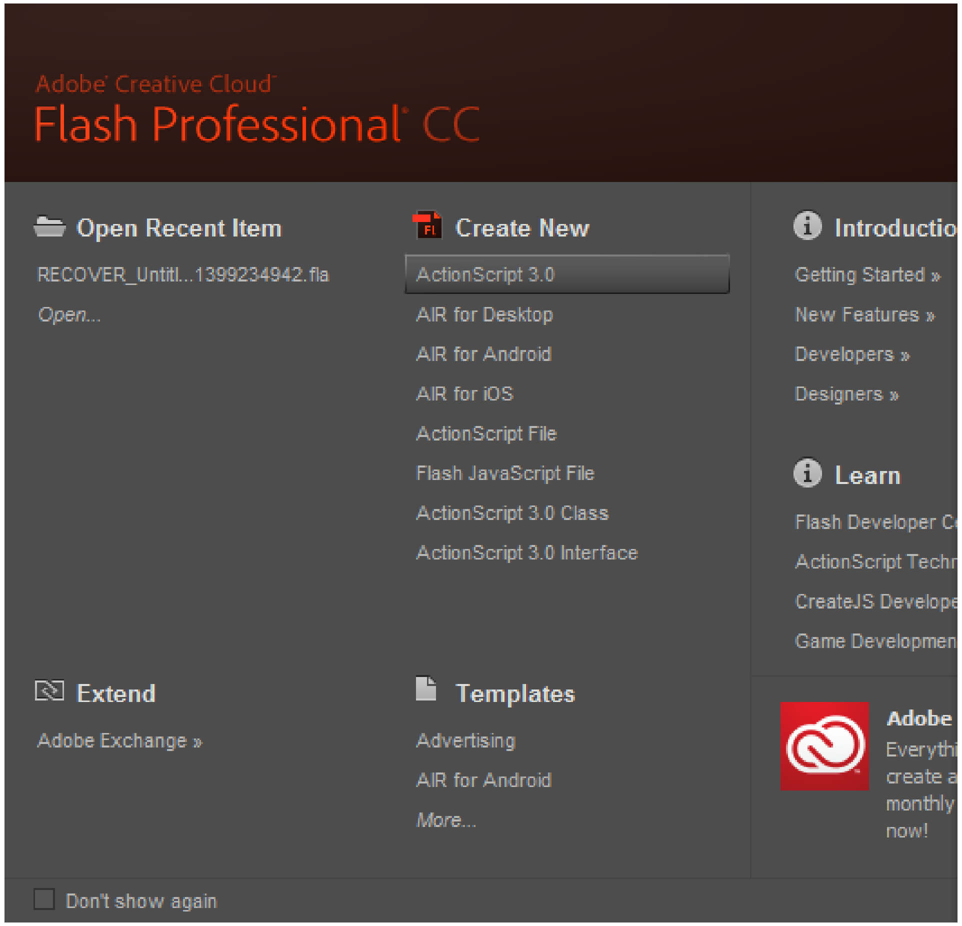
Creating a Game 2

October 8, 2013

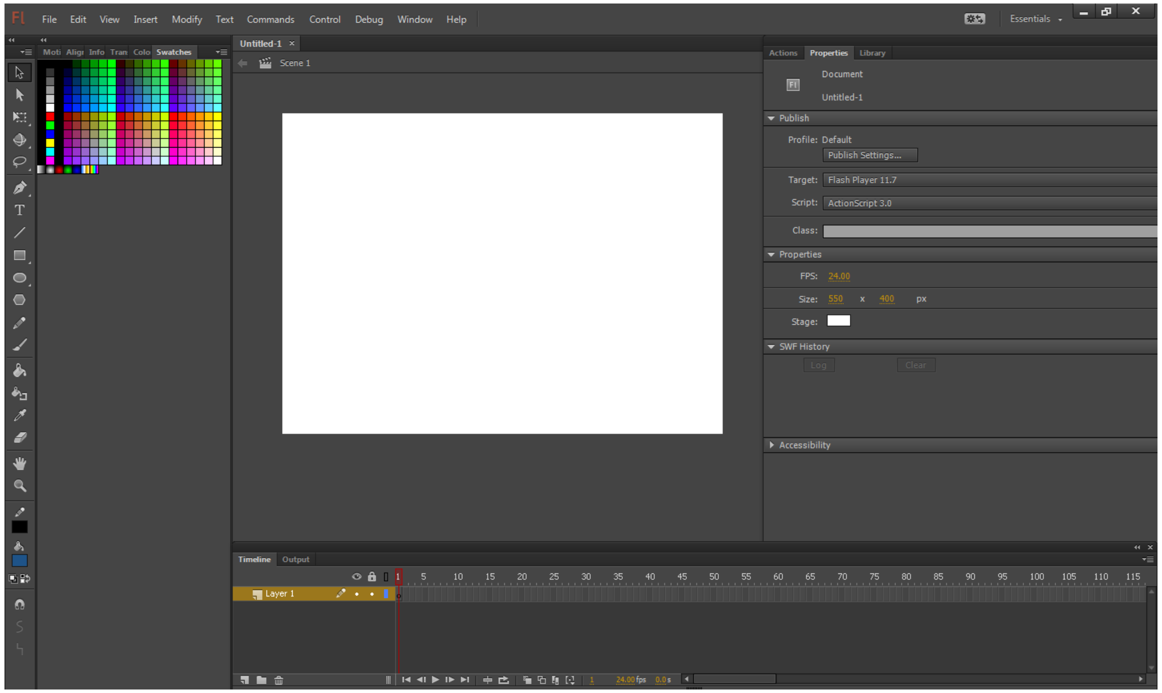
Open the Flash Program

Open the Adobe Flash Professional program and then we want to choose ActionScript 3.0 under the Create New section of the start menu.



The Flash Program

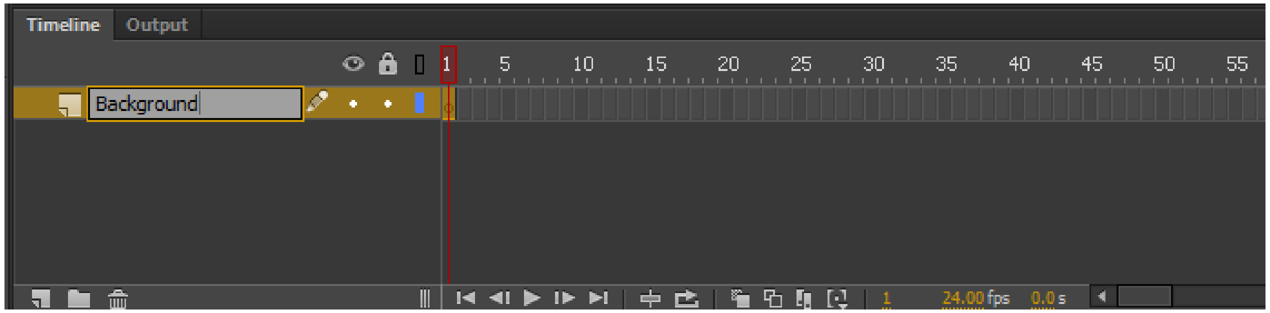
We can see the white Flash stage in the center pane. On the left, the Tools toolbar is seen along with the Swatches window. On the right of the center stage, we have the Actions, Properties and Library window. Under the stage, we see the Timeline.



Profile: Default
Target: Flash Player 11.7
Script: ActionScript 3.0
Class:
FPS: 24.00
Size: 550 x 400 px
Stage:

Renaming a Layer

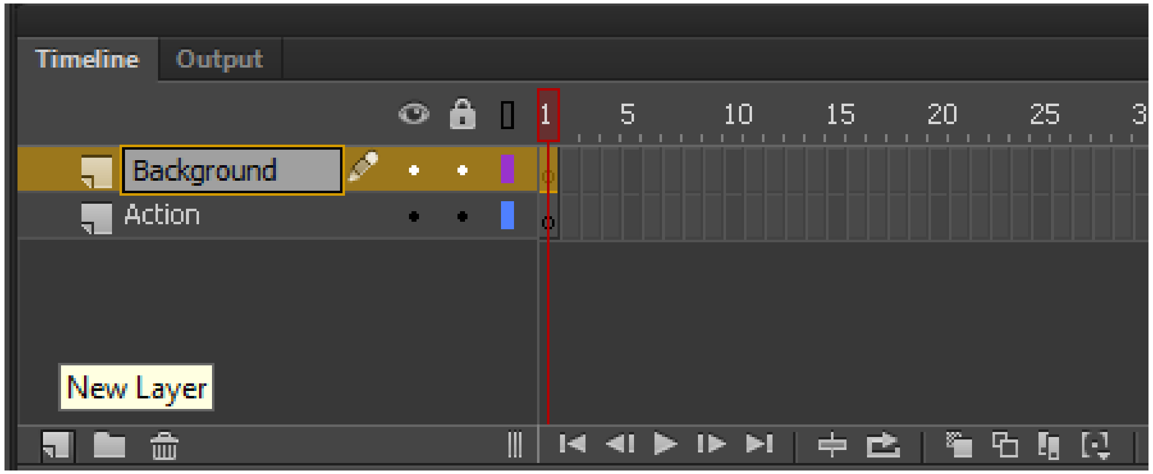
We will rename the layer called Layer1 to Background.

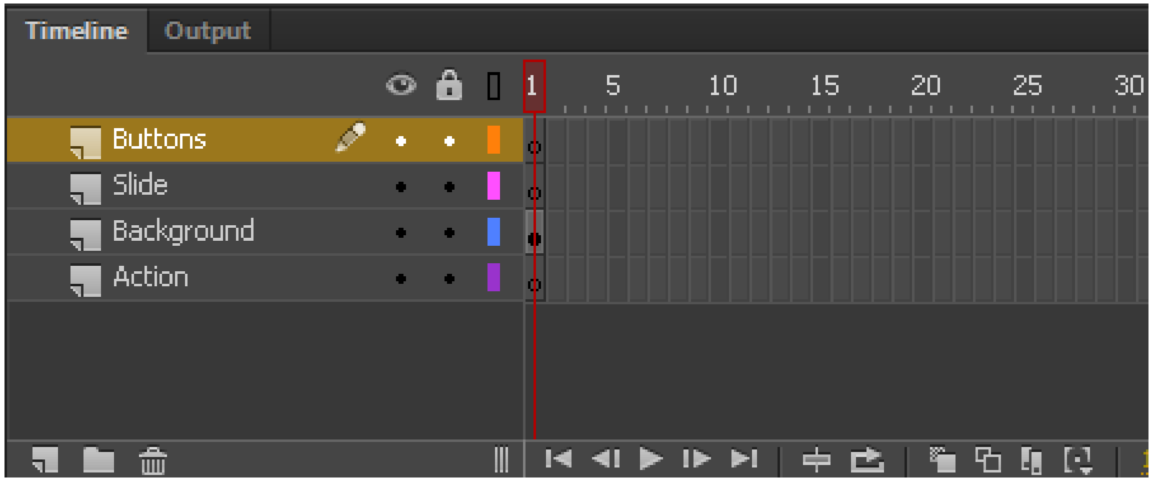


Creating a New Layer

Select the New Layer icon and when it appears in the list, we will rename it to Action.

We then add the layer called Slide and Buttons.



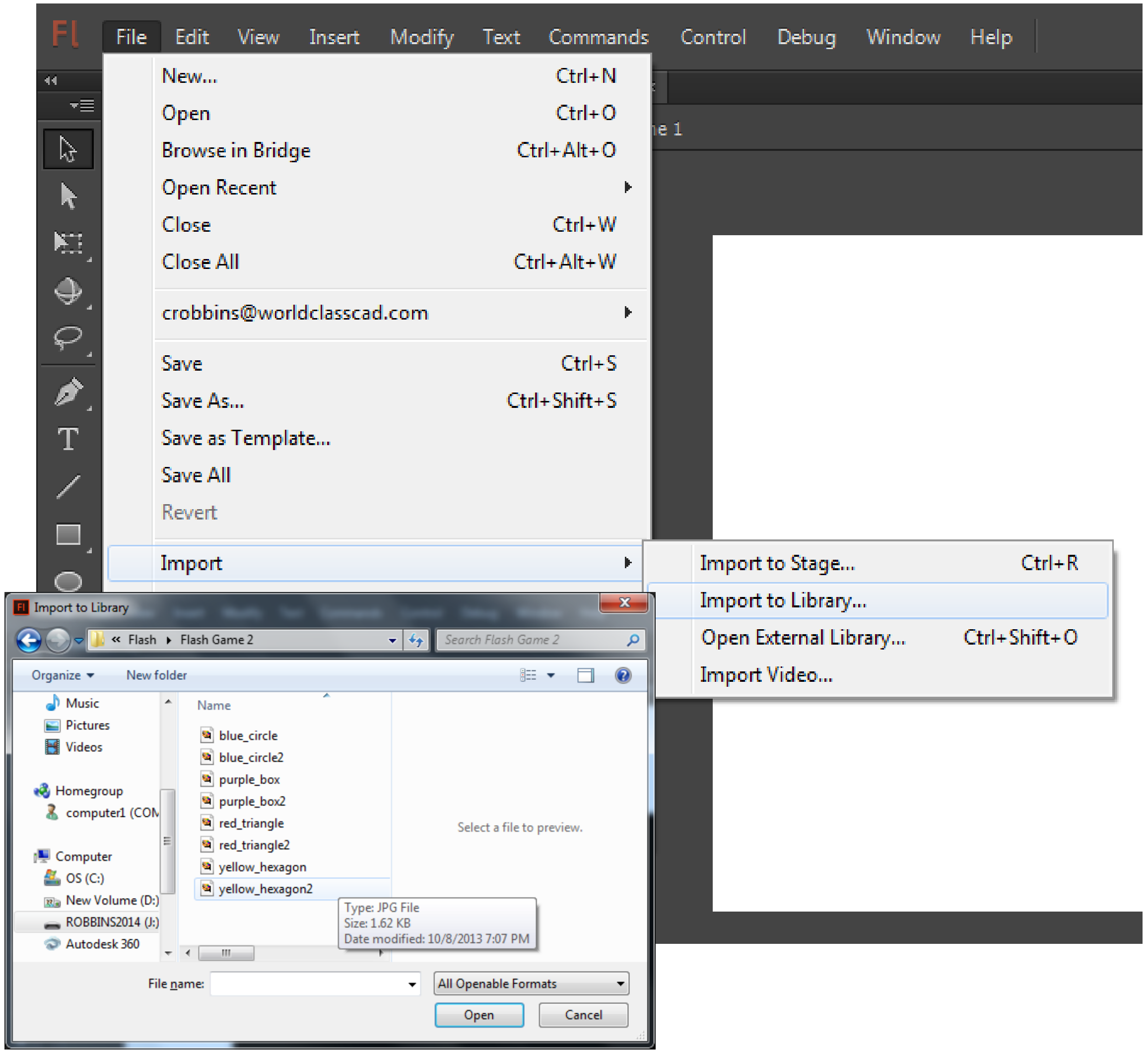


Adding Images to the Library

We made eight images that are on a 70 by 70 pixel white background. They are an blue_circle1 and 2, purple_box1 and 2, red_triangle1 and 2 and yellow_hexagon1 and 2.

To add the images to the Library, we will import them. We should choose File on the top menu, then Import and Import to Library.

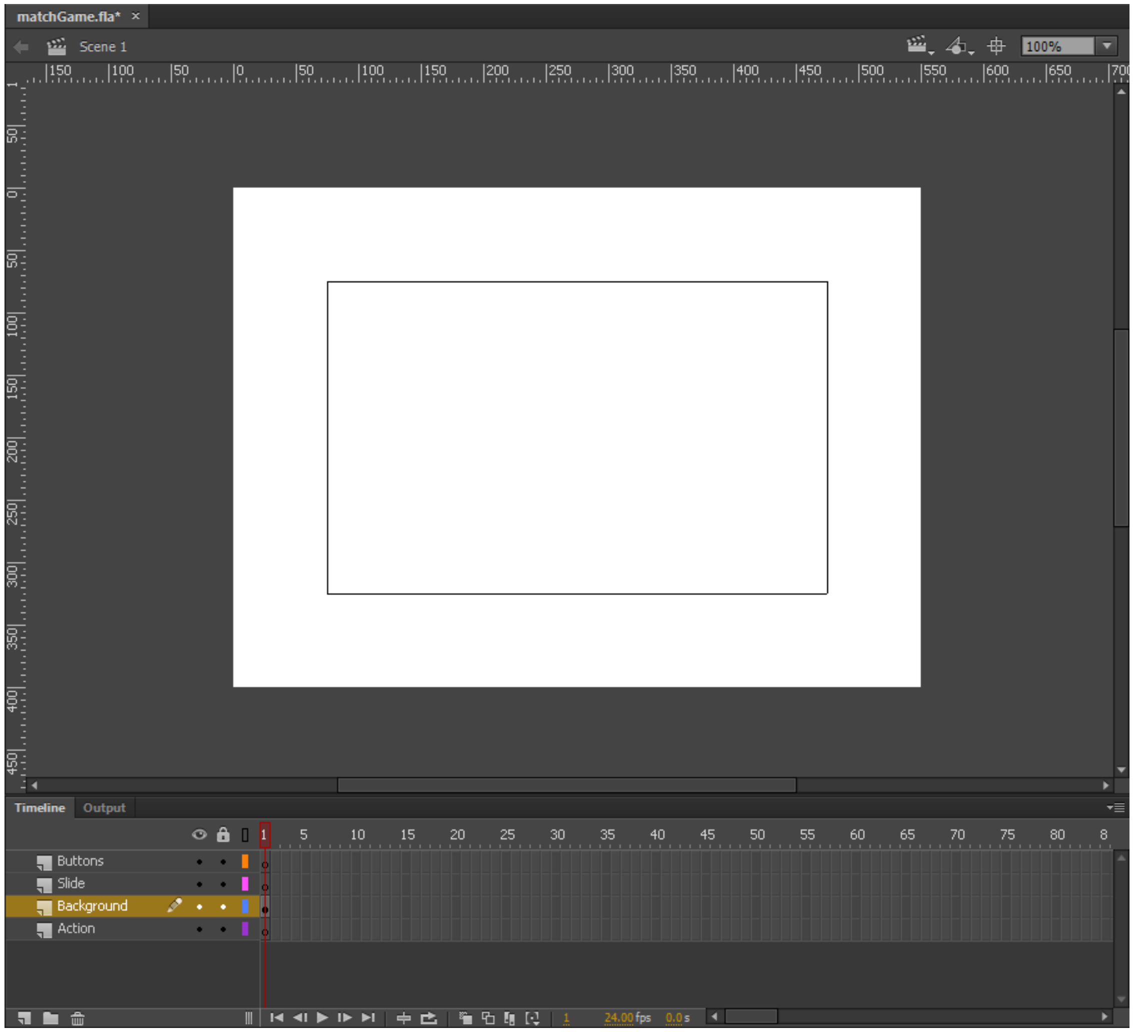
We select the eight images from the computer's folder and press Open



Type: JPG File
Size: 1.62 KB
Date modified: 10/8/2013 7:07 PM

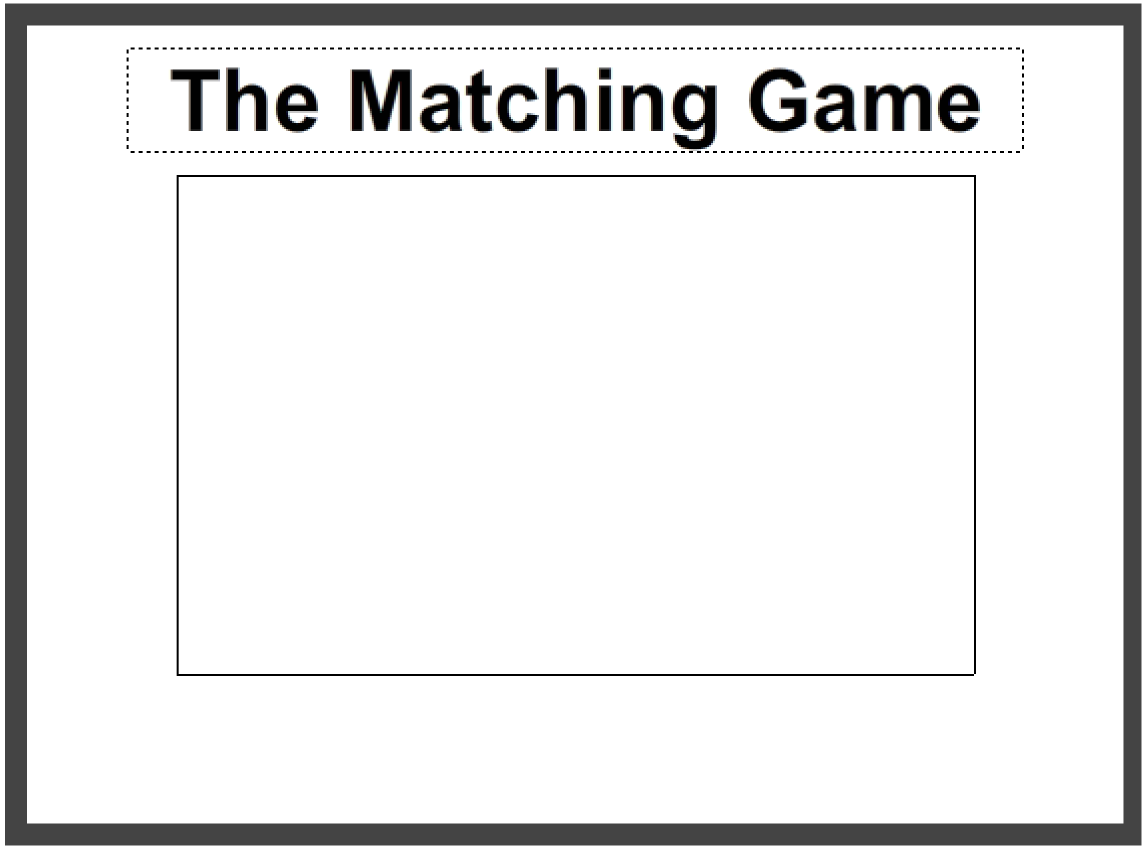
Adding Lines to the Stage

At the beginning of the game, on the background layer, we will add a black rectangle that is inserted at 75, 75 on the x and y axis and it is 400 pixels wide and 250 pixels tall.



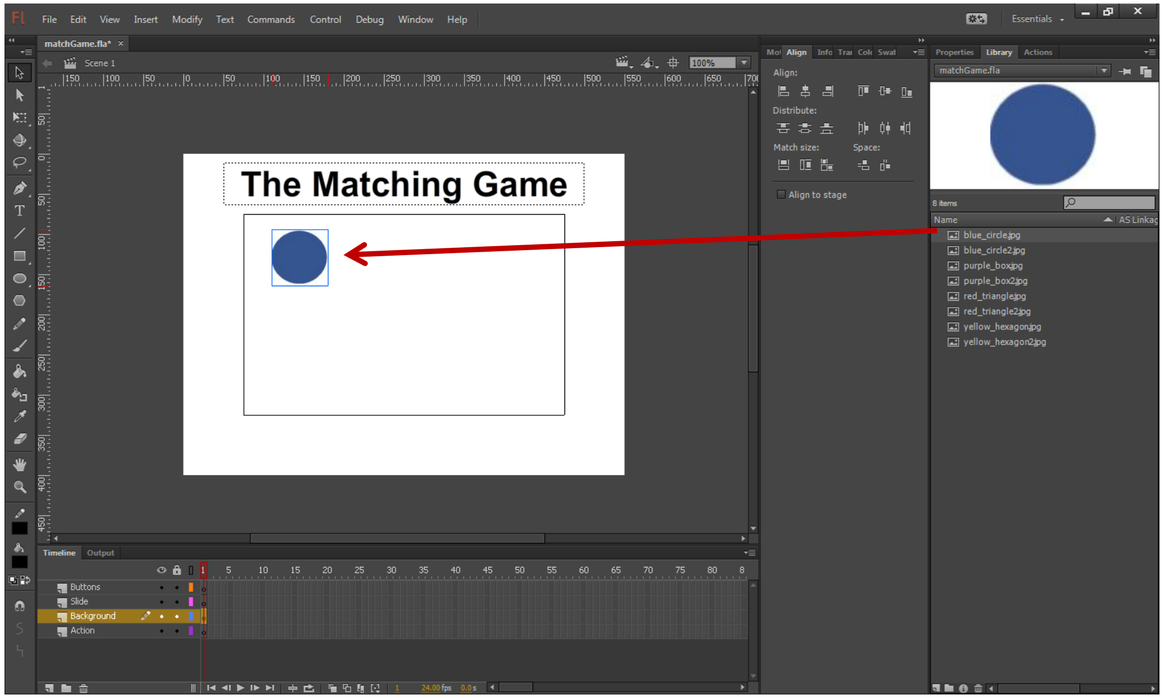
Adding Text to the Stage

We will then add text for the game name on the background layer. This will be The Matching Game.



Adding an Image to Button

Highlight the buttons layer and click on frame 1. We add the first image from Library to the first frame by pulling the image from the Library list to the stage.

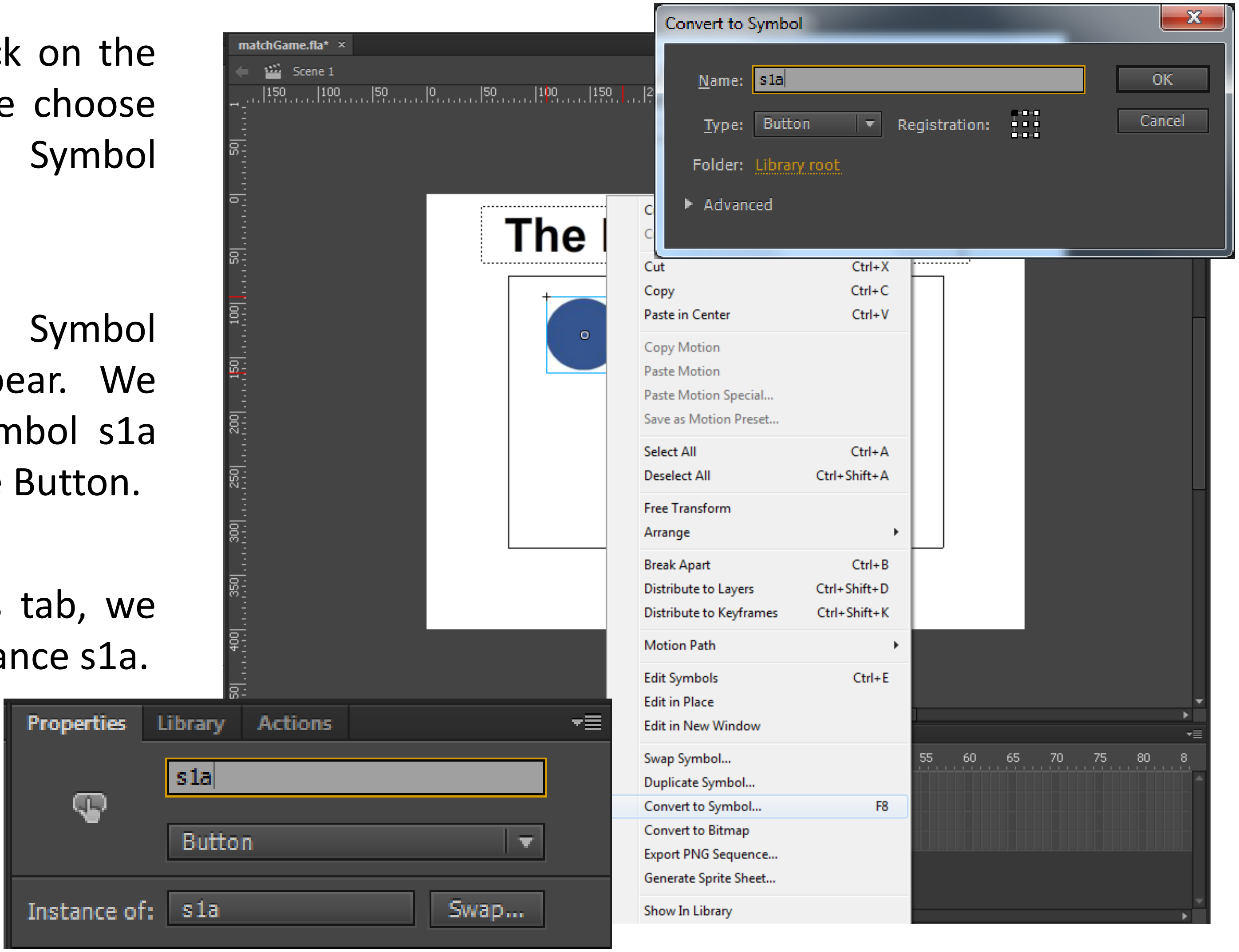


Convert the Image to a Button

We then right click on the blue circle and we choose the Convert to Symbol from the menu.

The Convert to Symbol window will appear. We will name the symbol s1a and make the type Button.

On the Properties tab, we will name the Instance s1a.



Convert to Symbol

Name: s1a

Type: Button

Registration:

Folder: Library root

s1a

Button

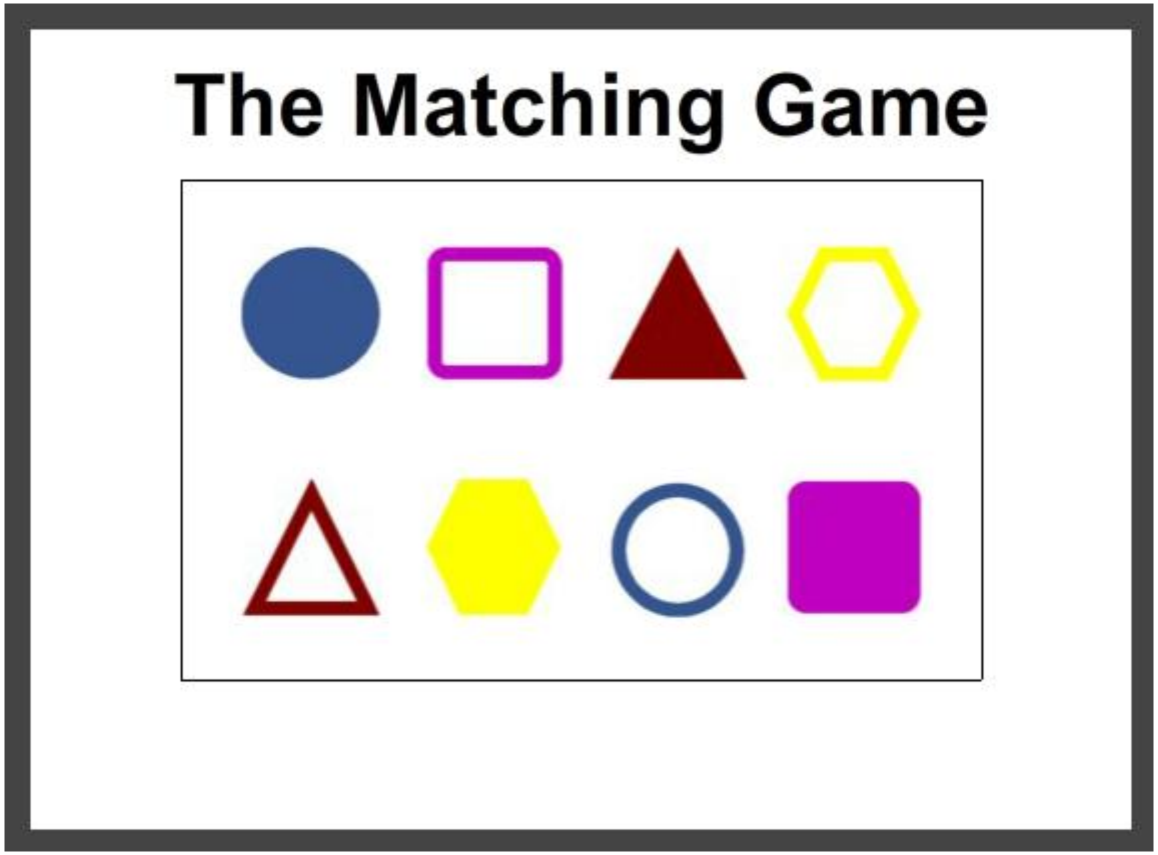
Instance of: s1a

Convert all the X Images to Buttons

We then add the other 7 images.

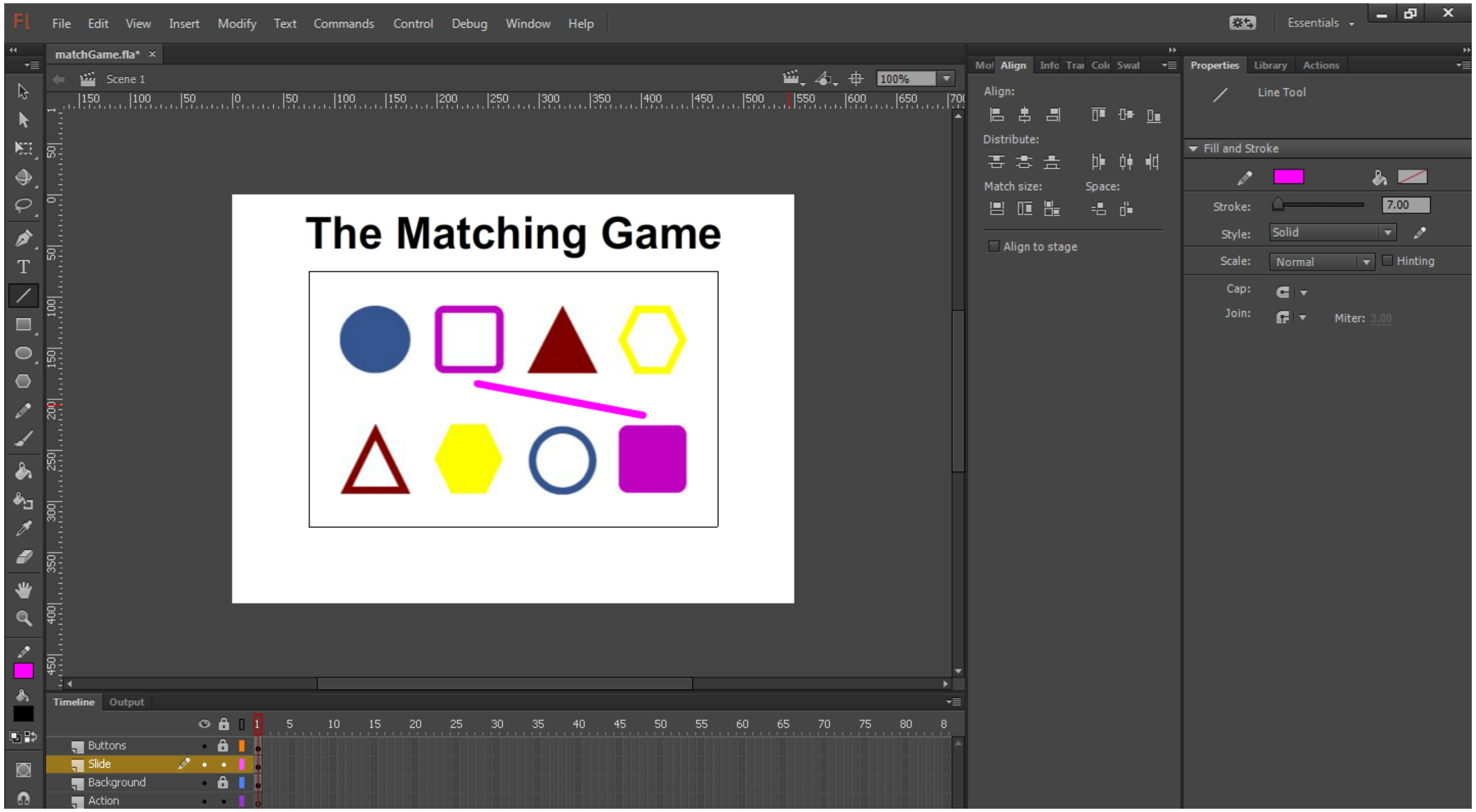
The Convert to Symbol window will appear. The symbols names will go from s1a to s1d on the top row to s1e to s1h on the bottom row. We will make the type Button.

On the Properties tab, we will name the Instance the same as the symbol name.



Adding a Line

Highlight the Slide layer on frame 1. Select the Line tool and set the Stroke height to 7. Set the line color to purple. Draw the line from the two square boxes.



Line Tool

Fill and Stroke

Stroke: 7.00

Style: Solid

Scale: Normal

Hinting

Cap:

Join:

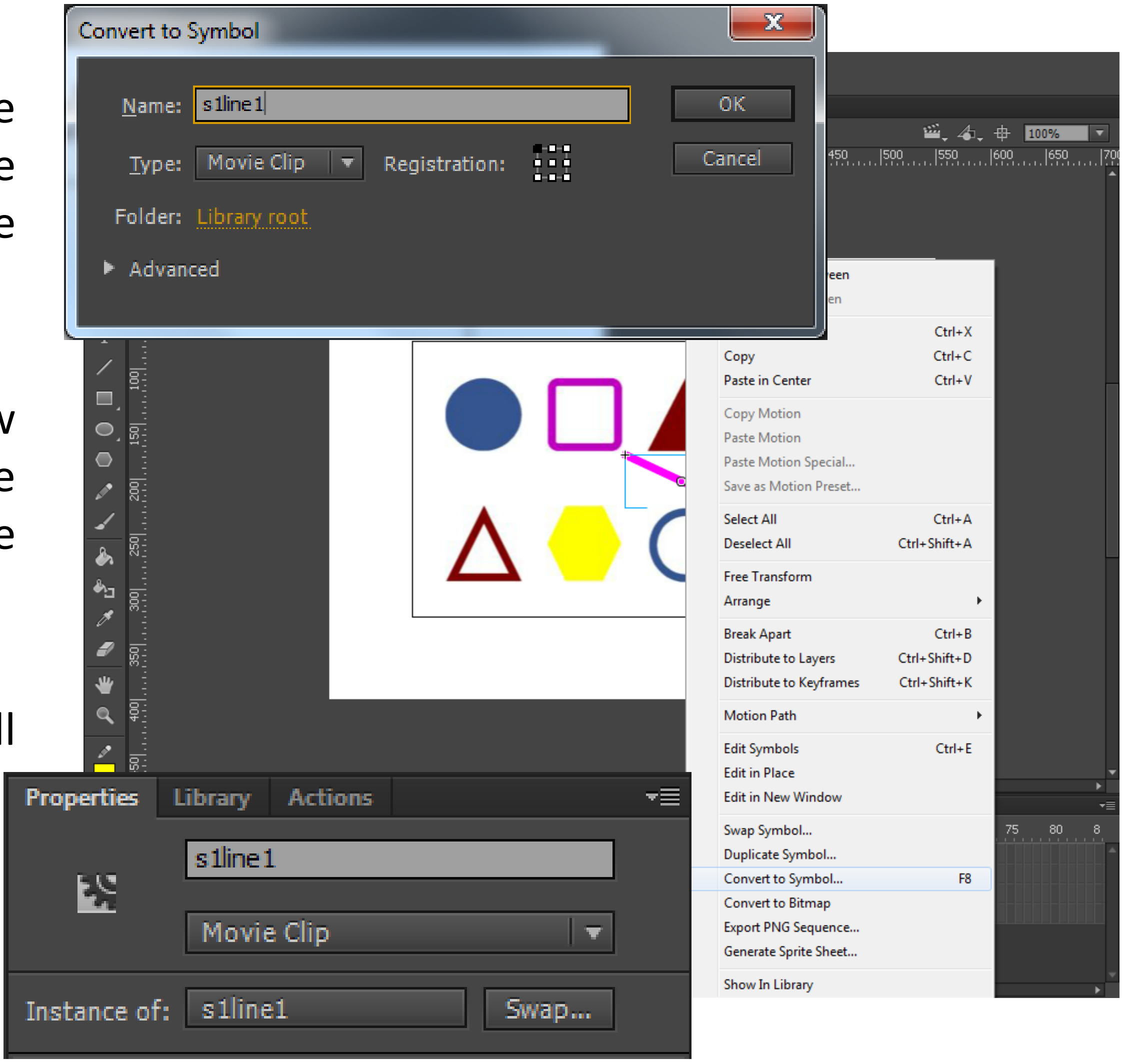
Miter: 3.00

Convert the O Image to a Button

We then right click on the purple line and we choose the Convert to Symbol from the menu.

The Convert to Symbol window will appear. We will name the symbol s1line1 and make the type Button.

On the Properties tab, we will name the Instance s1line1.



Convert to Symbol

Name: s1line1

Type: Movie Clip

Registration:

Folder: Library root

Advanced

OK

Cancel

Properties | Library | Actions

s1line1

Movie Clip

Instance of: s1line1

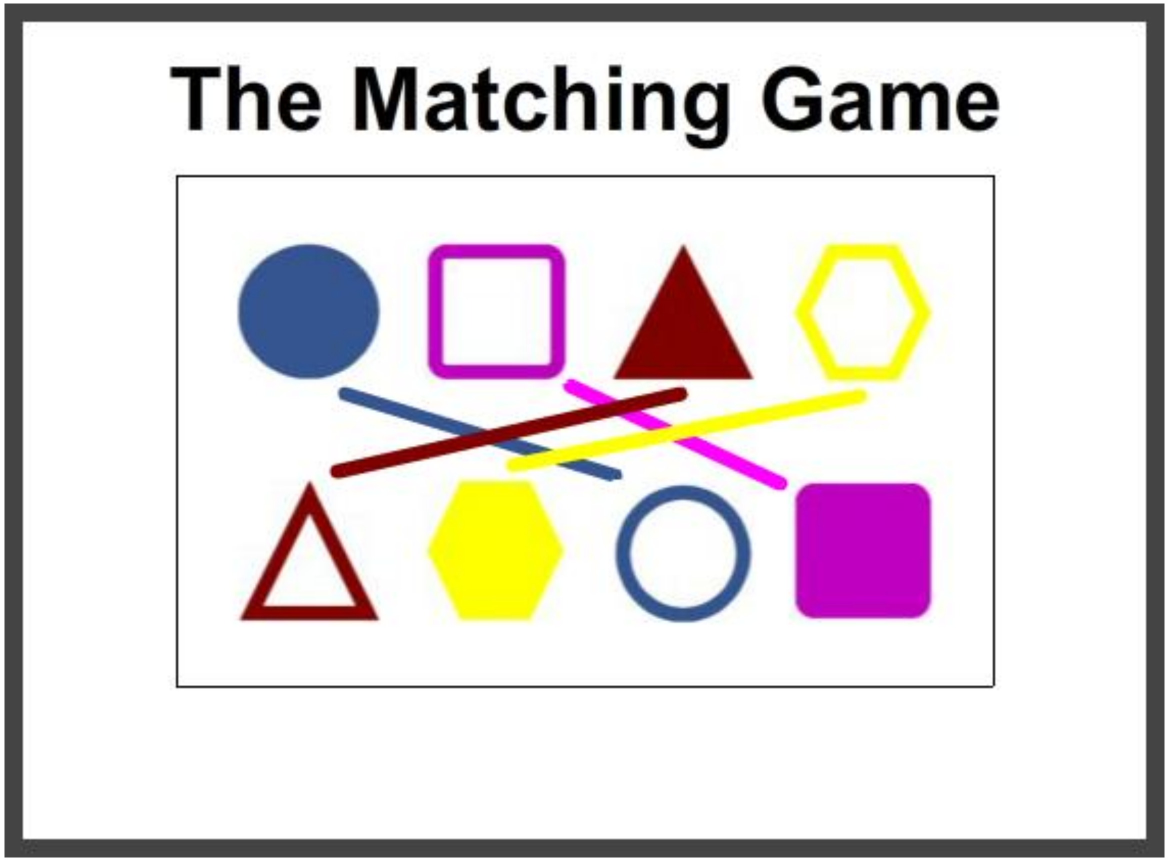
Swap...

Convert all the O Images to Buttons

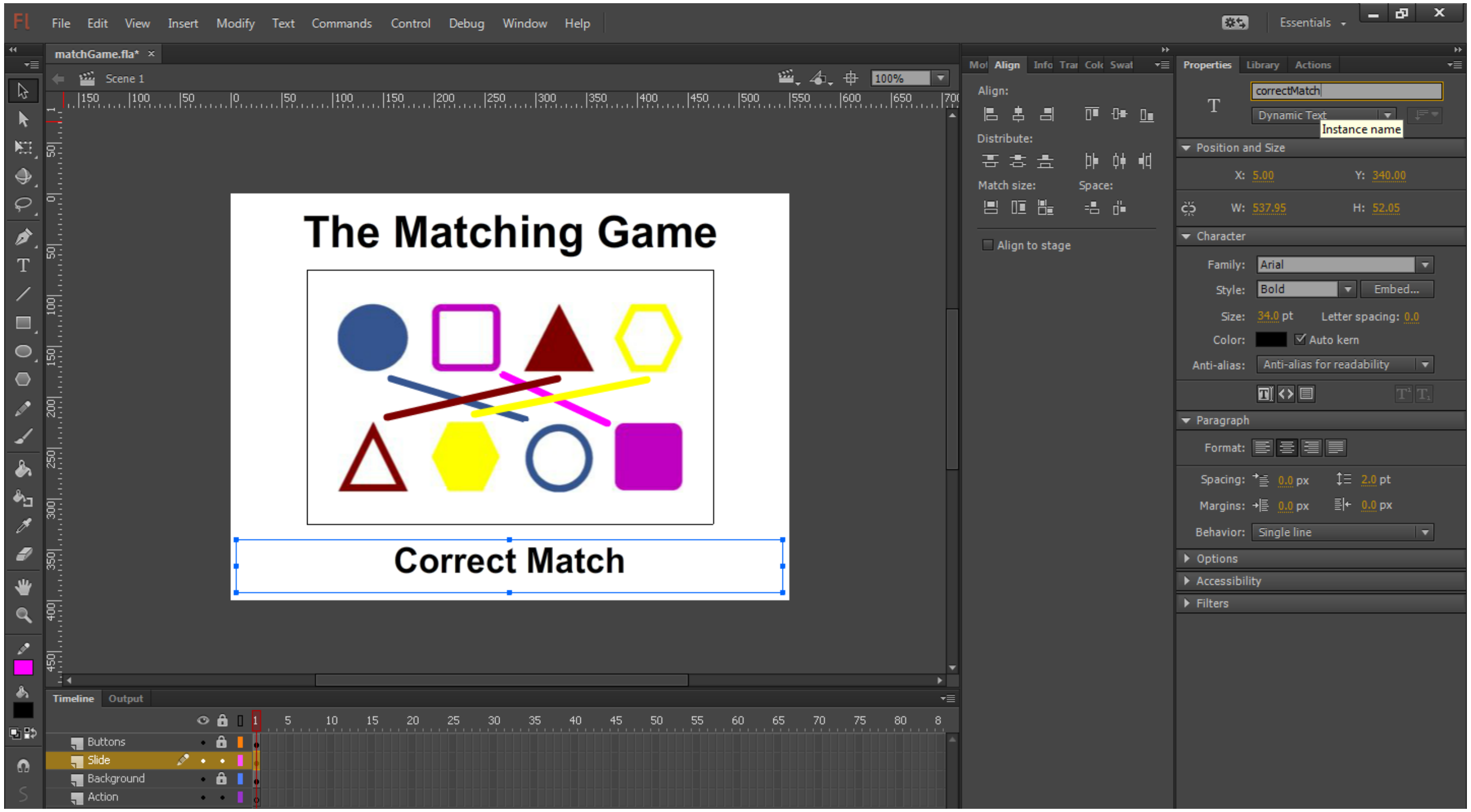
We then add the other 8 o's. The numbers go o1 to o3 on the top row, o4 to o6 on the middle row and o7 to o9 on the bottom row.

The Convert to Symbol window will appear. We will name the symbol the appropriate o and space number and make the type Button.

On the Properties tab, we will name the Instance the same as the symbol name.



Create a Dynamic Text


File Edit View Insert Modify Text Commands Control Debug Window Help
Essentials
matchGame.fla*
Scene 1
100%
The Matching Game
Correct Match
Align: Distribute: Match size: Space: Align to stage
Properties Library Actions
Instance name
Timeline Output
Buttons
Slide
Background
Action

correctMatch
Dynamic Text

Position and Size
X: 5.00 Y: 340.00
W: 537.95 H: 52.05

Character
Family: Arial
Style: Bold Embed...
Size: 34.0 pt Letter spacing: 0.0
Color: ■ ✓ Auto kern
Anti-alias: Anti-alias for readability

Paragraph
Format:
Spacing: 0.0 px 2.0 pt
Margins: 0.0 px 0.0 px
Behavior: Single line

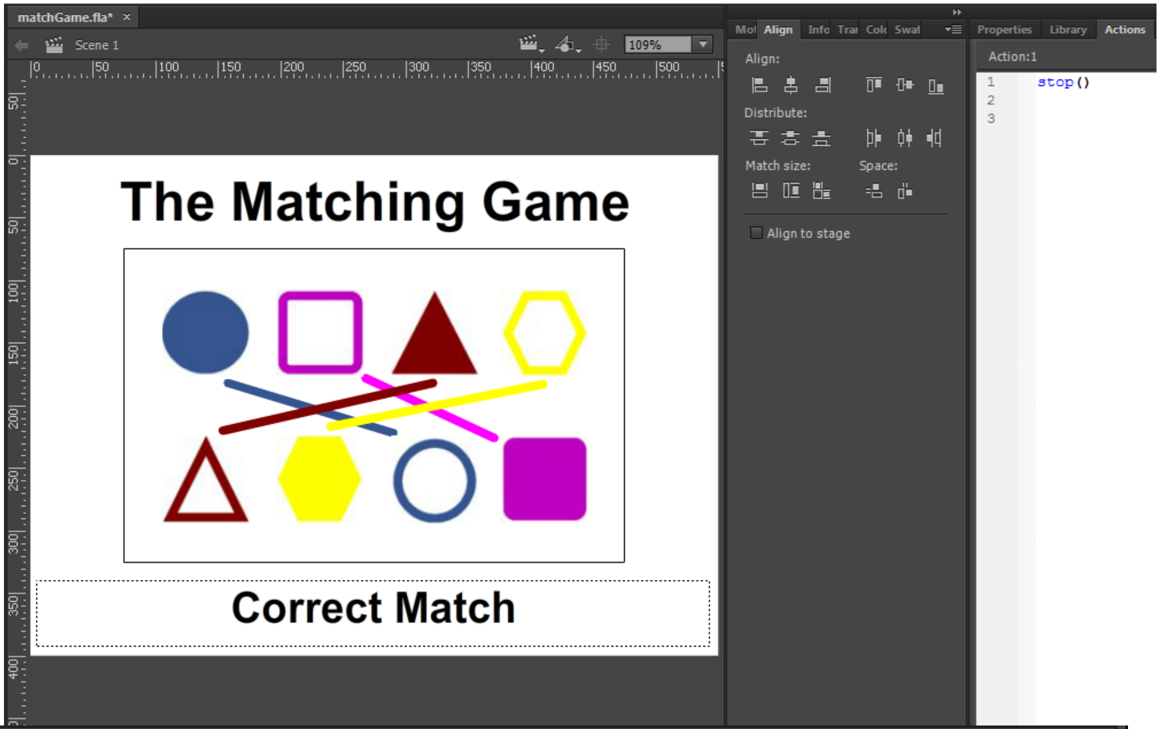
Options
Accessibility
Filters

On the Slide layer, we will add text “Correct Match” to the bottom of the stage. On the Properties tab, we will change the text type to Dynamic Text and the Instance name to correctMatch.

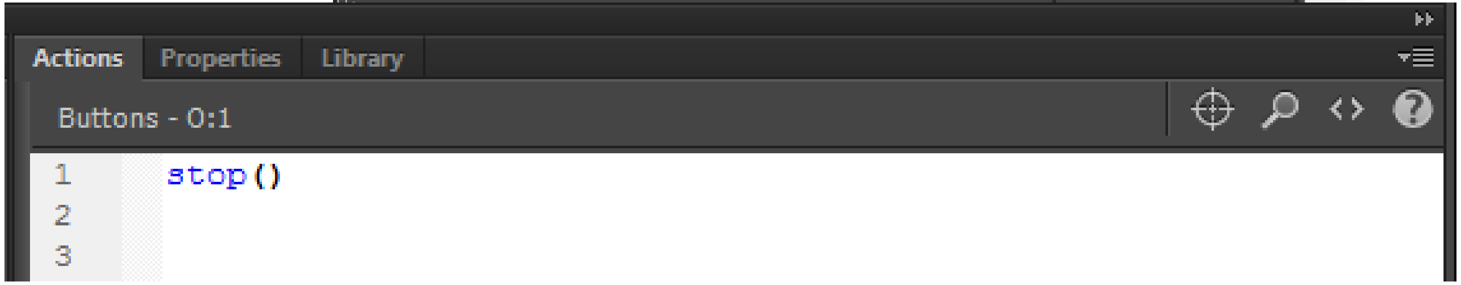
ActionScripts

We choose frame 1 of the Action layer. Then we select the Action tab and we type stop() on line 1 of the script.

This will stop the flash movie at the frame.

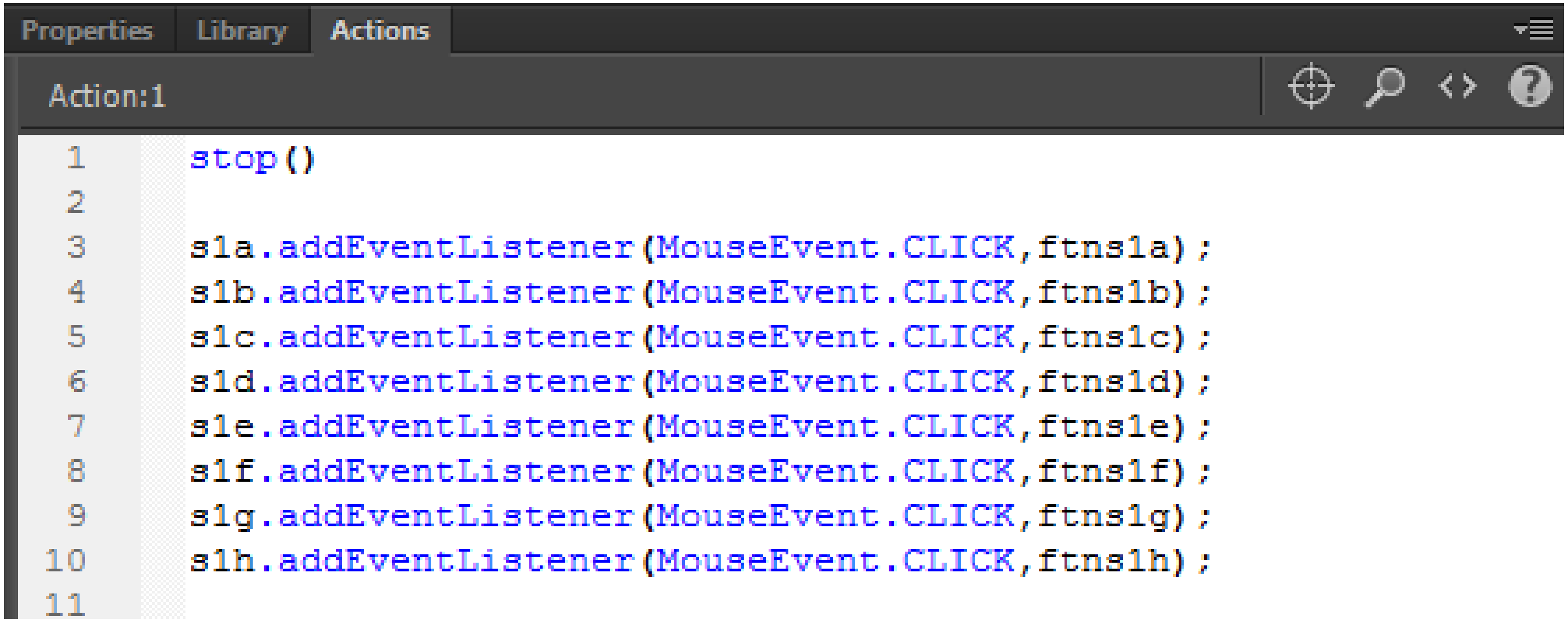


```
stop()
```



```
stop()
```

Add another Event Listener



```
stop()

s1a.addEventListener(MouseEvent.CLICK,ftns1a);
s1b.addEventListener(MouseEvent.CLICK,ftns1b);
s1c.addEventListener(MouseEvent.CLICK,ftns1c);
s1d.addEventListener(MouseEvent.CLICK,ftns1d);
s1e.addEventListener(MouseEvent.CLICK,ftns1e);
s1f.addEventListener(MouseEvent.CLICK,ftns1f);
s1g.addEventListener(MouseEvent.CLICK,ftns1g);
s1h.addEventListener(MouseEvent.CLICK,ftns1h);

```

We then add an event listener to determine when a select button was clicked. We type:

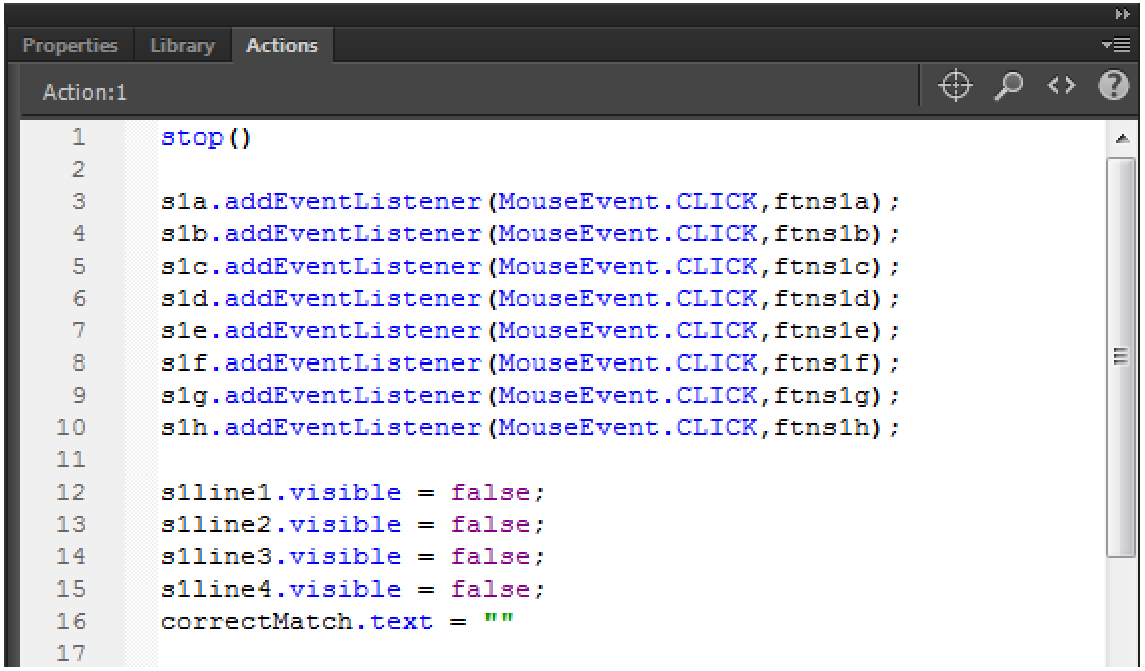
s1a.addEventListener(MouseEvent.CLICK,ftns1a);

Make the Matching Lines Invisible

We then type

s1line1.visible = false; to make the line invisible. We do this for matching line symbol.

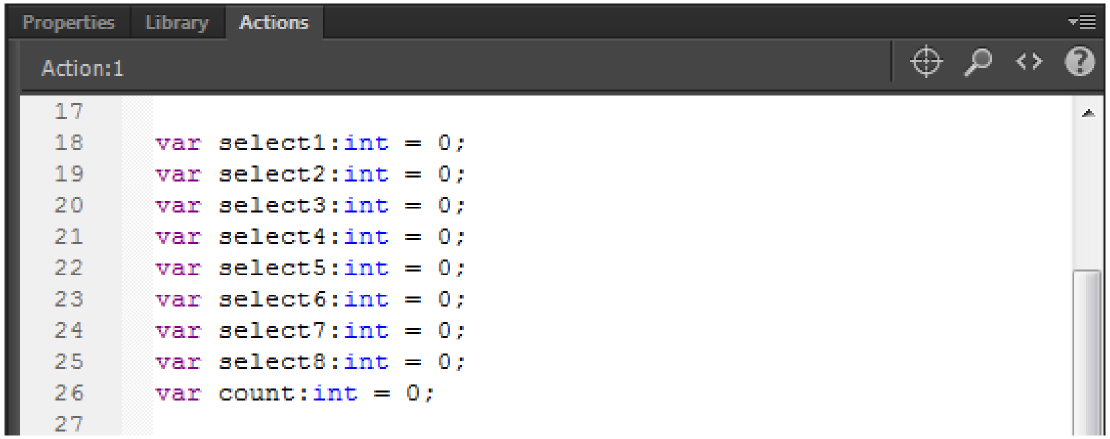
We will also blank out the correct match text.



```
stop()

s1a.addEventListener(MouseEvent.CLICK,ftns1a);
s1b.addEventListener(MouseEvent.CLICK,ftns1b);
s1c.addEventListener(MouseEvent.CLICK,ftns1c);
s1d.addEventListener(MouseEvent.CLICK,ftns1d);
s1e.addEventListener(MouseEvent.CLICK,ftns1e);
s1f.addEventListener(MouseEvent.CLICK,ftns1f);
s1g.addEventListener(MouseEvent.CLICK,ftns1g);
s1h.addEventListener(MouseEvent.CLICK,ftns1h);

s1line1.visible = false;
s1line2.visible = false;
s1line3.visible = false;
s1line4.visible = false;
correctMatch.text = ""

```

Declaring Variables

We need nine variables, one for count and the other eight for the select 1 thru 8. The count allows the player to pick two images they believe to match. When the player selects an image the select integer is set to 1.



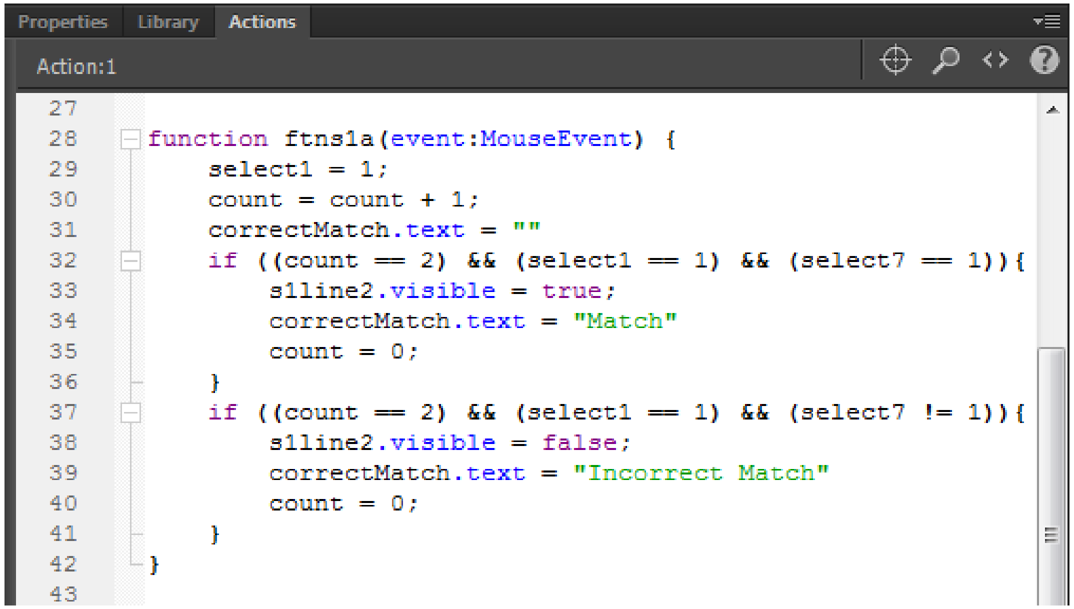
```

var select1:int = 0;
var select2:int = 0;
var select3:int = 0;
var select4:int = 0;
var select5:int = 0;
var select6:int = 0;
var select7:int = 0;
var select8:int = 0;
var count:int = 0;

```

Var allows us to declare a variable. Then comes the variable name. After the colon is the type of variable, which in this case is an integer. If we want to assign a value to the variable, we use the equal sign in the same statement.

The Select Function

We add the logic inside the function for the select button. When an image is selected we change the select variable to 1. We add one to the count variable. We blank out the correct match text.

Then we determine if the count equals 2, if the select1 equals 1 and if the matching image's selection variable equals 1. If this is true using the And statement (&&), we turn on the matching line, make the matching text state match and change the count to zero.



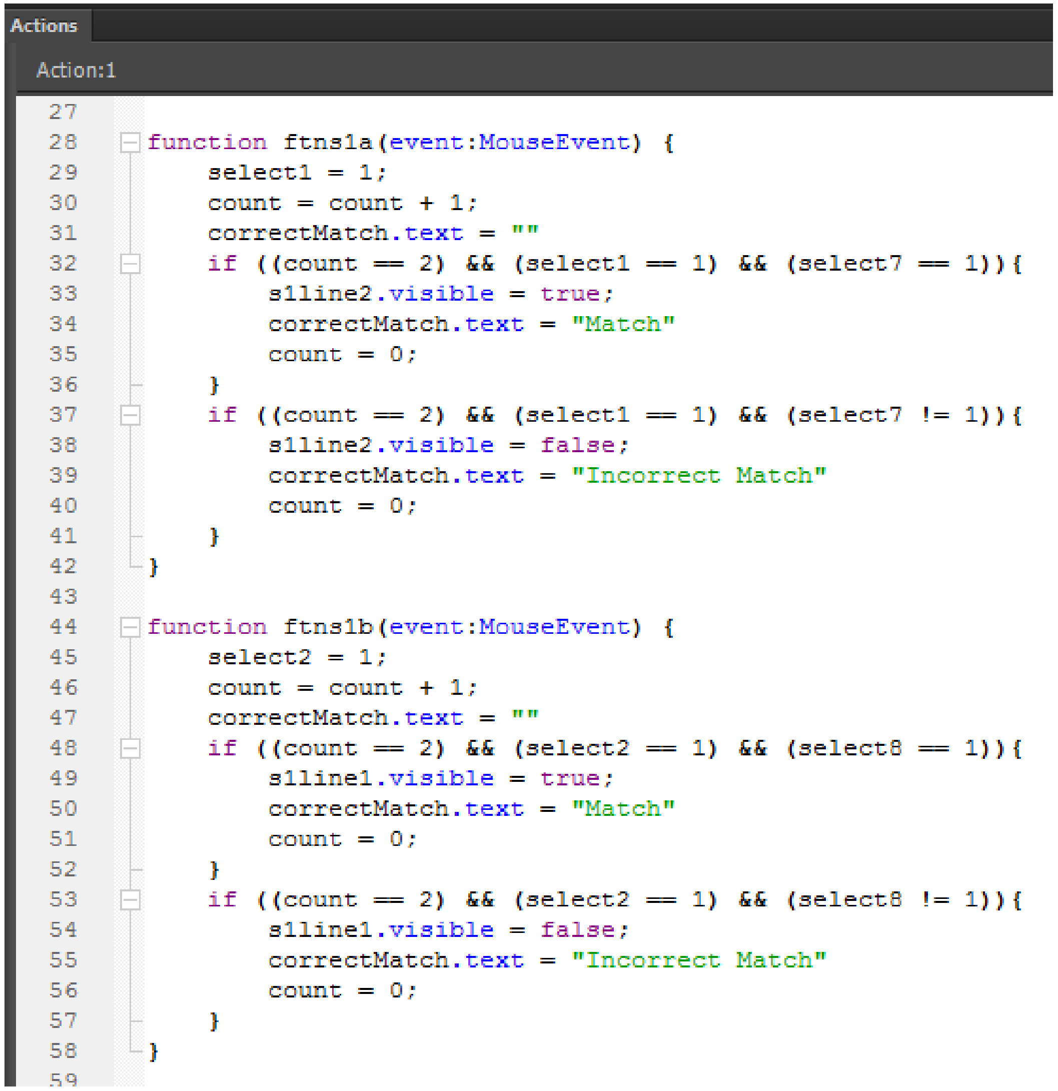
```
function ftns1a(event:MouseEvent) {
    select1 = 1;
    count = count + 1;
    correctMatch.text = ""
    if ((count == 2) && (select1 == 1) && (select7 == 1)){
        s1line2.visible = true;
        correctMatch.text = "Match"
        count = 0;
    }
    if ((count == 2) && (select1 == 1) && (select7 != 1)){
        s1line2.visible = false;
        correctMatch.text = "Incorrect Match"
        count = 0;
    }
}
```

Then we determine if the count equals 2, if the select1 equals 1 and if the matching image's selection variable does not equal 1. If this is true using the And statement (&&), we turn off the matching line, make the matching text state incorrect match and change the count to zero.

Copy Select1 Function

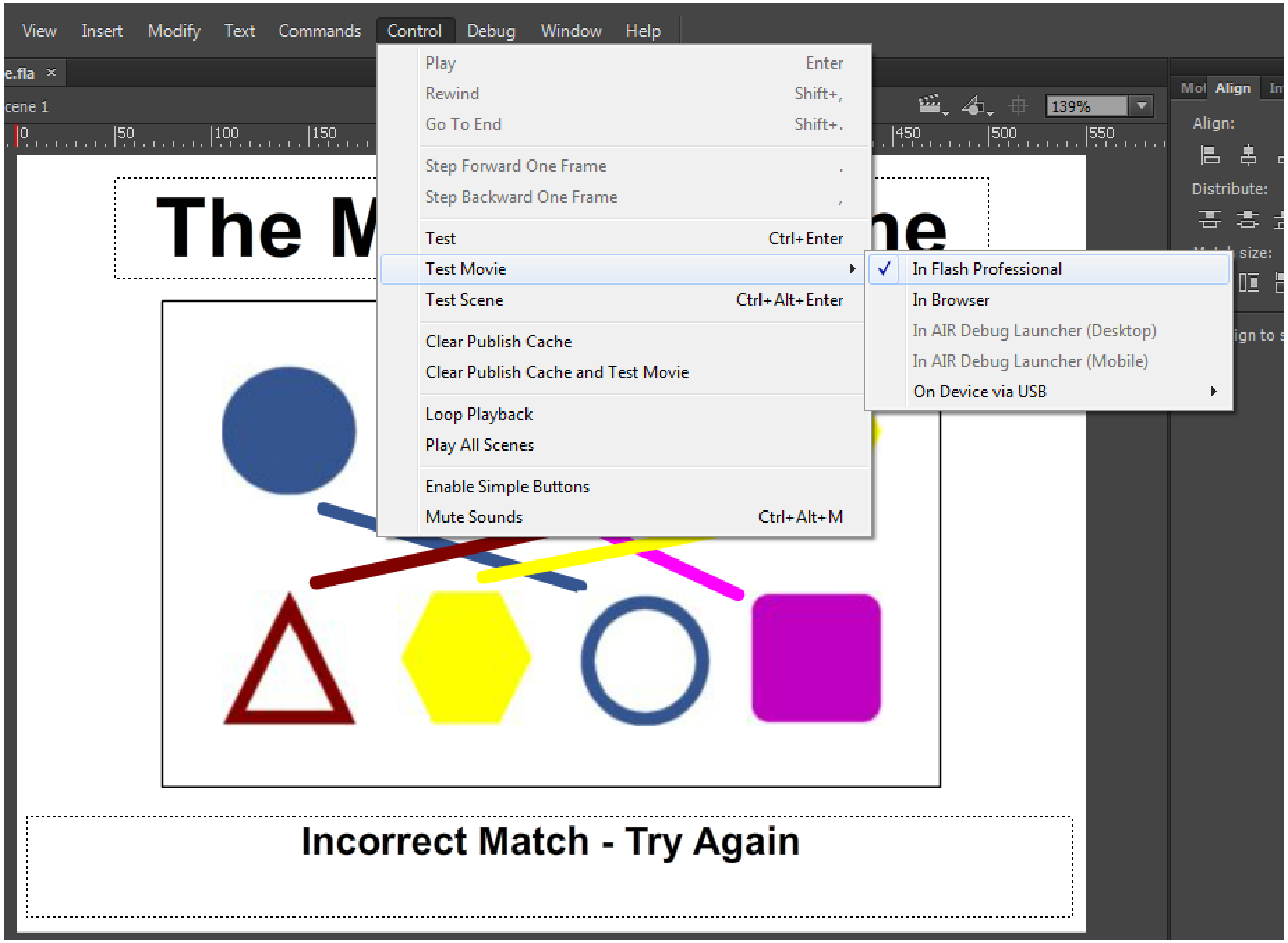
All we need to do now is copy the s1a function and change the a to b for the function name and variables.

Repeat this again until all 8 select functions are completed.



```
function ftns1a(event:MouseEvent) {
    select1 = 1;
    count = count + 1;
    correctMatch.text = ""
    if ((count == 2) && (select1 == 1) && (select7 == 1)){
        s1line2.visible = true;
        correctMatch.text = "Match"
        count = 0;
    }
    if ((count == 2) && (select1 == 1) && (select7 != 1)){
        s1line2.visible = false;
        correctMatch.text = "Incorrect Match"
        count = 0;
    }
}

function ftns1b(event:MouseEvent) {
    select2 = 1;
    count = count + 1;
    correctMatch.text = ""
    if ((count == 2) && (select2 == 1) && (select8 == 1)){
        s1line1.visible = true;
        correctMatch.text = "Match"
        count = 0;
    }
    if ((count == 2) && (select2 == 1) && (select8 != 1)){
        s1line1.visible = false;
        correctMatch.text = "Incorrect Match"
        count = 0;
    }
}
```

Test the Movie

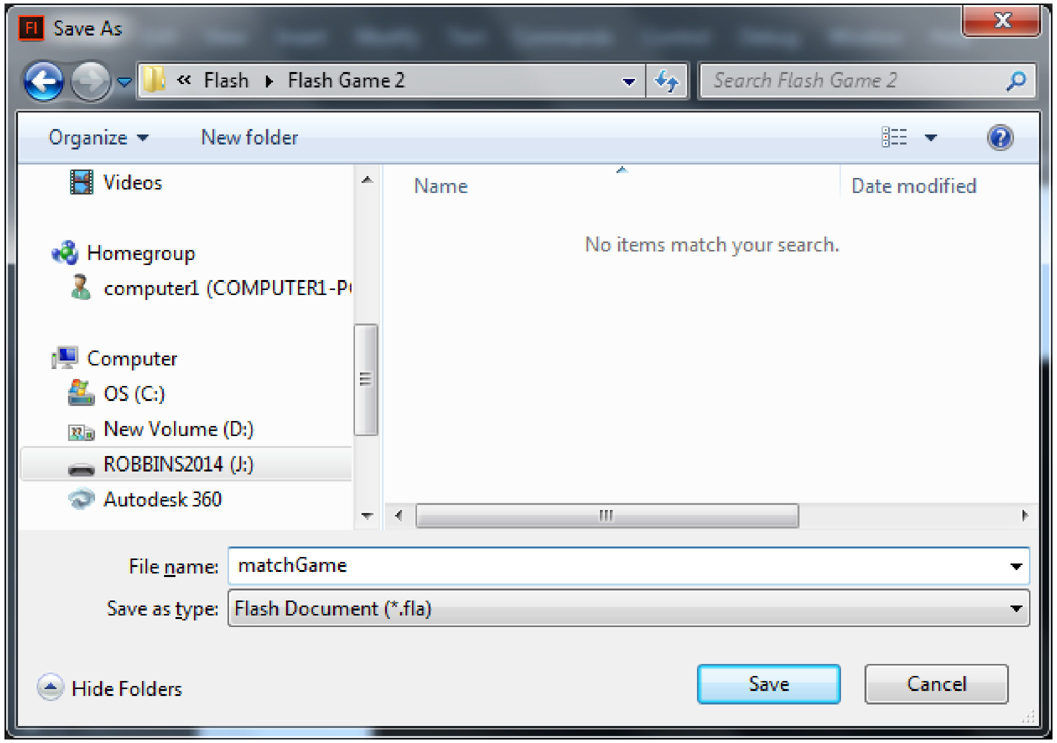


We should select Control on the menu and then Test Menu and In Flash Professional.

Saving the Movie

We need to save our work, so we choose File on the top menu bar and then we press Save on the drop down menu.

We will call our animation “matchGame” and we will depress the Save button.



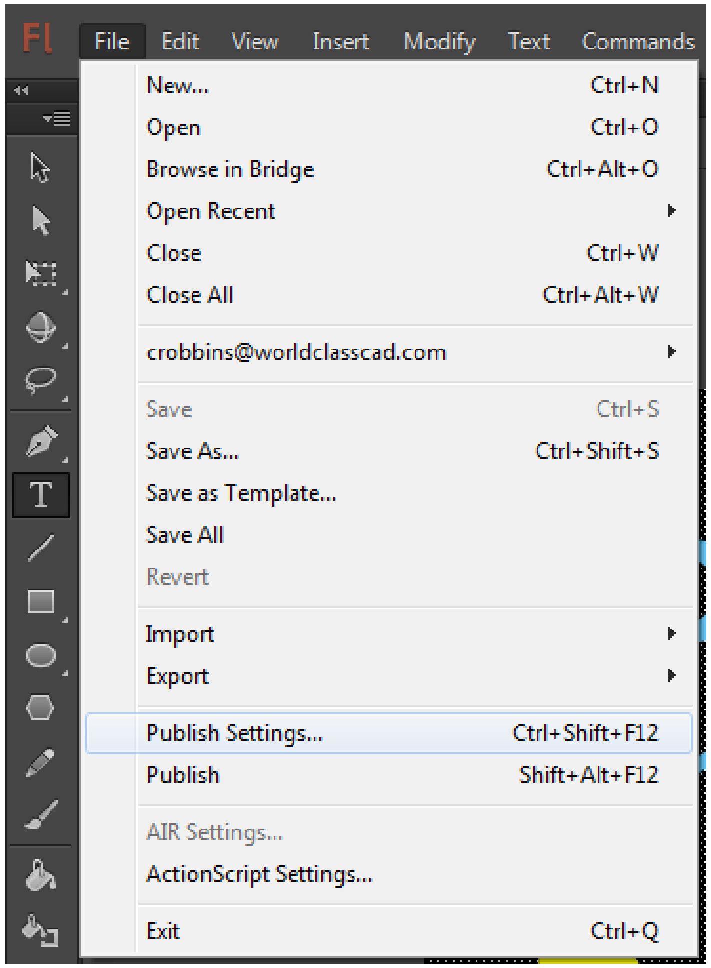
No items match your search.

File name: matchGame

Save as type: Flash Document (*.fla)

Publish the Flash Slide Show

We then choose File on the menu and Publish.


Fl
File
Edit
View
Insert
Modify
Text
Commands
New... Ctrl+N
Open Ctrl+O
Browse in Bridge Ctrl+Alt+O
Open Recent
Close Ctrl+W
Close All Ctrl+Alt+W
crobbins@worldclasscad.com
Save Ctrl+S
Save As... Ctrl+Shift+S
Save as Template...
Save All
Revert
Import
Export
Publish Settings... Ctrl+Shift+F12
Publish Shift+Alt+F12
AIR Settings...
ActionScript Settings...
Exit Ctrl+Q