ActionScript 3.0: Gaming Fundamentals

ADOBE Flash Professional Creative Cloud is a powerful platform for the creation of Web-based games. ActionScript 3.0, the language that powers Flash Professional, is an object-oriented programming (OOP) language. So learning about objects is crucial to creating complex games in Flash.



Objective:

Use ActionScript 3.0 gaming fundamentals to create Web-based games.

Key Terms:

Button
constructor methods (constructors)
event
event listener functions
event objects
event targets
instances
method
MovieClip
nested MovieClip
object
object instantiation
object-oriented programming (OOP)
property
TextField

Understanding ActionScript 3.0 Language

Program development is a structured process that can be broken down into a set of predetermined steps. In the past, methods of program development were procedural in nature. Emphasis was placed on developing the algorithm or functions. Data needed for the applications was developed separately from the procedures (functions) that worked with that data. As the number of programs grew, code from one program was reused in other programs to reduce application development time. When the shared procedure required data to work on, it was the programmer's responsibility to provide data in the format required by the shared code.

Copyright © by CAERT, Inc. — Reproduction by subscription only.

OBJECT-ORIENTED PROGRAMMING (OOP)

Object-oriented programming (OOP) is operating instructions that work with the philosophy that data is the most important asset in a software application. Procedures (or functions) are developed to protect the data. In OOP, data—along with procedures that work on it—are integrated into a single package called an object.

An **object** (class) is a data structure, together with a set of procedures that operate on the data. Once an object has been created, it can be used multiple times. If an object that depicts a circle has been created with the name circle_obj, it can be used to create two circles in an application. Each of the two circle objects is referred to as an instance. **Instances** are creations that may look identical, but they are allowed to behave differently. For example, one circle can be made to move upward, and the other circle may be made to move downward. To differentiate the two circles (instances) in the application, each may be given a unique instance name.

Properties and Methods

Objects are composed of properties and methods. A **property** in an object is treated as a variable. Therefore, placing a value in a property is accomplished via an assignment statement. A **method** is a function definition maintained inside the object. It is invoked using the same syntax used to invoke a function.

Object properties and methods are accessed with the "dot" operator.

To place a value in an object instance's property, the following statement is used:
objectInstance.property = some value;

To invoke an object instance's method or function, the following statement is used:
objectInstance.methodName();

ActionScript 3.0 Built-in Objects

ActionScript 3.0 contains many built-in objects that can be used in an application for a specific purpose, such as working with dates and formatting numbers. Using an object in a program requires that a set of distinct steps be followed. These steps are explained below, using the Date object provided by ActionScript 3.0 as an example.

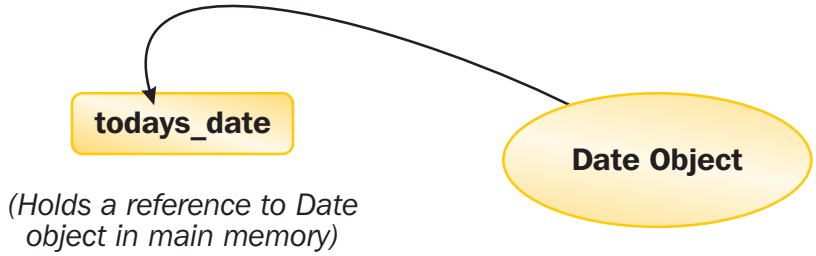
In FIGURE 1:

- Line 3 declares the todays_date variable with Date as the data type. It is a complex data type, unlike an int or Number data type. Object variables are reference variables. Reference variables cannot be used until they hold a reference to an object in main memory. The object reference can be seen as the memory address of the object in main memory. After line 3 has been executed, the todays_date reference variable does not hold a reference (or a memory address) to any object in main memory.
- Line 4 creates (or instantiates) a Date object in main memory using the new keyword, and places the reference (or memory address) of the newly created Date object in the todays_date variable. This statement is an **object instantiation** statement since it creates an instance of an object in main memory.

Copyright © by CAERT, Inc. — Reproduction by subscription only.

```
1  //Create an instance of the Date object
2
3  var todays_date:Date;
4  todays_date = new Date();
5
6  trace("todays_date contains ", todays_date);
```

```
Timeline | Output | Compiler Errors
todays_date contains  Sun Jun 28 15:53:46 GMT-0500 2015
```



(Holds a reference to Date object in main memory)

FIGURE 1. Code to create a Date object is shown here. (Adobe product screen shot(s) reprinted with permission from Adobe Systems Incorporated.)

- Line 6 prints out the object variable using the trace statement and the information maintained in the object is printed out.
- When a Date object is instantiated with no arguments provided to the instantiation function, as shown in line 4, it creates a Date object that hold the current system date.
- Objects have properties. In code, each property is treated as a variable. Some properties are read/write in nature, which means that they can be updated using assignment statements. Read-only properties can only be viewed, but they cannot be updated.
- Properties of the Date object are shown in VM–B, along with code that makes use of the properties. The "dot" operator is used in lines 13 through 17 to retrieve properties of the instance variable todays_date. The properties of the Date object shown in the table in VM–B are numeric properties. They are all read-only properties and may not be changed in code. As these properties contain integer values, they may be used as index values to access an element in an array. This is demonstrated in line 14, where the month property is used as an index to access elements from the monthNames string array declared in lines 2 through 4. The same approach is used to display the weekday name instead of a number in line 16.
- The Date object supports methods along with properties. These methods start with the word "get," which retrieves values from an object. These functions return integer values that can be used as an index across arrays. The methods shown here can be used interchangeably with the properties mentioned in the previous section. ActionScript 3.0 treats these function calls and property values in the same manner.

Copyright © by CAERT, Inc. — Reproduction by subscription only.

Method Name and Return Type	Description
Date(yearOrTimevalue:Object, month:Number, date:Number = 1, hour:Number = 0, minute:Number = 0, second:Number = 0, millisecond:Number = 0) Constructs a new Date object that holds the specified date and time.	Class constructor function that takes in

```
1  var start_date:Date;
2  start_date = new Date(2015,0,1);
3
4  trace(start_date);
```

```
Timeline | Output | Compiler Errors
Thu Jan 1 00:00:00 GMT-0600 2015
```

Source: http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/Date.html#includeExamplesSummary

FIGURE 2. Aside from these methods, the Date object supports other methods, such as the constructor methods. (Adobe product screen shot(s) reprinted with permission from Adobe Systems Incorporated.)

Aside from these methods, the Date object supports other methods, such as the constructor method. **Constructor methods (constructors)** are functions that share the same name as the class in which they are defined. The constructor method is automatically executed by the system when an object variable is created (or instantiated).

In FIGURE 2:

- Line 2 in the code invokes the constructor function and sends in three arguments to the function call. As per documentation, this first argument is the calendar year; the second argument is the month number (with 0 as the month number for January, 1 for February, and so on); and the third argument is the day of the month.
- Once the Date object has been created, the properties of the object variable can be retrieved, and class methods can be executed.

MANIPULATE MOVIECLIPS

Animations are implemented in ActionScript 3.0 using MovieClips. In ActionScript 3.0, a **MovieClip** is an object with properties and methods. To create a MovieClip:

- STEP 1: Place a graphic on the stage.

Copyright © by CAERT, Inc. — Reproduction by subscription only.



FURTHER EXPLORATION...

ONLINE CONNECTION: Importance of Learning Object-Oriented Programming in the Era of Mobile Development

Most modern applications are developed using OOP principles. Understanding the process of object instantiation and using object properties and methods is an essential skill for creating new content. While OOP environments differ, the basic principles are much the same.

Popular OOP environments include Java for Web and Android, Objective-C for iOS development, and .NET for C# for Visual Basic for Windows phones. Learn more at:

http://www.informit.com/articles/article.aspx?p=2036576

- STEP 2: Go to Modify|Convert to Symbol and provide it with a valid name and make sure that it is a MovieClip. Once a MovieClip object has been created, it appears in the library.
- STEP 3: To place another graphic on the stage, the MovieClip can be dragged from the library onto the stage.

MovieClips as Objects

MovieClips are objects. MovieClip instances can be manipulated by working with their properties and methods. MovieClip properties may be modified at design time or at runtime.

Properties that may be modified at design time are displayed in the Properties Panel. The properties that may be modified via the Properties Panel include the x, y, height, width, and color-related and display properties.

Property	Description
alpha:Number	The alpha transparency value of the MovieClip.
height:Number	The height of the movie clip, in pixels.
parent:MovieClip	The object that contains the MovieClip.
rotation:Number	The rotation of the MovieClip, in degrees, from its original orientation.
scaleX:Number	1 for the same size, 2 for doubling it, and 0.5 for halving it
scaleY:Number	1 for the same size, 2 for doubling it, and 0.5 for halving it
useHandCursor:Boolean	A Boolean value that indicates whether the pointing hand (hand cursor) appears when the mouse rolls over a MovieClip.
visible:Boolean	A Boolean value that indicates whether the movie clip is visible.
width:Number	The width of the MovieClip, in pixels.
x:Number	The x coordinate of MovieClip.
y:Number	The y coordinate of MovieClip.

FIGURE 3. The most commonly used properties of MovieClip objects are displayed here.

Copyright © by CAERT, Inc. — Reproduction by subscription only.

Properties may be modified at runtime using ActionScript 3.0 statements. The instance name of the red circle on stage is called red_circle_1. Once the code executes, the rotation of the instance is changed. Since the ActionScript 3.0 code resides in the first frame of the file, it executes as soon as execution begins.

MovieClip instances support a method called play. This method runs through the frames within a MovieClip instance.

NESTED MOVIECLIPS

A **nested MovieClip** is an animation inside the main animation—a movie inside the main movie. Nested MovieClips are created by combining MovieClips or nesting one MovieClip inside another. The act of nesting MovieClips allows for more granular levels. In a Flash CC Professional application, animation is created in ActionScript by manipulating properties and methods of MovieClips.

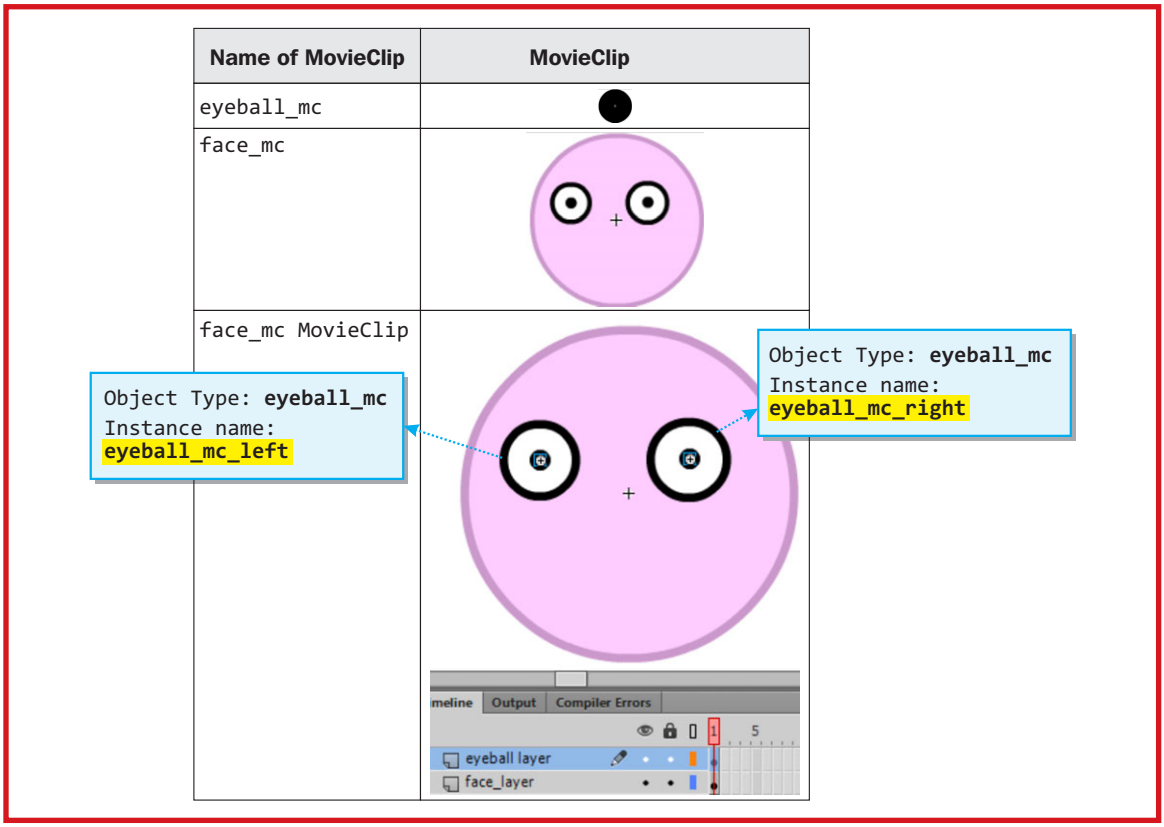


FIGURE 4. A Flash Professional CC application that has two MovieClips in its library is illustrated here. One MovieClip is called face_mc that displays a face and a second MovieClip called eyeball_mc that represents an eyeball. (Adobe product screen shot(s) reprinted with permission from Adobe Systems Incorporated.)

The eyeball_mc MovieClip is nothing more than a solid circle in black. The face_mc MovieClip is more complex. It contains two layers. The face_layer contains a circle that represents the face and two circles that represent the two eyes. The eyeball layer contains two instances of the eyeball_mc MovieClip. Each instance has a unique name.

Copyright © by CAERT, Inc. — Reproduction by subscription only.

The stage contains one instance of face_mc named face_mc_1. Using ActionScript 3.0 code, the face_mc MovieClip instance can be manipulated. Each instance of the face_mc MovieClip contains these MovieClip instances in code: the face_mc_1 (which is an instance of face_mc object), the eyeball_mc_right, and eyeball_mc_left instances inside face_mc_1. To access the eyeball_mc instances nested inside the face_mc instance, the following syntax is used:
<Owner MovieClip instance>.<Nested MovieClip Instance>.property

Line 1 modifies the alpha property of the face_mc object instance with the instance name face_mc_1. This results in the instance appearing transparent at run time. Lines 2 and 3 modify the x property of the two nested MovieClip instances inside the face_mc_1 object instance. At run time, it appears that the MovieClip instance is looking left.

CREATE EVENT HANDLER FUNCTIONS

An **event** is an action that can be detected by Flash and includes mouse movement and user activity on the keyboard. Events are constantly occurring in a Flash Application. Frames start, the mouse moves across the stage as per the user's actions, the keyboard is pressed by the user, and so on. These events occur even when no code has been written to execute when an event occurs.

Understanding Event Handling in ActionScript 3.0

Event handling in ActionScript 3.0 is a complex process with multiple components that work in tandem to cause program execution in the desired order. Before developing code to execute when an event occurs, the event model must be understood. The various components of the event model are:

- **Event objects** are items that represent real events by storing information about detailed events in a set of properties. When an event occurs, the system creates an event object instance that maintains information (e.g., event date and time). Event objects contain a set of methods that allow a person to manipulate event objects and affect the behavior of the event-handling system.
- **Event targets** are the object instances that initiate the event. For example, when the mouse is clicked on a MovieClip instance, the instance is the event target.
- **Event listener functions** are user-defined functions executed by the system when the event occurs.

Object Instances and Event Handling

For actions to happen when an event occurs, a specific object instance in the application has to be told to listen for the event and to act upon it. One object instance may respond to multiple events. For example, a MovieClip object instance may respond to mouse left click and mouse over events. In such cases, the object instance needs to be instructed to watch for both events.

Copyright © by CAERT, Inc. — Reproduction by subscription only.

Object instances, not objects, can be made event listeners. This means that two instances of the same object may respond to different events or respond differently to the same event. For example, clicking on one instance of a MovieClip may render that instance invisible, but clicking on another instance of the same object may render that instance larger.

Mouse Event Handling

ActionScript 3.0 contains statements that allow an object instance to be a listener for a specific mouse action.

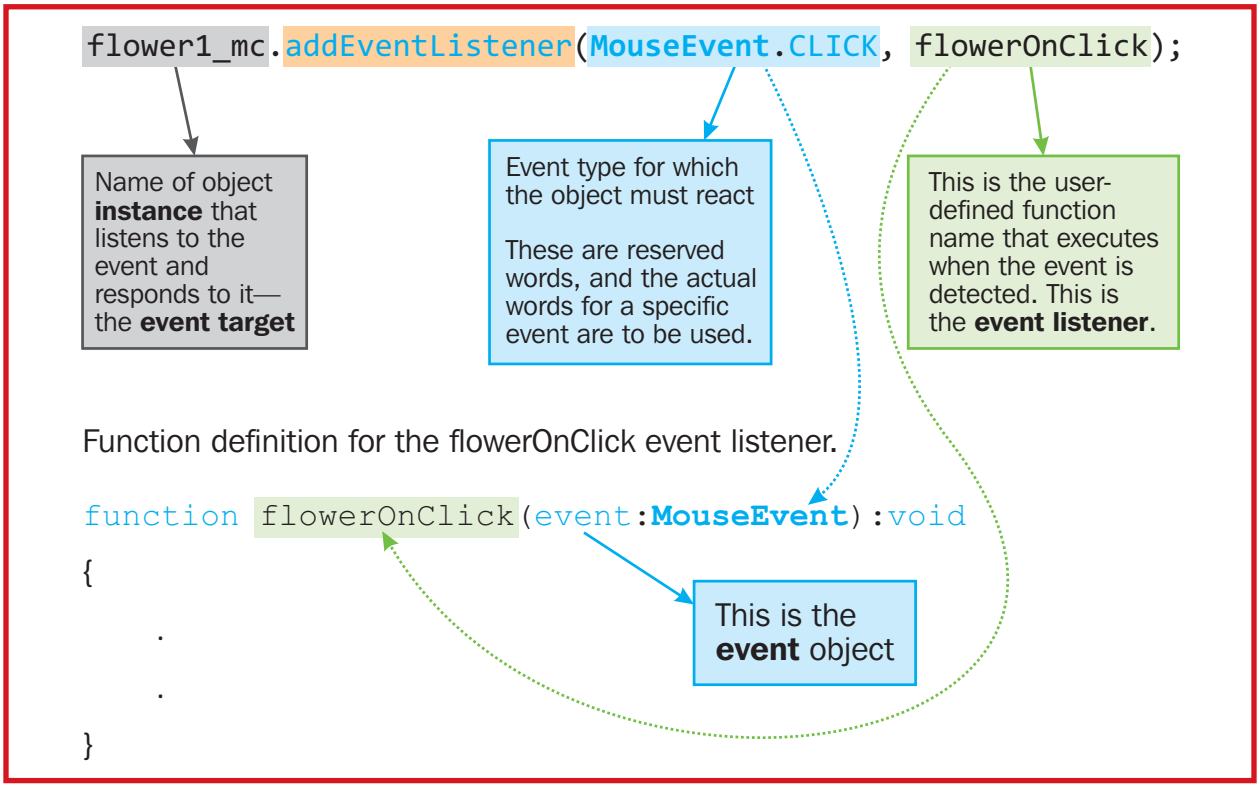


FIGURE 5. The code illustrates mouse event handling.

The statement to register an event handler function to an object instance starts with the instance that needs to handle the event. This is followed by a call to an "addEventListener" method. Also included in the statement are the names of the type of event and the event handler function. The event type will vary depending upon the event being handled.

The event handler function is a void function that takes in an event object as a parameter. All event handlers are required to have a function header in this format. If not, the function is not recognized by the system as a valid event handler.

- The general form for registering an object to an event is:

  <event target>.addEventListener(<EventType.EventName>, <eventHandler Function>)

Copyright © by CAERT, Inc. — Reproduction by subscription only.

- The general format for the event handler function is:

```
function <event handler function name>(event:Event):void
{
//ActionScript 3.0 statements
}
```

Keyboard Event Handling

Keyboard events are handled differently from mouse events. A mouse event can be handled by a MovieClip instance. Multiple mouse event handlers may be coded for MovieClip specific instance as needed. Keyboard events are handled by the stage object, which represents the entire application area.

```
1    stage.addEventListener(KeyboardEvent.KEY_DOWN,on_keydown);
2
3    //------- event handlers ----------
4    function on_keydown(event:KeyboardEvent):void
5    {
6         face_mc_1.y += 1;
7    }
```

FIGURE 6. This is as an illustration of code to move a MovieClip object instance called face_mc_1 to the right when any key in the keyboard is pressed down. When the key is released, no function is executed because there is no event handler for the key-up event. This event occurs when any key on the keyboard is pressed.

To determine which key on the keyboard has been pressed, a decision structure is implemented in the event handler. The complete list of values for the keyCode property can be viewed at http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/flash/ui/Keyboard.html.

Buttons and Event Handling

In Adobe Flash Professional Creative Cloud, a **Button** is an object with properties and methods. Additionally, Buttons can display different graphics for normal view, mouseover view and mousedown view. Button instances in an application can respond to events.

When a Button object instance is placed on the stage, the graphic in the Up frame is displayed.

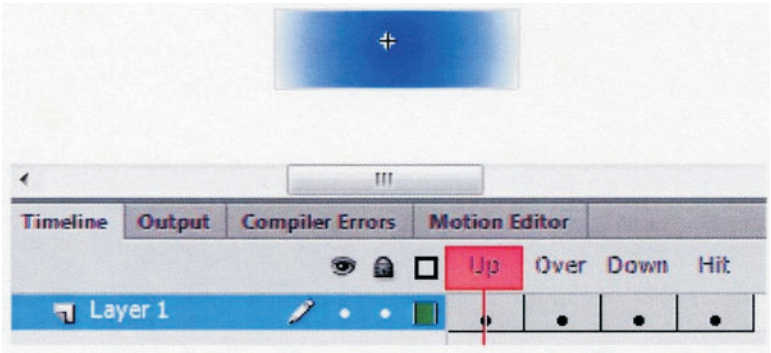


FIGURE 7. Button objects are created by going to Insert|New Symbol, and selecting Button as the object type. When a button is created, it contains four frames with the names Up, Over, Down, and Hit. (Adobe product screen shot(s) reprinted with permission from Adobe Systems Incorporated.)

Copyright © by CAERT, Inc. — Reproduction by subscription only.

When the file is executed, the graphic in the Over frame is displayed when the mouse moves over the Button, and the graphic in the Down frame is displayed when the mouse is clicked on the Button. The graphic displayed in the Hit frame determines the area on the stage that should register the click event. Text needs to be placed inside the object.

Other Events and Methods in ActionScript 3.0

Aside from mouse- and keyboard-related events, there are other events handled in ActionScript 3.0. One is the ENTER_FRAME event. ENTER_FRAME event is triggered at the frame rate of the movie. For example, if the movie has a frame rate of 24 frames per second, code within the ENTER_FRAME event handler function will execute 24 times per second.

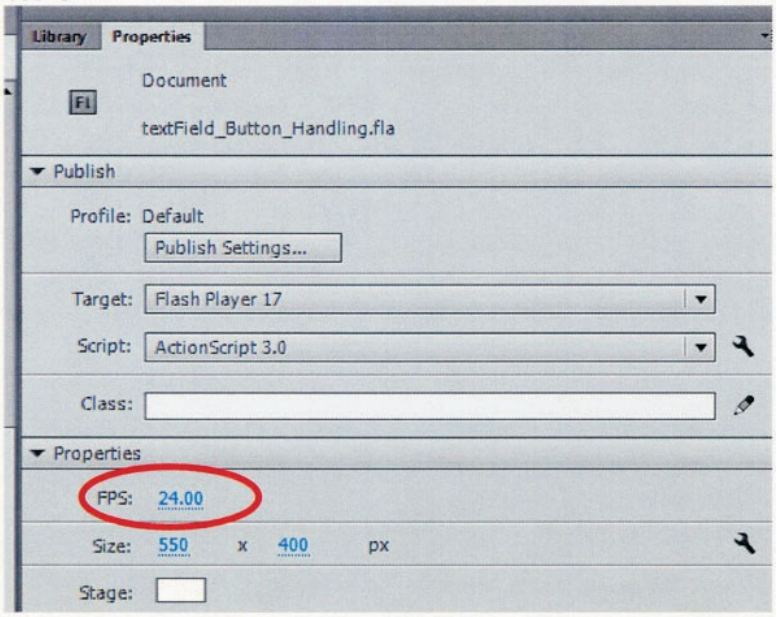


FIGURE 8. The properties inspector of the document, which displays the frame rate for the application, is illustrated here. (Adobe product screen shot(s) reprinted with permission from Adobe Systems Incorporated.)

```
1     stage.addEventListener(Event.ENTER_FRAME, frame_event);
2
3     function frame_event(event:Event):void
4     {
5           flower_mc_1.rotation+=10;
6     }
```

FIGURE 9. Code in the frame enter event handler that modifies the rotation of a MovieClip instance by 10 is illustrated here. Since this code will execute 24 times a second, the instance will continuously rotate at turn time.

Built-in ActionScript Methods

ActionScript 3.0 provides certain methods that can be used inside a MovieClip instance to control playback actions: goto methods. ActionScript 3.0 provides methods that work with frames within an application. These are the gotoAndStop and gotoAndPlay methods. Both of these methods require a numeric parameter. For example: gotoAndPlay(3);

- Method 1: gotoAndStop passes control to the frame number specified as a parameter and then stop.

Copyright © by CAERT, Inc. — Reproduction by subscription only.

- Method 2: gotoAndPlay passes control to the specified frame and continues to play the movie from that frame onward.
- Named frames: Since the two goto methods take in frame numbers, they can be used to manipulate MovieClips to play different parts of a movie based upon user input provided one knows the actual frame numbers where desired animations begin.

Instead of using frame numbers, frame labels can be used. Creating a separate layer to hold frame names typically does this.

WORKING WITH TEXTFIELDS

A **TextField** is an area in the application stage that can hold text. This includes text displayed at runtime and text areas on the stage where a user can enter text data. In Adobe Flash Professional Creative Cloud, three types of TextFields are Static TextFields, Input TextFields, and Dynamic TextFields.

Static TextFields

Static TextFields are rendered on the stage using the Text tool. The properties of this type of TextField can be changed using the Properties Panel. This type of TextField cannot be given an instance name. This means that the properties of a static TextField instance cannot be changed at runtime.

Input TextFields

When creating a TextField using the Text Tool, the TextField type can be set to "Input Text," and a name can be provided for the TextField. Properties can be set for this type of TextField during design time. Text may be placed in the field at runtime by updating the text property of the TextField. When the application executes, the user can enter text in the TextField and using the text property can retrieve user-entered text.

Dynamic TextFields

Dynamic TextFields are similar to Input TextFields. Using the Text Tool and setting it up as "Dynamic Text" creates it. Then the text can be assigned an instance name, and its contents can be set using its text property. However, unlike an Input TextField, the end-user cannot modify the contents of a Dynamic TextField.

TEXTFIELD CODE

Aside from the text property, TextFields have other properties that can be modified at runtime. Color is specified in RGB notation starting with a 0x followed by two digits for red, green, and blue,

Copyright © by CAERT, Inc. — Reproduction by subscription only.

When using the htmlText property, HTML tags may be used to render text. Some of the html tags can be used within the text property are <a>, , <br>, <i>, <u>, <p>. When using tags that will lead to multiple lines of output, one sets the multiline property to true before setting a value in htmlText.

INCORPORATE SOUND FILES IN A GAME

Sounds are incorporated in ActionScript 3.0 using a number of different classes. Different Sounds-related objects work together to create a rich music environment in a Flash application.

SoundMixer Class

The SoundMixer class in the ActionScript 3.0 library allows for greater control over all sounds in an application. [NOTE: This class will not be discussed in this lesson.]

An mp3 file can be placed in a Flash application by dragging it into the library. The mp3 file for this E-Unit was downloaded from http://freemusicarchive.org.

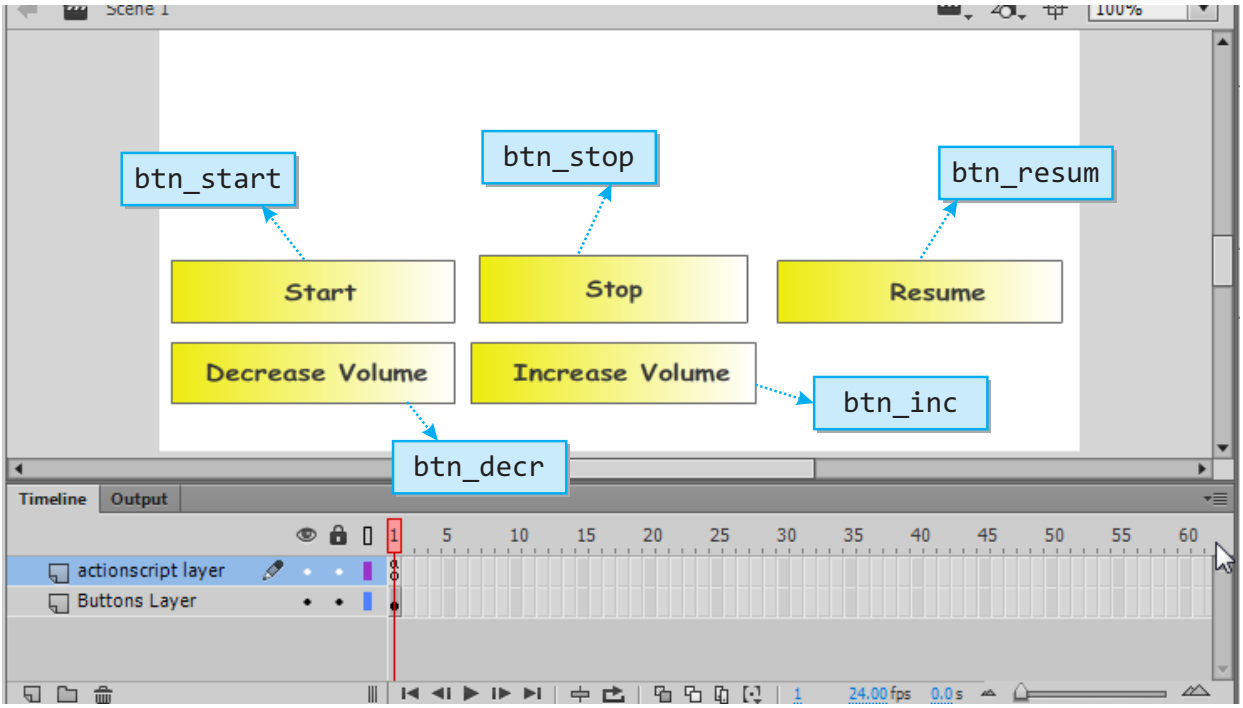


FIGURE 10. This is an illustration of a layout of a Flash Application that works with a sound file and the sound-related classes in ActionScript 3.0. As always, in the application, there is a layer to hold ActionScript code and another layer to hold buttons to control the sound file. (Adobe product screen shot(s) reprinted with permission from Adobe Systems Incorporated.)

Copyright © by CAERT, Inc. — Reproduction by subscription only.

Summary:



Learning to manipulate MovieClips and coding event handler functions allows for the creation of complex games. It also provides an understanding of application development using the principles of OOP.

Checking Your Knowledge:



1. Describe object-oriented programming (OOP).
2. Describe object properties.
3. How are MovieClips manipulated?
4. Describe a nested MovieClip.
5. What is an event?

Expanding Your Knowledge:



Practice using and writing ActionScript 3.0 code. Use the Web Links URLs below to get started.

Web Links:



ActionScript 3 Basics Tutorial: 001 Video
https://www.youtube.com/watch?v=pKj1IgWJd-A

ActionScript 3.0 Linking and Using Buttons: Flash Tutorial! Video
https://www.youtube.com/watch?v=LUJkqzCM3L4

How to Create Your First Flash Intro
https://www.youtube.com/watch?v=m6CBiq6_RZM

Copyright © by CAERT, Inc. — Reproduction by subscription only.