2 Getting St□rted

Java will be used in the examples in this section; however, the information applies to all supported languages for which you have installed a compiler (e.g., Ada, C, C++, Java) unless noted otherwise. In any of the language specific steps below, simply select the appropriate language and code. For example, in the "Creating a New File" below you may select C++ as the language and then enter a C++ example or select Java to enter a Java example.

If you have installed jGRASP on your own PC, you should see the jGRASP icon in the Windows desktop.

jGRASP

You can start jGRASP by double clicking on the icon. If you are working on a PC in a computer lab, you may not see the jGRASP icon on the desktop. Try the following:

Click Start -- All Programs -- UNC Courseware -- COMP 14 -- jGRASP

Depending on the speed of your computer, jGRASP may take about 30 seconds to come up. The jGRASP virtual **Desktop**, shown below, is composed of a Control Panel with a menu across the top plus three panes: (1) left pane with tabs for **Browse**, **Project**, **Find**, and **Debug**, (2) right pane for CSD Windows, and (3) lower pane with tabs for jGRASP messages, Compile messages, and input/output for Run.

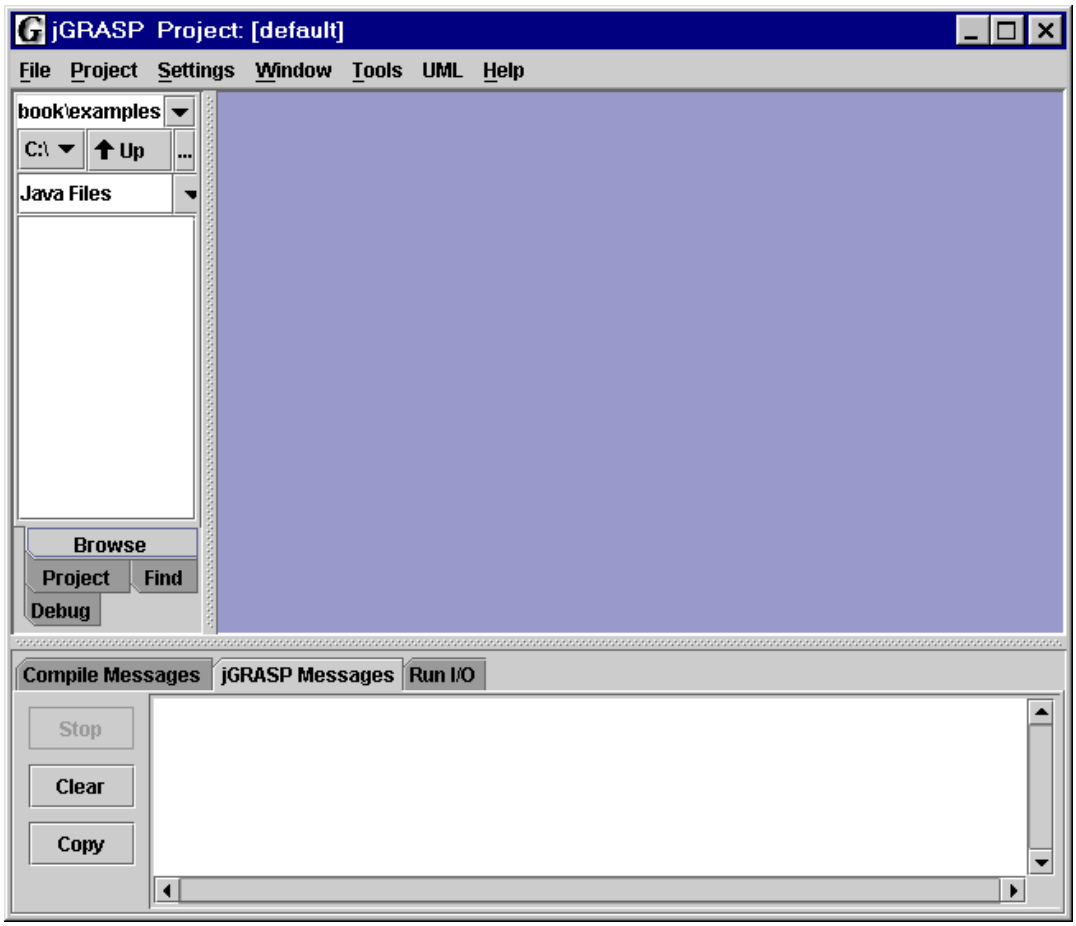

Figure 8. The jGRASP Virtu□l Desktop

2.1 Cre□ting □ New File

To open an empty CSD Window for Java within the Desktop, click on **File -- New File -- J□v□**. Note that the list of languages displayed by **File – New File** will vary with your use of jGRASP. If the language you want is not listed, click **Other** to see all available languages. The languages for the last 25 files opened will be displayed in the list; the remaining available languages will be under **Other**.

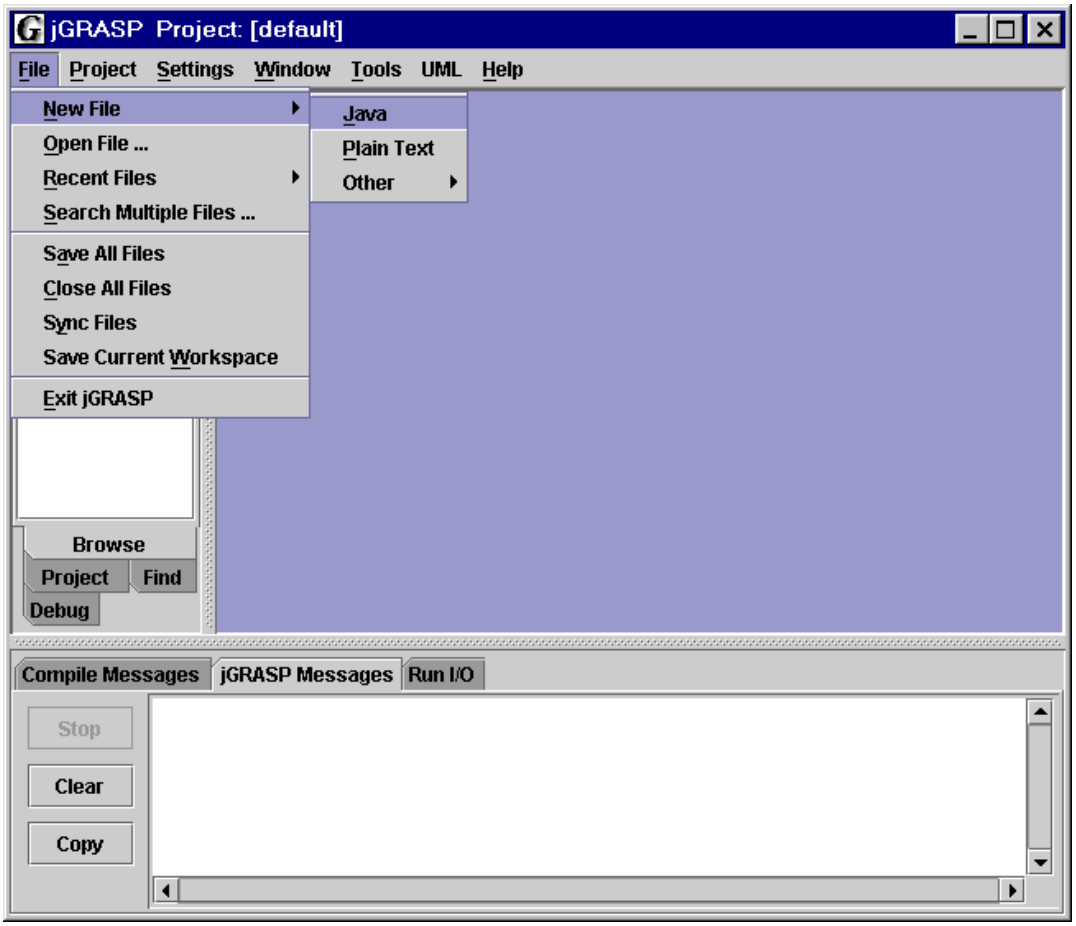

Figure 9. Opening □CSD Window for J□v□

After you click on **File -- New File -- J□v□** (Figure 9 above), a CSD Window is opened in the right pane of the Desktop as shown in Figure 10 below. Notice the title for the frame, jGRASP CSD (Java), indicates the CSD Window is Java specific.

If Java is not the language you intend to use, you should close the window, then open a CSD Window for the correct language.

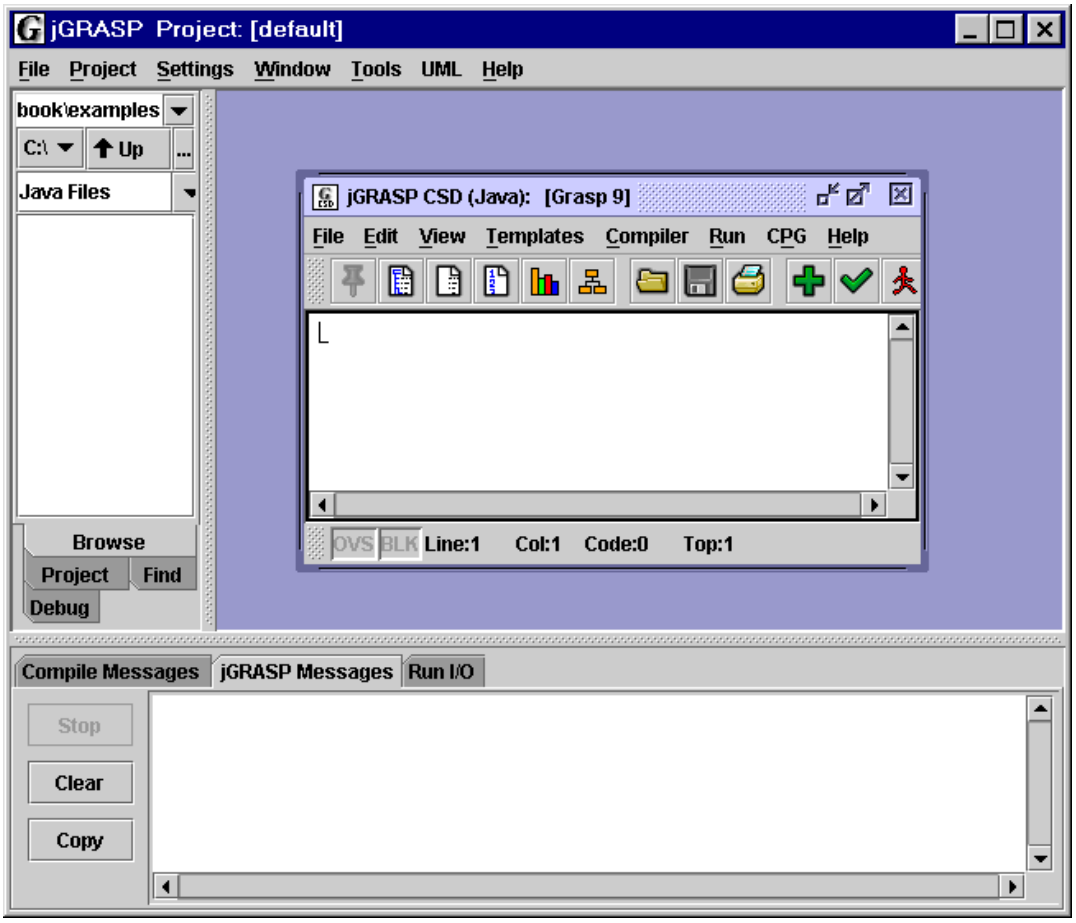

Figure 10. Empty CSD Window in Desktop

In the upper right corner of the CSD Window are three buttons that control its display:

The first button iconifies the CSD Window. The second either maximizes the CSD Window relative to the jGRASP Desktop, or if it is already maximized, the button restores the CSD Window to its previous size. The third button closes the CSD Window.

You may also make the Desktop full screen by clicking the appropriate icon in the upper corner of it. Notice the CSD Window has its own menu and toolbar with icons across the top.

Figure 11 shows the CSD Window maximized within the virtual Desktop.

HINT: If you want all of your CSD Windows to be maximized automatically when you open them, then under **Settings** on the Desktop menu, click on (indicated by a check mark) the option called **Open CSD Windows M□ximized**.

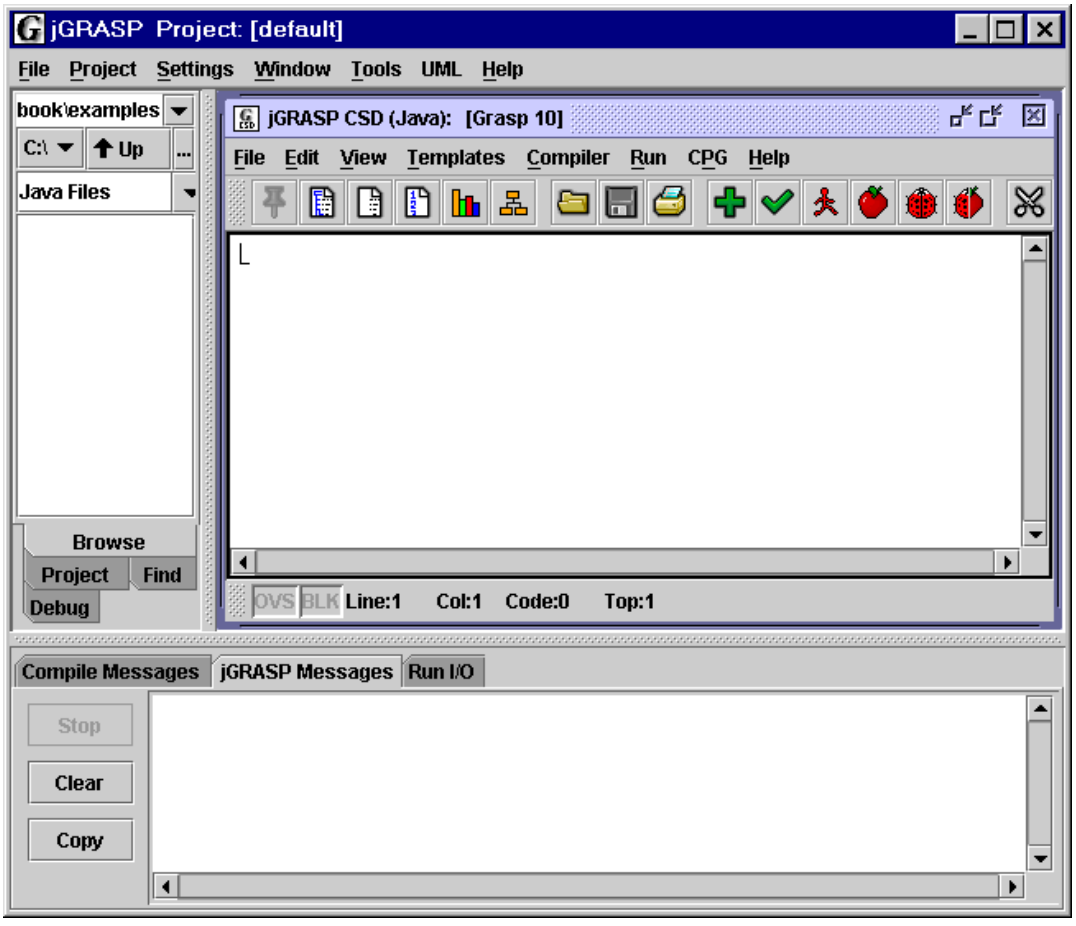

Figure 11. CSD Window exp□nded in Desktop

Type in the following Java program in the CSD Window, exactly as it appears. Remember, Java is case sensitive.

```
public class Hello
{
   public static void main(String[] args)
   {
      System.out.println ("Hello, world!\n");
   }
}
```

After you have entered the program, your CSD Window should look similar to the program shown in Figure 12. Notice in the source code that coloring is used to distinguish among comments, keywords in the language, strings, etc. Later, you will learn how set these colors to those of your on choice.

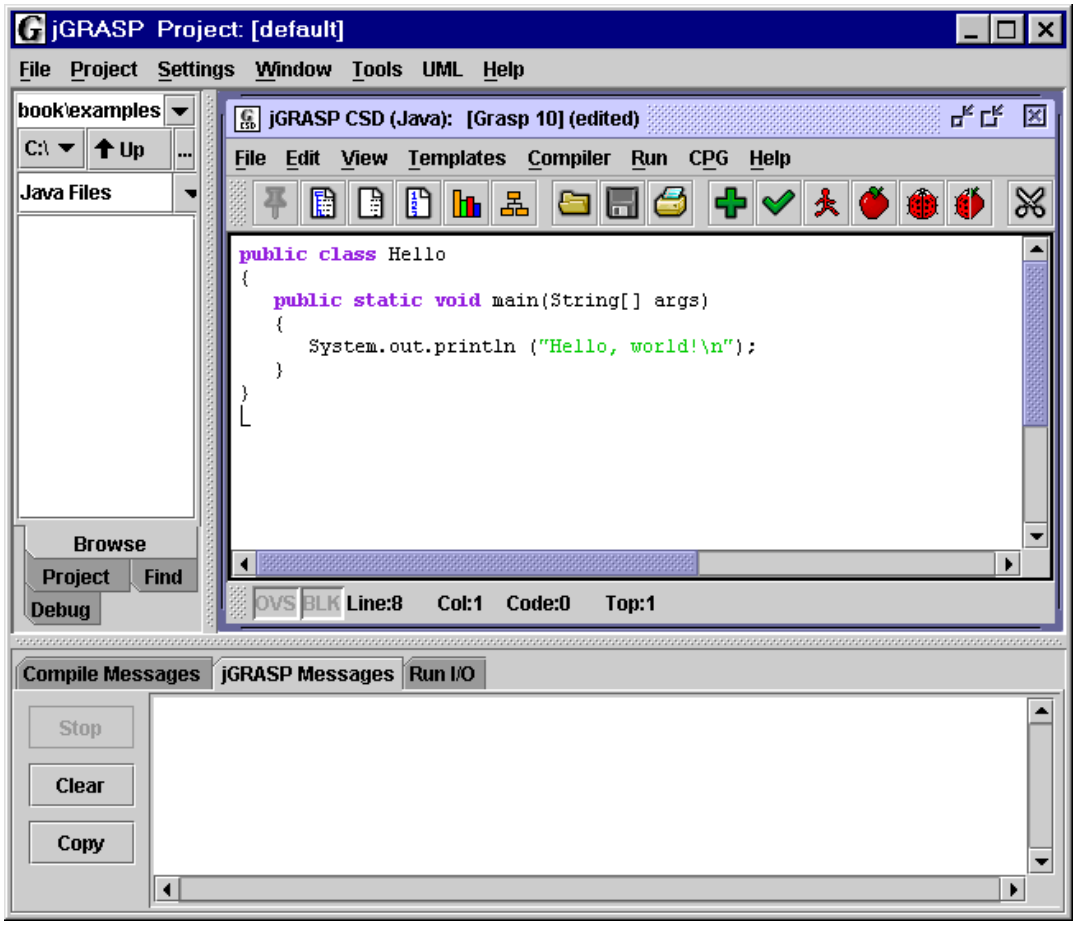

Figure 12. CSD Window with progr□m entered

2.2 S□ving □File

Save the program as "Hello.java" by clicking the Save icon on the tool bar of the CSD Window, or you can click **File** -- **S□ve** on the CSD Window menu (not the Desk Top menu). Note, in Java, the file name must match the class name (i.e., class Hello must be saved as Hello.java).

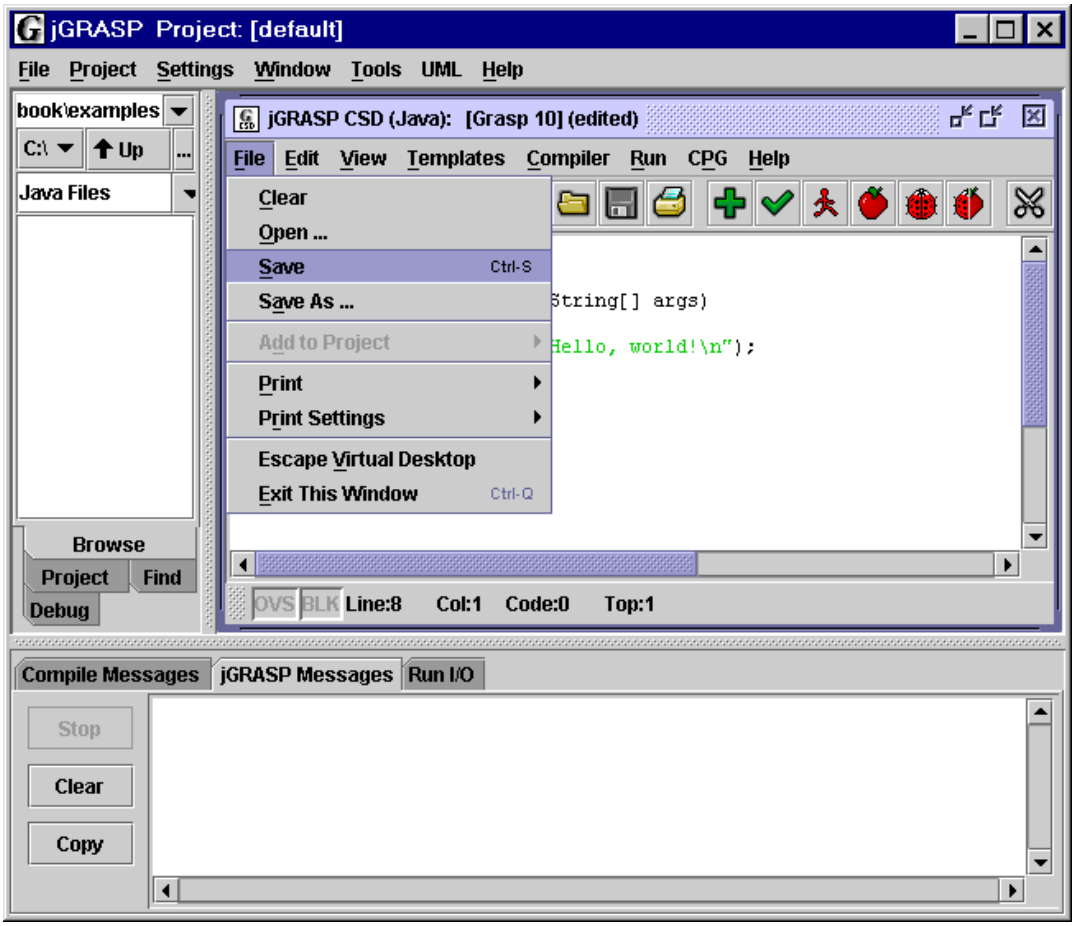

Figure 6. S□ving □file from the CSD Window

After you click on Save, the Save dialog box come up as illustrated in Figure 14.

After the program has been saved, it will be listed in the browse pane. If the program is not listed in browse pane, be sure the browse pane is set to the directory where the file was saved.

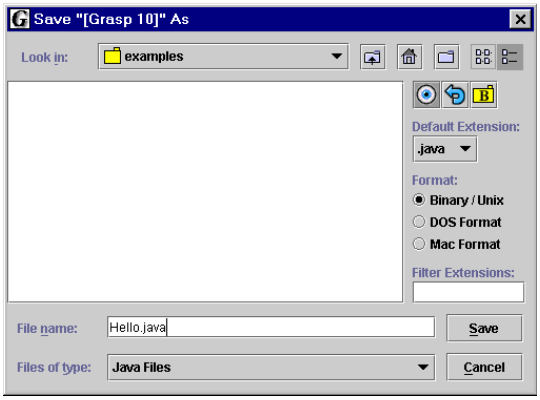

Figure 14. S□ve di□log to n□me file

2.3 Closing File

Now that the file has been saved, you can close the file the file by clicking the Close button in the upper right corner of the CSD Window.

After the file is closed, your Desktop should look like the figure below. Notice that Hello.java is listed in the Browse pane on the left.

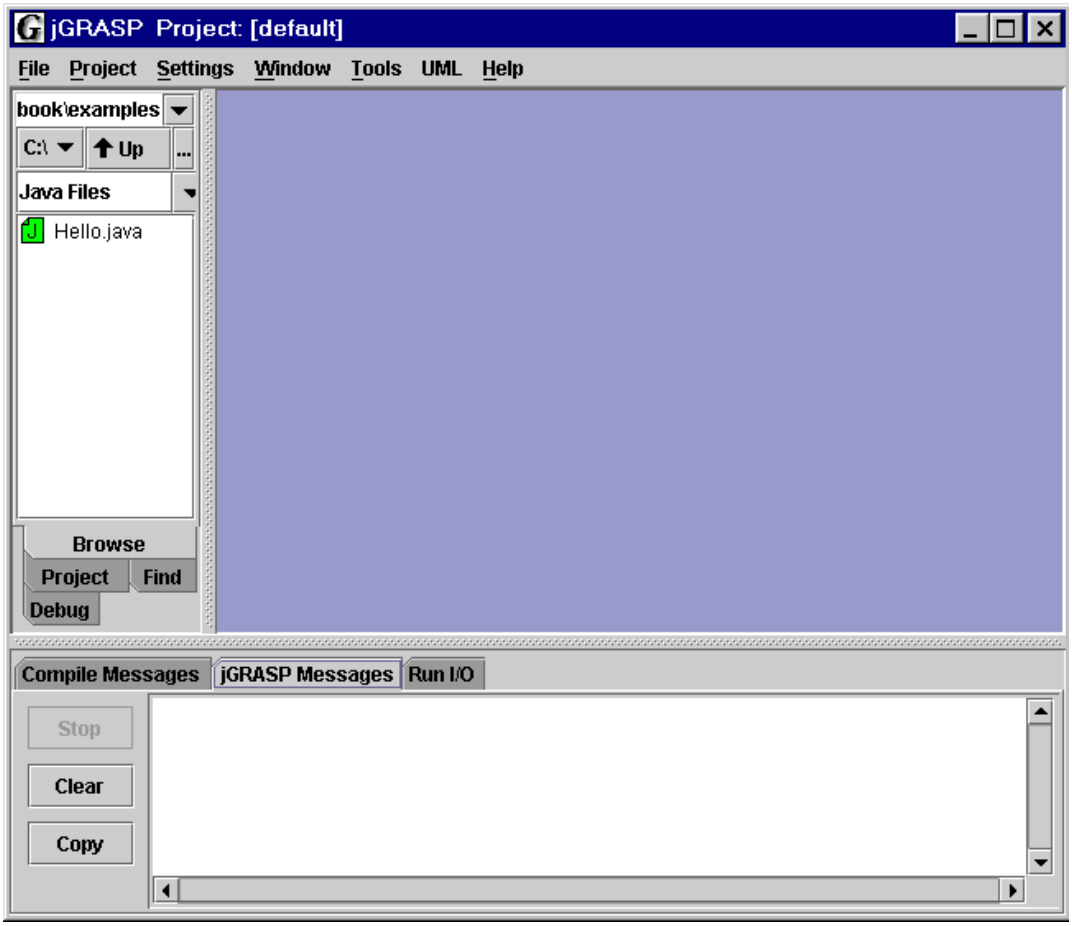

Figure 15. Desktop with CSD Windows closed

2.4 Lo□ding □File

A file can be loaded into a CSD Window in five distinct ways. Each of these is described below.

1) If the file is listed in jGRASP Browse pane (as in Figure 15), you can simply double click on the file name, and the file will be opened in a new CSD Window.

2) On the Desktop menu, click **File – Open** as illustrated in Figure 16. This will bring up the Open File dialog.

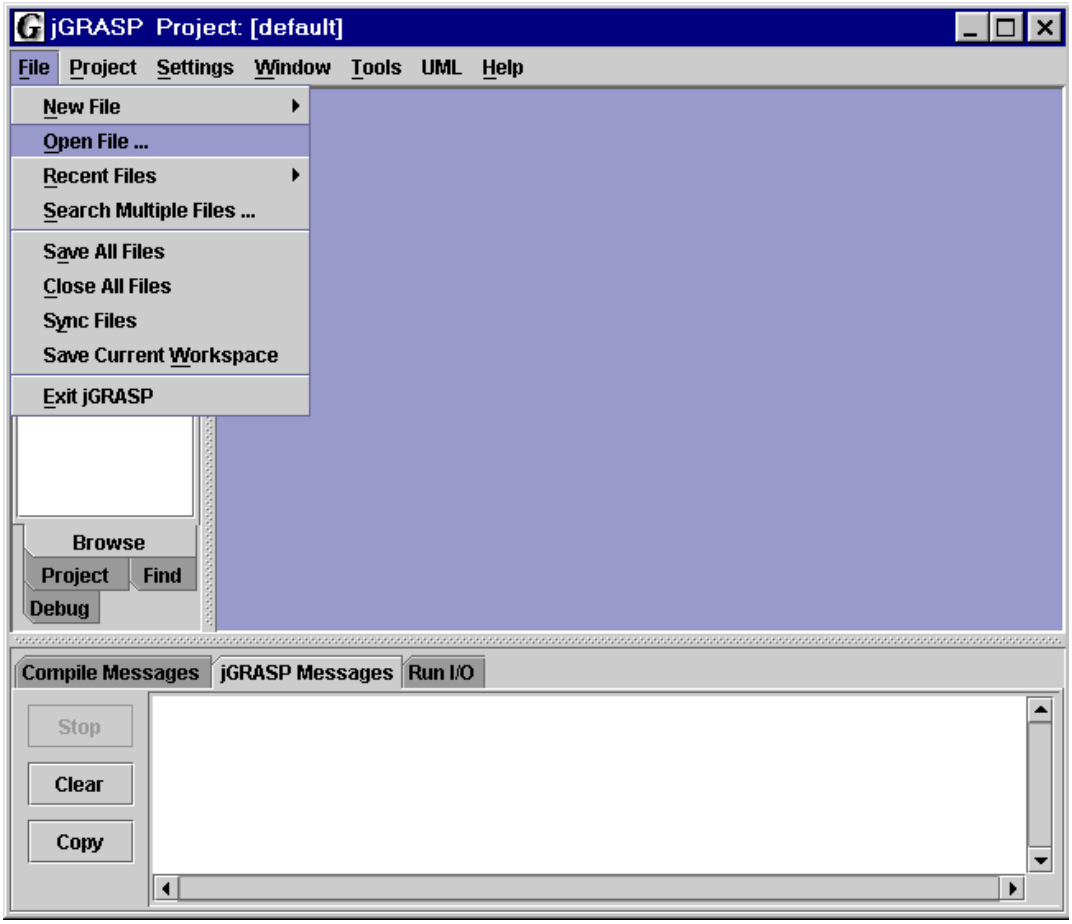

Figure 16. Opening □file from the Desktop

3) If you have a CSD Window open, click **File – Open** as shown in Figure 17. This will open the Open File dialog box, which will allow you browse up and down directories until you locate the file you want to open.

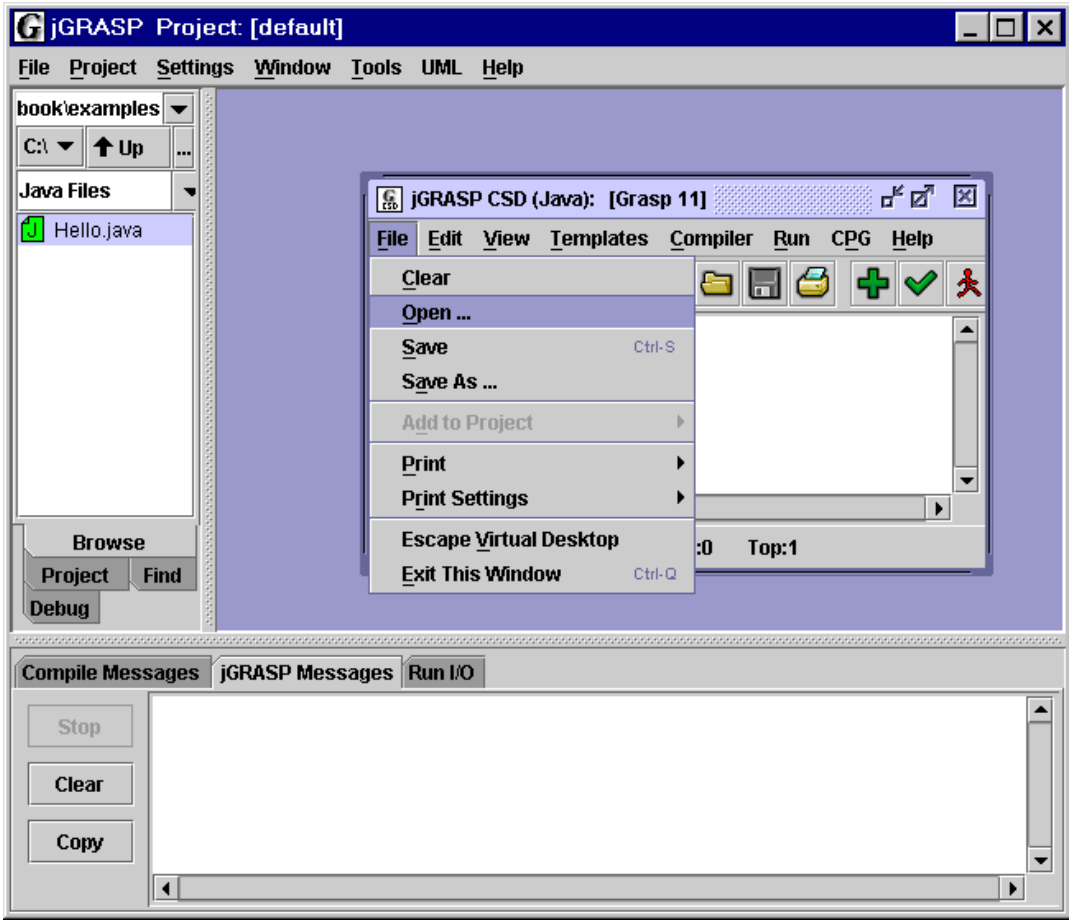

Figure 17. Opening file from the CSD Window

4) If you have a Windows file browser open (e.g., Windows Explorer, My Computer, or My Documents), and the file is marked as a jGRASP file, you can just double click the file name.

5) If you have a Windows file browser open (e.g., Windows Explorer or My Computer), you can drag-and-drop a file to the jGRASP Desktop canvas where the CSD Window will be displayed. However, files usually open more quickly by double-clicking (option 4 above) rather than using the drag-and-drop option.

In all cases above, if a file is already open in jGRASP, its CSD Window containing it will be popped to the top of the Desktop rather than jGRASP opening a second CSD Window with the same file.

Multiple CSD Windows

You can have multiple CSD Windows open, each with a separate file. Each program can be compiled and run from its respective CSD Window. In Figure 18, two CSD Windows have been opened. One contains Hello.java and the other contains Hello2.java. If the window you want to work in is visible, simply click the mouse on it to bring it to the top. Otherwise, click **Window** on the upper tool bar, and a drop down menu will list all of the open files (also shown in Figure 18). Clicking on the file name (e.g., Hello.java) will bring its CSD Window to the top.

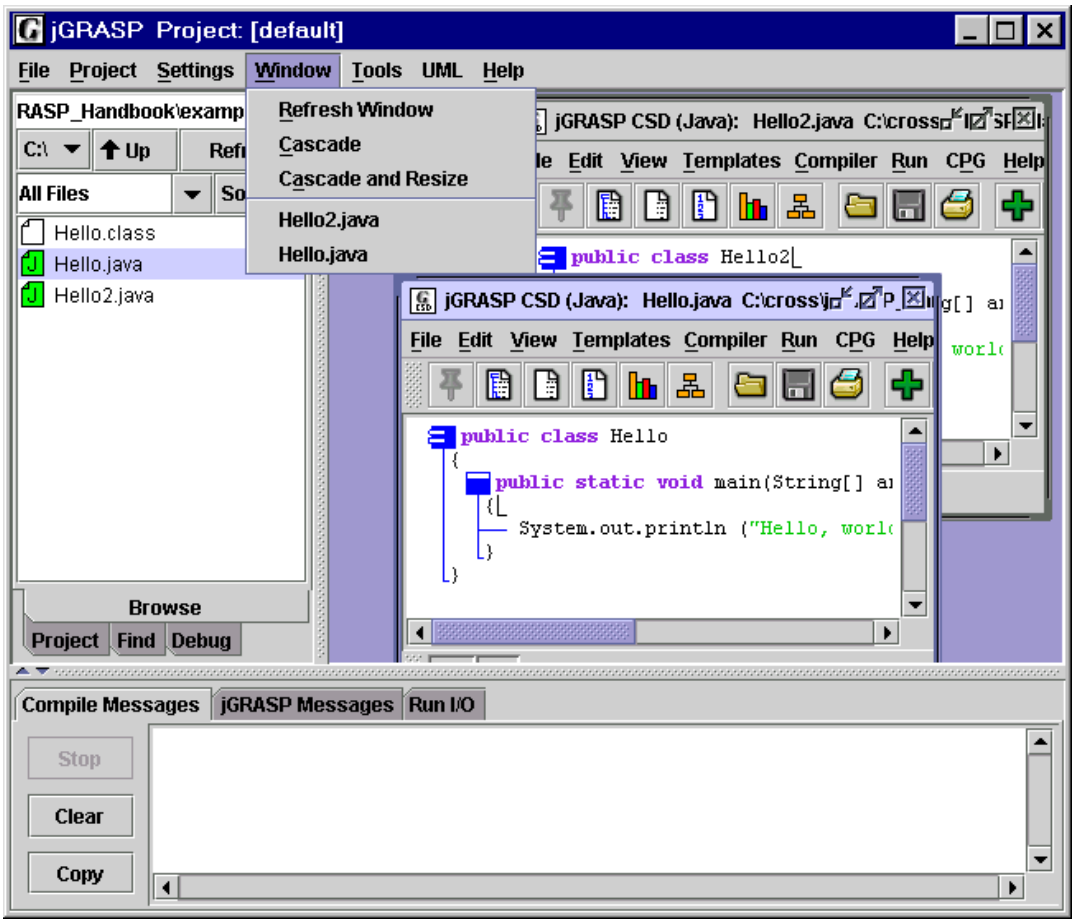

Figure 18. Multiple files open

2.5 Generating a Control Structure Diagram

Anytime you have a syntactically correct program or skeleton of a program in the CSD Window, you can generate a Control Structure Diagram (CSD). Generate the Control Structure Diagram (CSD) for the program by doing one of the following:

1) Clicking the Generate CSD icon  or
2) Clicking **View -- Generate CSD** on the menu or
3) Pressing F2

If your program is syntactically correct, the CSD will be generated as shown in Figure 19. After you are able to successfully generate a CSD, go on to the next section below.

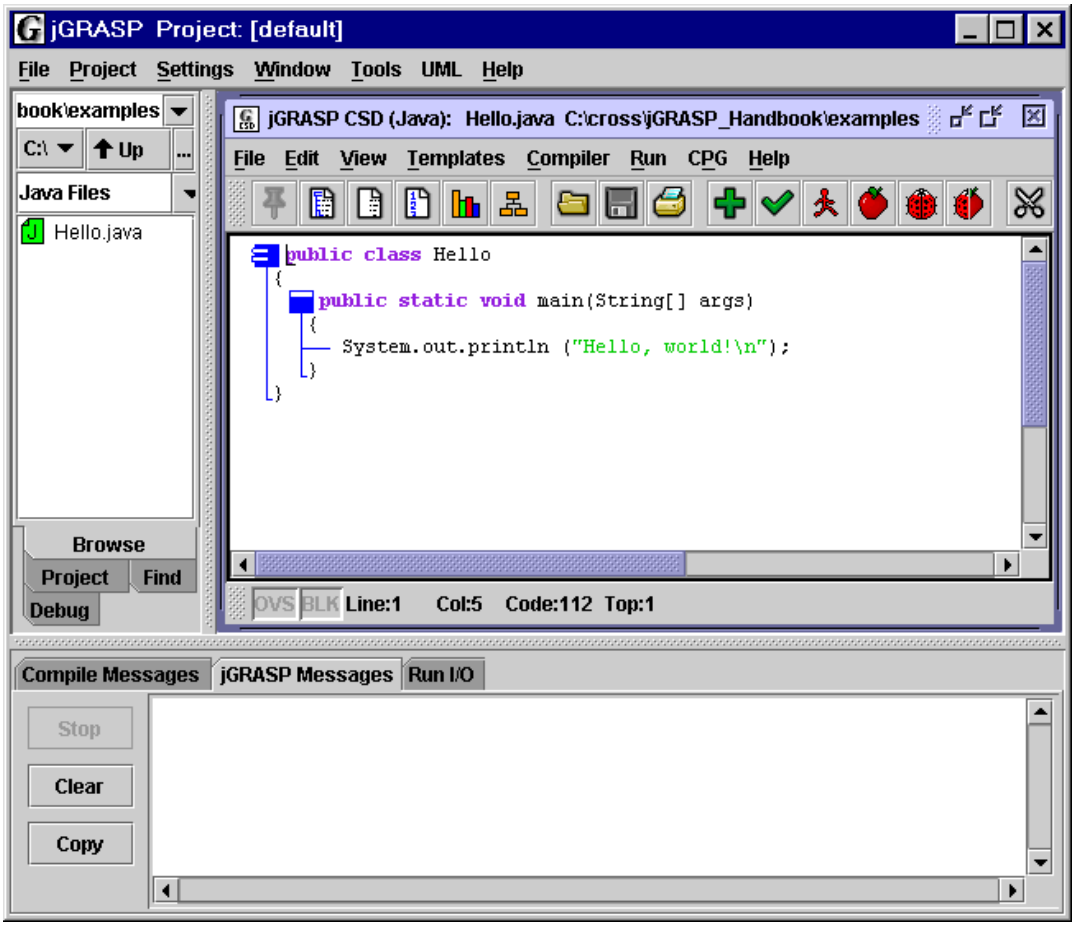

Figure 19. After CSD is generated

Otherwise, if you have a syntax error in your program that was detected during the Generate CSD, jGRASP will highlight the vicinity of the error and describe it in the message window.

If you do not find an error in the highlighted line, be sure to look for the error in the line just above it. In the example in Figure 20, the semi-colon was omitted at the end of the println statement. As you gain experience, these errors will become easier to spot.

If you are unable find and correct the error, you should try compiling the program, since the compiler usually provides a more detailed error message (see Compiling a Program below).

You can remove the CSD by doing one of the following:

1) Clicking the Remove CSD icon  or
2) Clicking **View -- Remove CSD** on the menu or
3) Pressing Shift-F2

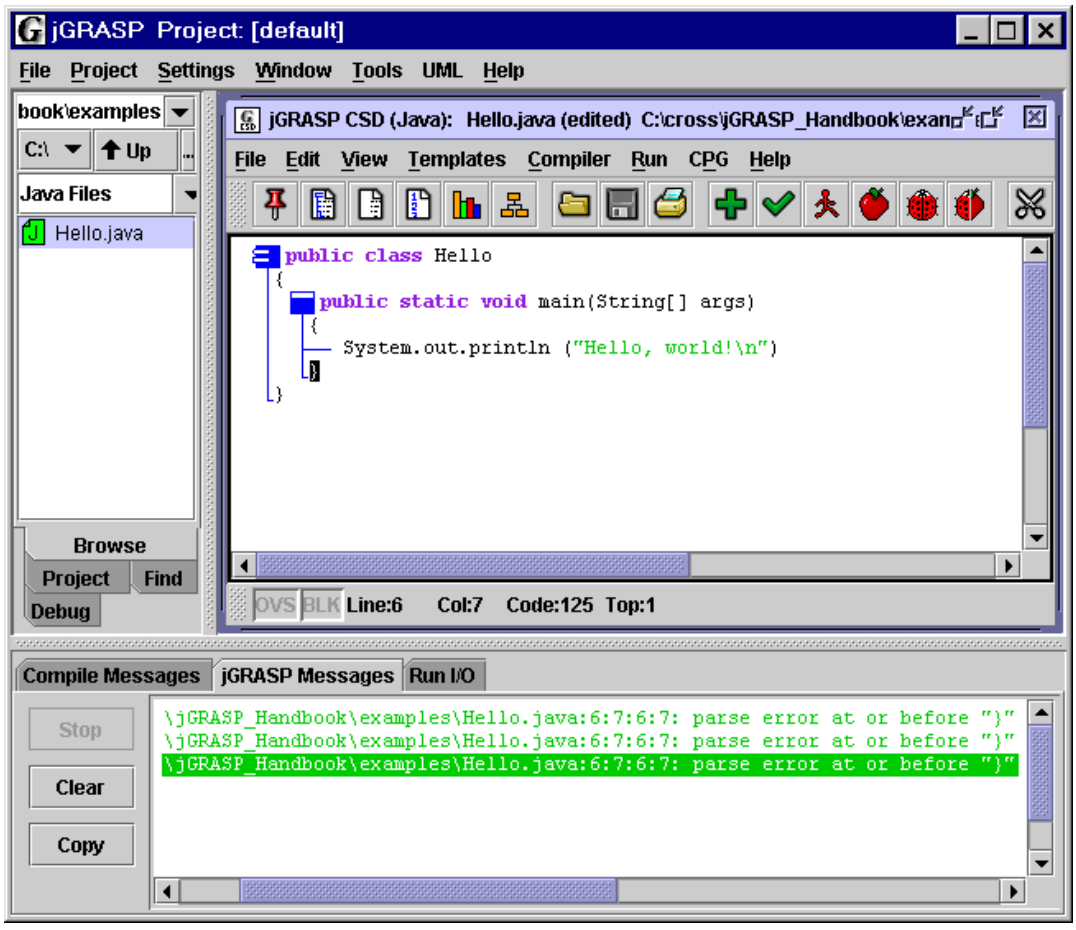

Figure 20. Syntax error detected

Remember, the purpose of using the CSD is to improve the readability of your program. While this is may not be obvious on a small simple program like the example, it should become apparent as the size and complexity of your programs increase.

TIP: As you enter a program, try to enter it in syntactically correct “chucks.” For example, the following is sufficient to generate the CSD.

```
public class Hello

{
}
```

As soon as you think you have entered a syntactically correct chuck, you should generate the CSD. Not only does this update the diagram, it catches your syntax errors early.

2.6 Folding □ CSD

“**Folding**” is another feature that many users find useful, especially as programs get larger. After you have generated the CSD, you can fold your program based on its structure.

For example, if you double-click on the class icon the entire program is folded (Figure 21). If you double-click on the “plus” icon, the first layer of the program is unfolded. You can continue to unfold the program layer by layer as needed.

Although the example program has no loops or conditional statements, these may be folded by double-clicking the corresponding CSD control constructs. For other folding options, seethe **View – Fold** menu.

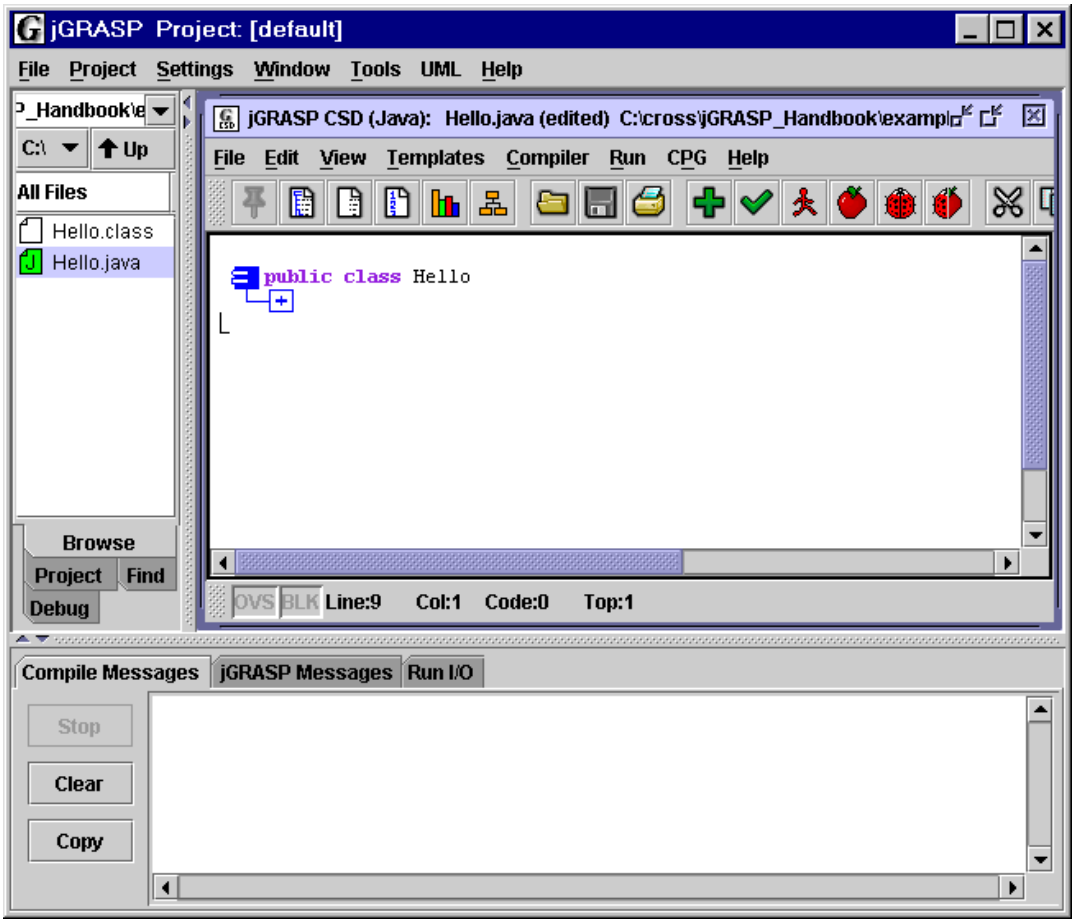

Figure 21. Folded CSD

2.7 Line Numbers

Line numbers can be very useful when referring to specific lines or regions of a program. Although not part of the actual program, they are displayed to he left of the source code as indicated in Figure 22.


Line numbers can be generated by clicking the line number icon on the CSD Window toolbar, and removed by clicking the icon again. Line numbers can also be generated/removed from the View menu.

With Line numbers turned on, new line numbers are inserted and/or added to the end each time you press “ENTER” on the keyboard. If you insert a line in the code, all line numbers below the new line are incremented.


You may “freeze” the line numbers to avoid the incrementing by clicking on the Freeze Line Numbers icon. To unfreeze the line number, click the icon again. This feature is also available on the View menu.

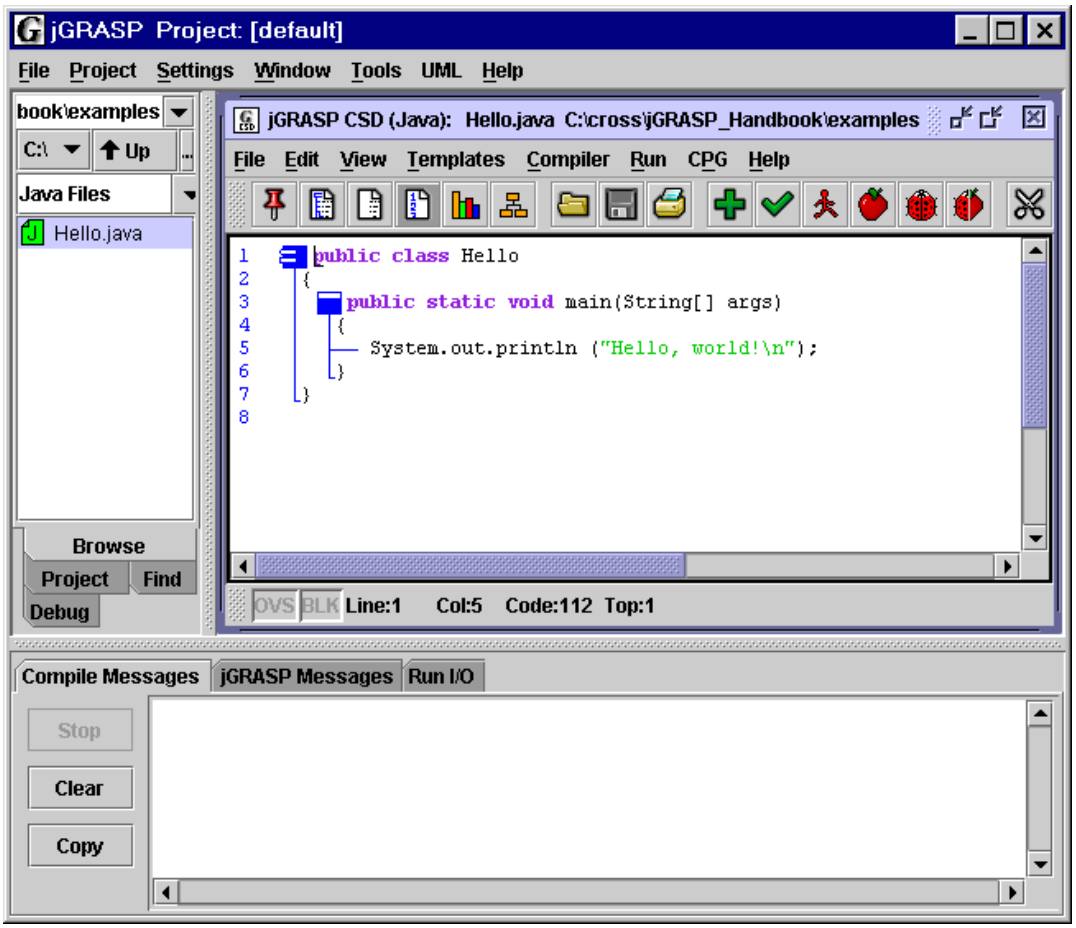

Figure 22. Line numbers in the CSD Window

2.8 Compiling □Progr□m

After you have a program in the CSD Window, either by loading a file or typing it in and saving it, you are ready to compile the program. If you are compiling a language other than Java, you will need to "compile and link" the program.

 Compile the program in jGRASP by clicking on **Compiler -- Compile** (Figure 23) or you click the Compile icon (green plus for Java).

 Compile and Link the program if you are compiling a language other than Java, by clicking on **Compiler – Compile □nd Link** or you click the Compile and Link icon (green plus plus). Note, these options will not be visible on the tool bar and menu in a CSD Window for a Java program.

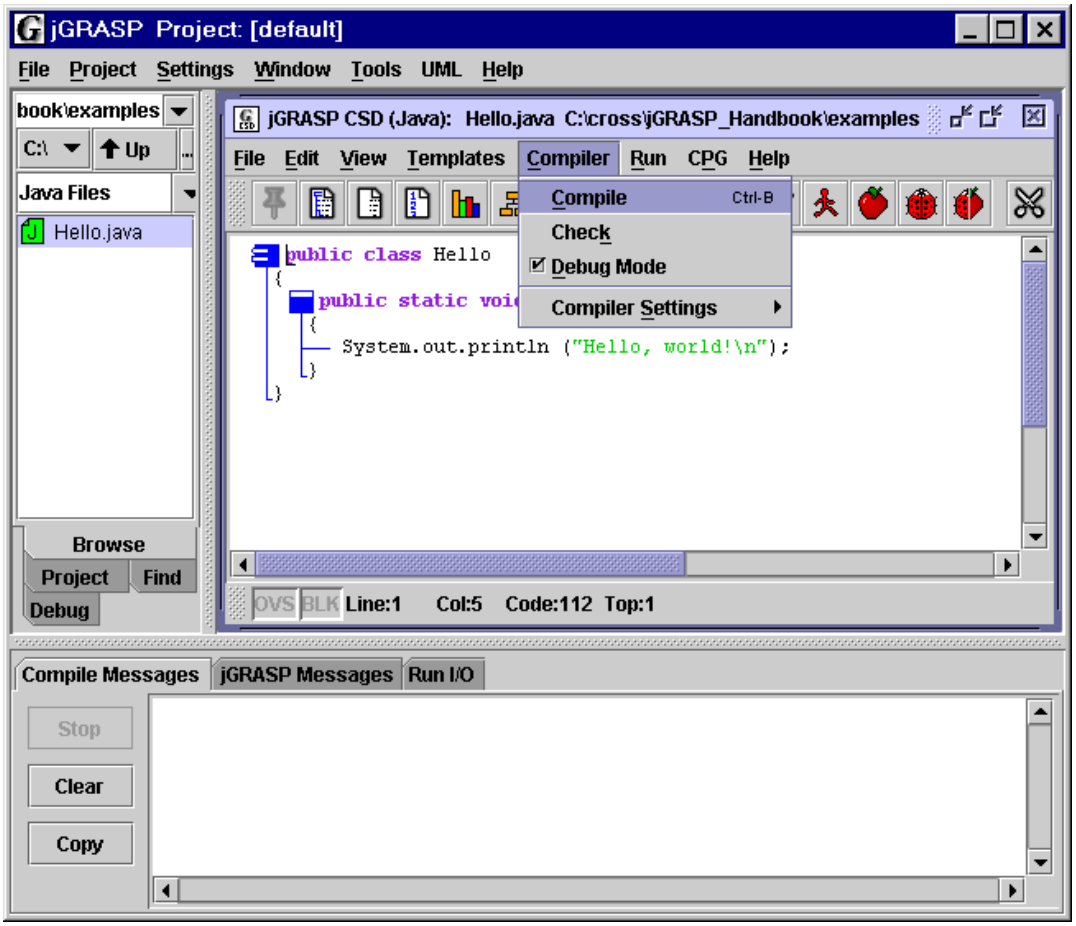

Figure 23. Compiling □progr□m

The results of the compilation will appear in the **Compile Messages tab** in the lower window of the Desktop. If your program compiled successfully, you should see the message “operation complete” with no errors reported, as illustrated in Figure 24, and you are now ready to "Run" the program (see next section).

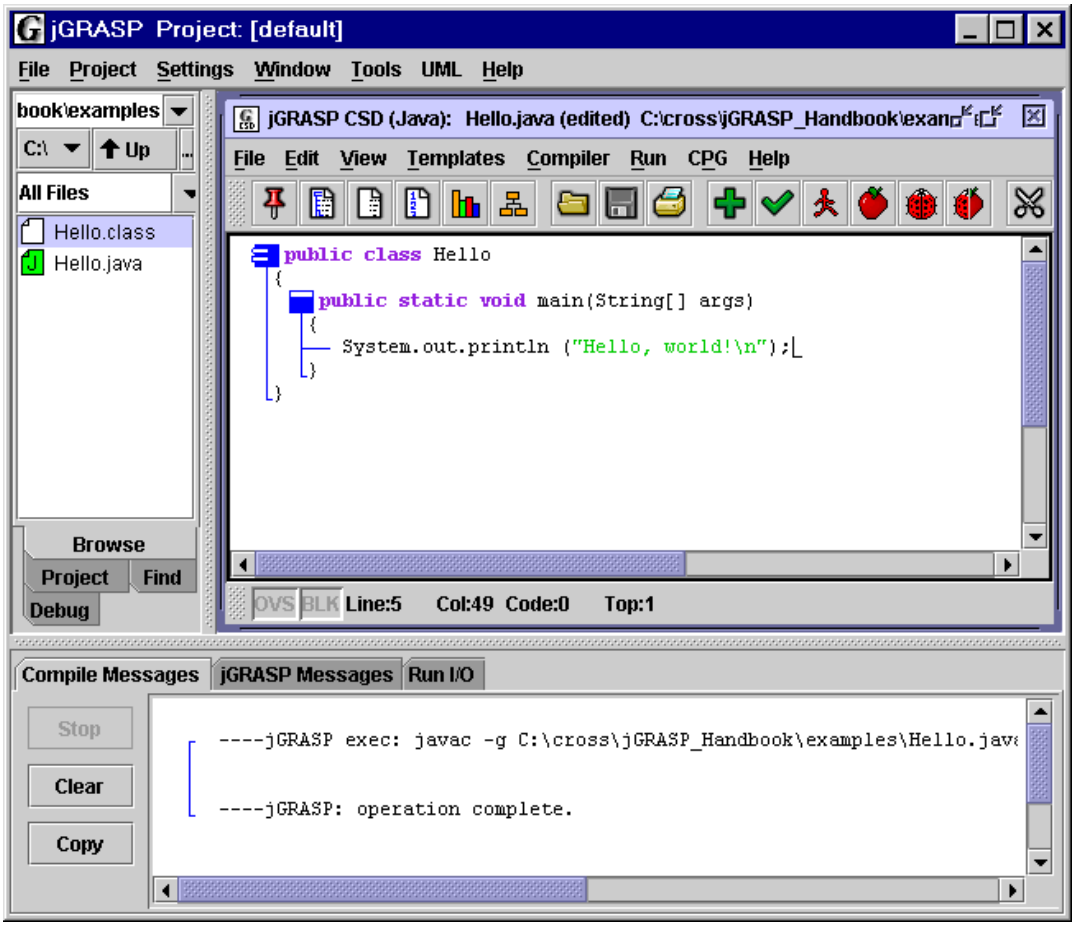

Figure 24. A successful compilation

Error Messages

If you receive an error message indicating “file not found,” this generally means jGRASP could not find the compiler. For example, if you are attempting to compile a Java program and the message may indicate that “javac” was not found. Go back to Section 1, Installing jGRASP, and be sure you have followed all the instructions. Once the compiler is properly installed and set up, any errors reported should be about your program.

If your program does not compile, the errors reported by the compiler will be displayed in the Compile Messages window (Figure 25). The description of first error detected will be highlighted, and jGRASP automatically scrolls the CSD Window to the line where the error most likely occurred and highlights it.

Even if multiple errors are indicated, as soon you correct the first error reported, you should attempt to compile the program again. Sometimes a single error causes a cascade of reported errors.

Only after you have "fixed" all these reported errors will your program actually compile. Only then you are ready to "Run" the program as described in the next section.

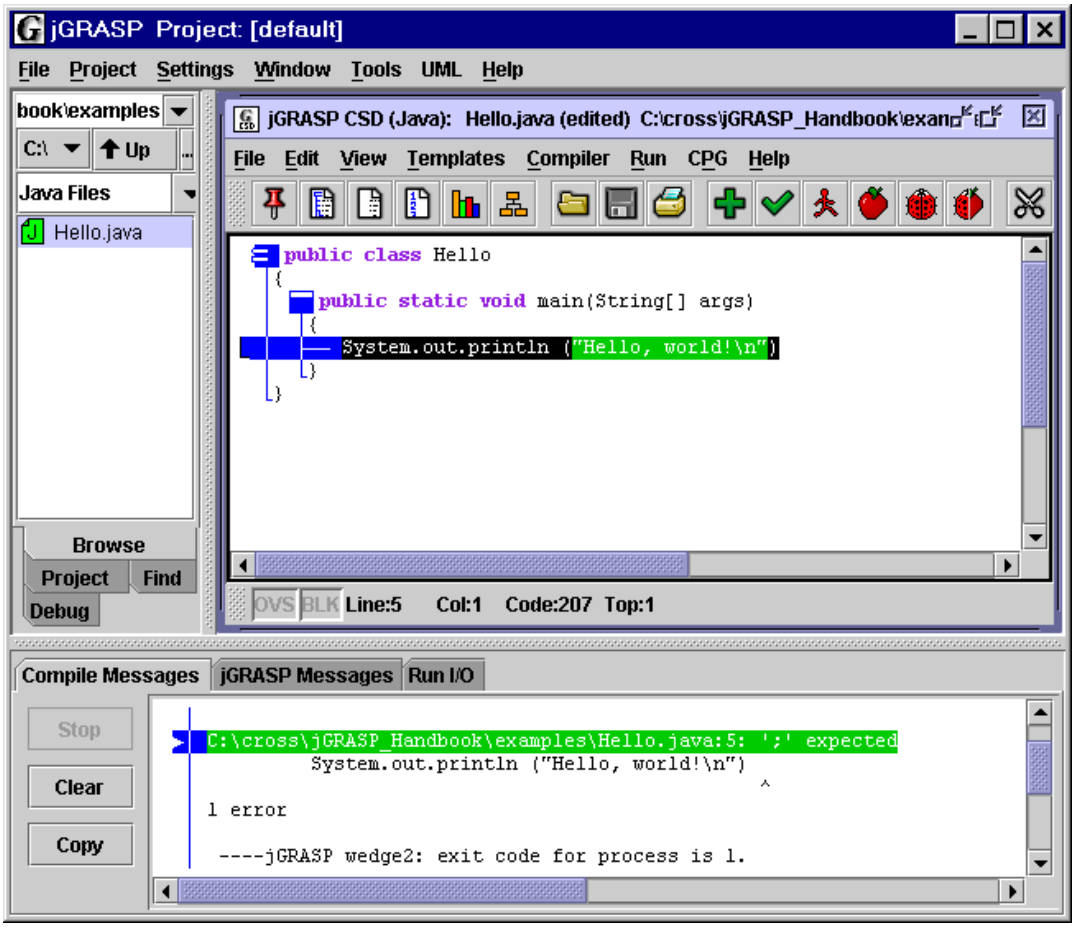

Figure 25. Compile time error reported

2.9 Running □ Progr□m

At this point you should have successfully compiled your program. Two things indicate this. First, there should be no errors reported in the Compile Messages window. Second, you should have a Hello.class file listed in the Browse pane, assuming the pane is set to list "All Files."

To run the program, click **Run – Run** on the CSD Window tool bar (Figure 26). The options on the Run menu allow you to run your program as an application (**Run**), as an Applet (**Run □s Applet**), as an application debug mode (**Debug**), as an Applet in debug mode (**Debug □s Applet**). Other options allow you to pass Run arguments and Run in an MS-DOS window rather than the jGRASP Run I/O message pane.

 You can also run the program by clicking the Run icon on the tool bar.

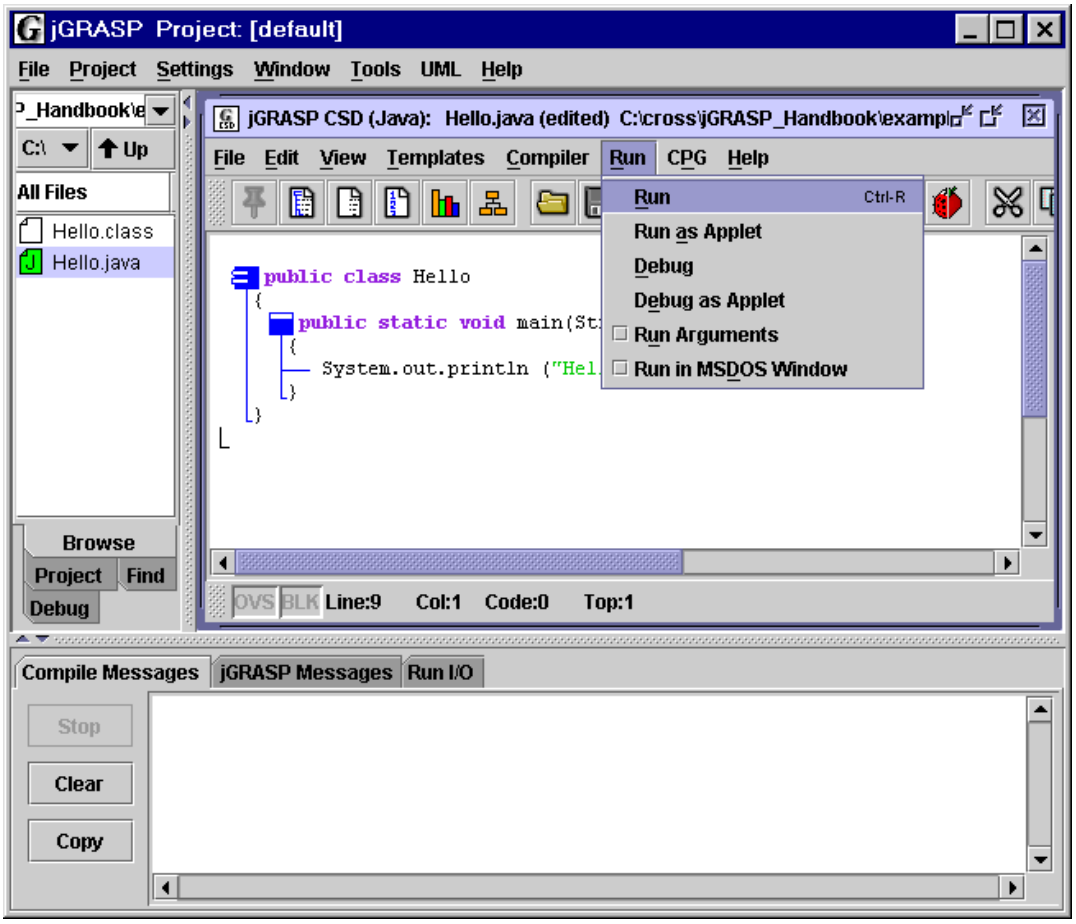

Figure 26. Running □ progr□m

Output

When you run your program, the Run I/O tab in the lower pane pops to the top of the Desktop. The results of running the program are displayed in this pane as illustrated in Figure 27.

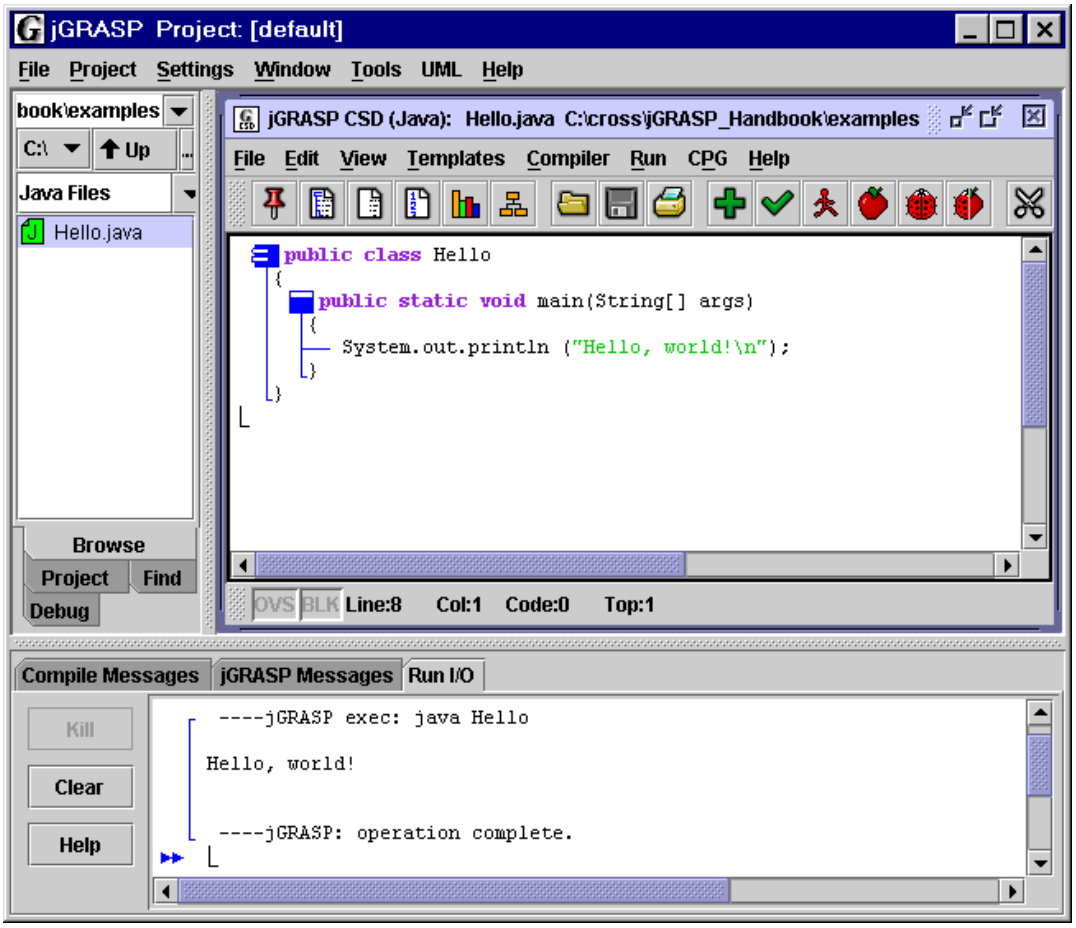

Figure 27. Output from running the progr□m

2.10 Exiting jGRASP

When you have completed you session with jGRASP, you should “exit” jGRASP rather than leaving it open for Windows to close when you log out or shut down your computer. When you exit jGRASP normally, it saves its current state and closes all the files you were working on. If a file was edited during the session, it prompts you to save or discard the changes. The next time you start jGRASP, it will open your files, and you will be ready to begin where you left off.