

## Loops

A loop is a block of statements that is repeated. A loop has a Boolean expression to test. If the expression is true the statements in the block are executed and then the test is performed again. When the condition is false, the next statement after the loop is executed.

- A while loop and a for loop test a Boolean expression at the beginning.
- A do loop tests the Boolean expression at the end.
- While and for loops are executed *zero or more* times.
- Do loops are always executed *at least once*.

The Boolean expression is a check point: The decision is made to execute the entire loop or not. If the value of the Boolean expression changes during execution of the loop, the loop continues until the test is made again. The loop does not end in the middle if the variable changes.

### While Loops

The format for a while loop is: **while** ( \_\_\_\_\_ ) \_\_\_\_\_ ;  
Boolean expression                      statement or block of statements

**Example:** Start a new flash movie and write the code shown below in the ActionScript panel.

```
var num:int=0;
while(num<=5) {
    trace(num);
    num=num+1;
} //loop
```

When you test the movie you will see the numbers 0 to 5 in the output window.

Notice the indentation: all statement within { and } are indented the same amount for readability.

### Endless Loops

Occasionally, a programmer writes a loop that never ends. In fact, it is really easy to write an endless loop. If you create an endless loop, the user will get a message that something is making the script run really slow, and asks if they would like to end the movie. You do not want the user to see that message, so you should check for endless loops very carefully. The code below is an endless loop because the values for num that are generated are 0, 2, 4, 6, ... and it is NEVER 5! If you want to try this save the file first, then wait for Flash to crash!

```
var num:int=0;
while(num!=5) {
    trace(num);
    num=num+2;
} //loop
```

### for

A **for** loop is a good choice when we know how many times we want the code to repeat.

The format is shown below:

for ( \_\_\_\_\_ ; \_\_\_\_\_ ; \_\_\_\_\_ ) { \_\_\_\_\_ };  
initialize list    Boolean expression to continue    execute at end    body of loop statements

The code below will display the same numbers as the previous example:

```
for(var num:int=0; num<=5; num++) {  
    trace(num);  
} //loop
```

### do ... while

The **do...while** loop does the test at the end of the loop. A **do...while** always executes at least once. After each execution, the Boolean expression is tested, if true the loop repeats, if false the next statement after the loop is executed. The program below is the same as the one above except that a **do...while** is used. Try the code below to display the numbers 0 to 5:

```
var num:int=0;  
do {  
    trace(num);  
    num++;  
} while(num<=5);
```

**Experiment:** Try to generate the number 10,9,8,7,6,5,4,3,2,1 using while, for and do loops.

### Spinning Stars

Displaying numbers in the output window is not very exciting, let's do something practical, like making a bunch of stars spin.

- On the stage draw a star and make it a movie clip called Star with the registration point in the center..
- Put 4 instances of the Star on the stage.
- Name the instances star0, star1, star2, and star3.
- Write the following code:

```
this.addEventListener(Event.ENTER_FRAME,frames);  
function frames(e:Event):void {  
    var num:int=0;  
    while(num<4) {  
        this["star"+num].rotation++; //You will get an error if there is no "star"+num  
        num++;  
    } //loop  
} //frames
```

Notice that a movie clip called myClip could be referred to as this["myClip"] by adding the number to the word star we can loop through all of the stars on the stage.

**Experiment:**

- Make the stars spin faster.
- Make all of the stars move in addition to spinning.
- Declare a variable numStars and use that in the loop instead of 4.