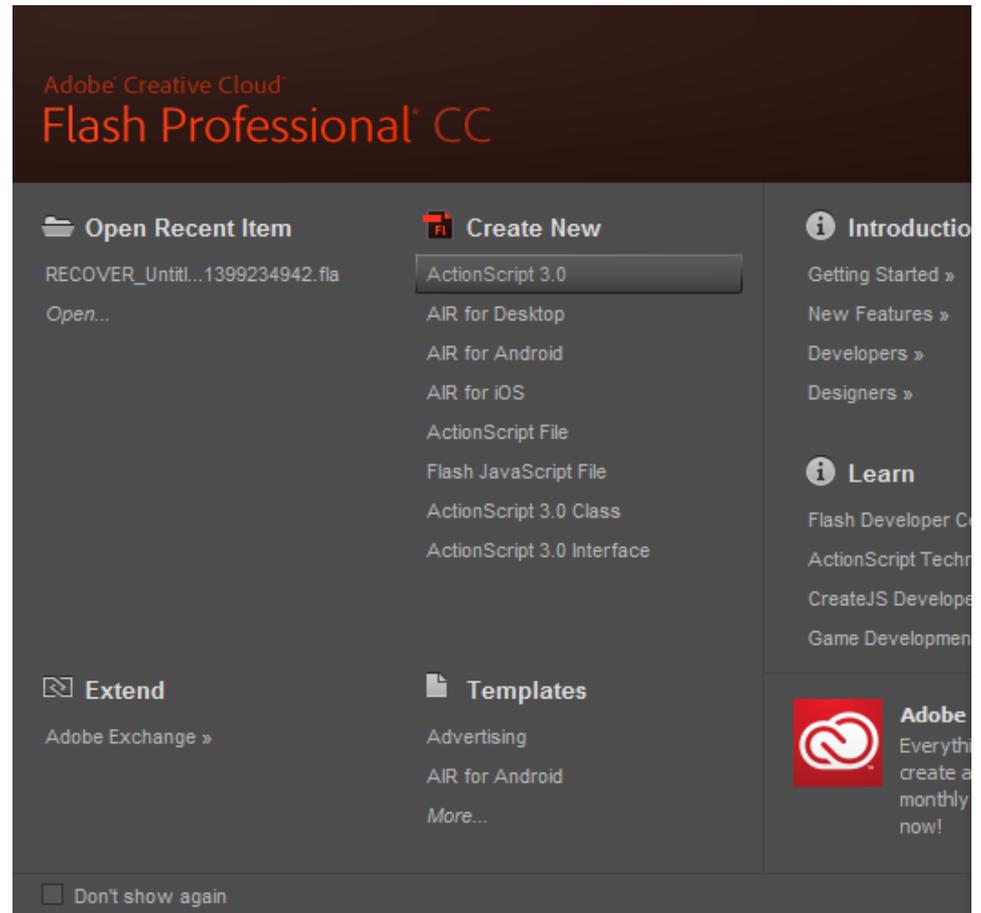


# Creating a Game 1

October 7, 2013

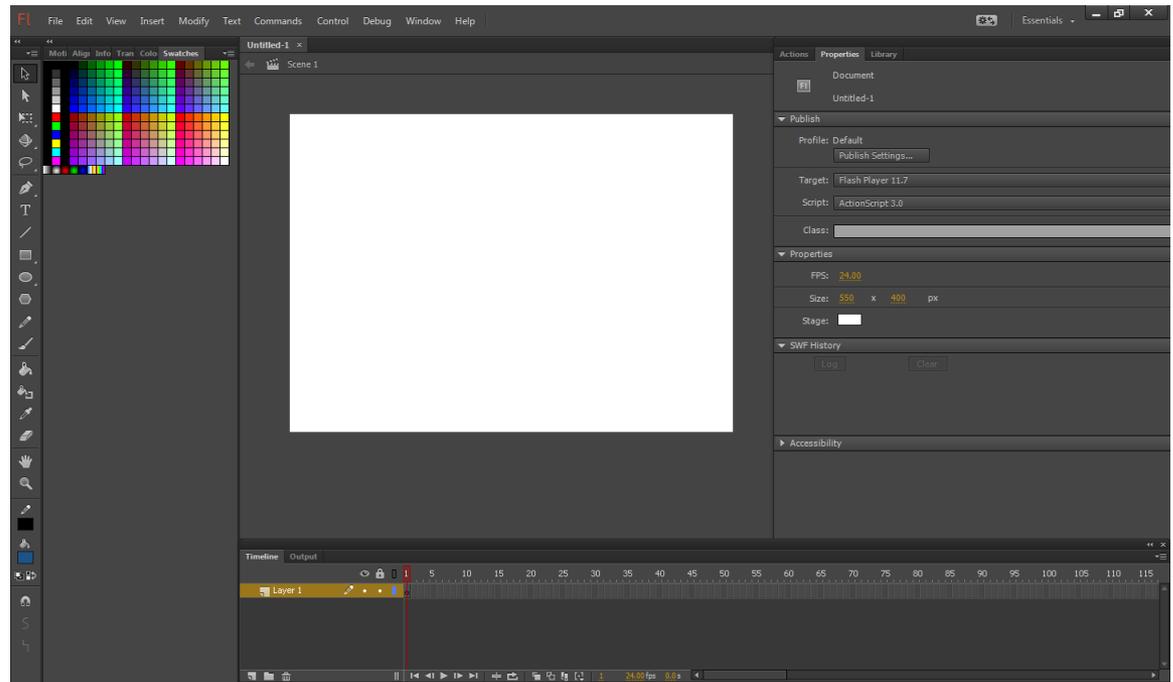
# Open the Flash Program

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



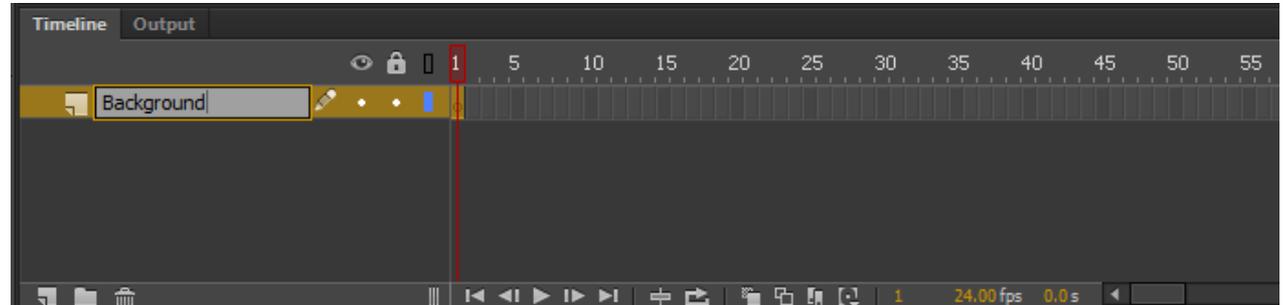
# The Flash Program

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.



# Renaming a Layer

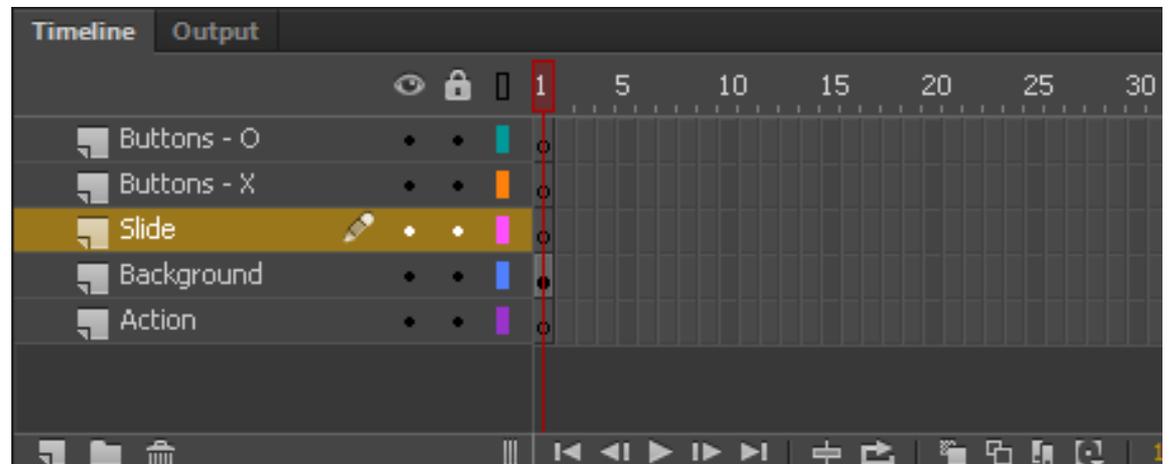
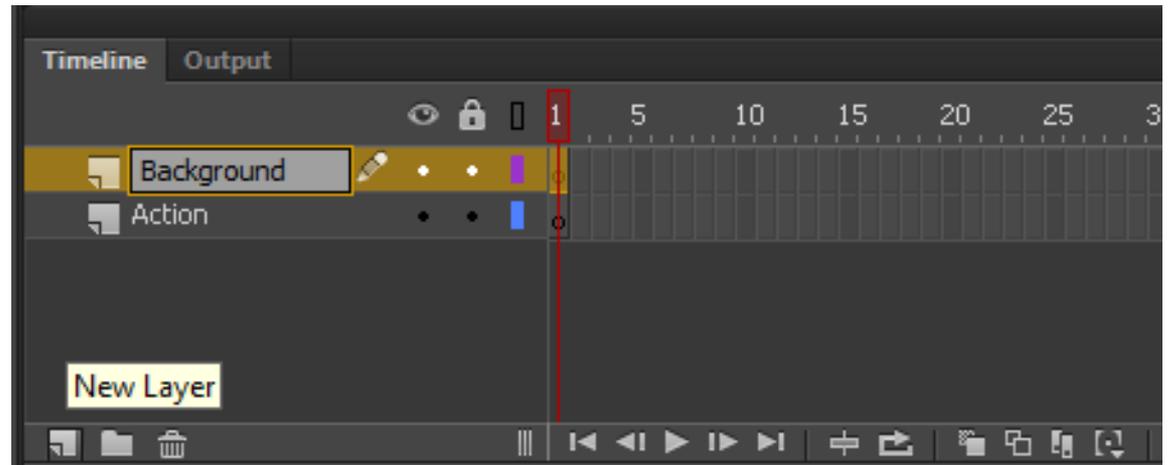
We will rename the layer called Layer1 to Background.



# 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-O and Buttons - X.

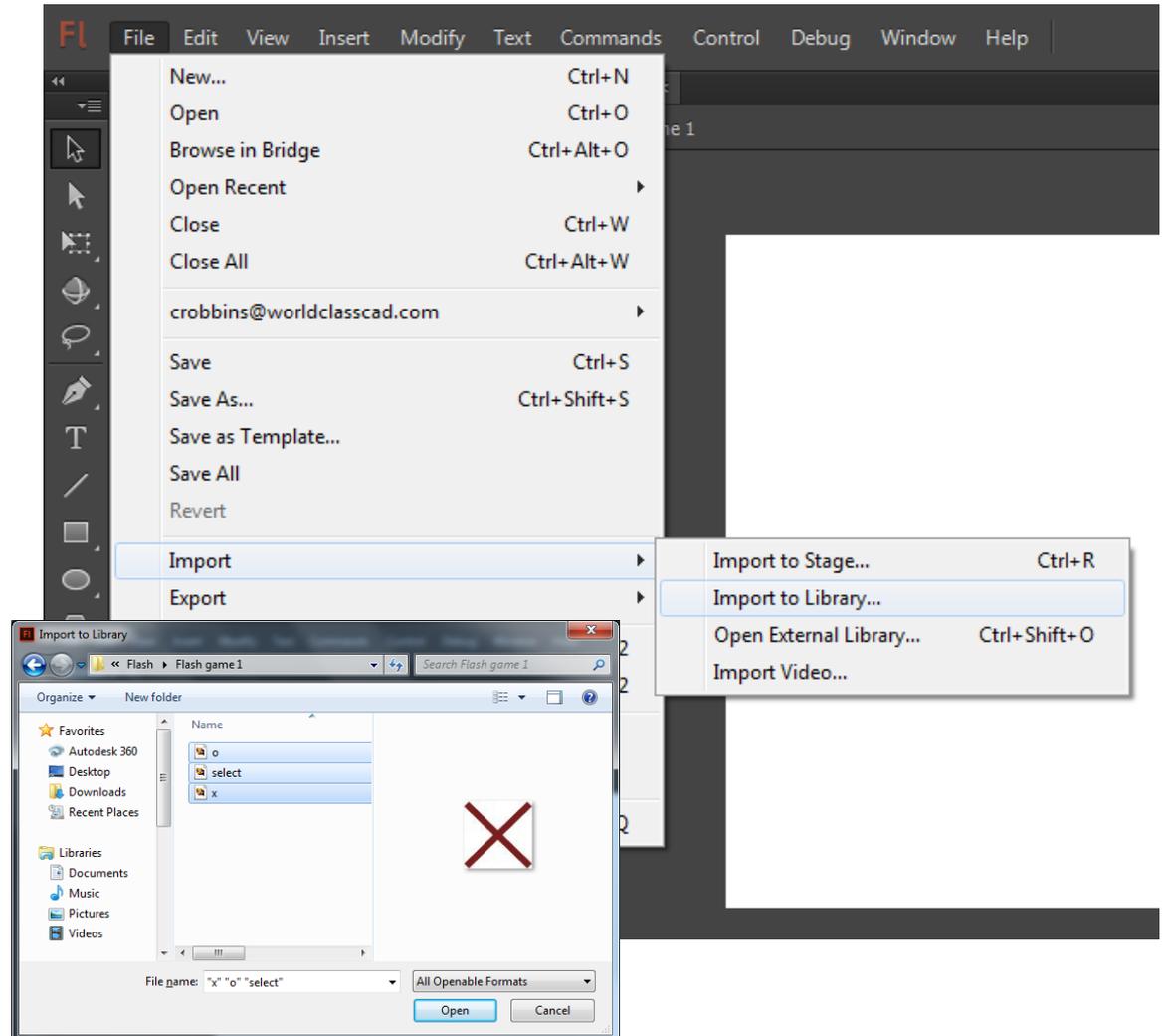


# Adding Images to the Library

We made three images that are on a 70 by 70 pixel white background. They are an X (x.jpg), an O (o.jpg) and the word “select” (select.jpg).

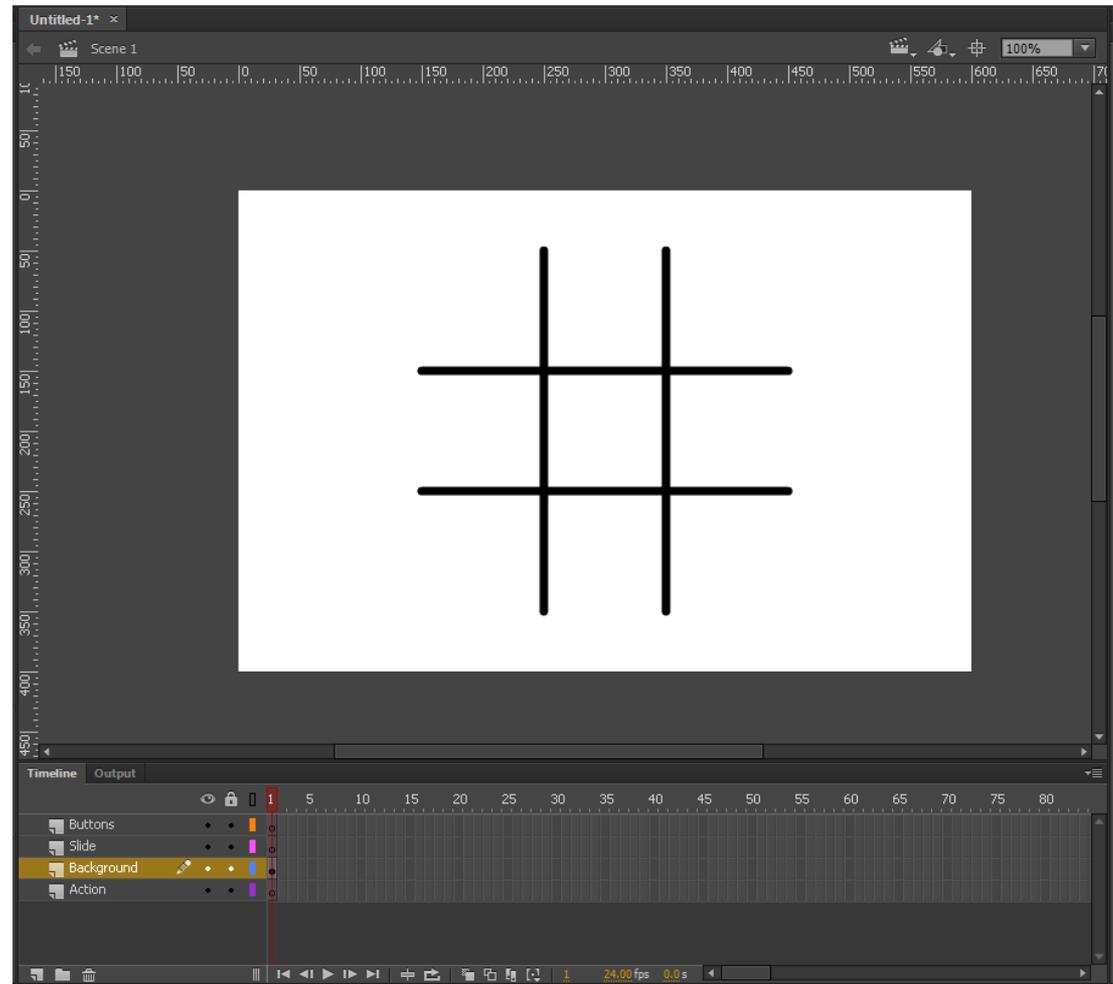
To add the images to the Library, we will import them. We should choose File on the top menu, then Import and Import to Library.

We select the three images from the computer’s folder and press Open



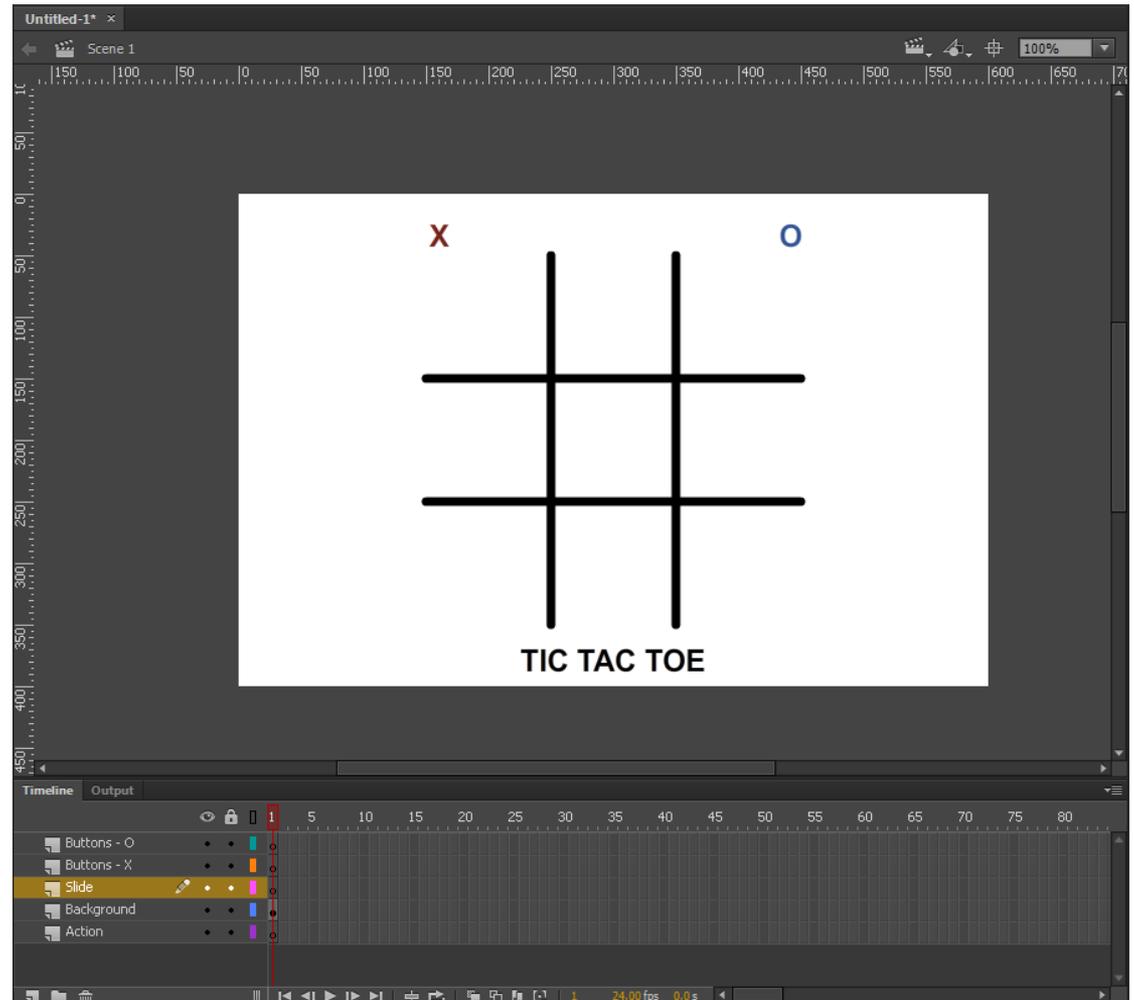
# Adding Lines to the Stage

At the beginning of the game, on the background layer, we will add four 7 pixels wide by 300 pixels long lines. On the Line Properties tab, we will make the Stroke color box black and the Stroke Height as 7. We will then draw the lines 100 pixels apart.



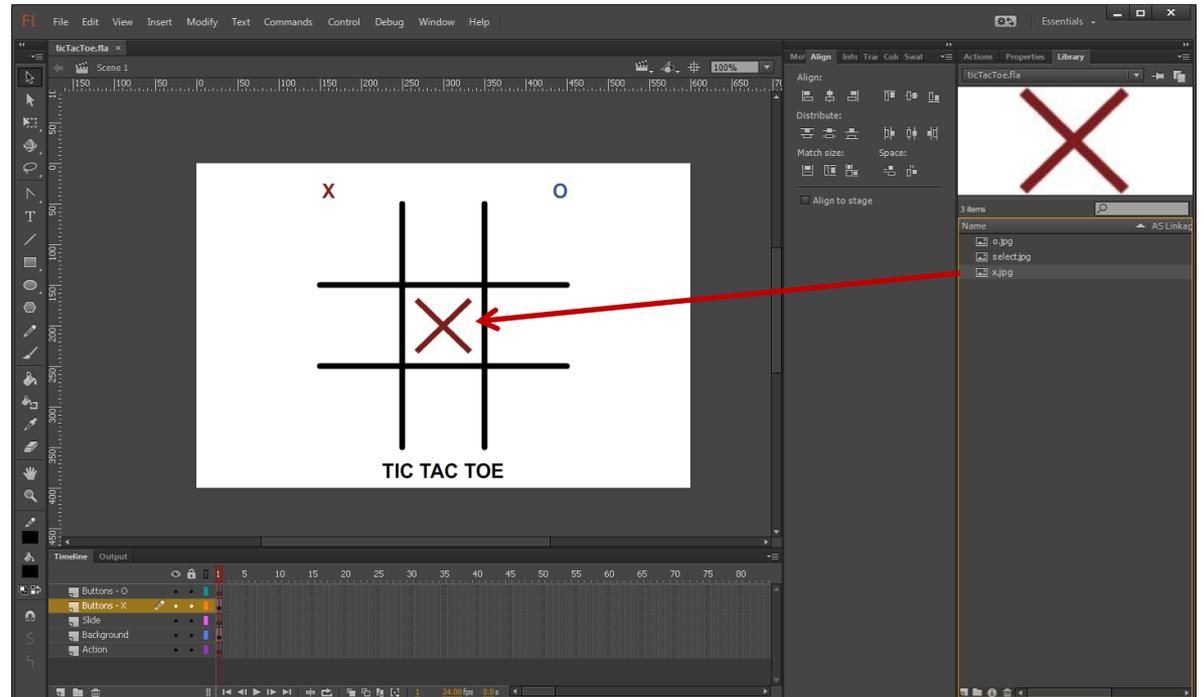
# Adding Text to the Stage

We will then add text for the game name on the background layer. This will be TIC TAC TOE and X and O.



# Adding an Image to Button – X Frame

Highlight the Buttons - X 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.

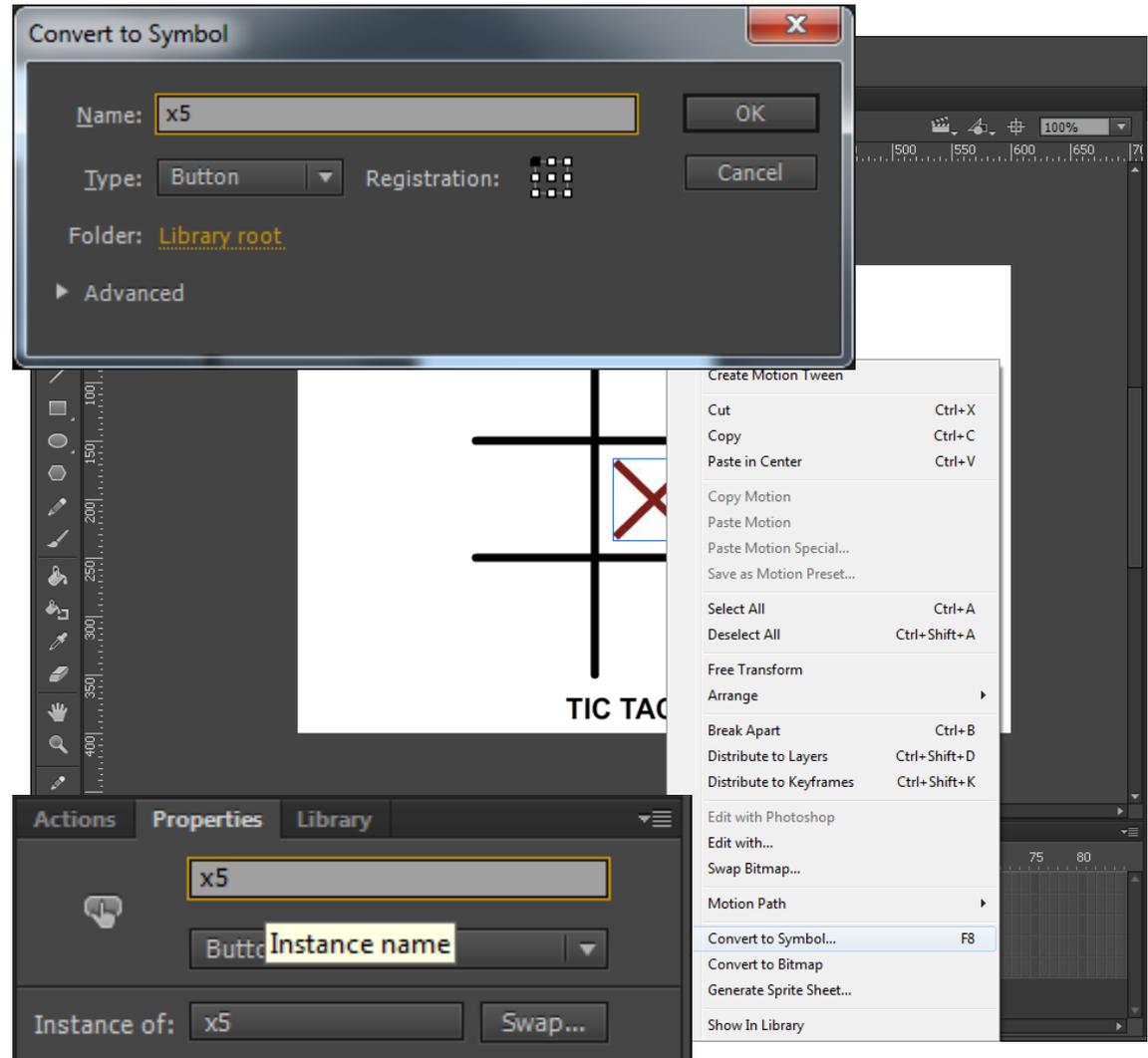


# Convert the Image to a Button

We then right click on the red x and we choose the Convert to Symbol from the menu.

The Convert to Symbol window will appear. We will name the symbol x5 and make the type Button.

On the Properties tab, we will name the Instance x5.

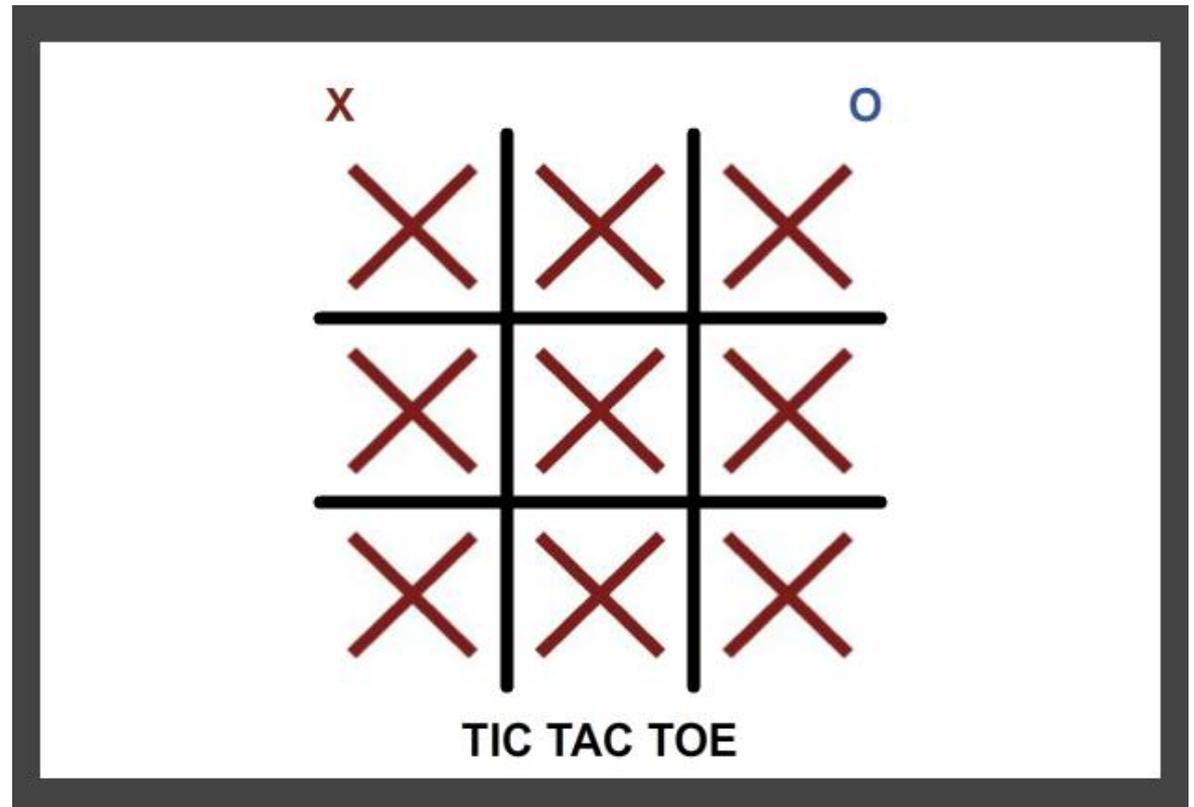


# Convert all the X Images to Buttons

We then add the other 8 x's. The numbers go x1 to x3 on the top row, x4 to x6 on the middle row and x7 to x9 on the bottom row.

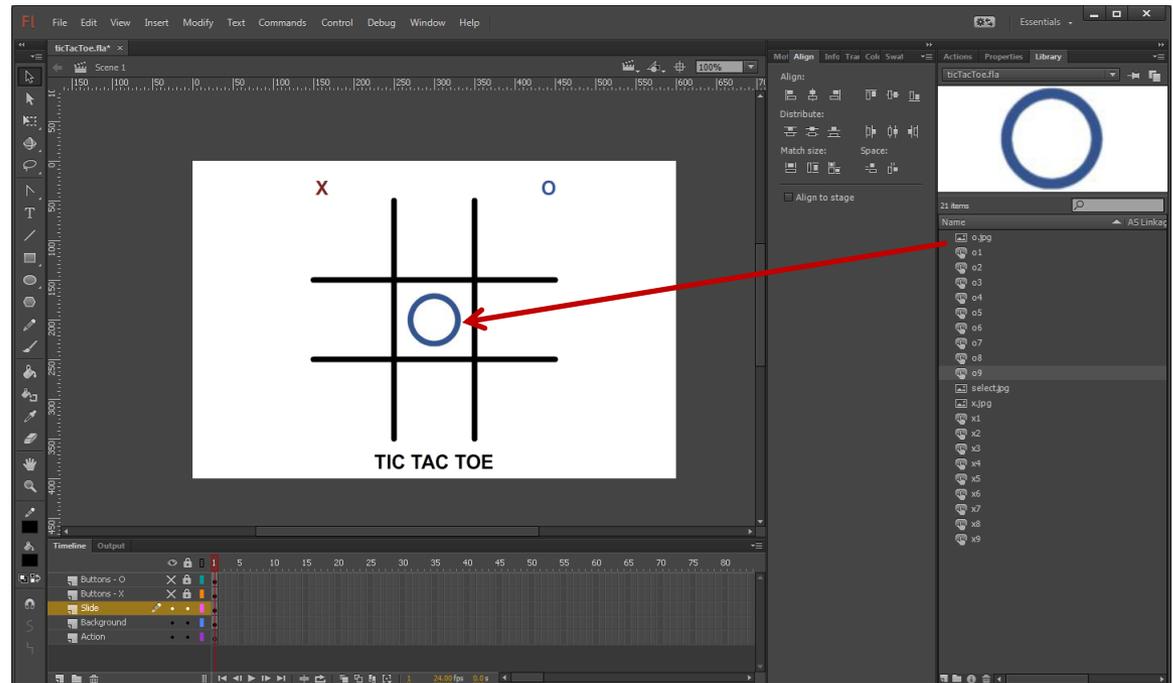
The Convert to Symbol window will appear. We will name the symbol the appropriate x and space number and make the type Button.

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



# Adding an Image to Button – O Frame

Highlight the Buttons - O 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.

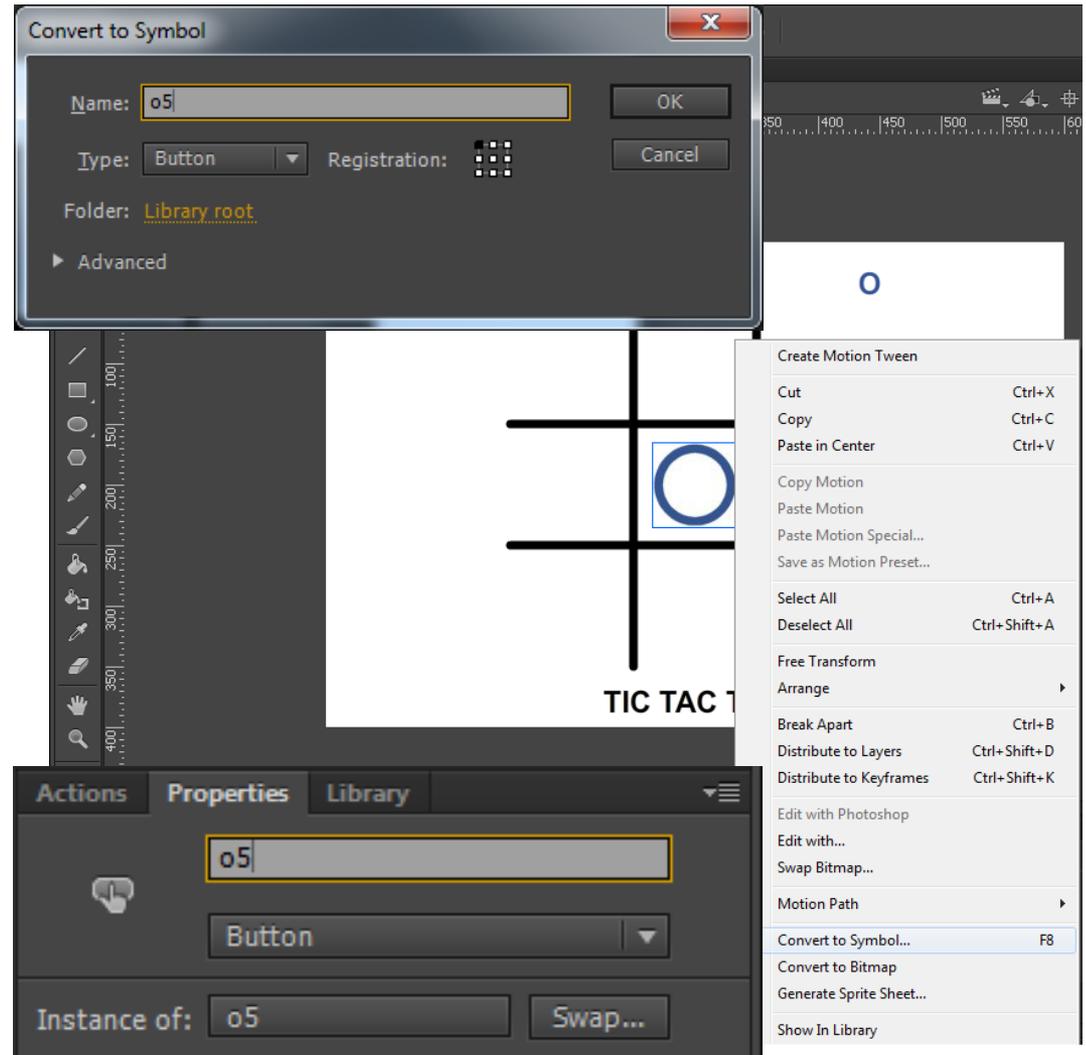


# Convert the O Image to a Button

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

The Convert to Symbol window will appear. We will name the symbol o5 and make the type Button.

On the Properties tab, we will name the Instance o5.

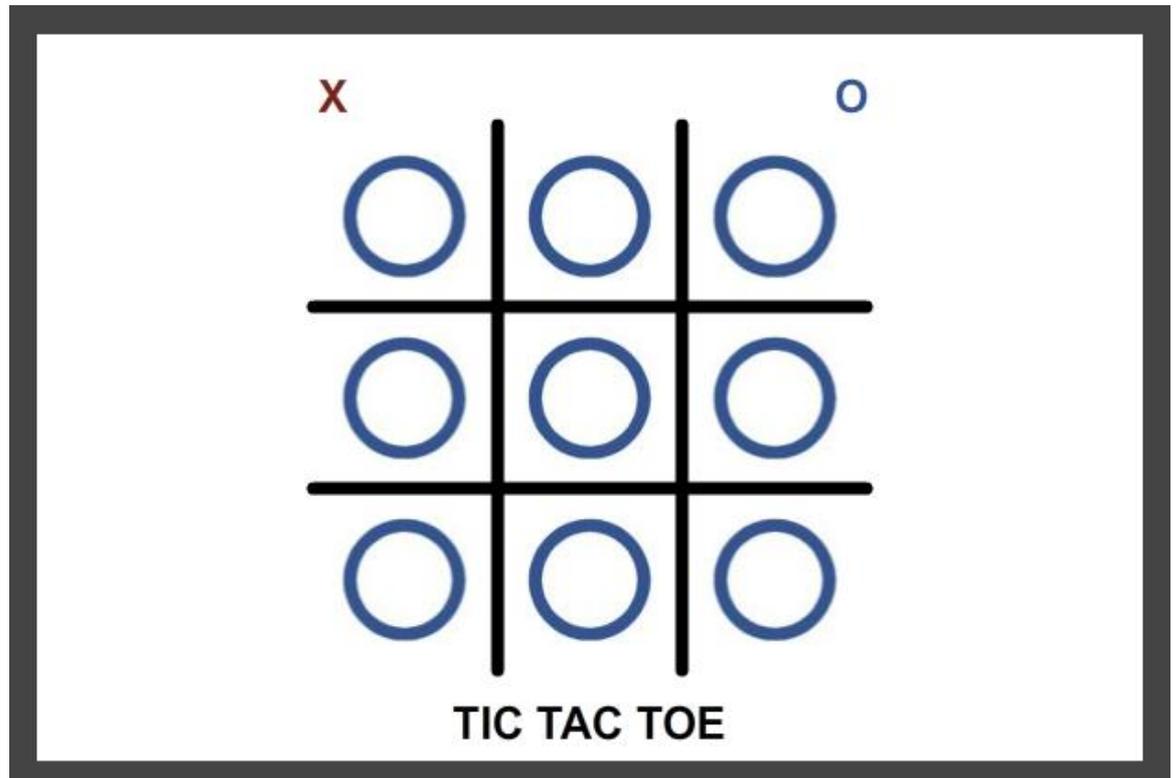


# Convert all the O Images to Buttons

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.

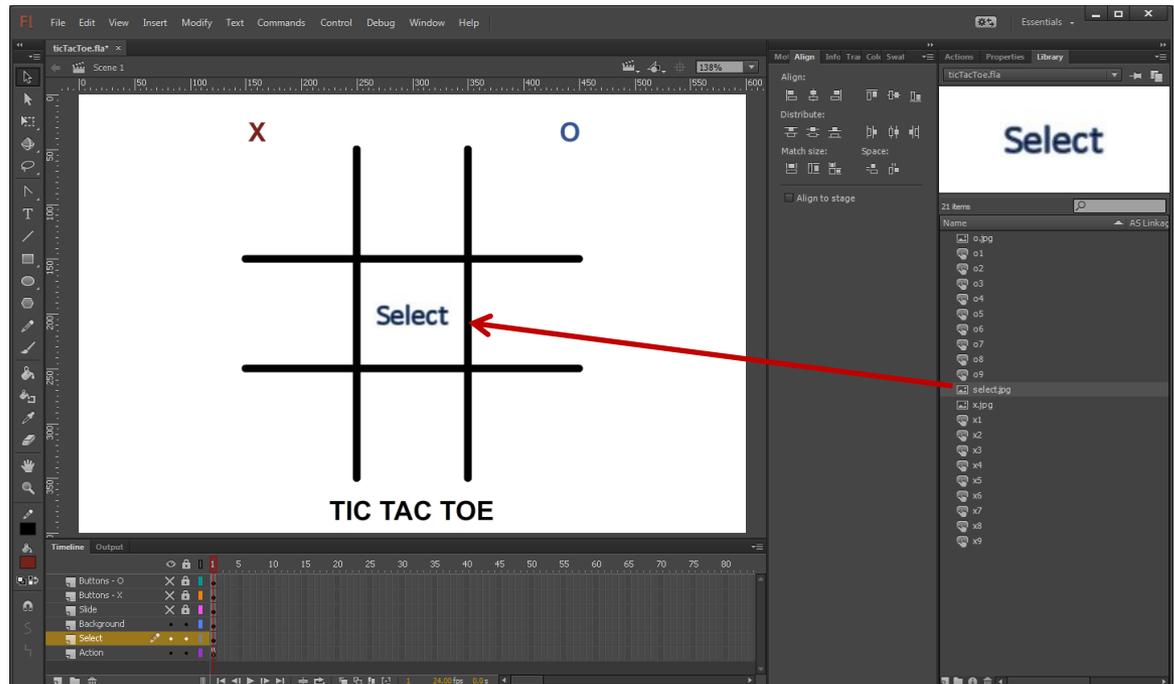
The Convert to Symbol window will appear. We will name the symbol the appropriate o and space number and make the type Button.

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



# Adding an Image to Button – Select

Highlight the Select 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.

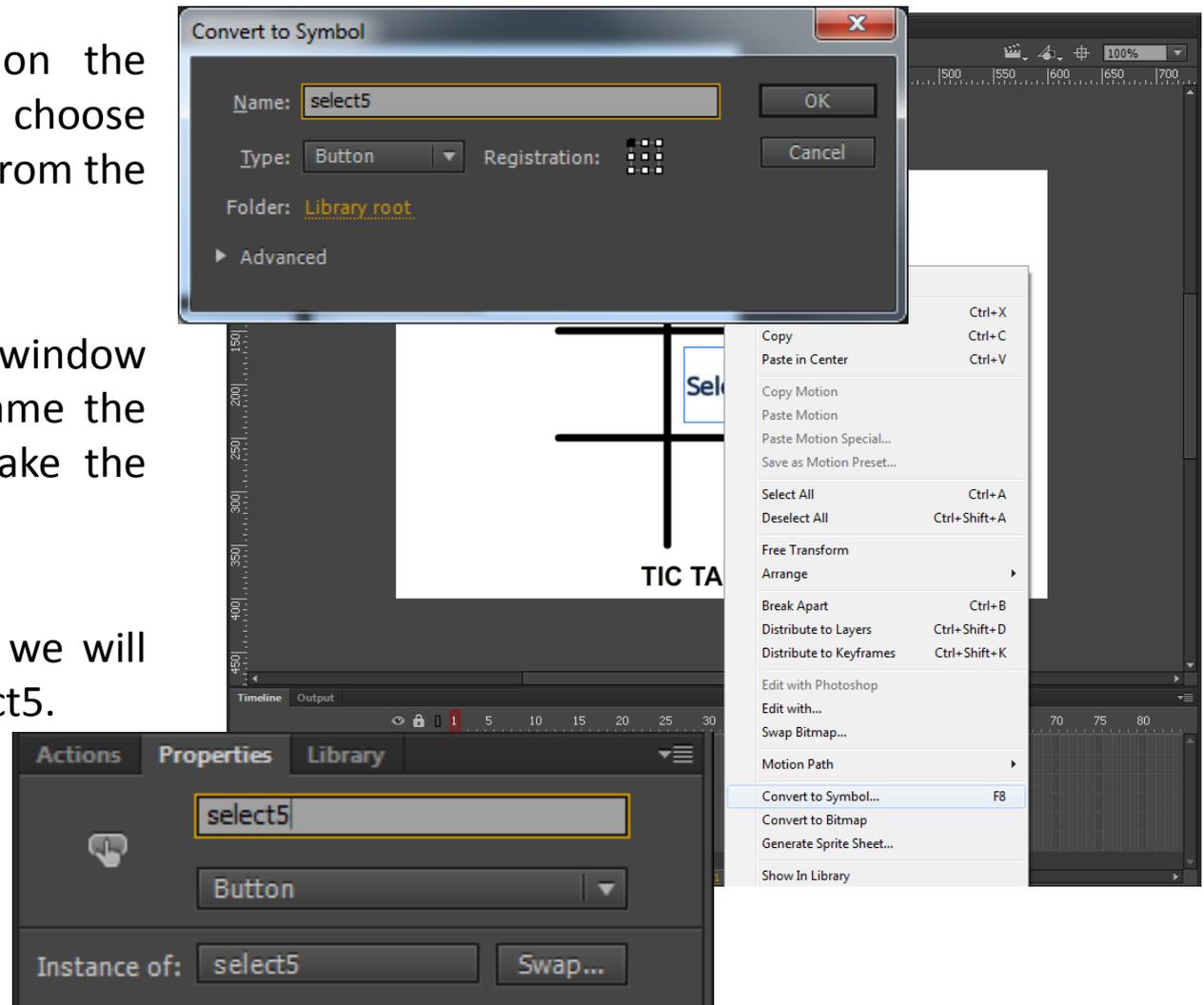


# Convert the Select Image to a Button

We then right click on the Select image and we choose the Convert to Symbol from the menu.

The Convert to Symbol window will appear. We will name the symbol select5 and make the type Button.

On the Properties tab, we will name the Instance select5.

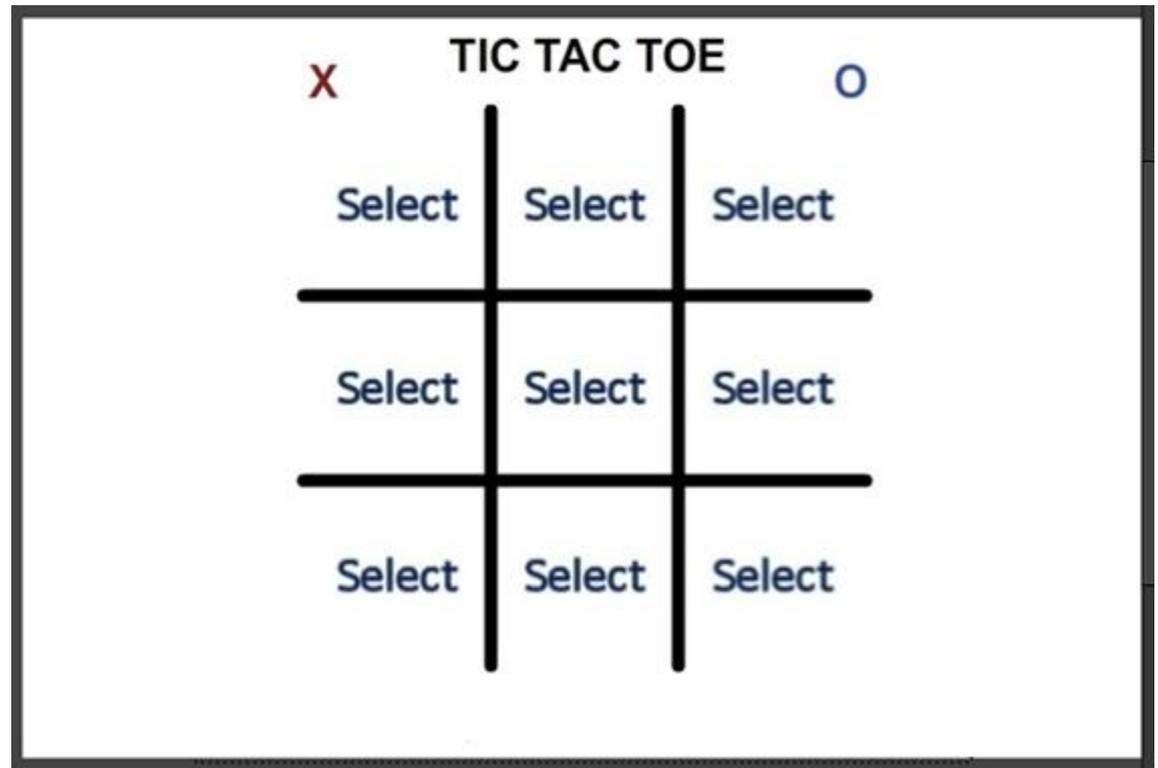


# Convert all the Select Images to Buttons

We then add the other 8 select buttons. The numbers go select1 to select3 on the top row, select4 to select6 on the middle row and select7 to select9 on the bottom row.

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

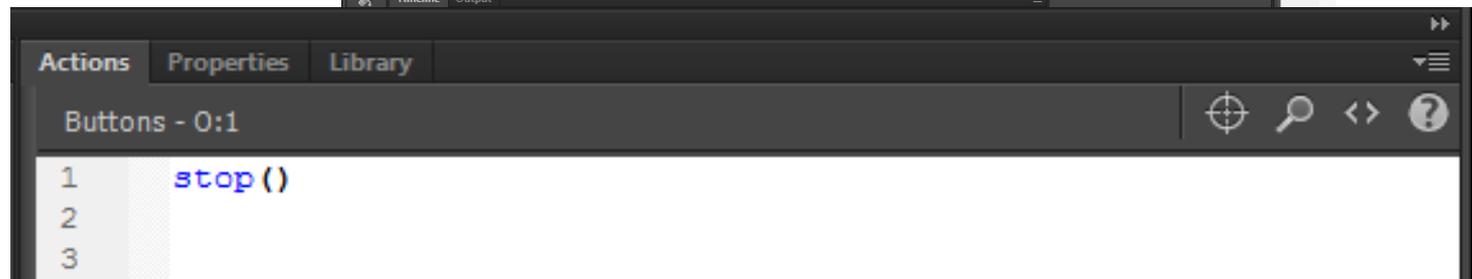
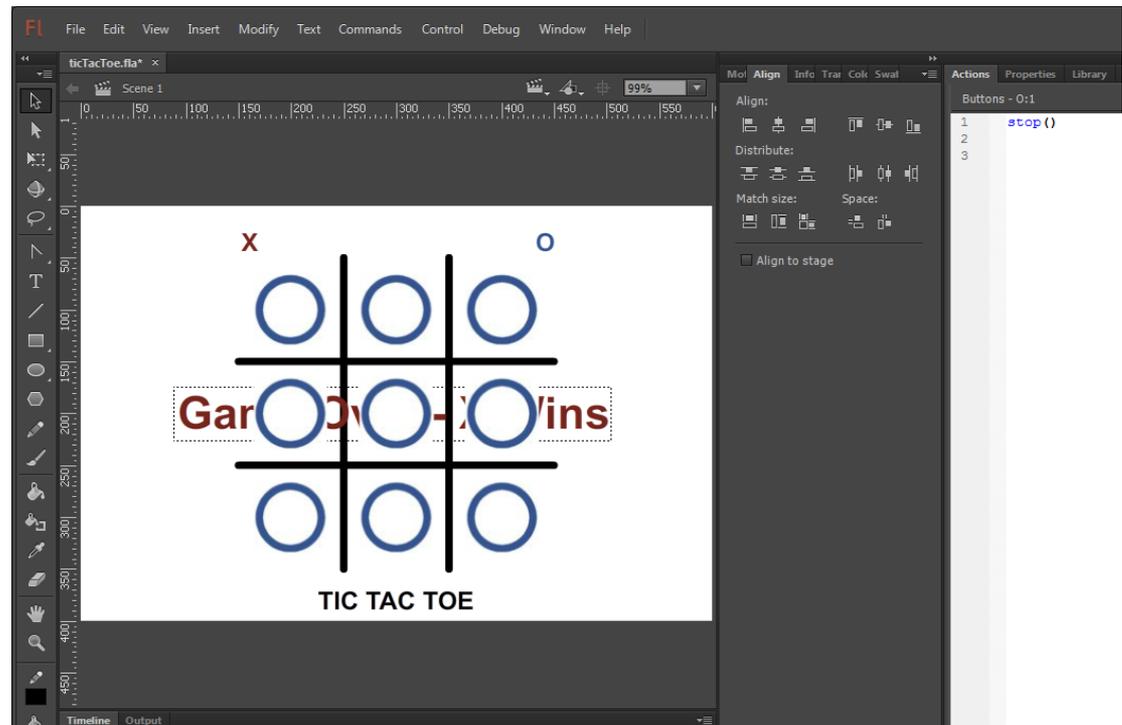
On the Properties tab, we will name the Instance the same as the symbol name.



# 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 fframe.



# Add another Event Listener



The screenshot shows an IDE's Actions panel. At the top, it says "Action:1". Below that is a search bar containing the number "2". To the right of the search bar are icons for search, navigation, and help, along with a "Find & Replace" dropdown menu. Below the search bar is a text input field containing the number "9". To the right of this field are "Replace" and "Replace All" buttons. The main area of the panel is a code editor showing a list of actions:

```
1  stop();
2
3  select1.addEventListener(MouseEvent.CLICK,ftnselect1);
4  select2.addEventListener(MouseEvent.CLICK,ftnselect2);
5  select3.addEventListener(MouseEvent.CLICK,ftnselect3);
6  select4.addEventListener(MouseEvent.CLICK,ftnselect4);
7  select5.addEventListener(MouseEvent.CLICK,ftnselect5);
8  select6.addEventListener(MouseEvent.CLICK,ftnselect6);
9  select7.addEventListener(MouseEvent.CLICK,ftnselect7);
10 select8.addEventListener(MouseEvent.CLICK,ftnselect8);
11 select9.addEventListener(MouseEvent.CLICK,ftnselect9);
12
```

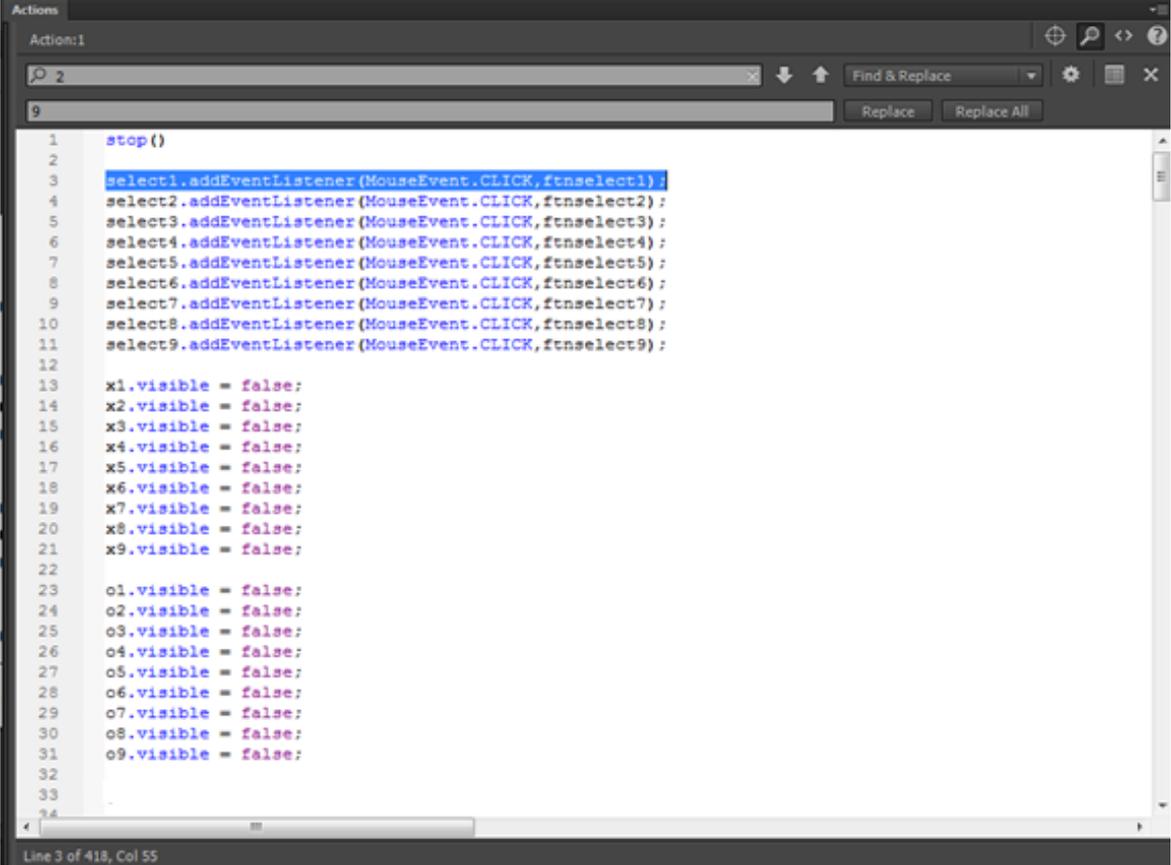
We then add an event listener to determine when a select button was clicked. We type:

```
select1.addEventListener(MouseEvent.CLICK,ftnselect1);
```

# Make the Xs and Os Invisible

We then type

`x1.visible = false;` to make the red X invisible. We do this for each X and O symbol.



```
1 stop()
2
3 select1.addEventListener(MouseEvent.CLICK, ftselect1);
4 select2.addEventListener(MouseEvent.CLICK, ftselect2);
5 select3.addEventListener(MouseEvent.CLICK, ftselect3);
6 select4.addEventListener(MouseEvent.CLICK, ftselect4);
7 select5.addEventListener(MouseEvent.CLICK, ftselect5);
8 select6.addEventListener(MouseEvent.CLICK, ftselect6);
9 select7.addEventListener(MouseEvent.CLICK, ftselect7);
10 select8.addEventListener(MouseEvent.CLICK, ftselect8);
11 select9.addEventListener(MouseEvent.CLICK, ftselect9);
12
13 x1.visible = false;
14 x2.visible = false;
15 x3.visible = false;
16 x4.visible = false;
17 x5.visible = false;
18 x6.visible = false;
19 x7.visible = false;
20 x8.visible = false;
21 x9.visible = false;
22
23 o1.visible = false;
24 o2.visible = false;
25 o3.visible = false;
26 o4.visible = false;
27 o5.visible = false;
28 o6.visible = false;
29 o7.visible = false;
30 o8.visible = false;
31 o9.visible = false;
32
33
34
```

Line 3 of 418, Col 55

# Declaring Variables

We need four variables, one for the count, one for the game on and off, the next for the winner and the last for the selection count. The count allows the X player to play on the odd turn and the O player on the even turn. We decided to toggle between the two words.

The game starts as the word “on” and when 3 X’s or O’s are achieved the game variable is set to “off”.

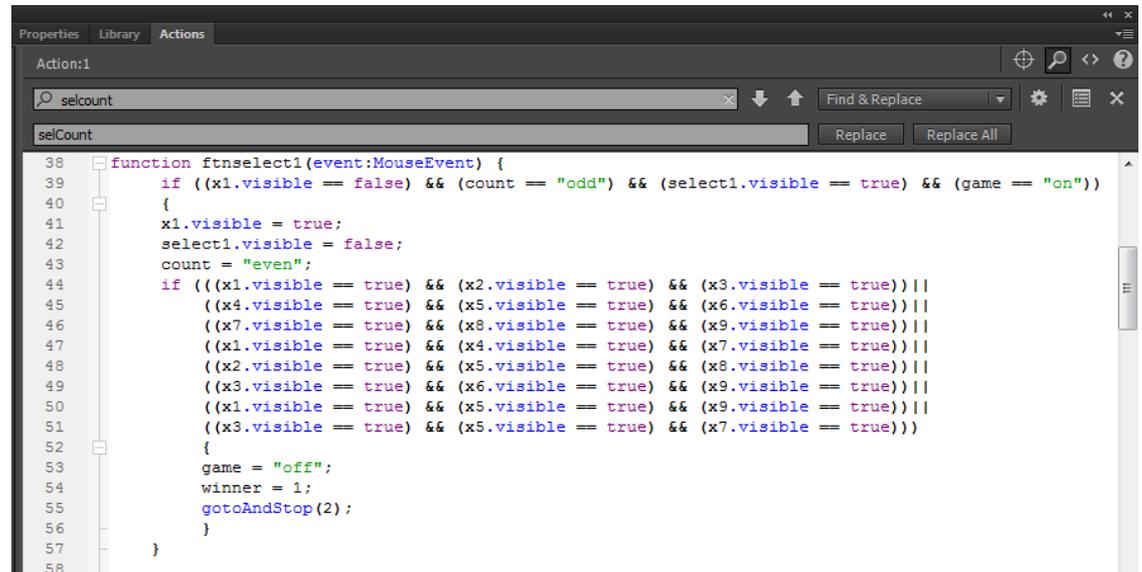
```
32
33     var count:String = "odd";
34     var game:String = "on";
35     var winner:int = 0;
36     var selCount:int = 0;
37
```

The winner variable captures if no one wins (0), if X wins (1), or if Y wins (2). The last variable counts each selection and at nine end the game.

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 a string. If we want to assign a value to the variable, we use the equal sign in the same statement.

# The Select Function - X

We add the logic inside the function for the select button. We check the X first. We determine if the X symbol is on, if the count is odd, if the select symbol is visible and if the game is on. If this is true using the And statement (&&), we turn on the X symbol, turn off the select symbol, change the count to even, and we determine if the X player has won the game.

A screenshot of an IDE window titled 'Action:1'. The search bar contains 'selcount'. The code editor shows the following code:

```
38 function ftselect1(event:MouseEvent) {
39     if ((x1.visible == false) && (count == "odd") && (select1.visible == true) && (game == "on"))
40     {
41         x1.visible = true;
42         select1.visible = false;
43         count = "even";
44         if (((x1.visible == true) && (x2.visible == true) && (x3.visible == true)) ||
45             ((x4.visible == true) && (x5.visible == true) && (x6.visible == true)) ||
46             ((x7.visible == true) && (x8.visible == true) && (x9.visible == true)) ||
47             ((x1.visible == true) && (x4.visible == true) && (x7.visible == true)) ||
48             ((x2.visible == true) && (x5.visible == true) && (x8.visible == true)) ||
49             ((x3.visible == true) && (x6.visible == true) && (x9.visible == true)) ||
50             ((x1.visible == true) && (x5.visible == true) && (x9.visible == true)) ||
51             ((x3.visible == true) && (x5.visible == true) && (x7.visible == true)))
52         {
53             game = "off";
54             winner = 1;
55             gotoAndStop(2);
56         }
57     }
58 }
```

For the win, we check the visibility of X's in 3 rows, 3 columns and 2 diagonals using the Or statement (||). If the condition is met, turn the game off, turn on the game over text and announce that the X player has won.

# The Select Function - O

We then add the logic inside the function for the select button for the O's. We determine if the O symbol is on, if the count is odd, if the select symbol is visible and if the game is on. If this is true using the And statement (&&), we turn on the O symbol, turn off the select symbol, change the count to even, and we determine if the O player has won the game.

```
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82

if ((o1.visible == false) && (count == "even") && (select1.visible == true) && (game == "on"))
{
o1.visible = true;
select1.visible = false;
count = "odd";
if (((o1.visible == true) && (o2.visible == true) && (o3.visible == true))||
((o4.visible == true) && (o5.visible == true) && (o6.visible == true))||
((o7.visible == true) && (o8.visible == true) && (o9.visible == true))||
((o1.visible == true) && (o4.visible == true) && (o7.visible == true))||
((o2.visible == true) && (o5.visible == true) && (o8.visible == true))||
((o3.visible == true) && (o6.visible == true) && (o9.visible == true))||
((o1.visible == true) && (o5.visible == true) && (o9.visible == true))||
((o3.visible == true) && (o5.visible == true) && (o7.visible == true)))
{
game = "off";
winner = 2;
gotoAndStop(2);
}
}
selCount = selCount + 1;
if (selCount == 9){
gotoAndStop(2);
}
}
```

For the win, we check the visibility of O's in 3 rows, 3 columns and 2 diagonals using the Or statement (||). If the condition is met, turn the game off, turn on the game over text and announce that the O player has won.

Lastly, we add one to the selection count and then we check the if the selection count is as nine and the game is over.

# Copy Select1 Function

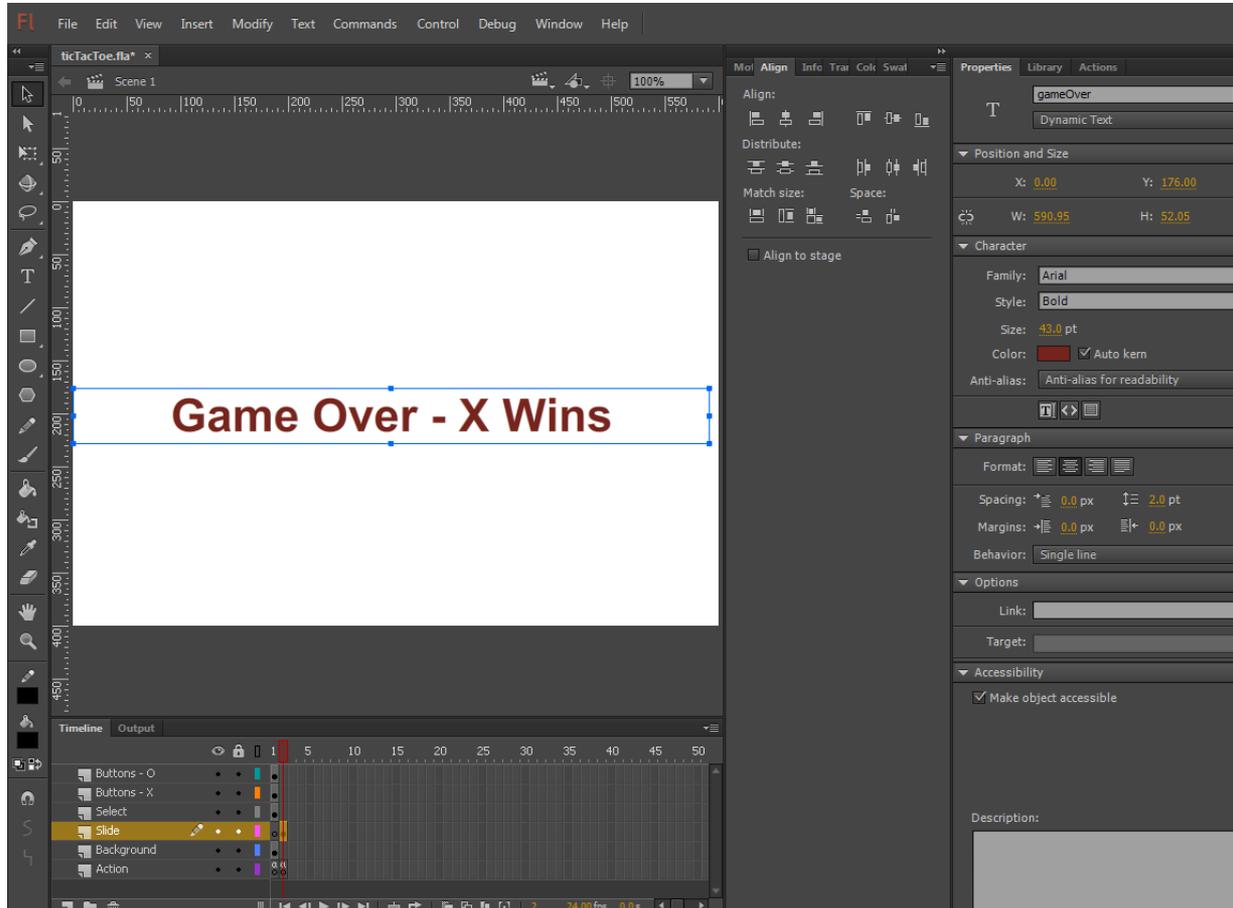
All we need to do now is copy the select1 function and change the 1 to 2 for the function name and variables. We should leave the 16 game ending if statements alone.

Repeat this again until all 9 select functions are completed.

```
function ftselect2(event:MouseEvent) {
  if ((x2.visible == false) && (count == "odd") && (select2.visible == true) && (game == "on"))
  {
    x2.visible = true;
    select2.visible = false;
    count = "even";
    if (((x1.visible == true) && (x2.visible == true) && (x3.visible == true)) ||
        ((x4.visible == true) && (x5.visible == true) && (x6.visible == true)) ||
        ((x7.visible == true) && (x8.visible == true) && (x9.visible == true)) ||
        ((x1.visible == true) && (x4.visible == true) && (x7.visible == true)) ||
        ((x2.visible == true) && (x5.visible == true) && (x8.visible == true)) ||
        ((x3.visible == true) && (x6.visible == true) && (x9.visible == true)) ||
        ((x1.visible == true) && (x5.visible == true) && (x9.visible == true)) ||
        ((x3.visible == true) && (x5.visible == true) && (x7.visible == true)))
    {
      game = "off";
      gameOver.visible = true;
      gameOver.text = "Game Over - X Wins";
    }
  }

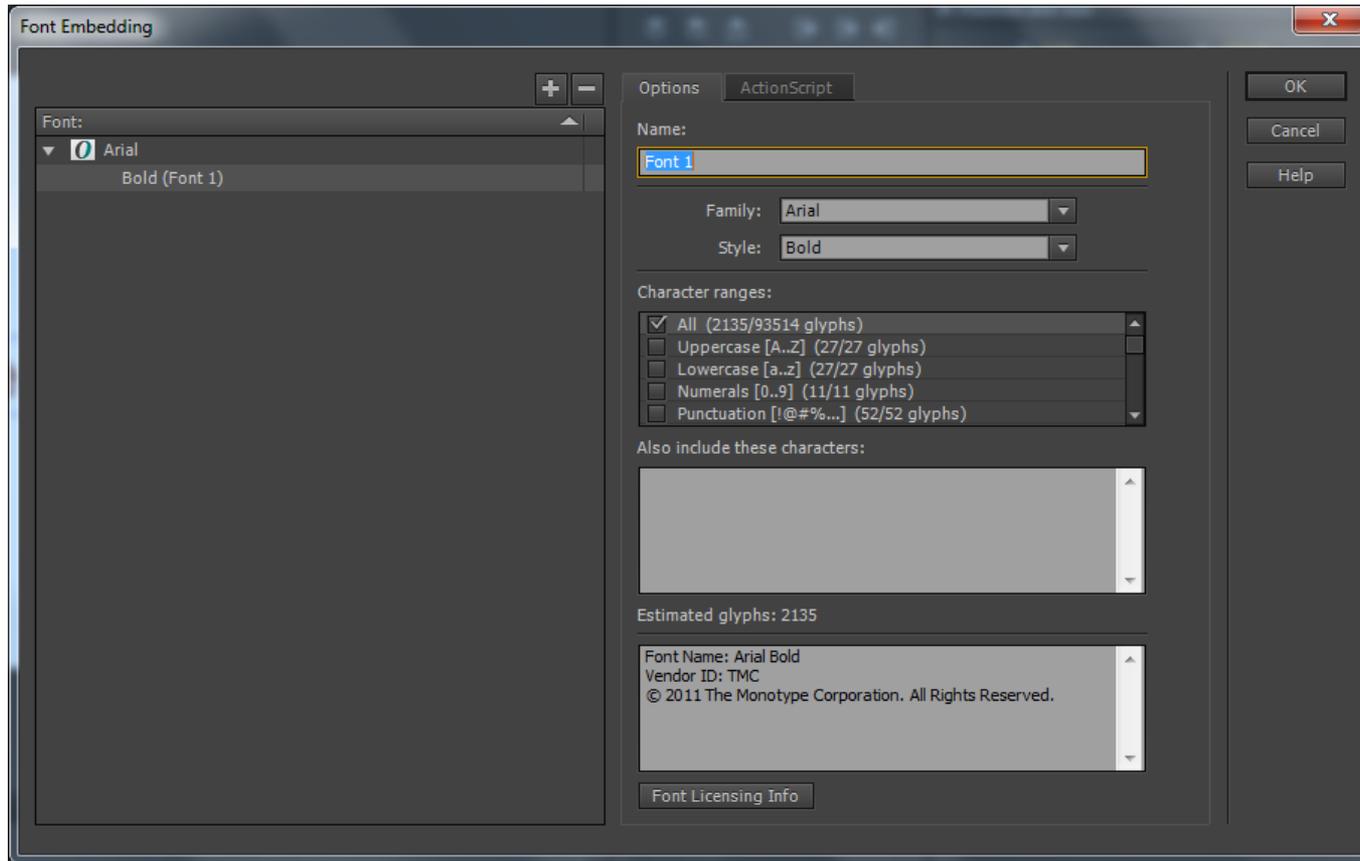
  if ((o2.visible == false) && (count == "even") && (select2.visible == true) && (game == "on"))
  {
    o2.visible = true;
    select2.visible = false;
    count = "odd";
    if (((o1.visible == true) && (o2.visible == true) && (o3.visible == true)) ||
        ((o4.visible == true) && (o5.visible == true) && (o6.visible == true)) ||
        ((o7.visible == true) && (o8.visible == true) && (o9.visible == true)) ||
        ((o1.visible == true) && (o4.visible == true) && (o7.visible == true)) ||
        ((o2.visible == true) && (o5.visible == true) && (o8.visible == true)) ||
        ((o3.visible == true) && (o6.visible == true) && (o9.visible == true)) ||
        ((o1.visible == true) && (o5.visible == true) && (o9.visible == true)) ||
        ((o3.visible == true) && (o5.visible == true) && (o7.visible == true)))
    {
      game = "off";
      gameOver.visible = true;
      gameOver.text = "Game Over - O Wins";
    }
  }
}
```

# Create a Dynamic Text



On the Slide layer, we will insert a keyframe at frame 2. We will add text “Game Over – X Wins” to the center of the stage. On the Properties tab, we will change the text type to Dynamic Text and the Instance name to gameOver.

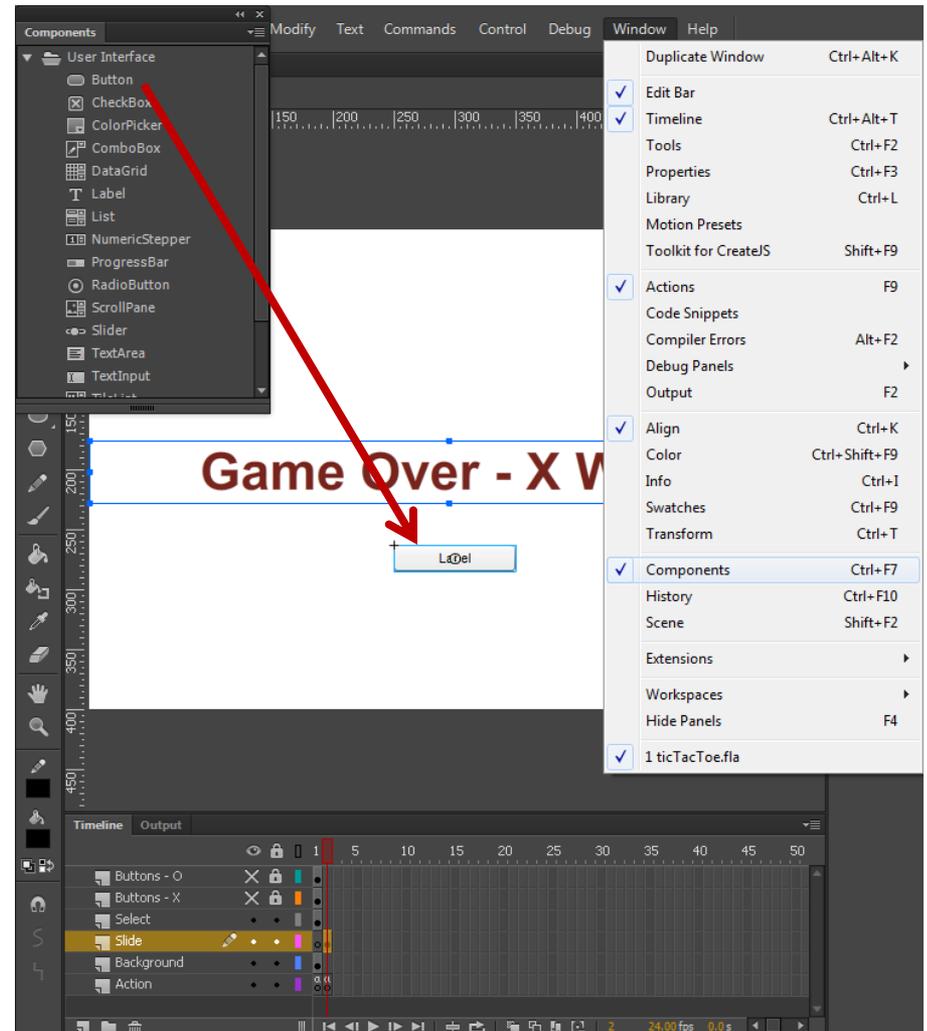
# Create a Dynamic Text



On the Properties tab, we will choose the Embed button. On the Characters ranges, we annotate the All checkbox.

# Make a Second Slide

To add a Button, we select Window on the menu and Components. In the Components window, we drag a button to the stage as shown.



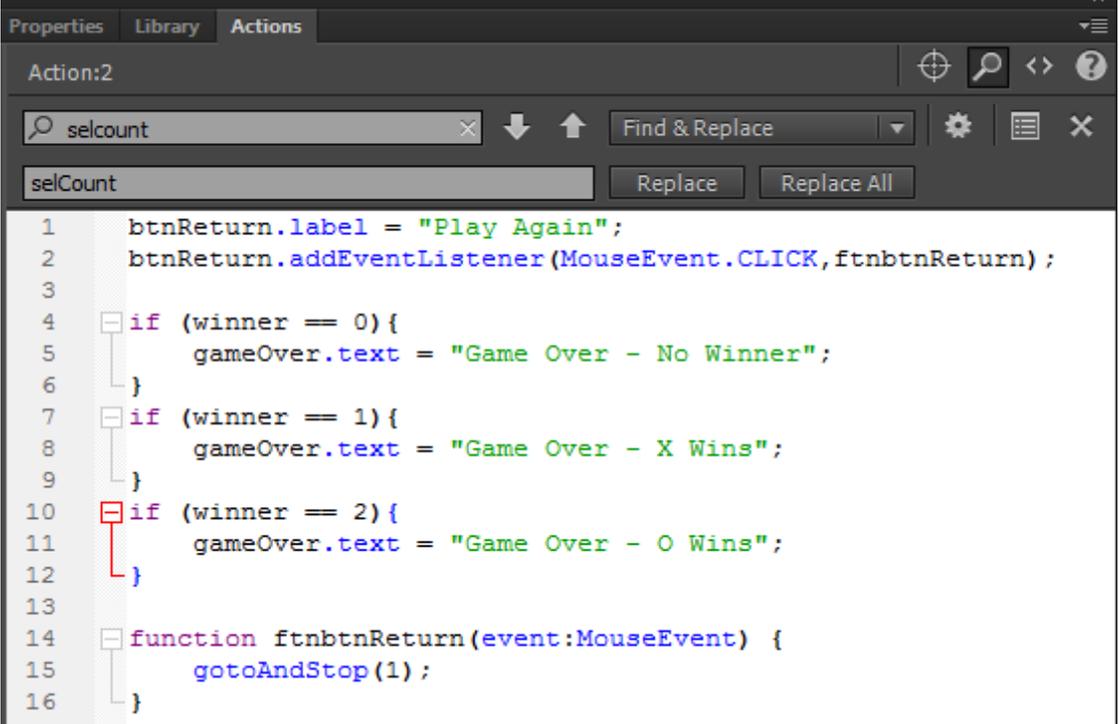
# Frame 2 – Action Script

We insert a keyframe in frame 2 on the Action layer.

We set the Return button label to Play again. We then add an event listener for the button.

We add three if statements to state whether there is a X, O or no winner.

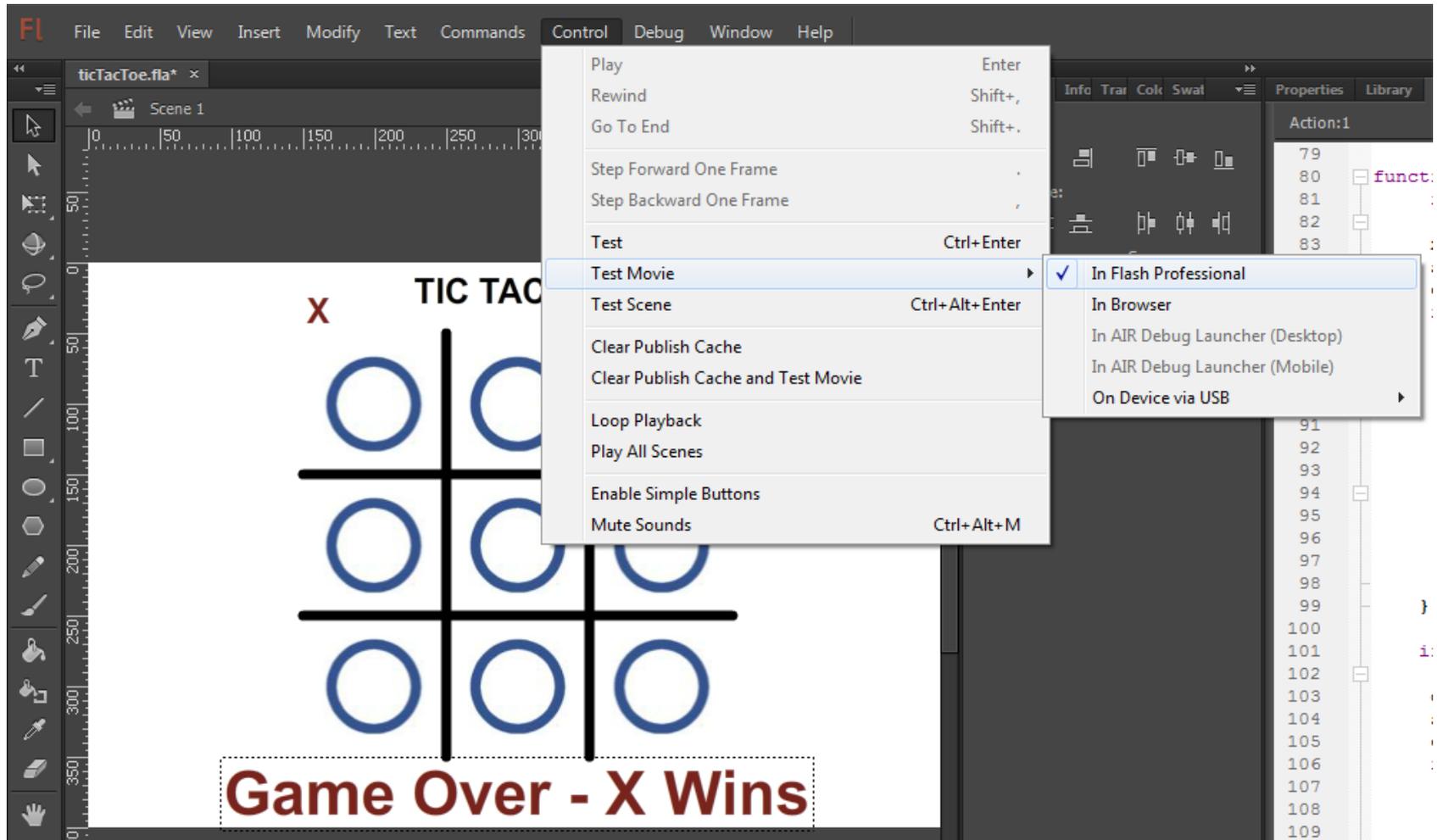
In the function for the return button, we use the gotoAndStop to frame 1.



The screenshot shows the ActionScript editor interface. The top bar includes 'Properties', 'Library', and 'Actions' tabs. Below the tabs, the text 'Action:2' is displayed. A search bar contains 'selcount', and there are 'Find & Replace', 'Replace', and 'Replace All' buttons. The main area contains the following code:

```
1 btnReturn.label = "Play Again";
2 btnReturn.addEventListener(MouseEvent.CLICK, ftnbtnReturn);
3
4 if (winner == 0){
5     gameOver.text = "Game Over - No Winner";
6 }
7 if (winner == 1){
8     gameOver.text = "Game Over - X Wins";
9 }
10 if (winner == 2){
11     gameOver.text = "Game Over - O Wins";
12 }
13
14 function ftnbtnReturn(event:MouseEvent) {
15     gotoAndStop(1);
16 }
```

# Test the Movie

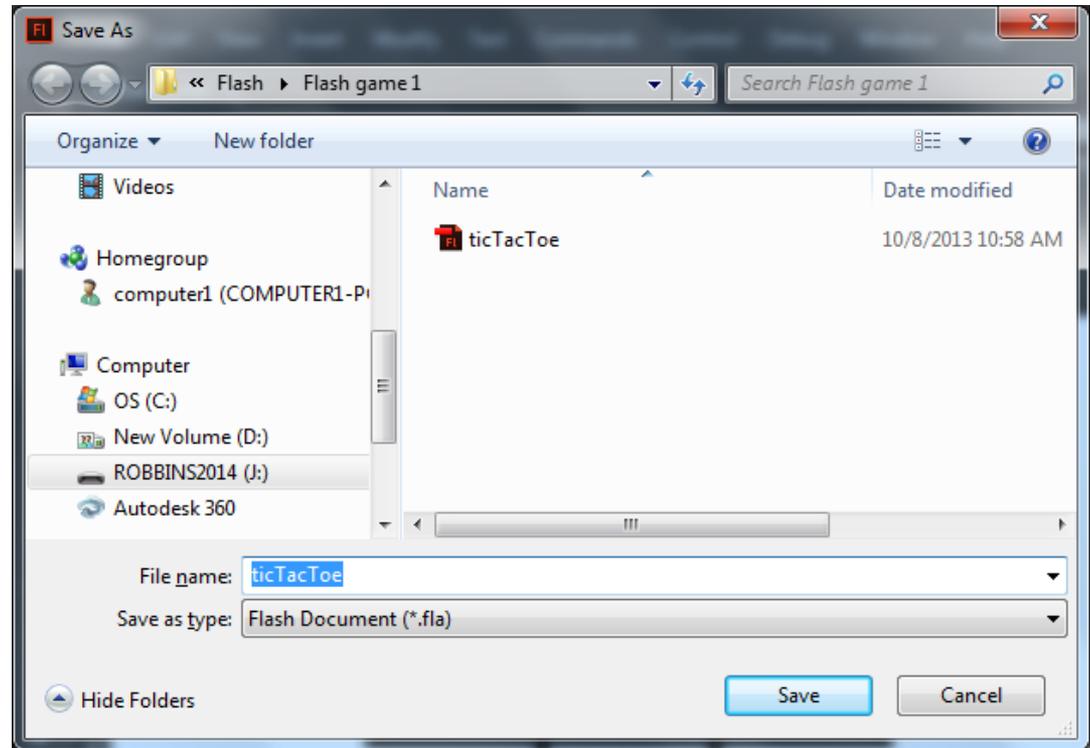


We should select Control on the menu and then Test Menu and In Flash Professional.

# 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 “ticTacToe” and we will depress the Save button.



# Publish the Flash Slide Show

We then choose File on the menu and Publish.

