Loops

Loops are used for counters or to repeat until a condition is met.

While Statement

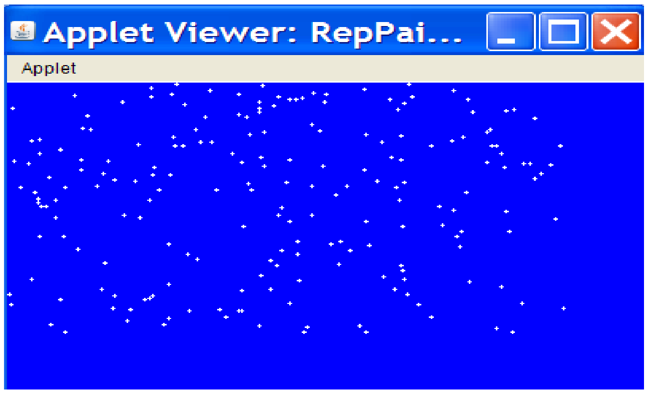
The *while* statement is used to repeat a block of code while some condition is true. The condition must become false somewhere in the loop, otherwise it will never terminate. The while loop always checks if the condition is false at the beginning of the statement.

In this example from Javalessons.com white dots are drawn on the screen at random locations until the counter is 200.

```
 1  import java.awt.* ;
 2  import java.applet.* ;
 3  /* javalessons.com
 4  This code is for educational purposes only.*/
 5
 6  public class RepPaint
 7          extends Applet
 8  {
 9      public void init()
10      {
11          setBackground ( Color.blue ) ;
12      }
13
14      public void paint( Graphics gr )
15      {
16          int x, y , count;
17          double r1 ;
18
19          gr.setColor ( Color.white ) ;
20
21          count = 0 ;
22          while ( count < 200 )
23          {
24              r1 = Math.random() ;
25              r1 = r1 * 350 ;
26              x = (int) r1 ;
27              r1 = Math.random() ;
28              y = (int)(r1 * 200) ;
29
30              gr.fillOval ( x, y, 3, 3 ) ;
31              count++ ;
32          }
33      }
34  }
35
36
```

Line 24 – 30 set the "x" and "y" coordinates to be drawn on the applet. This applet is using the random() method to generate a random locations for the dot to be placed to give it a starry sky effect.

RepPaint Applet



Here is another example of a While Statement in a simple application.

```
 1  // WhileLoop.java
 2  // Start with a penny
 3  // double it every day
 4  // how much do you have in a 30-day month?
 5
 6  import java.io.*;
 7
 8  public class WhileLoop
 9  {
10     public static void main(String args[]) throws IOException
11     {
12       int day = 0;
13       double money = .01, previousAmount = .01;
14
15       while(day < 30)
16       {
17        money = 2 * previousAmount;
18        previousAmount = money;
19        ++day;
20        System.out.println();
21        System.out.println("After day " + day + " you have " + money);
22
23      }
24     }
25  }
```

Note on line 19 the plus signs "++" are in front of the variable **day.** In the previous example on line 31, the plus signs came after the variable **count**. This next code example demonstrates a few different ways to increment a value. To de-increment a value replace the plus signs with the minus "-", sign.

```
int num;

num = 10;

++num; //adds one to 10 to equal 11

num = 10;

num++; //adds one to 10 to equal 11

num = 10;

num = num + 1; //adds one to 10 to equal 11

num = 10;

num += 1; //adds one to 10 to equal 11
```

On the next page is another example of a **While** loop. This example also has GUI swing components. To use the swing components you must first import javax.swing.* package. This package allows the use of message boxes and input boxes to name a few. This example uses the class JOptionPane for an input box and a message box. For more information on Java Swing Components please visit:
http://java.sun.com/docs/books/tutorial/ui/features/components.html

```java
// Input123.java
// Uses swing package to give user an input box
// Accepts input until  user quits

import javax.swing.*;

public class Input123
{
  public static void main (String args[])
  {
     int number;
     String entry, message;

     entry = JOptionPane.showInputDialog(null,"Enter 1, 2 or 3, or 4 to quit");
     number =  Integer.parseInt(entry);

     while(number != 4)
     {
        if(number == 1 || number == 2 || number == 3)

           message = "Good job!";

        else
           message = "You should enter 1, 2, 3, or 4";

        JOptionPane.showMessageDialog(null,message);

        entry = JOptionPane.showInputDialog(null, "Enter 1, 2 or 3, or 4 to quit");
        number =  Integer.parseInt(entry);
     }
      System.exit(0);
  }
}
```

For Statement

Many loops consist of three operations surrounding the body: (1) initialization of a variable, (2) testing a condition, and (3) updating a value before the next iteration. The **for** loop groups these three common parts together into one statement, making it more readable and less error-prone than the equivalent *while* loop. For repeating code a known number of times, the **for** loop is the right choice. In the example below line 9 initializes the variable "I", sets the initial value, test to see if the condition is met and increments the value of I by one.

```java
 1  // EverySum.java
 2  // An application that shows the sum of for every nunber from 1 to 50
 3
 4  public class EverySum
 5  {
 6    public static void main (String args[])
 7    {
 8       int sum = 0;
 9       for (int i = 1; i <= 50; i++)
10       {
11      sum += i;
12           System.out.print("   " + sum);
13       }
14    }
15  }
16
17
18
```

Programming Assignments

1. Recreate the 4 programs in this document.
    a. EverySum
    b. Input123
    c. Repaint
    d. Whileloop