

## Working with JDK on Windows

This supplement covers the following topics:

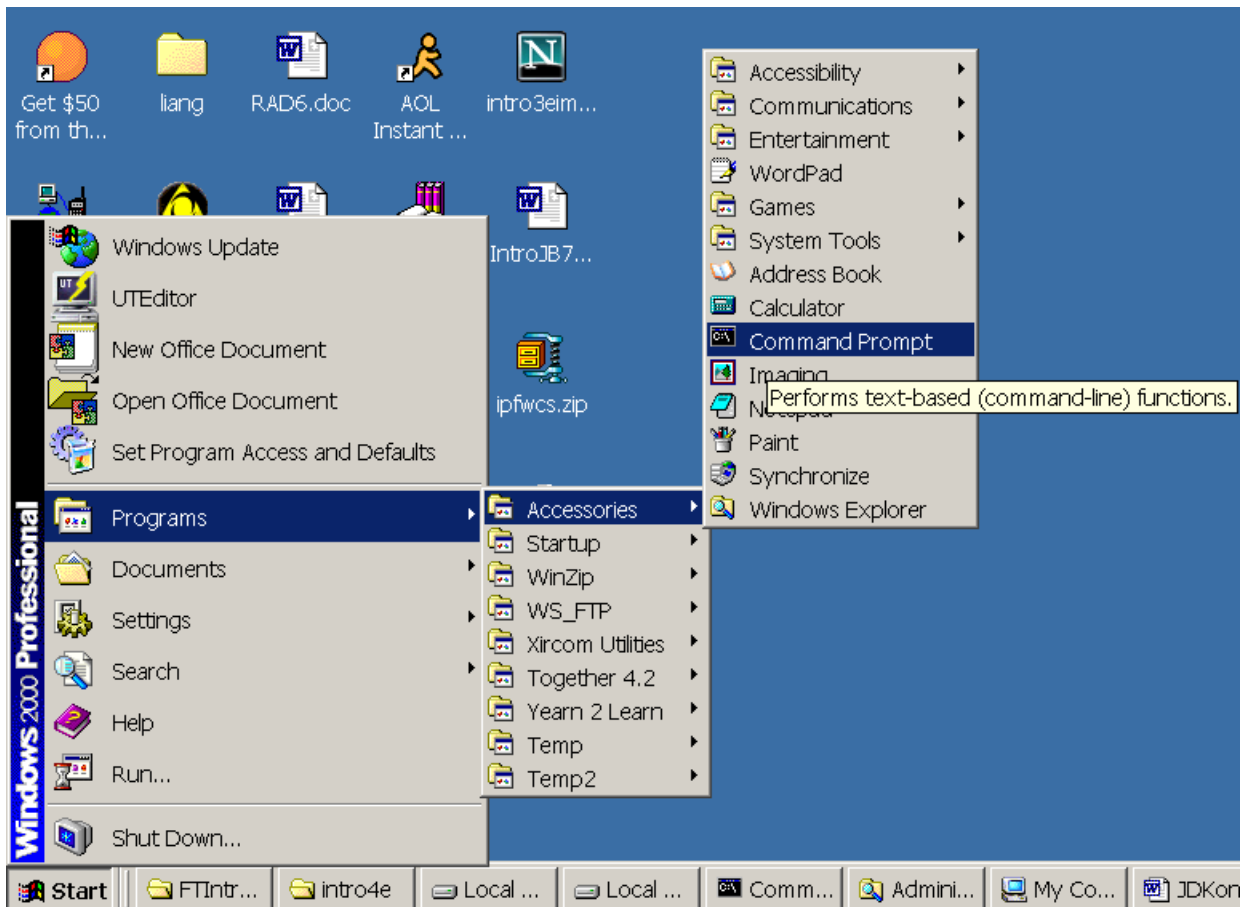
- Opening a Command Window
- Configuring JDK 1.4
- Using Simple DOS Commands (create, change directories, display files, delete files and directories)
- Creating and Editing Programs Using Notepad
- Compiling and Running Programs
- FAQs on Using JDK on Windows

### 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 1.4 (version 1.4.0\_02) is included in the CD-ROM of [Introduction to Java Programming, 4E](#). It can also be downloaded from <http://java.sun.com/j2se/1.4/download.html>.

### 1.2 Opening a Command Window

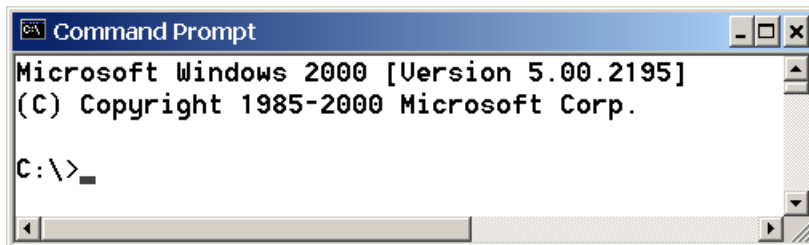
Using JDK from Windows, you have to type the commands from the command window. Assume you have successfully installed JDK 1.4. Start a Command window by clicking the Windows Start button, Programs, MS-DOS Prompt in Windows 98, or Programs, Accessories, Command Prompt in Windows 2000, as shown in Figure 1.1.



**Figure 1.1**

*You can start the command prompt from the Windows Start button.*

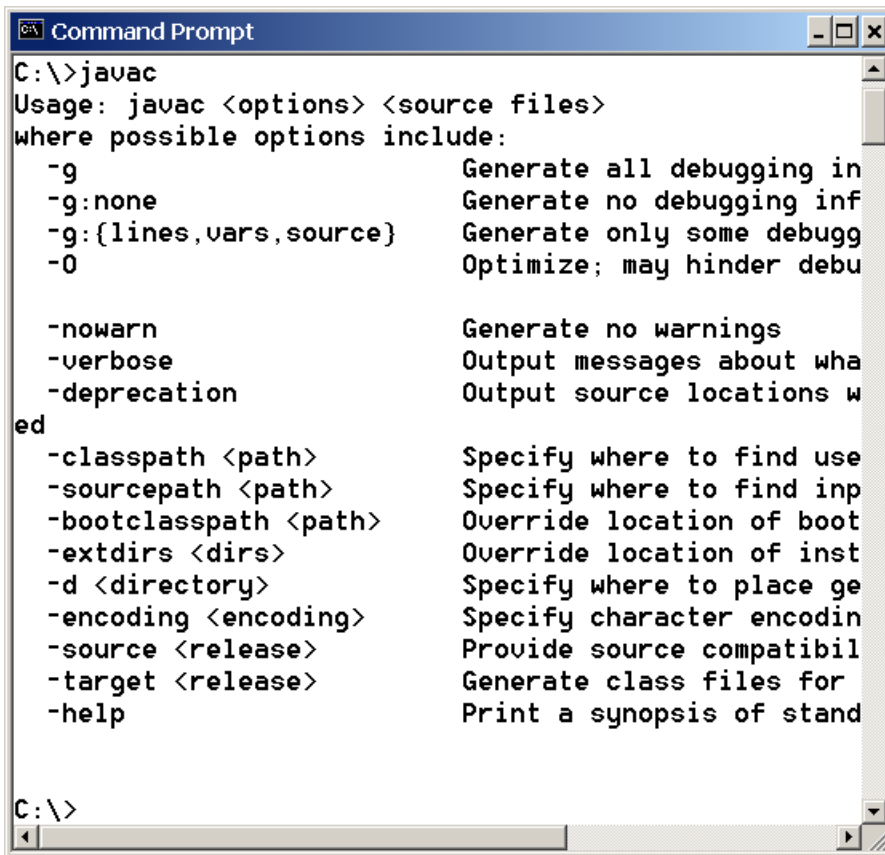
You will see the command prompt window displayed as shown in Figure 1.2.



**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.

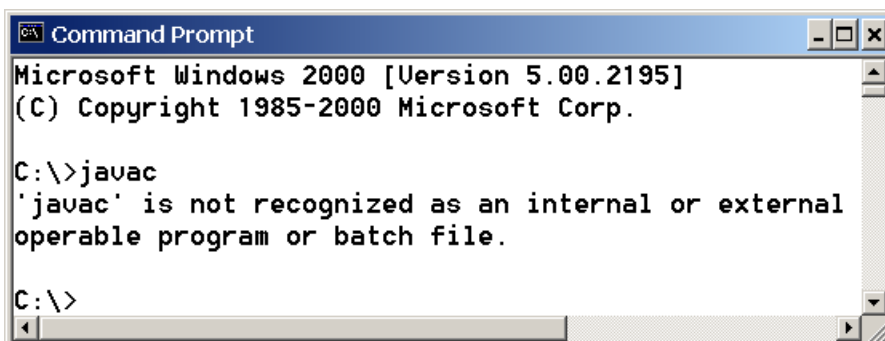


```
Command Prompt
C:\>javac
Usage: javac <options> <source files>
where possible options include:
  -g                Generate all debugging information
  -g:none           Generate no debugging information
  -g:{lines,vars,source}  Generate only some debugging information
  -O                Optimize; may hinder debugging
  -nowarn           Generate no warnings
  -verbose          Output messages about what is compiled
  -deprecation      Output source locations where deprecated
  -classpath <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
  -d <directory>  Specify where to place generated class files
  -encoding <encoding> Specify character encoding to use
  -source <release> Provide source compatibility: Specify the target release
  -target <release> Generate class files for the specified release
  -help            Print a synopsis of standard options

C:\>
```

Figure 1.3

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



```
Command Prompt
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

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

C:\>
```

Figure 1.4

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

### 1.3 Configuring JDK 1.4

To figure JDK is to make it available in the operating system so Windows can find your JDK commands such as **javac**.

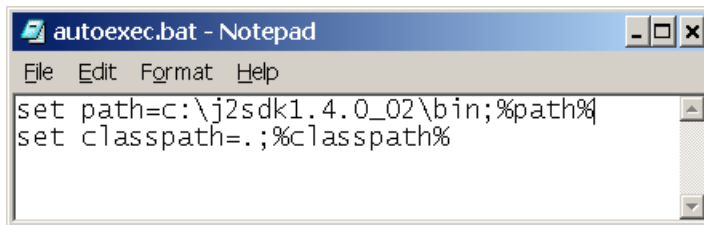
#### 1.3.1 Configuring JDK 1.4 on Windows 95 and 98

To configure JDK 1.4 on Windows 95 and 98, update the autoexec.bat file as follows:

1. Open a Windows command prompt window and type **notepad autoexec.bat** from the C:\ prompt to display the Notepad window.
2. Type the following two lines in the Notepad window, as shown in Figure 1.5.

```
set path=%path%;c:\j2sdk1.4.0_02\bin  
set classpath=.;%classpath%
```

NOTE: The first line ensures that the JDK commands can be found in the Windows. The second line ensures that the class files can be properly located. The . symbol in the second line indicates the current directory. Including the . symbol in the classpath enables you to load the classes from the current directory.



**Figure 1.5**

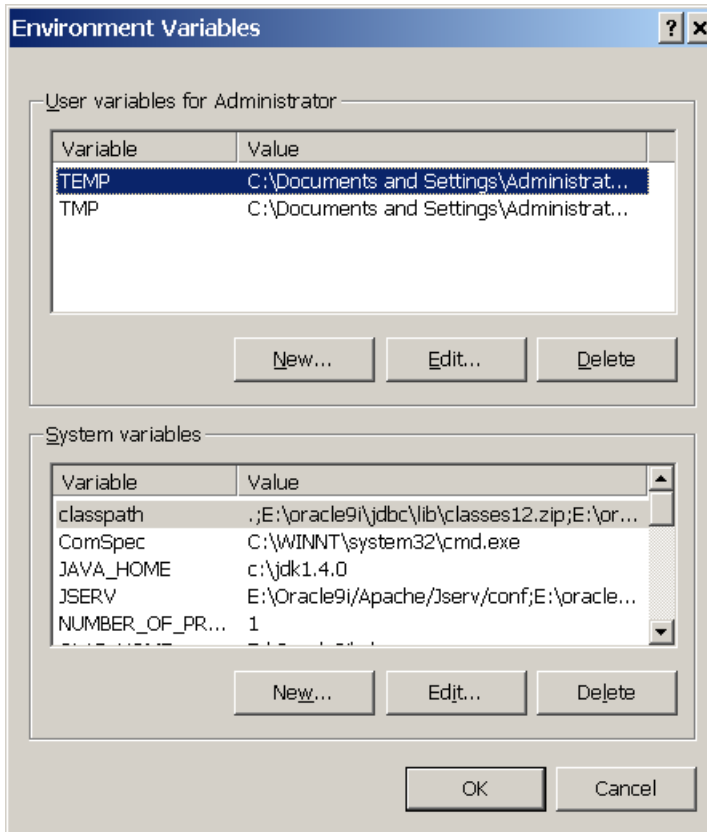
The **javac** command is not found if JDK is not properly configured.

3. Choose *File*, *Save* in the Notepad window to save the file and close the Notepad window.
4. autoexec.bat file is executed every time you start your computer. Reboot your system to have the new configurations to take effect.

#### 1.3.2 Configuring JDK 1.4 on Windows NT, 2000, and XP

To configure JDK 1.4 on Windows NT, 2000, and XP, set the environment variables as follows:

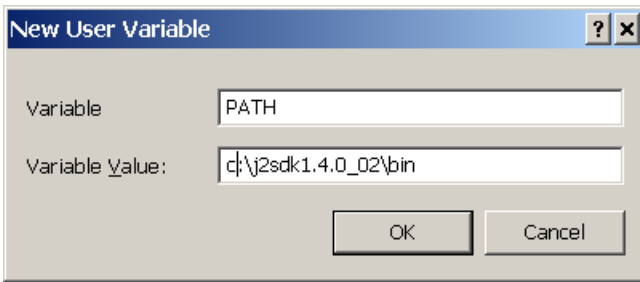
1. From the Windows Start button, choose *Settings, Control Panel* to display the Control Panel window.
2. Click *System* in the Control Panel window to open the System Properties window.
3. In the System properties window, click *Environment Variables* in the Advanced tab to display the Environment Variables window, as shown in Figure 1.6. (For Windows XP users, choose Classic View tab instead of the Advanced tab.)



**Figure 1.6**

*The Environment Variables window enables you to set user and system variables.*

4. You can set or modify user variables or systems variables. User variables affect the individual users and system variables affect all the users in the system. In the User variables section, select PATH and click *Edit* if PATH is already a user variable. Otherwise, click *New* to display the New User Variable window as shown in Figure 1.7.



**Figure 1.7**

*The New User Variable enables you to set a new user variable and its value.*

5. Type `PATH` in the Variable field and `c:\j2sdk1.4.0_02\bin` in the Variable Value field, as shown in Figure 1.7. Click OK.
6. Similarly set the user variable for CLASSPATH with value `.` (period).

NOTE: You don't have to reboot the computer, but you have to open a new command window to use JDK commands.

#### **1.4 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.

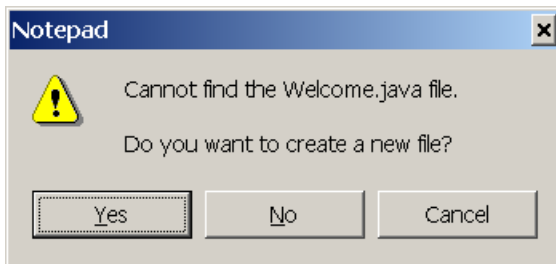
#### **1.5 Creating and Editing Programs Using Notepad**

You can create programs using Windows Notepad. To do so, first open a command window, change the directory to where

your programs should be stored, and type `notepad 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**

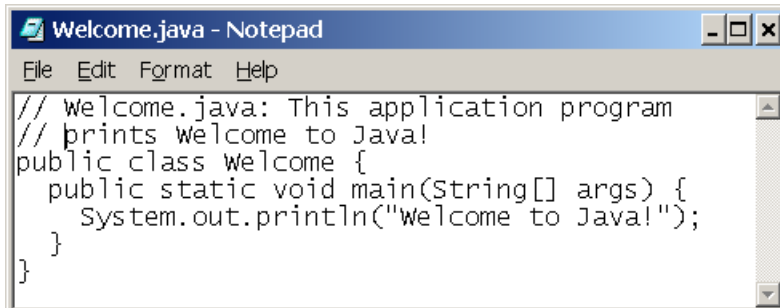
If `Test.java` does not exist, a dialog box is displayed to alert you that it is a new file, as shown in Figure 1.8.



**Figure 1.8**

*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 1.9. Choose *File, Save* to save the file.



**Figure 1.9**

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

## 1.6 Compiling and Running Java Programs

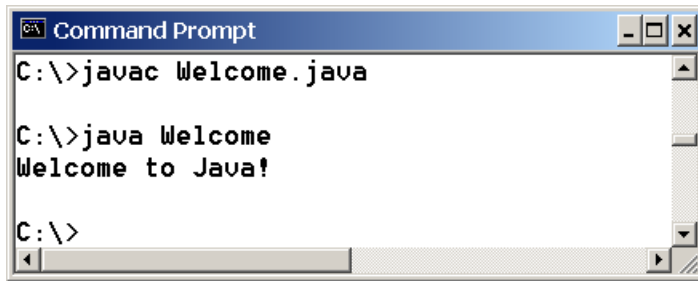
To compile the source code, use the **javac** command. For example, the following command compiles `Welcome.java`:

**javac Welcome.java**

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

**java Welcome**

Figure 1.10 shows a sample run.



```
Command Prompt
C:\>javac Welcome.java

C:\>java Welcome
Welcome to Java!

C:\>
```

**Figure 1.10**

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

### 1.7 Frequently Asked Questions

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

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.

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

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