ActionScript 3.0

1 Definitions

ActionScript is the programming language that runs in Animate. The newest version is ActionScript 3.0, which is what is described here. Note that ActionScript 2.0 is much different; the two are not interchangeable.

Below are some definitions that will be referenced later in these notes.

- **program** (software) - a set of instructions written in a programming language such as ActionScript that will be interpreted and executed (run) in an appropriate environment (Animate).

- **function** - an ActionScript pre-defined or a user-defined set of instructions within a program. A user-defined function is named with an arbitrary identifier.

- **keyword** or reserved word - the name of a specific instruction, function, or other item in ActionScript. A keyword can not be used for anything other than its pre-defined purpose.

- **identifier** - a user-defined name given to an instance, variable, or function. An identifier must start with a letter followed by any number of letters or digits. **No** spaces are allowed!

- **event driven** - programs in Animate are executed when an event occurs. An event is usually triggered by the user pressing a button, entering text, or the movie has reached a specific frame.

- **object** - a symbol that has been defined within an Animate movie. A symbol can be a movie clip, a button, a text box, and other items with which a user can interact. An example of an object in the real world would be an automobile.

- **instance** - a specific object which has a unique identifier. For example, your car is an instance of an automobile object. It is uniquely defined by its VIN (Vehicle Identification Number) and your license plate number. The VIN can be viewed as its instance name or identifier.

- **attribute** - things that define an object, such as color, size, movement, etc.

- **.** (dot) - the dot operator connects an instance with an attribute or a function, the latter of which can start or stop some pre-defined action.

- **variable** - a location in memory named by an identifier that can store a data value, such as a number or text.
2 Buttons and Movie Clips

In this section, identifiers are shown in *italics*, and ActionScript keywords are shown in **this font**. Remember that identifiers are completely arbitrary names the user defines; however, it is best for the name to describe the object or function so as to make the program easier to understand.

Movie clips are simply small versions of an Animate movie that can run on their own within the context of the overall movie (scene). We saw this in the various bulldozer examples: the wheels turned independently of the bulldozer action. The two wheels were made from a single wheel object but with different instance names (identifiers): *wheel1* and *wheel2*. Thus, the wheel clips can be accessed separately from the entire scene.

In the scene, we next created some buttons, named *stopButton* and *playButton*. These are identifiers for the two button instances, allowing each button to act independently. To make the buttons work, a function (an **event handler**) needs to be written by the user for each action. The following examples should serve as templates for any button. Note that the indentation shown is by convention and is for the convenience of the reader; identifiers and keywords are case sensitive!

```javascript
// Anything that begins with "//" is a comment that will be ignored by Animate
function stopDozer (myEvent:MouseEvent):void
{
    stop(); // start the function body
    wheel1.stop(); // stop the movie by calling built-in stop() function
    wheel2.stop(); // attach stop() function to a specific wheel
}
```

Note that *myEvent* is a user-defined identifier that will be matched up by Animate automatically to the appropriate MouseEvent defined below. The programmer now has to match the function with a specific button, called **registering** the event handler. Assuming the stop button’s instance name (identifier) is *stopButton*, the code would look like:

```javascript
stopButton.addEventListener (MouseEvent.CLICK, stopDozer);
```

Note that *stopDozer* is the **function name**. This name is what connects the event handler (the function) with the button. The **button’s name** is *stopButton*. These are not interchangeable!

The code for the play functionality is similar:

```javascript
function playDozer (myEvent:MouseEvent):void
{
    play(); // play the movie by calling built-in play() function
    wheel1.play();
    wheel2.play();
}
```

```javascript
playButton.addEventListener (MouseEvent.CLICK, playDozer);
```