

JB0115: JAVA OOP: FIRST PROGRAM*

R.G. (Dick) Baldwin

This work is produced by OpenStax-CNX and licensed under the
Creative Commons Attribution License 3.0[†]

Abstract

The purpose of this module is to present the first complete Java program of the collection that previews the most common forms of the three pillars of procedural programming: sequence, selection, and loop. The program also previews calling a method, passing a parameter to the method, and receiving a returned value from the method.

1 Table of Contents

- Preface (p. 1)
 - Viewing tip (p. 2)
 - * Figures (p. 2)
 - * Listings (p. 2)
- Discussion (p. 2)
 - Instructions for compiling and running the program (p. 2)
 - Comments (p. 2)
 - Program output (p. 2)
- Run the program (p. 3)
- Miscellaneous (p. 3)
- Complete program listing (p. 4)

2 Preface

The purpose of this module is to present the first complete Java program of the collection that previews the most common forms of the three pillars of procedural programming:

- sequence
- selection
- loop

The program also illustrates

- calling a method,
- passing a parameter to the method, and

*Version 1.1: Nov 25, 2012 8:26 am -0600

[†]<http://creativecommons.org/licenses/by/3.0/>

- receiving a returned value from the method.

As mentioned above, this is simply a preview. Detailed discussions of these topics will be presented in future modules.

2.1 Viewing tip

I recommend that you open another copy of this module in a separate browser window and use the following links to easily find and view the figures and listings while you are reading about them.

2.1.1 Figures

- Figure 1 (p. 3) . Program output.

2.1.2 Listings

- Listing 1 (p. 4) . Source code for FirstProgram.

3 Discussion

3.1 Instructions for compiling and running the program

Assuming that the Java Development Kit (JDK) is properly installed on your computer (see *Jb0110: Java OOP: Programming Fundamentals, Getting Started*¹), do the following to compile and run this program.

1. Copy the text from Listing 1 (p. 4) into a text file named **FirstProgram.java** and store the file in a folder on your disk.
2. Open a command-line window in the folder containing the file.
3. Type the following command at the prompt to compile the program:
javac FirstProgram.java
4. Type the following command at the prompt to run the program:
java FirstProgram

3.2 Comments

Any text in the program code that begins with `//` is a comment. The compiler will ignore everything from the `//` to the end of the line.

Comments were inserted into the program code to explain the code.

The compiler also ignores blank lines.

Note that this program was designed to illustrate the concepts while being as non-cryptic as possible.

3.3 Program output

The program should display the text shown in Figure 1 (p. 3) on the screen except that the time will be different each time you run the program.

¹<http://cnx.org/content/m45137>

Program output.

```
value in = 5
Odd time = 1353849164875
countA = 0
countA = 1
countA = 2
countB = 0
countB = 1
countB = 2
value out = 10
```

Figure 1: Program output.

4 Run the program

I encourage you to copy the code from Listing 1 (p. 4) . Compile the code and execute it. Experiment with the code, making changes, and observing the results of your changes. Make certain that you can explain why your changes behave as they do.

5 Miscellaneous

This section contains a variety of miscellaneous information.

NOTE: Housekeeping material

- Module name: Jb0115: Java OOP: First Program
- File: Jb0115.htm
- Published: November 25, 2012

NOTE: Disclaimers: Financial : Although the Connexions site makes it possible for you to download a PDF file for this module at no charge, and also makes it possible for you to purchase a pre-printed version of the PDF file, you should be aware that some of the HTML elements in this module may not translate well into PDF.

I also want you to know that, I receive no financial compensation from the Connexions website even if you purchase the PDF version of the module.

In the past, unknown individuals have copied my modules from cnx.org, converted them to Kindle books, and placed them for sale on Amazon.com showing me as the author. I neither receive compensation for those sales nor do I know who does receive compensation. If you purchase such a book, please be aware that it is a copy of a module that is freely available on cnx.org and that it was made and published without my prior knowledge.

Affiliation : I am a professor of Computer Information Technology at Austin Community College in Austin, TX.

6 Complete program listing

A complete listing of the program follows.

Listing 1: Source code for FirstProgram.

```
/* Begin block comment
This is the beginning of a block comment in Java.
Everything in this block comment is for human consumption
and will be ignored by the Java compiler.
```

```
File: FirstProgram.java
Copyright 2012, R.G. Baldwin
```

This program is designed to illustrate the most common forms of the three pillars of procedural programming in Java code:

```
sequence
selection
loop
```

The program also illustrates calling a method, passing a parameter to the method, and receiving a returned value from the method.

Assuming that the Java Development Kit (JDK) is properly installed on your computer, do the following to compile and run this program.

1. Copy this program into a file named FirstProgram.java and store the file in a folder on your disk.
2. Open a command-line window in the folder containing the file.
3. Type the following command to compile the program:
javac FirstProgram.java
- 4.4. Type the following command to run the program:
java FirstProgram

Any text that begins with // in the following program code is a comment. The compiler will ignore everything from the // to the end of the line.

The compiler also ignores blank lines.

Note that this program was designed to illustrate the

concepts while being as non-cryptic as possible.

The program should display the following text on the screen except that the time will be different each time that you run the program.

```
value in = 5
Odd time = 1353849164875
countA = 0
countA = 1
countA = 2
countB = 0
countB = 1
countB = 2
value out = 10

End block comment *****/

//The actual program begins with the next line.
import java.util.*;

class FirstProgram{
    //The program consists of a sequence of statements.

    //The next statement is the beginning of the main
    // method, which is required in all Java applications.
    public static void main(String[] args){
        //Program execution begins here.

        //Declare and initialize a variable.
        int var = 5;

        //Statements of the following type display
        // information on the screen
        System.out.println("value in = " + var);

        //Call a method and pass a parameter to the method.
        //Save the returned value in var, replacing what
        // was previously stored there.
        //Control is passed to the method named firstMethod.
        var = firstMethod(var);

        //Control has returned from the method named
        // firstMethod.
        System.out.println("value out = " + var);

        //Program execution ends here
    }//end main method

    /****visual separator comment*****/
```

```
public static int firstMethod(int inData){
    //Control is now in this method.

    //Illustrate selection
    //Get the elapsed time in milliseconds since Jan 1970.
    long time = new Date().getTime();

    //Select even or odd time and display the results
    if(time % 2 == 0){
        System.out.println("Even time = " + time);
    }else{
        System.out.println("Odd time = " + time);
    }//end if-else selection

    //Illustrate a while loop
    int countA = 0;
    while(countA < 3){
        System.out.println("countA = " + countA);
        //Increment the counter
        countA = countA + 1;
    }//end while loop

    //Illustrate a for loop
    for(int countB = 0; countB < 3; countB = countB + 1){
        System.out.println("countB = " + countB);
    }//end for loop

    //Illustrate returning a value from a method and
    // returning control back to the calling method.
    return 2*inData;
} //end firstMethod

} //end class FirstProgram
//The program ends with the previous line.

-end-
```