Periodic Table Draq and Drop Activity

Introduction to the Activity

This interactive is a drag-and drop-activity geared toward late-middle school, or early high school students in an introductory physical science course. A key learning objective in physical science is for students to understand that the periodic table is organized according to atomic number and atomic mass. By understanding this organizing principal, students should be able to predict where elements would appear on the periodic table. This interactive was created using **Flash CS4** and **Actionscript 3.0**. It incorporates the following techniques:

- Motion tweening (for the introduction)
- Shape tweening (for the introduction)
- Layers
- Actionscripting
    - Drag-and-drop functionality
    - "Snap-back" functionality for the drag-able items
    - Dynamic text-response box
- Embedded audio (a music file for fun)

Getting Started: Setting up the Layers and timeline

This particular interactive will have a number of components, thus careful planning of the timeline is critical. Create layers for:

- The audio file
-The Actionscript
-The background
-Images for the drag-and-drop
-The dynamic text for the drag-and-drop game
-A folder called, "startitems" that will contain layers for the motion and shape tweened items serving as a "fun" introduction for the interactive:

Figure 1a

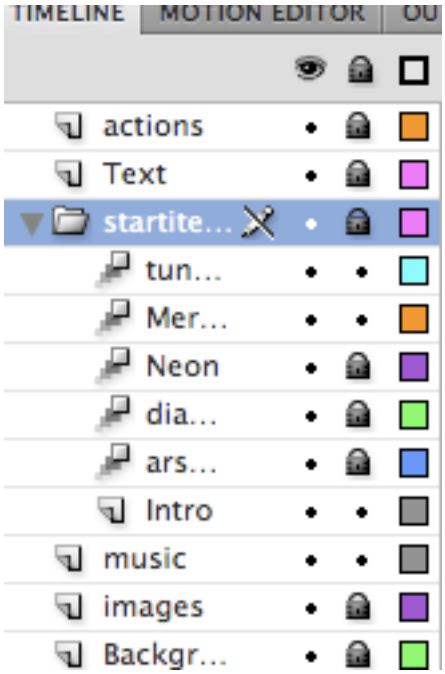


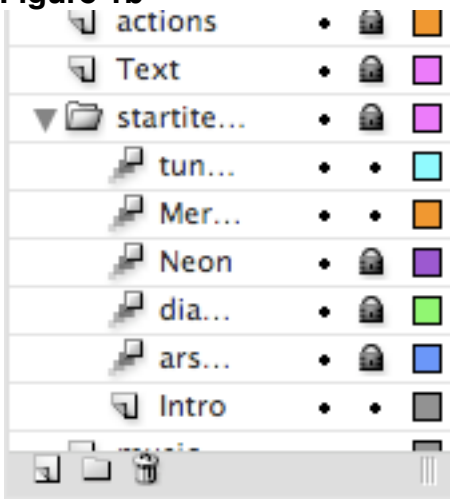
Figure 1a

Using a folder within layers was necessary to keep track of the motion tweened photos of the elements and the shape tweened names of the elements.

I typically lock a layer as soon as I am finished working with it.

Layers were added by going to the bottom of the timeline and selecting the "new layer" icon. The "new folder" icon is located right next to it. Layers can then be dragged and dropped into a folder. You can see these icons at the very bottom (in the gray bar) of Figure 1b.

Figure 1b

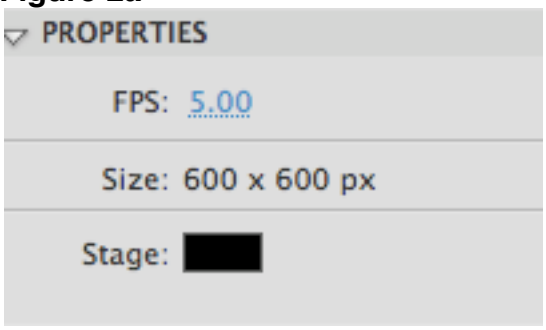


Setting the Stage

The stage area was modified by clicking directly on the stage and choosing the "Priorities" tab. Clicking on the individual items, I was able to customize my file by:

- Expanding the default dimensions
- Changing the background of the screen by selecting a different color
- Changing my frames per rate second

Figure 2a



I imported my background graphic into the library by selecting File>>>Import to Library and choosing the appropriate file. I placed the graphic into its desired location and locked it.

The periodic table didn't need to be visible until the drag-and-drop activity, and thus needed to be obscured for the introduction. I selected the "Intro" layer stored in the "startitems" and right-clicked on the very first square in the timeline. Right-clicking brought up a menu that allowed me to select, "insert keyframe". Once my keyframe was established, I used the rectangle tool and the paint bucket tool to draw a black square to hide the periodic table during the introduction.

Figure 2b

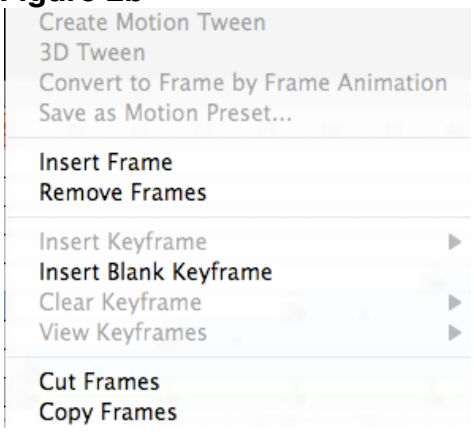


Figure 2c

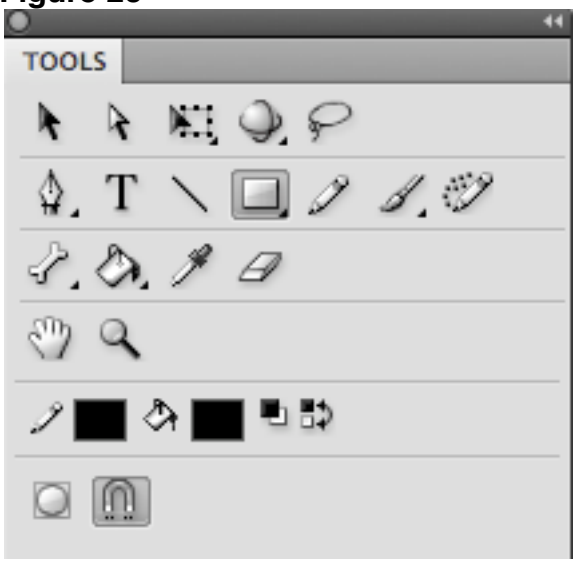


Figure 2c

The rectangle tool is selected in Figure 2b. Below, you can see the paintbucket tool. Click on the color square next to it to select the color

On this same frame, using the text tool, I typed the name of the introduction.

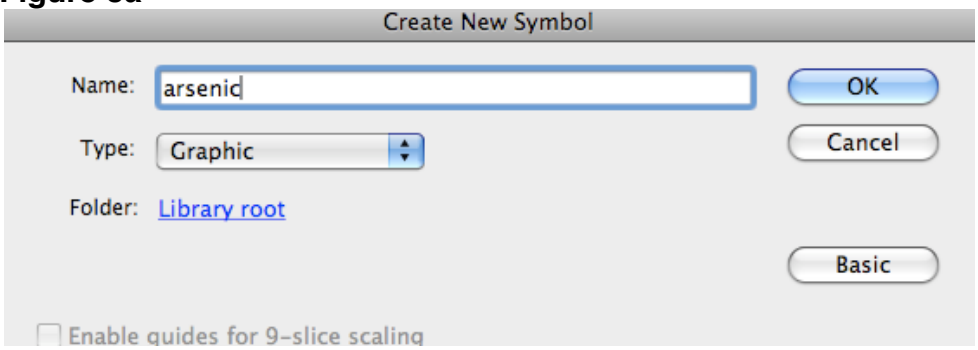
Motion Tweening:

I used photos to represent some of the more visually fascinating elements: neon, carbon, tungsten, and mercury. I imported each of these images into the library by going to:

-File
-Import to Library

and selecting the file name. Each element has a corresponding layer. Selecting that layer (and locking all others) I dragged the appropriate image onto the screen and right-clicked on the item to bring up a drop-down menu that allowed me to select "Convert to symbol". Upon converting the item to a symbol, I was presented with another menu that allowed me to select 1) the type of symbol I wanted (a graphic) and the 2) to name the symbol.

Figure 3a



For each element and (its layer) I selected a keyframe that was approximately 10 frames in from the previous element (the first element was introduced after the title had a little over once-second of screen time). I right-clicked and selected "Insert keyframe" into that particular square. This marked the beginning of that symbol's path.

Next, I selected a keyframe approximately 10 frames over (thus giving the symbol about 2 seconds of travel time) and right-clicked on that frame where I selected the top option, "Motion Tween". The appearance of a faint purple line meant that the motion tween was accepted, and I could use the "select" tool to modify the path of the object.

Figure 3b

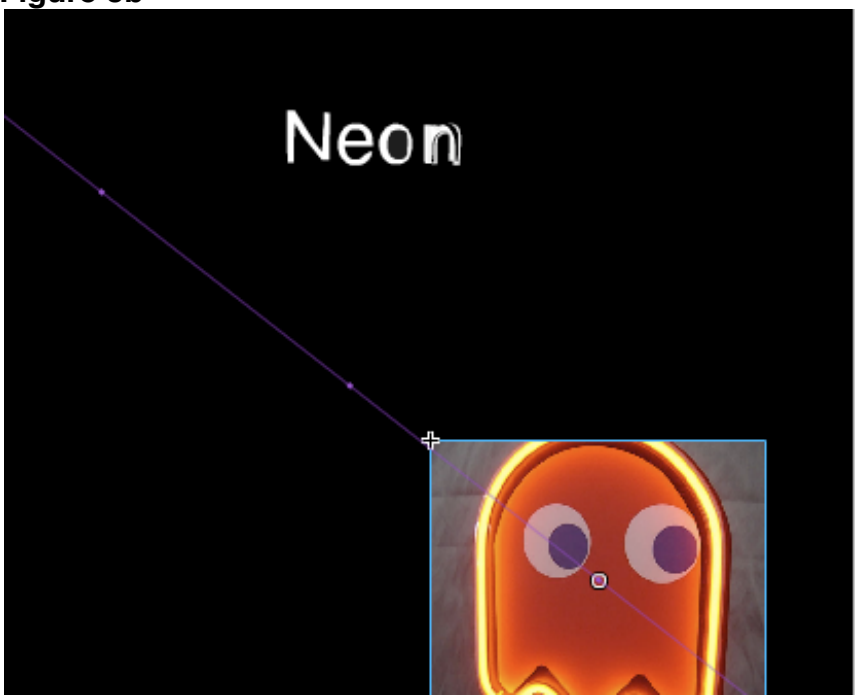
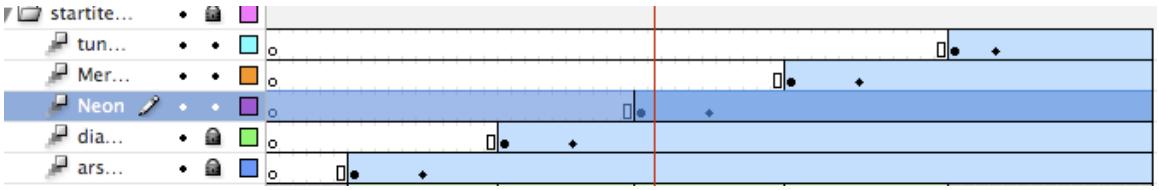


Figure 3b

Using the solid black "select" arrow in the "Tools" menu allows me to modify the path of my Neon symbol. Otherwise, I can just drag my symbol around the stage and influence the path it will take as part of its motion tween.

Figure 3c
Successful motion tweens turn blue on the timeline. The staggered nature of their appearance is readily visible.

Figure 3c



Shape Tweening

I used shape tweening to transition between each of the elements names. These were all done on the same layer ("intro") in a text box that did not change position. Each word was added by right-clicking on the frame where an element first made its appearance, and the selecting "Insert Keyframe" and the textbox was edited using the text tool.

Once the text was written in, I went back and selected the letters in each word. From the main Flash menu, I chose the "Modify" drop-down menu and selected "Break Apart" twice, resulting in the following display:

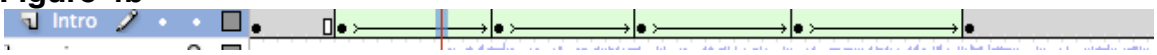
Figure 4a

Figure 4a

It is necessary to select "Break Apart" twice so that the text goes from blue boxes (top) to dots (bottom) to indicate that the text is now ready for shape tweening.

The next step was to go between each of the keyframes denoting where the new words would appear within the "intro" layer. Right-click between each of the keyframes and select "Add shape tween". A successful shape tween addition will become green with an arrow going through the timeline frames and will resemble the image below.

Figure 4b



All layers for the introduction need to have their frames truncated after the final element has been up for 10 frames. In the case of my interactive, all of the layers for the introduction stop at 65 frames.

Adding Audio

The audio component was imported to the library using File>>Import to Library as an MP3 file. Once in the library, I made sure all layers other than the "music" layer were locked and I dragged the file onto the stage. I decided to allow the audio to persist through the entire interactive, and thus let the frames spill over to the next part of the interactive.

Figure 5

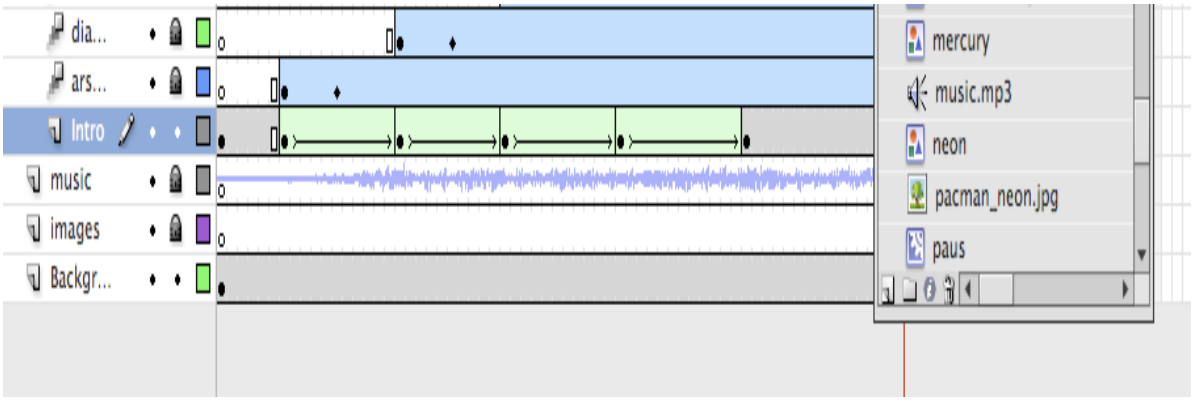


Figure 5

Music as it appears in the timeline and in the library.

Setting up the Drag-and-Drop

For the drag-and-drop section of the game, add a new keyframe to the, until now, quiet layer called, “images” to follow the end of the introduction. This layer will contain our drag and drop elements and our dynamic text box.

There are two main types of graphics I needed to create using Flash drawing tools: “pegs” and “targets”. The targets are four squares representing “holes” in the periodic table and the “pegs” represent the four elements that can be dragged and dropped in.

Figure 6a

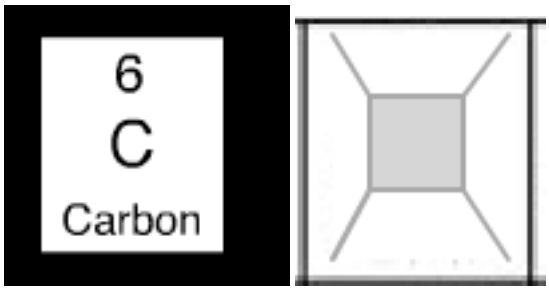


Figure 6a

A “peg” on the left and a “target” on the right.

Each of these items must be converted to a symbol via the Modify>>Convert to Symbol route. These symbols will all be movie clips. Each symbol will receive two names: one name will serve as its general, library name. The second one will be the specific instance name. The instance names can be assigned when the movie clip is dragged onto the stage, the movie clip is selected, and you pick the properties tab:

Figure 6b

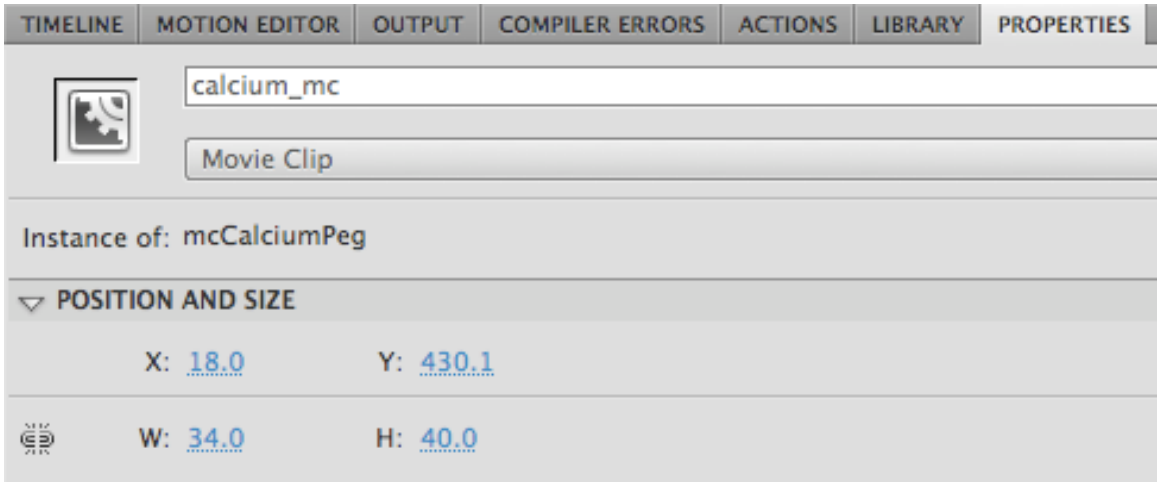


Table 6
Movie clip and instance names for drag-and-drop items

Movie Clip Name	**Instance Name**
mcHeliumPeg	helium_mc
mcHeliumTarget	targethelium
mcCalciumPeg	calcium_mc
mcCalciumTarget	targetcalcium
mcCarbonPeg	carbon_mc
mcCarbonTarget	targetcarbon
mcChlorinePeg	chlorine_mc
mcChlorineTarget	targetchlorine

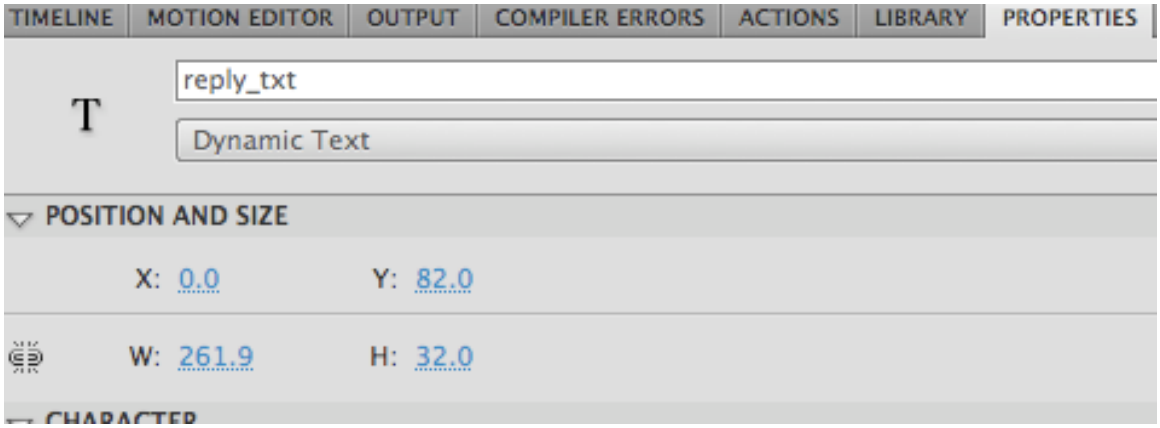
Warning: The text on the pegs had to be selected and "Modify">>"Break Apart" twice in before grouped and converted to a symbol. Otherwise, if a user clicked on the text during the drag-and-drop game, they selected the text rather than the peg. The peg functions much better if the text is broken down and then slowly integrated.

Creating a Dynamic Textbox

Find a spot in your "images" layer that is big enough to hold text. The text will change dynamically depending on the players' actions using Actionscript that will be described in the next step.

Select the text took and draw a box big enough for your text. Enter some general instructions. Next, select this box and go to the "Properties" pane. There you will see the option to name your textbox. For this interactive, I named it replay_txt. Right below the naming option is a drop-down menu. Selecting "Dynamic Text" from this menu tells Flash that we want this text to be able to change.

Figure 7

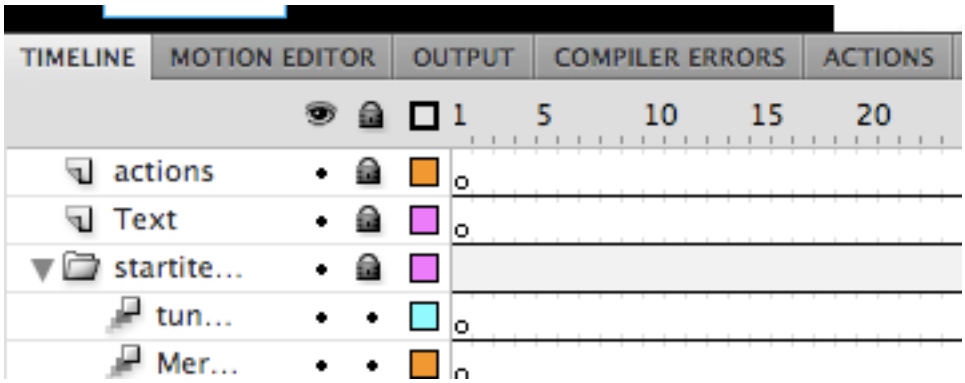


s

Actionscript for the Drag-and-Drop and Dynamic Text

In the final frame of the timeline, select the actions layer. At the top of the timeline, you will see the actions tab. Select the tab:

Figure 8



This will open the actions pain. Below, in blue, is the code for the interactive. Comments about the code will appear in red through out the code. These comments are not part of the code.

```
stop()

//This string prevents the Flash movie from running through this frame and
treating it as part of the animation.  It tells the movie to basically stop here and
allow the user to do all the good things below:

var startX:Number;
var startY:Number;
var counter:Number = 0;

// This code tells the movie that each of the pegs has a start position.  Later in the
code, I can tell the pegs to go back to this start position if the player does not get
the peg into its proper target.

helium_mc.addEventListener(MouseEvent.MOUSE_DOWN, pickUp);
helium_mc.addEventListener(MouseEvent.MOUSE_UP, dropIt);
carbon_mc.addEventListener(MouseEvent.MOUSE_DOWN, pickUp);
carbon_mc.addEventListener(MouseEvent.MOUSE_UP, dropIt);
chlorine_mc.addEventListener(MouseEvent.MOUSE_DOWN, pickUp);
chlorine_mc.addEventListener(MouseEvent.MOUSE_UP, dropIt);
calcium_mc.addEventListener(MouseEvent.MOUSE_DOWN, pickUp);
calcium_mc.addEventListener(MouseEvent.MOUSE_UP, dropIt);

//These are the instructions that tell the movie that the mouse will control picking
up and dropping the pegs.


function pickUp(event:MouseEvent):void {
        event.target.startDrag();
        reply_txt.text = "";
        event.target.parent.addChild(event.target);
        startX = event.target.x;
        startY = event.target.y;
}
function dropIt(event:MouseEvent):void {
        event.target.stopDrag();
        var myTargetName:String = "target" + event.target.name;
        var myTarget:DisplayObject = getChildByName(myTargetName);
        if (event.target.dropTarget != null && event.target.dropTarget.parent ==
myTarget){
                reply_txt.text = "Wow!";
                event.target.removeEventListener(MouseEvent.MOUSE_DOWN,
pickUp);
```

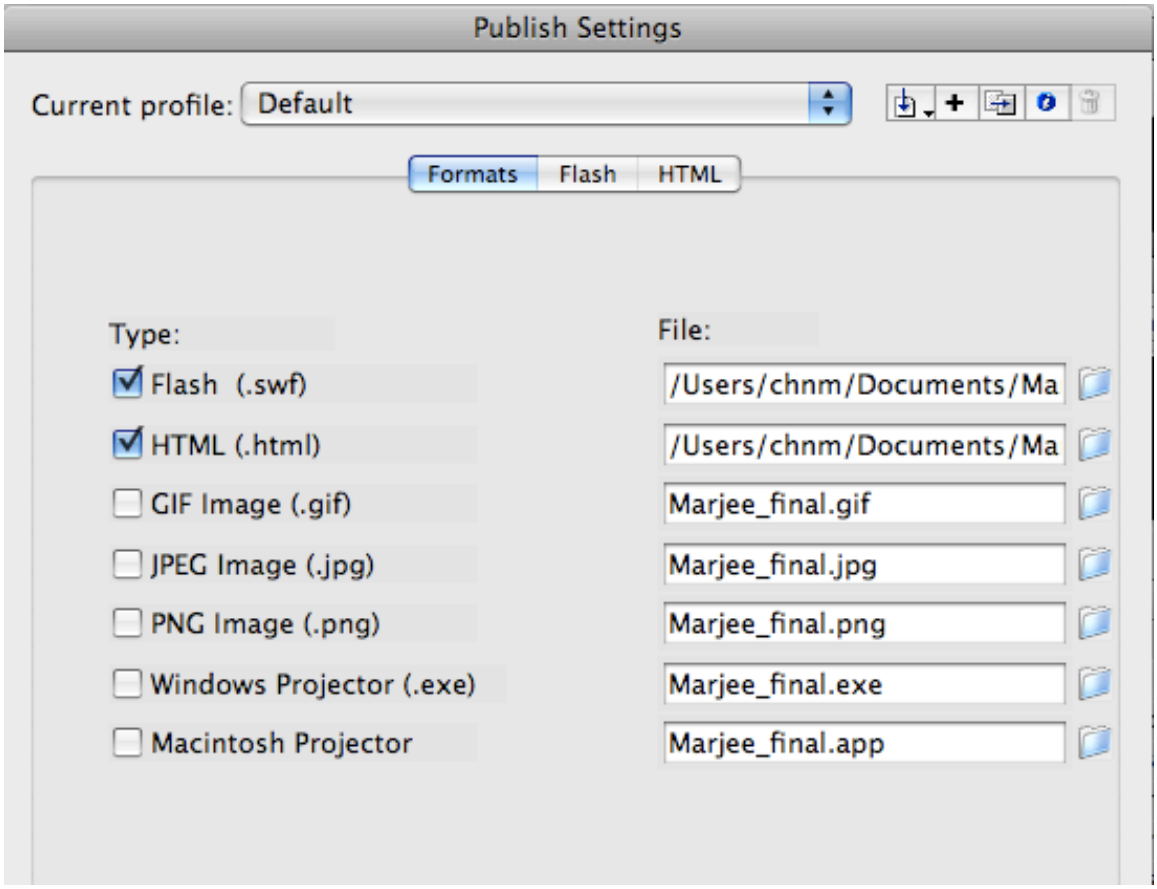
```
                event.target.removeEventListener(MouseEvent.MOUSE_UP,
dropIt);
                event.target.buttonMode = false;
                event.target.x = myTarget.x;
                event.target.y = myTarget.y;
                counter++;
        } else {
                reply_txt.text = "Nope";
                event.target.x = startX;
                event.target.y = startY;
        }
        if(counter == 4){
      reply_txt.text = "Outstanding!";
   }

//These items instruct the dynamic text on how to provide feedback to players
depending on whether the player is right or wrong.  Note the "if(counter)
command can have its total number changed to accommodate the quantity of
questions handled by the drag-and-drop.

}

helium_mc.buttonMode = true;
carbon_mc.buttonMode = true;
chlorine_mc.buttonMode = true;
calcium_mc.buttonMode = true;
```

Find the blue check-mark button at the top of our Actions pane and select it when you are done. IT will let you know if your code contains any syntax errors.

Publishing

Prior to publishing your work, be sure to take a look at how it is operating in the browser window. Go to File>>Publish Preview and select the default html view. Make sure there are no errors displayed in your Actions window. If any movie clips are mislabeled, it may break your Actionscript code entirely and your drag-and-drop won't work. You should also review the pace of the animations and make sure they make sense to you.

To publish the document, go to File>>Publish Settings where you can direct your file to the proper folder and select the file formats you will be publishing in:



References:

Lunn, G. (2007). Flash Drag and Drop | monkeyflash.com. Retrieved August 4, 2009: http://monkeyflash.com/tutorials/flash-drag-and-drop/.