

## Looping

Sometimes we need something to happen in our movies continuously or we'll want something to happen for a certain amount of time. That is where looping statements come into play. A simple example of a loop would be an animated gif that plays over and over again. The Timeline for that movie is set to repeat itself either infinitely (ghast!) or in a given number of cycles. We could create a Movie in Flash that would do this with a gotoAndPlay action at the last frame telling the Timeline to go to frame 1 and start playing the movie again. That's not what we're talking about here. We're talking about the built-in looping scripts available in ActionScript, namely, the While and For loop constructors.

The While loop is straightforward and basically says While this condition exists, do this. This type of loop is useful if you need to loop something until a certain condition exists and then stop it.

A For loop is useful when you know exactly how many times you want something to loop in a sequence. A For loop has 3 conditions:

For (Init; Condition; Next)

Here's an example in ActionScript:

```
For (I=0; I<10; I++)
```

Here you initialize a counter with the I=0, which fulfills our Init condition. Next, you create the circumstance, like you would with an if conditional, to look for, in this case I is less than 10. Then you increment the counter (sometimes decrement) with the I++ condition. ++ is the increment operator. So if the counter begins at 0 and is incremented, the numbers will increase until the Condition we set: I<10 is met. Once the counter reaches 10 the loop is done, there's no need to tell it to stop.

Note: I is a Variable name here and you'll see examples in other places that use J or another letter. It doesn't really matter as long as your consistent. Most programmers use I for "increment".

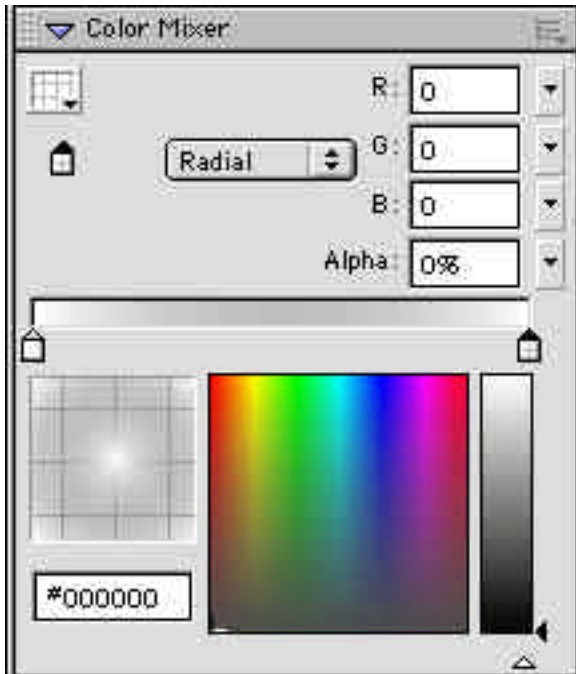
Note: If the Flash Player encounters a loop that runs continuously, it will eat up a lot of processor time and will give you a warning. If you get this message, you know you've done something wrong. Use the Debugger tools and check your syntax to see what's wrong.

## Creating a While ... Do Loop

In this example, we'll use the While Loop to create multiple copies of a Movie Clip to create the effect of a glowing, and growing, gas cloud in space. The duplicateMovieClip action will be used here and the Math.random property of the Math Object will also be used.

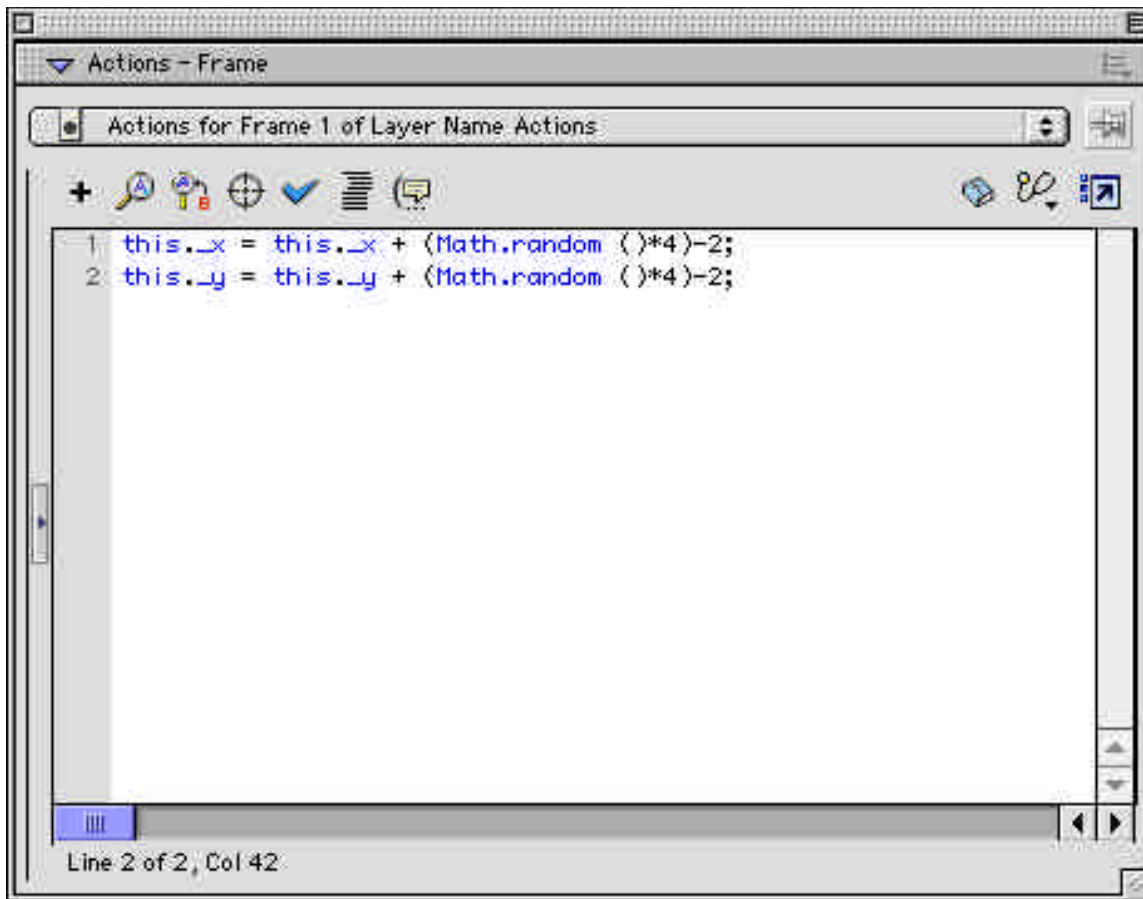
1. Open a new movie and give it a black background via Modify > Movie. Create a graphic symbol called plasma that consists of a white circle with a radial gradient that fades to

zero alpha at the edges. Click on the Circle tool in the Toolbox. Click on the Fill box and select the Black and White radial gradient. Click on the Stroke Fill box and select none. In the Color Mixer Panel, click on the black box with a triangle on the top. Reduce the Alpha value to 0.



2. Now create a Movie Clip symbol called random plasma.
3. Place the plasma symbol into the center of the random plasma Movie Clip on layer 1. Use the Align Panel to center align the plasma symbol on the stage.
4. Create a new layer and call it Actions.
5. Be sure that you are working in the Expert Mode of the Actions Panel by clicking on the icon in the upper right-hand corner of the Actions Panel and choosing Expert.
6. Type in the following code into the Actions Panel:

```
this._x = this._x + (Math.random ( )*4)-2  
this._y = this._y + (Math.random ( )*4)-2
```



7. This simply refers to the Movie Clip we are currently editing. `_x` and `_y` are the X and Y locations of the Movie Clip on the stage. The `Math.random` statement is changing that position to a random number and multiplying it by 4 then subtracting it by 2.
8. Add a keyframe in frame 2 of the Actions layer and add the following script:

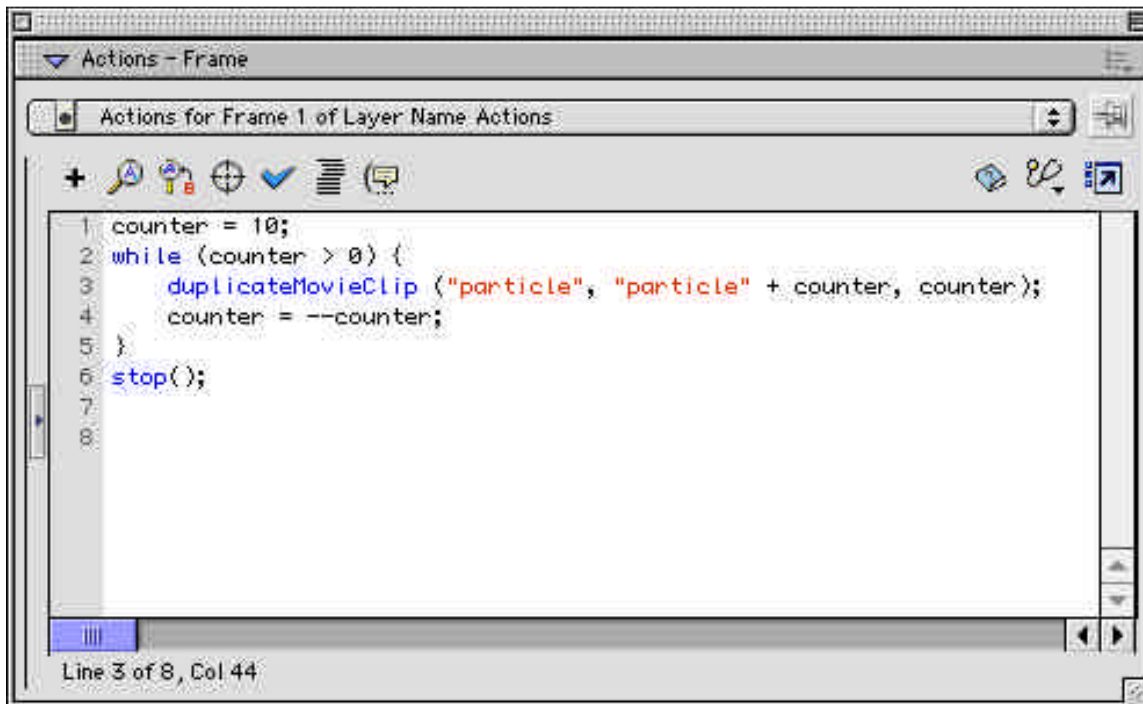
```
gotoAndPlay(1);
```

9. By sending the Timeline back to frame 1 we are invoking the `Math.random` statements again and therefore continuously changing the X and Y position of the Movie Clip.
10. Return to Scene 1 and drag random plasma onto the center of the stage.
11. Go to the Control Menu > Test Movie. You will see the plasma Movie Clip randomly pulsate and move around the screen. To make it look more like a cloud of gas we'll next use the `While` statement in conjunction with the `duplicateMovieClip` action to create more instances of the random plasma Movie Clip on the stage that will behave exactly in the same manner. Close the Test Movie Window.
12. Click on the random plasma Movie Clip on the stage. In the Properties Panel give it an Instance name `particle`.
13. On Scene 1 add a new layer called Actions.
14. Type in the following code:

```
counter = 10;
```

15. This Variable will be used 3 more times in our code, showing the power and flexibility of Variables.
16. Now type in the While statement:

```
while (counter > 0) {  
    duplicateMovieClip ("particle", "particle" + counter,  
        counter);  
    counter = --counter;  
}  
stop();
```



17. Let me break things down for you here. We've set up our counter initially to 10. The while loop uses the counter and checks to see if its greater than 0. Next the duplicateMovieClip action is used to create duplicate copies of the random plasma Movie Clip. As with all duplicateMovieClip statements, 3 conditions are required: first, the Movie Clip that will be duplicated (By the way, a Movie Clip doesn't have to be on the stage to be duplicated. There's an Export option for any Movie Clip that allows you to give it an Instance name and then you call on it via ActionScript), secondly, we have to give our new Movie Clips new Instance names. Here we are using the Instance name that we have already and then we are adding the value of the counter to it. So we would get particle0, particle1, particle2, etc. Finally, we have to set the Depth of the Movie Clip as they cannot all occupy the same level. So here again, we are cleverly using our Counter Variable to set the depth of each new Movie Clip. Didn't think you could get that much mileage out of a Variable did you?
18. The next line uses the counter Variable again, this time it Decrements or subtracts the number of the counter. Remember that we initially set the counter equal to 10. So the

While condition will last until we reach 0. The (--) decrement operator is being used here to achieve this.

19. Finally, and outside of the brackets of the While loop, we set a stop action to stop the loop from going on forever.
20. Test the Movie and let it play for a while. The blobs will go all over the stage.

Resources: Foundation ActionScript by Sham Bhargal. Friends of Ed press. ISBN: 1-903450-32-2.