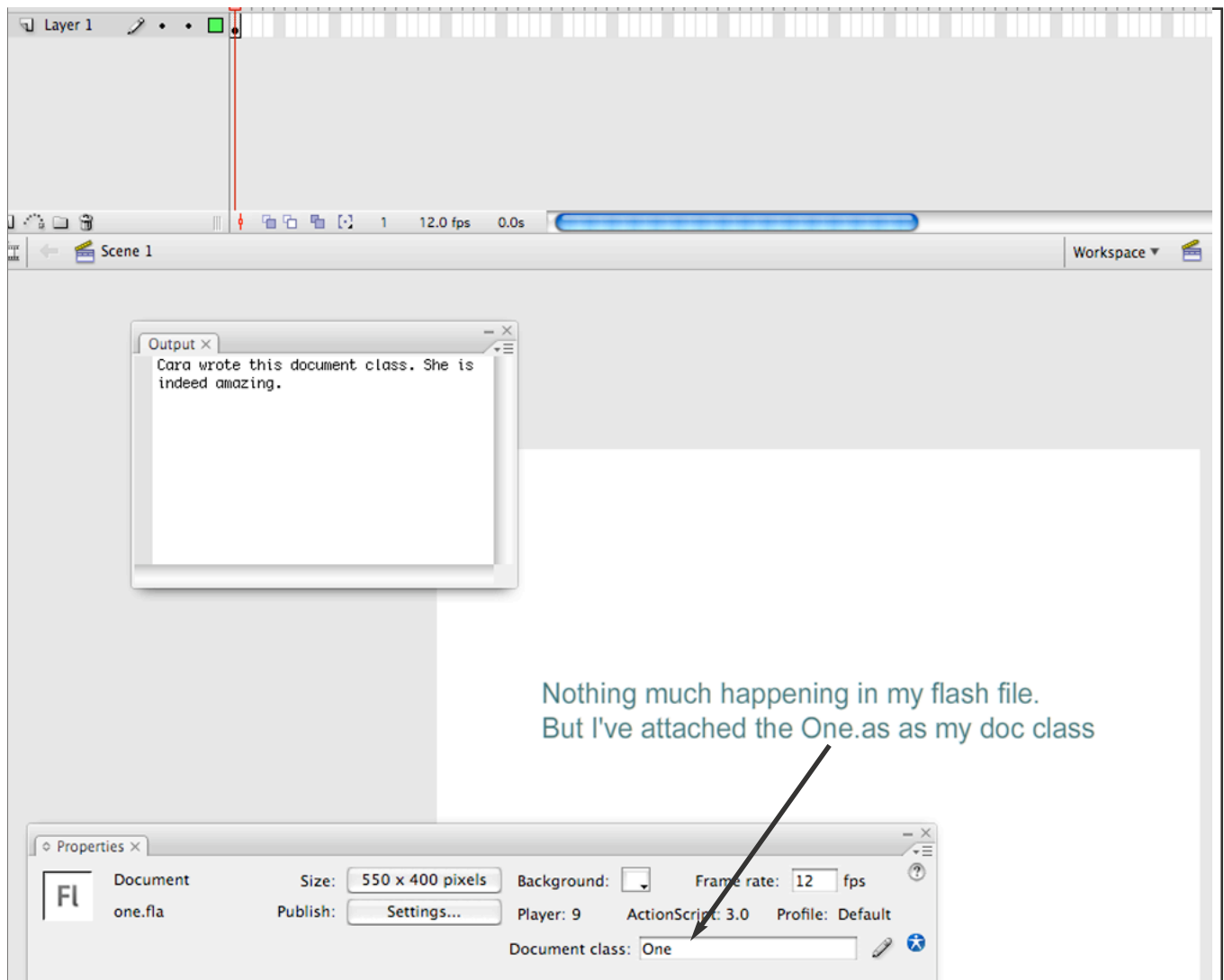


Intro to Classes: Exercises

Exercise One: Intro to the Document Class. Make a new Flash file and write a document class that traces how fabulous you are to the output window. See layout below. (to make a docClass , choose NEW actionScript file)



Nothing much happening in my flash file.
But I've attached the One.as as my doc class

```
package{  
    //if not in the same directory, you would place directory and class name here.  
    //I'm putting this in the same folder as the fla file, so I don't need that here.  
  
    import flash.display.MovieClip;  
    /**when creating a separate as file, you must import all classes. Think of this as file  
    as having "amnesia. It does not "remember" what a movieclip DOES. (as opposed to the  
    flash file which has all that info "built in". You must import this class (Movieclip)  
    to use it. All document classes must derive from either a movieclip or a SPRITE (a movie clip  
    with no timeline: ie: one frame.  
    */  
  
    public class One extends MovieClip{  
        //by convention, the class name is capitalized  
  
        public function One(){  
            /**this is the class constructor. This is the main "function". It runs  
            automatically when an instance of the class is created.  
            */  
            trace("Cara wrote this document class. She is indeed amazing.");  
        }  
    }  
}
```

Exercise Two: More work with the Document Class. Make a new Flash document and a new DocClass for it (put these in their own directory) Test your doc class with a trace statement to make sure it is working. REMEMBER: you will need to import any classes you are using. Such as:

```
-----  
import flash.display.*;
```

```
-----  
Knowing what you need to import takes time, practice and research. The above import statement imports visual aspects, such as movieClips. Since I'm going to extend the MovieClip class, I need to import that. The .* just says "import everything in that directory". If you don't import the right classes, you will get an error such as "the definition of base class MovieClip not found".
```

CREATE several movieClips in your Flash file. You will need to name these in the linkage palette of your library. You can then instantiate these through your doc class just like you did when timeline scripting except you will need an **access modifier** (public or private)

```
-----  
public var example:Example = new Example();  
-----
```

instantiate all your symbols and place them on the stage using addChild. In the constructor, make these act as buttons by adding listeners to them.

BELOW (not IN) the constructor, write functions that move your buttons, rotate them--- anything that you've learned up until now.

Need ideas? Create an object you can chase with your mouse. Make a button that changes the color of an object. Create a button that stops and starts a movieClip. Create a button that stops and starts your movie. Use this as a chance to get comfortable with the doc class. See my formatting on the next page. Get familiar with importing and syntax.

Next create a timer in your doc that runs a mc every 5 seconds. You will need to import the timer class. (hint: look it up) .

```
Package      flash.ui  
Class        public final class Keyboard  
Inheritance  Keyboard → Object
```

from the help file: this tells me that if I want to use the Keyboard class, (to access a specific keystroke , for example, I would need to import the flash.ui class. If I just want the keyboard , I would write:

```
import flash.ui.Keyboard;
```

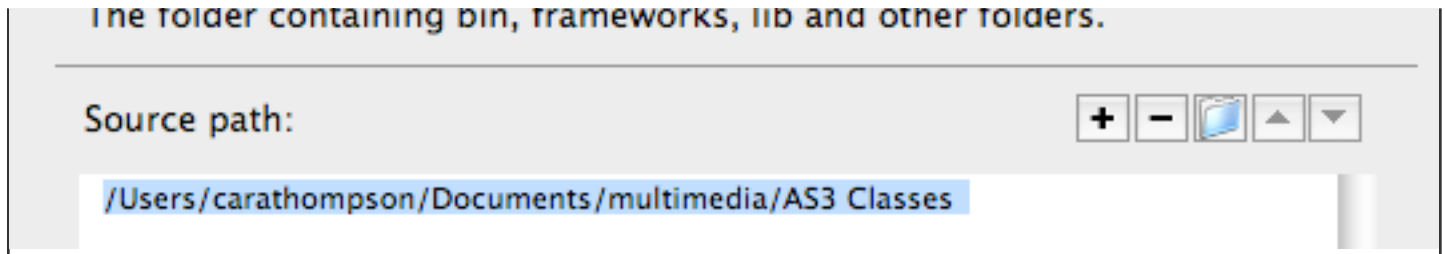
or I could use the wildcard to import all.

```
import flash.ui.*;
```


Exercise Three: Base Classes. For this exercise, you will be writing external classes that you will use as your base classes for several MovieClips.

First, set up your directory. This should be in your 348 directory on your computer. Ideally, your structure should be the name of your website in reverse order. This doesn't mean that this need have anything to do with your website; it just assures a unique "thumbprint" or name for your class. That way, if you share your classes with others, you know that no one will accidentally have a class with the same name.

Next, you need to set your classpath in Flash. To do this, go to your Publish settings>ActionScript3 settings> and point the path to the directory holding all your classes. (Click the plus symbol to add) In this case, my directory is called "AS3 Classes". IF you have your own computer, you can set this globally by going to FLASH>PREFERENCES>AS 3 SETTINGS. This will set the destination for all files and you won't have to do it for each new project (a good thing) .



Now, you will write your first external class. Create a new as3 file and name it MouseAlpha.as. Save it in your classes directory that you made in step one. I like to include subdirectories to make my classes clearer. You can make, for example, a subdirectory called buttons to hold all classes relating to buttons. (optional). Now, you will write a very simple class that will lower alpha by half on rollover. (see below)

```
package com.brewerthompson.mouse{ ← package name must include classpath

    import flash.display.MovieClip;
    //imports MovieClip class

    import flash.events.*;
    //imports all flash events-- like MouseEvents, for example.

    public class MouseAlpha extends MovieClip{ ← class name must match the constructor name
        //note that this EXTENDS the mc class

        public function MouseAlpha(){ ← constructor name
            buttonMode= true;
            addEventListener(MouseEvent.MOUSE_OVER, onOver);
            addEventListener(MouseEvent.MOUSE_OUT, onOut);
        }
    }
}
```

continued on next page

```

} //ends constructor

private function onOver(event:MouseEvent):void{
    this.alpha=.5;
} //ends onOver

private function onOut(event:MouseEvent):void{
    this.alpha=1;
} //eds onOut

```

I like to use comments to note where my functions, classes and packages end. Remember: curly braces are always in pairs {}. Each opening brace must be followed by an end brace. The package opens with a brace that ends at the end your code. It "wraps" everything else in it. At the next level are the class braces. To look at it very simply, the syntax is like this.

```

package com.brewerthompson.mouse{

    //import statements go up here.

    public class exampleClass extends MovieClip{
        // this is our class body. opens here, ends below. wraps all but the package

        public function exampleClass():void{
            //stuff you want to execute automatically
        } //ends constructor

    } //ends class

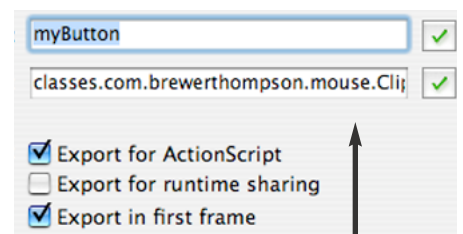
} //ends package

```

Now, write 4 more simple classes : one that rotates a MC on mouseOver, One that makes the MC draggable, one that grows larger on MouseOver and a 4th of your choice. IF your symbol is a movieClip, then the base class has to EXTEND MOVIECLIP or you will get an error. A Name them something sensible and place them in your class directory.

Next: apply these as base classes to items in your library. Make a new Flash file. Make 6 or 7 MovieClips. Change the base class in linkage to point to one of your new classes.

Test your movie. Correct any errors. Have a nice day.



base class path leads to your AS file