(Sign In/Register for Account | Help) United States Communities I am a... I want to... Secure Search

Products and Services Solutions Downloads Store Support Training Partners About Oracle Technology Network

Oracle Technology Network > Java

Support
Create or Update Service Request
Search Support Knowledge Database
Download Patches

Education
Find Oracle University Training
Choose an Oracle Certification Path

Partner
Find a Partner Solution
Access Software and Technical Support
Attend a Partner-Only Event
Attend Training
Download Logos
Manage Solutions Catalog

Other Tasks
Purchase Oracle Products Online
Browse Oracle Documentation
Search Oracle Documentation
Check Oracle Earnings
Check Oracle Price Lists
Download Oracle Software
Find an Oracle ACE
View Oracle Benchmarks

Java SE
Java EE
Java ME
Java SE Support
Java SE Advanced & Suite
Java Embedded
JavaFX
Java DB
Web Tier
Java Card
Java TV
New to Java
Community
Java Magazine
Java Advanced

Essentials, Part 1, Lesson 1: Compiling & Running a Simple Program

Training Index

[<<BACK] [CONTENTS] [NEXT>>]

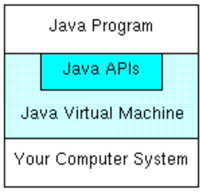
The computer age is here to stay. Households and businesses all over the world use computers in one way or another because computers help individuals and businesses perform a wide range of tasks with speed, accuracy, and efficiency. Computers can perform all kinds of tasks ranging from running an animated 3D graphics application with background sound to calculating the number of vacation days you have coming to handling the payroll for a Fortune 500 company.

When you want a computer to perform tasks, you write a program. A program is a sequence of instructions that define tasks for the computer to execute. This lesson explains how to write, compile, and run a simple program written in the Java language (Java program) that tells your computer to print a one-line string of text on the console.

But before you can write and compile programs, you need to understand what the Java platform is and set your computer up to run the programs.

A Word About the Java Platform

The Java platform consists of the Java application programming interfaces (APIs) and the Java [1] virtual machine (JVM).



Java APIs are libraries of compiled code that you can use in your programs. They let you add ready-made and customizable functionality to save you programming time.

The simple program in this lesson uses a Java API to print a line of text to the console. The console printing capability is provided in the API ready for you to use; you supply the text to be printed.

Java programs are run (or interpreted) by another program called the Java VM. If you are familiar with Visual Basic or another interpreted language, this concept is probably familiar to you. Rather than running directly on the native operating system, the program is interpreted by the Java VM for the native operating system. This means that any computer system with the Java VM installed can run Java programs regardless of the computer system on which the applications were originally developed.

For example, a Java program developed on a Personal Computer (PC) with the Windows NT operating system should run equally well without modification on a Sun Ultra workstation with the Solaris operating system, and vice versa.

Setting Up Your Computer

Before you can write and run the simple Java program in this lesson, you need to install the Java platform on your computer system.

The Java platform is available free of charge from the java.sun.com web site. You can choose between the Java® 2 Platform software for Windows 95/98/NT or for Solaris. The download page contains the information you need to install and configure the Java platform for writing and running Java programs.

Note: Make sure you have the Java platform installed and configured for your system before you try to write and run the simple program presented next.

Writing a Program

The easiest way to write a simple program is with a text editor. So, using the text editor of your choice, create a text file with the following text, and be sure to name the text file `ExampleProgram.java`. Java programs are case sensitive, so if you type the code in yourself, pay particular attention to the capitalization.

```
//A Very Simple Example
class ExampleProgram {
  public static void main(String[] args){
    System.out.println("I'm a Simple Program");
  }
}
```

Here is the ExampleProgram.java source code file if you do not want to type the program text in yourself.

Compiling the Program

A program has to be converted to a form the Java VM can understand so any computer with a Java VM can interpret and run the program. Compiling a Java program means taking the programmer-readable text in your program file (also called source code) and converting it to bytecodes, which are platform-independent instructions for the Java VM.

The Java compiler is invoked at the command line on Unix and DOS shell operating systems as follows:

```
  javac ExampleProgram.java
```

Note: Part of the configuration process for setting up the Java platform is setting the class path. The class path can be set using either the `-classpath` option with the `javac` compiler command and `java` interpreter command, or by setting the `CLASSPATH` environment variable. You need to set the class path to point to the directory where the `ExampleProgram` class is so the compiler and interpreter commands can find it. See Java 2 SDK Tools for more information.

Interpreting and Running the Program

Once your program successfully compiles into Java bytecodes, you can interpret and run applications on any Java VM, or interpret and run applets in any Web browser with a Java VM built in such as Netscape or Internet Explorer. Interpreting and running a Java program means invoking the Java VM byte code interpreter, which converts the Java byte codes to platform-dependent machine codes so your computer can understand and run the program.

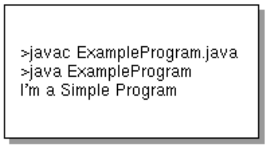
The Java interpreter is invoked at the command line on Unix and DOS shell operating systems as follows:

```
  java ExampleProgram
```

At the command line, you should see:

```
  I'm a Simple Program
```

Here is how the entire sequence looks in a terminal window:



```
>javac ExampleProgram.java
>java ExampleProgram
I'm a Simple Program
```

Common Compiler and Interpreter Problems

If you have trouble compiling or running the simple example in this lesson, refer to the Common Compiler and Interpreter Problems lesson in The Java Tutorial for troubleshooting help.

Code Comments

Code comments are placed in source files to describe what is happening in the code to someone who might be reading the file, to comment-out lines of code to isolate the source of a problem for debugging purposes, or to generate API documentation. To these ends, the Java language supports three kinds of comments: double slashes, C-style, and doc comments.

Double Slashes

Double slashes (`//`) are used in the C++ programming language, and tell the compiler to treat everything from the slashes to the end of the line as text.

```
//A Very Simple Example
class ExampleProgram {
  public static void main(String[] args){
    System.out.println("I'm a Simple Program");
  }
}
```

C-Style Comments

Instead of double slashes, you can use C-style comments (`/* */`) to enclose one or more lines of code to be treated as text.

```
/* These are
C-style comments
*/
class ExampleProgram {
  public static void main(String[] args){
    System.out.println("I'm a Simple Program");
  }
}
```

Doc Comments

To generate documentation for your program, use the doc comments (`/** */`) to enclose lines of text for the `javadoc` tool to find. The `javadoc` tool locates the doc comments embedded in source files and uses those comments to generate API documentation.

```
/** This class displays a text string at
*  the console.
*/
class ExampleProgram {
  public static void main(String[] args){
    System.out.println("I'm a Simple Program");
  }
}
```

`HTML`javadoc Home Page`javadoc`**API Documentation**

The Java platform installation includes API Documentation, which describes the APIs available for you to use in your programs. The files are stored in a `doc` directory beneath the directory where you installed the platform. For example, if the platform is installed in `/usr/local/java/jdk1.2`, the API Documentation is in `/usr/local/java/jdk1.2/doc/api`. **More Information**

See Java 2 SDK Tools for more information on setting the class path and using the `javac`, and `java` commands.

See Common Compiler and Interpreter Problems lesson in The Java Tutorial for troubleshooting help.

The javadoc Home Page has more information on the `javadoc` command and its output.

You can also view the API Documentation for the Java 2 Platform on the java.sun.com site.

[1] As used on this web site, the terms "Java virtual machine" or "JVM" mean a virtual machine for the Java platform.

[TOP]

Java SDKs and Tools

Java SE

Java EE and Glassfish

Java ME

JavaFX

Java Card

NetBeans IDE

Java Resources

New to Java?

APIs

Code Samples & Apps

Developer Training

Documentation

Java.com

Java.net

Student Developers

Tutorials

BOOKMARK E-mail this page Printer View