pressed, the Movie loads. This causes the script in the onLoad() Event Handling function to be executed.
The trace statement containing the string "Hello World" is then executed causing the string to be displayed in the Debug Panel.

Once the Movie has been loaded it starts playing. On frame 1 of Scene_1 the onFrame(1) Event Handling function containing the following script is executed:

```
x = 0;
y = 0;
while (x <= 10) {
   y = x * x;
   trace("y = " add y add " x = " add x);
   if (y > 50) {
      trace("wow that is big");
   }
   x = x + 1;
}
```

x and y are both Variables. All variables should be defined a value before use. For this reason, the statement x=0 should be included as the subsequent statement x = x + 1 assumes that x already has a value. As the variable y is only assigned values, the y=0 statement is not strictly necessary, but it is always good practice to set up initial values.

The while loop will continue to execute the contained statements until the value of x exceeds 10. As the last statement of the while loop is x = x + 1, the loop will execute with the values of x = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and terminate when x = 11.

The trace statement trace("y = " add y add " x = " add x); is used to display the values for both y and x. The add operator is used to join multiple strings into a single string. Note that the variables x and y are converted to strings for display in the Debug Panel.

On each iteration of the while loop, the value of y is compared to 50 by the if (y > 50) statement. If the variable y is greater than 50 then the statement contained by the if statement, trace("wow that is big"); is executed. This causes the line "wow that is big" statement to be displayed each time y is bigger than 50.

Button

Top Previous Next

Description
This is a step-by-step tutorial for creating a button that executes script when pressed.

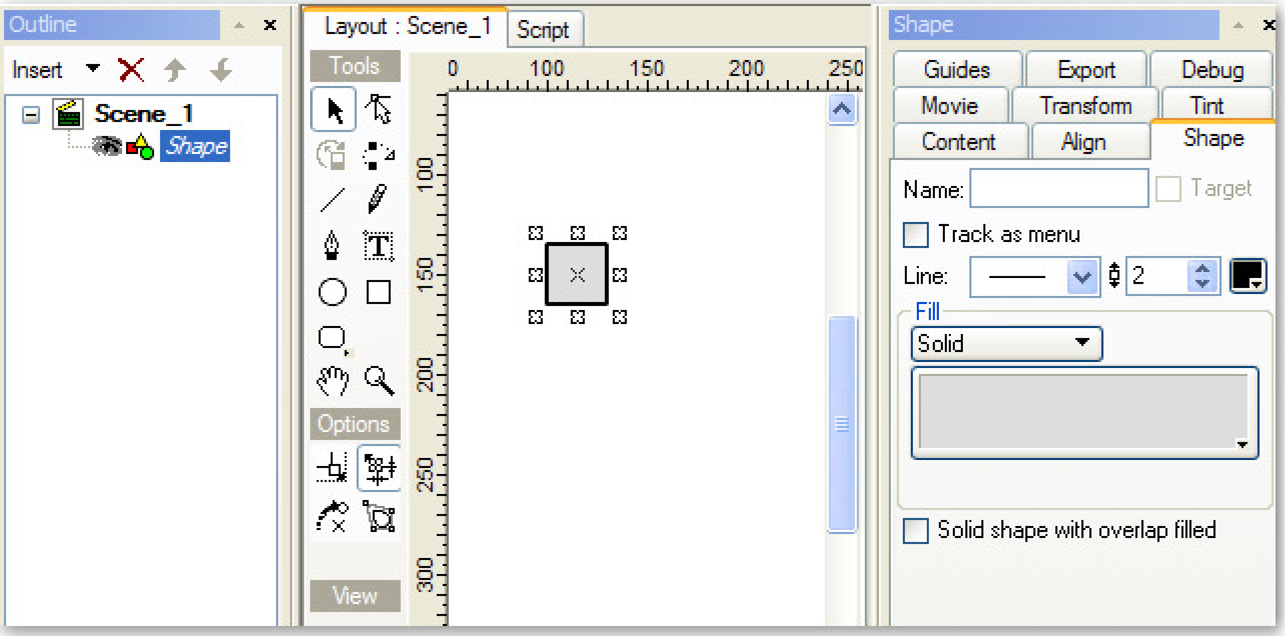
Aim
This tutorial introduces the procedure for making a simple button and demonstrates the on(press) Button Event.

.swi file
"button.swi"

1. Start with a new project (File | New) and save the file as "mybutton.swi"

2. Create a button.

   Select the Layout Panel and use the Rectangle tool to create a small rectangle or square.



Name: □ Target

□ Track as menu

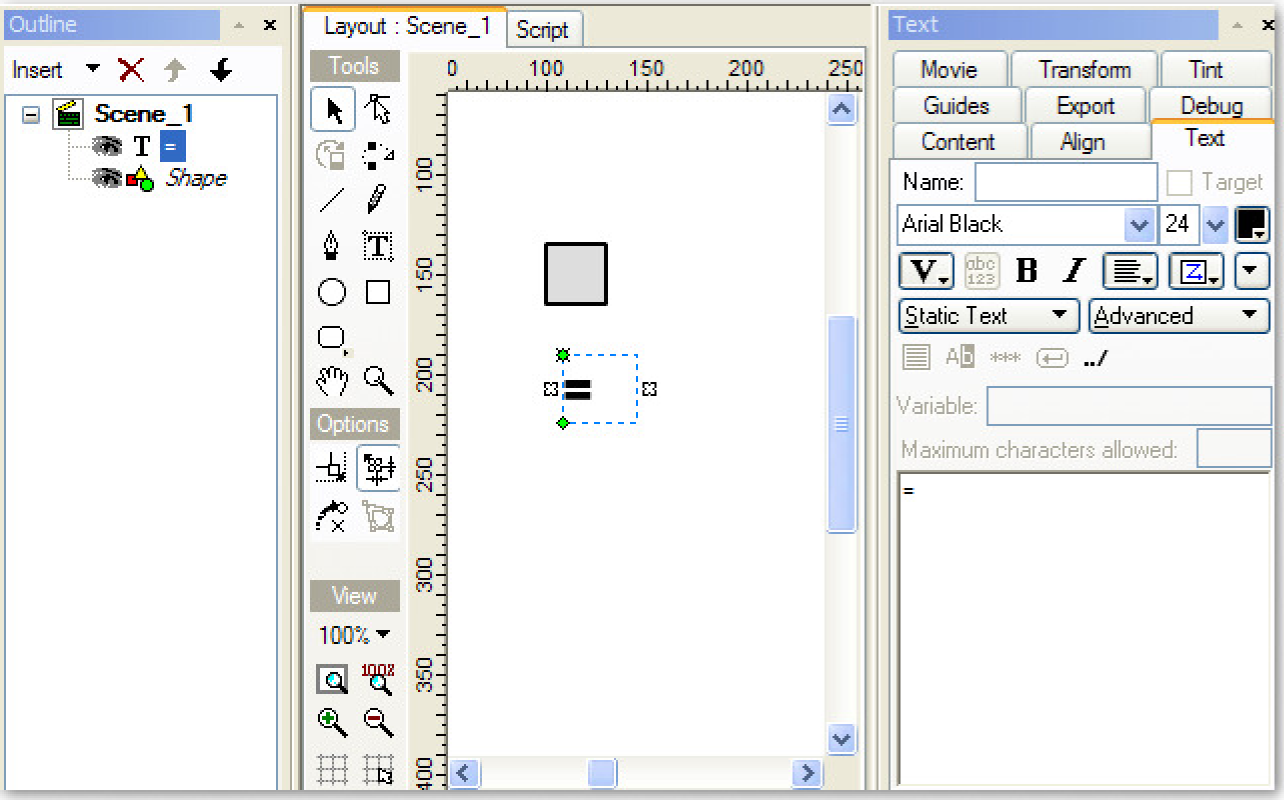
Line: 2

Fill

Solid

□ Solid shape with overlap filled

Use the Text Tool to create the static text "=".



Name: □ Target

Arial Black 24

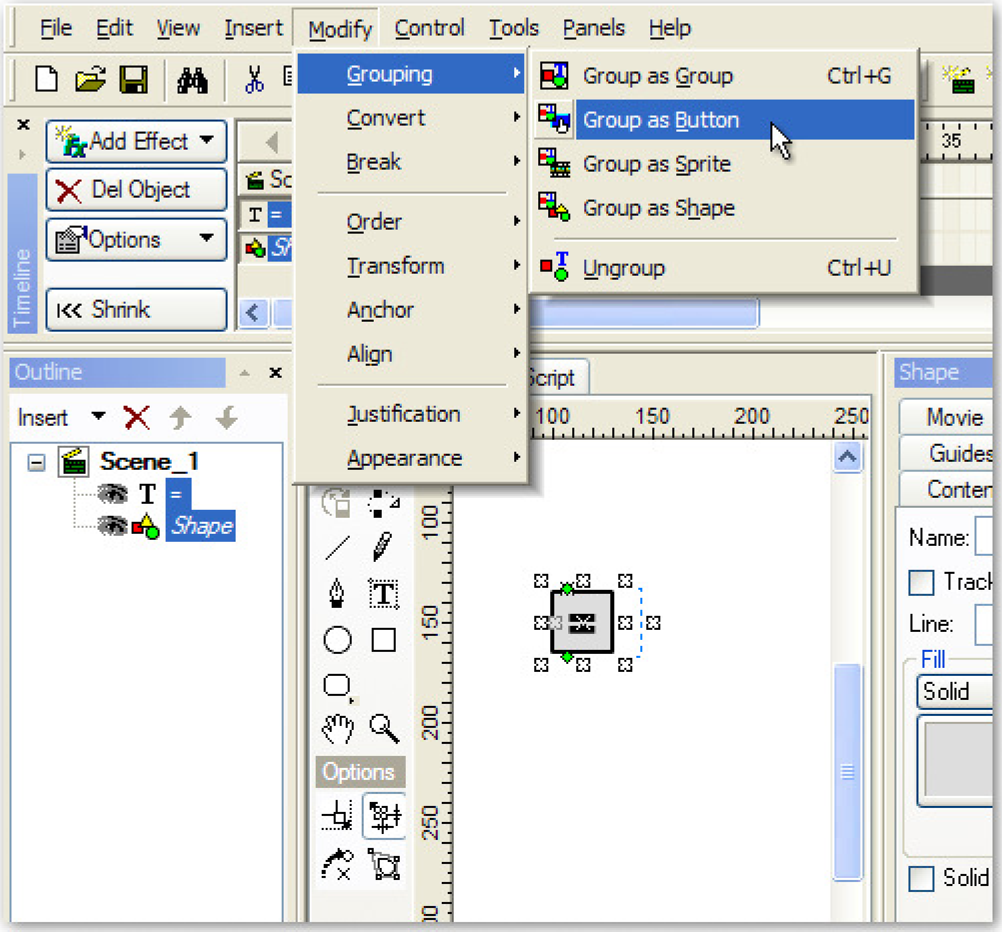
Static Text Advanced

Variable:

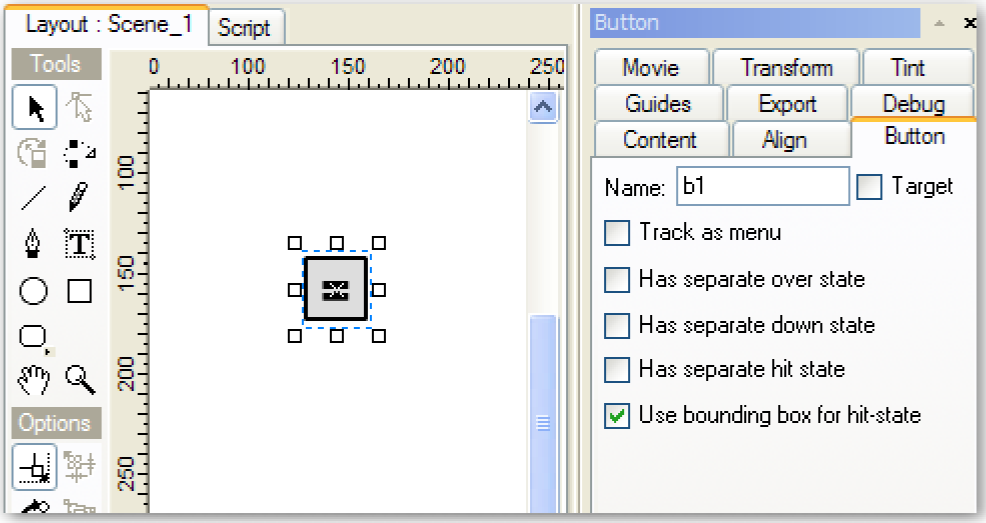
Maximum characters allowed:

=

Move the static text over the rectangle, then select both the text and the shape. Use the Modify menu and select Grouping then Group as Button.



Name the button b1 and select the checkboxes as shown below



Name: b1 □ Target

□ Track as menu

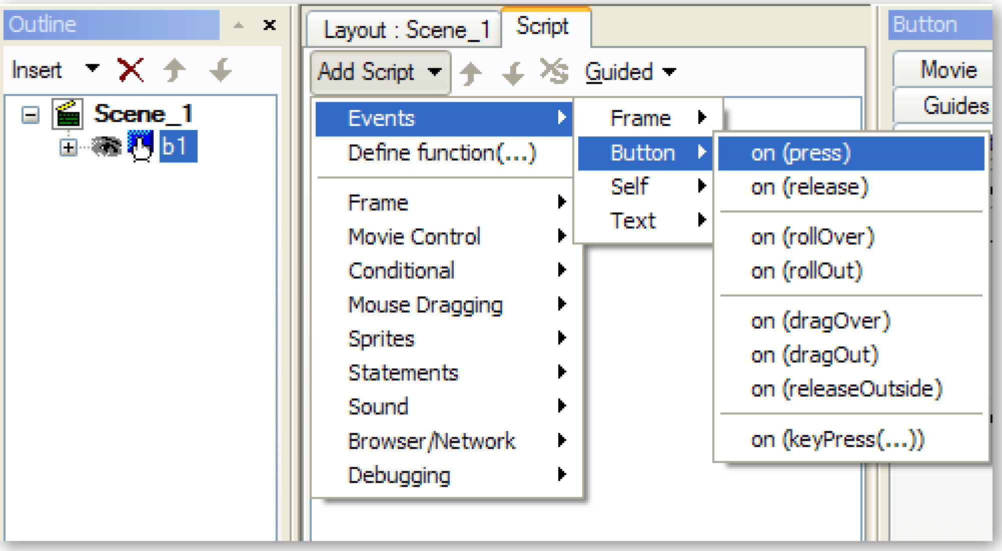
□ Has separate over state

□ Has separate down state

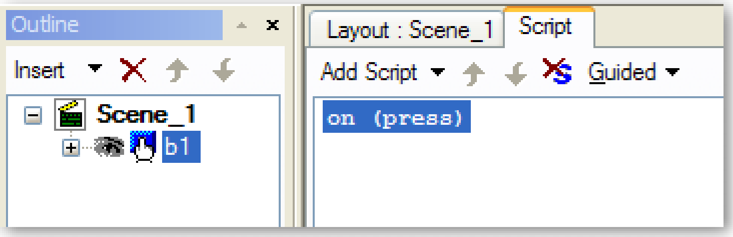
□ Has separate hit state

☑ Use bounding box for hit-state

3. Add some script to indicate when the button is pressed. Ensure the 'b1' button is selected in the 'Outline' Panel. Select the 'Script' Panel and use the 'Add Script' button to add an on(press) Button Event for the button. This is entered via the Menu items Events | Button | on(press).

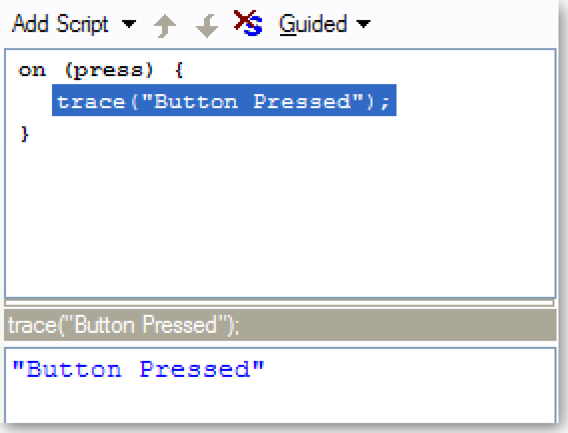


The resulting script should be displayed:



```
on (press)
```

Add the trace statement **trace("Button Released");** using the Menu commands Add Script | Debugging | trace(...). The resulting script should look like:

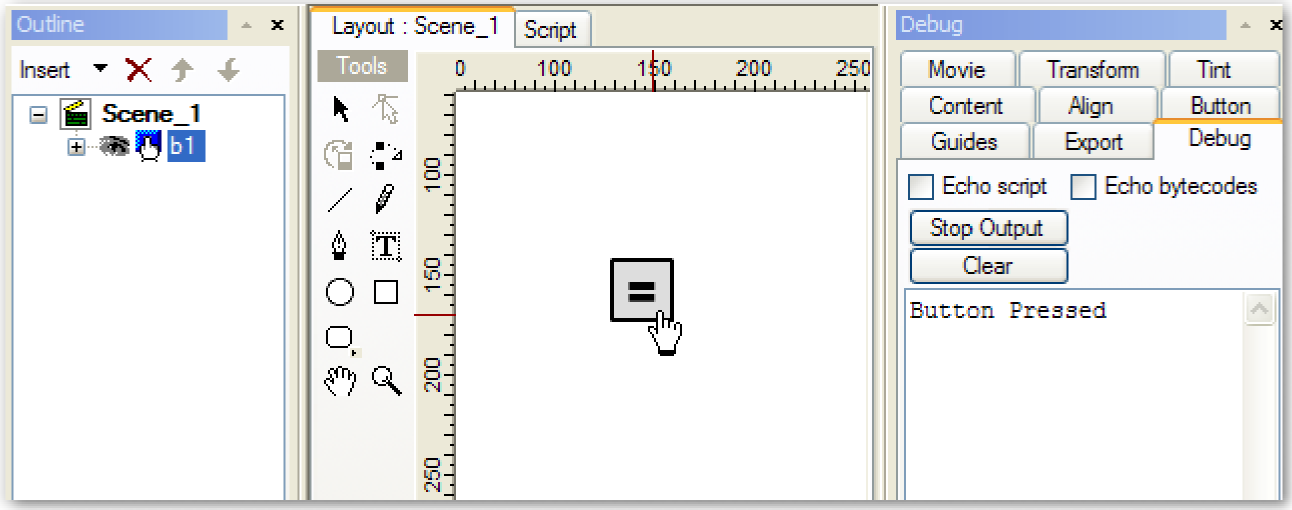


```
on (press) {
    trace("Button Pressed");
}
```

trace("Button Pressed");

```
"Button Pressed"
```

4. Test the button by selecting the Debug tab among the Panels displayed on the right hand side of the 'Layout' Panel. Press the 'Play' button to run your Movie. Each time the left mouse button is pressed while over the button, the statement "Button Pressed" should appear in the 'Debug' Panel



□ Echo script □ Echo bytecodes

Stop Output

Clear

```
Button Pressed
```

Analysis
Two Objects, a shape and a static Text Object, were grouped and made into a button using the Group as button command.

Once the button was created, an on(press) Button Event was added. The statements contained within this Event routine execute each time the left mouse button is pressed while the mouse is positioned over the button.

A trace statement was added to the on(press) Button Event to demonstrate the button Action in the 'Debug' Panel.

Button with rollOver

Top Previous Next

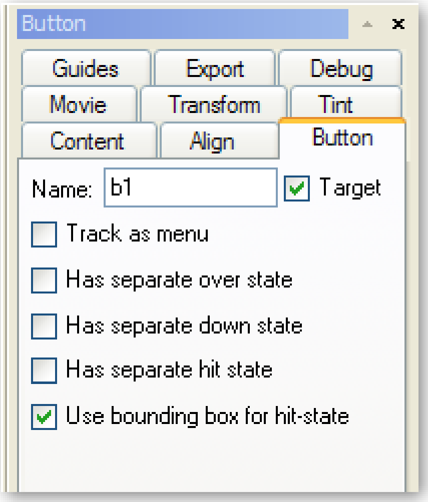
Description
This is a step-by-step tutorial for creating an animated button that changes size when the mouse is positioned over it.

Aim
This tutorial demonstrates the on(rollOver), on(rollOut) Button Events as well as the _xscale and _yscale properties.

.swi file
"button1.swi"

1. Continue from previous button tutorial, or load the "button.swi" file and save the file as "mybutton1.swi".

2. Check the 'Target' checkbox in the Button Panel. This makes the button a Scripting Object, which allows us to access and manipulate the properties associated with it



Name: b1 ☑ Target

□ Track as menu

□ Has separate over state

□ Has separate down state

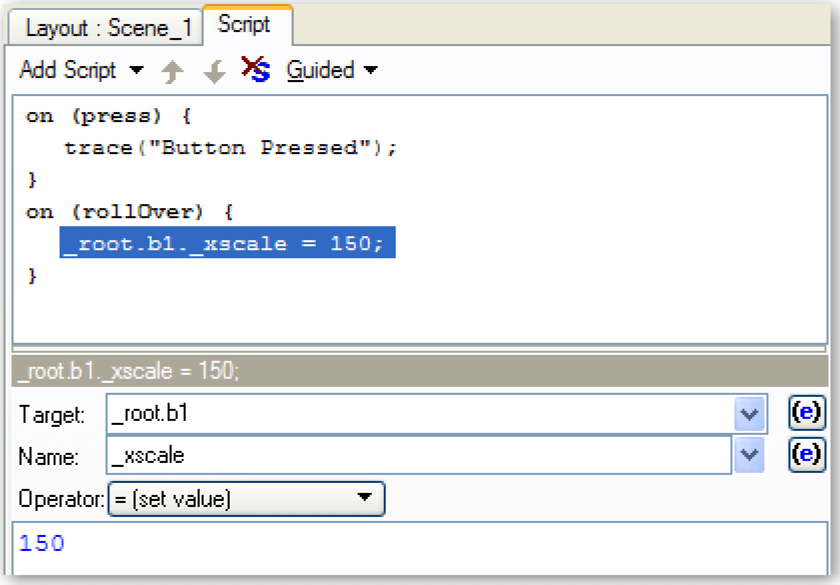
□ Has separate hit state

☑ Use bounding box for hit-state

3. Select the 'b1' button in the 'Outline' Panel, if not already selected. Select the 'Script' Panel and add an on (rollOver) Button Event for the button by selecting the Menu items Add Script | Events | Button | on(rollOver)

4. Use the right mouse button to click on the on (rollOver) function then add the script **_root.b1._xscale = 150;**
   This is added via the menu items Add Script | Statements | name = expr;
   In the 'Target' field, enter **_root.b1**
   In the 'Name' field, select _xscale using the drop-down menu.
   Leave the 'Operator' field as =
   In the bottom section enter the number **150**.

   The resulting script should look like this:



```
on (press) {
   trace("Button Pressed");
}
on (rollOver) {
   _root.b1._xscale = 150;
}
```

_root.b1._xscale = 150;

Target: _root.b1

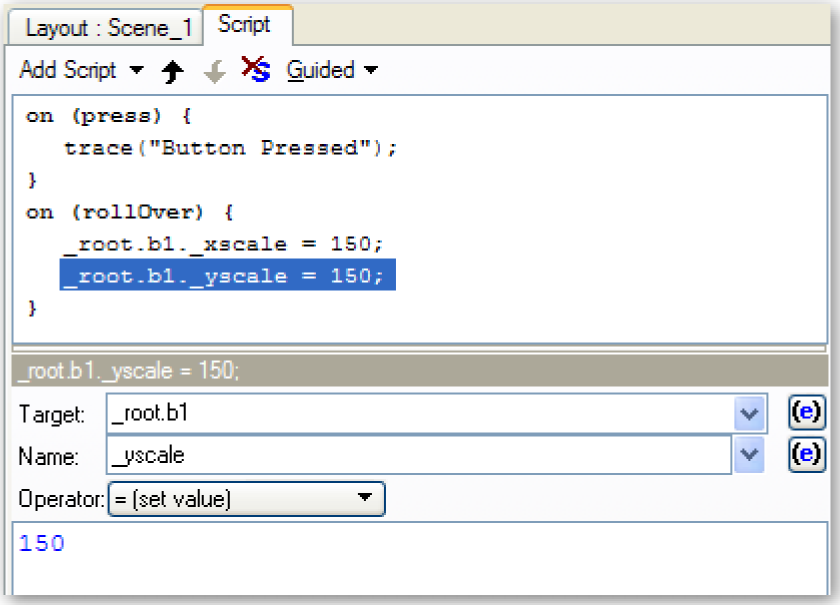
Name: _xscale

Operator: = (set value)

150

Use a similar procedure (or use right click, Copy and Paste then modify parameters) to enter the statement: **_root.b1._yscale = 150;**

The resulting script should look like this:



```
on (press) {
   trace("Button Pressed");
}
on (rollOver) {
   _root.b1._xscale = 150;
   _root.b1._yscale = 150;
}
```

_root.b1._yscale = 150;

Target: _root.b1

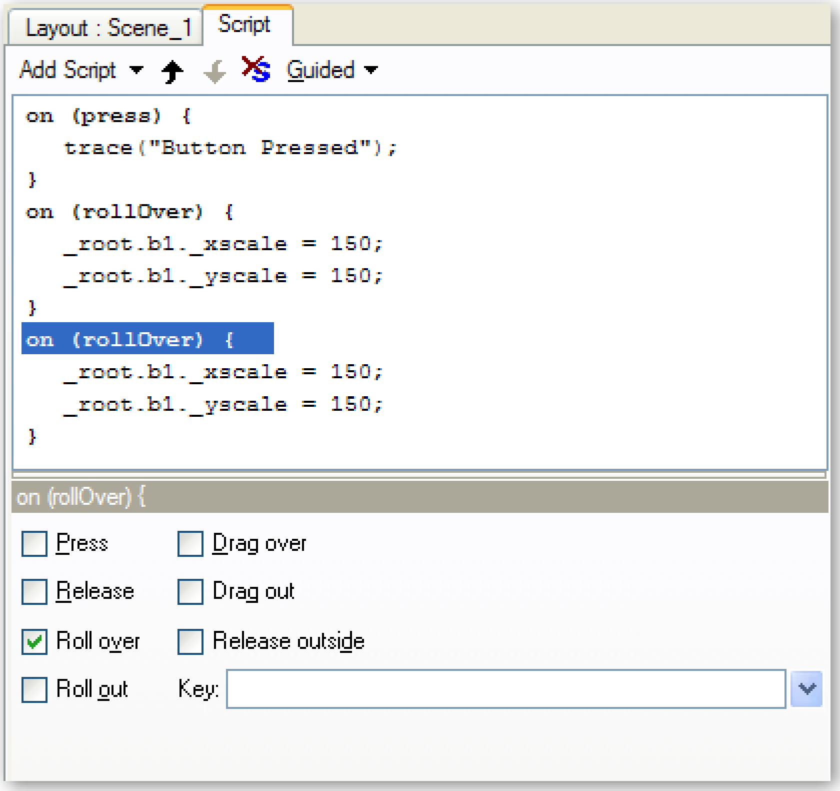
Name: _yscale

Operator: = (set value)

150

5. Create an on(rollOut) Button Event to return the button to its original size when the mouse moves away from the button. This Event can be created by repeating **Step 4** using the on(rollOut) Event function. A quicker way to create the new Event is to copy the existing one and modify it appropriately, as shown below:

   Use the right mouse button to right-click on the **on (rollOver) {** function definition. Use the Copy and Paste options to duplicate the existing function. The resulting script should look like this:



```
on (press) {
   trace("Button Pressed");
}
on (rollOver) {
   _root.b1._xscale = 150;
   _root.b1._yscale = 150;
}
on (rollOver) {
   _root.b1._xscale = 150;
   _root.b1._yscale = 150;
}
```

on (rollOver) {

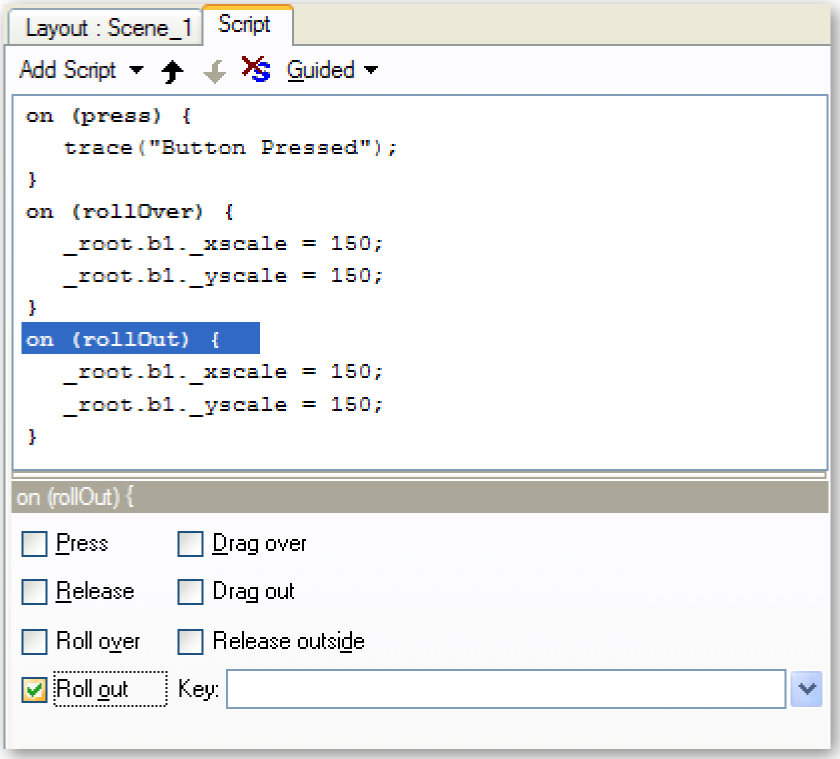
□ Press □ Drag over

□ Release □ Drag out

☑ Roll over □ Release outside

□ Roll out Key:

Uncheck the 'Roll over' checkbox and check the 'Roll out' checkbox.
The resulting script should look like this:



```
on (press) {
   trace("Button Pressed");
}
on (rollOver) {
   _root.b1._xscale = 150;
   _root.b1._yscale = 150;
}
on (rollOut) {
   _root.b1._xscale = 150;
   _root.b1._yscale = 150;
}
```

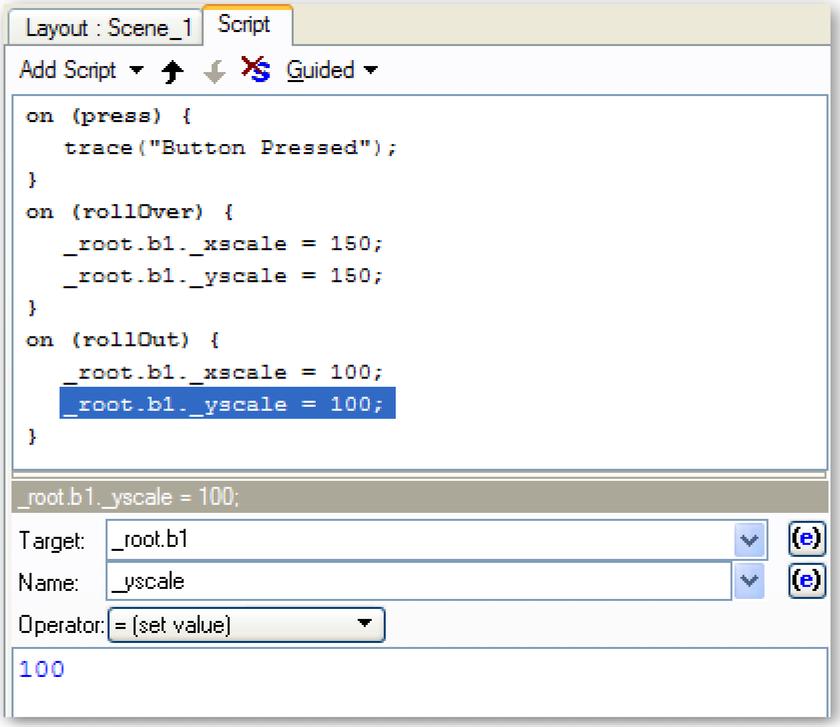
on (rollOut) {

☐ Press ☐ Drag over
☐ Release ☐ Drag out
☐ Roll over ☐ Release outside
☑ Roll out Key:

Now modify the statements contained in the on (rollOut) Event function so that they are:

```
_root.b1._xscale = 100;
_root.b1._yscale = 100;
```

The resulting script should look like this:



```
on (press) {
   trace("Button Pressed");
}
on (rollOver) {
   _root.b1._xscale = 150;
   _root.b1._yscale = 150;
}
on (rollOut) {
   _root.b1._xscale = 100;
   _root.b1._yscale = 100;
}
```

_root.b1._yscale = 100;

Target: _root.b1
Name: _yscale
Operator: = (set value)

100

6. Test the button by select the Debug tab among the Panels displayed on the right hand side of the 'Layout' Panel. Press the 'Play' button to run your Movie. Moving the mouse over the button, should cause the button

to become larger. Once the mouse moves away from the button, the button should return to its normal size. Pressing the left mouse button while the mouse is positioned over the button will cause the words "Button Pressed" to be displayed in the 'Debug' Panel (which was covered in the 'Button' tutorial)

Analysis
Additional Event Handling routines were added to the button to handle the on(rollOver) and on(rollOut) Button Events. When the on(rollOver) Event occurs, the button is scaled to 150% of its original size. When the on (rollOut) Event occurs, the button is scaled to 100% of its original size.

The scaling is achieved by changing the _xscale and _yscale properties. As the button is an Object that occurs in the main Movie, its properties can be referenced via _root.b1.<property>.

By design, Button Events are designed to work on the containing Object. Consequently, if the commands in the Button Event routines were entered as this._xscale = ... or simply _xscale = ..., the Effect would be to scale the main Movie, not the button. In contrast, the onSelfEvent() routine works on the current Object.

Movie Control

Top Previous Next

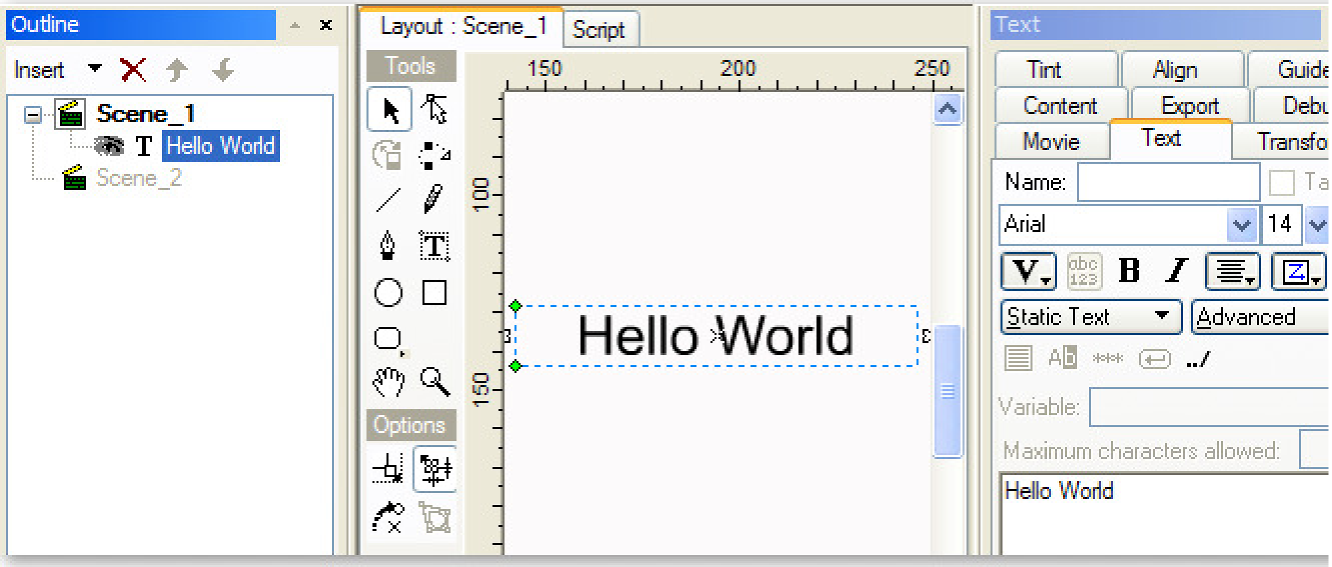
Description:
Demonstrates how to control timelines playing in Sprites and how Sprites can control the main movie timeline.

Aim
This tutorial introduces various uses of the gotoAndPlay() actions.

1. Open the file "helloworld.swi" ... save the file as "mymoviecontrol.swi".

2. On the 'Outline' panel, use the Insert menu and insert a new Scene into the movie - it should be titled Scene_2 by default.

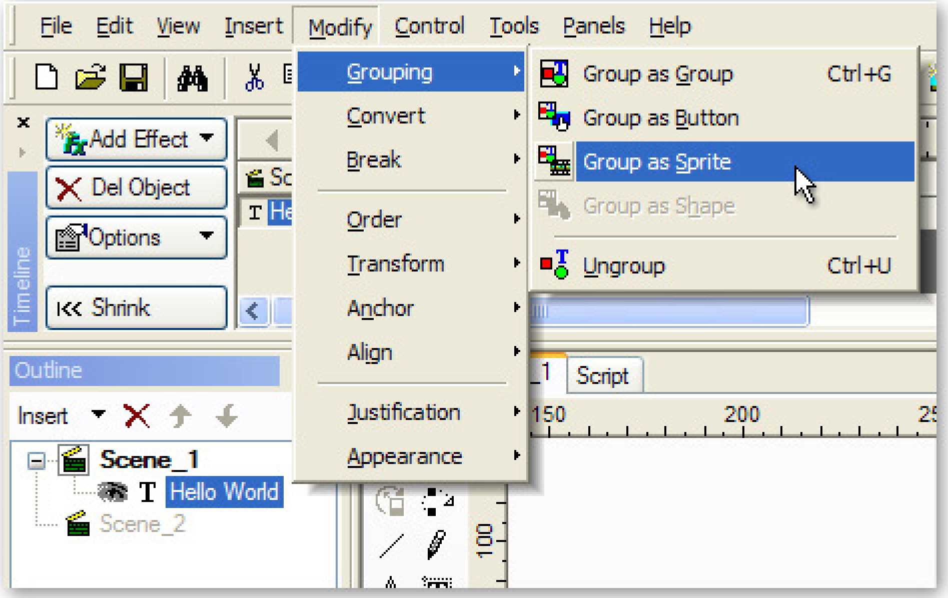
3. Click on "Scene_1" in the 'Outline' panel and insert a Text object into the movie by using the text tool. Change the default text on the text panel to read "Hello World" and make sure that it is a Static text object. Set the text justification to Center ≡ by using the Justification options drop-down menu ≡ on the text panel, and make sure the text is using the Vector character options V as shown below:



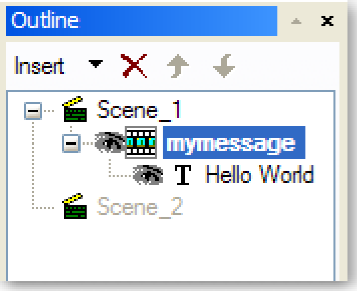
Name:
Arial 14
Static Text
Variable:
Maximum characters allowed:
Hello World

4. Make sure the text object is highlighted in the 'Outline' panel, then use the Modify menu and select **Grouping |**

Group as Sprite (or you can also Right-Click on the text object in the Outline panel and follow the same procedure.

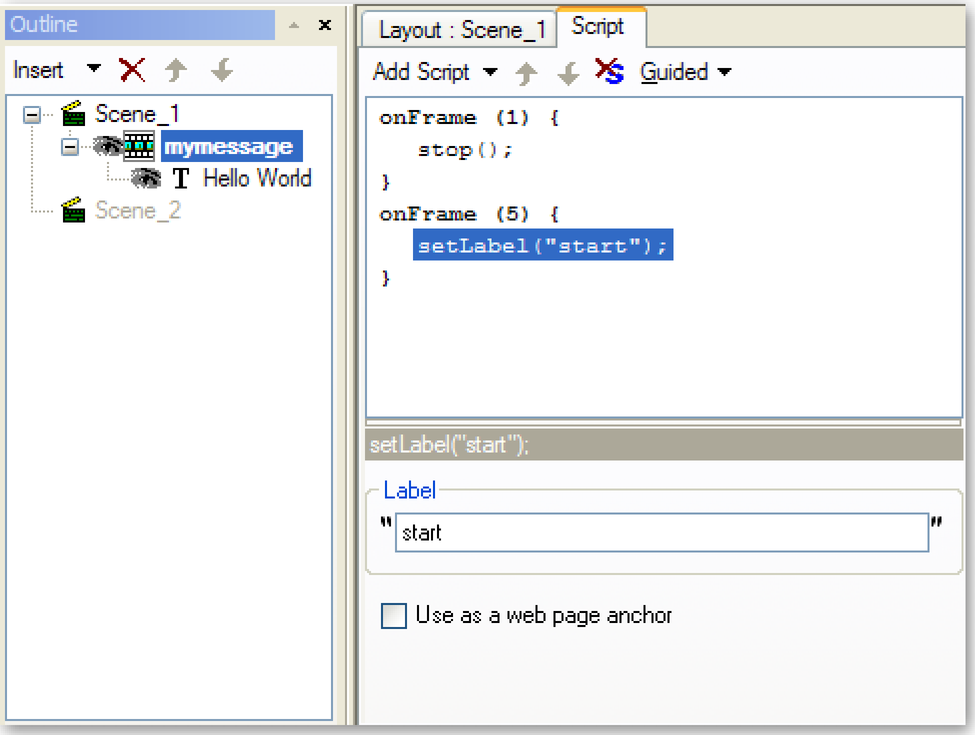


5. Name the sprite "mymessage" on the Sprite panel. Make sure to select (highlight) the sprite in the 'Outline' panel and click on the (+) icon to open it - this will allow you to edit the timeline inside the sprite.



5. Open the 'Script' panel. Press the 'Add Script' button and select **Events | Frame | onFrame(...)** from the menu - leave the frame number set to "1". Next, press the 'Add Script' button again and select **Movie Control | stop()** from the menu.

6. Press the 'Add Script' button again and add another **onFrame()** event and change the frame number to "5". Use the 'Add Script' button again and select **Frame | setLabel()** from the menu and set the label name as "start". The script panel should appear as show below:



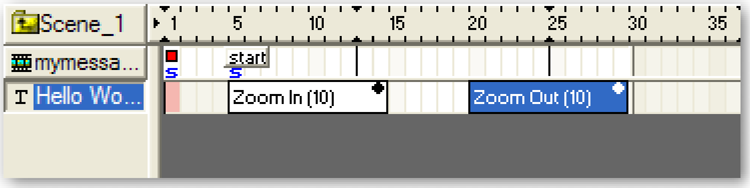
```
onFrame (1) {
   stop();
}
onFrame (5) {
   setLabel("start");
}
```

setLabel("start");

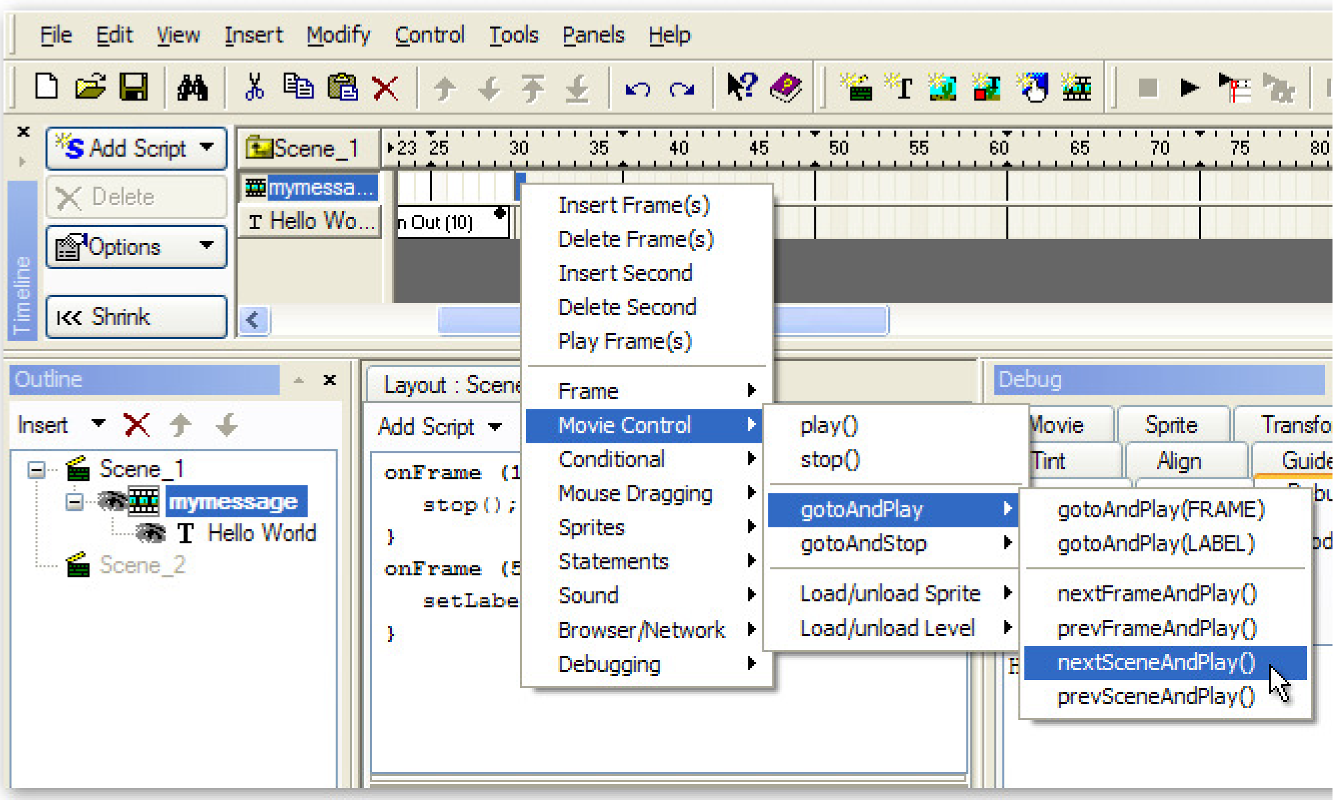
Label: "start"

☐ Use as a web page anchor

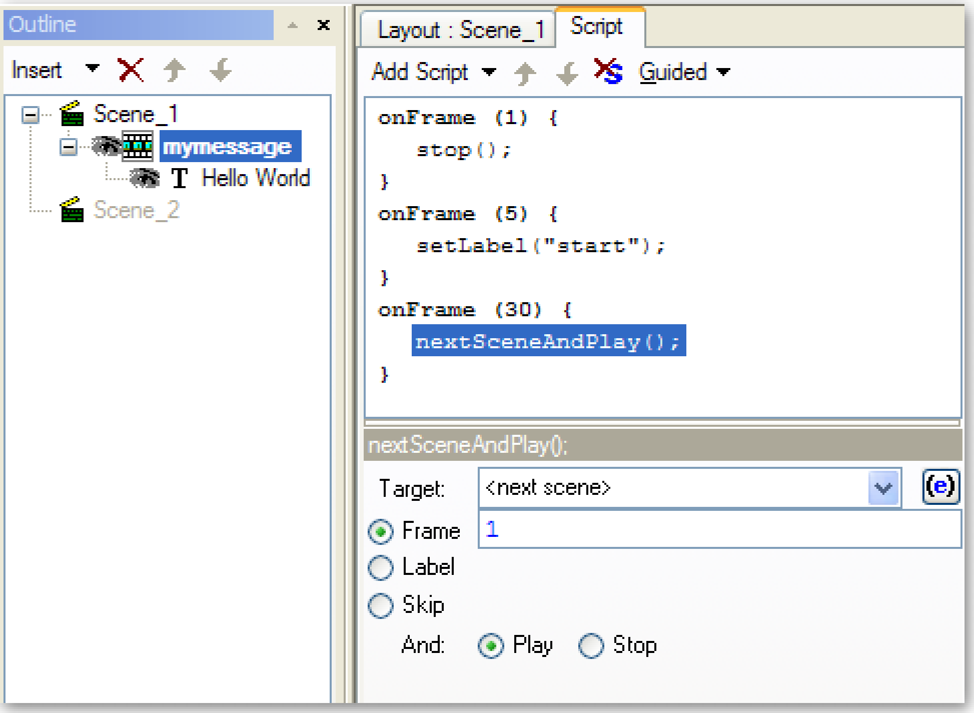
7. In the 'Timeline' panel, Right-Click at Frame 5 on the row for the "Hello World" text object and select **Zoom | Zoom In** from the menu. Next, Right-Click at Frame 20 on the same row and select **Zoom | Zoom Out** from the menu. The 'Timeline' panel should be shown as below:



8. Right-Click on Frame 30 for the "mymessage" sprite row (directly under the ruler) and select **Movie Control | gotoAndPlay | nextSceneAndPlay()** from the script menu.



9. The 'Script' panel should be show as below:



```
onFrame (1) {
   stop();
}
onFrame (5) {
   setLabel("start");
}
onFrame (30) {
   nextSceneAndPlay();
}
```

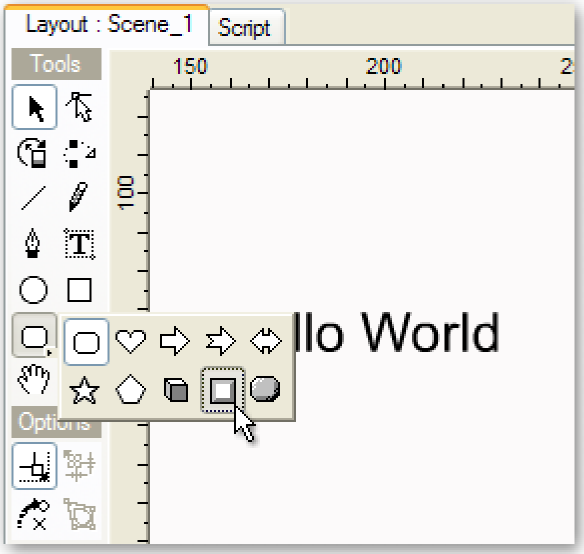
nextSceneAndPlay();

Target: <next scene>

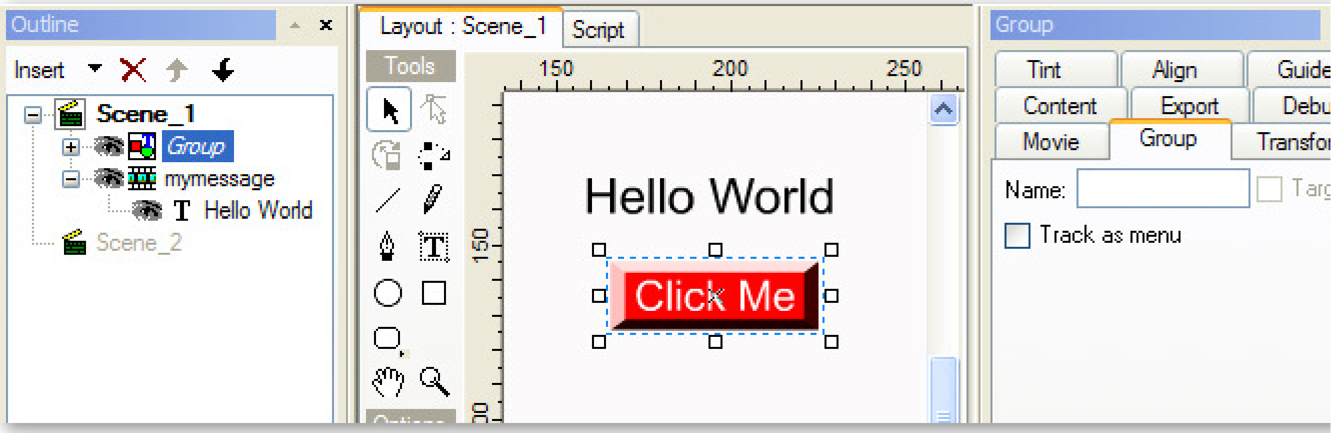
● Frame 1
○ Label
○ Skip

And: ● Play ○ Stop

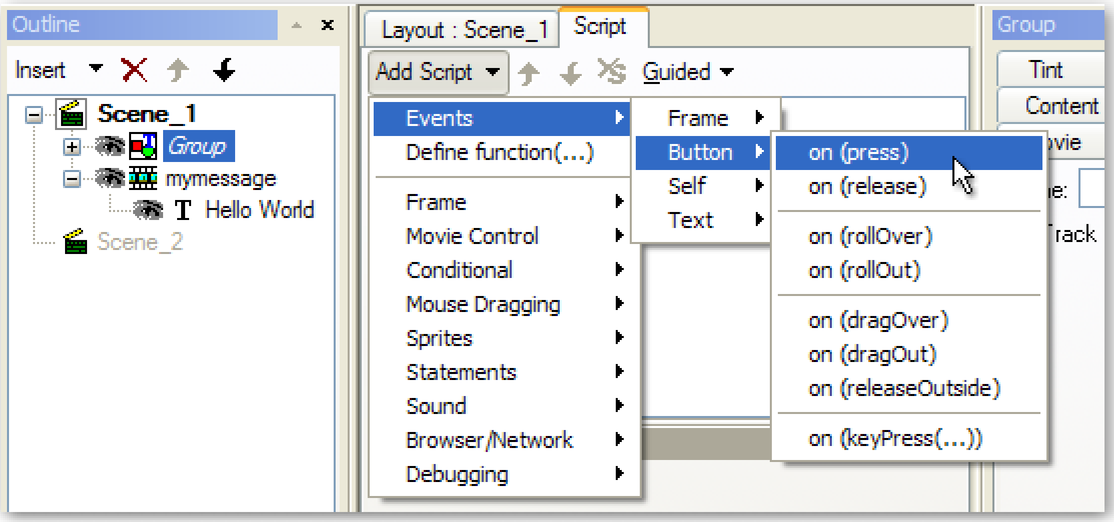
10. Click on "Scene_1" in the 'Outline' panel and click on the AutoShape tool. Draw a Beveled Button shape on the Layout panel below your text object.



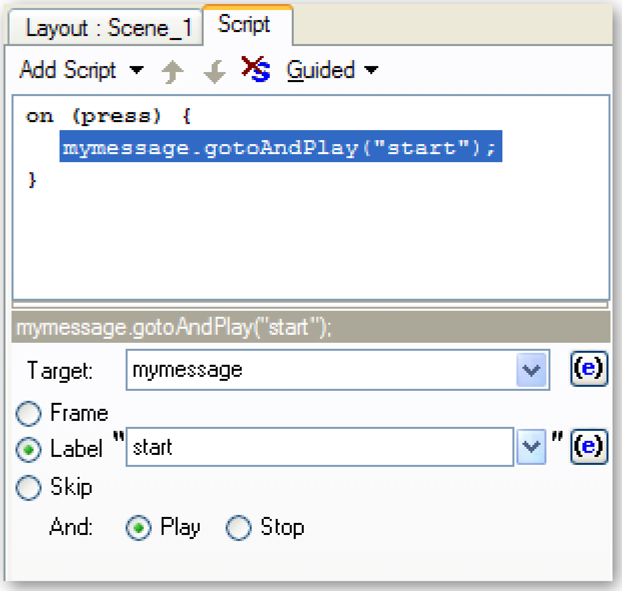
11. Insert a Static text object using the text "Click Me" and position it over top of the beveled button shape. Select both the text object and the beveled button shape (hold down the CTRL key while clicking on each object in the 'Outline' panel) then use the Modify menu and select **Grouping | Group as Group** from the menu.



12. With the Group selected in the 'Outline' panel, open the 'Script' panel and press the 'Add Script' button. Select **Events | Button | on(press)** from the menu.



13. Press the 'Add Script' button again and select Movie Control | gotoAndPlay | gotoAndPlay(LABEL) from the menu. Use the 'Target' drop-down menu and select the "mymessage" sprite from the list then select the "start" label. The resulting script should be shown as below:



```
on (press) {
  mymessage.gotoAndPlay("start");
}
```

mymessage.gotoAndPlay("start");

Target: mymessage

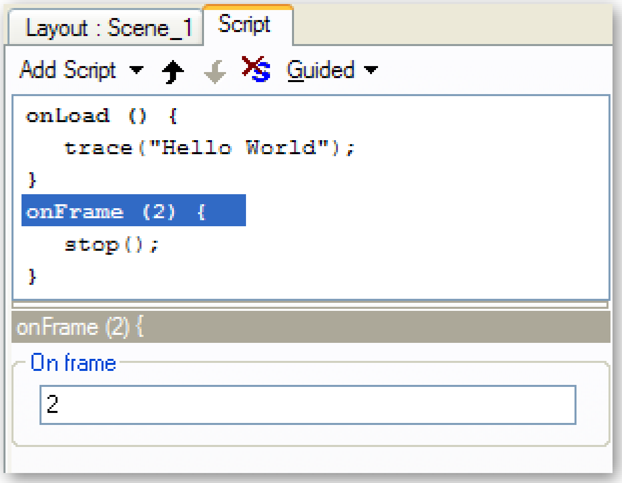
○ Frame

● Label "start"

○ Skip

And: ● Play ○ Stop

14. Select "Scene_1" on the 'Outline' panel and add a stop() action on Frame 2 (using the methods described above).



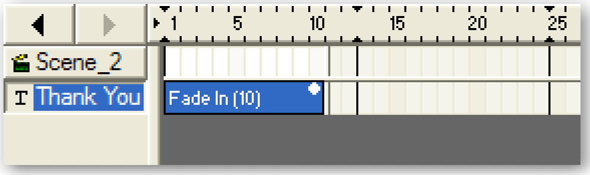
```
onLoad () {
   trace("Hello World");
}
onFrame (2) {
   stop();
}
```

onFrame (2) {

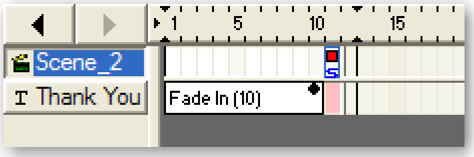
On frame: 2

15. Select "Scene_2" on the 'Outline' panel and insert a Static Text object. Change the text to read "Thank You".

16. In the 'Timeline' panel, Right-Click at Frame 1 on the row for the "Thank You" text object and select **Fade | Fade In** from the menu. The 'Timeline' panel should be shown as below:



17. Right-Click at Frame 11 on the row for "Scene_2" in the 'Timeline' panel and select **Movie Control | stop()** from the menu.



18. Make sure you are in the 'Layout' panel then press the ▶ 'Play Movie' button from the Control Toolbar. When the movie starts playing, press the "Click Me" button. You should see the text "Hello World" zoom in/out then the text "Thank You" will fade in.

19. Press the 'Stop' movie button and save the .swi file.

Analysis
When the "Click Me" button is pressed, it sends a command to the Sprite "mymessage" telling it to go to the label "start". At the label the text animation begins, and when the animation stops (at Frame 30) there is an action inside the Sprite telling the main timeline to go to the next Scene and play. By using the gotoAndPlay() action - you can set different Targets to send commands from the main timeline to a Sprite and from the Sprite back to the main timeline.

Calculator

Top Previous Next

Description
This is a step-by-step tutorial for creating a simple calculator that calculates the area of a circle, given the radius

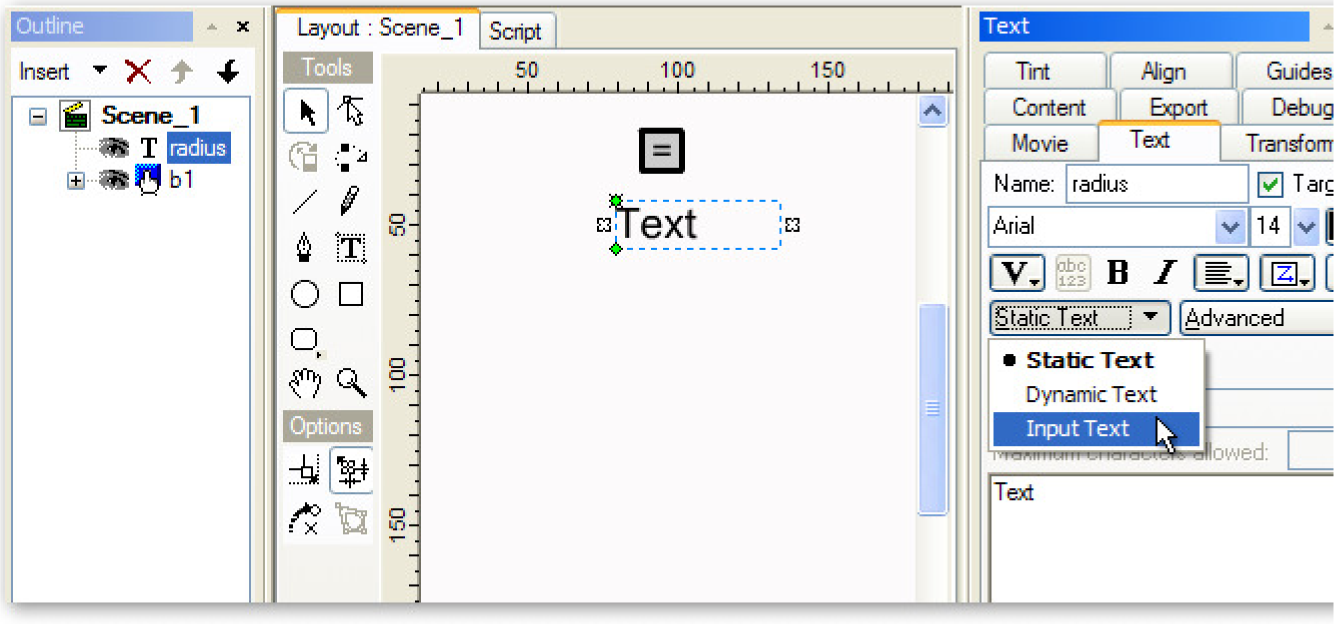
of the circle (PI * r * r).

Aim
This tutorial introduces the Text Object with Input and Dynamic text properties and the Math Object. It also uses the on(press) Button Event developed in the previous tutorial.

.swi file
"calculator.swi"

1. Continue from the previous tutorial or load the file "button1.swi" and save as "mycalculator.swi"

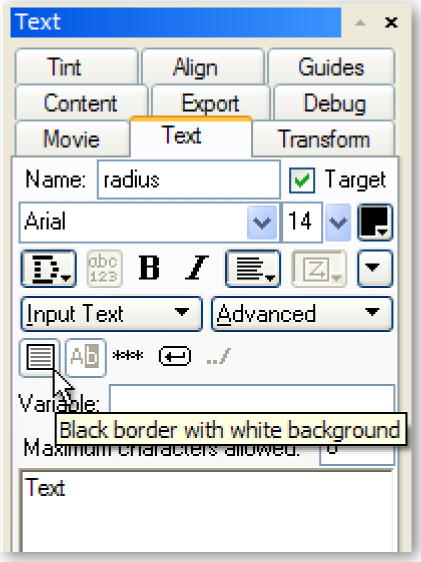
2. Insert a Text Object using the Text Tool or the Menu items Insert | Text. Name the new Text Object radius, tick the 'Target' checkbox and set to the text type to Input in the Text Panel



Name: radius ☑ Targ

Arial 14

3. Select the "Advanced Options" for this Text Object and set the option for 'Black Border with White Background' ▤ as shown below
Note: The handles around the Text Object change once the type is changed to an Input Text Object



Name: radius ☑ Target

Arial 14

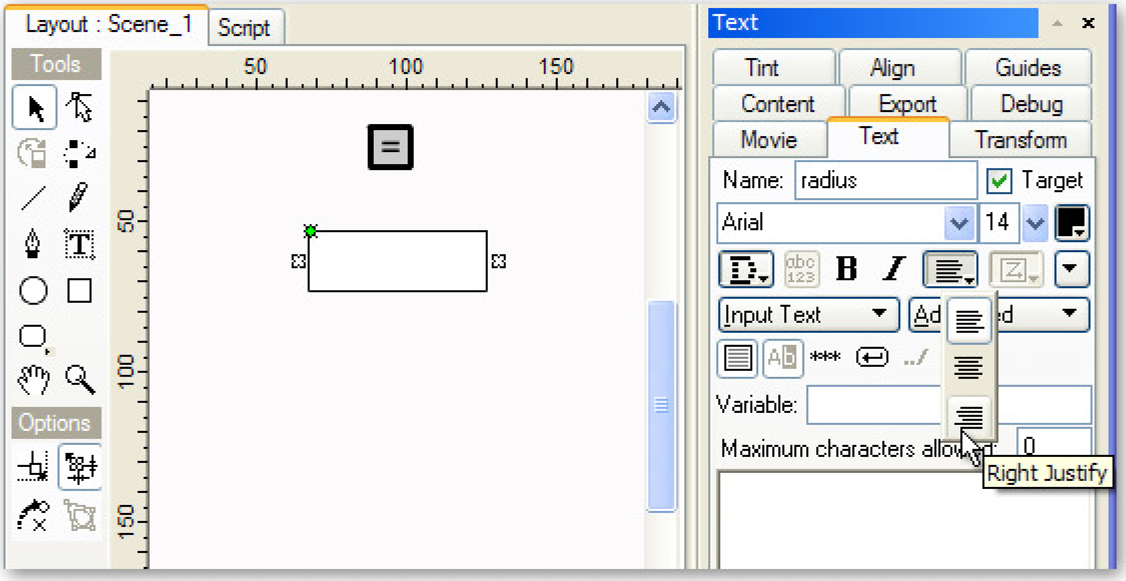
Input Text Advanced

Variable:

Maximum characters allowed: 0

Text

4. Clear the 'Text' field, so the Object appears as an empty input field. Select right justification, as shown below:



Name: radius ☑ Target

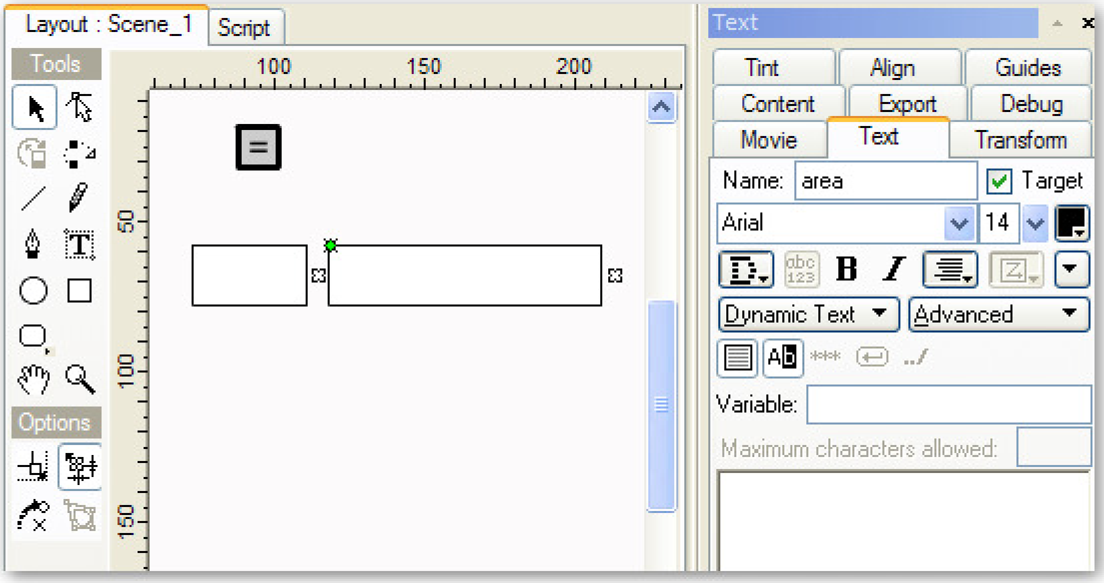
Arial 14

Input Text

Variable:

Maximum characters allowed: 0

5. Create a Dynamic Text Object to display the result of the calculation by using the Text Tool or Insert | Text to insert a Text Object. Name the new Text Object "area", tick the 'Target' checkbox and set to the text type to Dynamic in the Text Panel. Clear the text so that an empty text field is displayed and select right justification. Use the Selection Tool to stretch the text field using the cross-shaped handles, as shown below. The advanced settings will be the same as those selected for the previous Text Object and do not need to be altered at this stage

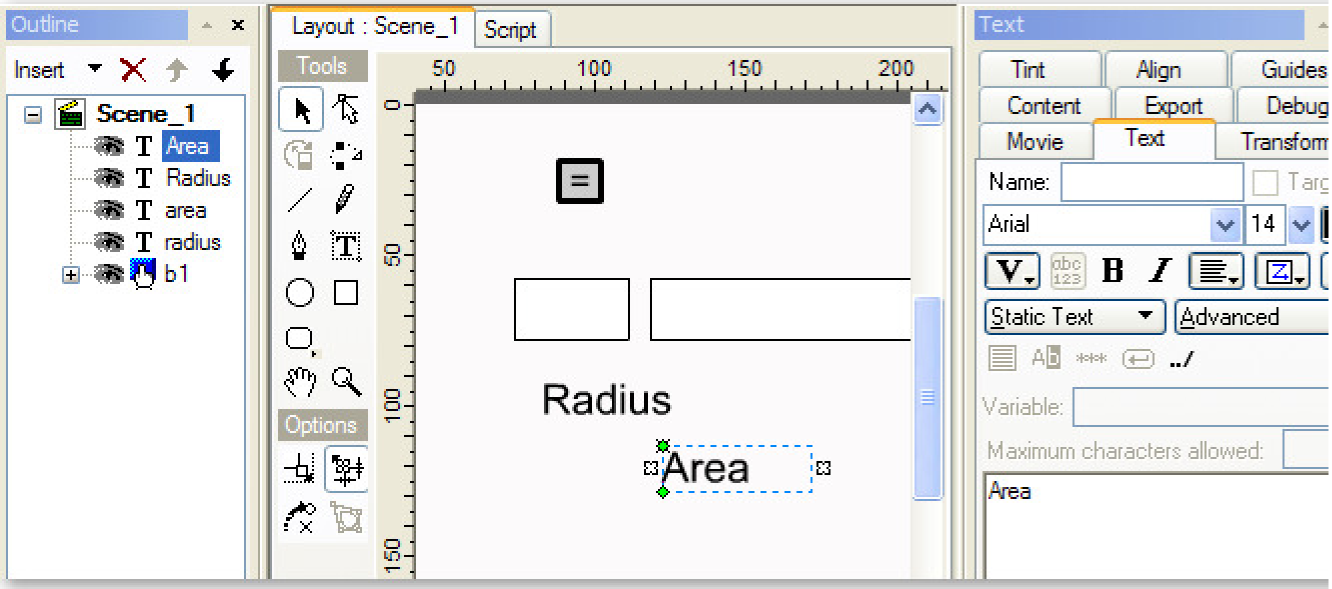


Name: area ☑ Target

Arial 14

Dynamic Text Advanced

6. Use the Text Tool or Insert | Text to insert some Static text ("area" and "radius") to be used as headings



Name: □ Targ

Arial 14

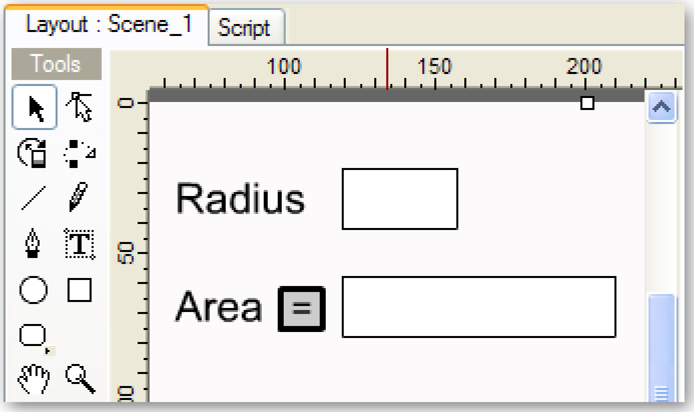
Static Text Advanced

Area

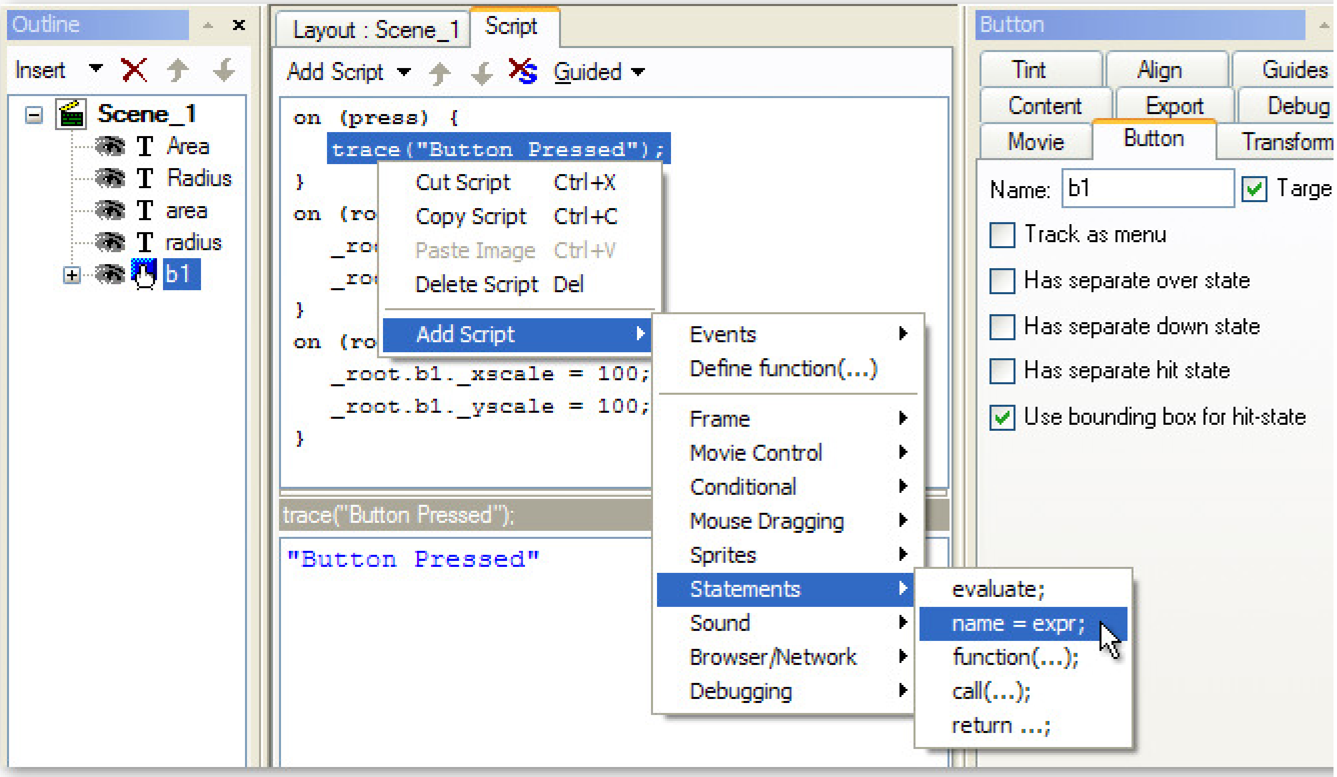
Note:

- If the text has a line border around it, use the 'Advanced...' button to turn the border off
- Although the new Text Objects appear in the 'Outline' Panel with the same names as the previously inserted Text Objects, a conflict will not occur at the Scripting level as the new Text Objects are not Scripting Objects (they are unnamed and do not have the 'Target' checkbox set)

7. Use the Select Tool to re-arrange the Objects as shown below:



8. Modify the on(press) Button Event function to calculate the area of the circle when the button is pressed. Select the button then select the 'Script' Panel. Right-click on the trace statement within the on (press) function and select Statements | name = expr;



```
on (press) {
    trace("Button Pressed");
}
on (ro
    _ro
    _ro
}
on (ro
    _root.b1._xscale = 100;
    _root.b1._yscale = 100;
}
```

trace("Button Pressed");

```
"Button Pressed"
```

Name: b1 ☑ Targe

□ Track as menu

□ Has separate over state

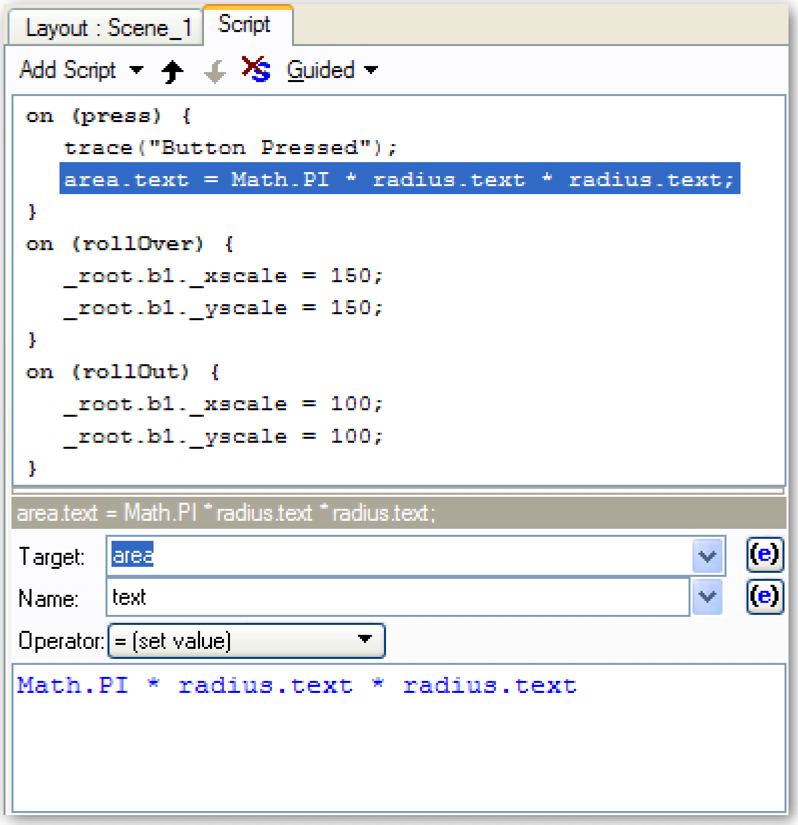
□ Has separate down state

□ Has separate hit state

☑ Use bounding box for hit-state

For the 'Target:' parameter, select area (the Dynamic Text Object added in step 5) from the pull-down list.
For the 'Name:' parameter, select Text | text from the pull-down list.
Leave the operator as "=".
Enter the formula "Math.PI * radius.text * radius.text" in the formula area

Note: "Math.PI" is case sensitive. "PI" could also have been used



```
on (press) {
   trace("Button Pressed");
   area.text = Math.PI * radius.text * radius.text;
}
on (rollOver) {
   _root.b1._xscale = 150;
   _root.b1._yscale = 150;
}
on (rollOut) {
   _root.b1._xscale = 100;
   _root.b1._yscale = 100;
}
```

area.text = Math.PI * radius.text * radius.text;

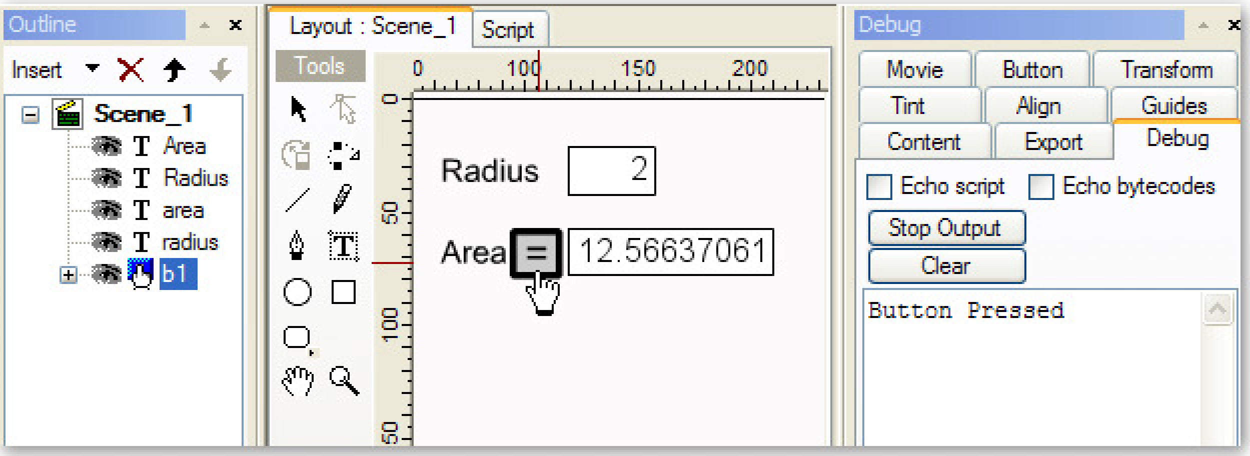
Target: area

Name: text

Operator: = (set value)

```
Math.PI * radius.text * radius.text
```

9. Test the calculator by reselecting the 'Layout' Panel and press the 'Play' button (fonts may appear to change because system fonts are used). Enter the number 2 into the 'radius' text field and click the '=' button. The following should be displayed in the 'Layout' Panel.



Radius 2

Area = 12.56637061

□ Echo script □ Echo bytecodes

```
Button Pressed
```

Analysis

The input Text Object radius is a Scripting Object. It is used to allow the entry of the input parameter, radius, the radius of a circle.

Within the script, the input value can be accessed via the name "radius.text". "radius" is the name of the Object, "text" is the property that contains the entered value.

When the button is pressed, the script contained within the button on(press) Event Handler is executed. This results in the calculation Math.PI * radius.text * radius.text being placed into "area.text", the text property of a Dynamic Text Object named "area". This results in the display of the calculated value.

Note: The area of a circle is given by the formula "PI * r * r"

Input and Dynamic Text

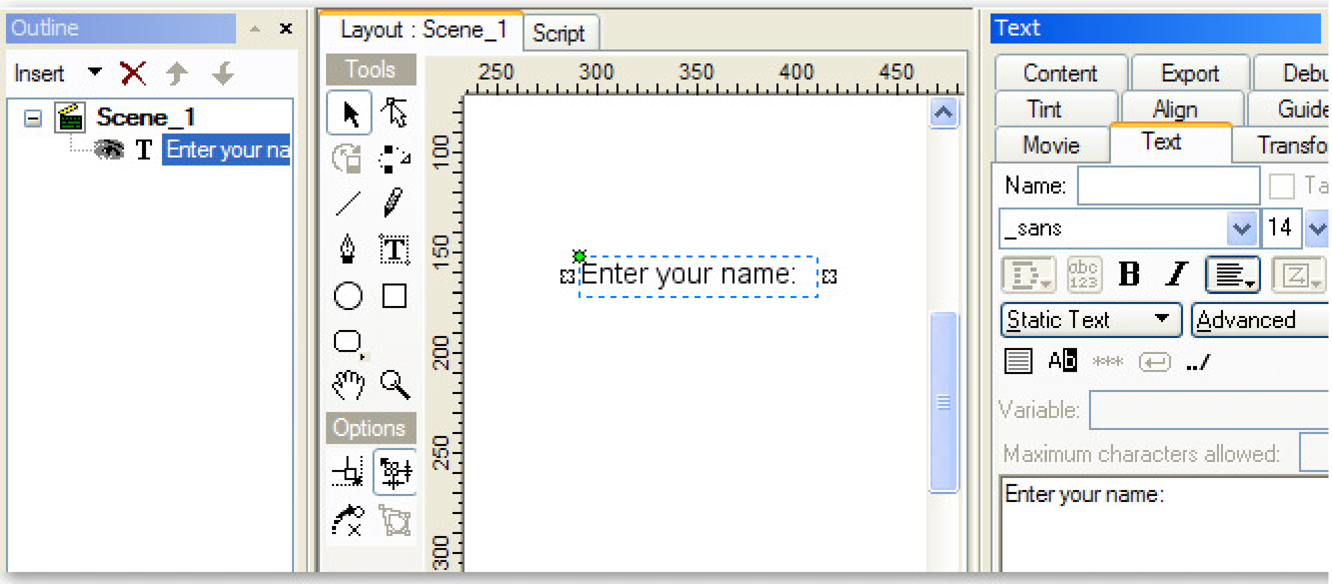
Top Previous Next

Description
Demonstrates how users can add text to the movie to be used at a later time in the same movie.

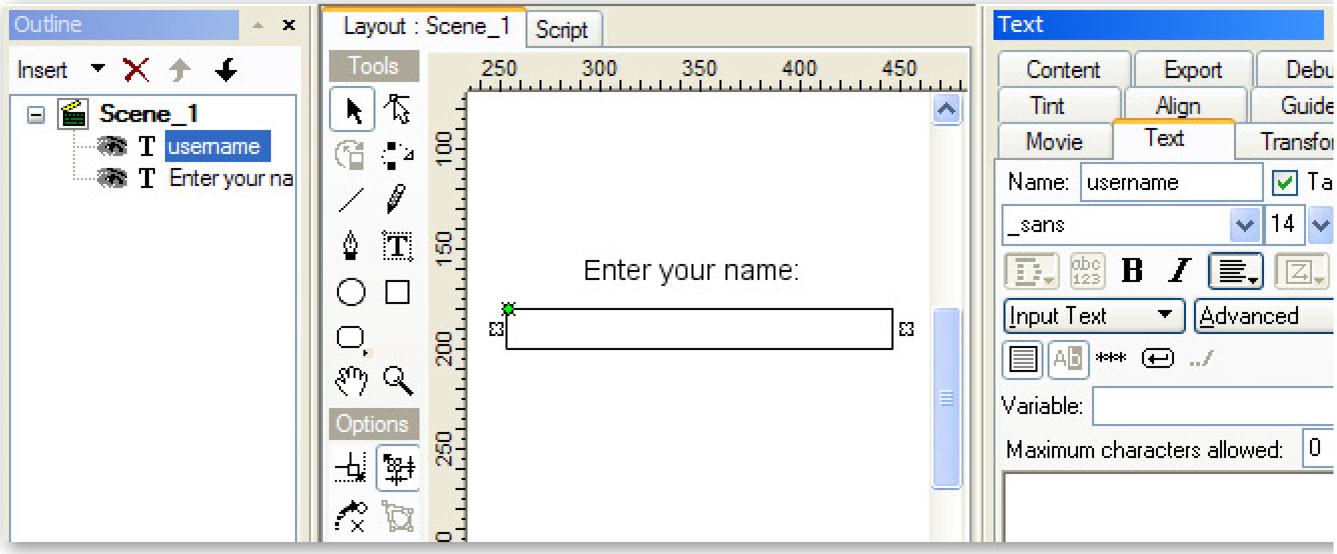
Aim
Introduces methods for user input through Input Text fields and storing that information as a variable to be used in dynamic content.

1. Create a new file and save the file as "mywelcome.swi"

2. Insert a Static text object and type in the text "Enter your name:" on the text panel.

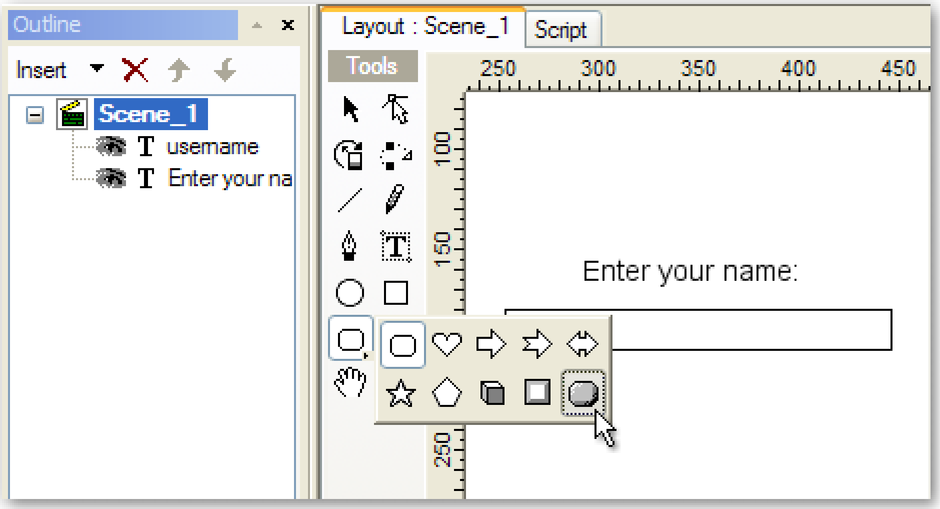


3. Insert an Input Text object. Name the text field "username" and tick the 'Target' option on the text panel. Delete the default "text". Open the Advanced text options and press the button for a black border and white background ▤

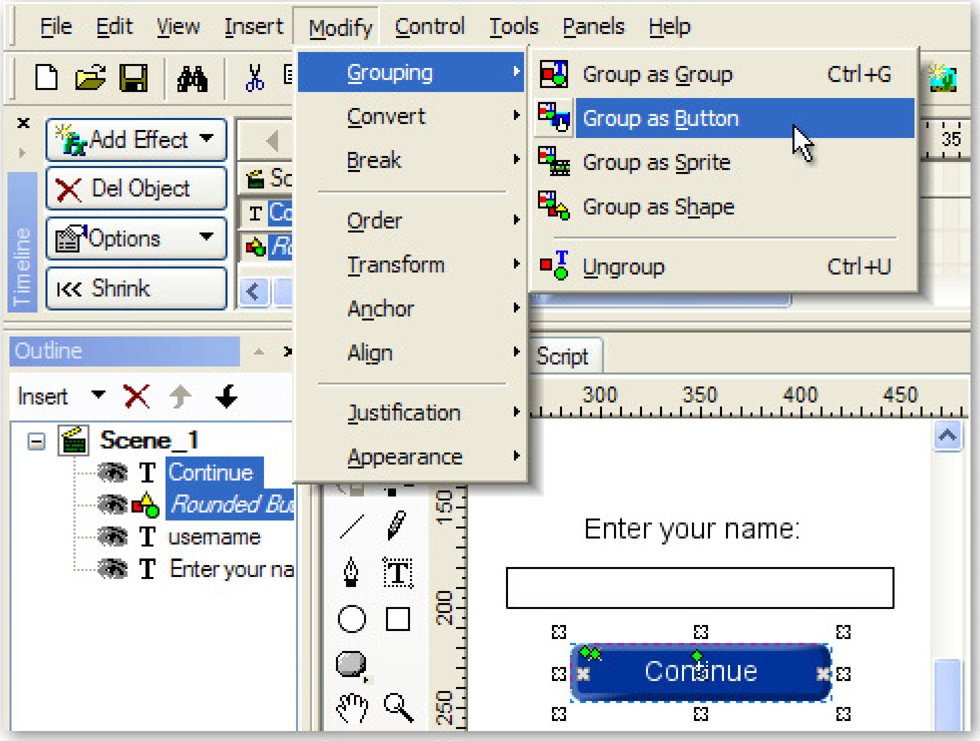


Name: username
_sans 14
Input Text
Variable:
Maximum characters allowed: 0

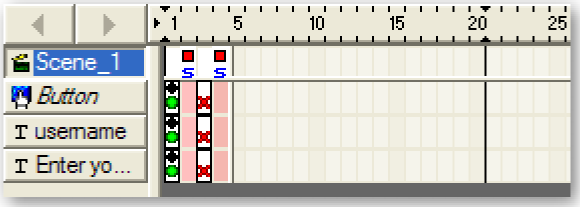
4. Use the AutoShape tool to draw a 'Rounded Button' below the Input text field. Change its color to Blue on the Shape panel.



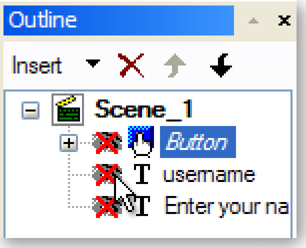
5. Insert a Static text object with the text "Continue" and place it over top of the Rounded Button. Hold down the CTRL key and click on both the Static text object and the Rounded Button shape to highlight them. Use the Modify menu and select **Grouping | Group as Button**.



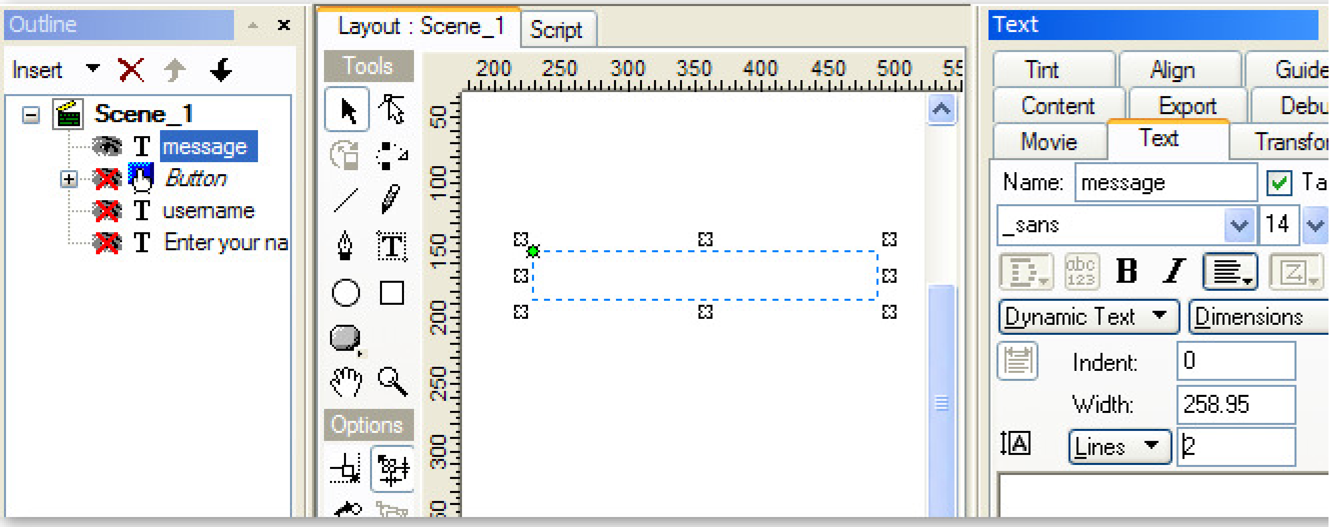
6. In the 'Timeline' panel, at Frame 1 add a Place effect for all the objects. Add a 'Stop' - **Movie Control | stop()** - action on Frame 2 of the Scene_1 row in the 'Timeline'. At Frame 3 for each of the objects add a Remove effect then add another 'Stop' action at Frame 4 of the Scene_1 row. The Timeline should be shown as below:



7. Click on the Eye icon next to each object in the 'Outline' panel to hide them from the stage.

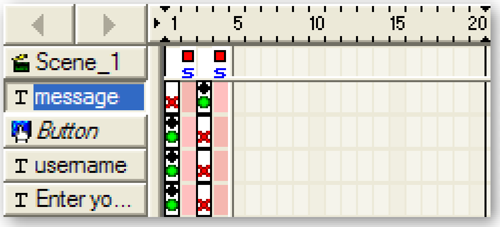


8. Insert a Dynamic text object and name it "message" on the text panel - remember to tick the 'Target' option as well. Open the 'Dimensions' text options and untick the Auto-Height option. Set it to use 'Lines' rather than 'Height' and enter '2' as the number of lines it should use.



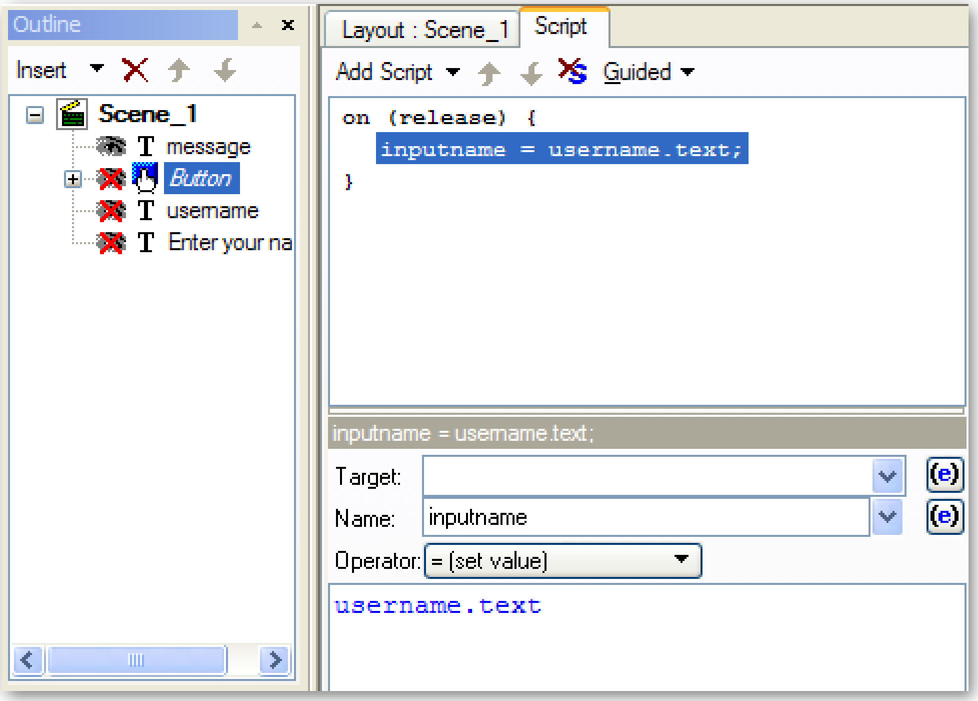
Name: message
_sans 14
Dynamic Text
Indent: 0
Width: 258.95
Lines 2

9. In the 'Timeline' panel on the "message" row, insert a Remove effect at Frame 1 and a Place effect at Frame 3. The 'Timeline' panel should be shown as below:



10. Select the Button object in the 'Outline' panel then open the 'Script' panel. Press the 'Add Script' button and select **Events | Button | on(release)** from the menu.

11. Press the 'Add Script' button again and select **Statements | name = expr;** from the menu. Leave the Target field empty and enter "inputname" into the Name field. Leave the Operator set at "=" then enter "username.text" in the value field. Here we are creating a new Variable named "inputname" that will store the text typed into the Input text field by the user. The 'Script' panel for the Button object should be shown as below:



```
on (release) {
    inputname = username.text;
}
```

inputname = username.text;

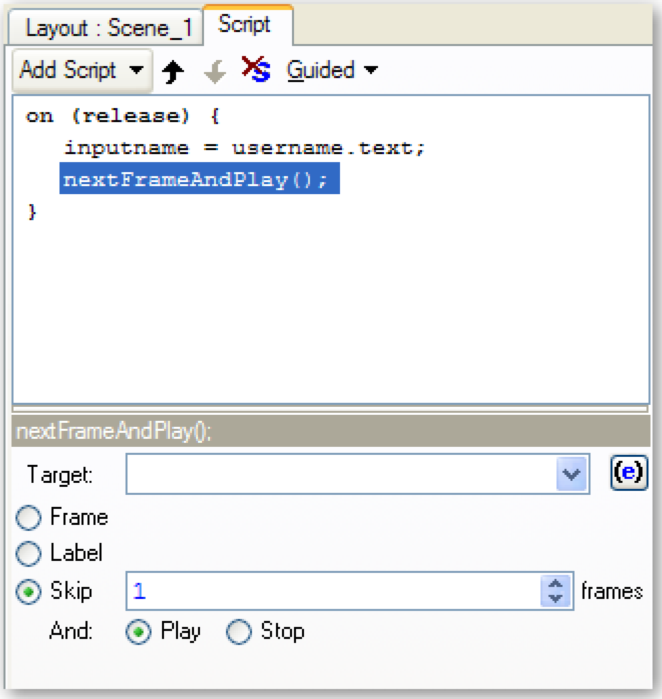
Target:

Name: inputname

Operator: = (set value)

```
username.text
```

12. Press the 'Add Script' button again and select **Movie Control | gotoAndPlay | nextFrameAndPlay()** from the menu.



```
on (release) {
    inputname = username.text;
    nextFrameAndPlay();
}
```

nextFrameAndPlay();

Target:

○ Frame

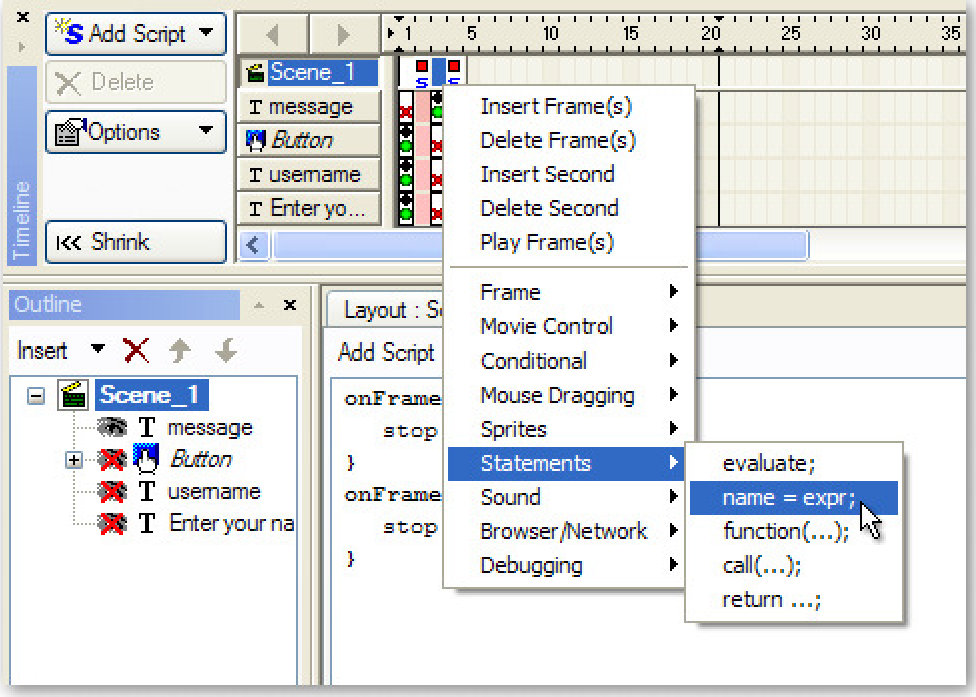
○ Label

● Skip 1 frames

And: ● Play ○ Stop

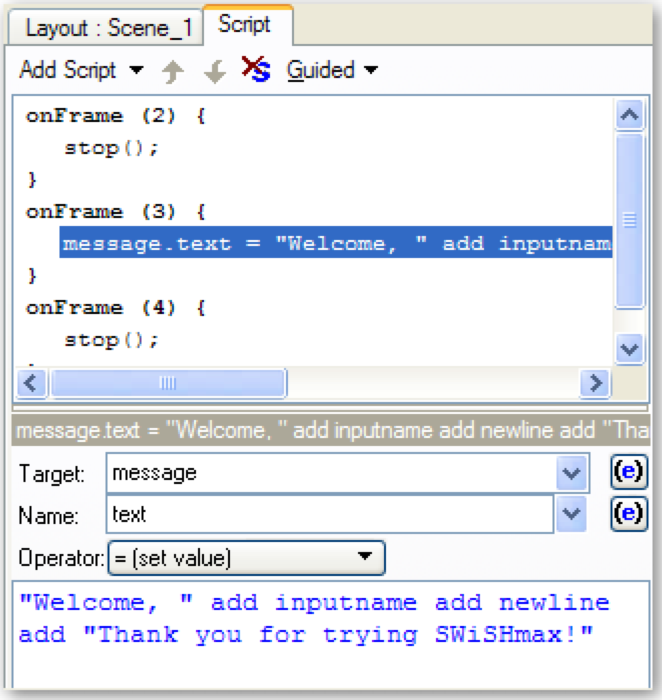
Note: It is important that the variable "inputname" be defined before the gotoAndPlay() action is executed.

13. In the 'Timeline' panel, Right-Click at Frame 3 on the row for Scene_1 and select **Statements | name = expr;** from the list.



```
onFrame
    stop
}
onFrame
    stop
}
```

14. Use the drop-down menu next to the Target field and select the Dynamic text object 'message' from the list. In the Name field use the drop-down menu and select **Text | text.** Leave the Operator set to "=" and enter the text shown in the image below:



```
onFrame (2) {
   stop();
}
onFrame (3) {
   message.text = "Welcome, " add inputnam
}
onFrame (4) {
   stop();
```

message.text = "Welcome, " add inputname add newline add "Tha

Target: message

Name: text

Operator: = (set value)

```
"Welcome, " add inputname add newline
add "Thank you for trying SWiSHmax!"
```

Note: The text displayed inside quotes is the static message that will be displayed. Using the "add" is the same as adding one string at the end of another (which is why you enter spaces manually inside quotes). We have added the variable "inputname" which will be entered by the user. The "newline" command will enter a carriage return into the text field (starting a new line).

15. Go back to the 'Layout' panel and press the 'Play' button to preview the movie. Type your name into the input

text field then press the Continue button. The welcome message should be displayed with the name you entered.

Note: If you only see one line of text displayed in the Message, you will either need to increase the width of the Dynamic text object or make sure that you have the Auto-Height option turned off and have it set to use 2 Lines of text.

16. Press the 'Stop' button and save your movie.

Analysis
When the 'Play' button is pressed, the movie begins with the Input text field in place. After the 'Continue' button is pressed, the variable "inputname" is created and contains whatever was typed into the Input text field. This variable is then displayed in the Dynamic text field "message".

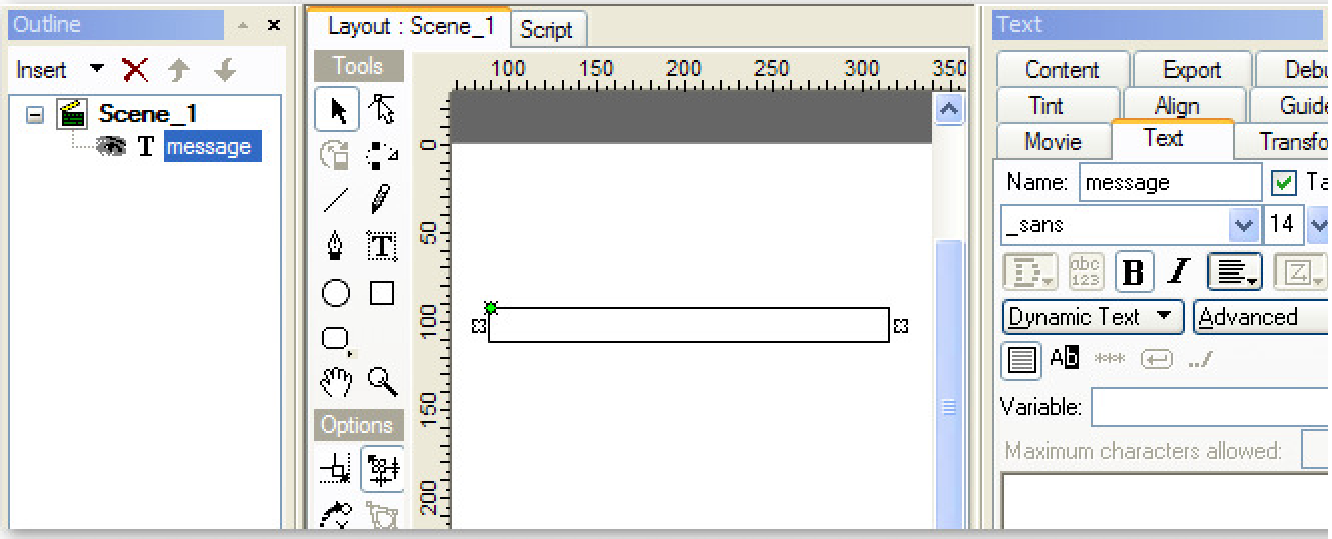
Arrays and Random

Top Previous Next

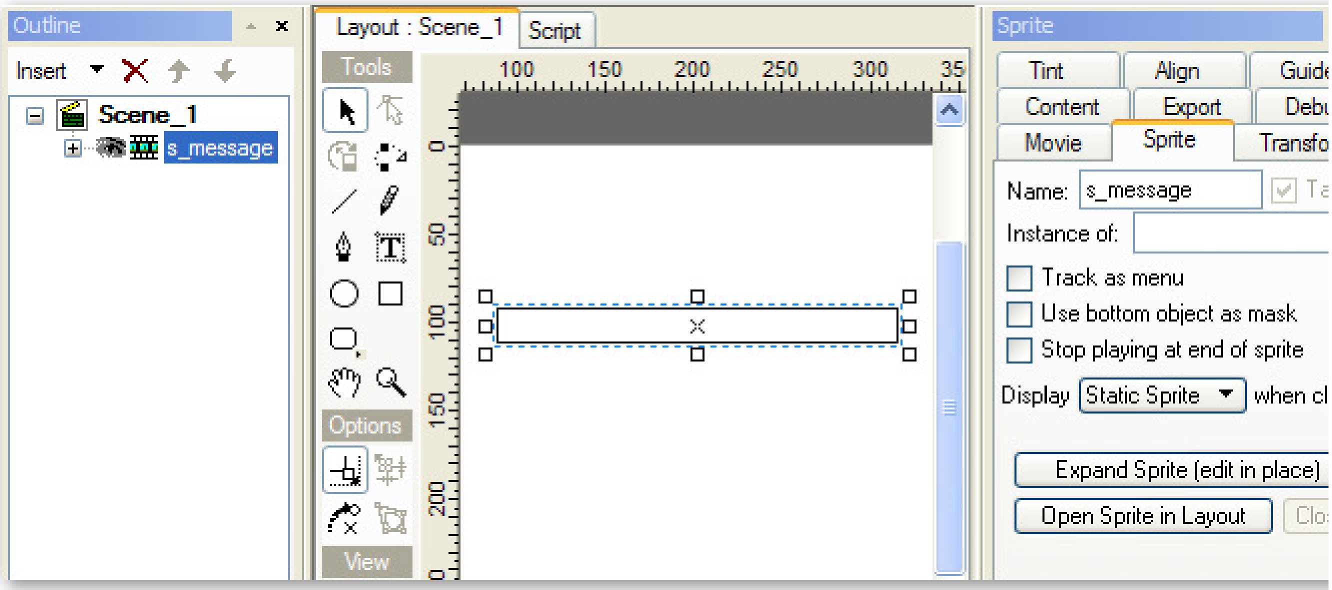
Description
Demonstrates how to use Arrays to display a random message.

Aim
Introduces methods for Arrays and Math.randomInt()

1. Create a new file and save it as "mymaxarray.swi". Insert a Dynamic text object and name it "message" on the 'Text' panel. Be sure to tick the 'Target' option so that it can be used in our script.



2. Right-Click on the text object in the 'Outline' panel and select **Grouping | Group as Sprite** from the menu. Name the sprite "s_message" on the 'Sprite' panel.

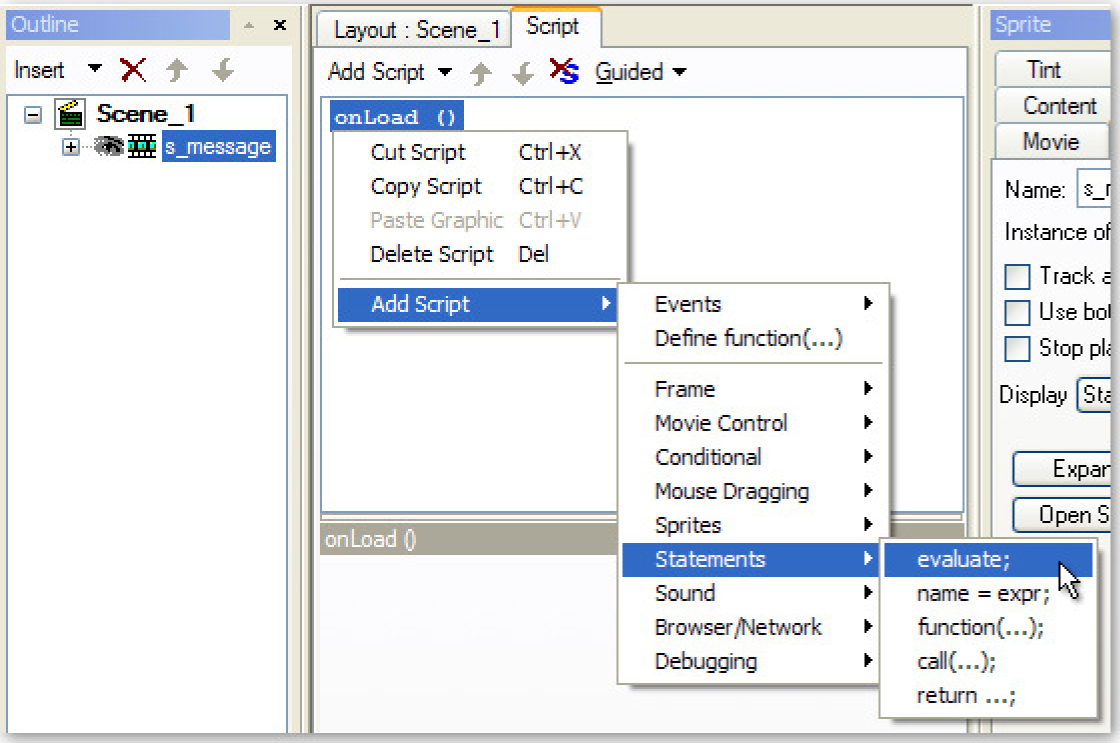


Name: s_message

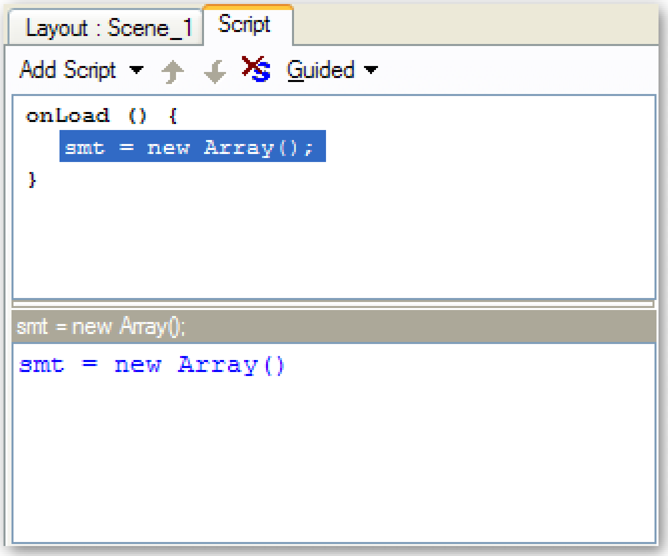
Display Static Sprite

3. With the sprite object selected in the 'Outline' panel, open the 'Script' panel and press the 'Add Script' button. Select **Events | Frame | onLoad()** from the menu.

4. Right-Click on the **onLoad()** event and select **Add Script | Statements | evaluate;** from the menu.



5. Type "smt = new Array()" into the evaluate field on the 'Script' panel as shown below:



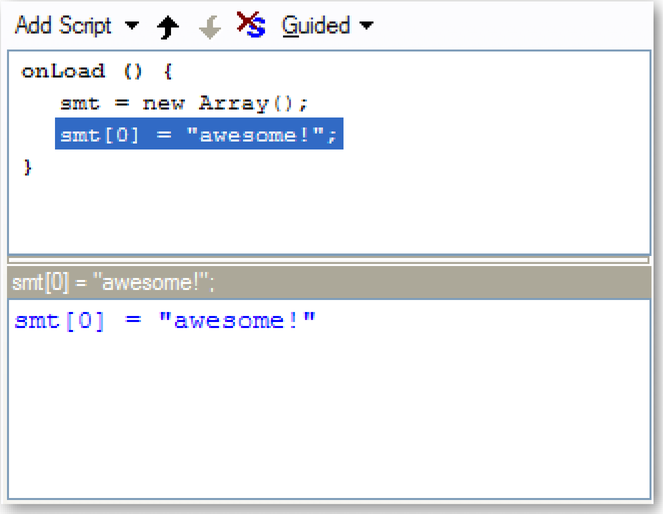
```
onLoad () {
    smt = new Array();
}
```

smt = new Array();

```
smt = new Array()
```

Note: In the script above, a new empty array object was created and assigned to the variable "smt".

6. Press the 'Add Script' button again and select Statements | evaluate; to add our first value to this array. In the evaluate field type what is shown on the image below:



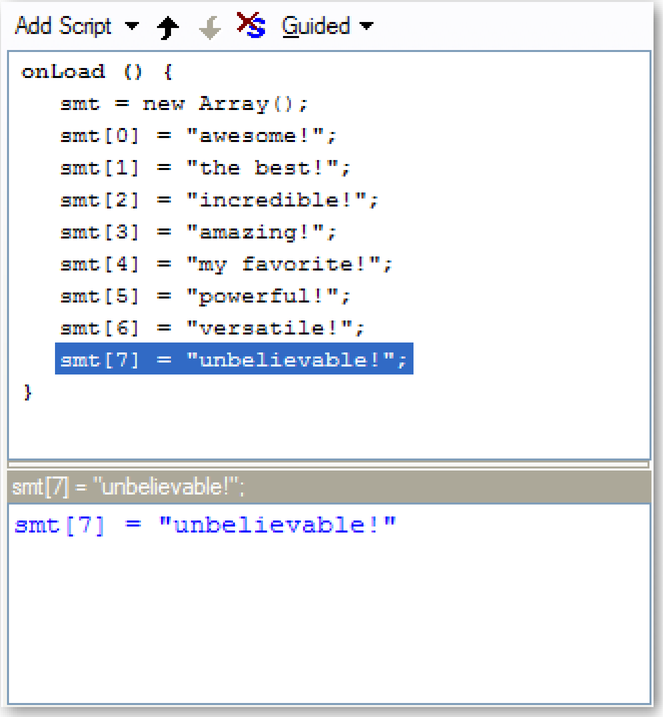
```
onLoad () {
    smt = new Array();
    smt[0] = "awesome!";
}
```

smt[0] = "awesome!";

```
smt[0] = "awesome!"
```

Note: First the variable "smt" is used with brackets after it. The number inside the square brackets indicates the position (or index) within the array (please note, array indices always start from zero as the first element of the array). The message "awesome!" is defined for this position in the array.

7. Repeat Step #6 to add several more values to this array. Make sure that you increase the number inside the brackets by one each time a new position in the array is defined. It may be faster to copy and paste the first value and simply change the values. An example is shown below:

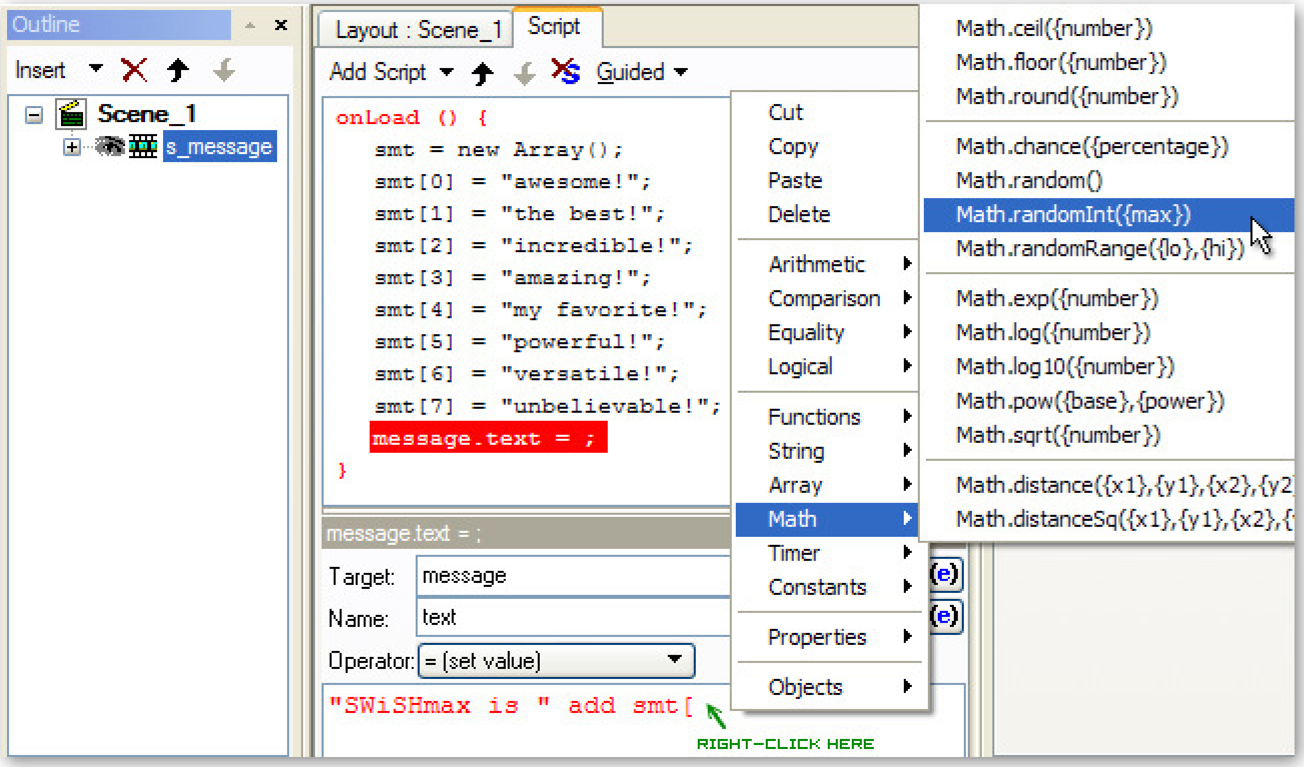


```
onLoad () {
    smt = new Array();
    smt[0] = "awesome!";
    smt[1] = "the best!";
    smt[2] = "incredible!";
    smt[3] = "amazing!";
    smt[4] = "my favorite!";
    smt[5] = "powerful!";
    smt[6] = "versatile!";
    smt[7] = "unbelievable!";
}
```

smt[7] = "unbelievable!";

```
smt[7] = "unbelievable!"
```

8. When you have created several new values for the array, press the 'Add Script' button again and select Statements | name=expr; from the menu. In the 'Target' field use the drop-down menu to select the dynamic text object "message" from the list of targets. In the 'Name' field, use the drop-down menu to select **Text | text**. Leave the Operator as "=" and begin typing what is shown in the image below "SWiSHmax is" etc. When you type the first bracket, Right-Click after it and select **Math | Math.randomInt({max})** from the menu as shown below:



```
onLoad () {
    smt = new Array();
    smt[0] = "awesome!";
    smt[1] = "the best!";
    smt[2] = "incredible!";
    smt[3] = "amazing!";
    smt[4] = "my favorite!";
    smt[5] = "powerful!";
    smt[6] = "versatile!";
    smt[7] = "unbelievable!";
    message.text = ;
}
```

message.text = ;

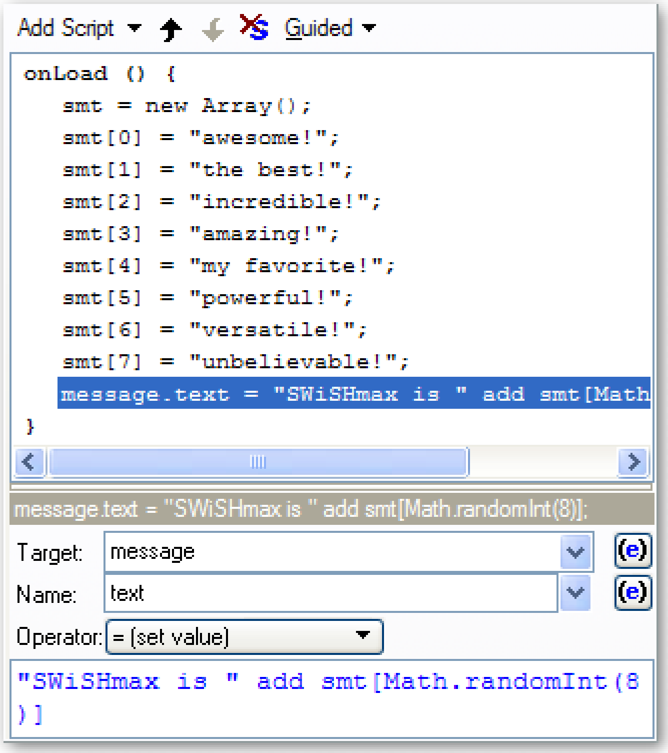
Target: message

Name: text

Operator: = (set value)

```
"SWiSHmax is " add smt[
```

9. When the Math.randomInt() script is added, the cursor will be inside the parentheses. Type the number "8" (which is one number higher than the last number in the array) into the parentheses. Then finish the script by adding the closing bracket "]". The 'Script' panel should appear similar to the image below:



```
onLoad () {
    smt = new Array();
    smt[0] = "awesome!";
    smt[1] = "the best!";
    smt[2] = "incredible!";
    smt[3] = "amazing!";
    smt[4] = "my favorite!";
    smt[5] = "powerful!";
    smt[6] = "versatile!";
    smt[7] = "unbelievable!";
    message.text = "SWiSHmax is " add smt[Math
}
```

message.text = "SWiSHmax is " add smt[Math.randomInt(8)];

Target: message

Name: text

Operator: = (set value)

```
"SWiSHmax is " add smt[Math.randomInt(8
)]
```

10. Go back to the 'Layout' panel and press the 'Play' button in the Control Toolbar to preview the movie. Press the 'Play' button several times to see a random message displayed.

Analysis:
The sprite "s_message" contains on onLoad() event which is triggered when the sprite is first loaded. Inside the onLoad() event a new Array object was defined. Array indices begin with 0 - zero - as the first position in the array and a value is defined for each position. Using the Math.randomInt() code, the movie will select a number between 0 and the number inserted into parentheses (note, the number in parentheses will not be one of the random numbers which is why we used a value one greater than the last position in the array).

The script that contains "smt[Math.randomInt()]" will created a random number for the array and the number that is picked will correspond with one of the positions in the array.

Object Properties

Top Previous Next

Description
This is a step-by-step tutorial demonstrating how to manipulate the properties of Text Objects.

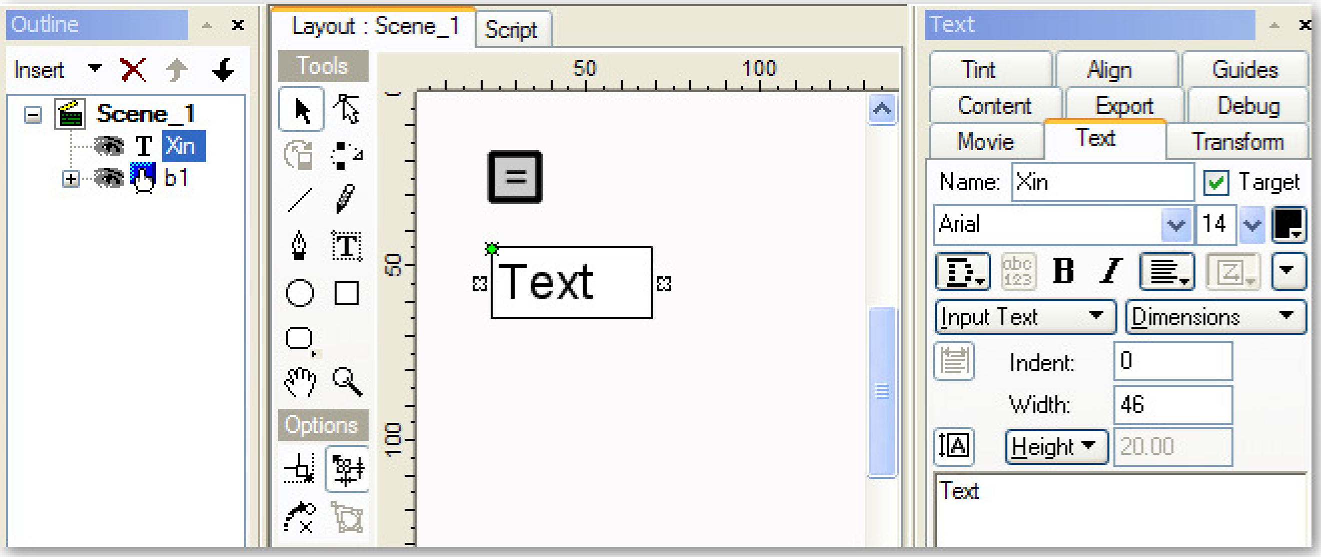
Aim
This tutorial uses multiple Text Objects with Input and Dynamic text properties and introduces the Object properties:
_X, _Y
_xscale, _yscale
_width, _height
_alpha
_rotation
_name

_visible.

It also uses the on(press) Button Event developed in the previous tutorial.

.swi file
"properties.swi"

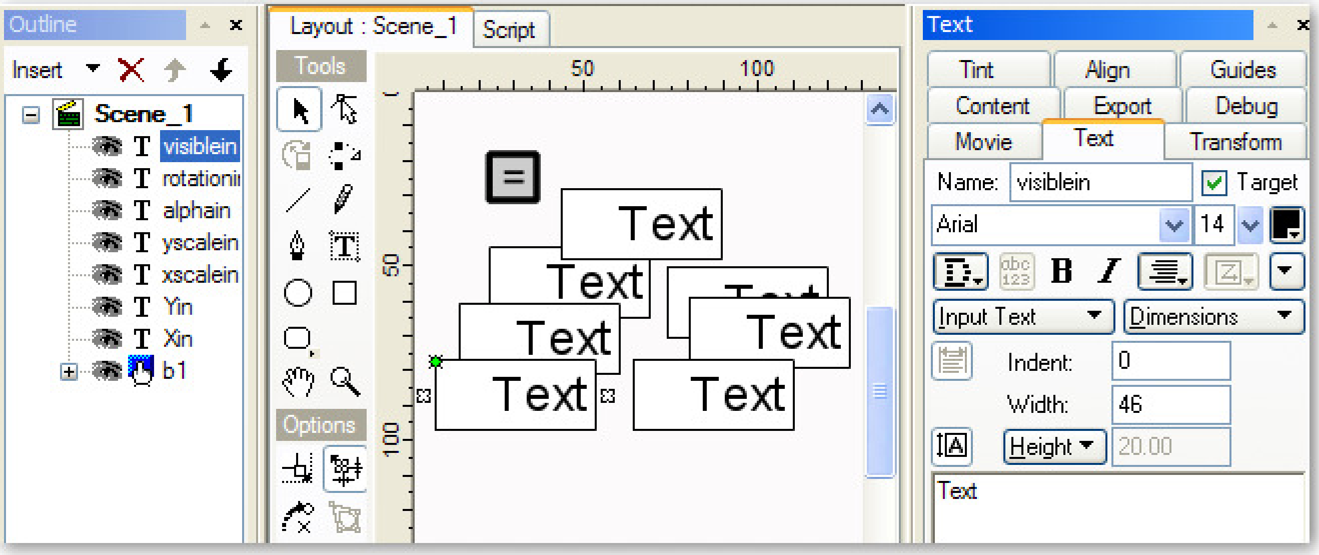
1. Load file "button1.swi". Create an Input Text Object called Xin. (See previous tutorial for more detail). Save the file as "myproperties.swi".



Name: Xin ☑ Target
Arial 14
Input Text
Dimensions
Indent: 0
Width: 46
Height 20.00
Text

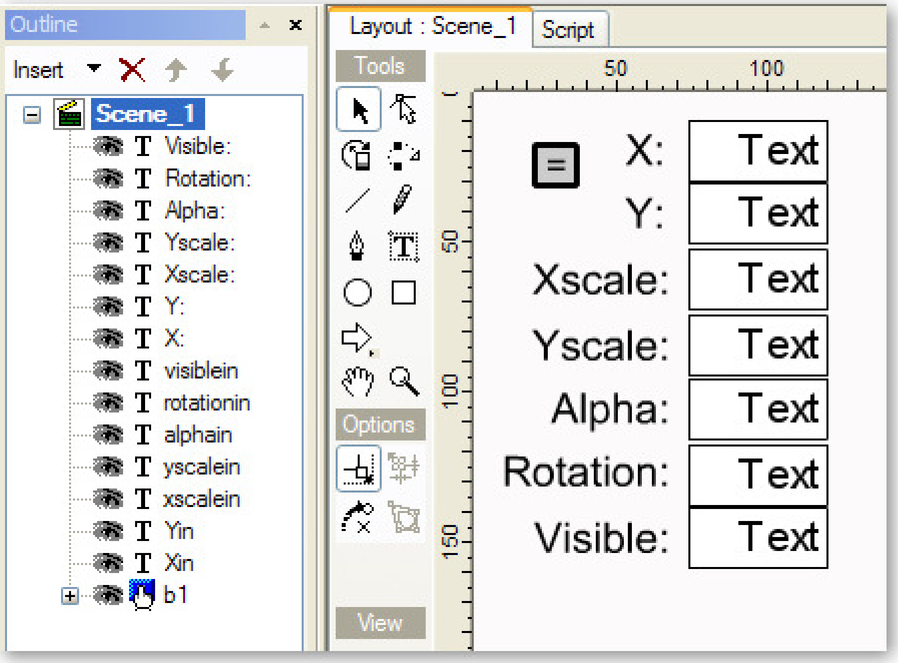
Select the right-justified text option.

2. Create similar Text Objects called Yin, xscalein, yscalein, alphain, rotationin, visiblein. You can create the new Objects using the Text Tool, or by copying and pasting the Xin Text Object and renaming it

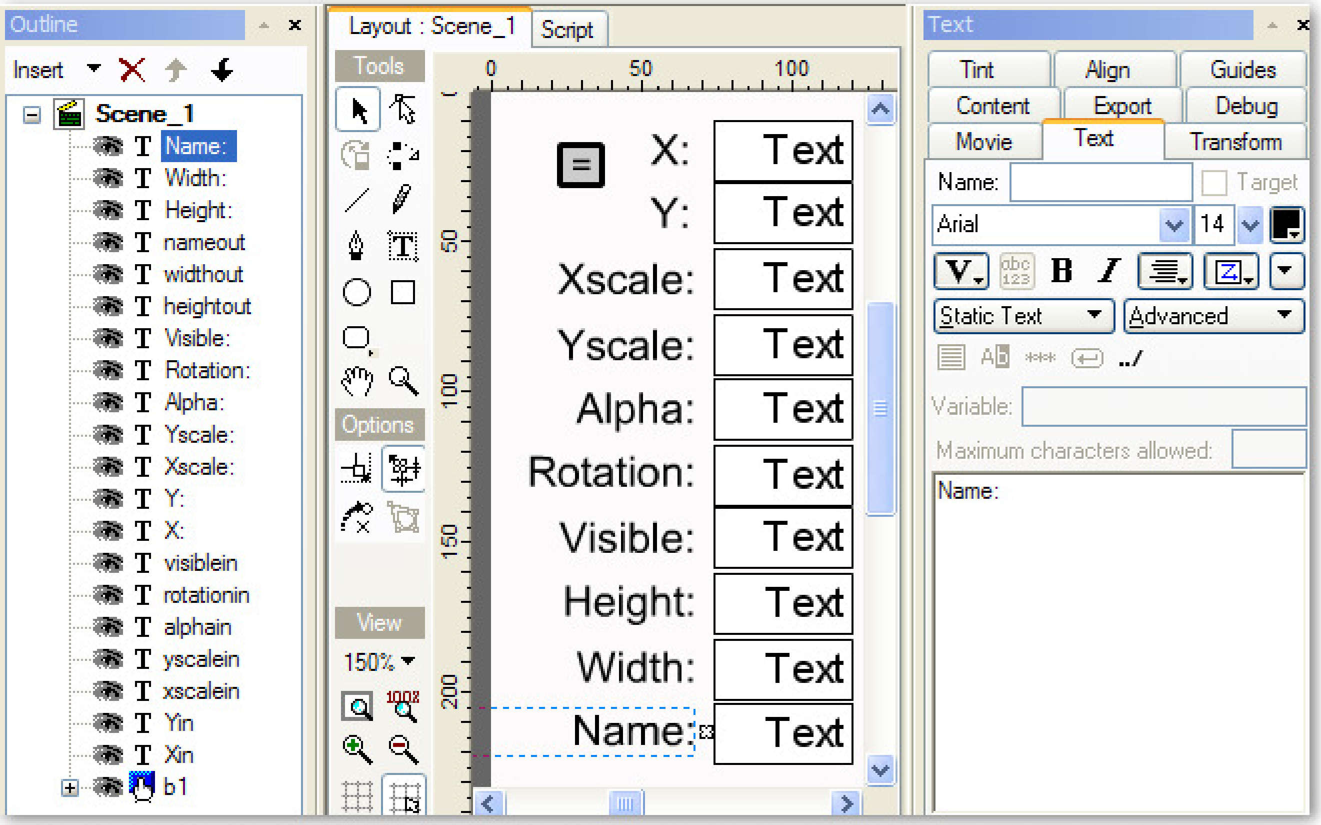


Name: visiblein ☑ Target
Arial 14
Input Text
Dimensions
Indent: 0
Width: 46
Height 20.00
Text

3. Add static text titles for the input boxes and re-arrange as shown below. You may find this easier if you enable Snap To Grid



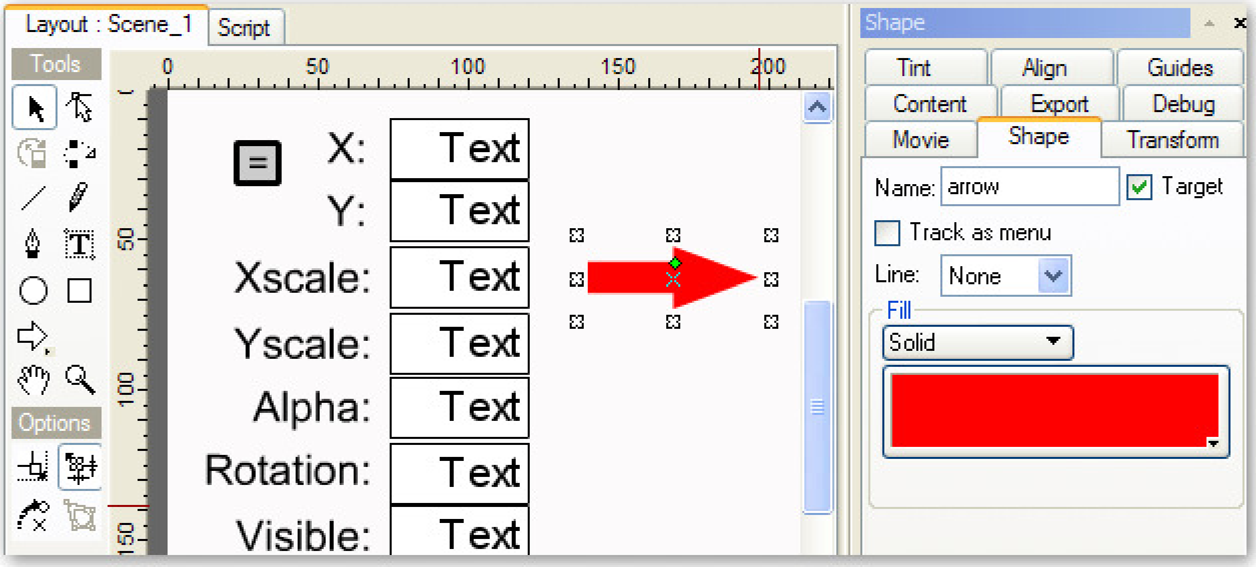
4. Add Dynamic Text Objects heightout, widthout and nameout. Add corresponding static text titles for the new fields.



5. Use the Autoshape Tool to draw an arrow ⇨ as shown below. You may have to position the mouse over the ▢ Autoshape icon in the Toolbar and hold the left mouse button to see the selection Menu that allows

selection of the arrow shape . Name the object "arrow" and tick the 'Target' checkbox.

Note: After naming the Object, you will have to move away from the 'Name:' field to allow the 'Target' checkbox to appear. The Object MUST be named and have its 'Target' checkbox ticked for it to become a Scripting Object



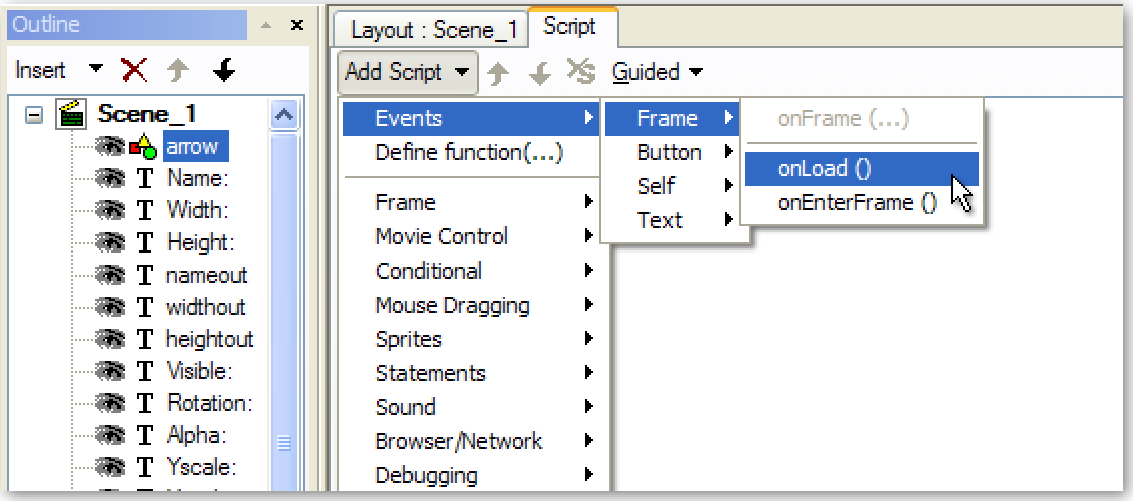
Name: arrow ☑ Target

□ Track as menu

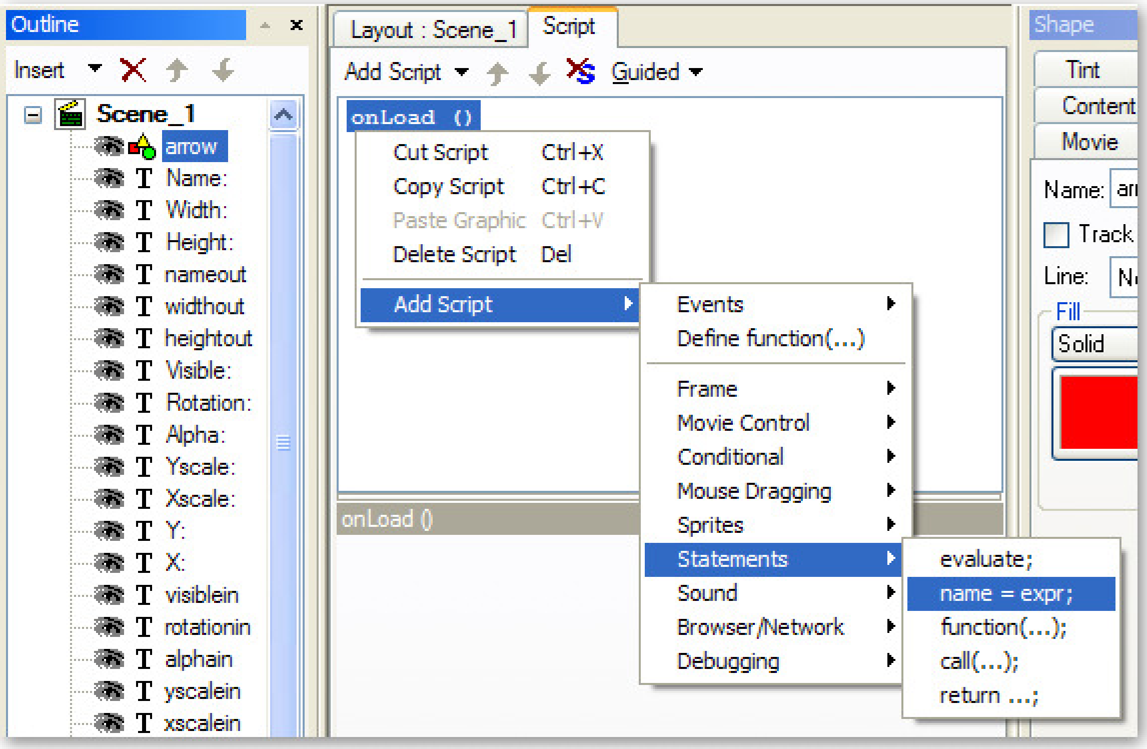
Line: None

Fill: Solid

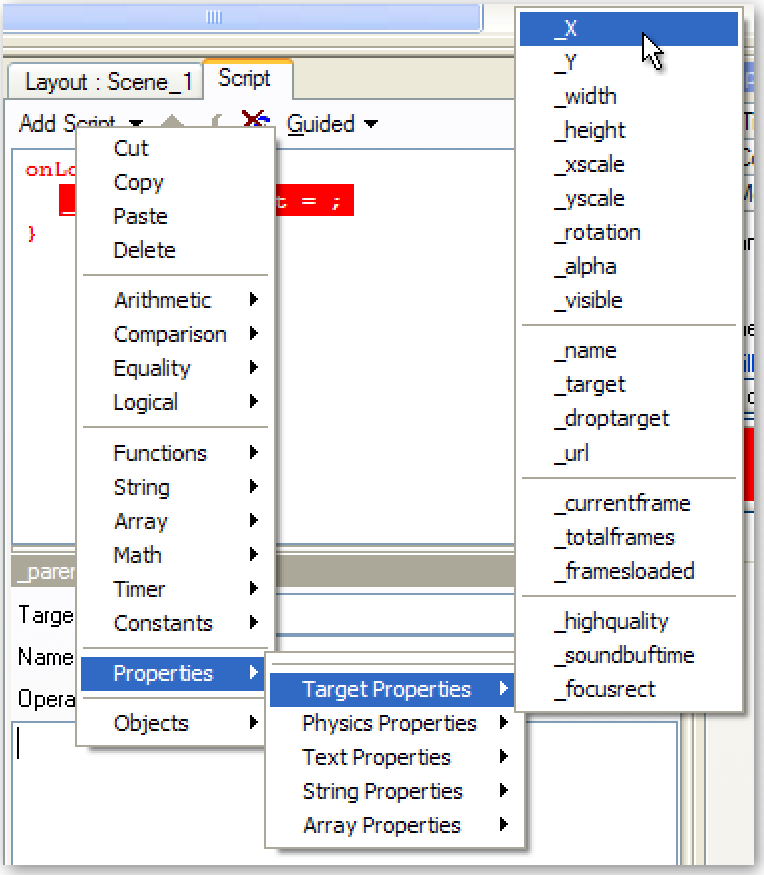
6. Select the arrow shape in the 'Outline' Panel and select the Script tab to add an onLoad() Event Handling function for the arrow. This function will be used to initialise the Input and Dynamic text values



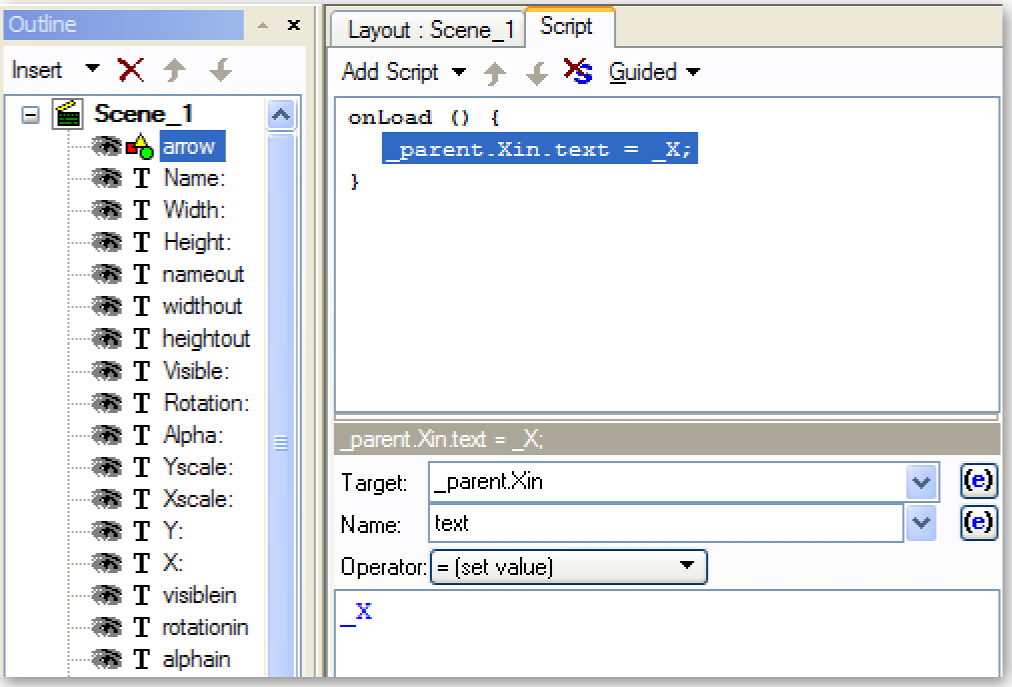
7. Add a **name=expr;** type statement to the **onLoad()** Event function.



For the 'Target:' parameter, select _parent.Xin from the pull-down list.
For the 'Name:' parameter, select Text | text from the pull-down list.
Leave the operator as =.
Enter the formula _X. This formula or value can also be entered by right clicking in the formula field then selecting the properties Menu item.



The resulting function should look like this:



```
onLoad () {
    _parent.Xin.text = _X;
}
```

_parent.Xin.text = _X;

Target: _parent.Xin

Name: text

Operator: = (set value)

_X

8. In a similar way, expand the onLoad function so that it contains the following statements:

```
_parent.Xin.text = _X;
_parent.Yin.text = _Y;
_parent.xscalein.text = _xscale;
_parent.yscalein.text = _yscale;
_parent.alphain.text = _alpha;
_parent.rotationin.text = _rotation;
_parent.visiblein.text = _visible;
_parent.heightout.text = _height;
_parent.widthout.text = _width;
_parent.nameout.text = _name;
```

Note:
You may find it easier to enter one of the statements and then use Copy and Paste to expand that to multiple statements. After entering the multiple statements, edit each individual statement to contain the required data

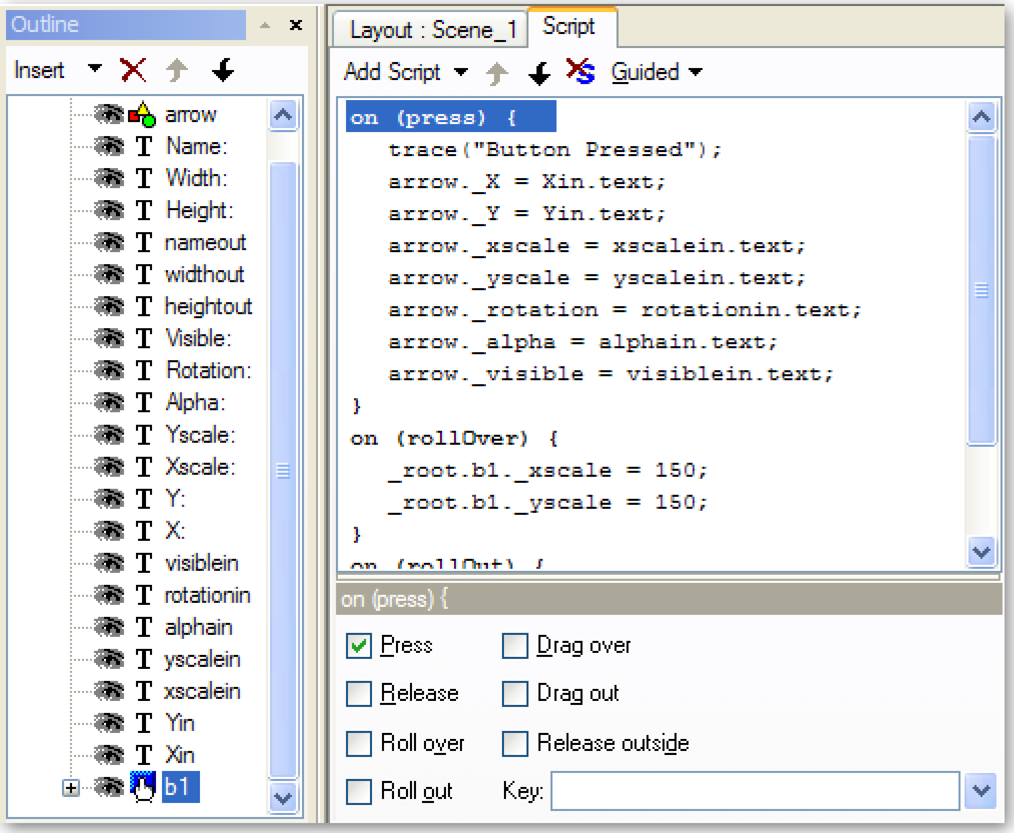
When the function is executed, the arrow properties will be assigned to the text property of the Input and Dynamic Text Objects

10. Add the following statements to the on (press) Event Handling function of the 'b1' button:

```
arrow._X = Xin.text;
arrow._Y = Yin.text;
arrow._xscale = xscalein.text;
arrow._yscale = yscalein.text;
arrow._rotation = rotationin.text;
arrow._alpha = alphain.text;
arrow._visible = visiblein.text;
```

Do this by selecting the 'b1' button in the 'Outline' Panel.

The on (press) function should appear as shown:



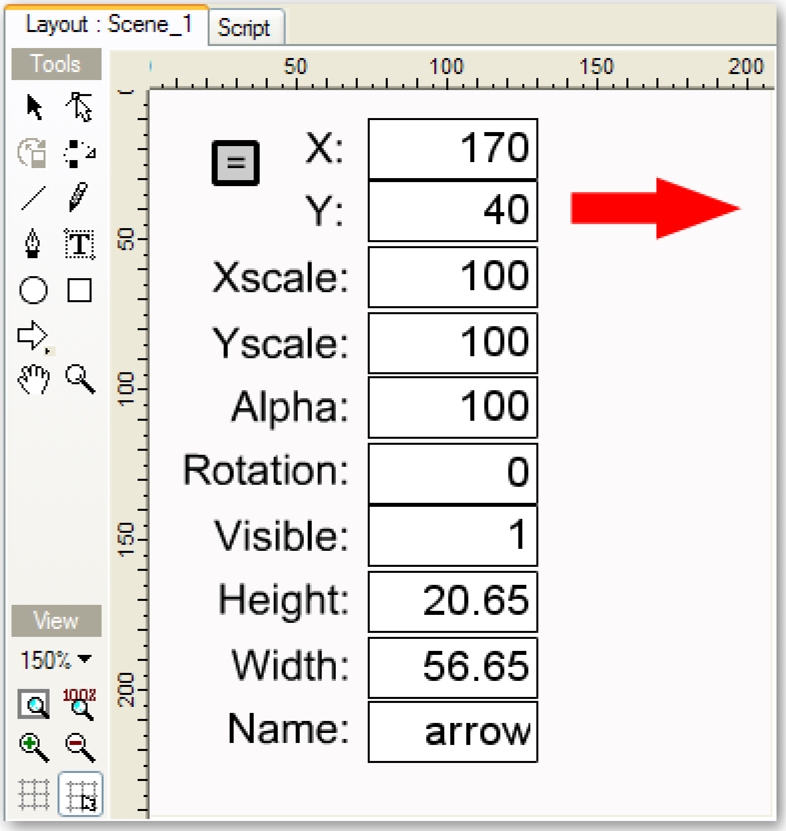
```
on (press) {
    trace("Button Pressed");
    arrow._X = Xin.text;
    arrow._Y = Yin.text;
    arrow._xscale = xscalein.text;
    arrow._yscale = yscalein.text;
    arrow._rotation = rotationin.text;
    arrow._alpha = alphain.text;
    arrow._visible = visiblein.text;
}
on (rollOver) {
    _root.b1._xscale = 150;
    _root.b1._yscale = 150;
}
on (rollOut) {
```

on (press) {

☑ Press ☐ Drag over
☐ Release ☐ Drag out
☐ Roll over ☐ Release outside
☐ Roll out Key:

When this function is executed, the arrow properties will be updated with new values obtained from the text property of the various input Text Objects

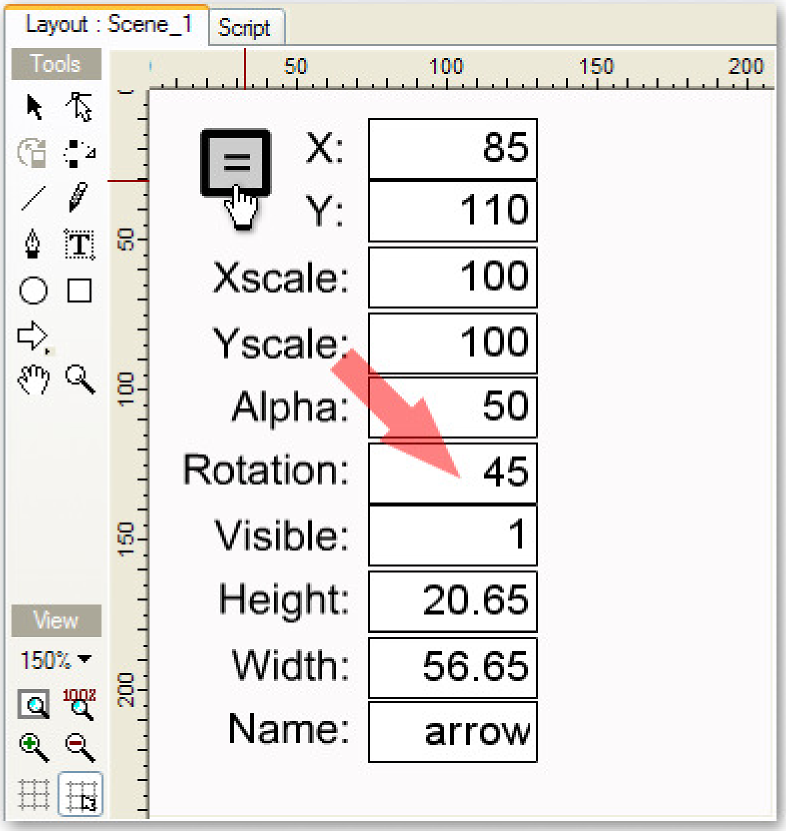
11. Select the 'Layout' tab and press the 'Play' button. The initial properties of the arrow Object should be displayed.



X: 170
Y: 40
Xscale: 100
Yscale: 100
Alpha: 100
Rotation: 0
Visible: 1
Height: 20.65
Width: 56.65
Name: arrow

Note: The values you achieve for X, Y, height and width may be different depending on where and how large you drew the arrow

It is possible to alter the X, Y, xscale, yscale, rotation and visible settings. If you modify those settings while the movie is playing then press the "=" button, the arrow with the revised settings will be displayed.



X: 85
Y: 110
Xscale: 100
Yscale: 100
Alpha: 50
Rotation: 45
Visible: 1
Height: 20.65
Width: 56.65
Name: arrow

Note: If you modify the "visible" property to "0" (zero) it will hide the Arrow.

Analysis
This example demonstrates the various properties that are associated with a Scripting Object.

The arrow onLoad() Event Handling function is used to load the Input and Dynamic Text Objects with the initial properties of the arrow Object. The arrow properties _X, _Y, _xscale, _yscale, _alpha, _rotation and _visible may be altered by modifying the appropriate Input Text Object and then clicking on the button. The button on (press) Event Handling function is used to copy the revised property settings from the Input Text Objects to the arrow Object.

Note: Altering the _xscale and _yscale properties will cause the _height and _width properties to change. The tutorial example will not display this change as the _height and _width properties are only read when the arrow is initially loaded

Mouse Dragging

Top Previous Next

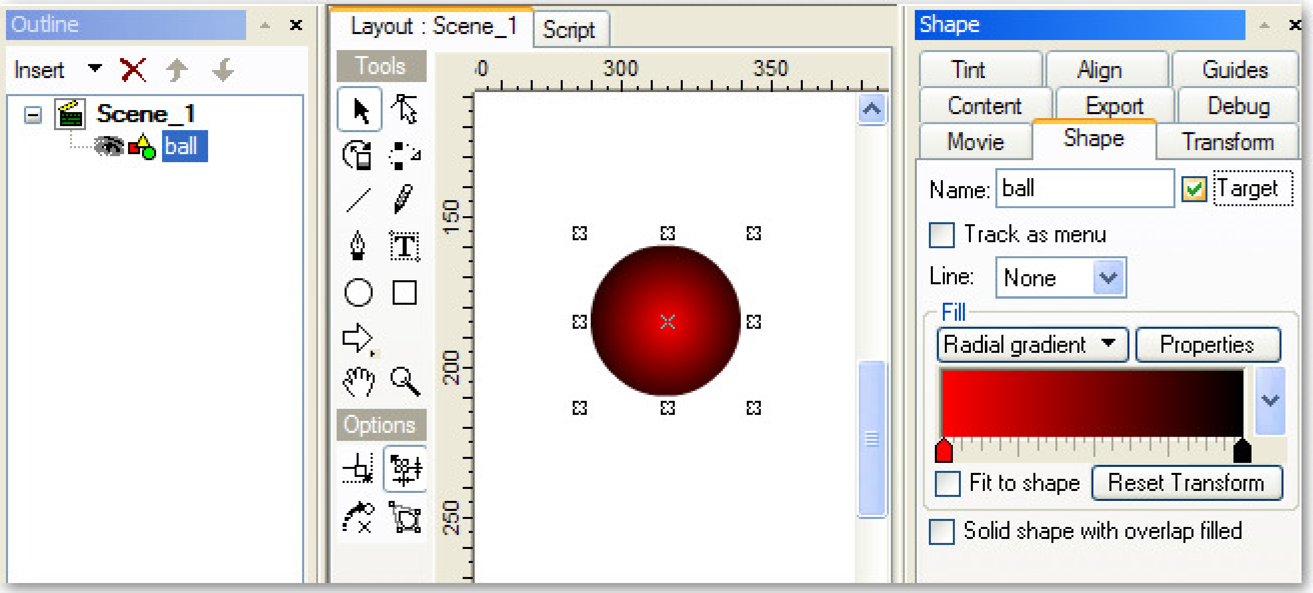
Description
This is a step-by-step tutorial demonstrating how to make draggable Objects.

Aim
This tutorial introduces startDragUnlocked(), startDragLocked() and stopDrag() methods and onSelfEvent(press) and onSelfEvent(release) Events to start and stop dragging.

.swi file
"dragging.swi"

1. Start with a new project (File | New) and save the file as "mydragging.swi"

2. Draw a small circle and name it "ball" on the 'Shape' panel then tick the 'Target' checkbox so that it can be used as a Scripting Object. This circle will be used as the Object that is dragged



Name: ball ☑ Target

□ Track as menu

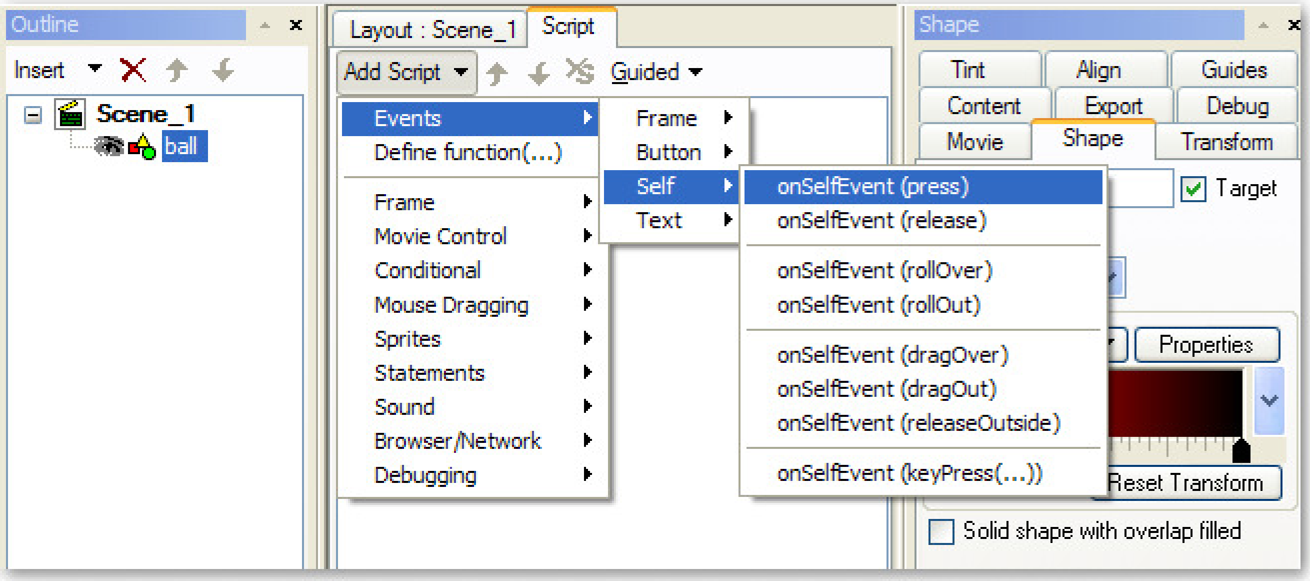
Line: None

Fill: Radial gradient

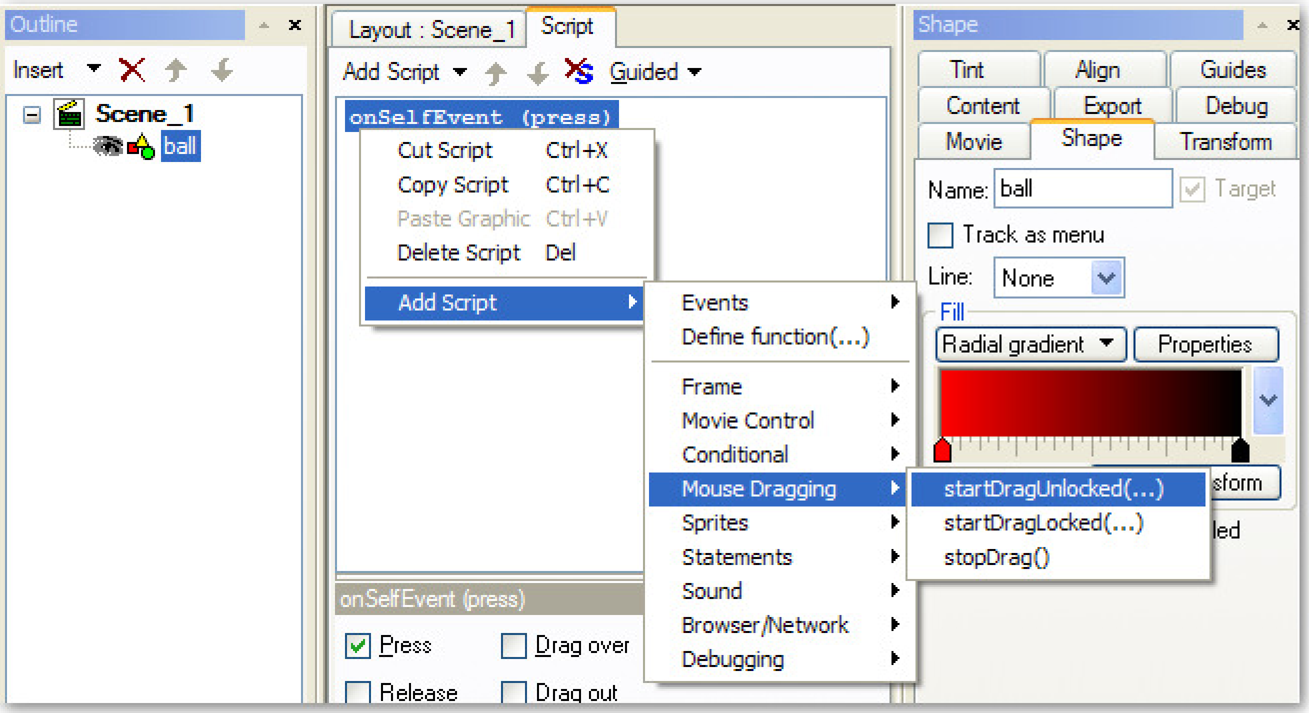
□ Fit to shape

□ Solid shape with overlap filled

3. Select the Script tab and add an onSelfEvent(press) event for the ball.

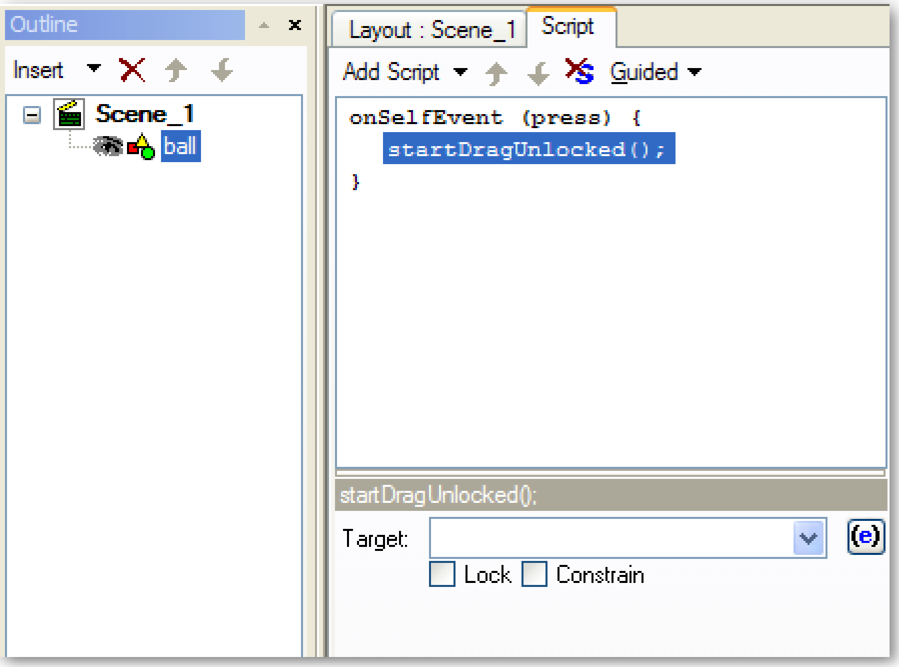


4. Right-click on the function name to add a Mouse Dragging | startDragUnlocked() command to the onSelfEvent(press) function.



```
onSelfEvent (press)
```

The resulting script should be as shown below:



```
onSelfEvent (press) {
    startDragUnlocked();
}
```

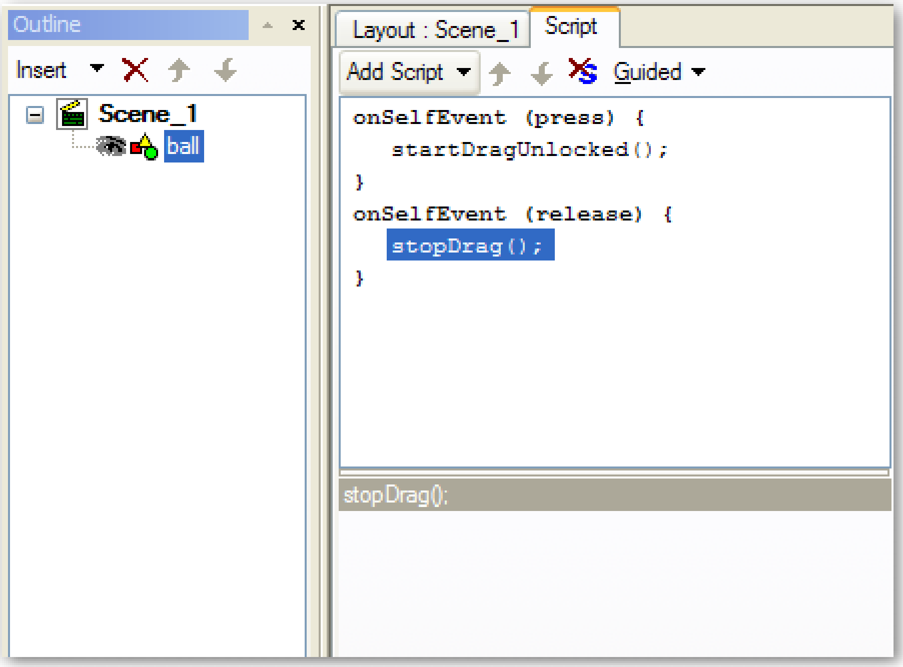
startDragUnlocked();

Target:

□ Lock □ Constrain

Leave the 'Target:' parameter blank. This will result in the current Object (ball) being the item intended to be dragged

5. In a similar way, create an onSelfEvent(release) function for the ball with the command Mouse Dragging | stopDrag();

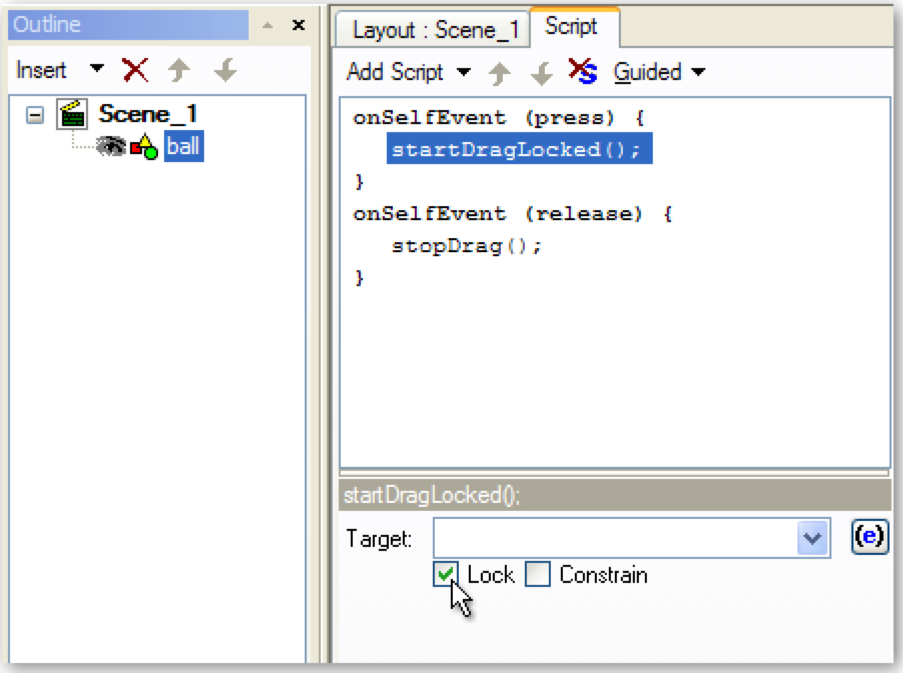


```
onSelfEvent (press) {
   startDragUnlocked();
}
onSelfEvent (release) {
   stopDrag();
}
```

stopDrag();

6. Select the 'Layout' tab and press the 'Play' button. Move the mouse over the ball Object and press the left mouse button. While the button is held down, you should be able to drag the ball around the 'Layout' Panel. When the button is pressed, the mouse position within the Object stays in the location where the mouse was first pressed

7. Select the 'Script' tab and then select the startDragUnlocked() command. Check the 'Lock' checkbox. This will change the command to startDragLocked(). Return to the 'Layout' Panel and observe the new behaviour (press play if it is not already running)



```
onSelfEvent (press) {
   startDragLocked();
}
onSelfEvent (release) {
   stopDrag();
}
```

startDragLocked();

Target:

☑ Lock □ Constrain

Note: When the left mouse button is pressed over the Object, the mouse position within the Object now snaps to the Anchor Point of the "ball" object.

Analysis
This tutorial demonstrates how to make an Object draggable according to mouse input. The onSelfEvent(press) and onSelfEvent(release) Events are used to start and stop dragging of the ball Object. The methods startDragUnlocked(), startDragLocked() and stopDrag() are used to start and stop the dragging. The startDragLocked() method can be used to force dragging from the Anchor Point of the object.

Droptarget / Target

Top Previous Next

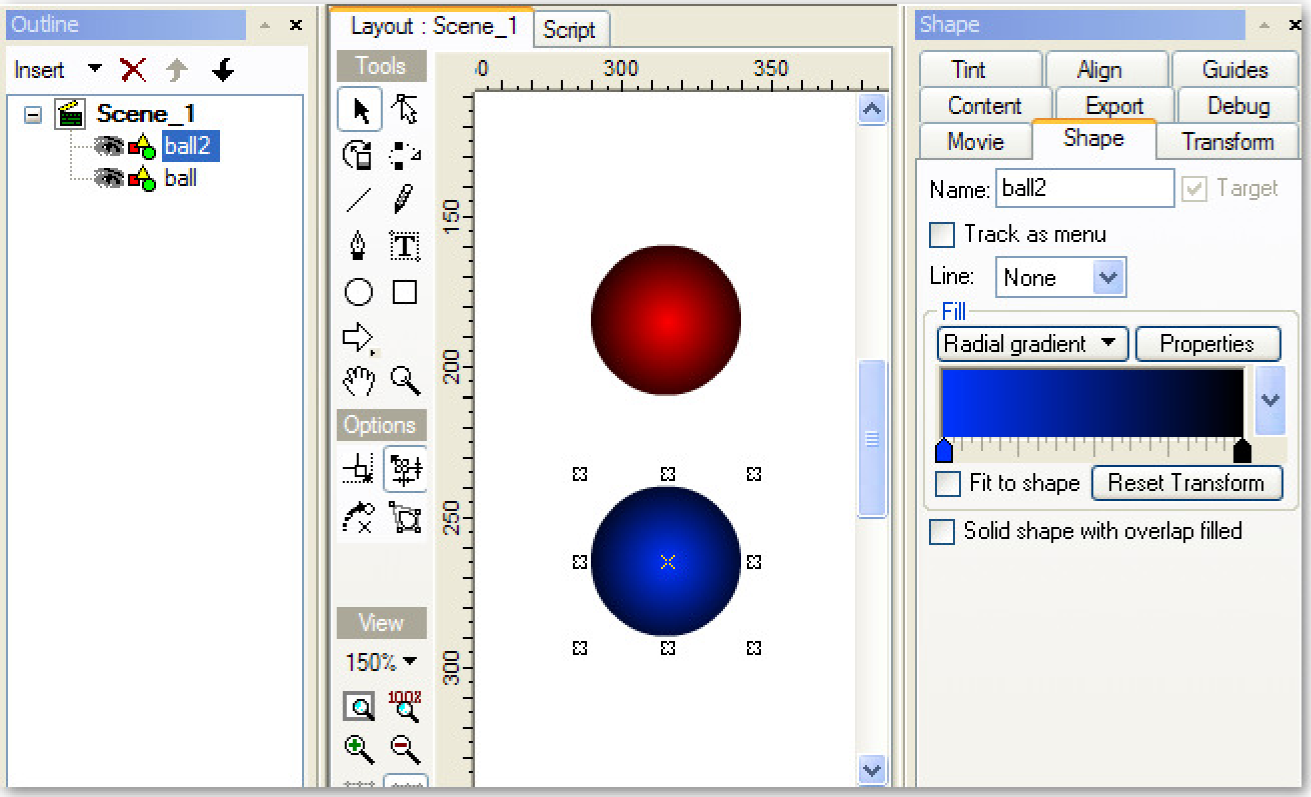
Description
This is a step-by-step tutorial demonstrating the use of the _target and _droptarget properties.

Aim
This tutorial introduces _target and _droptarget properties.

.swi file
"droptarget.swi"

1. Continue from the previous tutorial or load the file "dragging.swi". Save the file as "mydroptarget.swi"

2. Copy and paste the Object "ball". Rename the object "ball2" and change its color to blue



Name: ball2 ☑ Target
☐ Track as menu
Line: None
Fill: Radial gradient
☐ Fit to shape
☐ Solid shape with overlap filled

3. Select the Scene in the 'Outline' Panel and add the following script commands (see the tutorial on Variables and Flow Control for more information):

```
onEnterFrame() {
   if (dt1 != ball._droptarget || dt2 != ball2._droptarget) {
      trace("ball._droptarget: " add ball._droptarget);
```

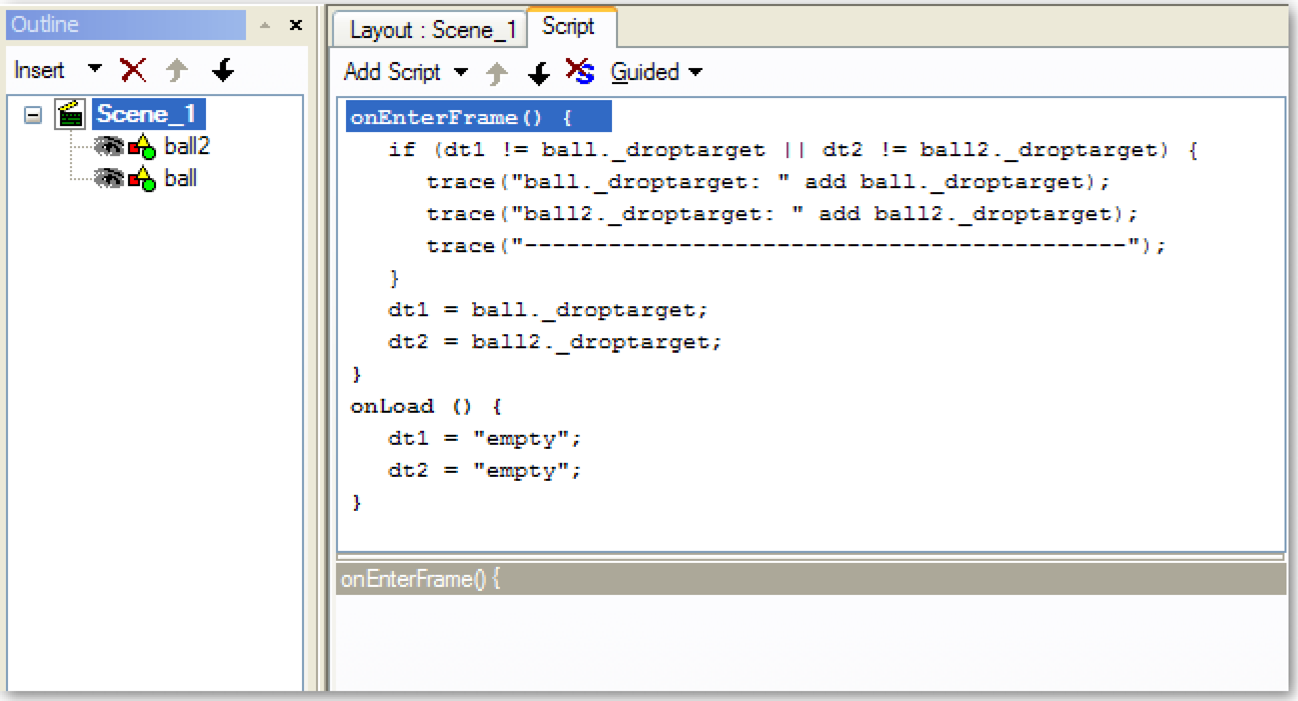
```
        trace("ball2._droptarget: " add ball2._droptarget);
        trace("-------------------------------------------");
    }
    dt1 = ball._droptarget;
    dt2 = ball2._droptarget;
}

onLoad () {
    dt1 = "empty";
    dt2 = "empty";
}
```

The onLoad() Event is used to initialise the variables dt1 and dt2.

The onEnterFrame() Event is called on each Frame. If the value of ball._droptarget or ball2._droptarget has changed from the previous value(s), then the trace statements are executed.

Your 'Script' Panel should look like this:



```
onEnterFrame () {
   if (dt1 != ball._droptarget || dt2 != ball2._droptarget) {
      trace("ball._droptarget: " add ball._droptarget);
      trace("ball2._droptarget: " add ball2._droptarget);
      trace("-------------------------------------------");
   }
   dt1 = ball._droptarget;
   dt2 = ball2._droptarget;
}
onLoad () {
   dt1 = "empty";
   dt2 = "empty";
}
```

4. Select the 'Debug' tab, then return to the 'Layout' Panel and press play. In the 'Debug' Panel you should see:

```
ball._droptarget:
ball2._droptarget:
-------------------------------------------
```

Drag the blue ball over the red one. In the 'Debug' Panel you should now see:

```
ball._droptarget:
ball2._droptarget:
-------------------------------------------
ball._droptarget:
ball2._droptarget: /ball
-------------------------------------------
```

This means that the blue ball (ball2) now sees /ball as its "droptarget". "/ball" is the name of _root.ball expressed in slash notation. Now "ball2._droptarget: /ball" appears the exact instant the mouse pointer crosses the boundary of the red ball.

Drag the blue ball away from the red ball and drag the red ball over the blue ball

Note:

- The red ball appears under the blue ball because of the current layer ordering. This can be altered by selecting the ball in the 'Outline' Panel, then using the Menu items Modify | Order
- When the red ball is under the blue ball the onSelfEvent (release) is not detected, so it is impossible to stop dragging the red ball while its center is obscured by the blue ball

Drag the red ball away from the blue ball then click the left mouse button to deselect it from dragging. In the 'Debug' Panel you should see:

```
ball._droptarget:
ball2._droptarget:
------------------------------------------
ball._droptarget:
ball2._droptarget: /ball
------------------------------------------
ball._droptarget:
ball2._droptarget:
------------------------------------------
ball._droptarget: /ball2
ball2._droptarget:
------------------------------------------
ball._droptarget:
ball2._droptarget:
------------------------------------------
```

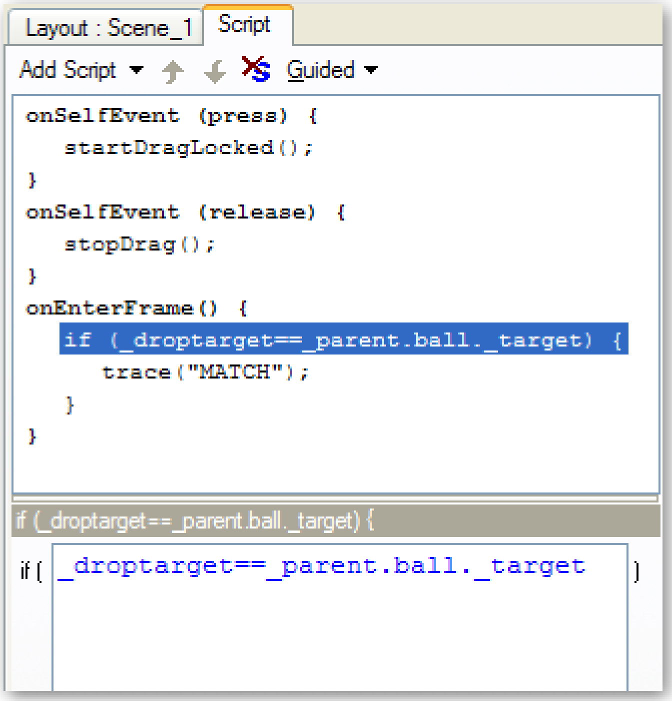
The line "ball._droptarget: /ball2" indicates that the red ball (ball) now sees /ball2 as its "droptarget". "/ball2" is the name of _root.ball2 expressed in slash notation.

Press the 'Stop' button.

5. Add the following script to ball2:

```
onEnterFrame() {
   if (_droptarget==_parent.ball._target) {
      trace("MATCH");
   }
}
```

Your script should now look like:



```
onSelfEvent (press) {
   startDragLocked();
}
onSelfEvent (release) {
   stopDrag();
}
onEnterFrame() {
   if (_droptarget==_parent.ball._target) {
      trace("MATCH");
   }
}
```

```
if (_droptarget==_parent.ball._target) {
```

```
if ( _droptarget==_parent.ball._target )
```

6. Select the 'Debug' tab, then return to the 'Layout' Panel and press play. Drag the blue ball over the red ball and you should see the following in the 'Debug' Panel:

```
ball._droptarget:
ball2._droptarget:
--------------------------------------------
MATCH
ball._droptarget:
ball2._droptarget: /ball
--------------------------------------------
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
MATCH
```

The word MATCH indicates that the comparison "_droptarget==_parent.ball._target" has returned the value true. The _target property is used to return the name of the target Object in slash notation.

This type of comparison can be used to check when an Object has been dragged over another specific Object

Analysis
The _target and _droptarget properties can be used to detect when one Object is dragged over (or under, depending on layering) another specific Object. This would be useful for implementing, say, a 'trash can'.

One difference between this approach and using (for instance) isNearTarget() and isNearThis(), is that the item being dragged can be easily identified as the _droptarget property only changes for the object that is currently being dragged.

The _droptarget property changes when the mouse pointer crosses the boundary of the other Object.

The onLoad() Event is used to initialise some variables that are used later by the onEnterFrame() Event. The onEnterFrame() Event called on each Frame and is used to check current conditions repetitively.

Collision Detection

Top Previous Next

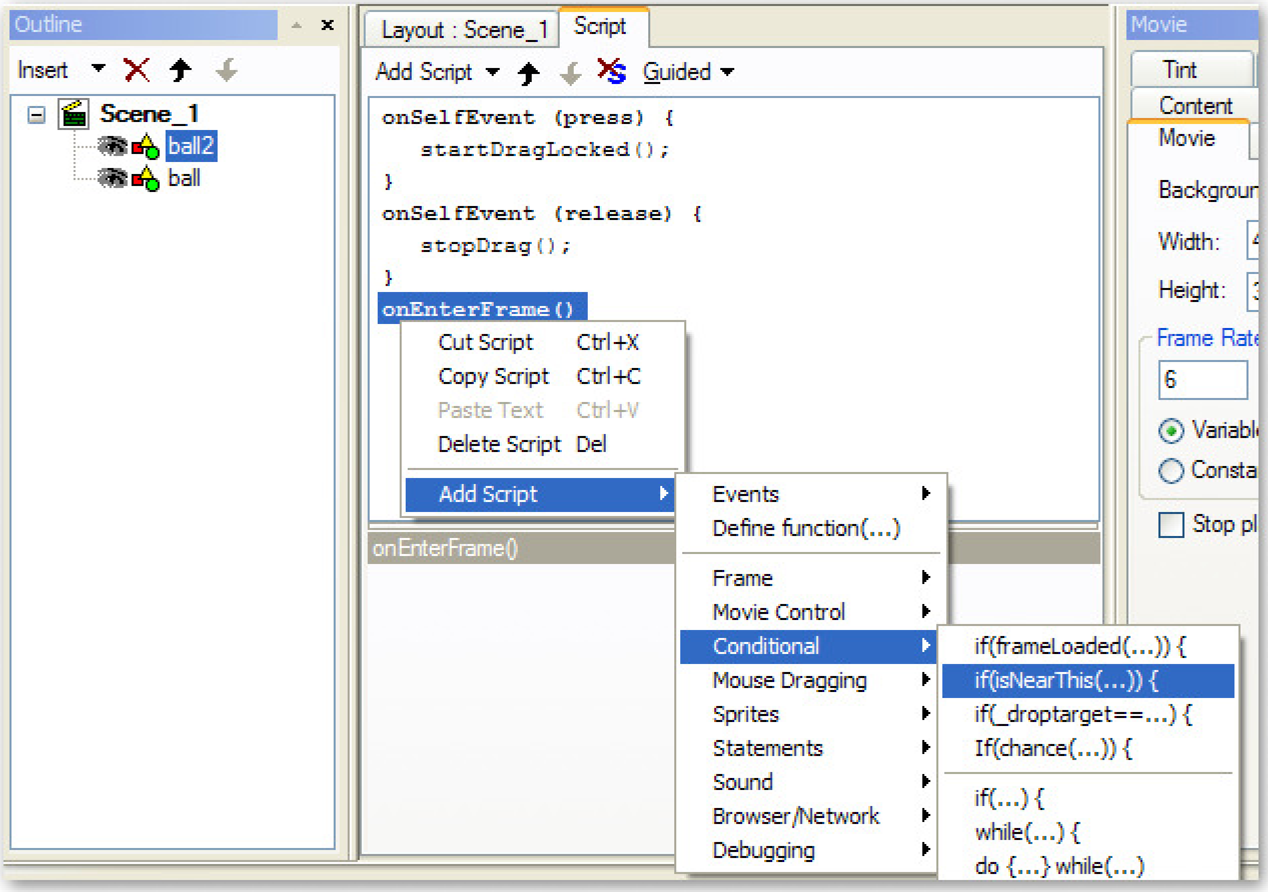
Description
This step-by-step tutorial builds on previous tutorials to demonstrate various methods of collision detection. Two methods isNearTarget() and isNearThis() provide a means of detecting proximity between two Objects.

Aim
This tutorial introduces isNearTarget() and isNearThis() methods.

.swi file
"collide.swi"

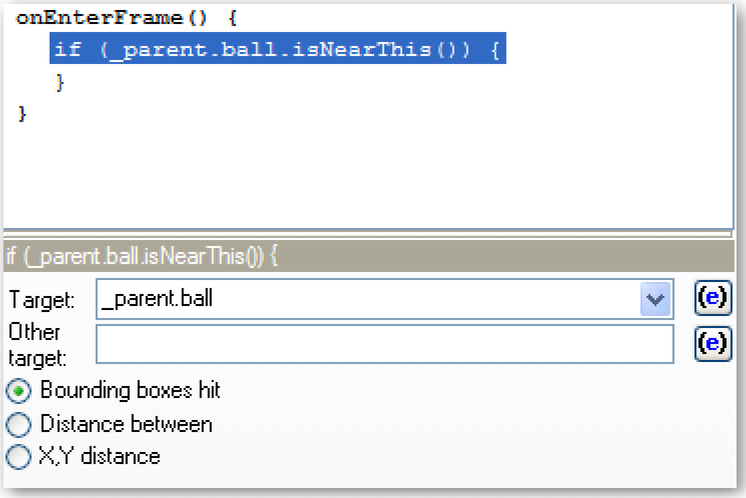
1. Continue from the previous tutorial or open the file "droptarget.swi". Save the file as "mycollide.swi"

2. Select the blue ball (ball2) in the 'Outline' Panel. Delete the existing Actions from the onEnterFrame() Event then select the if (isNearThis()) conditional statement.



```
onSelfEvent (press) {
   startDragLocked();
}
onSelfEvent (release) {
   stopDrag();
}
onEnterFrame()
```

Select _parent.ball from the pull-down list (_parent.ball is the other ball) and select the 'Bounding boxes hit' option



```
onEnterFrame() {
   if (_parent.ball.isNearThis()) {
   }
}
```

if (_parent.ball.isNearThis()) {

Target: _parent.ball

Other target:

● Bounding boxes hit

○ Distance between

○ X,Y distance

Note: As ball and ball2 are both Objects contained within the main Movie, ball can be referenced via either "_root.ball" or "_parent.ball" from inside the ball2 Object

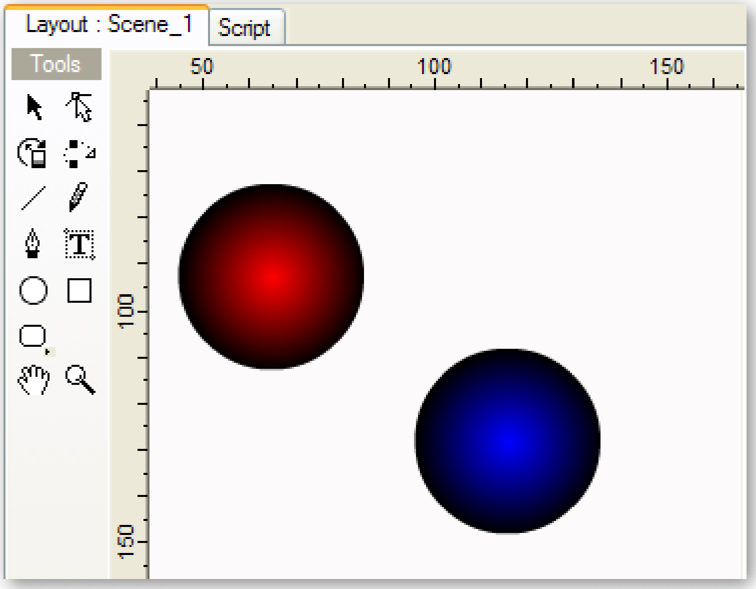
3. Complete the script for the onEnterFrame() event to be:

```
onEnterFrame() {
  if (_parent.ball.isNearThis()) {
    _alpha = 50;
  } else {
    _alpha = 100;
  }
}
```

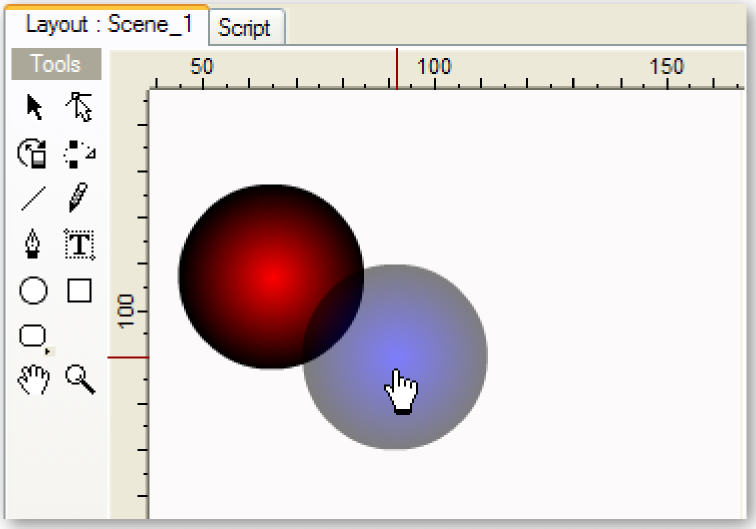
This script will change the _alpha (transparency) of the ball2 Object to 50% whenever it is near the other ball

4. Return to the 'Layout' Panel and press the 'Play' button. Drag the blue ball near the red ball from different directions noting when the blue ball becomes partially transparent.

   In this example, the balls are not considered 'Near'.



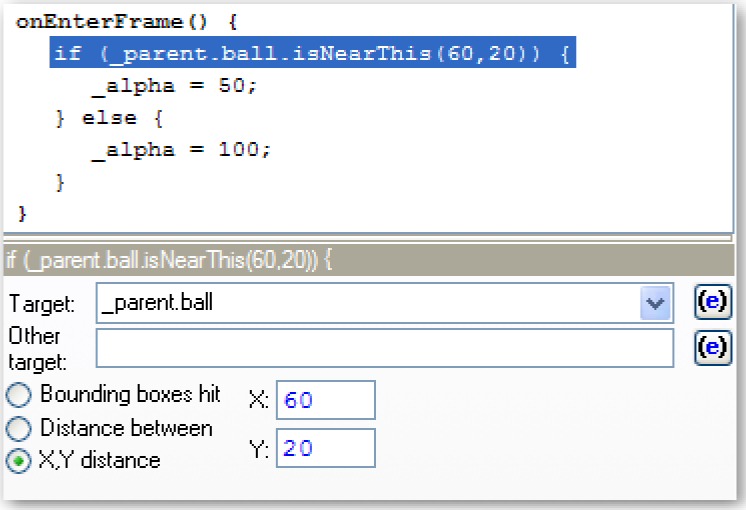
The balls are 'Near' in the example below. This is shown by the blue ball (ball2) becoming partially transparent (_alpha = 50%).



Note:

- The balls are 'Near' even though they are not over lapping. In this case, the definition of 'Near' is defined by the 'Bounding boxes hit' checkbox option. This means that the Objects are considered 'Near' if imaginary rectangles that enclose the Objects intersect
- Note that dragging the red ball towards the blue ball has the same effect (this is different to the previous example)

5. Press the 'Stop' button and return to the 'Script' Panel. Select the "X, Y" option for the if (isNearThis()) statement and enter values 60 for X and 20 for Y.



```
onEnterFrame() {
    if (_parent.ball.isNearThis(60,20)) {
        _alpha = 50;
    } else {
        _alpha = 100;
    }
}
```

if (_parent.ball.isNearThis(60,20)) {

Target: _parent.ball

Other target:

○ Bounding boxes hit

○ Distance between

● X,Y distance

X: 60

Y: 20

The "if (isNearThis())" statement should now look like:

```
if (_parent.ball.isNearThis(60,20)) {
```

Note the shape of the area where the balls are considered 'Near'. Press 'Play' to preview the movie.

6. Press the 'Stop' button and return to the 'Script' Panel. Select the 'Distance between' option for the if (isNearThis()) statement. Enter a distance that is equal to the sum of the radius of each of the balls. In this example:

Ball has a width of 40 (radius = 20)
Ball2 has a width of 40 (radius = 20)
Distance is therefore 20 + 20 = 40.

The "if (isNearThis())" statement should now look like:

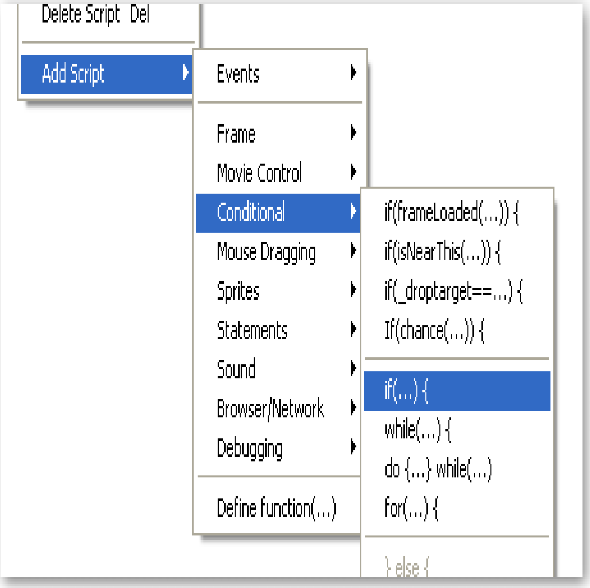
```
if (_parent.ball.isNearThis(40)) {
```

Press the 'Play' button and note that the blue ball changes transparency whenever the balls touch or overlap

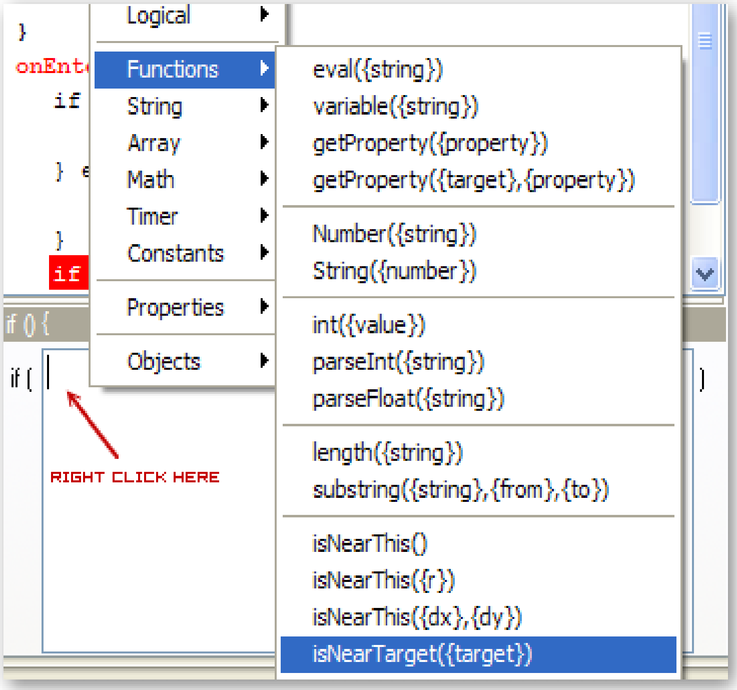
7. Press the 'Stop' button and return to the 'Script' Panel. Select the ball Object in the 'Outline' Panel (red ball) and add the following script:

```
onEnterFrame() {
  if (isNearTarget(_root.ball2._target)) {
    _alpha = 50;
  } else {
    _alpha = 100;
  }
}
```

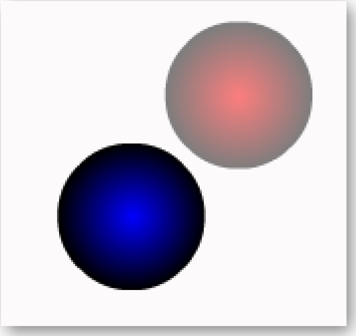
To enter the isNearTarget() function you will need to enter a standard if (...) statement.



You can then enter the function name by typing it or by right clicking and selecting from the pop-up Menu



8. Press the 'Play' button and note that the blue ball changes transparency whenever the balls touch or overlap. Note that the red ball changes transparency when the bounding rectangles of the balls overlap.



The command "if (_root.ball2.isNearThis()) {" would have the same effect

9. Change the command within the if statement to:

```
if (isNearTarget(_root.ball2._target, 60, 20)) {
```

Press the 'Play' button and observe

Note: This behavior is identical to:

```
if (_root.ball2.isNearThis(60,20)) {
```

10. Change the command within the if statement to:

```
if (isNearTarget(_root.ball2._target, 40)) {
```

Press the 'Play' button and observe. Both balls should change transparency when they touch.

Note: This behaves is identical to:

```
if (_root.ball2.isNearThis(40)) {
```

Analysis
isNearTarget() and isNearThis() methods provide identical functionality using a different parameter interface. Both methods allow proximity to be defined via the size of:

- bounding rectangles (isNearTarget(target) or target.isNearThis())
- arbitrarily sized rectangle (isNearTarget(target, X, Y) or target.isNearThis(X, Y))
- the distance between the centers (isNearTarget(target, D) or target.isNearThis(D)).

Unlike the _target and _droptarget properties discussed in the previous tutorial, the 'Near' condition is not affected by which Object is being dragged.

Simple Gaming

Top Previous Next

Description
This is a step-by-step tutorial using the On Frame Event to show how Objects can be animated to provide simple gaming functionality.

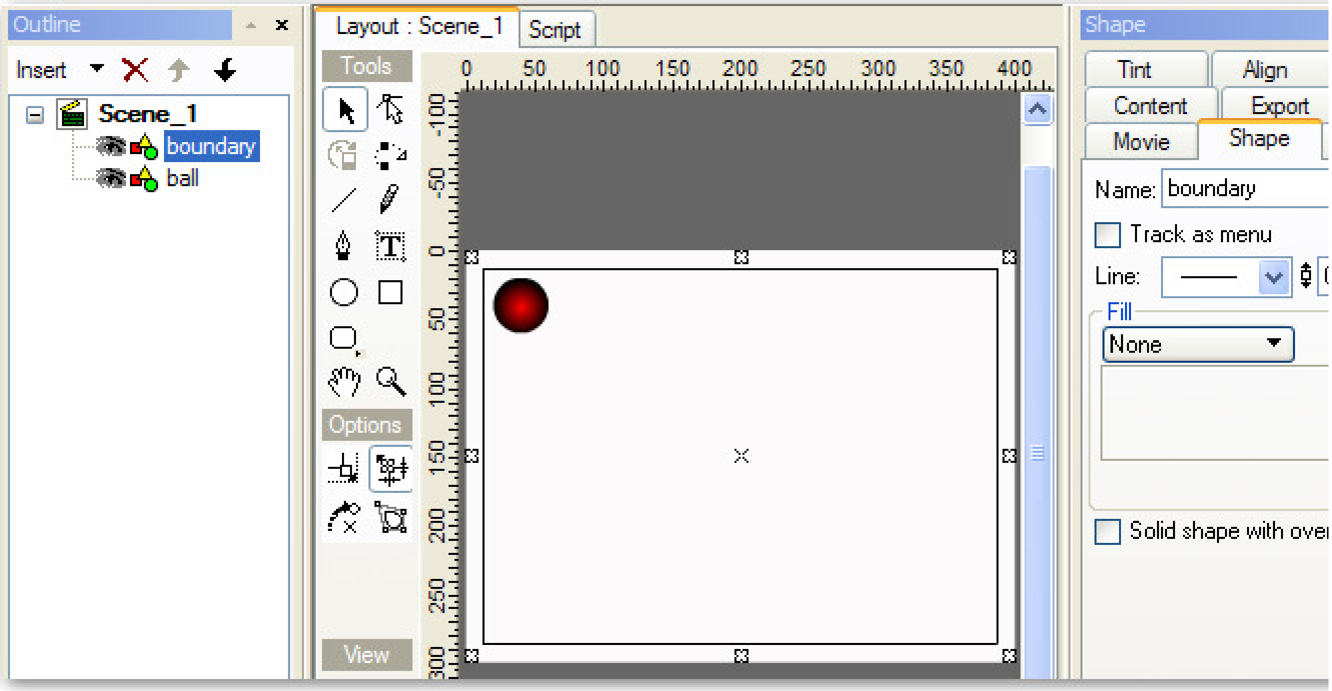
Aim
This tutorial uses the onEnterFrame() Event to animate Objects.

.swi file

"game.swi"

1. Open the "dragging.swi" file used in the Mouse Dragging tutorial and save the file as "mygame.swi"

2. Draw a rectangle to define the playing area. Name the rectangle "boundary" and tick the 'Target' checkbox



Name: boundary

□ Track as menu

Fill: None

□ Solid shape with over

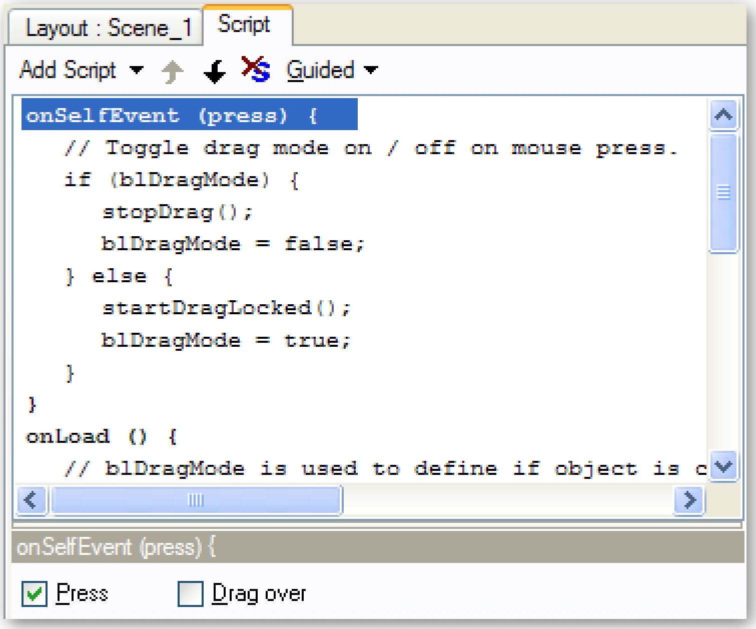
3. Rename the ball Object "bat" and change its Event scripts to be the following:

```
onSelfEvent (press) {
   // Toggle drag mode on / off on mouse press.
   if (blDragMode) {
      stopDrag();
      blDragMode = false;
   } else {
      startDragLocked();
      blDragMode = true;
   }
}
onLoad () {
   // blDragMode is used to define if object is currently being dragged.
   blDragMode = false;
   //
   // x0, y0 are used to store the position of the object in the previous frame.
   x0 = _X;
   y0 = _Y;
   //
   // dx, dy are the x and y velocity of the object.
   // ie. dx is the number of pixels in the x direction that the object moved since the previous frame.
   dx = 0;
   dy = 0;
}
onEnterFrame() {
   // Calculate and save the distance moved since the last frame. This is the velocity.
   dx = _X - x0;
   dy = _Y - y0;
   // Save the current position for calculation in the next frame
   x0 = _X;
   y0 = _Y;
}
```

As the scripting is becoming more complex, it is useful to add comments using Add Script | Debugging | Comments. Comments are shown as a line starting with // or surrounded by /* */.

The **onLoad()** function is used to declare variables and setup initial values.
The **onSelfEvent(press)** is used to toggle the dragging of the bat Object. Press the left mouse button once on the 'bat' object to start dragging. Press again to stop dragging. Note that the original **onSelfEvent (release)** has been deleted.
The **onEnterFrame()** function is called at the start of each Frame. This function is used to calculate and store the instantaneous speed of the bat. The speed is calculated by saving the number of pixels that the Object has moved since the function was previously called. The current position is then saved in x0, y0 for use when the function is called again on the next Frame.

Note that in the image below, only a subsection of the script is shown.

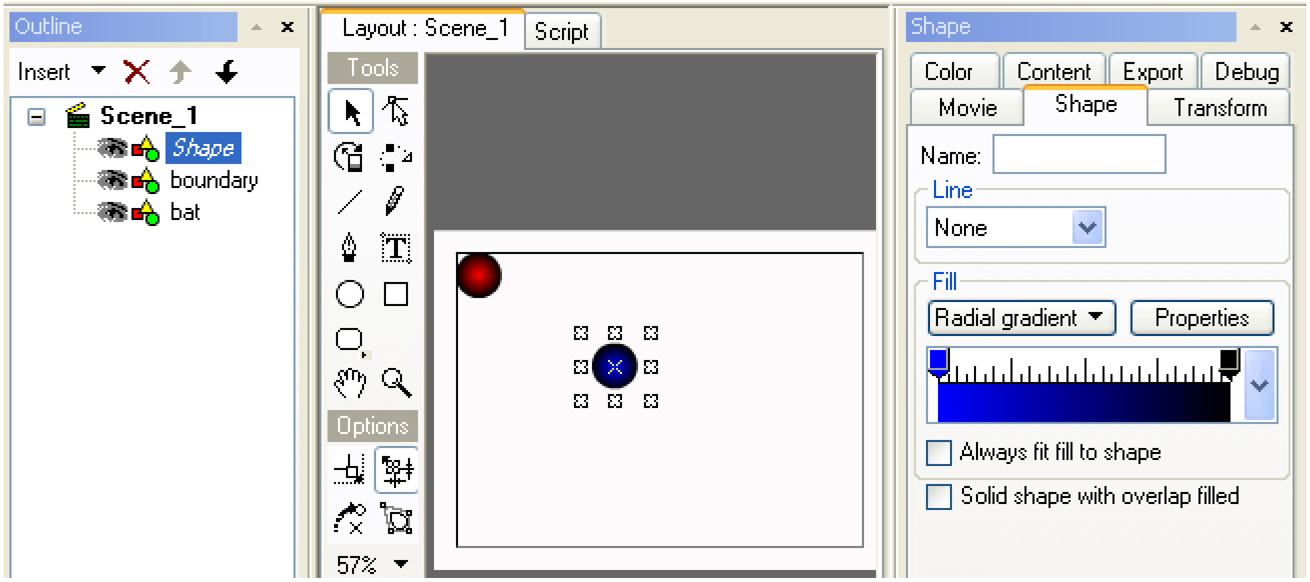


```
onSelfEvent (press) {
   // Toggle drag mode on / off on mouse press.
   if (blDragMode) {
      stopDrag();
      blDragMode = false;
   } else {
      startDragLocked();
      blDragMode = true;
   }
}
onLoad () {
   // blDragMode is used to define if object is c
```

onSelfEvent (press) {

☑ Press ☐ Drag over

4. Use the Circle Tool to create a new circle. Colour the circle blue. The diameter of this circle was 40



Name:

Line: None

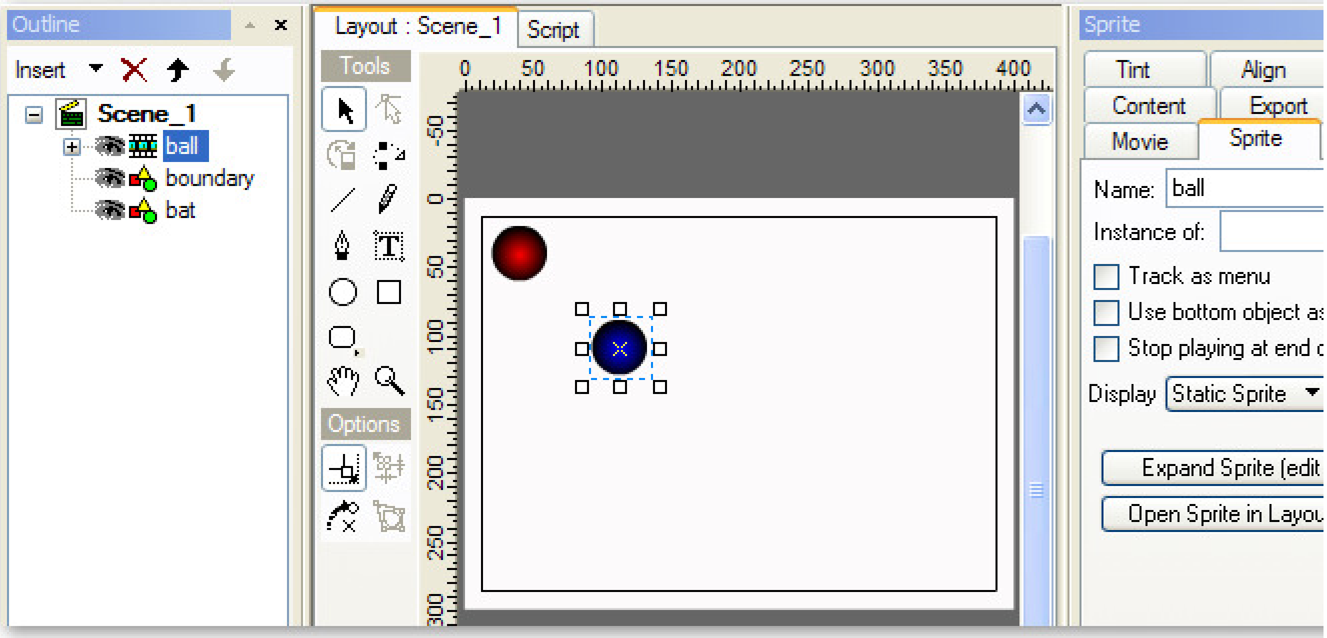
Fill: Radial gradient

□ Always fit fill to shape

□ Solid shape with overlap filled

Note: If you create the circle by copying the bat Object, be sure to remove any script associated with the original Object

5. Use the Menu function Modify | Grouping | Group as Sprite to change the unnamed shape into a Sprite. Name the sprite "ball". Using a Sprite for the ball allows multiple balls to be added later using Instances



Name: ball

Instance of:

□ Track as menu

□ Use bottom object as

□ Stop playing at end c

Display Static Sprite

6. Add the following script to the 'ball' Sprite. The script below contains some hyperlinks to assist with understanding.

```
onLoad () {
  // Define variables and initial conditions.
  // dx is the pixels the object moves per frame in the x direction.
  // dy is the pixels the object moves per frame in the y direction.
  //
  // MinX, MaxX, MinY and MaxY are the minimum and maximum allowable positions for the object.
  // These are calculated based on the size of the boundary object and the radius of this shape.
  dx = 0;
  dy = 0;
  MinX = _root.boundary._X - _root.boundary._width / 2 + this._width / 2;
  MaxX = _root.boundary._X + _root.boundary._width / 2 - this._width / 2;
  MinY = _root.boundary._Y - _root.boundary._height / 2 + this._width / 2;
  MaxY = _root.boundary._Y + _root.boundary._height / 2 - this._width / 2;
  // batradius is the combined radius of the bat and the ball
  batradius = this._width / 2 + _root.bat._width / 2;
}
onEnterFrame() {
  // Move ball to new position
  _X += dx;
  _Y += dy;
  //
  // Check for boundary collision. If collision detected, reverse velocity
  // Use of Math.abs is more robust than simply -dx, -dy.
  // This will work correctly if dx or dy > R but < 2 * R and ball straddles boundary.
  if (_X < MinX) {
    // Force dx to be positive.
    dx = Math.abs(dx);
  }
  if (_X > MaxX) {
    // Force dx to be negative.
    dx = -Math.abs(dx);
  }
  if (_Y < MinY) {
    // Force dy to be positive.
    dy = Math.abs(dy);
  }
  if (_Y > MaxY) {
    // Force dy to be negative.
    dy = -Math.abs(dy);
  }
  //
  // Check for collision with bat.
  if (_parent.bat.isNearThis(batradius)) {
    // We have collided with bat, change ball velocity.
    dx = CalcNewDx();
    dy = CalcNewDy();
  }
}
function CalcNewDx() {
  // Function to calculate the new x velocity of the ball
  // This is not correct physics, (no bounce). Just take on the velocity of the bat
  return _root.bat.dx;
}
function CalcNewDy() {
  // Function to calculate the new y velocity of the ball
  // This is not correct physics, (no bounce). Just take on the velocity of the bat
  return _root.bat.dy;
}
```

Again, many comments have been used to describe the script. To save time they can be omitted, but they do help explain the code.

For this Object the **onLoad()** function is used to declare and initialise variables. dx and dy are used to hold the Object's velocity. These are initialised to 0.

MinX, MinY, MaxX and MaxY are used to hold the boundary area that the ball is allowed to travel in. These points are based on the size of the boundary Object and are corrected to take into account the radius of the ball.

Batradius is used to hold the combined radius of the bat and the ball. This is calculated from bat._width / 2 + ball._width / 2. This radius is used when checking a collision condition. A collision has occurred when the two Objects are within Batradius of each other (ie. touching or overlapped).

The **CalcNewDx()** and **CalcNewDy()** functions can be entered via **Add Script | Define function (...)** (Leave all parameter fields blank). The functions are used to calculate the new x and y velocity of the ball after a collision with the bat. Functions are used here as they provide a way of grouping sections of code. This can be used to help keep each module small enough to be easily understood. For maximum readability, no function should extend beyond 1 page of display. Initially, the functions do not calculate the physics associated with the ball bouncing off the bat. The bat is assumed to be a 'sticky' bat, i.e. the ball simply adopts the current velocity of the bat.

The **onEnterFrame()** Event function is called on each Frame. It performs the following tasks:

- updates current ball position based on current dx, dy values
- checks for collision with boundary. Reverses dx, dy if collision is detected
- checks for collision with bat. Updates dx, dy according to values returned from CalcNewDx() and CalcNewDy() functions.

Note: When analysing the script, `a += b;` means `a = a + b;`.

7. Test the game by selecting the 'Layout' tab and pressing the 'Play' button. The balls should be stationary. Click on the red ball (bat object) with the mouse. The bat will now follow the mouse pointer until you click the mouse again. Use the bat to hit the blue ball. The blue ball should continue to bounce around the court until it meets the bat again

8. Press the 'Stop' button. Select the ball Sprite in the 'Layout' Panel and right click. Select Make Instance. If the new instance appears off the page, drag it back into the playing area. Press the 'Play' button. There are now two ball Objects that can be hit around the court. Note that the Sprite and the Instance behave independently of each other as each of their variables and associated properties (dx, dy, _X, _Y etc.) are local to the individual Object

Analysis
The onEnterFrame() Event Handling function can be used to animate Objects by altering their _X and _Y properties on a Frame-by-Frame basis.

The onLoad() Event Handling function is useful for defining and initialising variables. If the variables x0 and y0 were defined and initialised within the **onEnterFrame()** function, their previously stored values would be lost when the function is called on the following Frame.

User defined functions (e.g. **CalcNewDx()** and **CalcNewDy()**) can be used to group sections of code and improve readability.

An Instance of a Sprite can behave independently (in a scripting sense) of the original Sprite and is a useful way of creating new game Objects.

Physics Properties

Top Previous Next

Description
Shows how various gaming functions can be simplified by using Physics Properties.

Aim
Introduce the concepts of velocity, acceleration and friction.
Introduce the available SWiSH specific Physics Properties and demonstrate how they can simplify such scripting.

.swi file
"game1.swi"

Step 1
Continue from the previous tutorial or open the file "game.swi"
Save the file as "mygame2.swi"

Step 2
Enable Physics properties for the movie. See here for instructions.
The bat, ball and the ball instance objects. See here for detailed instructions.

Step 3
Modify the bat object's onEnterFrame() Event so that it calculates velocity based on pixels per second (as opposed to pixels per frame as it currently does). All physics properties are based on per second values.

```
onEnterFrame() {
   // Calculate and save the distance moved since the last frame. This is the velocity.
   // The _delta property is used to convert these values to pixels per second.
   dx = (_X - x0) / _delta;
   dy = (_Y - y0) / _delta;
   // Save the current position for calculation in the next frame
   x0 = _X;
   y0 = _Y;
}
```

The physics property _delta represents the time period since the last frame. Dividing by this value converts the

dx, dy velocities from pixel per frame into pixel per second velocities.
Note that the Physics properties must be enabled for the bat object to allow access to the _delta property.

Step 4
Modify the "ball" Sprite's onEnterFrame() Event so that it uses the physics properties _vX and _vY instead of dx and dy from the previous tutorial.

```
onEnterFrame() {
   // Move ball to new position (This is now done automatically by _vX and _vY properties).
   //
   // Check for boundary collision. If collision detected, reverse velocity
   if (_X < MinX) {
      // Force _vX to be positive.
      _vX = Math.abs(_vX);
   }
   if (_X > MaxX) {
      // Force _vX to be negative.
      _vX = -Math.abs(_vX);
   }
   if (_Y < MinY) {
      // Force _vY to be positive.
      _vY = Math.abs(_vY);
   }
   if (_Y > MaxY) {
      // Force _vY to be negative.
      _vY = -Math.abs(_vX);
   }
   //
   // Check for collision with bat.
   if (_parent.bat.isNearThis(batradius)) {
      // We have collided with bat, change ball velocity.
      _vX = CalcNewDx();
      _vY = CalcNewDy();
   }
}
```

Note that the lines

```
   // Move ball to new position
   _X += dx;
   _Y += dy;
```

Have now been removed. The dx, dy lines can also be removed from the ball.onLoad() event handling function as they are no longer used.

Step 5
Test the game.
Select the layout tab and press play. The game should behave in a similar fashion to the previous example.

Step 6
Press the stop button.

Analysis
In this tutorial, physics properties were introduced. These are SWiSHscript-specific. The properties _vX and _vY were used for Velocity, and _delta was used for measuring time. These properties, as used in this tutorial, determine how fast the ball moves - and in which direction. It is important that Physics Support be enabled for both the exported Script and the Selected Object in order for these scripts to work properly.

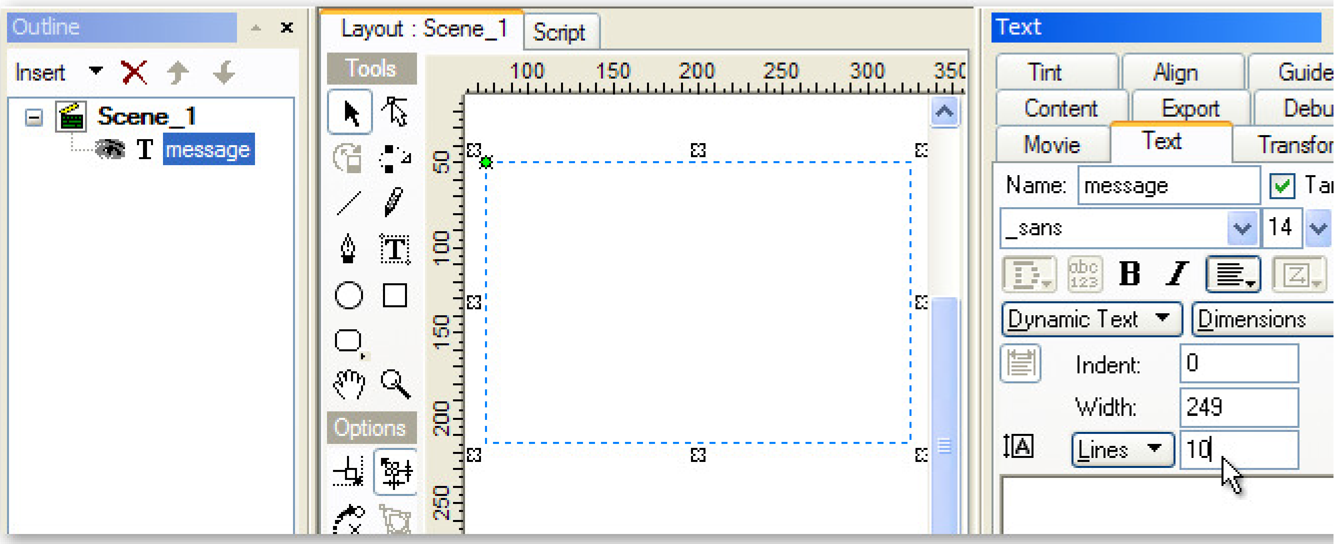
Data Transfer - text

Top Previous Next

Description
Demonstrates how data can be transferred to and from .htm, .asp, .php or text files that are resident on a server.

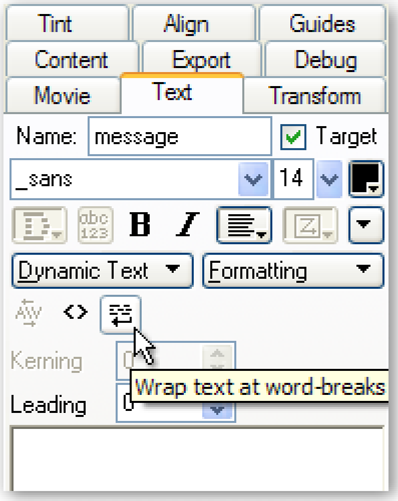
Aim
Introduces the methods loadVariables() and shows how you can load an external .txt file into your movie at runtime.

1. Create a New movie and save it as "mytxtmessage.swi". Insert a Dynamic Text object. In the 'Dimensions' options, untick the Auto-Height icon and choose 'Lines' rather than 'Height' - Set it for several lines to accommodate extra text. Name the text object 'message' and tick the 'Target' option as shown below:

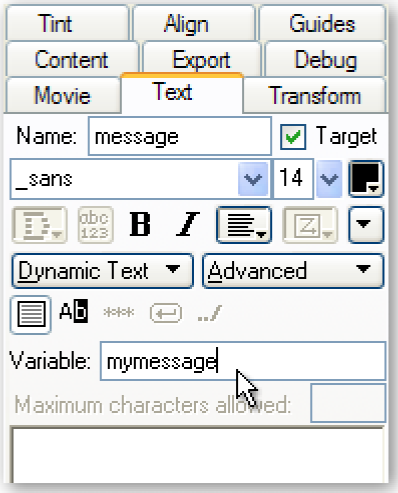


Name: message ☑ Tar
_sans 14
Indent: 0
Width: 249
Lines 10

2. In the 'Formatting' options, make sure the 'Word-Wrap' option is selected. In the 'Advanced' options, press the 'Black border with white background' option and assign a variable "mymessage" to this text object as shown below:



Name: message ☑ Target
_sans 14

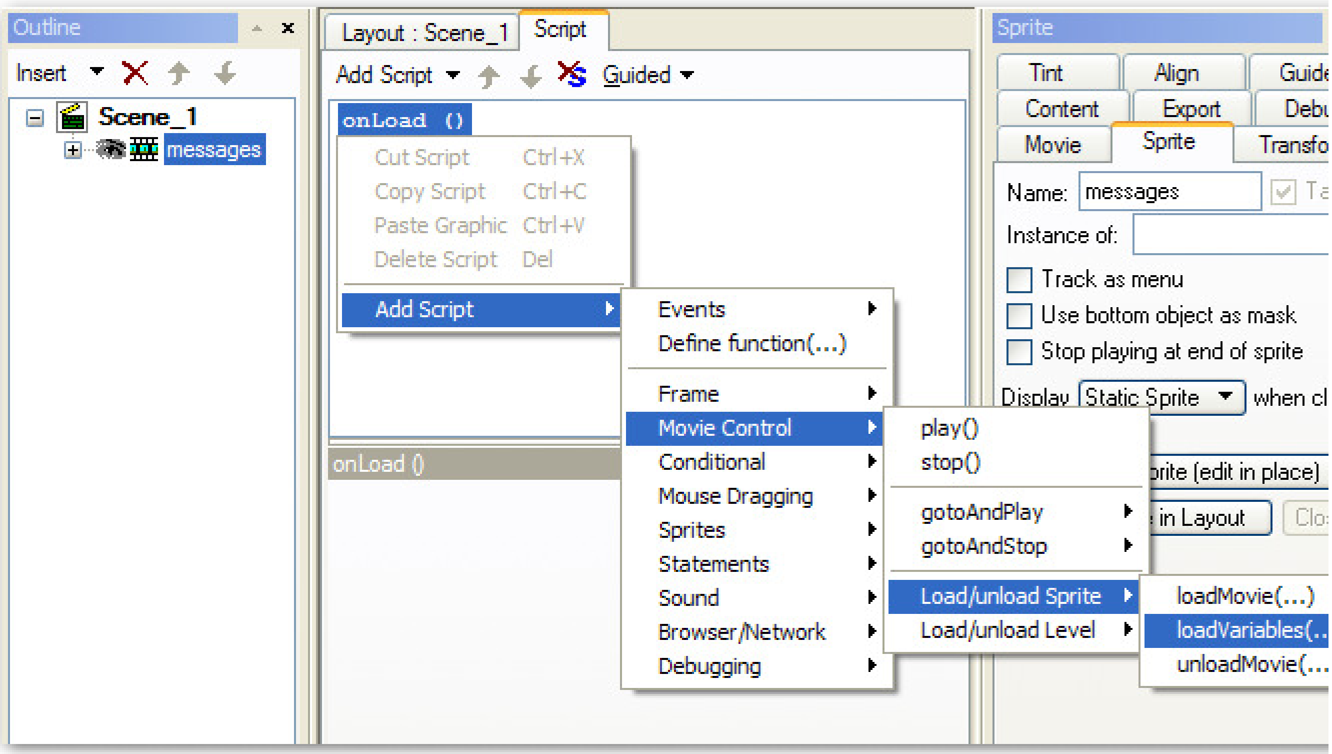


Name: message ☑ Target
_sans 14
Variable: mymessage

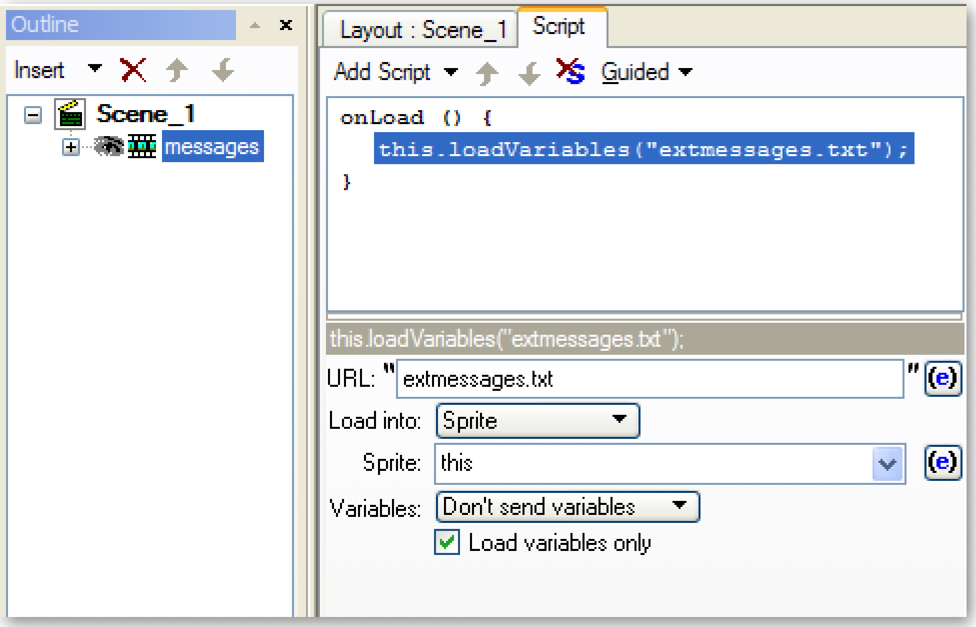
3. Use the Modify menu and select **Grouping | Group as Sprite**, then name the sprite "messages" on the 'Sprite' panel.

4. With the sprite object selected in the 'Outline' panel, open the 'Script' panel and press the 'Add Script' button. Select **Events | Frame | onLoad()** from the menu.

5. Right-Click on the onLoad() action and select **Add Script | Movie Control | Load/unload Sprite | loadVariables(...)** from the menu.



6. In the loadVariables() options, type "extmessages.txt" in the URL field, and type "this" into the Sprite field. If the option "Load variables only" is not ticked, do so now. The 'Script' panel should look similar to this:



```
onLoad () {
    this.loadVariables("extmessages.txt");
}
```

this.loadVariables("extmessages.txt");

URL: "extmessages.txt"

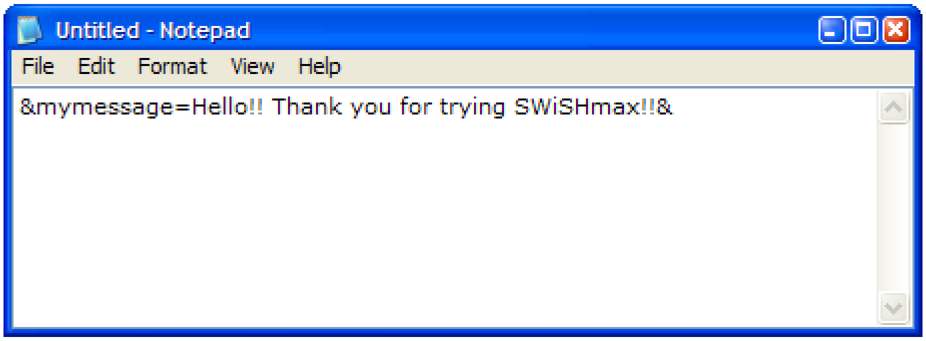
Load into: Sprite

Sprite: this

Variables: Don't send variables

☑ Load variables only

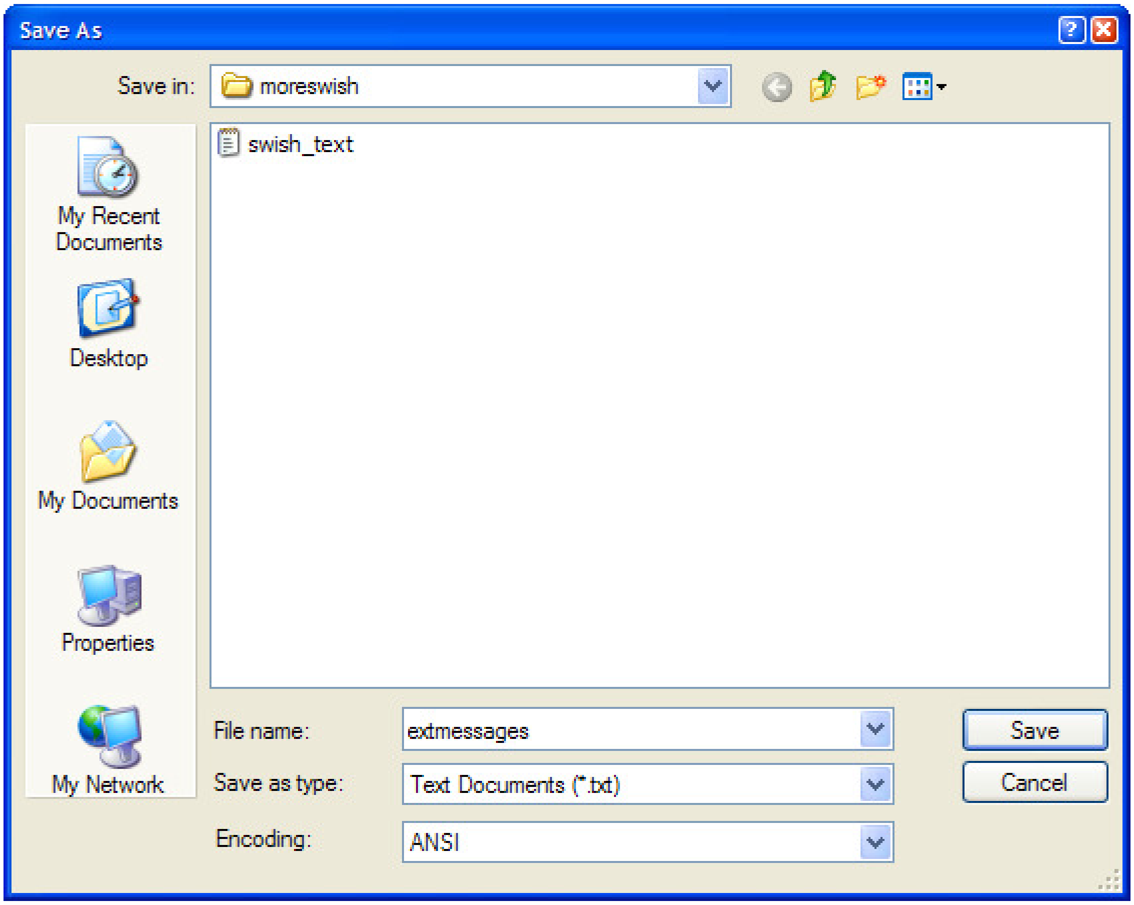
7. Next, open Windows Notepad (or any text-editing program) and type what is shown in the image below:



&mymessage=Hello!! Thank you for trying SWiSHmax!!&

Note: Notice that in the image above, the text is defined by two ampersand characters "&". When loading external .txt files into a SWF movie, you are not allowed to use the ampersand character in the text messages, because that character tells the Flash player that the text has ended. So, be sure to use "and" instead of "&" when typing the body of the text. Also, note that we have used the Variable assigned to the Dynamic text object - not the name.

8. After typing the body of the text message, save this file as "extmessages.txt" (which is the URL used above for the loadVariables() action) and note the folder you save this .txt file to.



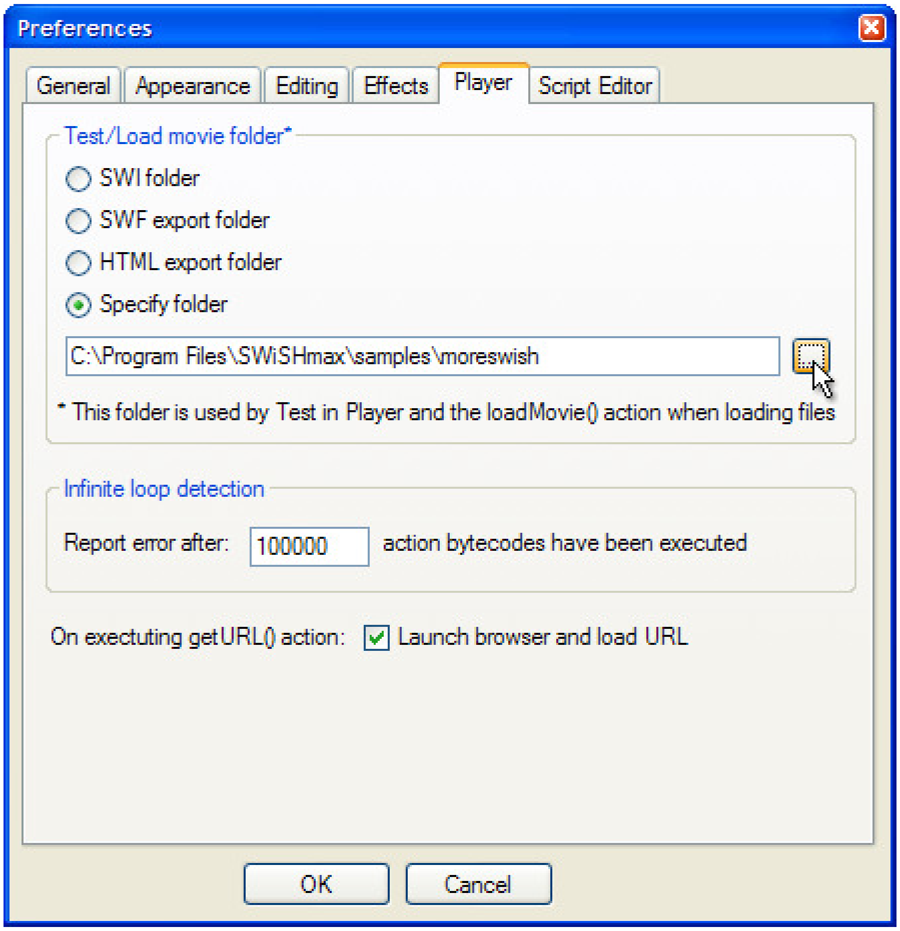
Save in: moreswish

File name: extmessages

Save as type: Text Documents (*.txt)

Encoding: ANSI

9. In SWiSHmax, open the Tools menu, select Preferences, then open the Player options. Tick the "Specify folder" option and press the browse button to locate the folder the "extmessages.txt" file was saved in on Step #8 above.



Test/Load movie folder*

○ SWI folder
○ SWF export folder
○ HTML export folder
● Specify folder

C:\Program Files\SWiSHmax\samples\moreswish

* This folder is used by Test in Player and the loadMovie() action when loading files

Infinite loop detection

Report error after: 100000 action bytecodes have been executed

On exectuting getURL() action: ☑ Launch browser and load URL

10. Make sure you are in the 'Layout' panel in SWiSHmax then press the 'Play' button in the Control Toolbar to preview the movie. The message typed in to the .txt file should be displayed inside the Dynamic text field.

Analysis

When the 'Play' button is pressed, the onLoad() event is triggered for the sprite "messages". The loadVariables() action was set to only load variables which will load the variables defined in the .txt file at the specified URL.

Using SWiSH with Other Programs

Top Previous Next

This section uses examples to describe the how to incorporate your SWiSH animation into the following programs:

- Macromedia Flash
- Macromedia Dreamweaver
- Microsoft FrontPage 2000
- Microsoft Powerpoint.

Using SWiSH with Flash

Top Previous Next

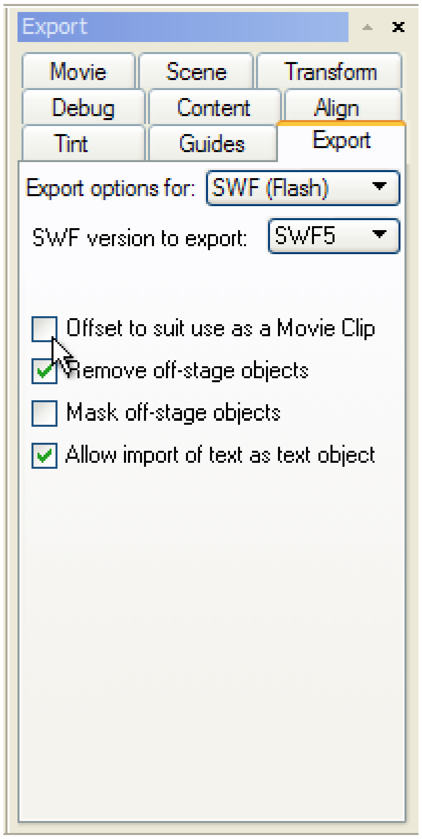
This is a step-by-step tutorial on using your SWiSH (.swf) Movie in Flash.

Note: Flash cannot embed your .swi (SWiSH working file format) Movie, so you will need to export your Movie from Swish as a .swf file

Exporting your .swf Movie from SWiSH

1. Start with the Movie you created in the first tutorial or open "first.swi" from the Samples Menu

2. Uncheck the 'Offset Movie to suit use as a Movie Clip' option on the SWF section of the Export Panel



Export options for: SWF (Flash)

SWF version to export: SWF5

□ Offset to suit use as a Movie Clip

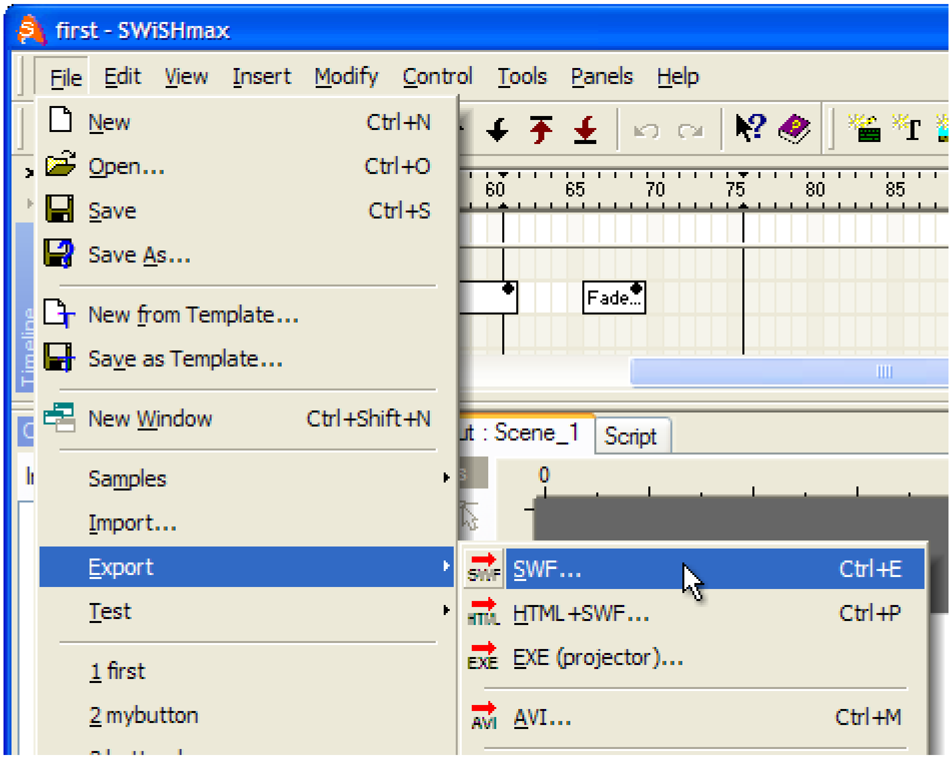
☑ Remove off-stage objects

□ Mask off-stage objects

☑ Allow import of text as text object

3. Press the SWF 'Export to SWF' button on the Export Toolbar, or select Export | Swf from the File Menu. You should now see the 'Export to SWF' dialog box

Note: If the 'Export' Toolbar is not visible, select View | Toolbars. The dialog box will enable you to turn on the Export Toolbar



4. Choose the folder that you want to export the .swf file to, enter a file name of "first.swf" (without quotation marks) and press the 'Save' button

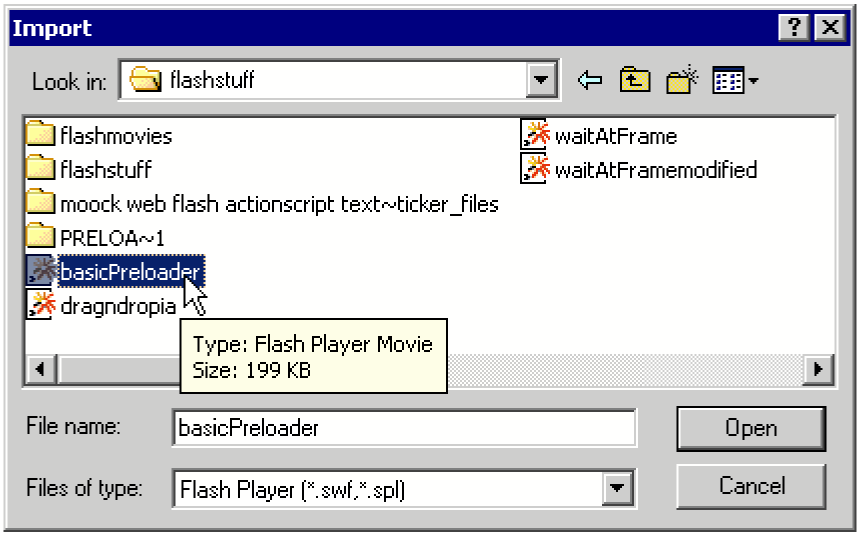
Importing your .swf file into Flash
There are three possible techniques for importing SWiSH .swf Movies into Flash.

Note: There is a bug in Flash, in addition to the limitations mentioned above, where it will not import some valid .swf files. In particular, it will crash if a .swf file does not have a particular record that is not required in a .swf file. There are two situations in which SWiSH leaves out this optional record. The first is when you select the **[x]** '**Solid shape with overlap filled'** checkbox on the Shape Panel. This option tells SWiSH to export a shape as a font, which has different behavior for self-overlapping shapes. The information Flash requires is not applicable and so is not exported. The second is when you explicitly turn off the font info record with the **[x] 'Allow import of text as text object'** checkbox on the Export Movie Panel. This tells SWiSH to omit the optional record for all text, which reduces the size of the resulting .swf file.

Method 1. Importing a SWiSH .swf Movie into a Flash layer
This is the simplest way to import a SWiSH .swf Movie and is an appropriate method if you do not want to reposition the SWiSH animation after you have imported it into Flash.

1. Start the 'Macromedia Flash' application and open the Flash Movie (.fla file) you wish to add the SWiSH Movie to

2. Select Import from the File Menu. You should now see the 'Flash Import' dialog box:



Import

Look in: flashstuff

flashmovies
flashstuff
moock web flash actionscript text~ticker_files
PRELOA~1
basicPreloader
dragndropia
waitAtFrame
waitAtFramemodified

Type: Flash Player Movie
Size: 199 KB

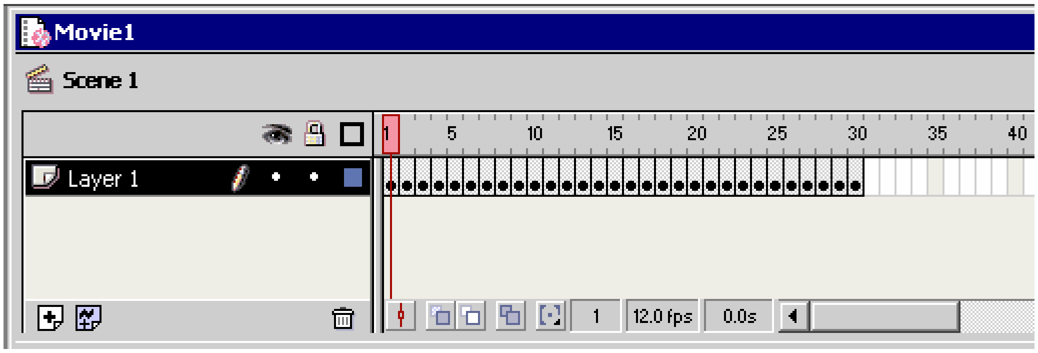
File name: basicPreloader

Files of type: Flash Player (*.swf,*.spl)

3. Select "Flash Player (*.swf, *.spl)" from the 'Files of type' box

4. Locate to the folder where you exported the .swf file, choose "first.swf", and press Open

5. You should now see a new layer in the Flash Timeline. This layer will contain the entire SWiSH Movie



Note:

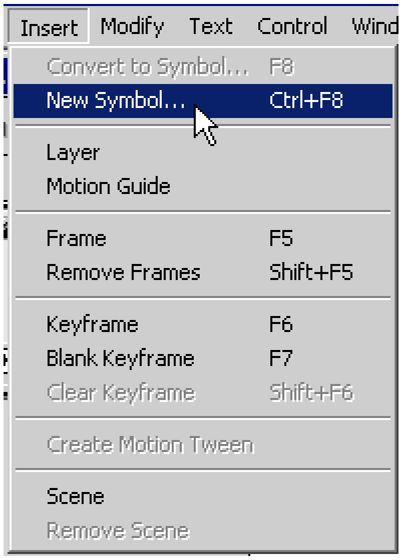
- Swish cannot export a .fla file directly because .fla is a closed format. Only Macromedia Flash can read and write .fla files
- All Keyframe and tweening information is lost when the SWiSH .swi Movie is converted to a .swf file and imported into Flash. Flash will import every Frame as a Keyframe. This does not mean .swf files made with SWiSH are larger than those made with Flash. This is not really a problem unless you want to edit the tweens on individual letters in Flash. It is much easier to do your edits in SWiSH and import the whole thing into Flash

- To relocate your imported elements, you will need to move every Keyframe, and for this reason, it may be better to import your SWiSH .swf into Flash as a Movie Clip symbol

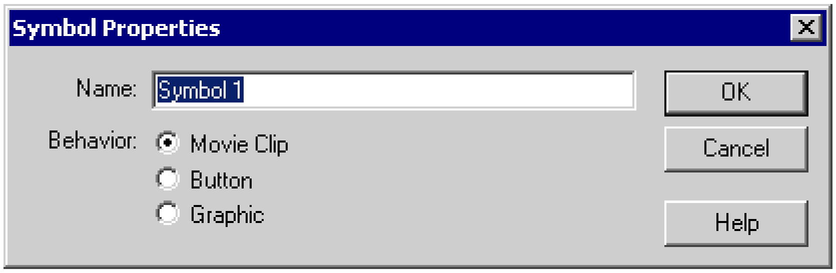
Method 2. Importing a SWiSH .swf Movie into a Flash Movie Clip
When a Swish .swf Movie is imported into Flash as a new layer, it can be difficult to select and move around the stage. The Swish Movie is much easier to manipulate if you import the .swf into a Movie Clip first, and then place an Instance of the Movie Clip on the stage.

1. Start the 'Macromedia Flash' application and open the Flash Movie (.fla file) you would like to add the SWiSH .swf Movie to

2. Select New Symbol from the Insert Menu



You should see the 'Symbol Properties' dialog box:



Name: Symbol 1

Behavior: ● Movie Clip ○ Button ○ Graphic

3. Call the symbol "first.swi", select the 'Movie Clip' Behavior and press OK

4. Select Import from the File Menu. You should now see the 'Flash Import' dialog box

5. Select 'Flash Player (*.swf, *.spl)' from the 'Files of type' box

6. Navigate to the folder where you exported the .swf file, choose "first.swf", and press Open

7. Now select the Flash Scene where you would like to add the SWiSH animation

8. Select Library from the 'Window' Menu if the Library is not already open

9. Drag and drop "first.swi" from the 'Library' window onto the stage. You can now select and move the SWiSH animation as a single Object

10. Select Control | Test Movie or File | Publish Preview to play the complete animation

Note:

- Unless instructed otherwise, the SWiSH Movie Clip will begin playing at the Frame where you dropped it
- You can stop and start the SWiSH Movie Clip by giving it an Instance name and using Tell Target to control the SWiSH animation
- Flash does not import the attached sounds and Actions with any .swf file, regardless of whether the .swf file was made with SWiSH, Flash or any other application

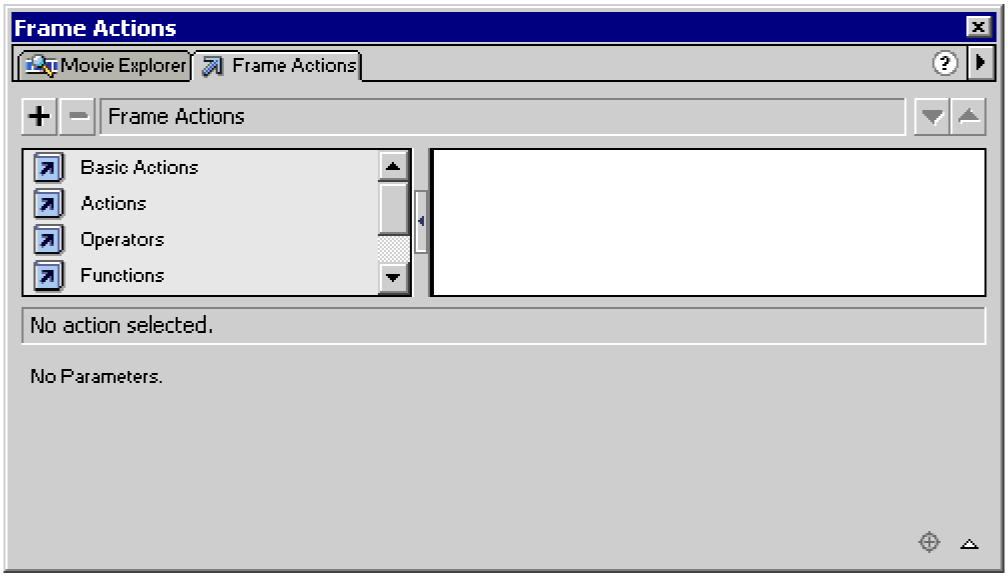
Method 3. Importing a SWiSH .swf Movie using the Load Movie Action
To get around the limitation of not being able to import sounds and Actions, it may be better to use the Load Movie Action in Flash.

1. Start the 'Macromedia Flash' application and open the Flash Movie (.fla file) you would like to add the SWiSH .swf Movie to

2. Click the 'Add Layer' button in the bottom left of the Flash Timeline

3. Double-click on the Frame where you would like the SWiSH Movie to begin

4. Select the Frame Actions tab on the 'Frame Actions Properties' dialog box



No action selected.

No Parameters.

5. Click the '+' button and choose Load/Unload Movie

6. In the 'URL' box type "first.swf"

7. Select Control | Test Movie or File | Publish Preview to play the complete animation

Note:

- The SWiSH .swf Movie should be exported to the same folder as the Flash Movie, otherwise the Load Movie Action will fail
- The 'Flash Output' window will display an error message when it cannot load a Movie
- Flash uses the Windows Temporary Folder when you select the Control | Test Movie option
- Flash uses the Flash installation folder when you select File | Publish Preview

Using SWiSH with Dreamweaver 4

Top Previous Next

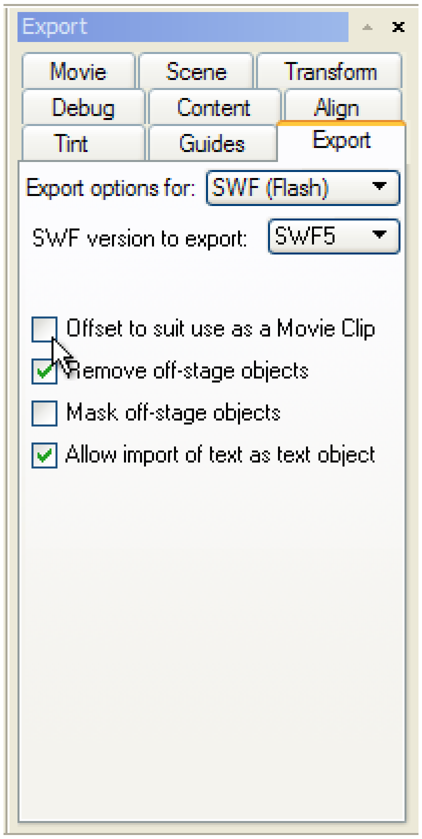
This is a step-by-step tutorial on using your SWiSH (.swf) Movie in Dreamweaver 4.

Note: Dreamweaver cannot embed your .swi (SWiSH working file format) Movie. You will need to export your SWiSH Movie to the Flash Player file format (.swf)

Exporting your .swf file from SWiSH

1. Start with the Movie you created in the first tutorial or open "first.swi" from the Samples Menu

2. Uncheck the 'Offset Movie to suit use as a Movie Clip' option on the SWF section of the Export Panel



Export options for: SWF (Flash)

SWF version to export: SWF5

□ Offset to suit use as a Movie Clip

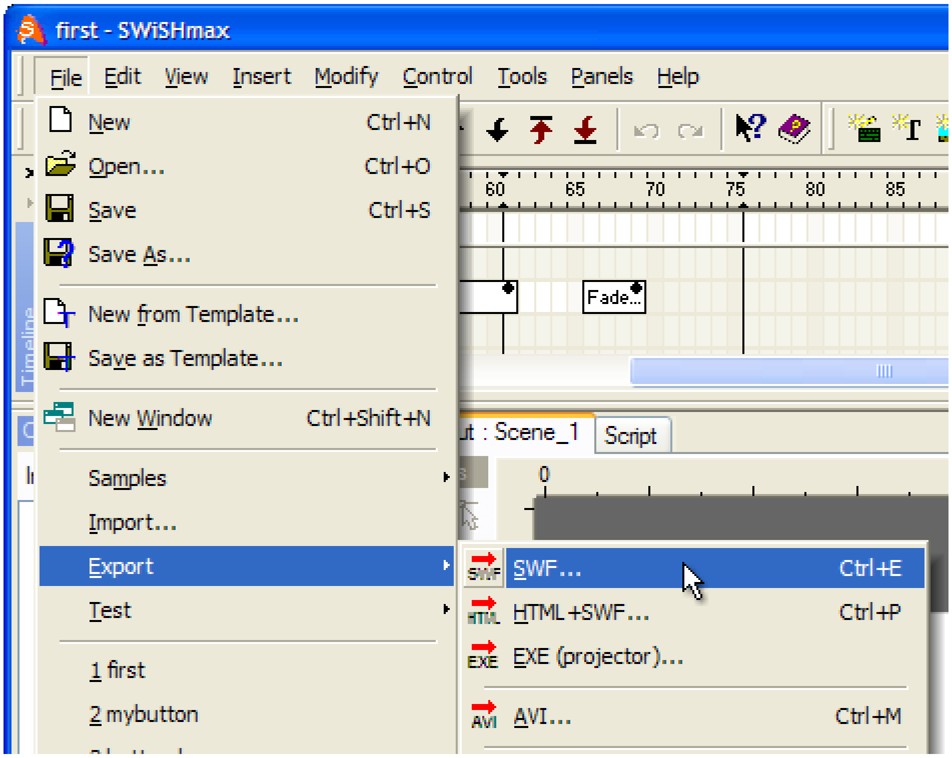
☑ Remove off-stage objects

□ Mask off-stage objects

☑ Allow import of text as text object

3. Press the 'Export to SWF' button on the Export Toolbar, or select Export | SWF from the File Menu. You should now see the 'Export to SWF' dialog box

Note: If the 'Export to SWF' button is not visible, select View | Toolbars. The dialog box will enable you to turn on the Export Toolbar



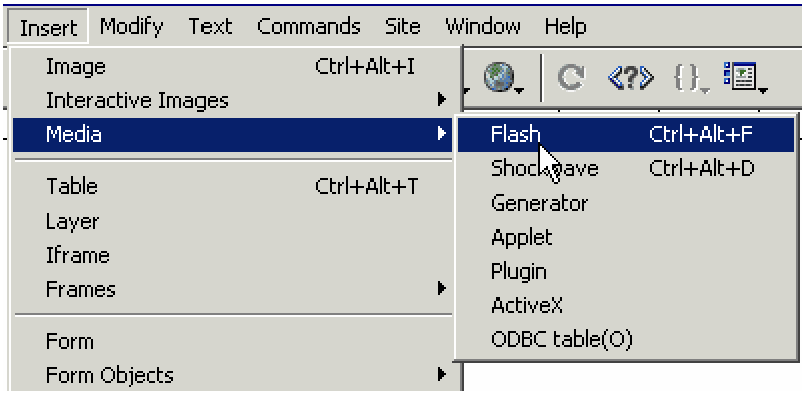
4. Choose the folder that you want to export the .swf file to, enter a file name of " first.swf " (without quotes) and press the 'Save' button

Note: Any export parameters that you set for HTML in the 'Export' Panel of SWiSH will be ignored by Dreamweaver if you simply import your .swf file rather than producing your HTML page in SWiSH and then opening it in Dreamweaver to edit

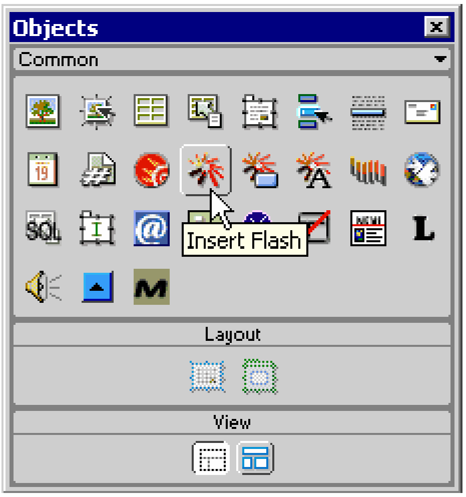
To insert your .swf Movie into an HTML page using Dreamweaver

Open dreamweaver and you will find that there are 2 ways to access the 'insert Flash' Object:

1. From the Main Menu bar select Insert | Media | Flash (Ctrl+Alt+F)



2. From the Dreamweaver 'Objects' Panel (Ctrl+F2), select the Flash icon and click on it, or drag it to the insertion point on your page



Once you have selected the Movie you wish to insert, Dreamweaver will generate the most common parameters automatically.

If you open your Code Inspector (F10), you will see the code that is produced.
It will look like this (the parameters automatically set by Dreamweaver are highlighted):

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=5,0,42,0"
id="mycalculator" width="400" height="300" align=right>
```

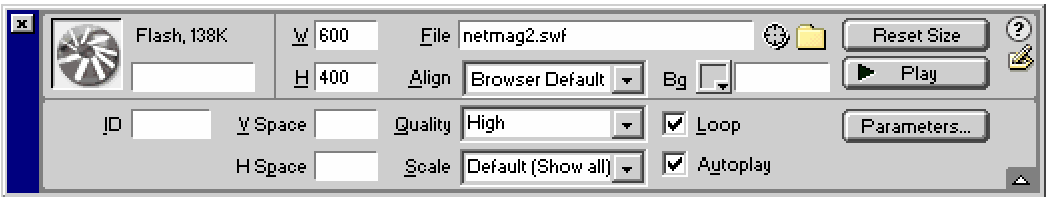
```
<param name="movie" value="mycalculator.swf">
<param name="quality" value="high">
<param name="scale" value="noborder">
<param name="salign" value="r">
<param name="align" value="right">
<param name="wmode" value="transparent">
<param name="bgcolor" value="#FCFAFA">
<param name="allowScriptAccess" value="sameDomain">
<embed name="mycalculator" src="mycalculator.swf"
quality="high" scale="noborder" salign="r" align="right" wmode="transparent" bgcolor="#FCFAFA"
swLiveConnect="true" allowScriptAccess="sameDomain"
width="400" height="300"
type="application/x-shockwave-flash"
pluginspage="http://www.macromedia.com/go/getflashplayer"></embed>
</object>
```

This code contains both the <embed></embed> tags recognized by Netscape Navigator and the <object></object> tags recognized by Internet Explorer.

The Properties Inspector

Your Movie properties can be altered and added to by using the Properties Inspector (Ctrl+F3).

Note: Dreamweaver 3 users may see a slightly different Properties Inspector (without the Play button)



Flash, 138K
W 600
H 400
File netmag2.swf
Align Browser Default
Bg
ID
V Space
H Space
Quality High
Scale Default (Show all)
☑ Loop
☑ Autoplay

Exploring the Movie parameters in the Properties Inspector

W, H: specify the width and height parameters of your Movie in pixels (you can also specify %).

Align: specifies the align parameter, which aligns your Movie to a selected side of the browser window and crops the other sides to fit if needed.

Bg: specifies the BGCOLOR (background color) parameter for the Movie area. This color also appears while the Movie is loading or has finished playing (this is an optional parameter).

Quality: sets the quality parameter for the Object and embed tags that run the Movie. The settings relate to the level of anti-aliasing used during playback of the Movie. With higher settings a Movie looks better, but requires a faster processor to render correctly, while lower settings emphasize speed above appearance (this is an optional parameter).

Scale: sets the scale parameter for the Object and Embed tags that run the Movie (this is an optional parameter).

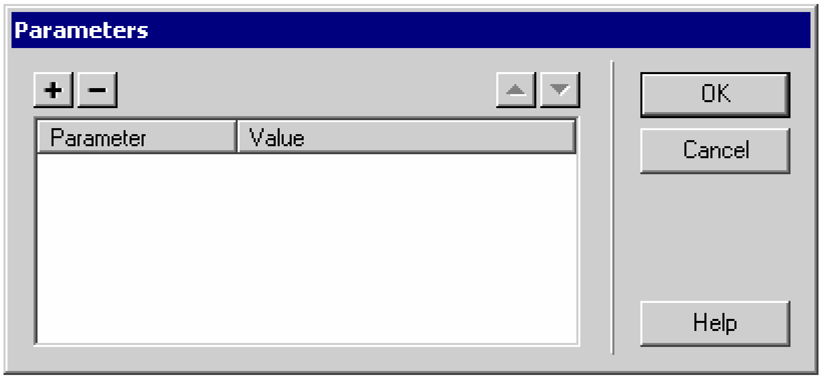
Autoplay: defines the play parameter. A value of 'true' plays the Movie automatically when the page loads (this is an optional parameter).

Loop: controls the loop parameter, if set to true it makes the Movie loop indefinitely (this is an optional parameter).

id: is an ActiveX ID parameter, which is used to pass information between ActiveX controls (this parameter is automatically generated in your SWiSH produced HTML page) [this is an optional parameter].

V Space, H Space: HTML parameters, which set the amount of white space between your Movie and the browser window.

'Parameters...' button: opens a dialog box where you can add additional parameters to pass on to the Movie.



You can preview your Movie from the 'Properties' Panel by pressing the 'Play' button.

Editing your SWiSH produced HTML page in Dreamweaver

Open your SWiSH-built HTML page in Dreamweaver. You will notice that SWiSH generates a more complete set of parameters than is produced by Dreamweaver. For this reason, it is considered better to produce your HTML page in SWiSH and then import it into Dreamweaver for any further editing.

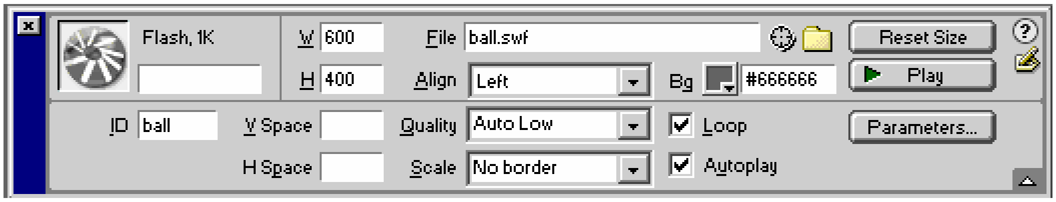
If you open your Code Inspector (F10), you will see the code that is produced.
It will look similar to this (the parameters set by the user in SWiSH are highlighted):

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=5,0,42,0"
id="mycalculator" width="400" height="300" align=right>
<param name="movie" value="mycalculator.swf">
<param name="quality" value="high">
<param name="scale" value="noborder">
<param name="salign" value="r">
<param name="align" value="right">
<param name="wmode" value="transparent">
<param name="bgcolor" value="#FCFAFA">
<param name="allowScriptAccess" value="sameDomain">
<embed name="mycalculator" src="mycalculator.swf"
quality="high" scale="noborder" salign="r" align="right" wmode="transparent" bgcolor="#FCFAFA"
swLiveConnect="true" allowScriptAccess="sameDomain"
width="400" height="300"
type="application/x-shockwave-flash"
pluginspage="http://www.macromedia.com/go/getflashplayer"></embed>
</object>
```

This code contains both the <embed></embed> tags recognized by Netscape Navigator and the <object></object> tags recognized by Internet Explorer. It also covers a more comprehensive range of .swf Movie parameters.

To edit your Movie properties open the Properties Inspector (Ctrl+F3).

Note: Dreamweaver 3 users may see a slightly different Properties Inspector (without the Play button)



Flash, 1K

W 600 File ball.swf

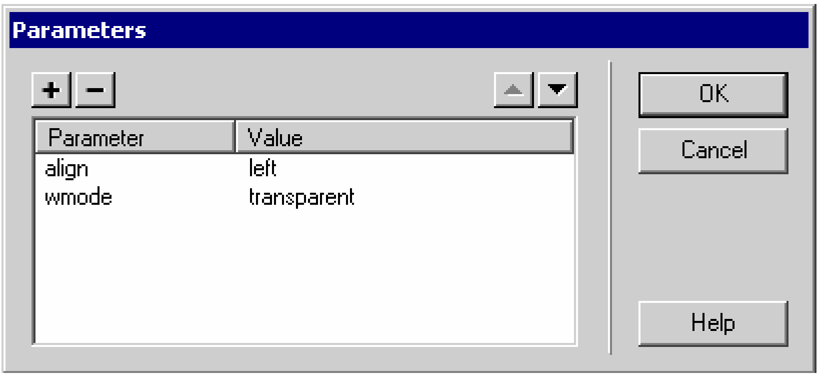
H 400 Align Left Bg #666666

ID ball V Space Quality Auto Low ☑ Loop

H Space Scale No border ☑ Autoplay

You will see some of the parameters that you specified in the 'Export' Panel in SWiSH. You can change these by selecting alternatives from the drop-down Menus, or typing a value into the text input boxes.

To see all the parameters you set, you will also need to open the 'Parameters' dialog box by pressing the 'Parameters' button on the Properties Inspector.



Parameters

Parameter	Value
align	left
wmode	transparent

This dialog box and the Properties Inspector will allow you to edit all the .swf file parameters of your Movie. You can also edit the code directly by opening the Code Inspector (F10) and typing in your changes.

Note: Some .swf Movie parameters should not be changed, or changed with care:

- classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
  Identifies the media type and should always be left as shown for .swf Movies
- codebase="http://active.macromedia.com/flash2/cabs/swflash.cab#version=4,0,0,0"
  You *can* change the cab#version to =5,0,0,0 to force the Movie to get the ActiveX control for Flash Player Version 5
- pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_Version=ShockwaveFlash"
  This is the location of the Flash player plugin for Netscape Navigator or browsers that do not use ActiveX controls

Using SWiSH with FrontPage 2000

Top Previous Next

This is a step-by-step tutorial on how to import your SWiSH (.swf) Movie into FrontPage 2000.

Note: FrontPage 2000 cannot embed your .swi (SWiSH working file format) Movie directly. You will need to export your SWiSH Movie in the Flash Player file format (.swf)

Importing your SWiSH Movie into FrontPage is a three-part process involving three files:

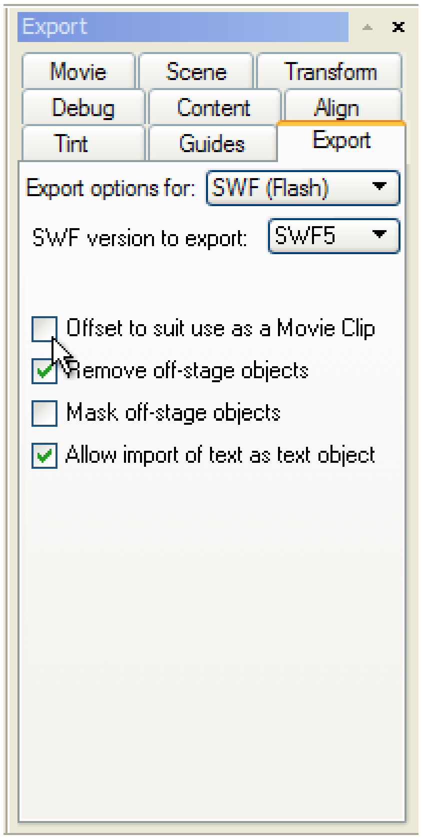
- the .swi file is your SWiSH Movie saved in the SWiSH native file format
- the .swf file is your SWiSH Movie saved in the Flash Player file format
- the .htm file is the web page that acts as a 'container' for the SWiSH Movie when it is displayed in a web browser.

The .swi file is not required to play your SWiSH Movie in a web page, and does not need to be uploaded to your web server. Both the .swf and .htm files must be uploaded your web server and they must be in the same folder.

Step 1: Export your SWiSH Movie to SWF

1. Start with the Movie you created in the first tutorial or open "first.swi" from the Samples Menu

2. Uncheck the 'Offset Movie to suit use as a Movie Clip' option on the SWF section of the Export Panel)



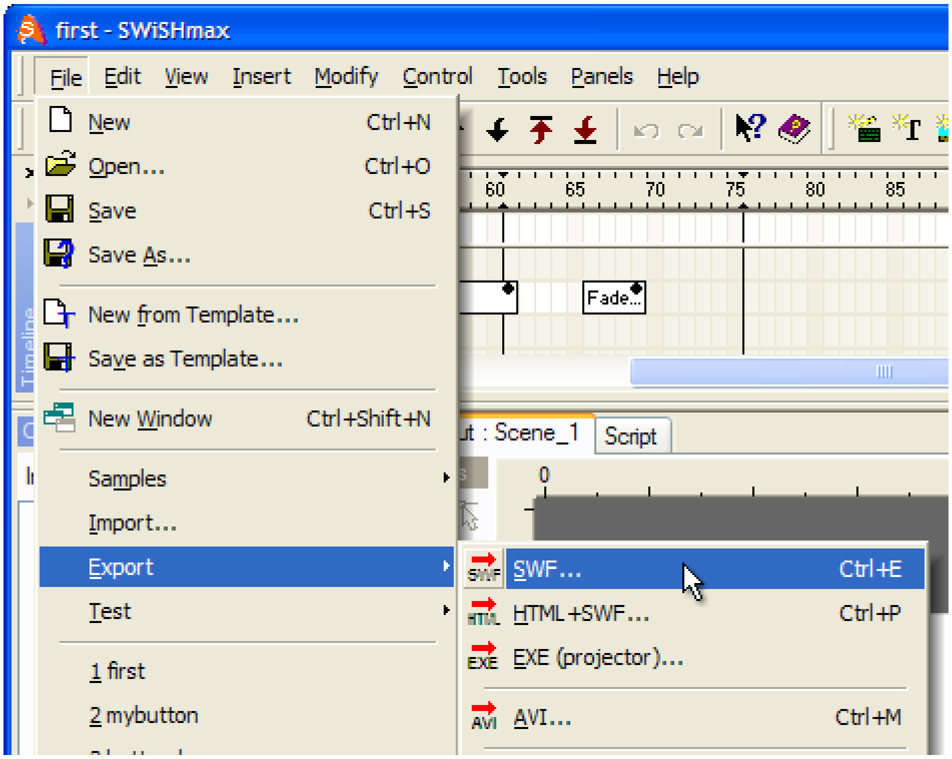
Export options for: SWF (Flash)

SWF version to export: SWF5

☐ Offset to suit use as a Movie Clip
☑ Remove off-stage objects
☐ Mask off-stage objects
☑ Allow import of text as text object

Note: If you can't see the 'Export' Panel, select Export from the Panels Menu

3. Select Export | SWF from the File Menu



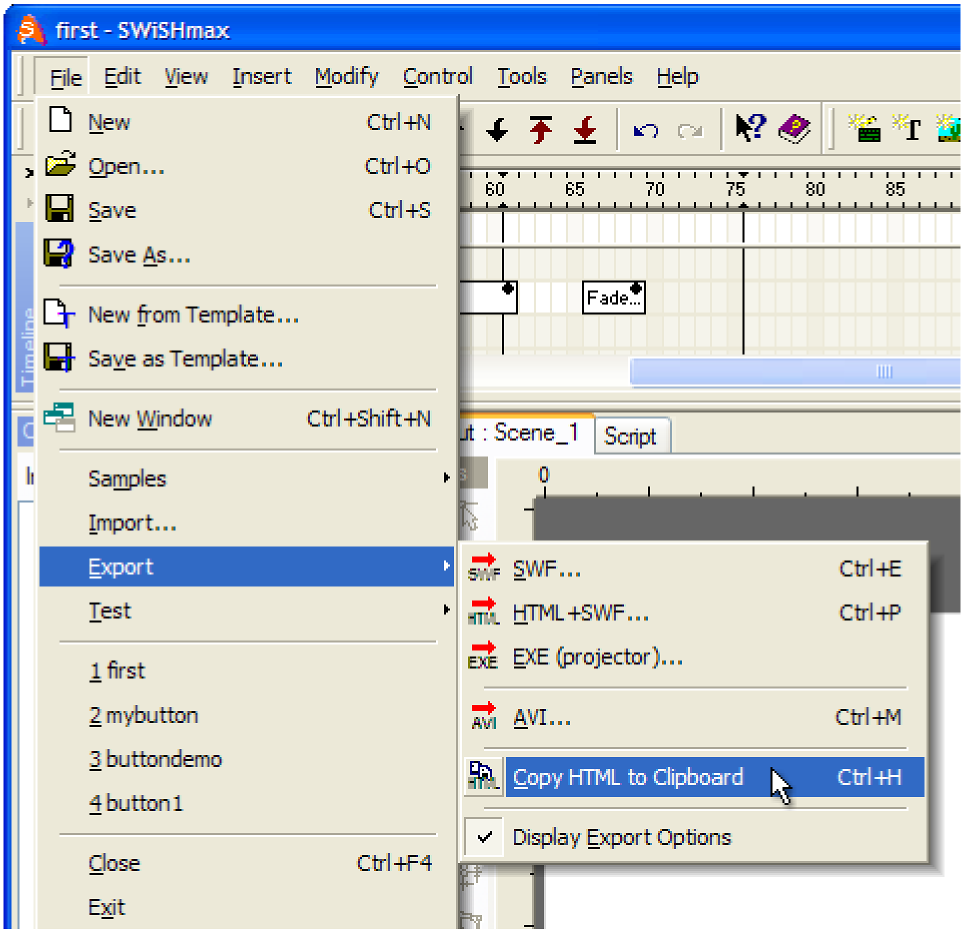
You can also press the 'Export to SWF' button on the Export Toolbar, if this toolbar is visible. If the button is not visible, select View | Customise | Toolbars. The dialog box will enable you to turn on the Export Toolbar

4. You should now see the 'Export to SWF' dialog box. Choose the same folder where your web page (.htm) file is located, enter a file name of "first.swf" (without quotes) and press the 'Save' button

Note: You must save the Flash Player (.swf) file in the same folder as the web page (.htm) file

Step 2: Copy HTML to Clipboard

1. Select Export | HTML to Clipboard from the File Menu. This copies the HTML code required to display the SWiSH Movie onto the Windows Clipboard so you can paste it into your web page

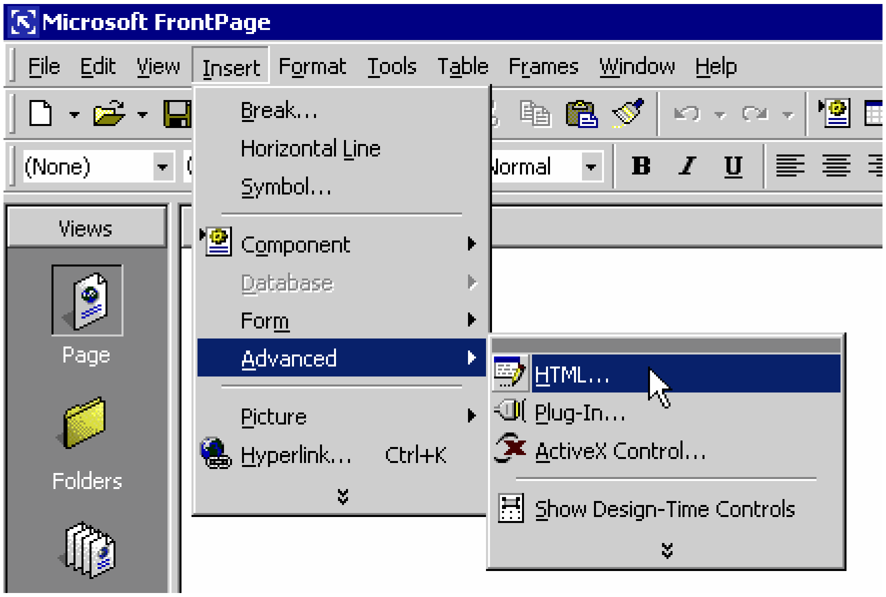


Step 3: Insert your SWiSH Movie into a web page using FrontPage

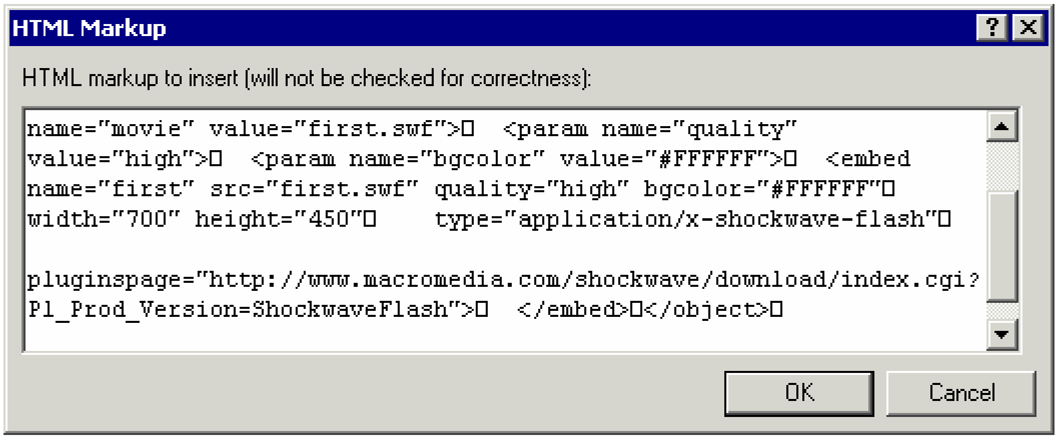
1. Start FrontPage and open the web page (.htm file) that you want to insert the SWiSH Movie into. If you are inserting the SWiSH Movie into a new web page, you should first save the empty web page in the same folder as the .swf file

2. Position the caret at the place in your web page where you want to insert the SWiSH Movie

3. From the FrontPage Main Menu, select Insert | Advanced | HTML...



4. You should now see the 'HTML Markup' window. Right-click in the text area and choose Paste. The 'HTML Markup' window should now look like the screenshot below:



HTML markup to insert (will not be checked for correctness):

```
name="movie" value="first.swf">□  <param name="quality"
value="high">□  <param name="bgcolor" value="#FFFFFF">□  <embed
name="first" src="first.swf" quality="high" bgcolor="#FFFFFF"□
width="700" height="450"□     type="application/x-shockwave-flash"□

pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?
P1_Prod_Version=ShockwaveFlash">□  </embed>□</object>□
```

Press OK on the 'HTML Markup' window

5. Change to the HTML tab in FrontPage.

You should now see the following HTML code inserted into your web page:

```
<!--webbot bot="HTMLMarkup" startspan -->
   <object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#vers

      id="mycalculator" width="400" height="300" align=right>
      <param name="movie" value="mycalculator.swf">
      <param name="quality" value="high">
      <param name="scale" value="noborder">
      <param name="salign" value="r">
      <param name="align" value="right">
      <param name="wmode" value="transparent">
      <param name="bgcolor" value="#FCFAFA">
```

```
        <param name="allowScriptAccess" value="sameDomain">
        <embed name="mycalculator" src="mycalculator.swf"
   quality="high" scale="noborder" salign="r" align="right" wmode="transparent" bgco

         width="400" height="300"
         type="application/x-shockwave-flash"
         pluginspage="http://www.macromedia.com/go/getflashplayer"></embed>
      </object>
   <!--webbot bot="HTMLMarkup" endspan -->
```

Note: You should not modify any of the HTML code between the <object> and </object> tags

6. Change to the Preview tab in FrontPage.

You should now see the SWiSH Movie playing inside your FrontPage web page

Trouble-shooting
The most common mistake is to not save the web page (.htm) file and Flash Player (.swf) file in the same folder. If these files are not in the same folder, or the .swf file is missing, your SWiSH Movie will not play. When you upload the web page to your web server, you should make sure that the .swf and .htm files are in the same folder on your web server as well

Using SWiSH with PowerPoint

Top Previous Next

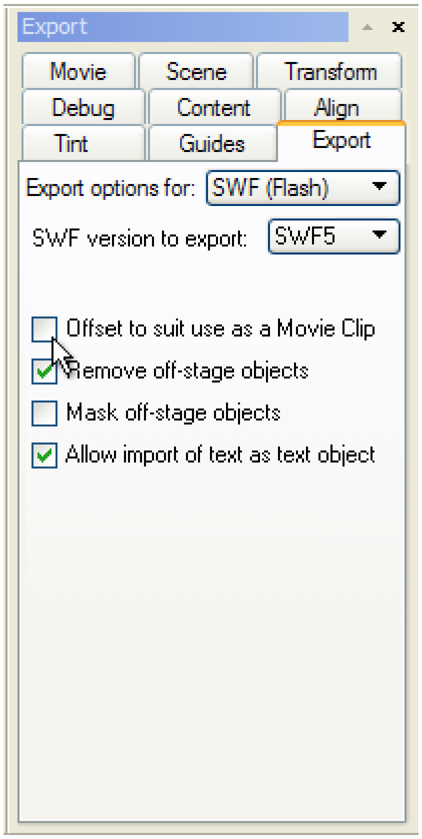
This is a step-by-step tutorial on how to use your SWiSH (.swf) Movie in a PowerPoint presentation.

Note: PowerPoint cannot embed your .swi (SWiSH working file format) Movie. You will need to export your SWiSH Movie to the Flash Player file format (.swf)

Exporting your SWF from SWiSH

1. Start with the Movie you created in the first tutorial or open "first.swi" from the Samples Menu

2. Uncheck the 'Offset Movie to suit use as a Movie Clip' option on the SWF section of the Export Panel



Export options for: SWF (Flash)

SWF version to export: SWF5

□ Offset to suit use as a Movie Clip

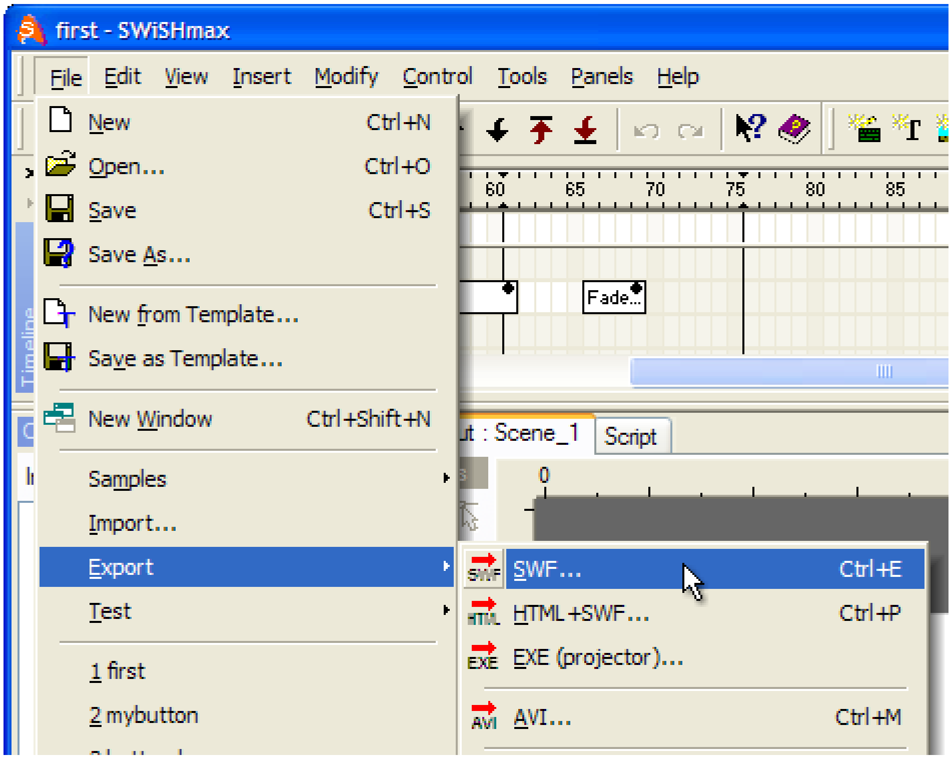
☑ Remove off-stage objects

□ Mask off-stage objects

☑ Allow import of text as text object

3. Press the 'Export to SWF' button on the Export Toolbar, or select Export | SWF from the File Menu. You should now see the 'Export to SWF' dialog box

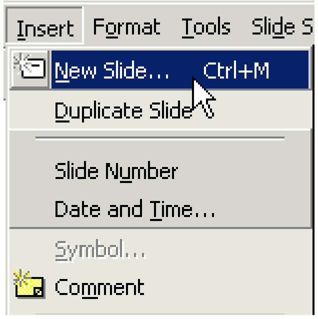
Note: If this button is not visible, select View | Toolbars. This menu will enable you to turn on the Export Toolbar



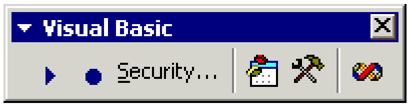
4. Choose the folder where you want to export the .swf file, enter a file name of " first.swf " (without quotes) and press the 'Save' button

To Embed a SWiSH .swf Movie into PowerPoint

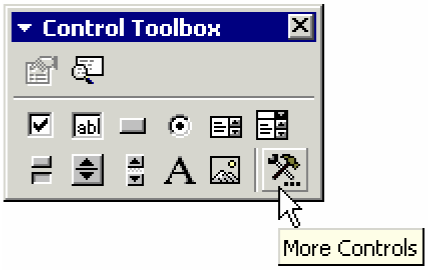
1. Open PowerPoint and and select the project where you wish to place your Movie. Create a new slide



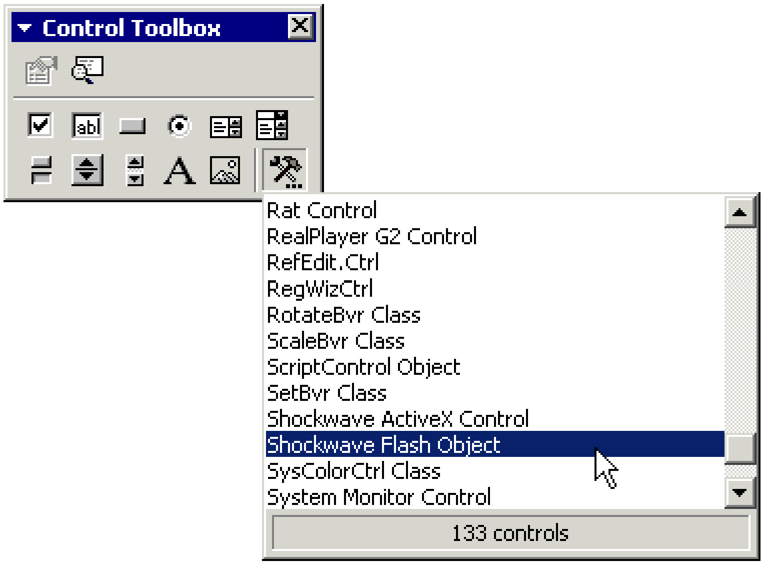
2. On your Main Menu bar select View | Toolbars | Visual basic. This will display the 'Visual Basic' Toolbar



3. Select the 'Control' Toolbox icon and open the 'Control' Toolbox



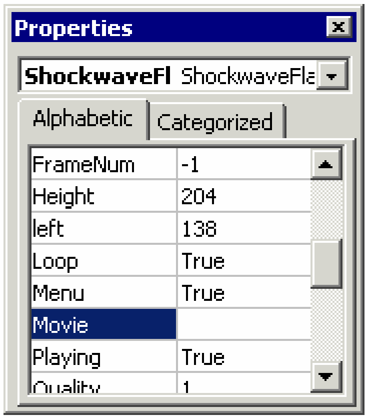
4. Select the 'More Controls' icon and scroll down the list to Shockwave Flash Object



Note: What you are doing is inserting an Active X control. You must use the Shockwave Flash Object to see your .swf Movie

5. You will notice that your mouse pointer has changed to a cross hair cursor. Draw out a rectangle on your slide (the size can be adjusted later)

6. Right-click on the rectangle you have just drawn and select Properties. This will open a list of Movie properties that you can apply



FrameNum	-1
Height	204
left	138
Loop	True
Menu	True
Movie	
Playing	True
Quality	1

7. To embed your Movie, make the following changes:

- Embed Movie: change this value to True
- Loop: change to False (if you want your .swf Movie to loop, leave this parameter as True)
- Movie: type in the absolute path of your .swf movie (c:/movie_directory/movie_name.swf)

8. Test your Movie by selecting Slide show | View show. You will see your .swf Movie playing on the slide you placed it on

9. Adjust the size to suit your requirements by clicking on the Movie to turn on the drag control points. Drag the Movie boundaries to the dimensions you require and save the presentation