

Supplement I.C

Creating, Compiling and Running Java Programs from the Command Window

For Introduction to Java Programming

By Y. Daniel Liang

This supplement covers the following topics:

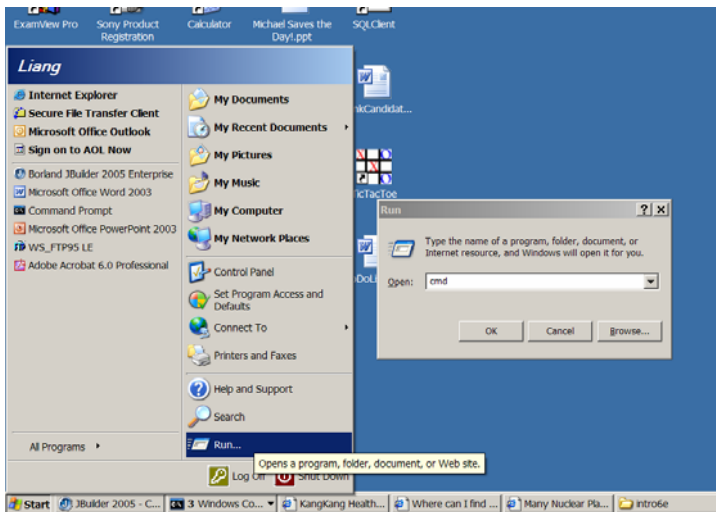
- Opening a Command Window (§1.2).
- Using Simple DOS Commands (create, change directories, display files, delete files and directories) (§1.3).
- Creating and Editing Programs Using Notepad and WordPad (§1.4).
- Compiling and Running Programs (§1.5).
- Common Errors (§1.6).

1.1 Introduction

Sun releases each version of Java with a Java Development Toolkit (JDK). This is a primitive command-line tool set that includes software libraries, a compiler for compiling Java source code, an interpreter for running Java bytecode, and an applet viewer for testing Java applets, as well as other useful utilities. JDK 1.6 can be downloaded from <http://java.sun.com/javase/downloads/index.jsp>. See Supplement I.B, "Installing and Configuring JDK," on how to install JDK 1.6.

1.2 Opening a Command Window

Using JDK from Windows, you have to type the commands from the command window. Assume you have successfully installed JDK. Start a Command window by clicking the Windows Start button and choosing Run to open the Run dialog box, as shown in Figure 1.1. Enter cmd in the Open field and click OK to display a command window, as shown in Figure 1.2.



← The Run dialog

Figure 1.1

You can start the command window from the Windows Start button.

You will see the command prompt window displayed as shown in Figure 1.2.



Figure 1.2

You can type the JDK commands in the command window.

Now type **javac** in the DOS prompt. If your JDK is configured properly, the command displays the usage of the command, as in Figure 1.3. Otherwise, an error message is displayed to indicate that the command is not found, as shown in Figure 1.4. Please refer to Supplement I.B, "Installing and Configuring JDK," to configure JDK 1.6.

```

C:\>javac
Usage: javac <options> <source files>
where possible options include:
  -g                Generate all debugging info
  -g:none           Generate no debugging info
  -g:<lines,vars,source> Generate only some debugging info
  -nowarn           Generate no warnings
  -verbose          Output messages about what the compiler is doing
  -deprecation      Output source locations where deprecated APIs are used
  -classpath <path> Specify where to find user class files
  -cp <path>        Specify where to find user class files
  -sourcepath <path> Specify where to find input source files
  -bootclasspath <path> Override location of bootstrap class files
  -extdirs <dirs>   Override location of installed extensions
  -endorseddirs <dirs> Override location of endorsed standards path
  -d <directory>   Specify where to place generated class files
  -encoding <encoding> Specify character encoding used by source files
  -source <release> Provide source compatibility with specified release
  -target <release> Generate class files for specific VM version
  -version          Version information
  -help            Print a synopsis of standard options
  -X               Print a synopsis of nonstandard options
  -J<flag>         Pass <flag> directly to the runtime system
C:\>_

```

Figure 1.3

The *javac* command displays the usage of the command.

```

C:\>javac
'javac' is not recognized as an internal or external command,
operable program or batch file.
C:\>

```

Figure 1.4

The *javac* command is not found if JDK is not properly configured.

1.3 Simple DOS Commands

To work with JDK on Windows, you need to know some simple DOS commands. Here are several frequently needed commands:

- **dir** -- Lists all files in the directory.
- **mkdir dirName** -- Creates a new directory named dirName.
- **cd dirName** -- Changes to the specified directory. For example, **cd c:** changes to the directory c:\
- **cd ..** -- Changes to the parent directory.
- **del filename** - Deletes a file. For example, **del Test.java** deletes the file named Test.java in the current directory.
- **del *.*** - Deletes all files in the directory. [Caution: files deleted from the command window cannot be recovered.]
- **rmdir dirName** - Deletes the specified directory dirName.

- **type filename.java** - Displays the contents of the specified file.

1.4 Creating and Editing Programs Using Notepad and WordPad

You can create programs using Windows Notepad or WordPad. To do so, first open a command window; change the directory to where your programs should be stored, and type **notepad filename.java** or **write filename.java** to create a file for Java source code. For example, the following command starts Notepad to create or edit Welcome.java:

notepad Welcome.java

If Test.java does not exist, a dialog box is displayed to alert you that it is a new file, as shown in Figure 1.5.

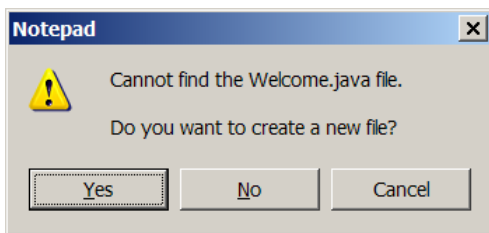


Figure 1.5

The Notepad alerts you that the specified file is new.

Click **Yes** to create a new file. You can type the code for the program in the Notepad, as shown in Figure 1.6. Choose **File, Save** to save the file.

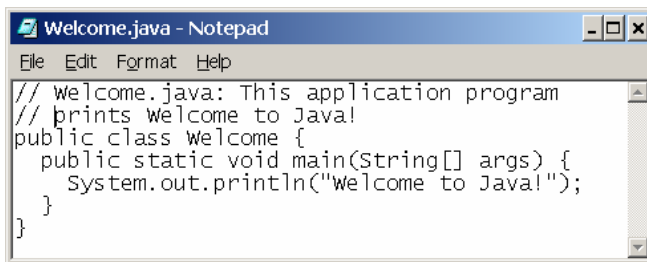


Figure 1.6

You can use Notepad to create and edit Java source files.

1.5 Compiling and Running Java Programs

To compile the source code, use the **javac** command. For example, the following command compiles Welcome.java:

javac Welcome.java

To run the class, use the **java** command. For example, the following command runs Welcome:

java Welcome

Figure 1.7 shows a sample run.



Figure 1.7

Use the **javac** command to compile and **java** to programs.

1.6 Common Errors

Question 1. I tried to run the program from the DOS prompt, but got java.lang.NoClassDefFoundError exception. What is wrong?

Answer: The class file is not found. There are several reasons: 1. the class has been compiled; 2. the classpath was not set properly. You need to set classpath to **.;%classpath%**. 3. If the class has a package statement, you didn't invoke it with the full class name including the complete package name. For example, if the class path is **c:\liang** and the package for the class is **chapter1**, you have to type **java chapter1.ClassName** from the **c:\liang** directory.

Question 2. I tried to run the program from the DOS prompt, but got java.lang.NoSuchMethodFoundError exception. What is wrong?

Answer: The class does not have a main method or the main method signature is incorrect.