

**COMSC-032**

---

**Web Site Development-  
Animate**



**Part-Time Instructor: Joenil Mistal  
Feb 13, 2014**

# Topic 10

---



10

## Adding Interactivity with *ActionScript*

You can use *ActionScript* to ask your *Animate* movie for information, and to tell your *Animate* movie what to do.

# Topics: Adding Interactivity with ActionScript

---

- Introduction to Animate Actions
- Using the Action Panel
- Assign Frame Actions
- Add Action to Movie Clips
- Jump to a Specific Frame or Scene
- Assign Stop and Play Actions
- Load a new Movie into the Current Movie
- Control Instances with Behaviors
- Link a Button to a Web Page
- Customize the Action Panels
- Add a Component



# Introduction to Animate Actions

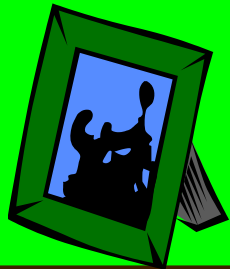
- You can add interactivity to your Animate movies by assigning an **action** or **behavior** to a frame, a button, or movie clip instance.
- Action are based on principles of **cause and effect**.



# Introduction to Animate Actions

- The occurrence that triggers the action is called an **event**.
- An **event** might be a click of button or reaching a certain frame in your movie.

**Frame**



**Button**



**Movie  
Clip**



# Introduction to Animate Actions

---

- The result of the action is the **target**, the object that is affected by the event.
- For example, a target might be linking to a Web page or playing another Animate movie

# Introduction to Animate Actions

---

## Action and ActionScripts

- Animate actions are built on a programming language called **ActionScript**.
- This scripting language allows you to write **scripting programming** code, you can certainly write your own actions in Animate.
- However, you do not need to know a scripting language to create actions.
- Animate includes hundreds of **pre-written scripts – Code Snippets**, or actions you can assign.

# Introduction to Animate Actions

---

## Using Actions in Animate

- You can use the Actions panel to add actions to **frames, buttons** or **movie clips** -mini movies within the main movie.
- You can also assign any of the built-in actions found in the **Behaviors** panel.

# Introduction to Animate Actions

---

## Using Actions in Animate

- When you assign an action, Animate adds it to a list of actions for that particular frame or button.
- This list is called an **action list** or script. Animate then executes the actions in the list based on the order in which they appear.

# Introduction to Animate Actions

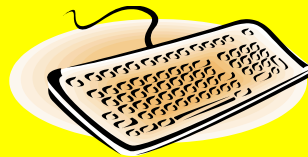
## Events

- Anything that causes an action is called an **event**.
- In Animate terminology, an event **triggers** an action in your movie.

**Mouse**



**Keyboard**



**Frame**



**Clip**



# Introduction to Animate

## Buttons

---

### Events

Animate recognizes several types of events:

- **Mouse** event occurs when you a user interacts with a button.
- **Keyboard** events occur when a user presses a keyboard key.
- **Frame** events are placed in keyframes in your movie.
- **Clip** events control movie clips

# Introduction to Animate Buttons

---

## Targets

- A target is the **object affected** by the action.
- Targets are directed toward the current movie (called, the default target), other movies (called the Tell Target) or a browser application (called an external target)

# Introduction to Animate Buttons

---

## Targets

- For example, you might place a button in your movie that, when clicked, opens a Web page.
- You direct most of your frame action toward the current movie, which is the default target.

# Introduction to Animate Buttons

---

## Types of Actions

- Animate group actions into categories in the **Actions panel**.
- The most common actions include navigational actions such as: Go To, Play and Stop; browser actions, such as Get URL and Load Movie; and movie clip control actions.

# Introduction to Animate Buttons

---

## Action Script Versions.

- You can find more than one version of Action Scripts in Animate.

**ActionScript**  
**1.0**

**ActionScript**  
**2.0**

**ActionScript**  
**3.0**

# Introduction to Animate Buttons

---

## Action Script Versions.

- You can find more than one version of Action Scripts in Animate.
- **ActionScript 3.0** works best for users familiar with object-oriented programming.
- **ActionScript 2.0** is simpler than 3.0, but is slower to execute in the Animate Player

# Using the Actions Panel

---

- You can use the **Actions panel** to add actions and write your ActionScript for your Animate movies.
- Actions enable you to **add interactively** to your movies.

# Using the Actions Panel

---

The Action panel offers the following:

- **Scripting environment** in a single panel, with a script pane for assembling scripts.
- An **Action Toolbox** with pieces of code you can use to write scripts
- **Script Assist** mode to help you write scripts accurately.

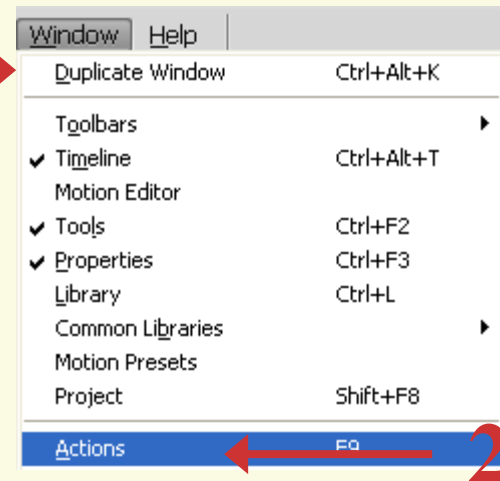
# Using the Actions Panel



## Using the Action Panel

1. Click **Window**
2. Click **Actions**

You press F9 for the Actions Panel.



# Using the Actions Panel



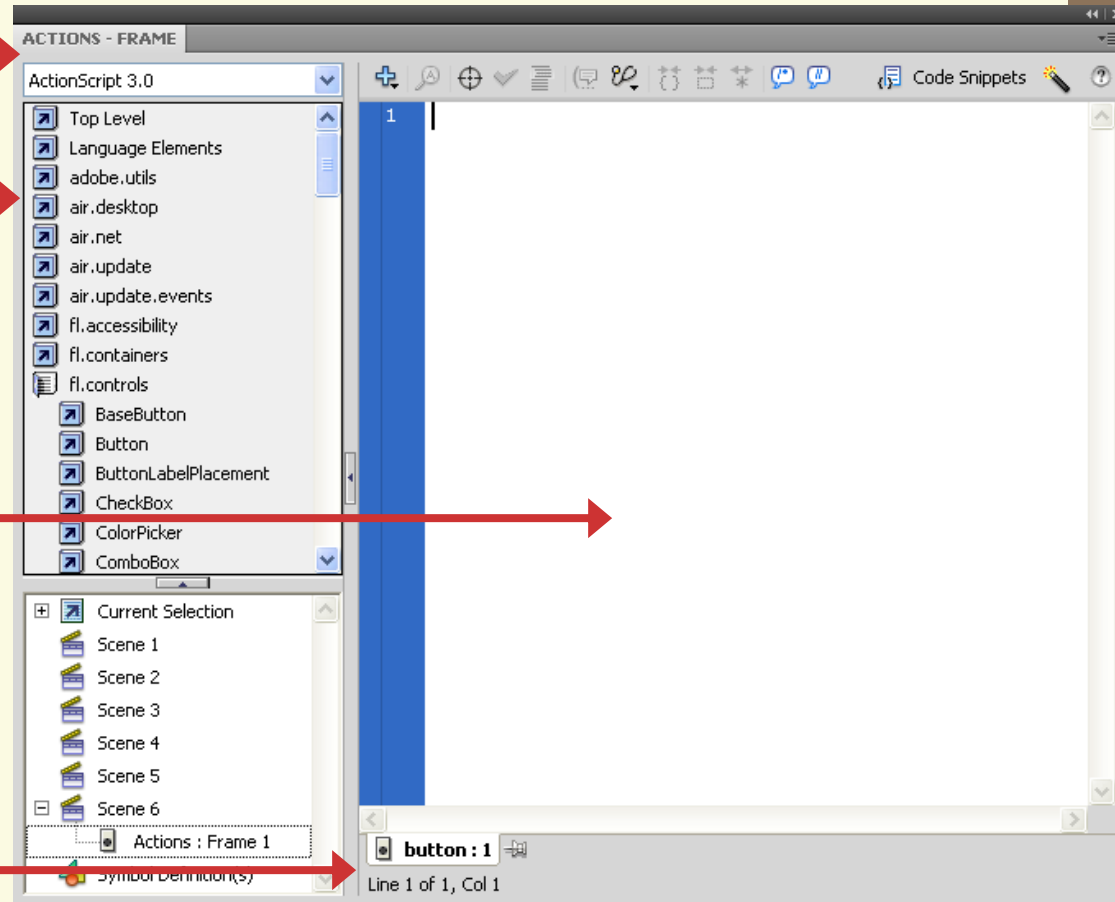
## Using the Action Panel

The Action panel opens.

**Toolbox** - Allows you to select the ActionScript version and displays a menu of all the commands available

**ActionScript Editor** - displays a visual representation of your file structure.

**Pin Active Script**- the current frame or object is listed here.



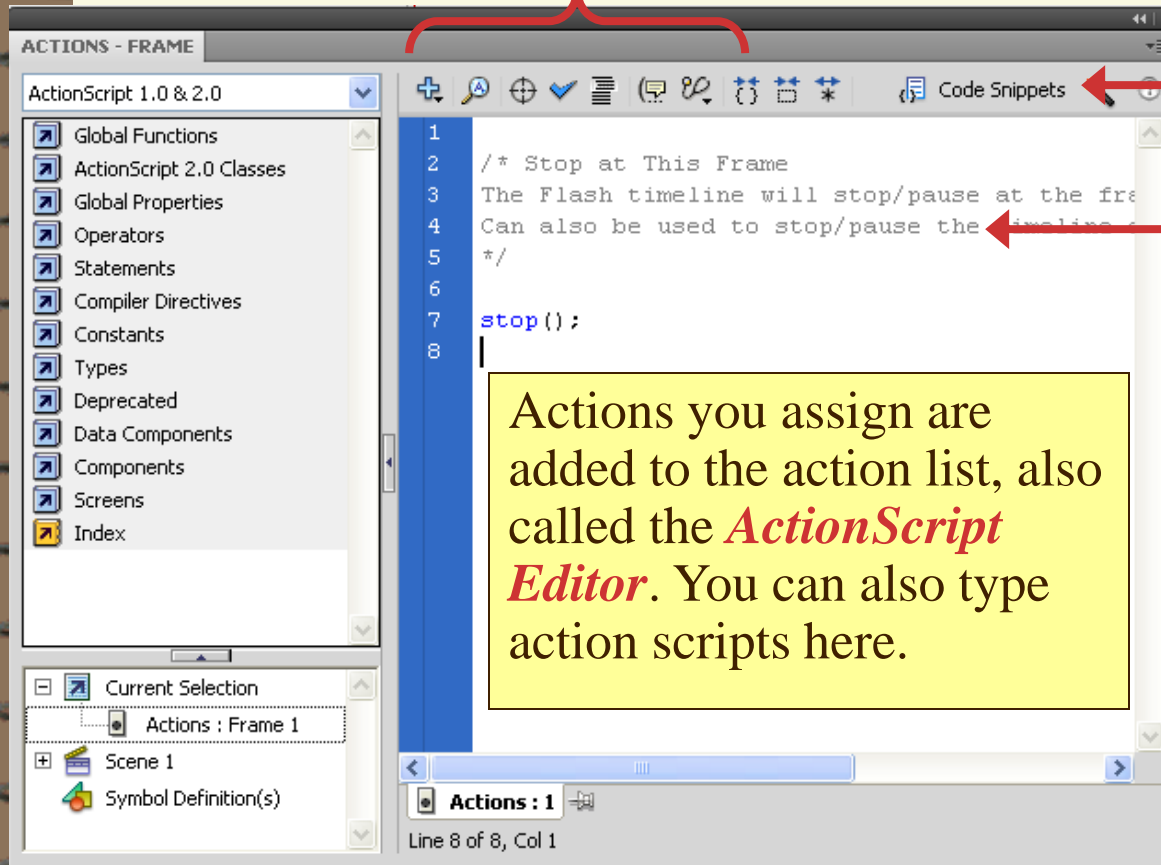
# Using the Actions Panel

You can use the buttons to help you edit and construct your ActionScript statements.

**Code Snippets-** assist tool to help you write script in Animate.

When you assign an action, a code hint tooltip appears with information about which parameters to assign.

Actions you assign are added to the action list, also called the *ActionScript Editor*. You can also type action scripts here.



# Using the Actions Panel



## Can I move the panel out of the way?

You can **move** and **resize** the Actions panel just as you can with other panels available in Animate.

You can also dock the panel.

Click the panel's title bar to quickly hide or display the panel contents.


# Assign Frame Actions

---

- You can use **frame actions** to control the main Timeline or to control a movie clip.
- Frame actions must be placed on a **keyframe**.
- For clarity, it is a best practice to reserve a layer in the Timeline only for actions and to place other elements on the other layers.

# Assign Frame Actions

---

- When you assign an action, it appears in the actions list on the right side of the **ActionScript Editor**.
- As soon as you assign an action to a frame, the frame is marked with a **tiny icon**  of the letter **a**, for action.
- After assigning an action to a frame, you can return to the Actions panel and make changes to the action as needed.

# Assign Frame Actions



## Assign Frame Actions

1. Insert a **New Layer** for the ActionScripts
2. Right-click on frame 1 and select **Insert Keyframe**.
3. Press **F9** for the ActionScript Panel
4. In the Action window, type **stop();**
5. Close the Action Panel.

Note: This save the ActionScripts

# Assign Frame Actions



## Assign Frame Actions

The screenshot shows the Adobe Animate interface. The timeline at the top is set to frame 50. A context menu is open over the timeline, with the "Insert Keyframe" option highlighted. A red arrow labeled "1" points to the "Actions" layer in the timeline, and another red arrow labeled "2" points to the "Insert Keyframe" option in the context menu. The main canvas displays a hot air balloon in a field.

- Timeline: 50
- Layers: Actions, Mask, background
- Context Menu Options:
  - Create Motion Tween
  - Create Shape Tween
  - Create Classic Tween
  - Insert Frame
  - Remove Frames
  - Insert Keyframe**
  - Insert Blank Keyframe
  - Clear Keyframe
  - Convert to Keyframes
  - Convert to Blank Keyframes
  - Cut Frames
  - Copy Frames
  - Paste Frames
  - Clear Frames
  - Select All Frames
  - Copy Motion
  - Copy Motion as ActionScript 3.0...
  - Paste Motion
  - Paste Motion Special...
  - Reverse Frames
  - Synchronize Symbols
  - Actions

Replay

Next

# Assign Frame Actions



## Assign Frame Actions

The screenshot displays the Adobe Animate workspace. At the top, the 'TIMELINE' panel shows a sequence of frames from 1 to 55. A red arrow points from the number '3' to the 'Actions' menu item in the context menu. The context menu is open, listing various actions such as 'Create Motion Tween', 'Insert Frame', 'Insert Keyframe', 'Copy Frames', and 'Actions'. The 'Actions' menu item is highlighted in blue. The main workspace shows a scene with a hot air balloon over a field. At the bottom, there are two green buttons labeled 'Replay' and 'Next'.

# Assign Frame Actions



## Assign Frame Actions

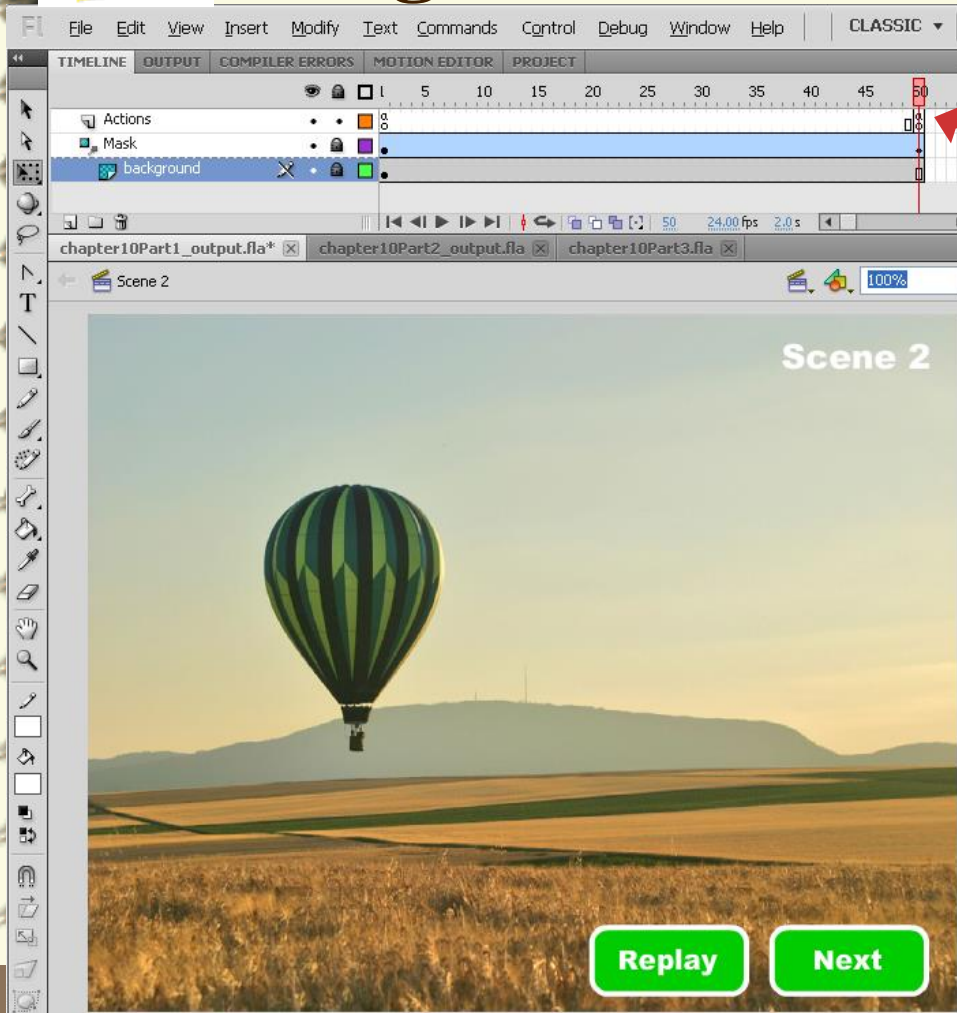
The screenshot shows the 'ACTIONS - FRAME' panel in an animation software. The left sidebar contains a tree view of the timeline, with 'Actions : Frame 50' selected. The main workspace shows the code for the selected frame: `stop();`. A red arrow labeled '4' points to the code, and another red arrow labeled '5' points to the window's title bar. The status bar at the bottom indicates 'Line 2 of 2, Col 1'.


```
1 stop();  
2
```

# Assign Frame Actions



## Assign Frame Actions



Animate also add a tiny  to indicate that an action is assigned to the frame.

# Assign Frame Actions



## Assign Frame Actions

1. Insert a **New Layer** for the ActionScripts
2. Right-click on frame 75 and select **Insert Keyframe**.
3. Press **F9** or right-click your new **keyframe** and select **Actions**
4. In the Action window, type **stop();**
5. Close the Action Panel.  
Note: This save the ActionScripts

# Assign Frame Actions



## Assign Frame Actions

1. Insert a **New Layer** for the ActionScripts
2. Right-click on frame 50 and select **Insert Keyframe**.
3. Right-click your new **keyframe** and select **Actions**
4. In the Action window, type **stop();**
5. **Close** the Action Panel.

Note: This save the ActionScripts

# Assign Frame Actions



## Assign Frame Actions

1



The screenshot shows the software interface with the following elements:

- Timeline:** A horizontal axis at the top with frame numbers from 5 to 50.
- Actions Panel:** A vertical panel on the left containing a list of elements: "Actions", "button", "bear flying", and "background". A red arrow points to this panel.
- Context Menu:** A menu is open over the "Actions" panel, listing options such as "Create Motion Tween", "Insert Keyframe", "Copy Frames", and "Reverse Frames". A red arrow points to the "Insert Keyframe" option.
- Scene 3:** The main workspace displays a scene with a purple and blue gradient background and a bear holding a red balloon.
- Buttons:** A green "Next" button is located in the bottom right corner.

2



# Assign Frame Actions



## Assign Frame Actions

The screenshot displays the 'ACTIONS - FRAME' panel in an animation software. The left sidebar shows a hierarchy of elements, including 'Current Selection' with 'Actions : Frame 1' selected. The main workspace shows a code editor with the following content:

```
1 stop();  
2  
3  
4 |
```

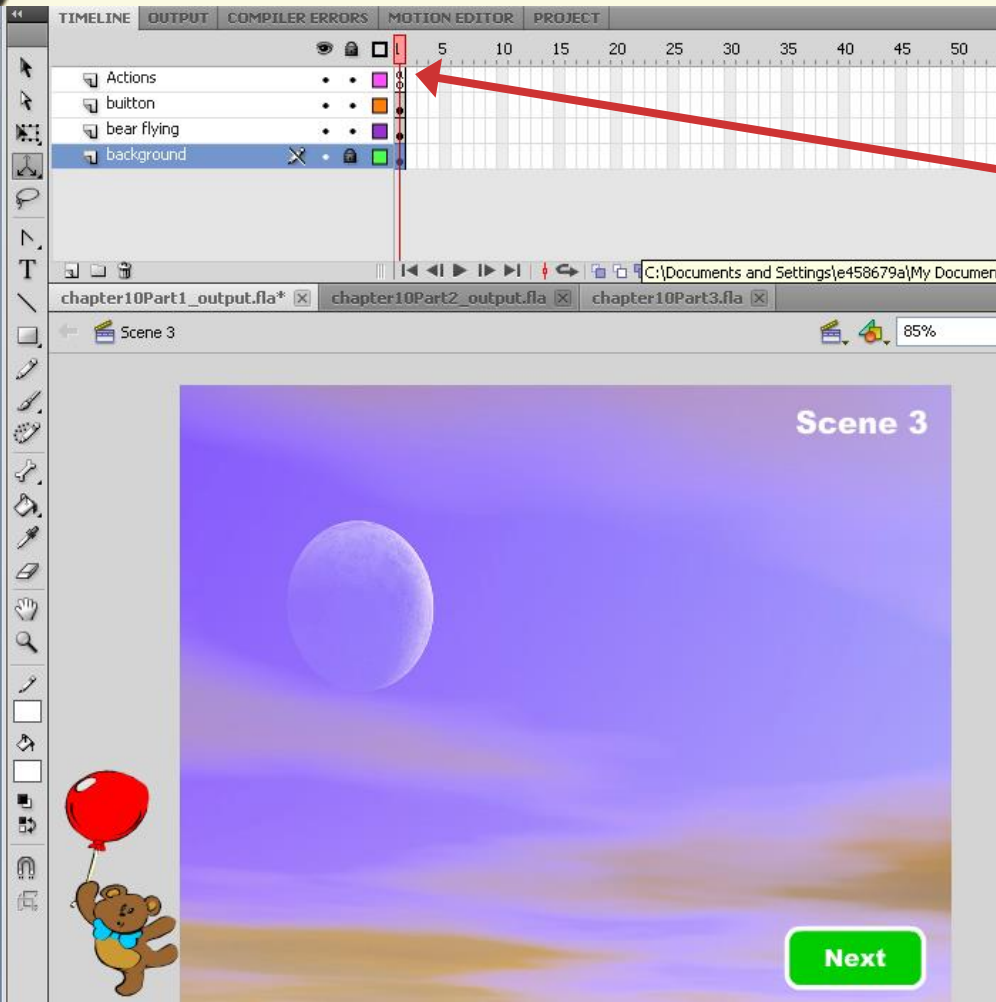
Red annotations highlight key elements: a red arrow labeled '4' points to the 'stop()' code on line 1, and another red arrow labeled '5' points to the top-right corner of the panel's title bar.


At the bottom of the panel, the status bar indicates 'Actions : 1' and 'Line 4 of 4, Col 1'.

# Assign Frame Actions



## Assign Frame Actions



Animate also add a tiny  to indicate that an action is assigned to the frame.

# Assign Frame Actions



## Assign Frame Actions

The screenshot shows the Adobe Animate software interface. The top panel includes tabs for TIMELINE, OUTPUT, COMPILER ERRORS, MOTION EDITOR, and PROJECT. The timeline shows a sequence of frames from 100 to 145. A context menu is open over the timeline, listing various actions such as "Create Motion Tween", "Insert Keyframe", and "Copy Frames". The "Insert Keyframe" option is highlighted. Below the timeline, the main stage area displays a scene with a blue car, a cow, and a tree. A "play" button is visible in the bottom right corner of the stage area.

2

Flash Animation Example

# Assign Frame Actions



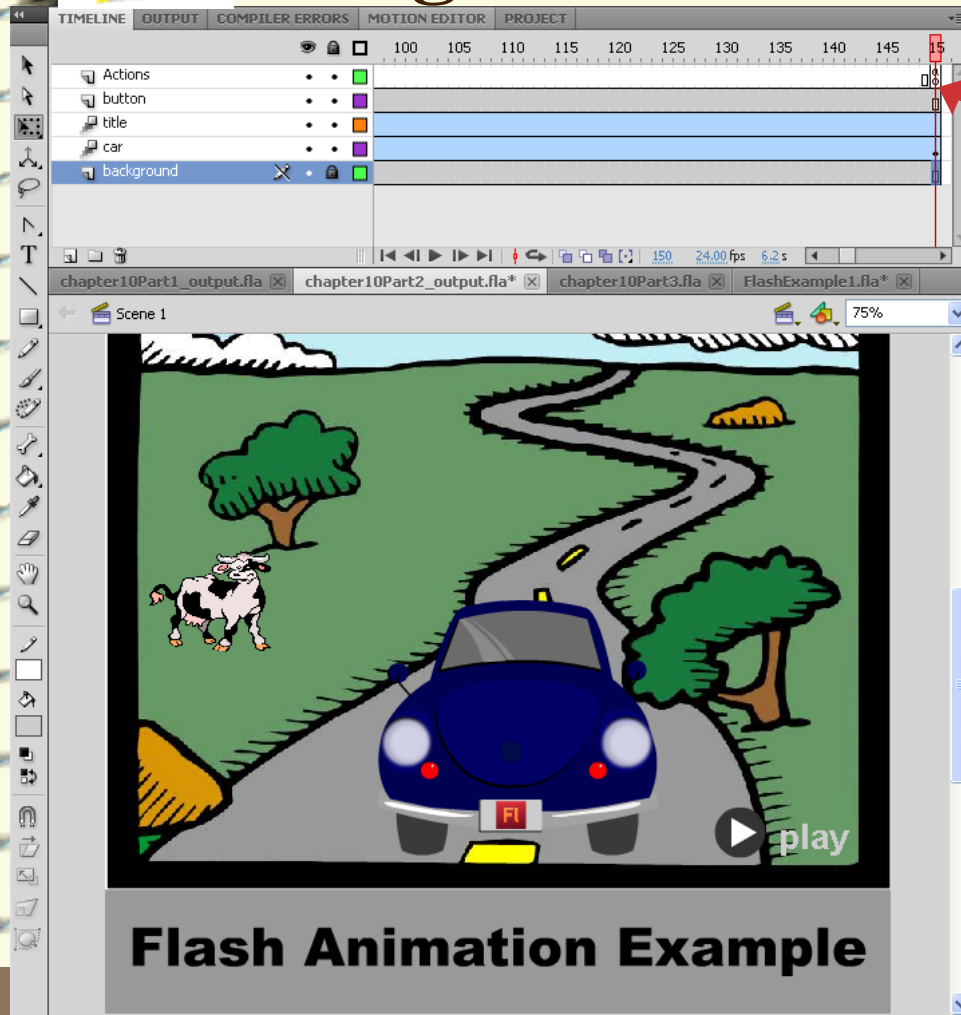
## Assign Frame Actions


The screenshot shows an IDE window titled "ACTIONS - FRAME". The left sidebar contains a "Library" pane with "ActionScript 3.0" selected, listing various classes like "Top Level", "Language Elements", and "air.update". Below it is a "Timeline" pane showing "Current Selection" with "Actions : Frame 150" and "Scene 1" with "Actions : Frame 1" and "Actions : Frame 150". The main editor area shows a single line of code: `1 stop();`. A red arrow labeled "4" points to the semicolon at the end of the code. Another red arrow labeled "5" points to the close button in the top right corner of the IDE window. The status bar at the bottom indicates "Line 1 of 1, Col 8".

# Assign Frame Actions



## Assign Frame Actions



Animate also add a tiny  to indicate that an action is assigned to the frame.

**Flash Animation Example**

# Assign Frame Actions



## Assign Frame Actions

1. Insert a **New Layer** for the ActionScripts
2. Right-click on frame 75 and select **Insert Keyframe**.
3. Press **F9** or right-click your new **keyframe** and select **Actions**
4. In the Action window, type **stop();**
5. Close the Action Panel.  
Note: This save the ActionScripts

# Assign Frame Actions



## Assign Frame Actions

The screenshot displays the Adobe Animate software interface. At the top, the menu bar includes File, Edit, View, Insert, Modify, Text, Commands, Control, Debug, Window, and Help. The main workspace is divided into several panels: TIMELINE, OUTPUT, COMPILER ERRORS, MOTION EDITOR, and PROJECT. The TIMELINE panel shows a sequence of frames from 5 to 85, with frame 75 selected. A red arrow labeled '2' points to frame 75. The MOTION EDITOR panel shows a list of layers: Actions, buttons, and background. A red arrow labeled '1' points to the 'Actions' layer. A context menu is open over frame 75, listing various actions such as 'Create Motion Tween', 'Insert Frame', 'Insert Keyframe', 'Copy Frames', and 'Actions'. A red arrow labeled '3' points to the 'Actions' option at the bottom of the menu. The main stage area shows a scene titled 'Scene 4' with a video of the Golden Gate Bridge at night. The text 'Golden Gate Bridge' is overlaid on the video in yellow. The bottom of the interface shows a toolbar with various tools and a color palette.

# Assign Frame Actions



## Assign Frame Actions

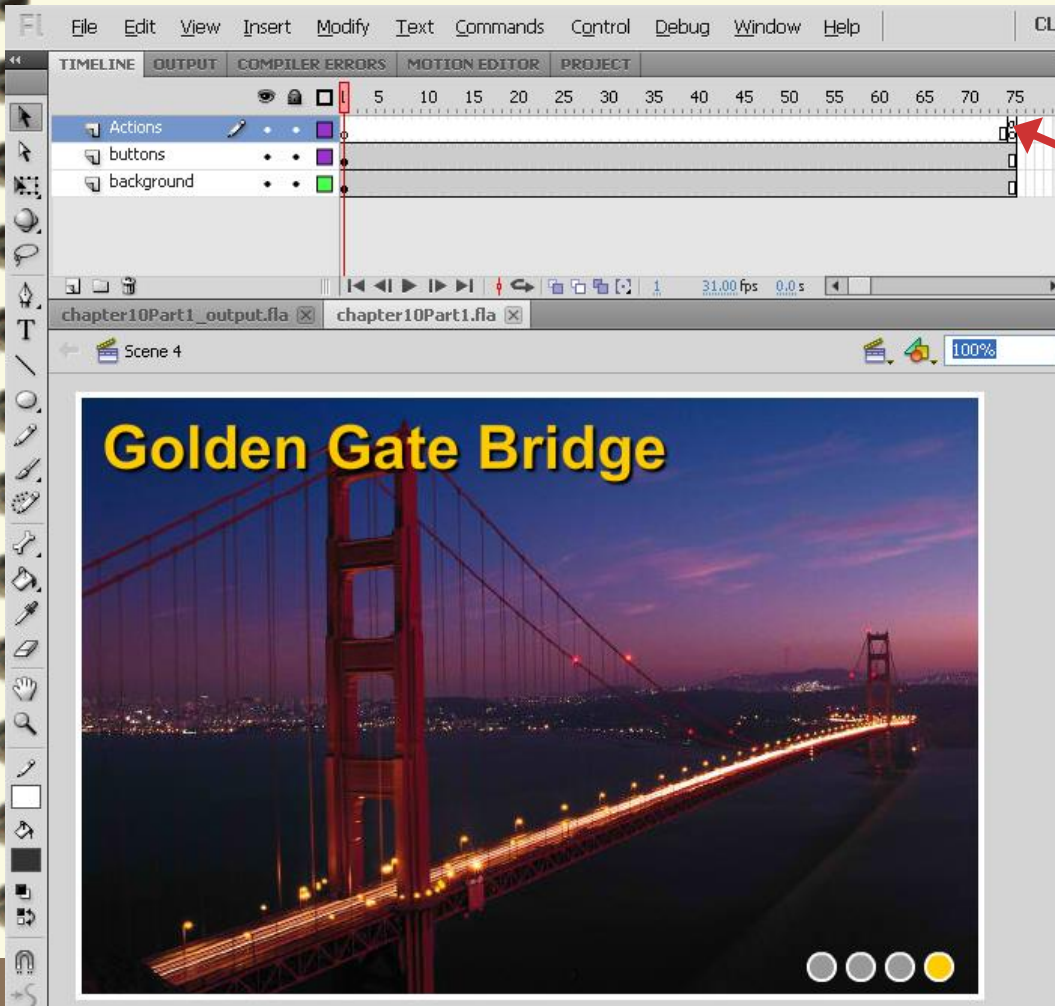
The screenshot shows the 'ACTIONS - FRAME' panel in an animation software. The panel is titled 'ACTIONS - FRAME' and shows the 'ActionScript 3.0' library on the left. The main area contains a code editor with two lines of code: '1 stop();' and '2'. A red arrow labeled '4' points to the 'stop();' code. Another red arrow labeled '5' points to the right side of the panel's title bar. The bottom status bar shows 'Actions : 75' and 'Line 2 of 2, Col 1'.


```
1 stop();
2
```

# Assign Frame Actions



## Assign Frame Actions



Animate also add a tiny  to indicate that an action is assigned to the frame.

# Assign Frame Actions



## How do organize actions in my movie?

To help you clearly identify actions you assign to frames, consider **creating a layer** specifically for actions in your movie.

This technique simplifies the process of finding the action you want to edit.

# Assign Frame Actions



**How can I edit my frame actions when the play head is on a different frame?**

You can click the Pin Active Script button in the Actions panel. Doing so creates a tab in the Actions panel for the current script, which is accessible no matter what frame of the Timeline the play head is on.

# Add a gotoAndPlay() Action to the Timeline

---

- You can add actions to a movie clip's Timeline to control how it plays.
- You can make a movie clip animation play repeatedly using the gotoAndPlay() action, creating a looping effect..

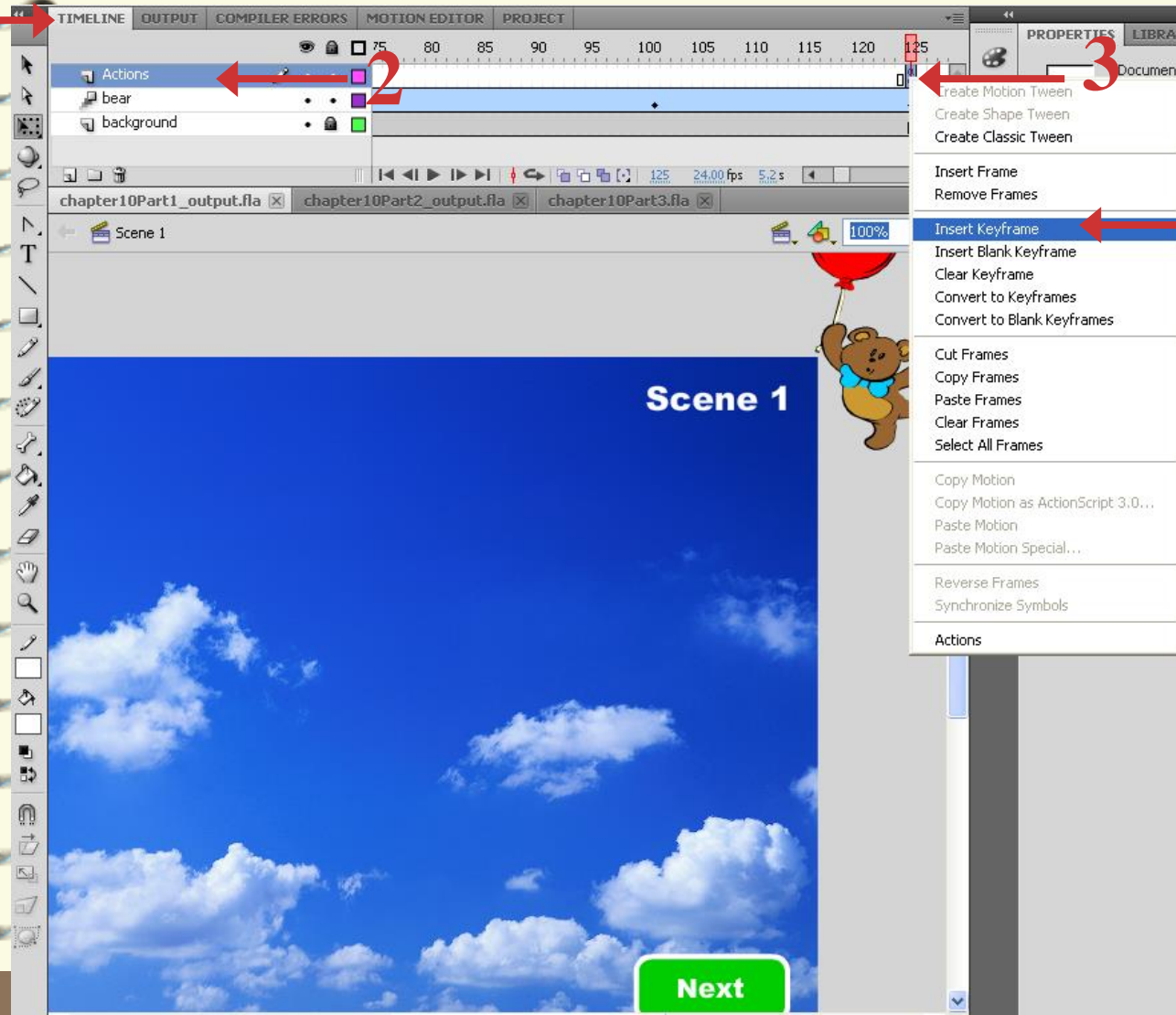
A yellow rectangular sticker with the word "demo" written in red, tilted slightly to the right.

# Add a gotoAndPlay() Action to the Timeline

1. Open the **Timeline**.
2. Insert a **New Layer** for the ActionScript.
3. Right-click on frame 125
4. Select **Insert Keyframe**.
5. After inserting the keyframe, press **F9** for the ActionPanel.
6. In the Actions panel, type *gotoAndPlay(1);*
7. Close the Action panel.  
Note: This save the ActionScripts

demo

# Add a gotoAndPlay() Action to the Timeline





# Add a gotoAndPlay() Action to the Timeline

The screenshot shows the 'ACTIONS - FRAME' panel in Adobe Animate. The left sidebar contains a tree view of the timeline structure, including 'Scene 1' with 'Actions : Frame 1' and 'Actions : Frame 125', and 'Scene 2', 'Scene 3', and 'Symbol Definition(s)'. The main code editor area shows the following code:

```
1 gotoAndPlay(1)
2 |
```

Red annotations highlight key elements: a red arrow labeled '6' points to the code 'gotoAndPlay(1)', and a red arrow labeled '7' points to the window title bar 'ACTIONS - FRAME'.

At the bottom of the panel, the status bar indicates 'Actions : 125' and 'Line 2 of 2, Col 1'.

# Add a gotoAndPlay() Action to the Timeline



**Can I preview my movie's frame actions without publishing or exporting?**

Yes. Click Control and then Enable Simple Frame Actions. Animate allows certain actions like play(), stop(), and gotoAndPlay() to work when you play through the Timeline.

# Working with Instances in ActionScript

---

- Any instances of a button, movie clip or other object with an instance name can have properties.
- These properties can be read and modified.
- Instances can also have functions that can be called that affect their behavior.

# Working with Instances in ActionScript

## Instance Names and Target Paths

- Any instances that you want to be able to talk to through ActionScript needs an instance name.
- This name allows you to target the instance you want.
- You need to use the full target path to talk to an instance that is nested inside other objects.

# Working with Instances in ActionScript

## Instance Names and Target Paths

- For example, from the main Timeline, to target a movie clip with the instance name “ball” on them main Timeline, you would use the following:

*this.ball*

- To target a movie clip named ball that is inside an object named field that is inside an object named stadium, you would use the path:

*this.stadium.field.ball*

# Working with Instances in ActionScript

## Relative Target Paths in Animate

- Animate reserves three keywords to make it easier to access your movie clips and objects in ActionScript 3:  
*this, parent and root*
- The keyword refers to the Timeline you are currently working on.
- The parent keyword refers to the object that your current objects is inside.

# Working with Instances in ActionScript

## Relative Target Paths in Animate

- For example, if you are writing a script on the Timeline of ball in the previous example, and you want to reference field, your target path would be the following:  
*this.parent*
- The root keyword always refers to the main Timeline of your Animate movie.
- So, to reference field another way from the Timeline of ball, you could use the following path:  
*root.stadium.field*

# Working with Instances in ActionScript

---

## Getting Instance Properties

- You can ask an object what the values of certain properties are in ActionScript.
- To find the x position of the ball movie clip from the previous example, you could use the following syntax:

```
var x_position = this.stadium.field.ball.x;  
trace(x_position);
```

# Working with Instances in ActionScript

---

## Setting Instances Properties

- You can also set instance properties through ActionScript
- To set the alpha of the ball movie clip to 50 percent, you can assign the value of the alpha property to 0.5 as showing the following:

*this.stadium.filed.ball.alph= 0.5;*

# Working with Instances in ActionScript

---

## Calling Instance Methods

- An instance method is a function that specifies a behavior that your object can have.
- Movie clip objects have instance methods like `play ()`, `stop()`, `gotoAndPlay()`, `gotoAndStop()`, `nextFrame()`, and `prevFrame()`.

# Working with Instances in ActionScript

---

## Calling Instance Methods

- To call an instance method, you first type the target path to your object, and a dot (.), and then the method name.
- For example, to tell a movie clip with the instance name ball to go to its next frame, you would use the following script.

```
this.ball.nextFrame();
```

# Working with Instances in ActionScript

---



## Naming an instance

1. Click on the **symbol** on the stage
2. Select the **Properties** panel
3. Click on **instance name** box
4. Enter a **unique name** for the instance
5. Repeat steps 1 to 3 for all the symbols that will be used in ActionScript

# Working with Instances in ActionScript



## Naming an instance for button symbol

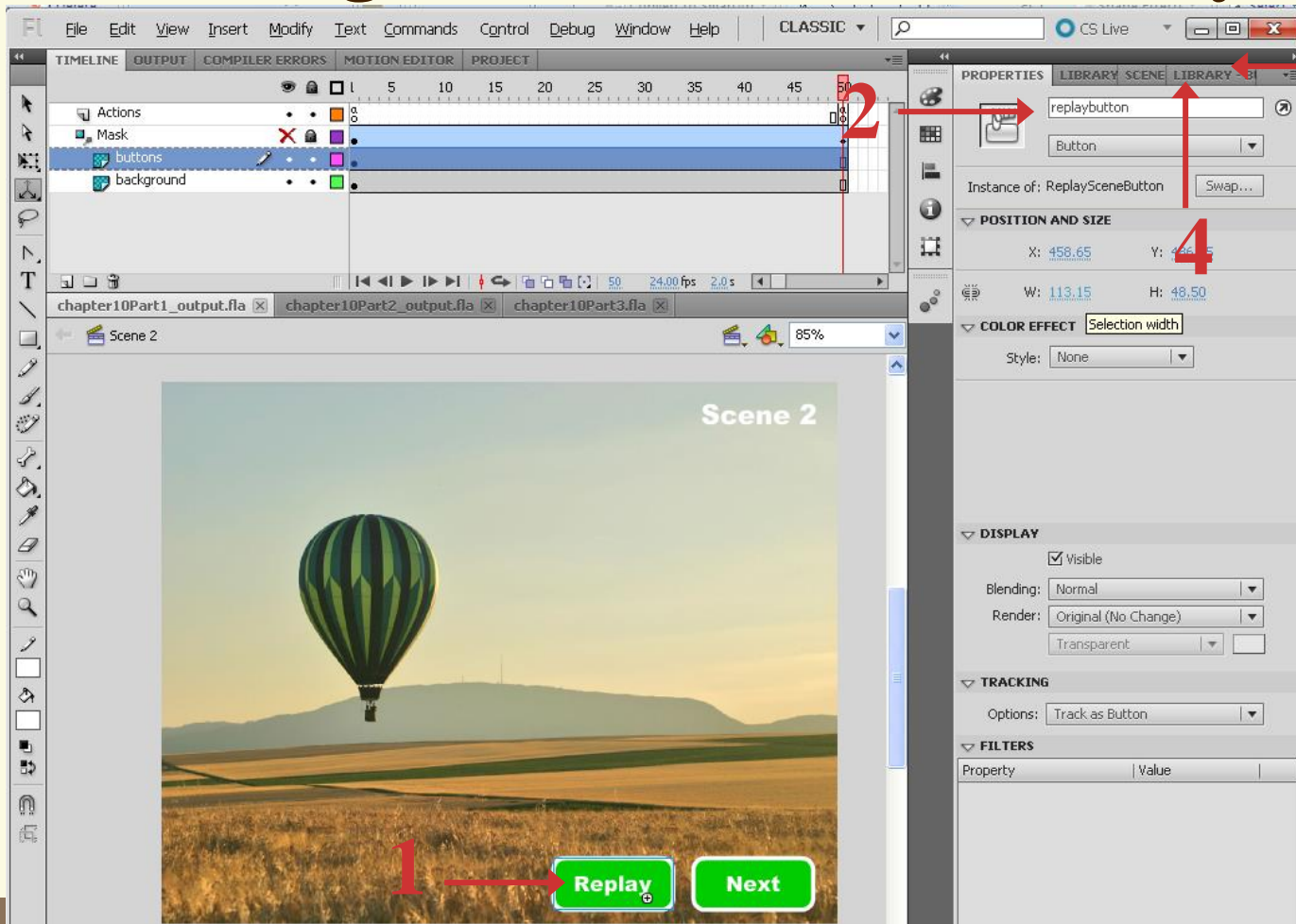
The screenshot displays the Adobe Flash CS4 interface. The main stage shows a scene with a blue sky and white clouds. A green button labeled "Next" is positioned in the bottom right corner. A red arrow labeled "1" points to this button. The timeline at the top shows a sequence of frames with a red arrow labeled "2" pointing to the Properties panel. The Properties panel on the right shows the instance name "nextbutton" in the top field, with a red arrow labeled "3" pointing to it. Below this, the "Instance of:" dropdown menu is set to "NextSceneButton", with a red arrow labeled "4" pointing to it. The scene name "Scene 1" is visible in the top right of the stage area.



# Working with Instances in ActionScript



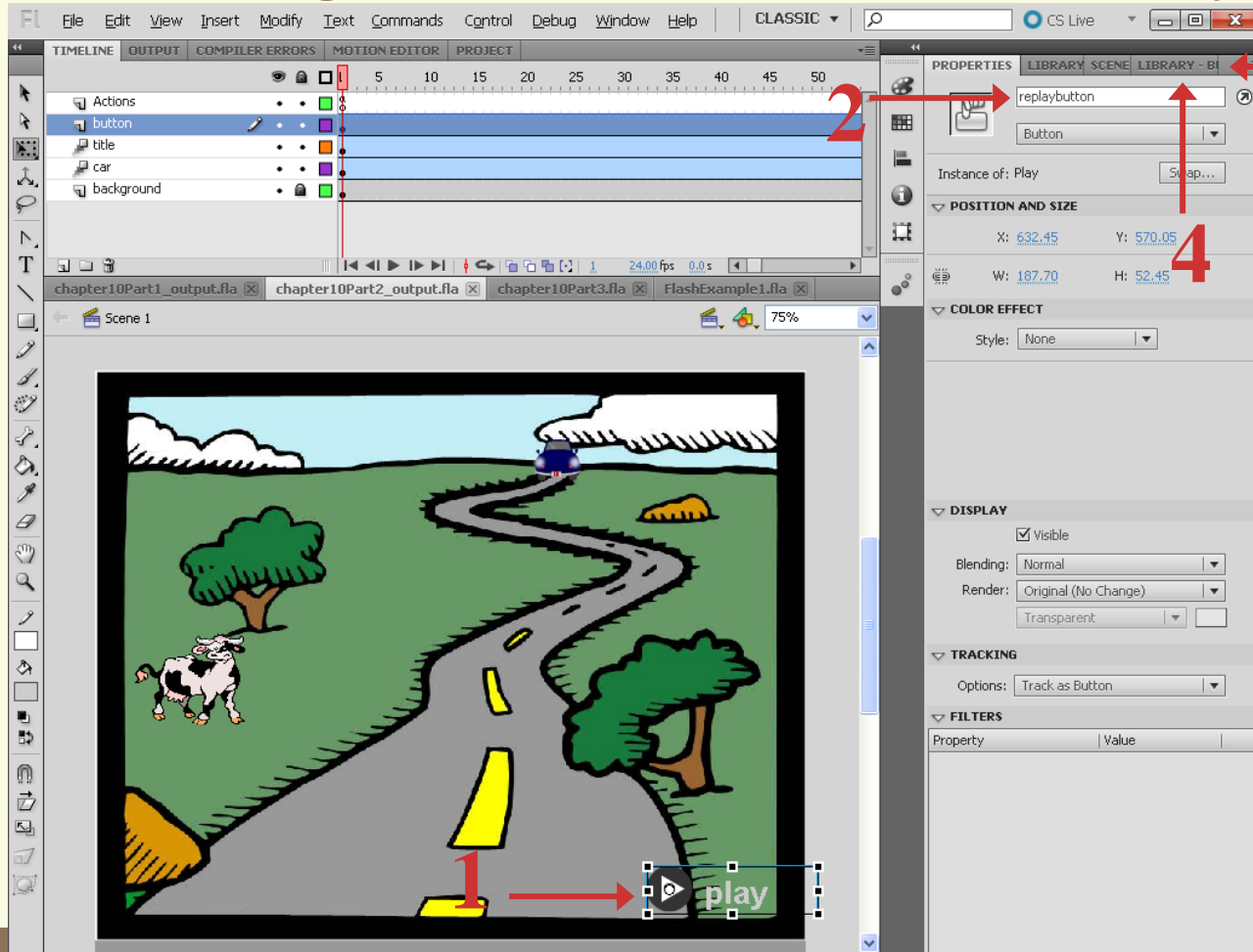
## Naming an instance for button symbol



# Working with Instances in ActionScript



## Naming an instance for button symbol



3

2

4

# Working with Instances in ActionScript



## Naming an instance for button symbol

The screenshot shows the Adobe Animate workspace. The timeline at the top displays two layers: 'buttons' and 'background'. The main canvas shows a scene with a photograph of the Golden Gate Bridge and the text 'Golden Gate Bridge' in yellow. A red arrow labeled '1' points to the 'Add New Symbol Instance' button (a yellow circle with a plus sign) in the bottom right corner of the canvas. The Properties panel on the right is open to the 'LIBRARY' tab. A red arrow labeled '2' points to the 'Instance of:' dropdown menu, which currently shows 'buttonActiveScene'. A red arrow labeled '3' points to the text input field next to it, which contains the name 'button1'. A red arrow labeled '4' points to the 'Button' symbol in the library list below the dropdown.

# Working with Instances in ActionScript



## Naming an instance for button symbol

The screenshot shows the Adobe Animate interface. The timeline at the top shows two layers: 'buttons' and 'background'. The scene view displays a photograph of the Golden Gate Bridge with the text 'Golden Gate Bridge' overlaid in yellow. In the bottom right corner of the scene view, there are four circular color swatches; a red arrow labeled '1' points to the blue swatch. The Properties panel on the right is open to the 'LIBRARY' tab. A red arrow labeled '2' points to the top of the panel. A red arrow labeled '3' points to the instance name field, which contains 'button2'. A red arrow labeled '4' points to the 'button' symbol in the library dropdown menu. The Properties panel also shows the instance name 'button2' in the 'Instance of:' field, and various other settings like position, size, color effect, display, tracking, and filters.

# Working with Instances in ActionScript



## Naming an instance for button symbol

The screenshot shows the Adobe Animate interface. The timeline at the top displays two layers: 'buttons' and 'background'. The scene view shows a Golden Gate Bridge image with the text 'Golden Gate Bridge' overlaid. The Properties panel on the right is open, showing the instance name 'button3' in the 'NAME' field. The instance is identified as 'buttonNextScene'. The 'POSITION AND SIZE' section shows X: 534.95, Y: 372.95, W: 18.00, and H: 18.00. The 'COLOR EFFECT' section shows 'Style: None'. The 'DISPLAY' section shows 'Visible' checked, 'Blending: Normal', and 'Render: Original (No Change)'. The 'TRACKING' section shows 'Options: Track as Button'. The 'FILTERS' section is empty.

Red arrows indicate the following steps:

1. Click on the instance in the scene view.
2. Click on the instance name field in the Properties panel.
3. Type the name 'button3' into the instance name field.
4. Click on the instance name field in the Properties panel.

# Working with Instances in ActionScript



## Naming an instance for a button symbol

The screenshot shows the Adobe Animate software interface. The timeline at the top displays two layers: 'buttons' and 'background'. The scene view at the bottom shows a video of the Golden Gate Bridge with the text 'Golden Gate Bridge' overlaid in yellow. The properties panel on the right is open to the 'LIBRARY' tab, showing a list of symbols. A red arrow labeled '2' points to the 'LIBRARY' tab. A red arrow labeled '3' points to the 'button4' instance name in the library list. A red arrow labeled '4' points to the 'button' symbol in the library list. A red arrow labeled '1' points to the 'Instance of: buttonNextScene' field in the properties panel.

# Working with Instances in ActionScript



## Naming an instance for a movie clip

The screenshot displays the Adobe Animate software interface. The main workspace shows a scene with a background image of the Golden Gate Bridge and the text "Golden Gate Bridge" overlaid. The timeline at the top shows a movie clip instance named "photo" starting at frame 5. The Properties panel on the right is open, showing the instance name "photo" in the top field, which is highlighted with a red arrow labeled "3". Below this, the instance name "Instance of: GGBridge1" is visible. The Properties panel also shows various settings for the instance, including Position and Size (X: 2.00, Y: 2.00, W: 600.00, H: 400.00) and 3D Position and View (X: 2.0, Y: 2.0, Z: 0.0, W: 600.0, H: 400.0). A red arrow labeled "4" points to the Position and Size section. A red arrow labeled "2" points to the timeline, and a red arrow labeled "1" points to the main workspace area.

# Move a MovieClip with ActionScript

---

- You can animate movie clip instances by listening for an ENTER\_FRAME event.
- Every time Animate renders a frame, it broadcasts a message to every object that is listening.
- At 30 frames per second (fps), you can reposition a movie clip 30 times a second, create a smooth animation.



# Move a MovieClip with ActionScript

---

1. Open the Library Panel
2. Drag a MovieClip symbol from the Library to the Stage
3. Click the movie clip you want to animate with the Selection tool
4. Open the Properties panel
5. Click here and type an instance name for your movie clip.



# Move a MovieClip with ActionScript

---

6. Press F9 for the Action Panel.
7. Click the Target icon.
8. Click your movie clip to select it
9. Click OK
10. Click the Add new item to the script button.
11. Click Animate.events
12. Click EventDispatcher
13. Click Methods
14. Click addEventListener

# Move a MovieClip with ActionScript



**How can I set my font and color preferences for the ActionScript window?**

Click Animate, Preferences, and then click on ActionScript in the left column.

The Preference panel has options to change your font and syntax colors.

There are also preferences that you can set for code hits, import and export file types, and language settings.

# Add Actions to Movie Clips

---

- You can add actions to **movie clip instances** that appear in your main movie
- For example, you may have an animation sequence of moving car that includes movie clips for making each wheel rotate.

# Move a MovieClip with ActionScript *(continued)*

---

- To move your movie clip, you can simply change its x and/ or y properties.
- These property values are in pixels, so make sure you do not set their values to a number larger than the Stage size (or to a negative number), unless you want your movie clip to disappear.

# Move a MovieClip with ActionScript *(continued)*



## What other properties can I animate using ActionScript?

You can animate any property of a movie clip or object using ActionScript.

To view a list of DisplayObject properties, click the Animate.display category in the Action Window.

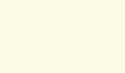
Inside Animate.display, click DisplayObject. Inside DisplayObject, click Properties. All display properties are listed inside.

With this example, you can easily change the script to move your movie clip along the y-axis. Change `evt.currentTarget.x` to `evt.currentTarget.y`. Test the movie again, and watch your movie clip move vertically rather horizontally.

# Move a MovieClip with ActionScript *(continued)*



## How do I check my script for errors?

If you click the Check Syntax button , Animate displays the Compiler Errors panel with your errors.

The Compiler Error panel also appears when you test your movie if your script contains errors.

# Fade In a Movie Clip with ActionScript

---

- You can use the ENTER\_FRAME event to perform other tasks over time in Animate.
- The following demonstrates how to gradually change the alpha property of a movie clip, creating a fade effect.



# Fade In a Movie Clip with ActionScript

---

1. Click on the instance of the **movie clip symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window**
4. Click on **Action Panel** or press F9



# Fade In a Movie Clip with ActionScript

The screenshot shows the Adobe Flash CS5.5 interface. The timeline at the top shows a movie clip named 'background' on the 'background' layer. A red arrow labeled '1' points to the movie clip on the stage. A red arrow labeled '2' points to the 'photo' instance name in the Properties panel. A red arrow labeled '3' points to the 'Window' menu, and a red arrow labeled '4' points to the 'Actions' option in the menu. The Properties panel on the right shows the 'photo' instance with a width of 600.00 and a height of 400.00. The 'COLOR EFFECT' section is set to 'None', and the 'DISPLAY' section has 'Visible' checked and 'Blending' set to 'Normal'.



# Fade In a Movie Clip with ActionScript

---

6. Click on the **Code Snippets**
7. Expand the **Animation folder**
8. Double click on “**Fade In a Movie Clip**”
9. Change the speed in which the symbol instance fade in to 0.08.
10. Close the Action Panel window.



# Fade In a Movie Clip with ActionScript

10

The screenshot shows an IDE window titled "ACTIONS - FRAME" with the following components:

- Language Panel (Left):** Shows "ActionScript 3.0" selected. A red arrow labeled "6" points to the "Code Snippets" button in the top right of this panel.
- Code Snippets Panel (Bottom Left):** Shows a tree view with folders for "Actions", "Timeline Navigation", "Animation", and "Load and Unload". The "Animation" folder is expanded, and "Fade In a Movie Clip" is selected. A red arrow labeled "7" points to the "Animation" folder, and a red arrow labeled "8" points to the "Fade In a Movie Clip" item.
- Code Editor (Center):** Contains the following ActionScript code:

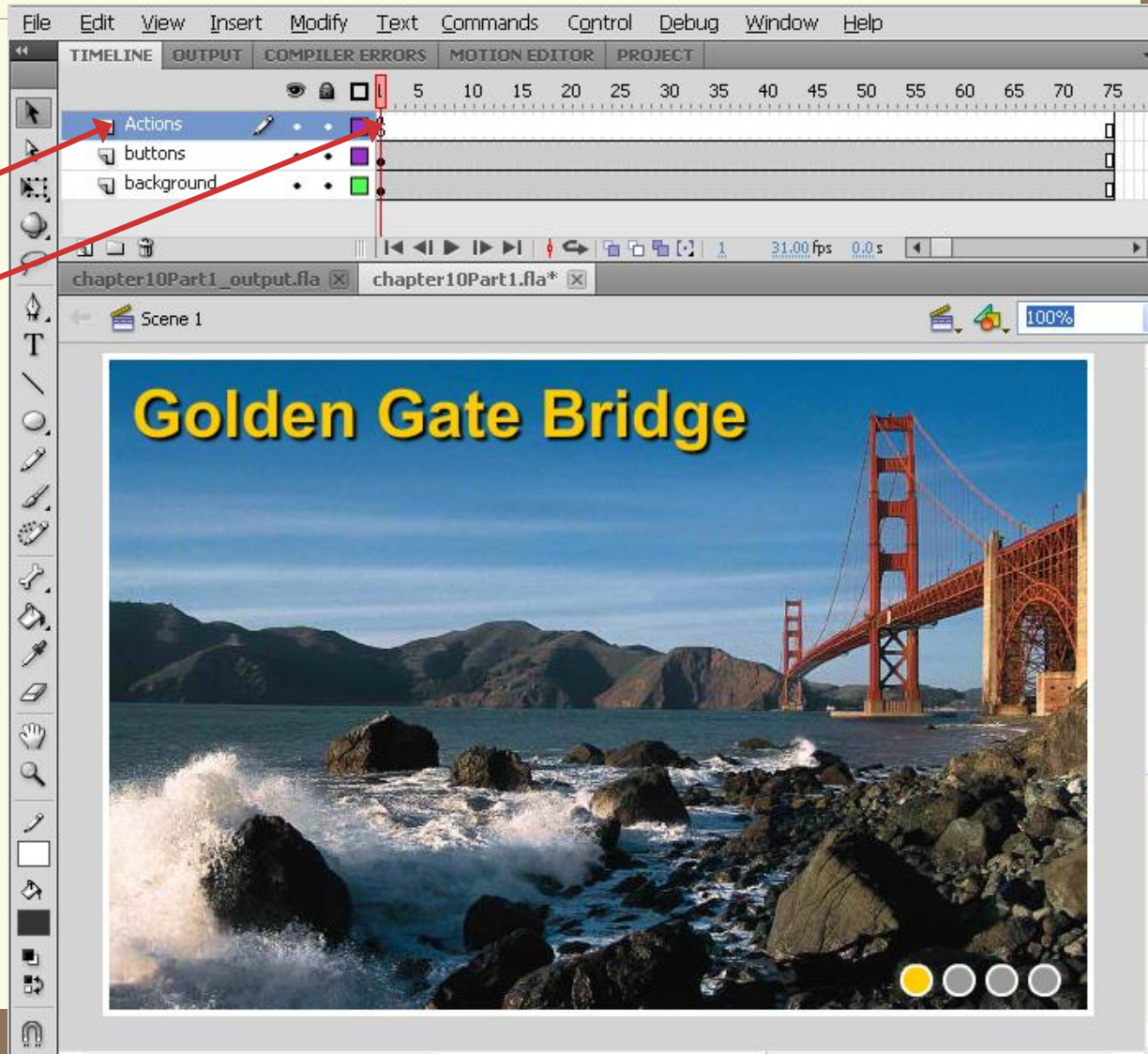
```
1  
2  /* Fade In Movie Clip  
3  Fades in the symbol instance by increasing its alpha property within an ENTER_FRAME  
4  
5  Instructions:  
6  1. To change the speed at which the symbol instance fades in, change the 0.01 val  
7  2. Because the animation uses an ENTER_FRAME event, it progresses only when the p  
8  */  
9  
10 photo.addEventListener(Event.ENTER_FRAME, fl_FadeSymbolIn);  
11 photo.alpha = 0;  
12  
13 function fl_FadeSymbolIn(event:Event)  
14 {  
15     photo.alpha += 0.08;  
16     if(photo.alpha >= 1)  
17     {  
18         photo.removeEventListener(Event.ENTER_FRAME, fl_FadeSymbolIn);  
19     }  
20 }  
21
```

A red arrow labeled "9" points to the `photo.alpha += 0.08;` line.



# Fade In a Movie Clip with ActionScript

The **Actions layer** and **action keyframe** are created automatically after the action script is entered into the Action Panel



# Fade In a Movie Clip with ActionScript



**How can I change the appearance of my scripts in the Actions panel?**

You can click Animate, Preferences, and then click on the ActionScript category to customize the Actions panel.

Then, you can change your indentation preference, code hint timing, font, style, and syntax colors.

# Fade In a Movie Clip with ActionScript *(continued)*

---

- By creating your animation in ActionScript, you can create scripts that can be used by multiple objects.
- This both saves you development time and gives your project a consistent animation style.

# Fade In a Movie Clip with ActionScript *(continued)*



**How can I test to see my if statement is working properly?**

You can place a `trace()` statement inside your if statement to see if the condition is met. In this example, you might add:

```
trace("face completed.");
```

Between the curly braces of your statement. If your condition is met, `face completed` appears in the Output window when your test movie.

# Understanding Events and Event Handlers

---

- In Animate, an Event is any type of interaction, such as a mouse click, mouse move, rollover, or key press.
- Events can also be triggered by object instances in Animate.
- For example, MoveClip object can broadcast events like ENTER\_FRAME, and Loader objects can broadcast events like COMPLETE.

# Understanding Events and Event Handlers

## About Events and Event Handlers

- A good way to think about events and event listeners is to think about the police responding to a crime, like a robbery.
- In ActionScript, your event handler is the function that performs actions when an event occurs.

# Understanding Events and Event Handlers

---

## Adding Event Listeners

- In ActionScript 3.0 you can add events to objects by calling the `addEventListener()` function like  
*`myButton.addEventListener(MouseEvent.CLICK, onButtonClick);`*
- This statement tells Animate to listen for a mouse click on an instance called `myButton`
- When that event occurs, the function `onButtonClick` is executed.

# Understanding Events and Event Handlers

## Creating Event Handlers

- The event handler is the function that is executed when an event occurs.
- In ActionScript 3.0, event handlers take an object of type Event as a parameter.
- A handler for the mouse click example might look like this:

```
Function onMouseClick(evt:Event):void { trace  
    (“my button was clicked! “);  
}
```

# Understanding Events and Event Handlers

---

## Event Targets

- There are two places you use your event target in ActionScript.
- You provide the target when you create the listener, but you can also use the target in your event handler.
- For example, if you want the myButton object to disappear when clicked, your handler might look like this:

```
Function onMouseClick(evt:Event):void {  
    myButton.visible = false;  
}
```

# Understanding Events and Event Handlers

## Event Targets

- But, if you wanted any button to become invisible when clicked, you could add the same event listener to all of them, but change your handler to look like this:

```
function onMouseClick(evt:Event):void {  
    evt.currentTarget.visible = false;  
}
```

- In this case, `evt.currentTarget` is a reference to whichever listening button is clicked.

# Understanding Events and Event Handlers

---

## Removing Event Listeners

- Sometimes you want your objects to stop listening for events.
- To do so, you remove the listener by calling `removeEventListener`.
- In the practical example, you would delete your mouse click listener like this:

```
myButton.removeEventListener(MouseEvent.  
    MOUSE_DOUT, onMouseClick);
```

# Timeline Navigation using a Button

---

- You can create interactivity in your Animate movies by combining animation with controls like buttons.
- Your buttons can listen for mouse events, and perform actions when those events occur.



# Timeline Navigation using a Button

---

## Click to Go to Next Scene and Play

1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window** and click **Action Panel** or press **F9**



# Timeline Navigation using a Button

The screenshot displays an animation software interface with several key components:

- Timeline:** Located at the top, it shows a sequence of frames from 0 to 50. A red vertical line marks frame 5. A red arrow labeled "3" points to the "Help" menu item in the top toolbar.
- Scene View:** The main workspace shows "Scene 1" with a blue sky background and white clouds. A red balloon is held by a brown bear in the bottom left. A green button labeled "Next" is positioned in the bottom right. A red arrow labeled "1" points to this button.
- Properties Panel:** On the right, the "LIBRARY" tab is active, showing a "nextbutton" asset. A red arrow labeled "2" points to this asset. Below it, the "POSITION AND SIZE" section shows X: 590.65, Y: 486.65, W: 113.15, and H: 48.50.



# Timeline Navigation using a Button

---

## Click to Go to Next Scene and Play

5. Click on the **Code Snippets**
6. Expand the **Timeline Navigation folder**
7. Double click on “**Click to Go to Next Scene and Play**”
8. Close the Action Panel window.

demo

# Timeline Navigation using a Button

9

The screenshot displays an IDE interface with two main windows. The top window, titled 'ACTIONS - FRAME', shows an 'ActionScript 3.0' code editor. The code defines an event listener for a button click that navigates to the next scene and plays. The code is as follows:

```
1  
2 /* Click to Go to Next Scene and Play  
3 Clicking on the specified symbol instance moves the playhead to the next  
4 */  
5  
6 nextbutton.addEventListener(MouseEvent.CLICK, f1_ClickToGoToNextScene);  
7  
8 function f1_ClickToGoToNextScene(event:MouseEvent):void  
9 {  
10     MovieClip(this.root).nextScene();  
11 }
```

The bottom window, titled 'CODE SNIPPETS', shows a tree view of code snippets. The 'Timeline Navigation' folder is expanded, and the snippet 'Click to Go to Next Scene and Play' is selected. The status bar at the bottom indicates 'Line 1, Column 16'.

Red arrows and numbers highlight key elements: '5' points to the 'Code Snippets' button in the top window's toolbar; '6' points to the 'Timeline Navigation' folder in the bottom window; '7' points to the selected 'Click to Go to Next Scene and Play' snippet; and '9' points to the top-right corner of the IDE window.



# Timeline Navigation using a Button

---

## Click to Go to Frame and Play

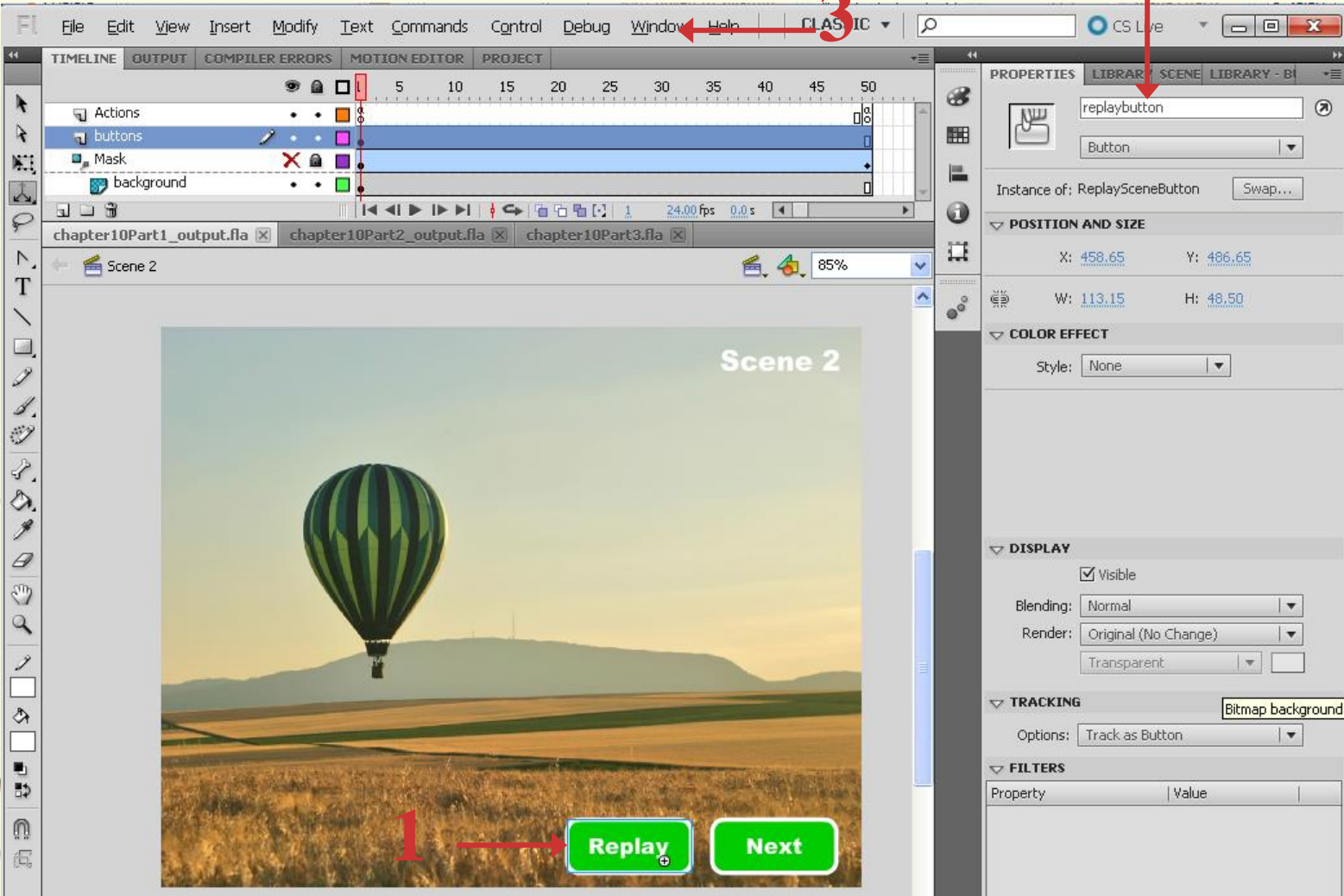
1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window** and click **Action Panel** or press **F9**

demo

# Timeline Navigation using a Button

2

3





# Timeline Navigation using a Button

---

## Click to Go to Frame and Play

4. Click on the **Code Snippets**
5. Expand the **Timeline Navigation folder**
6. Double click on “**Click to Go to Frame and Play**”
7. Replace with the number with the frame you would like to go and play in the current scene.
8. Close the Action Panel window.



# Timeline Navigation using a Button

8

The screenshot shows the 'ACTIONS - FRAME' panel in Adobe Animate. The main code editor contains the following code:

```
13  /* Click to Go to Frame and Play
14  Clicking on the specified symbol instance moves the playhead to the sp
15  Can be used on the main timeline or on movie clip timelines.
16
17  Instructions:
18  1. Replace the number 5 in the code below with the frame number you wo
19  */
20
21  replaybutton.addEventListener(MouseEvent.CLICK, f1_ClickToGoToAndPlayF:

function f1_ClickToGoToAndPlayFromFrame (event:MouseEvent):void
{
    gotoAndPlay(1);
}
```

The 'CODE SNIPPETS' panel is open, showing a list of actions under the 'Timeline Navigation' folder. The action 'Click to Go to Frame and Play' is selected. Red arrows point to various elements: arrow 4 points to the 'Code Snippets' button in the top toolbar; arrow 5 points to the 'Timeline Navigation' folder; arrow 6 points to the selected 'Click to Go to Frame and Play' snippet; arrow 7 points to the 'gotoAndPlay(1);' line in the code; and arrow 8 points to the top-right corner of the Animate window.



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window**
4. Click on **Action Panel** or press **F9**



# Timeline Navigation using a Button

2

3

1

The screenshot displays the Adobe Flash CS5.5 interface. The main workspace shows a scene titled "Scene 3" with a purple and blue gradient background, a large white sphere, and a small brown bear holding a red balloon. A green button labeled "Next" is positioned in the bottom right corner of the scene. A red arrow labeled "1" points to this button. The timeline at the top shows a red vertical line indicating the current frame position. A red arrow labeled "3" points to the "Window" menu in the top menu bar. The Properties panel on the right shows the selected "nextbutton" instance, with its "Options" set to "Track as Button". A red arrow labeled "2" points to the "Scene" tab in the Properties panel.



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

6. Click on the **Code Snippets**
7. Expand the **Timeline Navigation folder**
8. Double click on “**Click to Go to Scene and Play**”
9. Replace with the name of the scene you would like to play
10. Close the Action Panel window.



# Timeline Navigation using a Button

10

**6** → Code Snippets

**7** → Timeline Navigation

**8** → Click to Go to Scene and Play

**9** → "Scene 1"

```
1 stop();
2
3 /* Click to Go to Next Frame and Stop
4 Clicking on the specified symbol instance moves the playhead
5 */
6
7
8 /* Click to Go to Scene and Play
9 Clicking on the specified symbol instance plays the movie from
10
11 Instructions:
12 1. Replace "Scene 3" with the name of the scene you would like
13 2. Replace 1 with the frame number you would like the movie to
14 */
15
16 nextbutton.addEventListener(MouseEvent.CLICK, f1_ClickToGoToS
17
18 function f1_ClickToGoToScene_3(event:MouseEvent):void
19 {
20     MovieClip(this.root).gotoAndPlay(1, "Scene 1");
21 }
22
```



# Timeline Navigation using a Button

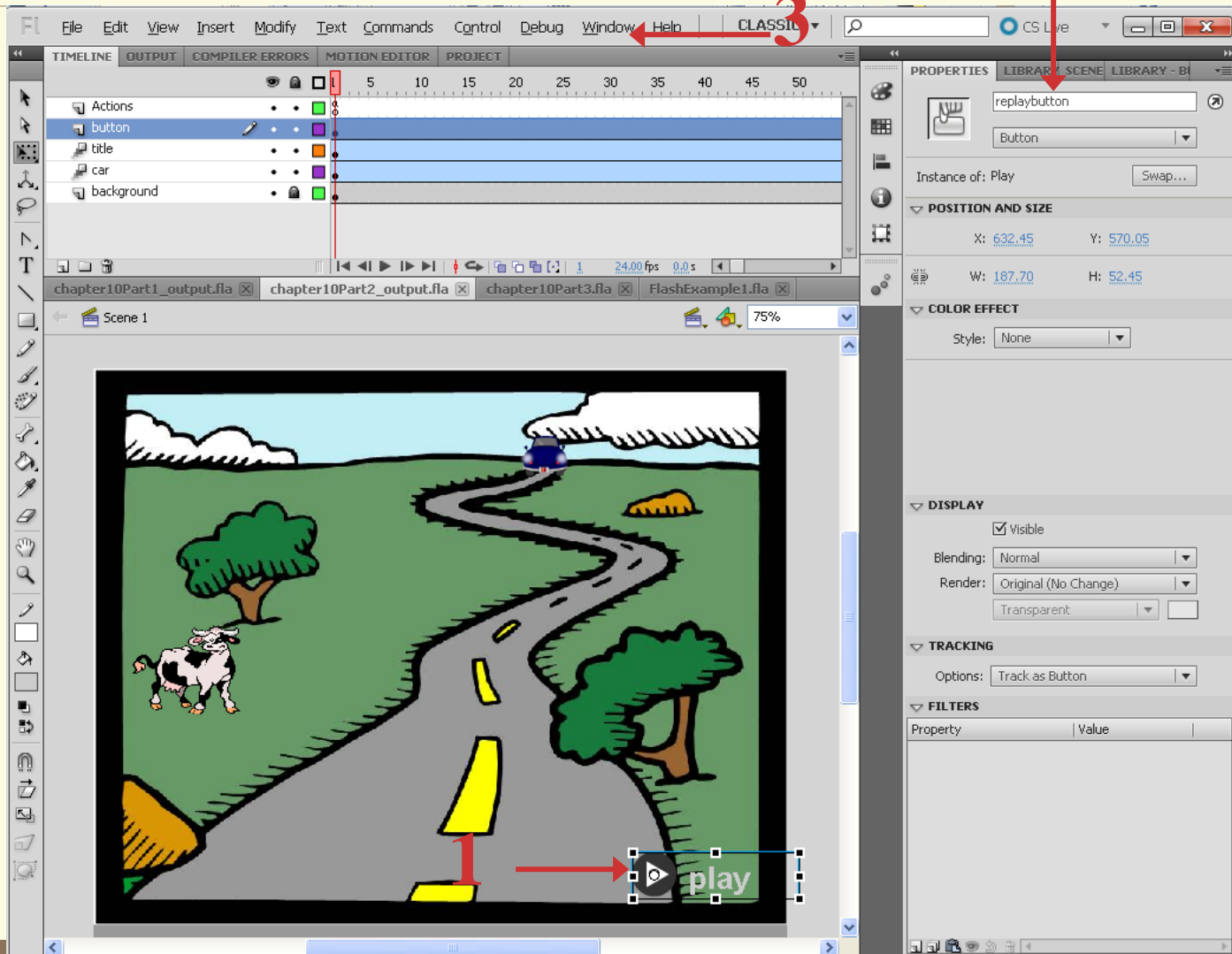
---

## Click to Go to Frame and Play

1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window** and click **Action Panel** or press **F9**

demo

# Timeline Navigation using a Button





# Timeline Navigation using a Button

---

## Click to Go to Frame and Play

4. Click on the **Code Snippets**
5. Expand the **Timeline Navigation folder**
6. Double click on “**Click to Go to Frame and Play**”
7. Replace with the number with the frame you would like to go and play in the current scene.
8. Close the Action Panel window.



# Timeline Navigation using a Button

8

**ACTIONS - FRAME**

ActionScript 3.0

- Top Level
- Language Elements
- adobe.utils
- air.desktop
- air.net
- air.update

**CODE SNIPPETS**

- Actions
- Timeline Navigation
  - Stop at this Frame
  - Click to Go to Frame and Stop
  - Click to Go to Frame and Play
  - Click to Go to Next Frame and Stop
  - Click to Go to Previous Frame and Stop
  - Click to Go to Next Scene and Play
  - Click to Go to Previous Scene and Play
  - Click to Go to Scene and Play
- Animation
- Load and Unload
- Audio and Video
- Event Handlers

```
1  
2 /* Click to Go to Frame and Play  
3 Clicking on the specified symbol instance moves the playhead  
4 Can be used on the main timeline or on movie clip timelines.  
5  
6 Instructions:  
7 1. Replace the number 5 in the code below with the frame number  
8 */  
9  
10 replaybutton.addEventListener(MouseEvent.CLICK, f1_ClickToGoToAndPlayFromFrame);  
11  
12 function f1_ClickToGoToAndPlayFromFrame(event:MouseEvent):void  
13 {  
14     gotoAndPlay(1);  
15 }  
16  
17  
18
```

Actions : 1

Line 17 of 18, Col 1



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window**
4. Click on **Action Panel** or press **F9**



# Timeline Navigation using a Button

The screenshot displays the Adobe Animate workspace. The Timeline panel at the top shows a video clip named 'Golden Gate Bridge' with a playhead at approximately 5 seconds. The 'Window' menu is open, listing various panels. The 'Actions' panel is highlighted in blue. The Properties panel on the right shows the instance name 'button1' and its position and size.

| Property          | Value             |
|-------------------|-------------------|
| Instance of:      | buttonActiveScene |
| POSITION AND SIZE |                   |
| X:                | 486.95            |
| Y:                | 372.95            |
| W:                | 18.00             |
| H:                | 18.00             |



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

6. Click on the **Code Snippets**
7. Expand the **Timeline Navigation folder**
8. Double click on “**Click to Go to Scene and Play**”
9. Replace with the name of the scene you would like to play
10. Close the Action Panel window.

demo

# Timeline Navigation using a Button

10

ACTIONS - FRAME

ActionScript 3.0

- Top Level
- Language Elements
- adobe.utils

CODE SNIPPETS

- Actions
  - Timeline Navigation
    - Stop at this Frame
    - Click to Go to Frame and Stop
    - Click to Go to Frame and Play
    - Click to Go to Next Frame and Stop
    - Click to Go to Previous Frame and Stop
    - Click to Go to Next Scene and Play
    - Click to Go to Previous Scene and Play
    - Click to Go to Scene and Play
  - Animation
  - Load and Unload
  - Audio and Video

```
21
22
23
24
25  /* Click to Go to Scene and Play
26  Clicking on the specified symbol instance plays the movie from the specified scene and frame.
27
28  Instructions:
29  1. Replace "Scene 3" with the name of the scene you would like play.
30  2. Replace 1 with the frame number you would like the movie to play from in the specified scene.
31  */
32
33  button1.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene);
34
35  function f1_ClickToGoToScene(event:MouseEvent):void
36  {
37      MovieClip(this.root).gotoAndPlay(1, "Scene 1");
38  }
39
```

Line 37 of 39, Col 46

6

7

8

9



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window**
4. Click on **Action Panel** or press **F9**



# Timeline Navigation using a Button

The screenshot displays the Adobe Animate workspace. The top menu bar includes File, Edit, View, Insert, Modify, Text, Commands, Control, Debug, and Window. The 'Window' menu is open, showing options like Duplicate Window, Toolbars, Timeline, Tools, Properties, Library, Common Libraries, Motion Presets, Project, Actions, Code Snippets, Behaviors, Compiler Errors, Debug Panels, Movie Explorer, Output, Align, Color, Info, Swatches, Transform, Components, Component Inspector, Other Panels, Extensions, Workspace, and Hide Panels. The 'Timeline' menu item is highlighted with a red arrow labeled '1'. The 'Actions' menu item is also highlighted with a red arrow labeled '4'. The main workspace shows a timeline with a playhead at 5 seconds. The video preview window displays a scene with the text 'Golden Gate Bridge' overlaid on a landscape. The Properties panel on the right shows the instance name 'button2' and various settings like Position and Size, Color Effect, Display, Tracking, and Filters. A red arrow labeled '2' points to the 'button2' instance name. A red arrow labeled '3' points to the 'Window' menu. A red arrow labeled '4' points to the 'Actions' menu item.



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

6. Click on the **Code Snippets**
7. Expand the **Timeline Navigation folder**
8. Double click on “**Click to Go Scene and Play**”
9. Replace with the name of the scene you would like to play
10. Close the Action Panel window.

demo

# Timeline Navigation using a Button

10

The screenshot displays an IDE interface with the following components and annotations:

- Code Snippets Panel (Top Left):** Shows a tree view with 'Timeline Navigation' expanded. A red arrow labeled '7' points to this folder.
- Code Editor (Center):** Contains the following code:

```
38 }
39
40 /* Click to Go to Scene and Play
41 Clicking on the specified symbol instance plays the movie from the specified scene and frame.
42
43 Instructions:
44 1. Replace "Scene 3" with the name of the scene you would like play.
45 2. Replace 1 with the frame number you would like the movie to play from in the specified scene.
46 */
47
48 button2.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene_2);
49
50 function f1_ClickToGoToScene_2(event:MouseEvent):void
51 {
52     MovieClip(this.root).gotoAndPlay(1, "Scene 2");
53 }
54
```

A red arrow labeled '6' points to the 'Code Snippets' icon in the top right. A red arrow labeled '8' points to the function name 'f1\_ClickToGoToScene\_2'. A red arrow labeled '9' points to the string 'Scene 2' in the code.
- Actions Panel (Bottom):** Shows 'Actions: 1' and 'Line 52 of 54, Col 46'.



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window**
4. Press **F9** or select on **Action Panel**.



# Timeline Navigation using a Button

The screenshot displays the Adobe Animate workspace with the following elements:

- Timeline:** Shows a sequence of layers: 'Actions', 'buttons', and 'background'. The 'buttons' layer is selected, and a red vertical line indicates the current time position on the timeline.
- Window Menu:** Opened from the 'Window' menu in the top bar. It lists various panels. Red arrow 1 points to the entry '1 chapter10Part1.fla\*' at the bottom of the menu.
- Properties Panel:** Located on the right, showing the instance 'button3'. Red arrow 2 points to the instance name. The 'POSITION AND SIZE' section shows X: 538.95, Y: 372.95, W: 18.00, and H: 18.00. Red arrow 4 points to the 'Actions' menu item in the Window menu.
- Annotations:** Red arrows and numbers 1, 2, 3, and 4 are overlaid on the image to highlight these specific features.



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

6. Click on the **Code Snippets**
7. Expand the **Timeline Navigation folder**
8. Double click on “**Click to Go Scene and Play**”
9. Replace with the name of the scene you would like to play
10. Close the Action Panel window.

demo

# Timeline Navigation using a Button

10

The screenshot shows an IDE window titled "ACTIONS - FRAME" with the following components:

- Top Panel:** "ActionScript 3.0" dropdown and a toolbar with icons for adding, deleting, and other actions.
- Left Panel:** A project tree showing "Top Level", "Language Elements", "adobe.utils", and "air.desktop".
- Code Snippets Panel:** A list of snippets under "Timeline Navigation", with "Click to Go to Scene and Play" selected and highlighted in blue. A red arrow labeled "8" points to this snippet.
- Code Editor:** Contains the following code:

```
54  
55 /* Click to Go to Scene and Play  
56 Clicking on the specified symbol instance plays the movie from the specified scene and frame.  
57  
Instructions:  
1. Replace "Scene 3" with the name of the scene you would like play.  
2. Replace 1 with the frame number you would like the movie to play from in the specified scene.  
*/  
button3.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene_3);  
  
function f1_ClickToGoToScene_3(event:MouseEvent):void  
{  
    MovieClip(this.root).gotoAndPlay(1, "Scene 3");  
}
```

A red arrow labeled "6" points to the "Code Snippets" button in the top right. A red arrow labeled "7" points to the "Timeline Navigation" folder in the snippets panel. A red arrow labeled "9" points to the string "Scene 3" in the code. A red arrow labeled "10" points to the top right corner of the IDE window.



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

1. Click on the instance of the **button symbol** on the stage.
2. Verify the symbol has an **instance name** in the Properties panel
3. Select **Window**
4. Press **F9** or select on **Action Panel**.



# Timeline Navigation using a Button

The screenshot displays the Adobe Animate workspace with the 'Window' menu open. The 'Actions' menu item is highlighted, and a red arrow labeled '4' points to it. Another red arrow labeled '3' points to the 'Window' menu header. A third red arrow labeled '2' points to the 'button4' text in the Properties panel. A fourth red arrow labeled '1' points to the 'Actions' menu item in the 'Window' menu. The main canvas shows a scene titled 'Golden Gate Bridge' with a background image of the bridge and waves. The timeline at the top shows a sequence of actions and buttons.

| Menu Item             | Shortcut     |
|-----------------------|--------------|
| Duplicate Window      | Ctrl+Alt+K   |
| Tgolbars              |              |
| Timeline              | Ctrl+Alt+T   |
| Motion Editor         |              |
| Tools                 | Ctrl+F2      |
| Properties            | Ctrl+F3      |
| Library               | Ctrl+L       |
| Common Libraries      |              |
| Motion Presets        |              |
| Project               | Shift+F8     |
| <b>Actions</b>        | <b>F9</b>    |
| Code Snippets         |              |
| Behaviors             | Shift+F3     |
| Compiler Errors       | Alt+F2       |
| Debug Panels          |              |
| Movie Explorer        | Alt+F3       |
| Output                | F2           |
| Align                 | Ctrl+K       |
| Color                 | Alt+Shift+F9 |
| Info                  | Ctrl+I       |
| Swatches              | Ctrl+F9      |
| Transform             | Ctrl+T       |
| Components            | Ctrl+F7      |
| Component Inspector   | Shift+F7     |
| Other Panels          |              |
| Extensions            |              |
| Workspace             |              |
| Hide Panels           | F4           |
| 1 chapter10Part1.fla* |              |



# Timeline Navigation using a Button

---

## Click to Go to Scene and Play

6. Click on the **Code Snippets**
7. Expand the **Timeline Navigation folder**
8. Double click on “**Click to Go Scene and Play**”
9. Replace with the name of the scene you would like to play
10. Close the Action Panel window.



# Timeline Navigation using a Button

10

The screenshot shows an IDE window titled "ACTIONS - FRAME". The top toolbar contains various icons, with a red arrow labeled "6" pointing to the "Code Snippets" icon. Below the toolbar, the "ActionScript 3.0" library is visible on the left, and the "CODE SNIPPETS" panel is open, showing a tree view with "Timeline Navigation" expanded. A red arrow labeled "7" points to the "Timeline Navigation" folder, and another red arrow labeled "8" points to the "Click to Go to Scene and Play" snippet. The main code editor displays the following code:

```
64  /* Click to Go to Scene and Play
65  Clicking on the specified symbol instance plays the movie from the specified scene and frame.
66
67  Instructions:
68  1. Replace "Scene 3" with the name of the scene you would like play.
69  2. Replace 1 with the frame number you would like the movie to play from in the specified scene.
70  */
71  button4.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene_4);
72
73  function f1_ClickToGoToScene_4(event:MouseEvent):void
74  {
75      MovieClip(this.root).gotoAndPlay(1, "Scene 4");
76  }
```

A red arrow labeled "9" points to the string "Scene 4" in the code. At the bottom right of the IDE window, a red arrow labeled "10" points to the window's title bar.

# Start and Stop an Animation with a Button



**How can I create comments to make my script more readable?**

You can write comments in ActionScript to give yourself or someone else reading your code information that you want Animate to ignore.

A single line comment is written with two backslashes like this:

*// this is a single-line comment*

*/\* Multiline comments start with a slash followed by an asterisk. To end a multiline comment type an asterisk followed by a slash \*/*

# Start and Stop an Animation with a Button



**My ActionScript has odd indentation is there an easy way to format it properly?**

You can click the Auto Format script button to have Animate automatically reset the indentation of each line of your script.

However, if you script contains errors, the formatting will fail.

# Create and Include an External ActionScript File

---

- You can compose your scripts in a separate **ActionScript (.as)** file and include them in your Animate movie.
- This is great way to keep your Scripts organized, and it also makes them easy to reuse in other projects.

# Create and Include an External ActionScript File

---



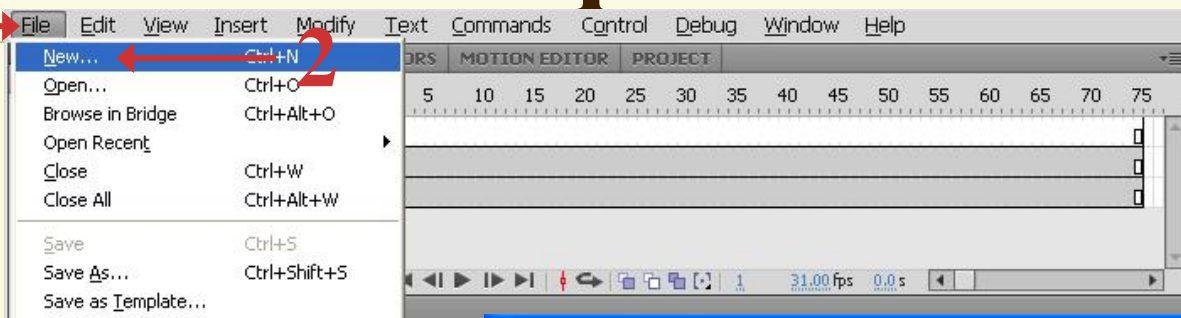
## Create the Script File

1. Click **File**.
2. Click **New**
3. Click **ActionScript** File
4. Click **OK**

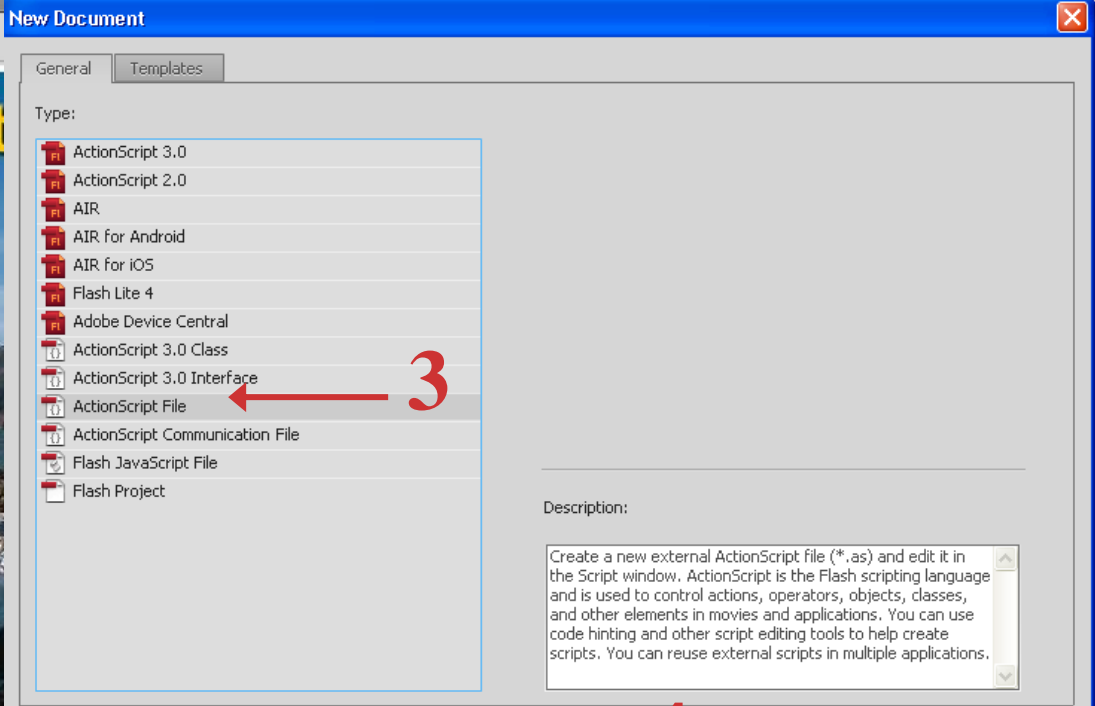


# Create and Include an External ActionScript File

1



- New... (Ctrl+N)
- Open... (Ctrl+O)
- Browse in Bridge (Ctrl+Alt+O)
- Open Recent
- Close (Ctrl+W)
- Close All (Ctrl+Alt+W)
- Save (Ctrl+S)
- Save As... (Ctrl+Shift+S)
- Save as Template...
- Check In...
- Save All
- Revert
- Import
- Export
- Publish Settings... (Ctrl+Shift+F12)
- Publish Preview
- Publish (Alt+Shift+F12)
- AIR Settings...
- ActionScript Settings...
- File Info...
- Share my screen...
- Page Setup...
- Print... (Ctrl+P)
- Send...
- Exit (Ctrl+Q)



3

4



# Create and Include an External ActionScript File

---

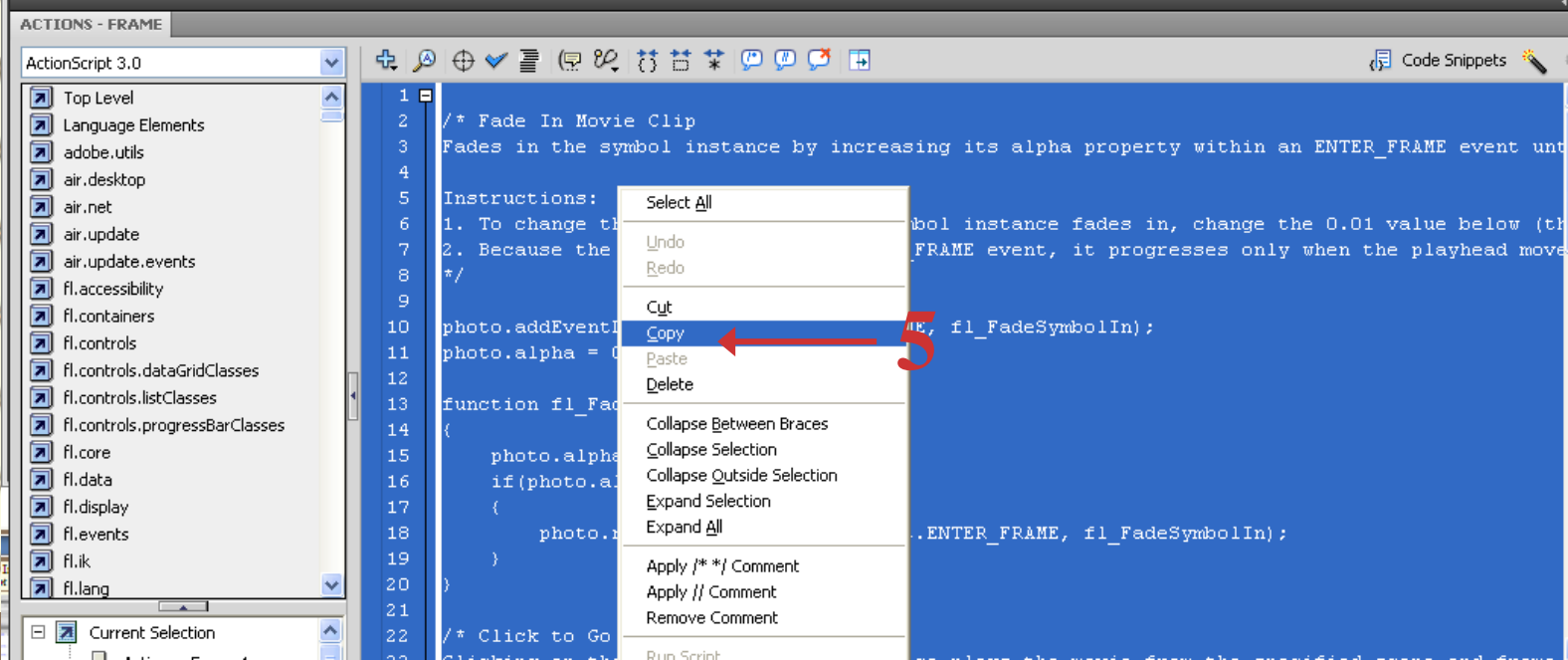
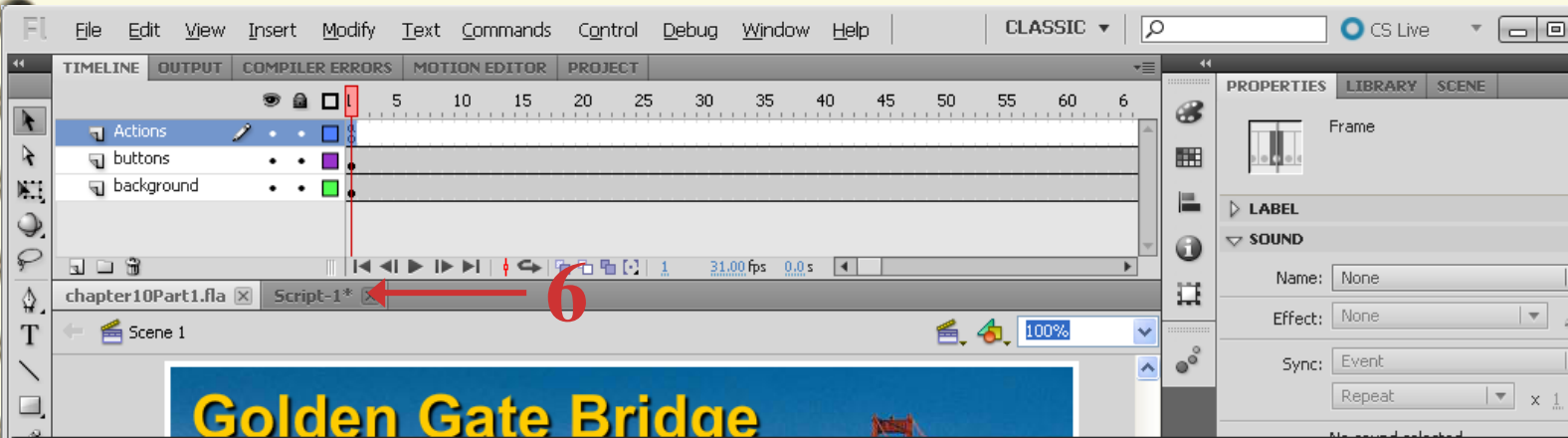


## Create the Script File

5. **Copy** the ActionScripts you created in Scene 1 for the movie clip and buttons.
6. Toggle back the ActionScript
7. **Paste** the ActionScripts in the ActionPanel.
8. Remove all lines except the **calling functions**.

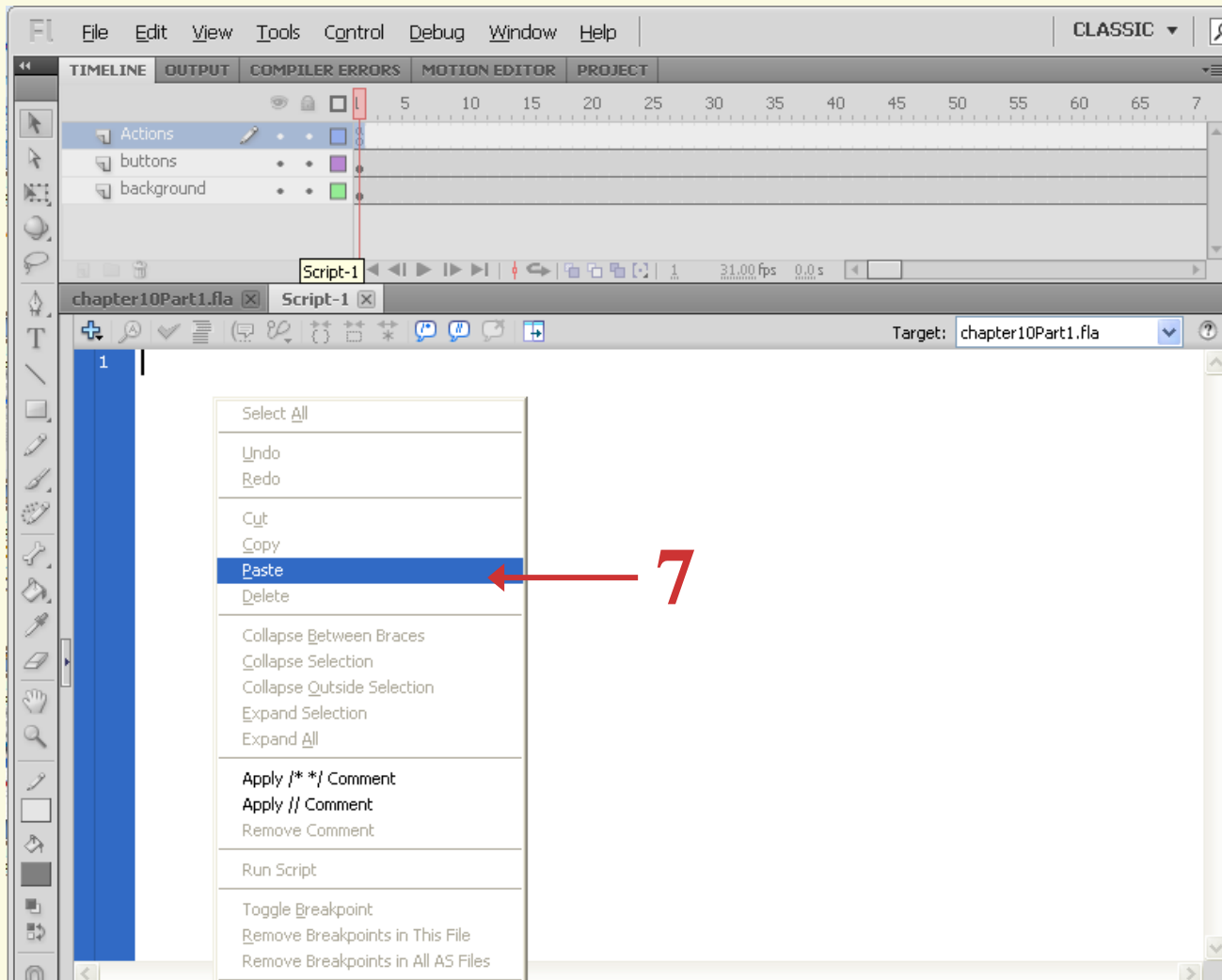


# Create and Include an External ActionScript File





# Create and Include an External ActionScript File





# Create and Include an External ActionScript File

The screenshot shows the Adobe Animate interface. At the top, the menu bar includes File, Edit, View, Tools, Control, Debug, Window, and Help. Below the menu bar are tabs for TIMELINE, OUTPUT, COMPILER ERRORS, MOTION EDITOR, and PROJECT. The timeline shows a sequence of frames from 1 to 60. The main workspace displays the 'chapter10Part1.fla' file with an external ActionScript file named 'Script-1\*' included. The code in the script file is as follows:

```
1 photo.addEventListener(Event.ENTER_FRAME, f1_FadeSymbolIn);
2 photo.alpha = 0;
3
4
5 button1.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene);
6
7
8 button2.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene_2);
9
10
11 button3.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene_3);
12
13
14 button4.addEventListener(MouseEvent.CLICK, f1_ClickToGoToScene_4);
15
16
```

A red bracket on the right side of the code block groups lines 5 through 14, with a large red number '8' next to it, indicating that these eight lines of code are included from an external file.



# Create and Include an External ActionScript File



## Create the Script File

9. Select **File** and then **Save As**
10. Enter the **name** of the ActionScript -  
*“callingfunctions.as”*
11. Click on the **Save** button



# Create and Include an External ActionScript File

The screenshot shows the 'File' menu with 'Save As...' selected. A 'Save As' dialog box is open, showing the file name 'callingfunctions.as' and the save type 'ActionScript Files (\*.as)'. Red arrows point to the 'Save As...' menu item (9), the file name field (10), and the 'Save' button (11).

```
File Edit View Tools Control Debug Window Help CLASSIC
New... Ctrl+N
Open... Ctrl+O
Browse in Bridge Ctrl+Alt+O
Open Recent
Close Ctrl+W
Close All Ctrl+Alt+W
Save Ctrl+S
Save As... Ctrl+Shift+S
Check In...
Save All
Revert
Import Script.. Ctrl+Shift+I
Print... Ctrl+P
Share My Screen...
Exit Ctrl+Q
```

Save in: chapter10

File name: callingfunctions.as

Save as type: ActionScript Files (\*.as)

button2.addEventListener(Event.CLICK, f1\_ClickToGoToScene\_4);

button3.addEventListener(Event.CLICK, f1\_ClickToGoToScene\_4);

button4.addEventListener(MouseEvent.CLICK, f1\_ClickToGoToScene\_4);

# Create and Include an External ActionScript File *(continue)*

---

- Your external scripts are executed on the frame where you include them, unless they only contain function and variable declarations.

# Create and Include an External ActionScript File

---



## Include the Script in Your Movie

1. Go to the **Movie scene**
2. Insert a **New Layer**
3. Right click on the blank keyframe and select **Action**

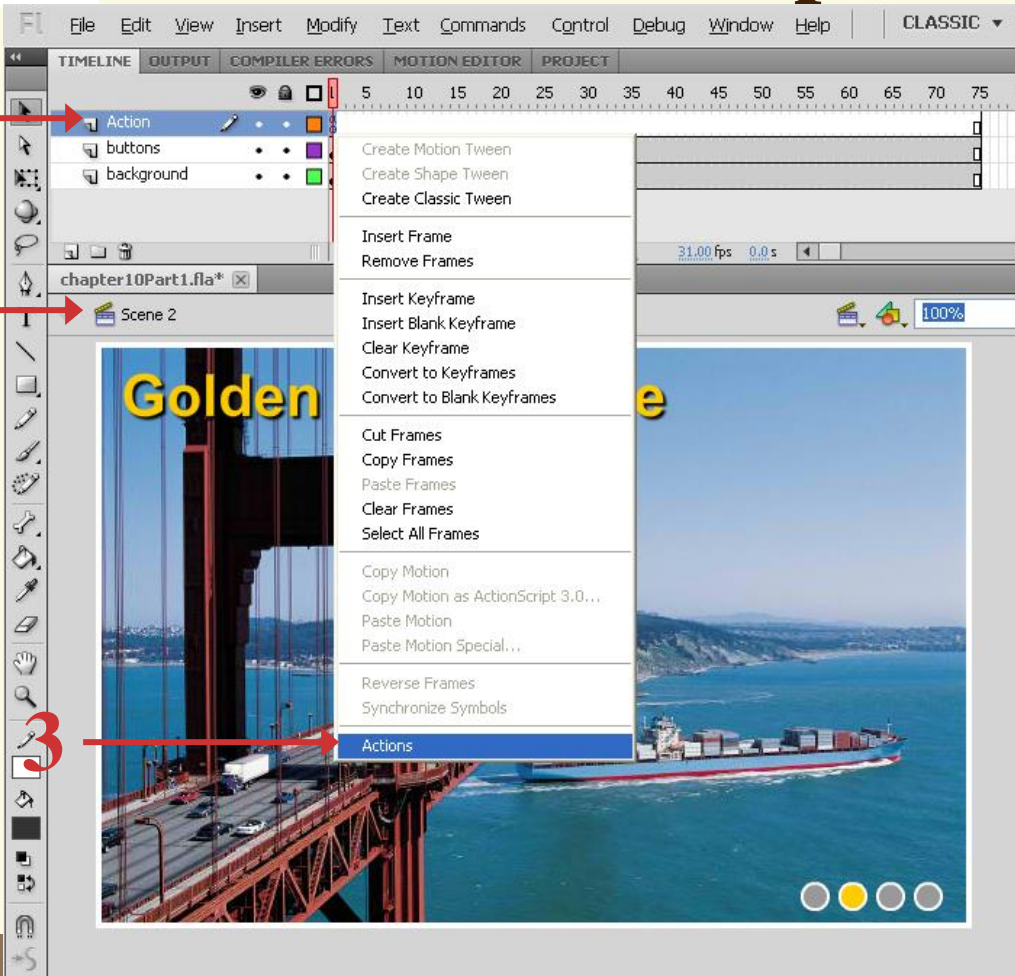
# Create and Include an External ActionScript File



## Include the Script in Your Movie

2

1



# Create and Include an External ActionScript File

---



## Include the Script in Your Movie

4. In the ActionPanel, type *include*  
*“callingfunctions.as”*
5. Close the Action Panel window
6. Repeat steps 1 to 5 for the rest of the scenes.

# Create and Include an External ActionScript File



## Include the Script in Your Movie

The screenshot shows the 'ACTIONS - FRAME' panel in an animation software. The panel is divided into three main sections:

- Left Panel:** A tree view showing the project structure. It includes 'ActionScript 3.0' with various sub-folders like 'Top Level', 'Language Elements', and 'adobe.utils'. Below this is a 'Current Selection' tree showing a hierarchy of 'Scene 1' > 'Action : Frame 1'.
- Right Panel:** A code editor window containing the following ActionScript code:

```
1 include "callingfunctions.as"  
2
```
- Bottom Panel:** A status bar showing 'Action : 1' and 'Line 2 of 2, Col 1'.

Two red arrows point to specific elements in the code editor: arrow '4' points to the file name `"callingfunctions.as"`, and arrow '5' points to the window's title bar.

# Create and Include an External ActionScript File



**Can I create external ActionScript files with other programs?**

Yes. Many developers prefer to use external editors to write their script, like TextMate or Eclipse.

You must make sure that whatever editor you use, that you save your file as plain text, to ensure that Animatees can read it properly.