Chapter 2: Java's while Loop

Java's while Loop

The while loop repeats a block of code as long as (or while) a condition is true. Its form is:

while (<*boolean expression*>) <*body*>

The rules for <*boolean expression*> in a while statement are the same as for an if statement. Likewise, the rules for the <*body*> of a while are the same as in an if statement.

Here's an example of a simple while loop that will display numbers from 1 to 10.

```
int x = 1;
while (x <= 10) {
    System.out.println(x);
    x = x + 1;
}
```

Text equivalent start.

In the while loop with the expression x less than or equal 10, and a variable x with an initial value of 1, the while loop begins to execute. If you increment x by 1 each time through the loop, it will continue to loop until x is 11. Then the loop exits.

Text equivalent stop.

The first thing this loop does is check to see if x is 10 or less. The loop body won't execute unless the condition is true. A while loop checks the condition before the loop's body executes, and if the condition is false, it skips the body, stops the loop, and continues with whatever code follows the loop.

Since the variable x in this case starts out with a value of 1, the body of the loop executes. The body's first line displays the value of x on the screen, and the next line *increments* (increases by 1) the value of x.

Let's look at the increment statement for a moment.

In this case, x is on both sides of the equal sign. This is one of the programming practices that may seem strange to someone just learning to program. But remember that this statement isn't an equation; it's an assignment. Java evaluates the right side of the statement before storing anything in the left side. So in this case, the statement tells Java to take the value from x, add 1 to it, and store it back into x. You'll see lots of variations on this theme in programming.

At the end of the first time through the loop, then, x has a value of 2. Since this is still less than 10, the loop repeats, then repeats again, incrementing x each time until x reaches a value of 11. At that point the condition becomes false, and the loop stops.

Using a while Loop

Now that you know what a while loop looks like, how can we use it in our program?

The first loop we're going to add is the one that will repeat the whole process until our user enters Q for a temperature type. This loop will include all the actions in our main() method, so we'll start it right after our variable declarations.

Before starting the loop, however, let's discuss about how to control its repetitions. One way would be to use a condition that checks that the entry isn't Q. That would look like this:

```
while ( ! temperatureType.equals("Q"))
```

As you can see, that type of condition is somewhat awkward to deal with and read. I try to avoid negative conditions when I can because they're confusing. So I'll control my loop with another common programming technique.

I'm going to declare a boolean variable whose only purpose is to control my loop. Since it will tell me whether my user has more temperatures to convert, I'm going to name it moreTemperatures and set its initial value to true. We'll repeat our loop as long as the value remains true. So let's add a line to our variable declarations that looks like this:

```
boolean moreTemperatures = true; // more temperatures to convert?
```

Now that we have a simple variable we can test to control our loop, let's start the loop like this:

```
while (moreTemperatures) {
```

I want to make sure it's clear that everything following main() is part of the loop. So let's indent everything from there to the end of main() and then add another closing curly bracket to end the loop.

Before I show you the complete new main() method, we need to add the option for the letter Q. Once I do that and change the text slightly using equal signs to shorten it, our main() method looks like this:

```
public static void main(String[] args) {
    double inputTemperature = 0.0;
    Scanner keyInput = new Scanner(System.in);
    Temperature t1;
    String temperatureType = "";
    boolean moreTemperatures = true;

    while (moreTemperatures) {
        System.out.print("Enter a temperature type (C=Celsius, " +
                "F=Fahrenheit, K=Kelvin, Q=Quit): ");
        temperatureType = keyInput.next();
        System.out.print("Enter a temperature: ");
        inputTemperature = keyInput.nextDouble();
        t1 = new Temperature(temperatureType, inputTemperature);

    if (temperatureType.equalsIgnoreCase("F")) {
        System.out.println("You entered " + inputTemperature +
                " degrees Fahrenheit");
        System.out.println("which is " + t1.getDegreesCelsius() +
                " degrees Celsius");
        System.out.println("and " + t1.getDegreesKelvin() +
                " degrees Kelvin.");
    }
    else if (temperatureType.equalsIgnoreCase("C")) {
        System.out.println("You entered " + inputTemperature +
                " degrees Celsius");
        System.out.println("which is " + t1.getDegreesFahrenheit() +
                " degrees Fahrenheit");
        System.out.println("and " + t1.getDegreesKelvin() +
                " degrees Kelvin.");
        }
        else if (temperatureType.equalsIgnoreCase("K")) {
            System.out.println("You entered " + inputTemperature +
                    " degrees Kelvin");
            System.out.println("which is " + t1.getDegreesCelsius() +
                    " degrees Celsius");
            System.out.println("and " + t1.getDegreesFahrenheit() +
                    " degrees Fahrenheit.");
        }
    }
}
```

Okay, now I have a loop, but it will never end because I never change the value of moreTemperatures inside the loop. One of the most common programming errors related to loops is failing to update the variable that ends the loop. I can fix that with a simple if statement:

```
if (temperatureType.equalsIgnoreCase("Q")) { // quit
    moreTemperatures = false;
    System.out.println("\nProgram ended.");
}
```

I put this code just after reading the temperature type and before reading the temperature itself. Of course, the flip side of this if statement is that if the condition is false, we want to convert a temperature. So let's indent the rest of the code in main() again and put it all into an "else" block that will read the temperature and then convert it to the other types.
This program will now keep running until our user enters Q as the temperature type. We don't have to keep restarting the program to enter another temperature. Here's a screen shot of a single run of my program at this point:

Text equivalent start.

Terminal Window displaying example output:

Enter a temperature type (C=Celsius, F= Fahrenheit, K= Kelvin, Q=Quit): C
Enter a temperature: 0
You entered 0.0 degrees Celsius
which is 32.0 degrees Fahrenheit
and 273.15 degrees Kelvin.
Enter a temperature type (C for Celsius, F for Fahrenheit, K for Kelvin): F
Enter a temperature: 98.6
You entered 98.6 degrees Fahrenheit
which is 37.0 degrees Celsius
and 310.15 degrees Kelvin.
Enter a temperature type (C for Celsius, F for Fahrenheit, K for Kelvin): K
Enter a temperature: 300
You entered 300.0 degrees Kelvin
which is 26.850000000000023 degrees Celsius
and 80.33000000000004 degrees Fahrenheit.
Enter a temperature type (C for Celsius, F for Fahrenheit, K for Kelvin): q

Program ended.

Text equivalent stop.

I called this the *outer loop* in the caption because we're going to put more loops inside it. These *inner loops* will execute independently inside the outer loop—possibly many times, depending on user input. These inner loops will control edits so we get good data for our conversions. We aren't going to assume any longer that our users are perfect and won't make mistakes.

These inner loops and edits will be the topic of the next two chapters.

In case you have any problems at this point and would like to compare your program to mine, here's a link to my complete driver program:

Hide answer

```
import java.util.Scanner;
/**
 * TemperatureDriver runs and tests the Temperature class.
 *
 * @author Merrill Hall
 * @version 3.0
 */
public class TemperatureDriver {
    /**
     * main() reads a temperature type and value, then
     * converts it to the other two temperature scales.
     */
    public static void main(String[] args) {
        double inputTemperature = 0.0;
        Scanner keyInput = new Scanner(System.in);
        Temperature t1;
        String temperatureType = "";
        boolean moreTemperatures = true;

        while (moreTemperatures) {
            System.out.print("Enter a temperature type (C=Celsius, " +
                "F=Fahrenheit, K=Kelvin, Q=Quit): ");
            temperatureType = keyInput.next();
            if (temperatureType.equalsIgnoreCase("Q")) { // quit
                moreTemperatures = false;
                System.out.println("\nProgram ended.");
            }
            else {
                System.out.print("Enter a temperature: ");
                inputTemperature = keyInput.nextDouble();
                t1 = new Temperature(temperatureType, inputTemperature);

                if (temperatureType.equalsIgnoreCase("F")) {
                    System.out.println("You entered " + inputTemperature +
                        " degrees Fahrenheit");
                    System.out.println("which is " + t1.getDegreesCelsius() +
                        " degrees Celsius");
                    System.out.println("and " + t1.getDegreesKelvin() +
                        " degrees Kelvin.");
                }
                else if (temperatureType.equalsIgnoreCase("C")) {
                    System.out.println("You entered " + inputTemperature +
```

```
                    " degrees Celsius");
                System.out.println("which is " + t1.getDegreesFahrenheit() +
                    " degrees Fahrenheit");
                System.out.println("and " + t1.getDegreesKelvin() +
                    " degrees Kelvin.");
            }
            else if (temperatureType.equalsIgnoreCase("K")) {
                System.out.println("You entered " + inputTemperature +
                    " degrees Kelvin");
                System.out.println("which is " + t1.getDegreesCelsius() +
                    " degrees Celsius");
                System.out.println("and " + t1.getDegreesFahrenheit() +
                    " degrees Fahrenheit.");
            }
        }
    }
}
```