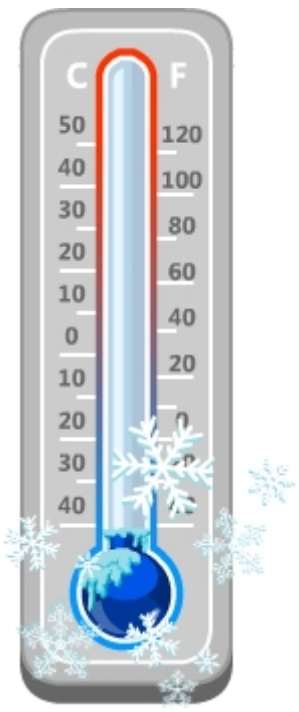
Chapter 4: Checking for Valid Temperatures

Checking for Valid Temperatures

The last thing we need to check is the value our user enters for a temperature. We need to check several things before we can decide if it's valid.

First we need to make sure it's a number. And once we know it's a number, we need to make sure it's valid for the type of temperature we're converting. For a Kelvin temperature, the value can't be less than zero. For Celsius, it can't be less than −273.15. And for Fahrenheit, it can't be less than −459.67. These temperatures represent absolute zero, the coldest temperature that's possible in our universe.



We'll need a loop for this, since we need to repeat the checks until we get a valid temperature. Back in the introduction to this lesson, I said we'd look at two Java loop formats in this lesson. I showed you the while loop already. Now we're going to use a do-while loop, sometimes just called a do loop. Its format is a little different from the while loop's, and it looks like this:

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

As you can see, there's the keyword do in front of the loop, and then the body with the statement or statements to be repeated. Last, there's the keyword while with the condition to be checked.

Notice the semicolon at the end of the loop. This while clause isn't followed by any other part of the loop, so the semicolon's there to end the statement.

What are the execution differences between the do and while loops?

- The do loop always executes the body of the loop at least once.
- The do loop checks the condition at the end of each repetition, and the while loop checks its condition at the start of each pass through the loop. So the while loop may never execute the loop body if the condition is false the first time it checks.

Text equivalent start.

Setting a false condition for a while loop means that the loop won't be executed. In contrast, a do or while loop will always run once, even for a false condition, since the condition isn't checked until the end of the loop.

Text equivalent stop.

We're going use another boolean variable for this loop. Since this variable will tell me whether we have a good temperature, we're going to name it goodTemperature and initialize it to false. So let's add this declaration to the top of our main() method:

boolean goodTemperature = false;

Also, since we may repeat this process for multiple temperatures, we'll need to reset it every time we start this loop, just as we did above for goodType.

We're ready to lay out our last loop now. I'm going to show you an abbreviated form of our main() method with most of the code missing so you can see the loop and decision structure. That way we'll all be on the same page before we make any more changes.

```
public static void main(String[] args) {
    // previous variable definitions
    boolean moreTemperatures = true; // more temperatures?
    boolean goodType = false; // good temperature type?
    boolean goodTemperature = false; // good temperature value?

    while (moreTemperatures) {
        // read the temperature type

        // check type
        goodType = false;
        while ( ! goodType) {
            if (inputType.equalsIgnoreCase("Q") ||
                    inputType.equalsIgnoreCase("C") ||
                    inputType.equalsIgnoreCase("F") ||
                    inputType.equalsIgnoreCase("K")) {
                goodType = true;
            }
            else {
                // generate error message and ask for type again
            }
    }

    if (inputType.equalsIgnoreCase("Q")) {
        // code to quit the program
    }
    else {
        goodTemperature = false;
        do {
            // code to read temperature
        } while ( ! goodTemperature);
        // code to process temperature
    }
}
```

This last inner loop goes at the start of the else clause of the if statement that checks for Q. Everything that was inside that else clause goes after this loop. The loop will repeat until goodTemperature becomes true, and then the loop will stop.

Once we know we have a good temperature, we can set goodTemperature to true, and then process the temperature value. Let me show you how that will work.

What's the first thing we need to do in this last loop in order to make sure we have a valid temperature? We need to get a temperature value from our user in order to check it. So let's put our prompt and input statement in there, which will make the loop look like this:

```
goodTemperature = false;
do {
    System.out.print("Enter a temperature: ");
    inputTemperature = keyInput.nextDouble();
    // the rest of the code to check temperatures goes here
} while ( ! goodTemperature);
```

We definitely want to start our loop with the prompt, but what about the second statement? Remember what happened when we tried to read a number, and the user typed something else? Our program crashed. We want to determine if there's a number to read before we try to read it. Fortunately, the Scanner class has a method that will help us.
In addition to its nextDouble() method, Scanner has a method named hasNextDouble() that will do what we want. Before we actually try to read the value, it'll tell us if the user has entered a valid number by giving us back a value of true if the user entered a number and a value of false if not.

We can use a call to this method as the condition in an if statement that will decide whether to read a number. If a call to hasNextDouble() comes back true , we'll read the number. If it comes back false, we'll display an error message. If the input isn't a number, we'll also need to read it to get it out of the input stream, but we don't really care what it is, so we don't need to save it in a variable.

Here is code that does what we want:

```
if (keyInput.hasNextDouble()) {
    inputTemperature = keyInput.nextDouble();
}
else {
    System.out.println("You entered an invalid temperature!");
    System.out.println("It must be a numeric value.");
    System.out.println("Try again.");
    keyInput.next();
}
```

Now we know whether we have a number, and we've handled the case where we don't by telling our user to try again.

The final edits will check if we have a valid temperature for each temperature type. Since we already know that our temperature is numeric, the only check we have left to make is whether that number is above absolute zero. So let's add that check and set goodTemperature if everything's okay and put out an error message if it's not. Here's the if statement we'll use:

```
if ((temperatureType.equalsIgnoreCase("C") && inputTemperature >= -273.15) ||
    (temperatureType.equalsIgnoreCase("F") && inputTemperature >= -459.67) ||
    (temperatureType.equalsIgnoreCase("K") && inputTemperature >= 0))
    goodTemperature = true;
else {
    System.out.println("You entered an invalid temperature!");
    System.out.println("It must be greater than absolute zero.");
    System.out.println("Try again.");
}
```

That statement should go right after the call to nextDouble(), which makes our loop (including the boolean reset in front of it) look like this:

```
goodTemperature = false;
do {
    System.out.print("Enter a temperature: ");
    if (keyInput.hasNextDouble()) {
        inputTemperature = keyInput.nextDouble();
        if ((temperatureType.equalsIgnoreCase("C") && inputTemperature >= -273.15) ||
            (temperatureType.equalsIgnoreCase("F") && inputTemperature >= -459.67) ||
            (temperatureType.equalsIgnoreCase("K") && inputTemperature >= 0))
            goodTemperature = true;
        else {
            System.out.println("You entered an invalid temperature!");
            System.out.println("It must be greater than absolute zero.");
            System.out.println("Try again.");
        }
    }
    else {
        System.out.println("You entered an invalid temperature!");
        System.out.println("It must be a numeric value.");
        System.out.println("Try again.");
        keyInput.next();
    }
} while ( ! goodTemperature);
```

Believe it or not, we're done! (Well, we're done with this lesson.) Try your program, and see if it does what we advertised up front. Just to prove it works, here's some output from my version of it:

Text equivalent start.

Terminal Window displaying example output:

```
Enter a temprature type (C=Celsius, F= Fahrenheit, K= Kelvin, Q=Quit): X
Invalid temprature type!
The type must be C, F, K or Q.
Please enter the temprature type again: C
Enter a temprature: x
You entered invalid temprature!
It must be a numeric value.
Try again.
Enter a temprature: -40
You entered -40 degrees Celsius
which is -40.0 degrees Fahrenheit
and 233.1499999999998 degrees Kelvin.
Enter a temprature type (C for Celsius, F for fahrenheit, K for Kelvin): q

Program ended.
```

Text equivalent stop.

And here's a copy of my program for you to compare to if you'd like.

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; // another temperature?
        boolean goodType = false; // good temperature type?
        boolean goodTemperature = false; // good temperature value?

        while (moreTemperatures) {
            System.out.print("Enter a temperature type (C=Celsius, " +
                "F=Fahrenheit, K=Kelvin, Q=Quit): ");
            temperatureType = keyInput.next();
            goodType = false;
            while ( ! goodType) {
                if (temperatureType.equalsIgnoreCase("Q") ||
                temperatureType.equalsIgnoreCase("C") ||
                temperatureType.equalsIgnoreCase("F") ||
                temperatureType.equalsIgnoreCase("K")) {
                    goodType = true;
                }
                else {
                    System.out.println("Invalid temperature type!");
                    System.out.println("The type must be C, F, K, or Q.");
                    System.out.print("Please enter the temperature type again: ");
                    temperatureType = keyInput.next();
                }
            }
            if (temperatureType.equalsIgnoreCase("Q")) { // quit
                moreTemperatures = false;
```

```
                System.out.println("\nProgram ended.");
            }
            else {
                goodTemperature = false;
                do {
                    System.out.print("Enter a temperature: ");
                    if (keyInput.hasNextDouble()) {
                        inputTemperature = keyInput.nextDouble();
                        if ((temperatureType.equalsIgnoreCase("C") &&
inputTemperature >= -273.15) ||
                                (temperatureType.equalsIgnoreCase("F") &&
inputTemperature >= -459.67) ||
                                (temperatureType.equalsIgnoreCase("K") &&
inputTemperature >= 0))
                            goodTemperature = true;
                        else {
                            System.out.println("You entered an invalid
temperature!");
                            System.out.println("It must be greater than absolute
zero.");
                            System.out.println("Try again.");
                        }
                    }
                    else {
                        System.out.println("You entered an invalid temperature!");
                        System.out.println("It must be a numeric value.");
                        System.out.println("Try again.");
                        keyInput.next();
                    }
                } while ( ! goodTemperature);

                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.");
        }
                    }
                }
            }
        }
}
```