

## **Supplement II.C: Learning Java Effectively with JBuilder**

**For Introduction to Java Programming**

**By Y. Daniel Liang**

### **0 Introduction**

Supplement II.B, "JBuilder Tutorial," gives a brief tutorial on how to use JBuilder. JBuilder is not only a powerful and productive Java program development tool, but it is also a valuable pedagogical tool for teaching and learning Java programming. This supplement shows how to use JBuilder effectively with the text.

The supplement is written for instructors, but it is also useful to students.

### **1 Important Tips**

The objective of the course is to teach Java, not JBuilder. JBuilder is a complex and powerful tool. All you need for this course, however, is a small and simple set of features that enable students to create, compile, run, and debug programs. So students should avoid exploring unnecessary features.

If your students follow the instructions in Supplement II.B, "JBuilder Tutorial," or the instructions from you, students can master all essential skills in sixty minutes. It is important that your students adhere to the instructions to avoid frustrating mistakes. If a mistake is made, simply read the instructions and restart from scratch.

### **2 JBuilder as a Valuable Pedagogical Tool**

The following sections demonstrate how to utilize JBuilder in the first seven chapters.

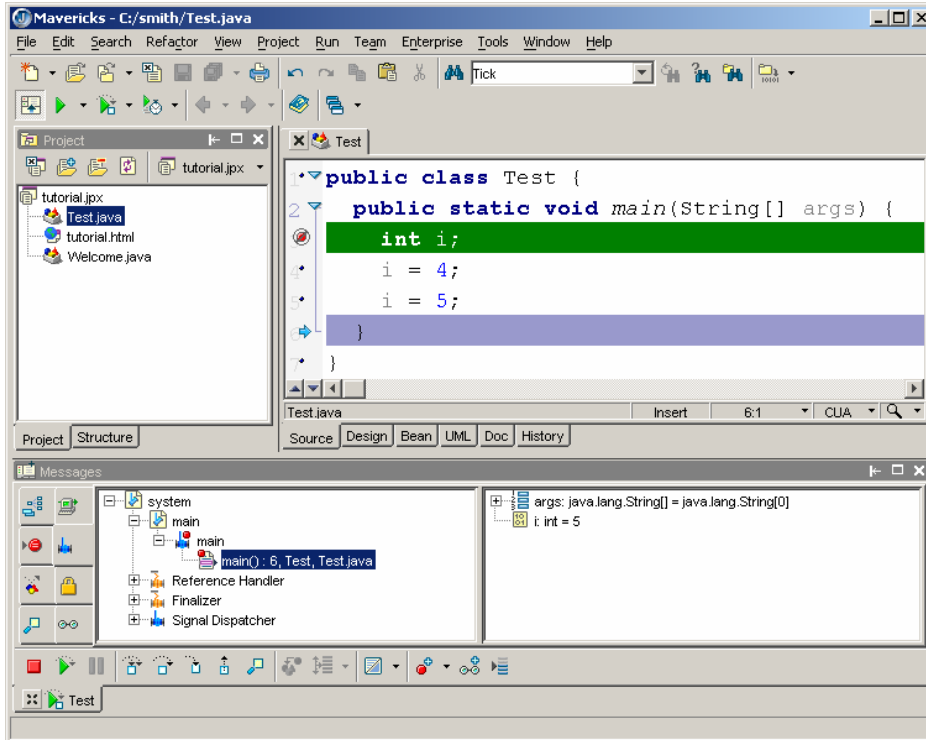
#### *2.1 Using JBuilder in Chapter 1*

After Listing 1.1, you can start to cover how to create, compile, and run a program in JBuilder. You may also introduce how to use JBuilder online help.

#### *2.2 Using JBuilder in Chapter 2*

You may start to introduce debugging when you cover variables. You can use debug to show the value of a variable in the memory and show the change of the value during execution. Figure 1 shows a simple test program with

variable *i*.

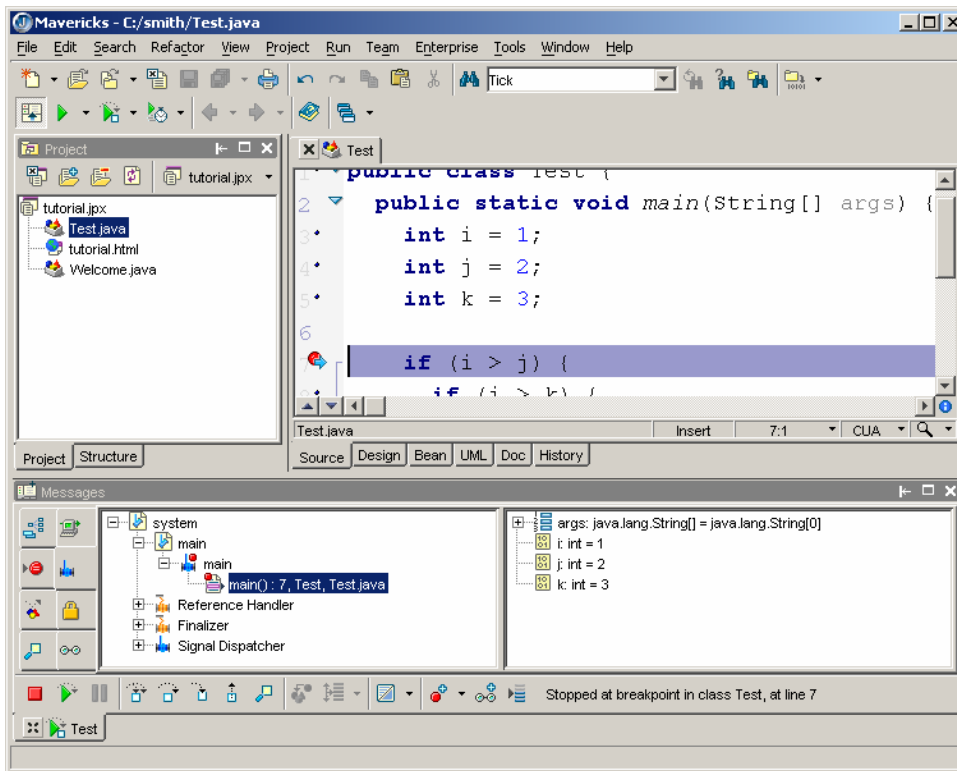


**Figure 1**

*Displaying values of variables in JBuilder debugger.*

### 2.3 Using JBuilder in Chapter 3

Use the debugger to trace the *if* statements in Section 3.3.3, “Nested *if* Statements,” in the text, as shown in Figure 2.

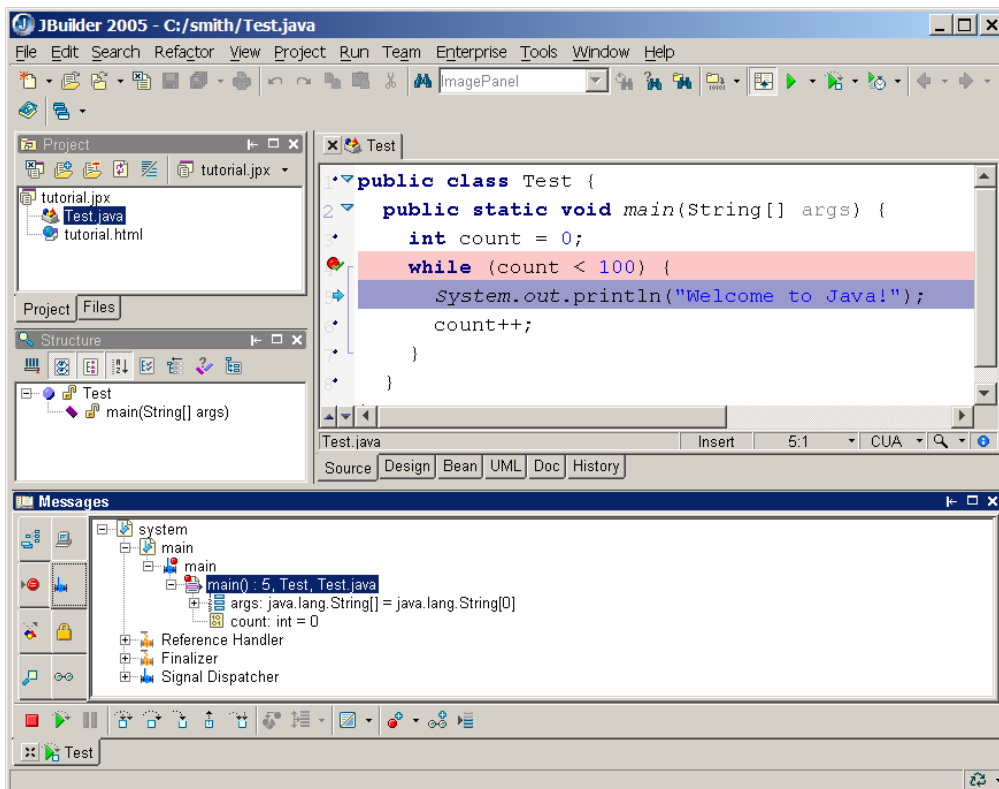


**Figure 2**

*Trace the execution of an if statement.*

#### 2.4 Using JBuilder in Chapter 4

Use the debugger to trace the while loop for the code on page 96, as shown in Figure 3.



**Figure 3**

*Trace the execution of a loop statement.*

### 2.5 Using JBuilder in Chapter 5

You can use the debugger to show the call stack, which is very effective to help understand method invocation. Let us use Listing 5.1 to demonstrate method invocation. Set a breakpoint at Line 6. Start debugger, and the debugger pauses at Line 6. Choose Step into to step into the max method, as shown in Figure 4. Now in the Message pane, you will see the arguments are passed to the method.

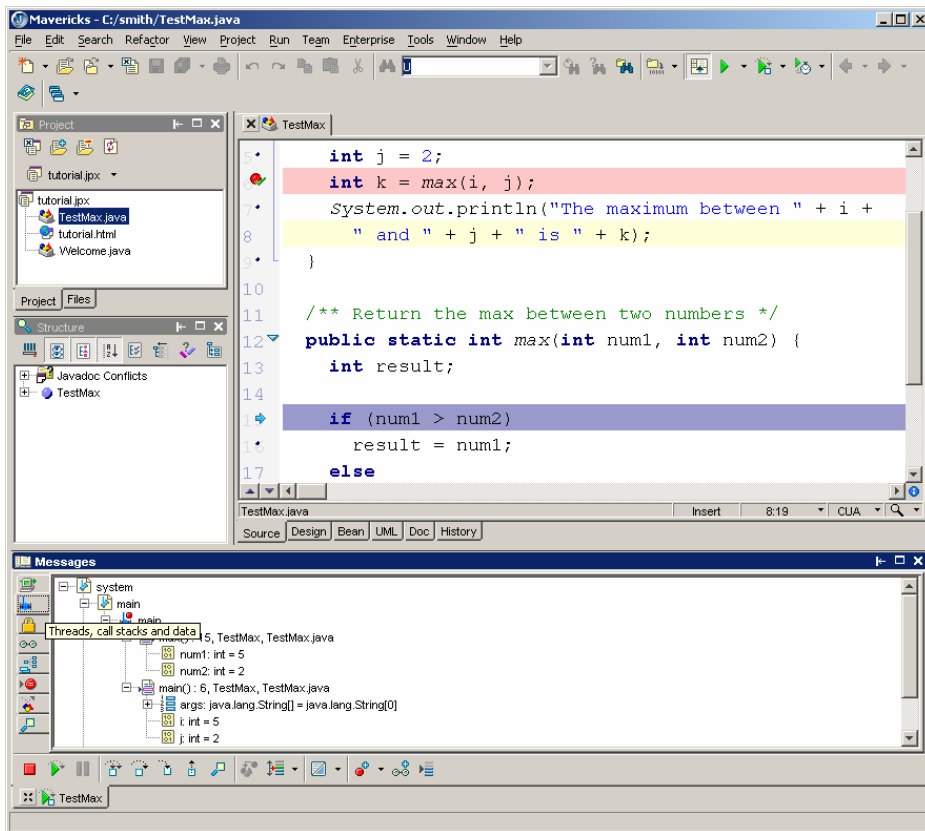


Figure 4

*Trace method invocation.*

A very useful feature is to get online documentation for Java API classes. For example, to get the documentation for the `Math` class, highlight `Math` in the source code editor and choose `Find Definition` from its context menu, as shown in Figure 5. The source code of the class is shown in Figure 6. To see the documentation in HTML, choose the `Doc` tab, as shown in Figure 7.

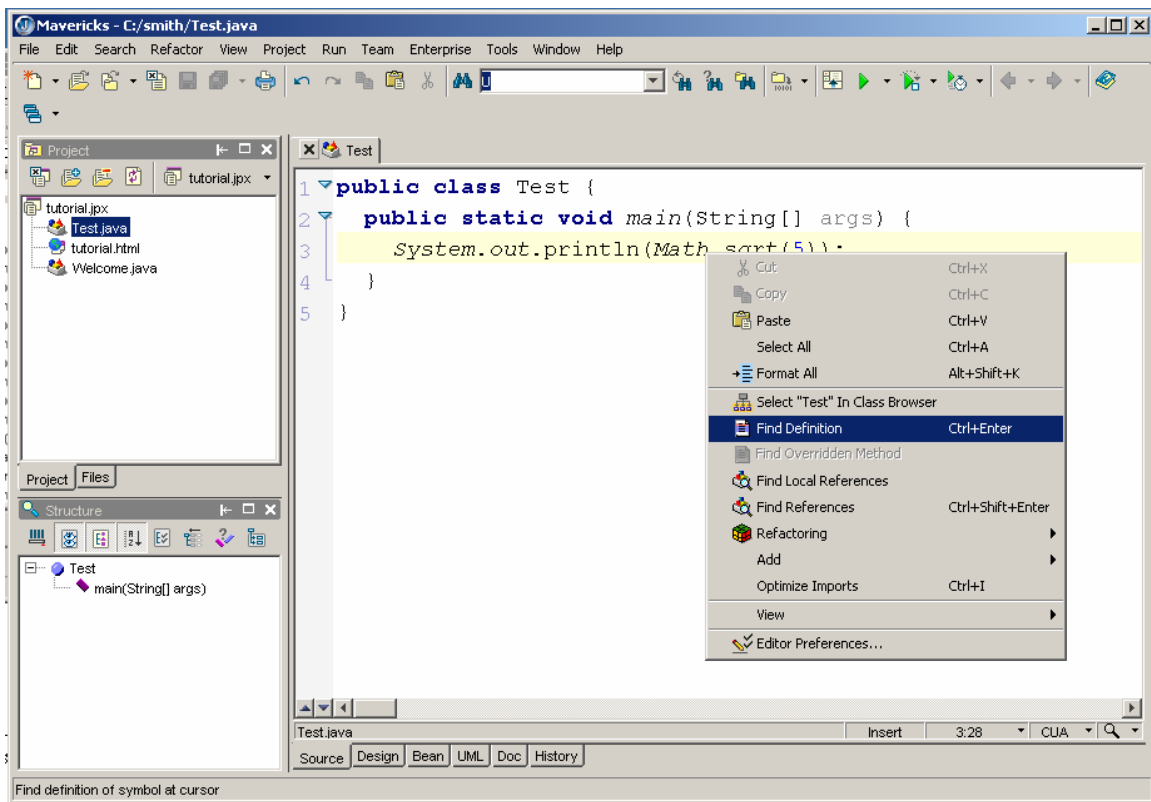
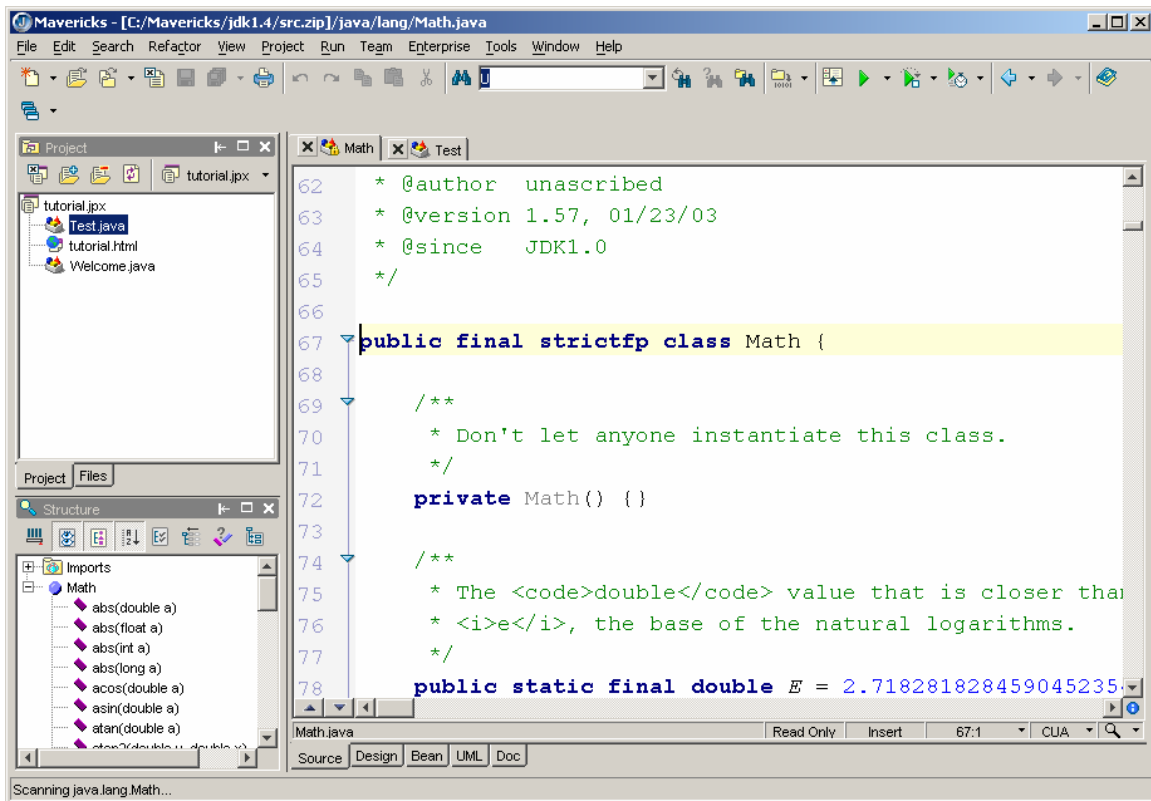


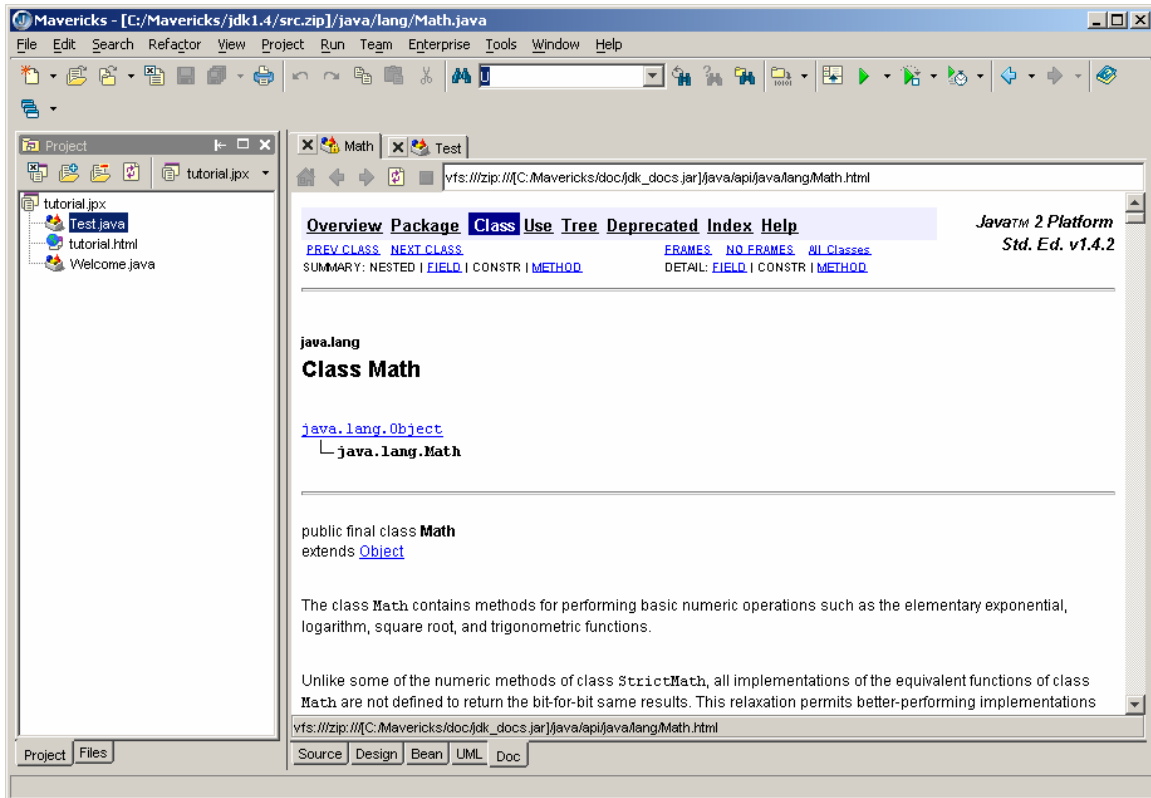
Figure 5

*You can get documentation for a class in the Java API.*



**Figure 6**

*You can view the source code of a class in the Java API.*

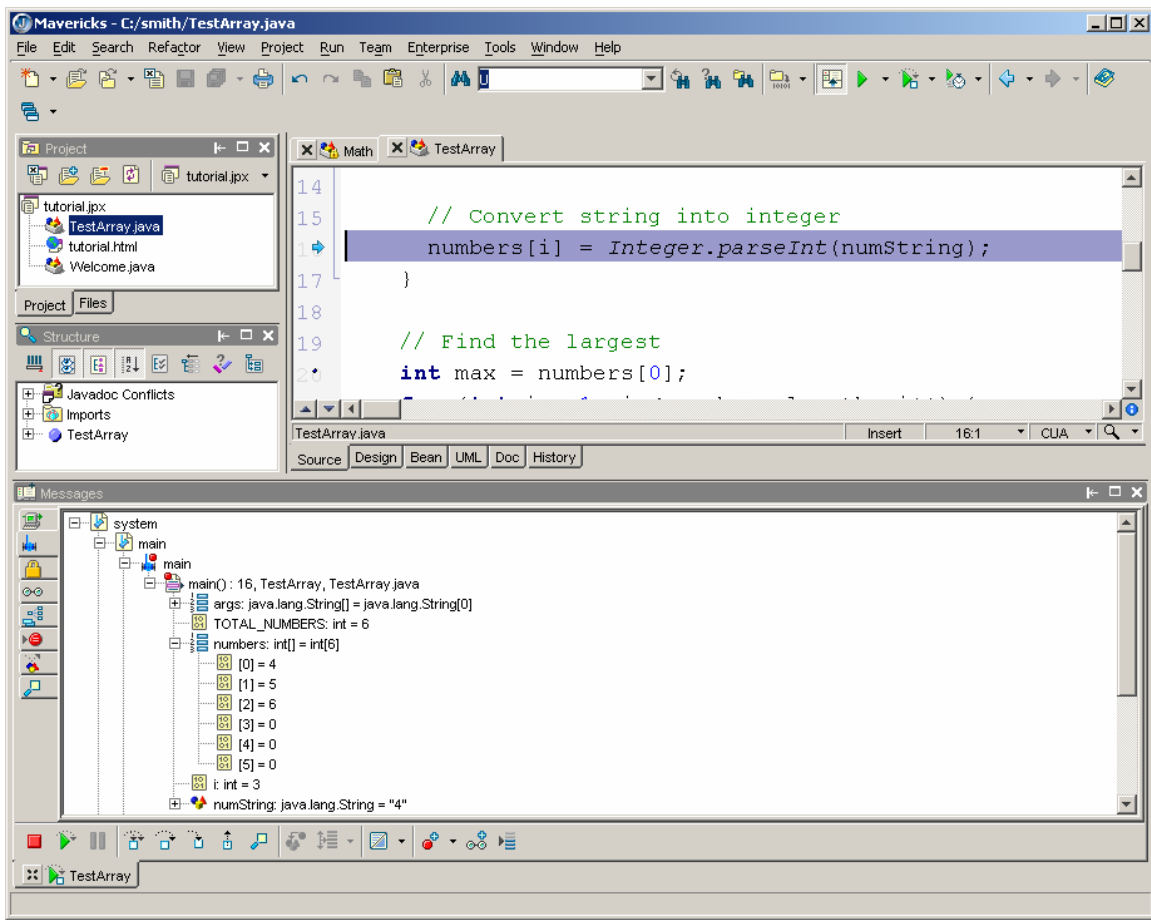


**Figure 7**

*Choose the Doc tab to display the class documentation in HTML.*

## 2.6 Using JBuilder in Chapter 6

You can use the debugger to show the values of all the elements in an array. Figure 8 shows debugging `TestArray.java` in Listing 6.1.



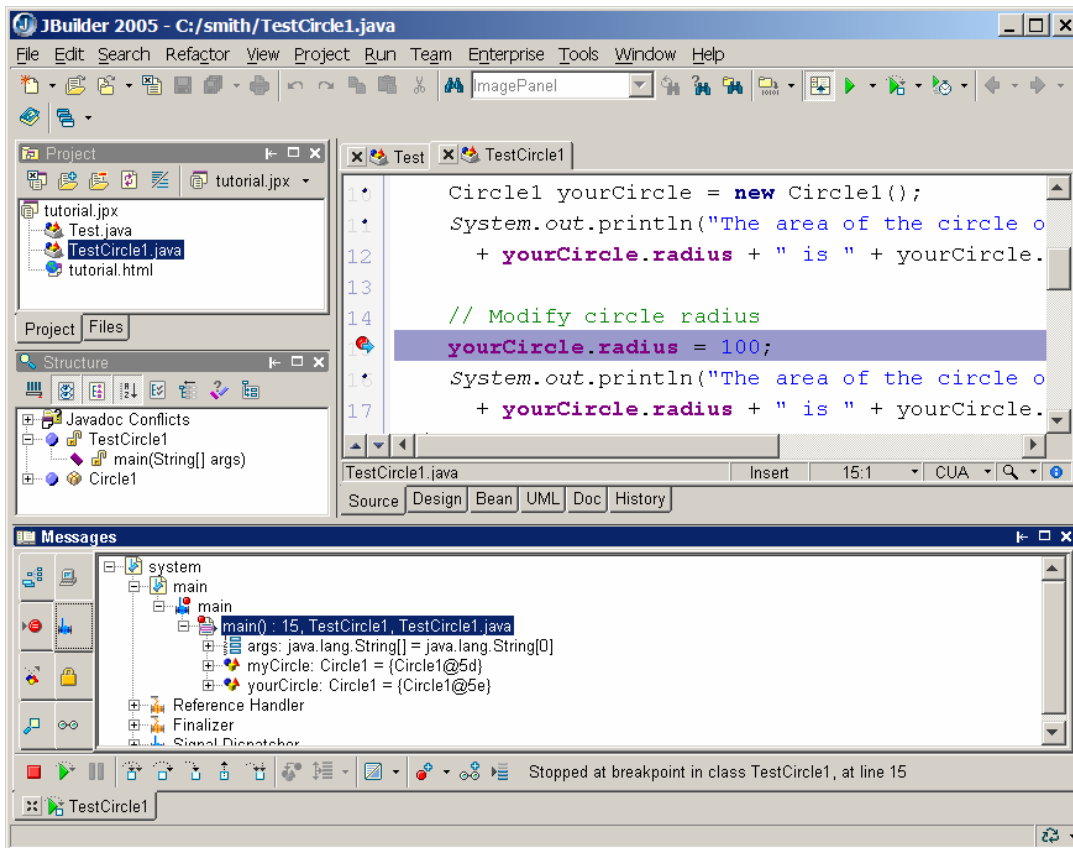
**Figure 8**

*You can see the change of values in an array.*

You can use the debugger to demonstrate how arguments are passed and to see the differences between passing primitive type values and arrays.

## 2.7 Using JBuilder in Chapter 7

You can use the debugger to show the contents of an object. Figure 9 shows debugging TestCircle1.java in Listing 7.1.



**Figure 9**

*You can see the change of values in an object.*

You can use the debugger to demonstrate how arguments are passed and to see the differences between passing primitive type values and objects.