

Java Basics Unit

Lesson 6: Debugging

Copyright © 2016 The Learning House, Inc.

All rights reserved. No part of these materials may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of The Learning House. For permission requests, write to The Learning House, addressed "Attention: Permissions Coordinator," at the address below.

The Learning House

427 S 4th Street #300

Louisville KY 40202

Lesson 6 Debugging

Overview

In this lesson, we will take a closer look at how to debug a program. Using the code from the original WindowMaster program, we'll see how to set breakpoints, execute the program in debug mode, step through the program, and examine the values of variables while the program is executing.

Example

If you have the original code for WindowMaster, open the project and make sure that it compiles and runs correctly. If you don't have that code anymore, create a new project called WindowMasterDebug and paste the following code into the main method. Once that is done, save the file and make sure that the program runs correctly.

The Code

```
public static void main(String[] args) {
    // Declare variables for height and width
    float height;
    float width;
    // Declare String variables to hold the user's height and width
    // input
    String stringHeight;
    String stringWidth;

    // Declare other variables
    float areaOfWindow;
    float cost;
    float perimeterOfWindow;

    // Declare and initialize our Scanner
    Scanner sc = new Scanner(System.in);

    // Get input from user
    System.out.println("Please enter window height:");
    stringHeight = sc.nextLine();
    System.out.println("Please enter window width:");
    stringWidth = sc.nextLine();

    // Convert String values of height and width to floats
    height = Float.parseFloat(stringHeight);
    width = Float.parseFloat(stringWidth);

    // Calculate area of window
    areaOfWindow = height * width;

    // Calculate the perimeter of the window
    perimeterOfWindow = 2 * (height + width);

    // Calculate total cost - use hard coded for material cost
    cost = ((3.50f * areaOfWindow) + (2.25f * perimeterOfWindow));

    System.out.println("Window height = " + stringHeight);
    System.out.println("Window width = " + stringWidth);
    System.out.println("Window area = " + areaOfWindow);
    System.out.println("Window perimeter = " + perimeterOfWindow);
    System.out.println("Total Cost = " + cost);
}
```

Setting Breakpoints

Setting breakpoints in NetBeans is easy — just click in the leftmost margin of the code window on the line at which you want to put the breakpoint. For our example, set breakpoints at lines 34 and 43:

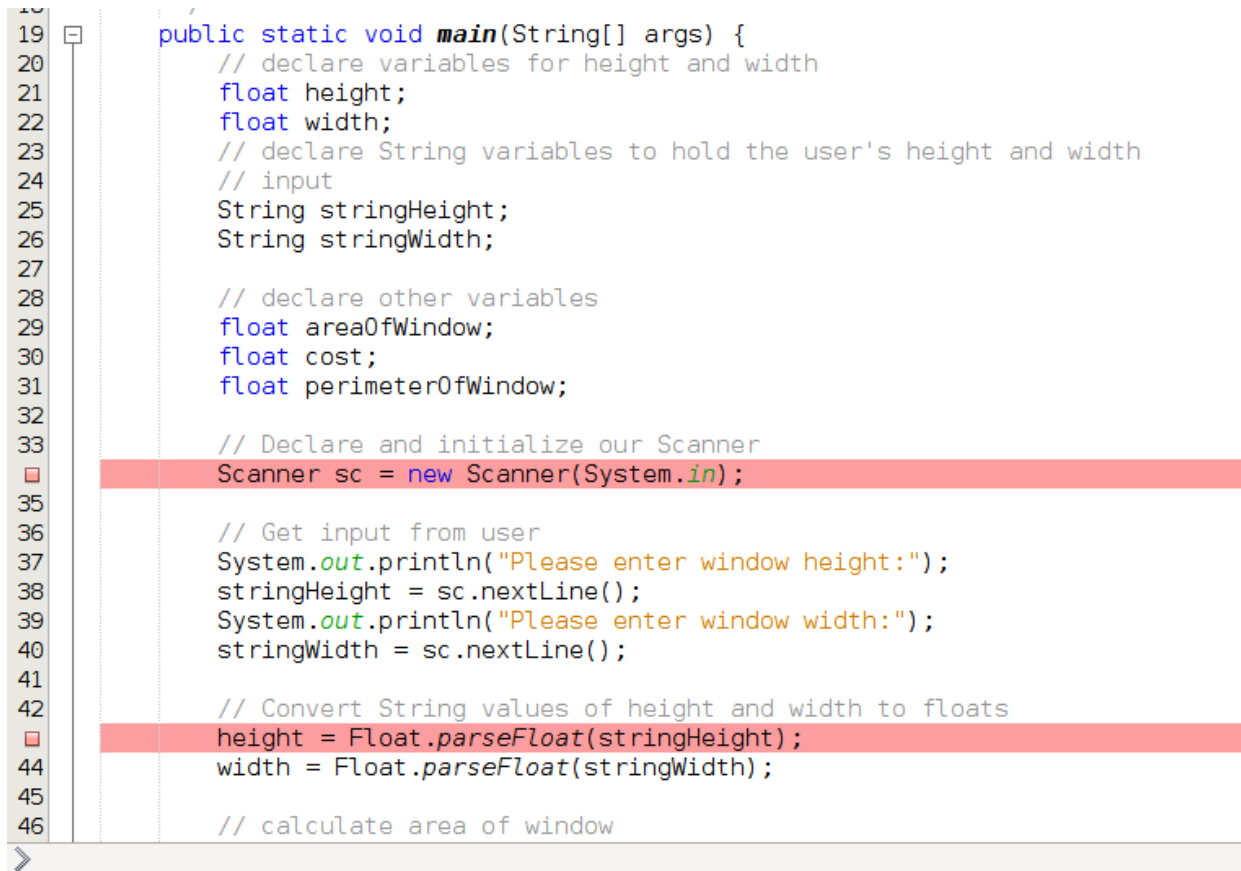


Figure 1: WindowMaster Breakpoints

Stepping Through Code

Once breakpoints are set, we are ready to start debugging. To do this, you must execute your program in **Debug Mode**, which is done by clicking on the “Debug Project” button:

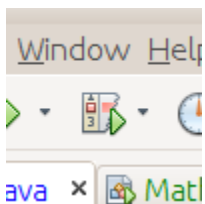


Figure 2: Debug Project Button

After pressing this button, the project will begin execution and then pause at your first breakpoint. Your first breakpoint will change from pink to green like this:

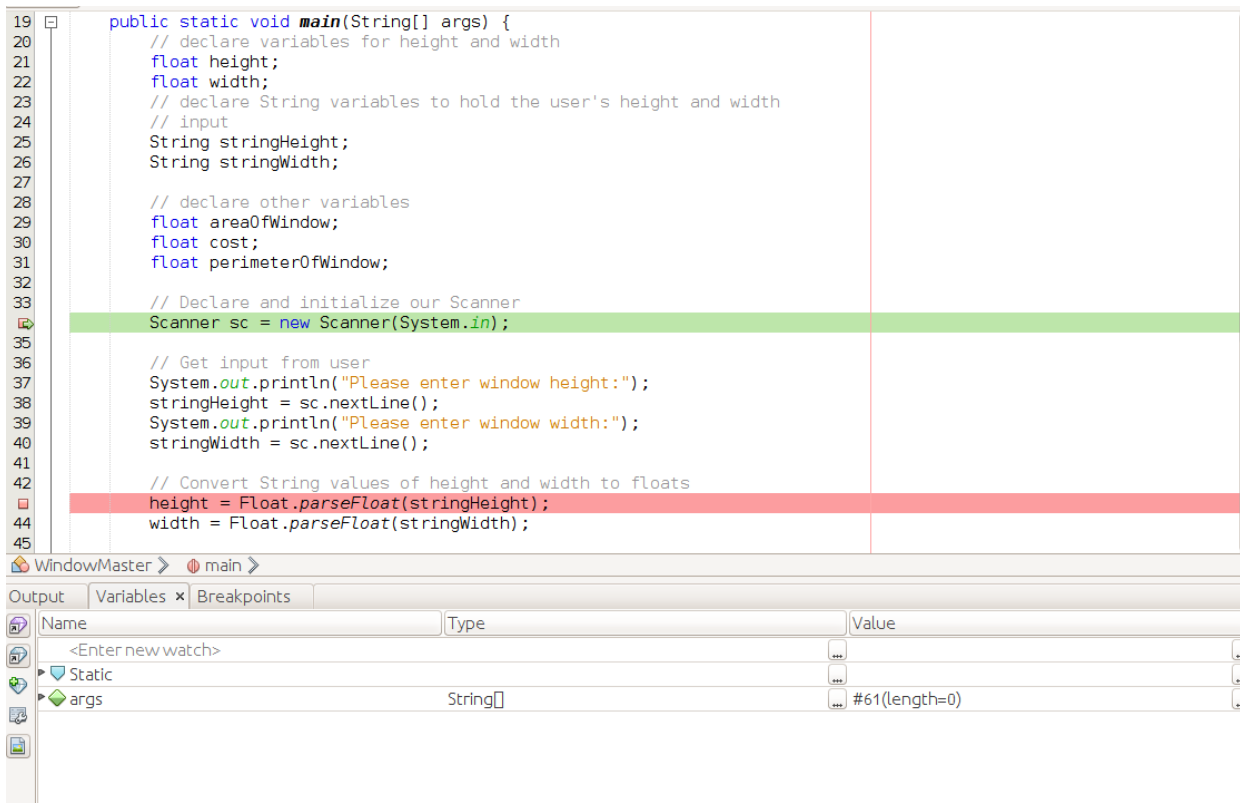


Figure 3: Execution Paused at First Breakpoint


Note that, even though we have declared several variables, none of them are yet visible in the Variables window. They become visible as values are assigned to them.

Now, let's step through a statement. Press F8 (Step Over) — this will allow the code on line 34 to execute and then execution will pause again on the next executable statement (line 37). The breakpoint on line 34 will turn pink again and line 37 will turn green (indicating that this is where execution has halted), and you will now see the Scanner variable **sc** in the Variables window:



Figure 4: Execution Paused at Line 37

Let's continue program execution by pressing F5 — this will allow the program to execute until the breakpoint on line 43. You will notice that nothing seems to happen after you press F5. This is because your program is waiting for user input (see line 38). Provide input by clicking on the Output window tab (just beside the Variables window tab) and then typing the requested values. After you have typed in the second value, execution will be halted on line 43:



```
21 float height;
22 float width;
23 // declare String variables to hold the user's height and width
24 // input
25 String stringHeight;
26 String stringWidth;
27
28 // declare other variables
29 float areaOfWindow;
30 float cost;
31 float perimeterOfWindow;
32
33 // Declare and initialize our Scanner
34 Scanner sc = new Scanner(System.in);
35
36 // Get input from user
37 System.out.println("Please enter window height:");
38 stringHeight = sc.nextLine();
39 System.out.println("Please enter window width:");
40 stringWidth = sc.nextLine();
41
42 // Convert String values of height and width to floats
43 height = Float.parseFloat(stringHeight);
44 width = Float.parseFloat(stringWidth);
45
46 // calculate area of window
47 areaOfWindow = height * width;
```

WindowMaster > main >

Output x Variables Breakpoints

Java-Jan2014 - /home/apprentice/Documents/GitHub/Java-Jan2014 x WindowMaster (debug) x Debugger Console x

debug:
Please enter window height:
5
Please enter window width:
5

Figure 5: Execution Paused at Second Breakpoint (after user input is provided)

Examining Variables

Now we have some variables to look at — we provided input for height and width and the code on lines 38 and 40 assigned those values to **stringHeight** and **stringWidth**. To look at those variables, click on the Variables window tab and they will appear (along with **sc**):

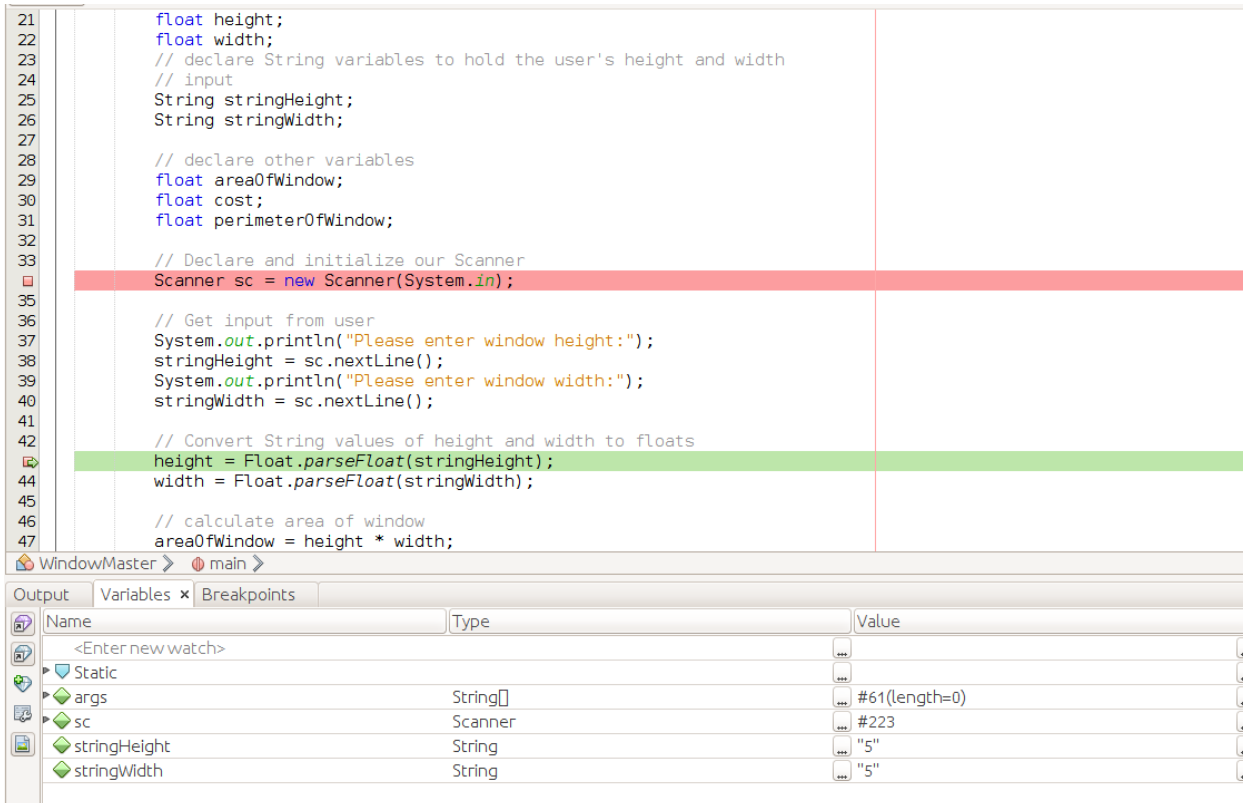


Figure 6: Examining Variables

You can now step through the remaining statements of the program using F8. As you execute each statement, notice how variables appear in the Variables window.

Wrap-up

That does it for debugging. In this lesson, we covered:

1. Executing a program in debug mode
2. Setting breakpoints
3. Stepping through a program statement by statement
4. Examining the values of program variables during program execution