PLATT COLLEGE
SAN DIEGO

WAS400 Flash and ActionScript ~ Tiziana d'Agostino

ACTIONSCRIPT 3.0 CHEAT SHEET

BASIC DEFINITIONS

- **Object:** any element in Flash with specific properties, methods and events that can be controlled via ActionScript
- **Class:** blueprint for an object. It defines what properties, methods and events apply to that particular object
- **Static class:** a class that does not requires instantiation (creating an instance of that class). Example: Math class
- **Package:** a collection, or subdirectory of related classes
- **Variable:** a container for data, with a custom name.
- **Function:** a container for ActionScript with a custom name
- **Array:** a list of similar items
- **Loop:** a way to repeat your code automatically and BEFORE any other code executes. Must contain an end condition

COMMON DATA TYPES

There are many data types, but some of the most common ones are:

Number: Any number, including negative and decimals. Example: -4.5
int: Any integer or whole number Example: -6
uint: Any positive integer Example: 8
String: text or a string of characters Example: "hello"
Boolean: True or False
Array: More than one value in a variable. Example: [2,4,6]
Object: Any Class, either built-in or custom made

DISPLAY LIST

- Visual assets are handled by the Display List, a hierarchy containing all your visual object. The display list also brings the opportunity to create objects like shapes and sprites via AS. The best part about the display list is how consistent it is to add any object to it.
- The display list is useful for adding and removing objects, managing names, positions and data types, changing the hiearachy of objects
- it requires a two steps method; first you create the visual object, then you add it to the display list, in any nesting order:

example:

```
1) var myCar_mc:MovieClip=new MovieClip();
2) addChild(myCar_mc);
```

To display a visual object AND nest it inside another object, you write the code:

```
1) var myCar_mc:MovieClip=new MovieClip();
2) myRoad_mc.addChild(myCar_mc);
```

Some of the methods you can use with the display list:

- `addChild()` and `AddChildAt(index)`: adding at the end of the display list, i.e. making them visible. `addChild()` can also be used to move objects at the top of the stack.
- `removeChild()` and `removeChildAt(index)`: remove object from the display list
- `swapChildren(mc1,mc2)` `swapChildrenAt(0,1)`: will switch the depths between two objects. In AS3 you can't specif and index higher than existing objects in the display list or you will get an out of bound error.

CREATING A VARIABLE

basic syntax:

```
var myVariable:DataType;
```

You can assign a value to a variable immediately, following the variable declaration with an **=** and a value. A variable can have different values over time, but the data type itself can't change.

```
var myAge: Number=20;

var isPlaying:Boolean=true;

var mymessage:String="Hello"
```

EVENT MODEL

- Events come in many varieties: beside mouse and keyboard events, most classes have their own specific events.
- In AS 3 there is only one way of responding to events: by using a special function called listener function or event handler function, and an event handlers. it is a two steps process:

1) create the function to be executed. This listener function (or event handler function) is different form a custom function because it has as **only one, mandatory parameter:** the kind of event you are listening to. The name of the parameter is not fixed, as long as the data type is correct, but the word `event` or `evt` or `e` is common practice. This argument, also called the event object, retrieves information about the event, including the target and type properties, which provide essential information about the event:
   - `type`= A string indicating the type of the event.
   - `target`= A reference to the component instance broadcasting the event.

   Some events have additional properties and can use different properties to identify the object that is broadcasting the event. If you attach the listener to an object, the actual object can be referenced as `evt.target` (assuming you called your function parameter evt). This allows your event handler functions to be reusable.
2) add a listener to the necessary object and choose the appropriate event. Built in events are in classes specifically dedicated to events, and the specific event is usually a constant, oftentimes spelled all in caps.

Basic syntax

```
function functionName(event type){
code statements;
}


object.addEventListener(eventName, function);
```

example:

```
function myClick(evt:MouseEvent){
evt.target.y=15;
evt.target.rotation=45;
}
myButton_mc.addEventListener(MouseEvent.CLICK, mySlideshow);
myButton_mc.buttonMode=true;
```

Notice the use of `evt.target` to refer to myButton_mc and the last line of code, that shows an hand cursor.

CREATING AN INSTANCE OF ANY CLASS

To create an instance of any class (except static classes), you would use the following syntax:

```
var myObject:Class=new Class();
```

It is important to include the parenthesis at the end of the class name because that is actually a function (constructor function) for that class. Some classes require parameters when you instantiate them.

Examples:

```
var myCar_mc:MovieClip=new MovieClip();

var myTween:Tween=new Tween(myMc,"x",regular.easeIn,30,100,3,true);

var myTimerc:Timer=new Timer(500);
```

CREATING A CUSTOM FUNCTION

basic syntax is:

```
function myFunction(optional parameters){
    code block/s;
}
```

Parameters are optional, but often they help you making your function more modular.

BASIC STRUCTURE OF A CLASS:

1) package: This line defines the package block for an ActionScript 3 class. This is a new block that was not present in ActionScript 2. It defines the package path for the class.
2) imports: import the necessary classes. In contrast with the regular timeline, Flash needs to import all classes to be used within the class, even the built-in ones
3) class name: Here the class block is being defined. This consists of the ***class*** keyword along with the class name. The class name should be the same as the file name of the class file. The ***public*** attribute before class indicates that this class is accessible to all other scripts., including the .fla file that uses it.
4) variables: if necessary create variables here
5) constructor function: this is the function that is executed automatically when the class is instantiated. The constructor should not have a return type and the constructor can accept parameter
6) methods and properties: Properties and methods are called members of the class
7) event handlers: if necessary create event handlers here

NOTE: the class File name, the constructor function and the class declaration must all be the exact same name. Only the package, the class name and the constructor function are mandatory elements.

THREE WAYS OF USING AN EXTERNAL CLASS:

1) as a document class: all the code inside the class is executed immediately and automatically. Write the path to your class in the property panel in the appropriate area
2) linking it to an element or more in the library. Usage the properties dialog box for the object and write the path to the class either as the mainClass or a a base class. Main classes can't be shared by multiple objects. Don't forget to check the Export For ActionScript box
3) using the import statement at the beginning of your document.

In any case, make sure to define your classpath(the path to the folder where your classes are stored) in your document

COMMON TASKS

Link to a website

1) you create a URLRequest object to contain the url you want to link to. The url is actually the value of the url property if the URLRequest object
2) you write the function, and you add the necessary code. The `getURL` now becomes `navigateToURL`. This is definitely a much more complex way to handle events, but it will be used consistently in AS 3.0
3) You add an event listener to your button, specifying you are waiting for a mouse event, the CLICK event, and you want a particular function, called onClick to be executed

```
var link:URLRequest= new URLRequest("http:www.apple.com");

function onClick(event:MouseEvent):void{
            navigateToURL(link);
}

blue_btn.addEventListener(MouseEvent.CLICK,onClick);
```

Load an external visual asset (images or swf) with a preloader

The Loader class requires few steps. The steps are:
1) create a URLRequest, passing as a parameter the filePath
2) create an instance of the Loader and add it to the display list
3) create a progress function to monitor the downloading and to create a preloader
4) create a complete function that would display the images after the file has been completely loaded
5) Load the file with the `load` method

example:

```
var myLoader:Loader=new URLLoader();
var imageRequest:URLRequest = new URLRequest("photos/insect1.jpg.txt");


function progresshandler(evt:ProgressEvent):void {
    addChild(display_txt);
    addChild(bar_mc);
    var percentage:Number=evt.target.bytesLoaded/evt.target.bytesTotal;
    bar_mc.scaleX=percentage;
    display_txt.text = Math.round(percentage*100)+"%";
    }


    function finished(evt:Event):void {
    removeChild(display_txt);
    removeChild(bar_mc);
    addChild(myLoader);
    }

myLoader.contentLoaderInfo.addEventListener(ProgressEvent.PROGRESS,
                                            progresshandler);
myLoader.contentLoaderInfo.addEventListener(Event.COMPLETE, allDone);


myLoader.load(imageRequest);
```

Load an external text to style with CSS

The URLLoader class requires similar steps and uses a similar syntax to the Loader object. The steps are:
1) create a URLRequest , passing as a parameter the filePath
2) create an instance of the URLLoader
3) create a onLoad function that would display the text in the textfield after the file has been completely loaded
4) Load the text with the `load` method

example:

```
var htmlFile:URLLoader=new URLLoader();
var TxtRequest:URLRequest = new URLRequest("../src/KarimFinal.txt");


function onLoadHtml(event:Event):void {
    myText=event.target.data;
    article_txt.htmlText=myText;
    }


htmlFile.addEventListener(Event.COMPLETE, onLoadHtml);
htmlFile.load(TxtRequest);
```

To use CSS in Flash, you need to use the `StyleSheet` class. The basic steps are as following:
1) create a URLRequest , passing as a parameter the filePath to the css file
2) create an instance of the URLLoader
3) create a onLoad function that creates an instance of the StyleSheet class, parse the Css (using the .data property of the URLLoader class) and associate it with a textfield, after the file has been completely loaded
4) Load the css with the load method

example:

```
var myURLRequest:URLRequest = new URLRequest("../src/design.css");
var cssFile:URLLoader=new URLLoader();

function onLoadCss(event:Event):void {
    var myCss:StyleSheet=new StyleSheet();
    myCss.parseCSS(cssFile.data);
    article_txt.styleSheet = myCss;
    }

cssFile.addEventListener(Event.COMPLETE, onLoadCss);
cssFile.load(myURLRequest);
```