

# Programming Java



# Lesson 1

## Introduction to Java

### Contents

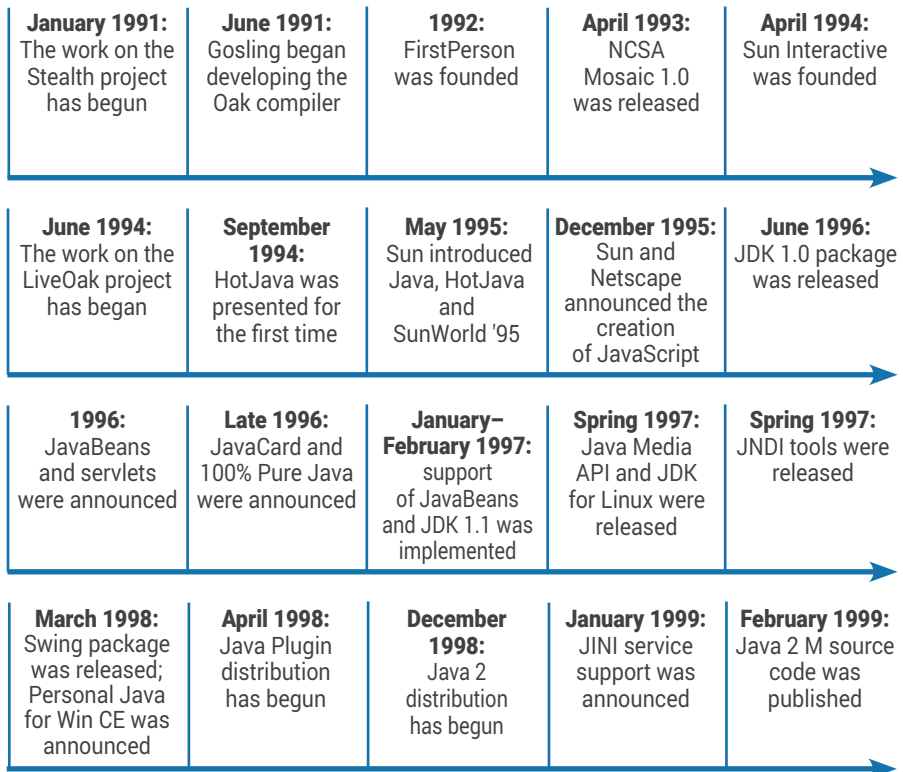
<b>1. Introduction to Java Technology.....</b>	<b>3</b>
1.1. Basic Concepts of Java Technology . . . . .	6
1.2. Lifespan, Compilations, and Execution of Java Application. . . . .	7
1.3. Installing Java Platform. . . . .	8
1.4. Example of Writing the First Application. . . . .	17
1.5. Overview of Existing IDEs . . . . .	20

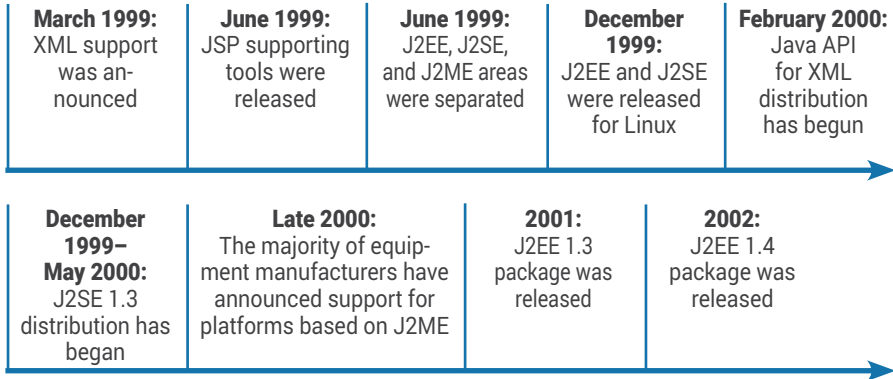
## 1. Introduction to Java Technology

Nowadays, there are many different operating systems (*Windows, Mac OS, Unix-like*). Software written for one operating system will not work on another operating system. This is due to the differences in structure of executables and functions of operating systems.

Java allows solving the problem of launching the same application on different operating systems, providing the users not only with the development language but with the entire platform.

### Stages of Development of Java Technology





### Aims and Objectives of the Java Technology

The main purpose of java technology is the principle “Write once – run anywhere” (WORA). The idea is that once the application code is written, it can be reused on various devices and operating systems without any additional effort.

An important task of Java is to ensure high performance of applications, which is achieved by quick conversion of bytecode into machine code language.

Much attention is paid to security in Java. The platform has a lot of classes for encryption and work with certificates.

Java language was conceived as a “language for housewives”, i.e. it is very easy to understand and learn.

### Comparison of Java Technology with Other Programming Technologies (.Net Framework, C++, etc.)

When Java was developed, C++ was taken as a basis for it, that’s why there are many similarities in syntax and semantics. Smalltalk language approach was taken as the basis of the object model.

### **Comparison of C++ and Java:**

- there is no operator overloading in Java;
- there is no direct memory access via the pointer in Java;
- there are no destructors in Java;
- there is no possibility of procedural programming in Java;
- in Java, the length of arrays is static, and the index is checked when accessing an element;
- in Java, instance variables (class fields) take the default values in case of absence of initialization (0 for integer and floating-point types, false for boolean, null for reference types);
- Java does not use preprocessor operations;
- Java has no header files;
- in Java, memory usage is controlled by the special garbage collection mechanism, while in C++, this is done by the programmer him/herself.

### **Java Platform and its Components**

Java platform is a set of tools for development in Java language, which are distributed to a developer as a single framework. The platform includes:

- standard libraries;
- compilation tools;
- code execution tools;
- tools for preparing documentation on the code;
- examples;
- other development tools.

## Java Platform Editions

Java platform is distributed as a single package of various editions (configurations). There is a separate platform version number for each edition.

- **JSE or J2SE** (*Java Standard Edition*) is a standard edition of Java platform used in console application development in the form of applets or standalone GUI applications that operate with databases and network.
- **JEE or J2EE** (*Java Enterprise Edition*) is used for developing distributed enterprise applications. It contains the Servlet, JSP, and EJB technologies.
- **JME** (*Java Micro Edition*) is an edition of the platform, which is used for application development for mobile devices (smartphones, tablets, etc.). It includes simplified standard classes, as well as classes for writing midlets.
- **Java FX** is a platform (2007) for the development of GUI applications. It is included in 8 JSE and was previously a separate library.

### 1.1. Basic Concepts of Java Technology

**Virtual Machine (JVM)** is a specification describing an abstract machine, which is able to execute Java applications. It is considered virtual since it mostly consists of various software components. There are JVMs implemented by different manufacturers, but they all adhere to a common [specification](#). JVM implementation is software that interprets bytecode operators of the program into operating system commands.

**Compiler** is a tool that converts the source code into bytecode. Java compiler is written in Java language. There are

compilers that convert the source code from other languages (Ada, JavaScript, Python, Ruby) into bytecode. There are also languages adapted for running on a virtual java machine (Scala, Groovy).

**Bytecode** is a set of operators in an intermediate language designed for execution by a virtual machine. Each operator is encoded by a single byte. There are 205 operators, and the rest (51) are reserve. Bytecode is created as a result of compiling a program code, which is located in the files with java extension. Bytecode is stored in files with the class extension.

The main principle of Java technology is “Write once – run anywhere” (WORA). Due to the features of a platform, the programs written in java are guaranteed to work on all operating systems, for which there is an implementation of java virtual machine (platform independence).

### **Garbage Collection**

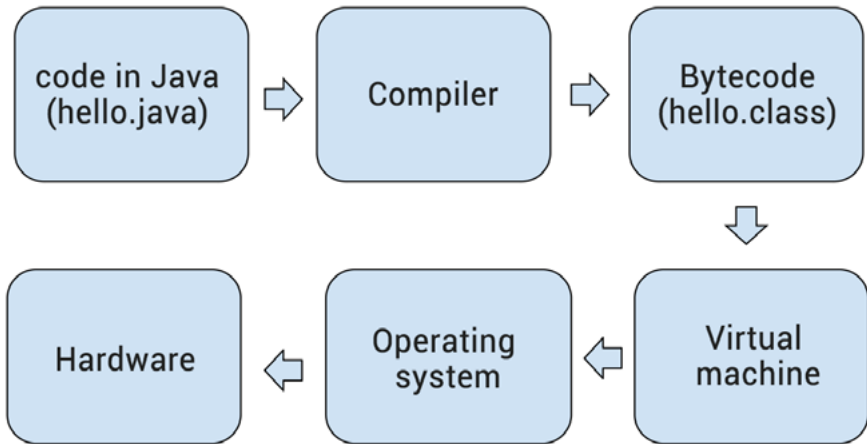
In some programming languages (such as C++), programmer is responsible for memory usage control, which complicates development process and causes a lot of errors and memory leaks.

In Java, memory usage control is performed by a special mechanism called garbage collector. The presence of automatic garbage collection mechanism (data that are no longer used) simplifies the development process and reduces the number of errors when working with memory.

## **1.2. Lifespan, Compilations, and Execution of Java Application**

Java application development begins with writing the source code. The code is stored in the files with java

extension. The source code is converted (compiled) by the compiler program into bytecode (files with the class extension). To execute a program, the bytecode is run on a virtual machine, which interprets it into operating system commands, which in turn, send commands to the hardware of an electronic device.



### 1.3. Installing Java Platform

#### Step 1.

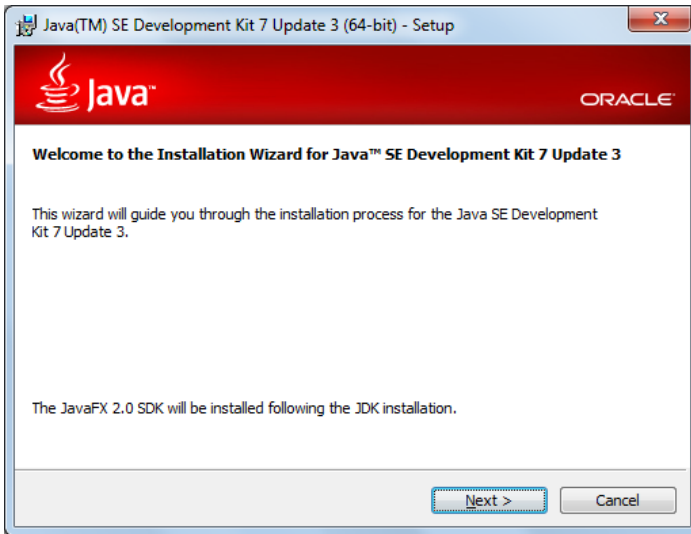
Installation package can be found and downloaded here: <http://www.oracle.com/technetwork/java/javase/downloads/index.html>.

Choose the JDK version and an operating system (note the bit set), on which the platform will be installed.

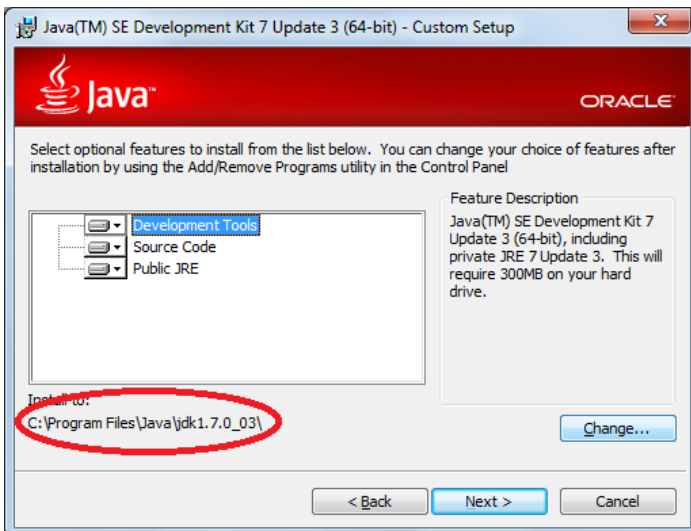


**Step 2.**

Run the install package.

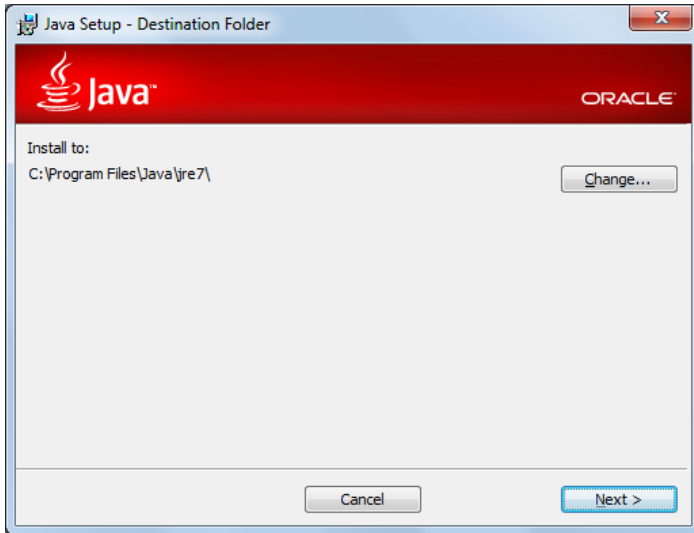
**Step 3.**

Change JDK installation path if necessary.



**Step 4.**

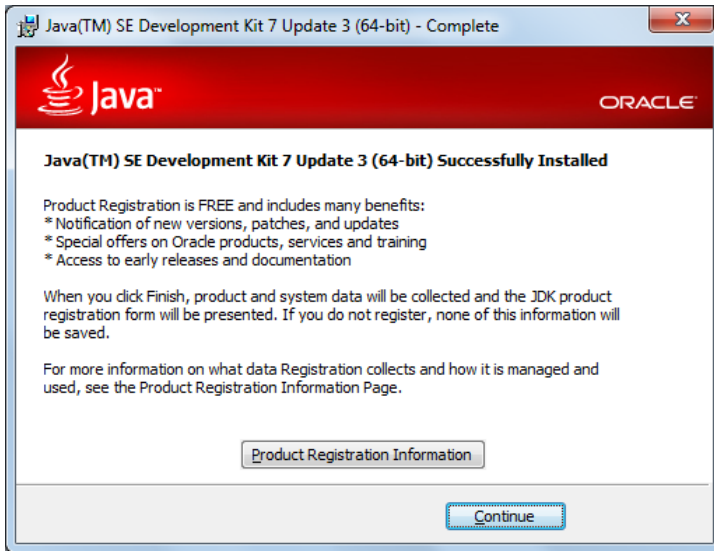
JRE will begin to install immediately after the JDK installation is complete.



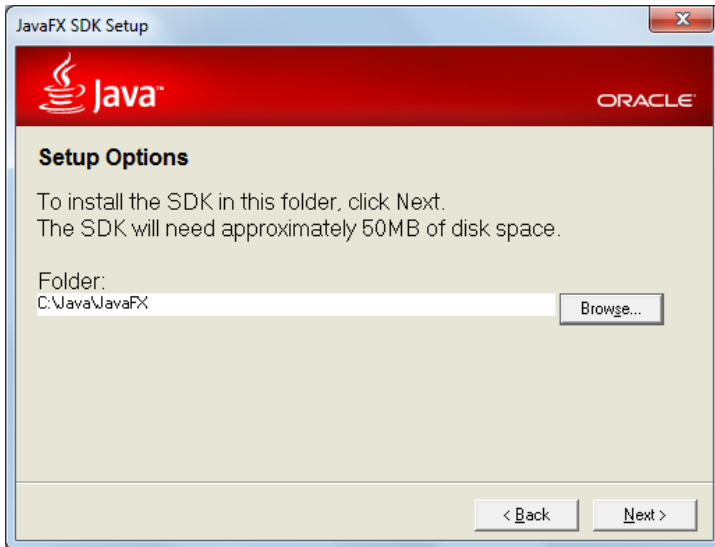
Installation process is displayed as the “Status” progress on the window.



*Installation is completed.* You can register the installed JDK on the “Oracle” website by pressing the “Product Registration Information” button. But this step is optional.



Starting with JSE version 1.7, installation package includes the JavaFX platform, and its installation will begin immediately after the installation of JRE.

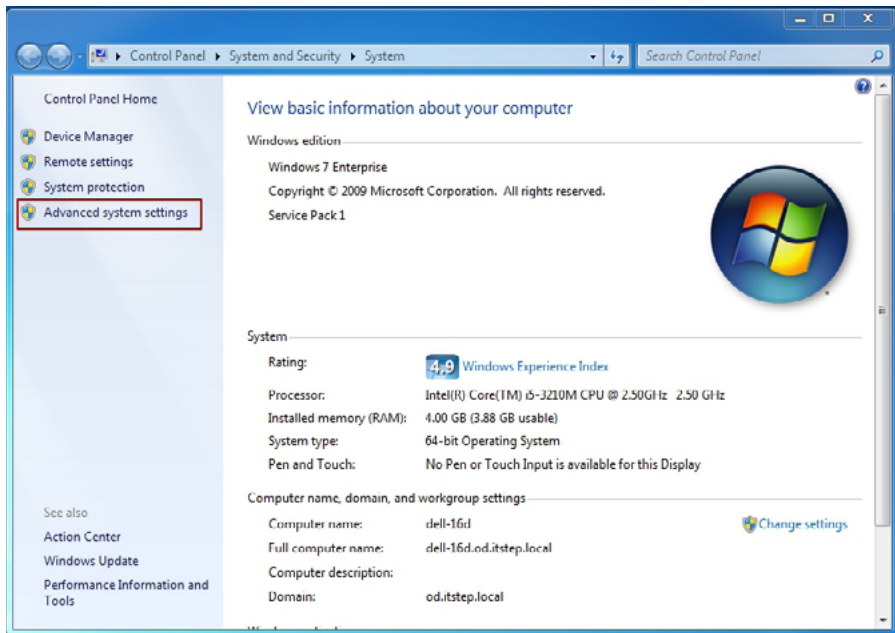


**Step 5.**

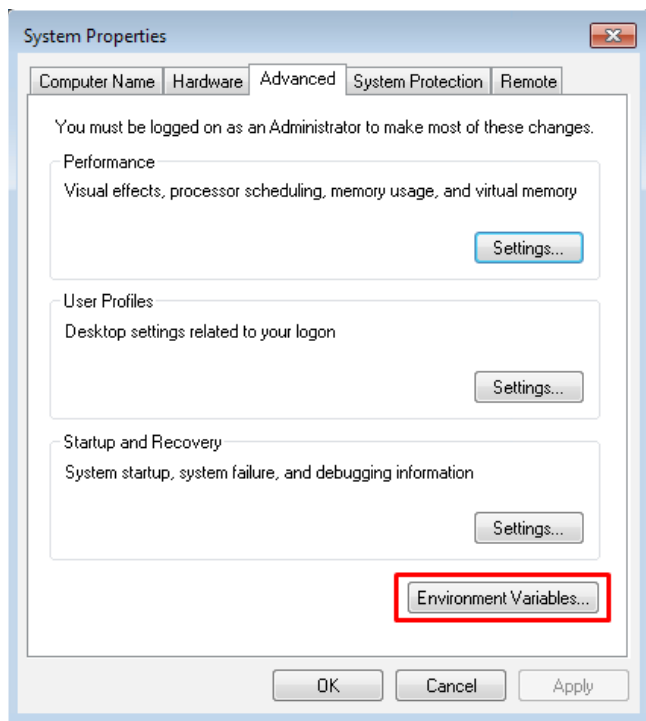
After the java platform is installed, it is necessary to configure operating system for working with java.

If you use Windows, right-click the “This PC” (“My Computer”) icon and select “Properties” from the context menu.

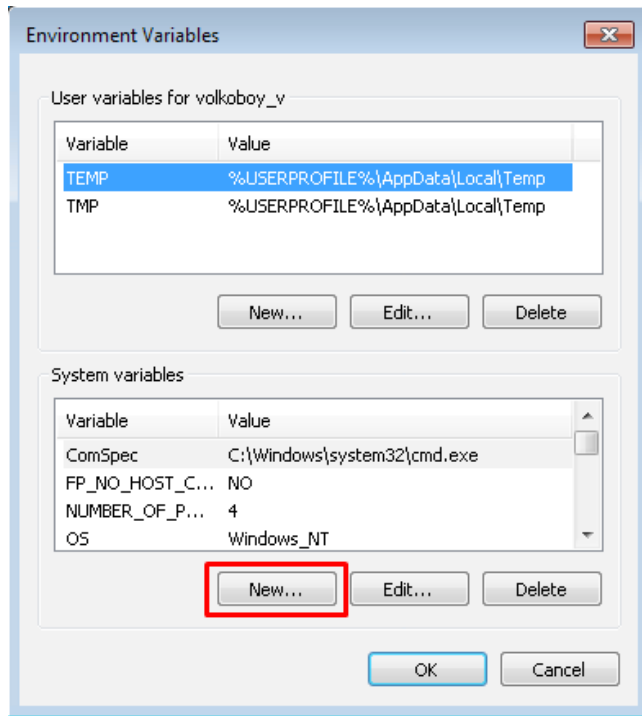
Select “Advanced System Settings” in the appeared window.



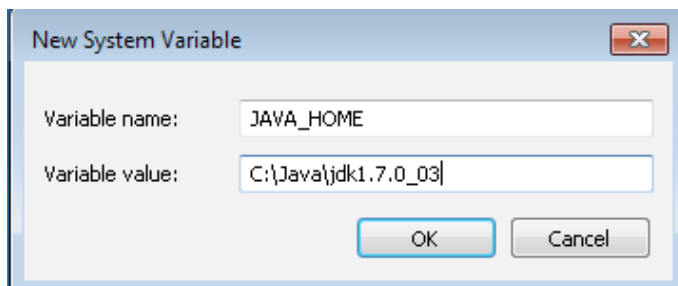
In the new window, press the “Environment variables” button.



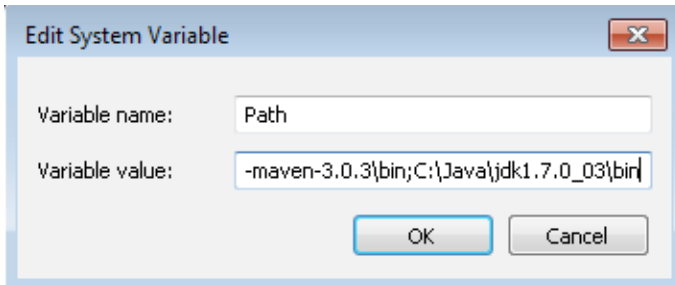
Then press the “New” button.



Enter the “JAVA\_HOME” value in the “Variable name” field, enter the path of JDK installation in the “Variable value” field, and then press “OK”.



Find and select the “Patch” item in the “System variables” field of the “Environment variables” window, then press the “Edit” button. In the “Variable value” field, add a semicolon and the path to the directory where the JDK utilities are installed (bin directory).



**ATTENTION!!!** Do not clear or remove anything from the “Variable value” field.

**JRE** (*Java Runtime Environment*) is a runtime for Java applications. This is a minimal set required for running Java applications. It contains the virtual machine and java class libraries.

**JDK** (*Java Development Kit*) is a set for java developer that includes:

- compiler (javac);
- standard class libraries;
- examples of code;
- documentation;
- utilities;
- runtime (java).

**java** is an executable file (located in the ...\\Java\\jdk\\bin folder) that launches a virtual machine and sends



bytecode for interpretation. Bytecode files are specified as parameters.

*For example:*

```
java.exe -d HelloWorld.class
```

***javac*** is a compiler of source code into bytecode.

*For example:*

```
javac.exe HelloWorld.java
```

***jar*** is a utility (java ARchive) for the creation of ZIP archives, in which the bytecode files and program resources are packaged. It is used for combining program modules into the integral logical units. Jar file can be executable if it contains a manifest file.

***javadoc*** is a utility for creating documentation on the basis of the source code.

***appletviewer*** is a utility for viewing applet type applications.

## 1.4. Example of Writing the First Application

In almost any Java program code development environment, the processes of compilation and running the program are hidden from the user by a set of various startup configurations. As a result, many novice developers do not know how to compile and run their program without the IDE, although such knowledge may be useful in compilation of various assembly scripts.

Our first program can be launched without the IDE in three easy steps.

**Step 1.**

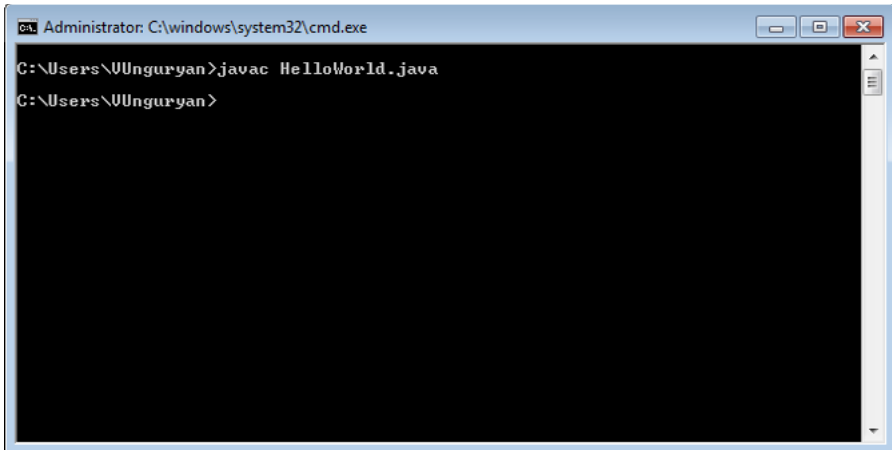
Create the HelloWorld.java file in the notepad. Add the code from Listing 1.1 to the file and save it:

```
Listing 1.1.
/**
 * My first program
 */
public class HelloWorld
{
    /**
     * Program entry point
     */
    public static void main(String[] args)
    {
        System.out.println("Hello world!");
    }
}
```

**NOTE!!!** *The file name and the letter case of the file name must exactly match the name of the class in the program (following the class keyword).*

**Step 2.**

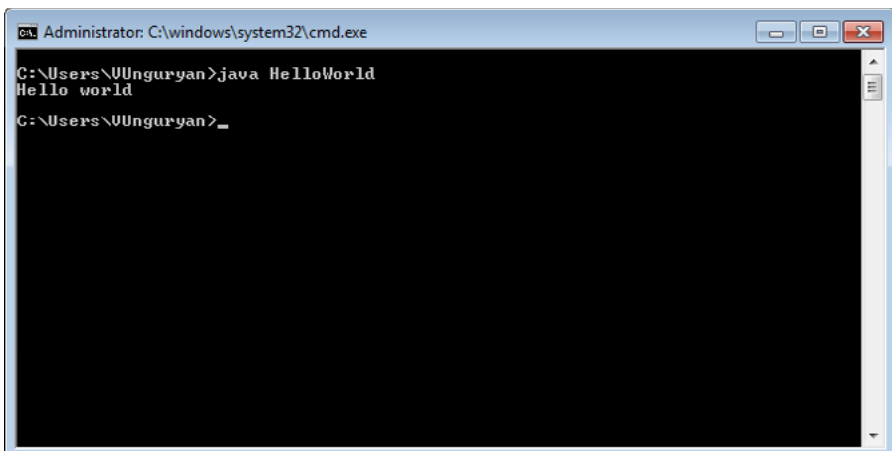
Open the command line utility. Enter “javac HelloWorld.java” and press “Enter”. In case of successful compilation, a new file will appear with the same name, but with the “class” extension. This is a file containing the bytecode of your program.



```
Administrator: C:\windows\system32\cmd.exe
C:\Users\Uunguryan>javac HelloWorld.java
C:\Users\Uunguryan>
```

### Step 3.

Enter “javac HelloWorld.class” in the command line and press Enter. The program will be executed, and the following message will be displayed in the command line window after its execution: **Hello world!**



```
Administrator: C:\windows\system32\cmd.exe
C:\Users\Uunguryan>java HelloWorld
Hello world
C:\Users\Uunguryan>_
```

## 1.5. Overview of Existing IDEs

A large number of different integrated development environments (*Hereinafter* — *IDE*) were created to simplify and speed up the process of program development in Java.

All IDEs can be divided into two main categories, which significantly influence the choice:

- paid;
- free.

Other important parameters when selecting the IDE include the time required for initial boot, environment installation time, the amount of occupied RAM and virtual memory.

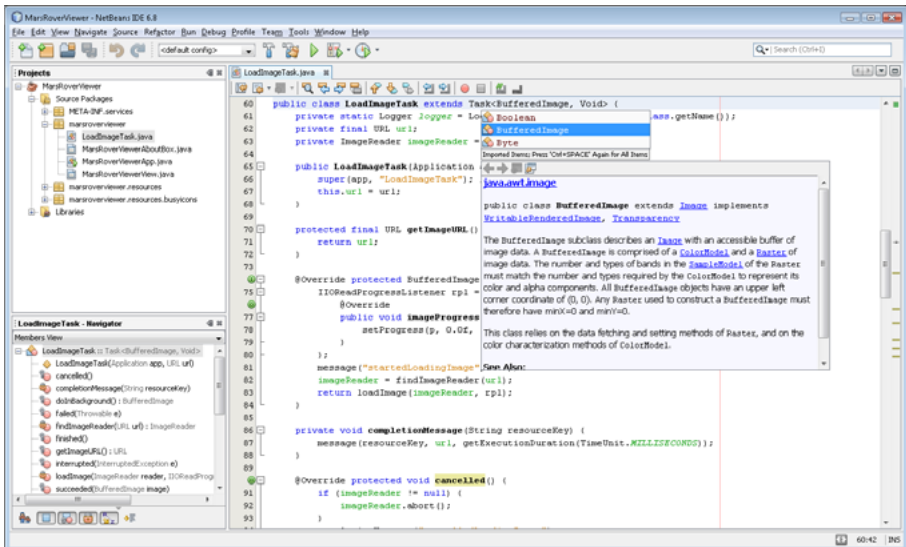
The most frequently used IDEs are: *NetBeans*, *Eclips*, *Intelej IDEA*.

Generally, many product companies dictate the programmer which IDE should he/she use in the project, so it would be great if you will have an idea and a little experience in each of the most popular IDEs.



# NetBeans

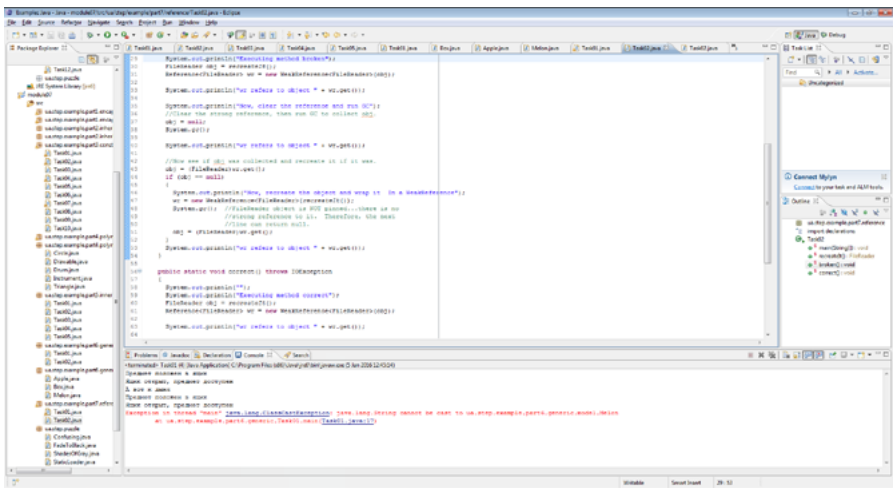
**NetBeans** is a freeware environment with an open source code. It supports a variety of programming and markup languages. It also supports most of the modern Java technologies. It has an interface in different languages and a large international community.



Official website <https://netbeans.org/>.



*IDE Eclipse* is a freeware environment for developing multi-module and cross-platform applications. It has an open source code. It is developed and maintained by the non-profit organization “Eclipse Foundation”. Initially, this environment belonged to IBM. It is a multi-layer constructor consisting of the “OSGI” and “SWT/JFace” service platforms. The functionality can be enhanced via a large number of plugins.

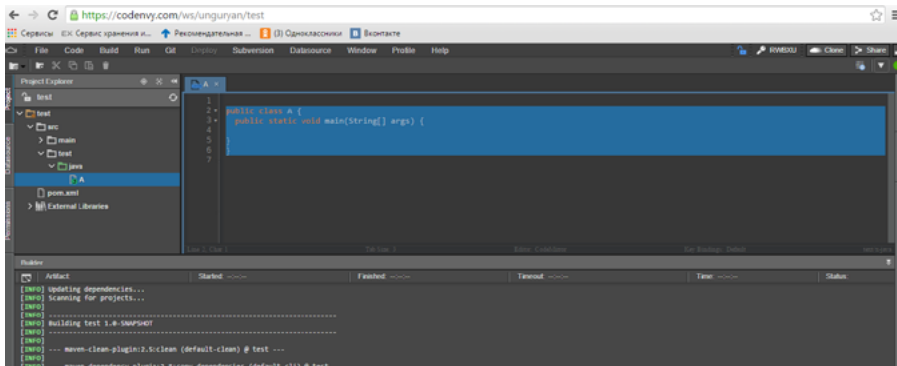


Official website <https://eclipse.org/>.

**IntelliJ IDEA** is a shareware intellectual development environment. It is developed by “Jet Brains”. A distinctive feature of this environment is the ability to understand context and a large number of supported technologies and frameworks right out of the box. There is a free version called Community (does not support JEE technologies), as well as the paid Ultimate version (30-days trial).

Official website <https://www.jetbrains.com/idea/>.

**Codenvy** is a freeware development environment that does not require installation on your computer. The development takes place in the browser after loading the required content in it.



Official website <https://codenvy.com/>.



# Lesson 1

## Introduction to Java

© Vitaliy Unguryan  
© STEP IT Academy  
[www.itstep.org](http://www.itstep.org)

All rights to protected pictures, audio, and video belong to their authors or legal owners. Fragments of works are used exclusively in illustration purposes to the extent justified by the purpose as part of an educational process and for educational purposes in accordance with Article 1273 Sec. 4 of the Civil Code of the Russian Federation and Articles 21 and 23 of the Law of Ukraine "On Copyright and Related Rights". The extent and method of cited works are in conformity with the standards, do not conflict with a normal exploitation of the work, and do not prejudice the legitimate interests of the authors and rightholders. Cited fragments of works can be replaced with alternative, non-protected analogs, and as such correspond the criteria of fair use.

All rights reserved. Any reproduction, in whole or in part, is prohibited. Agreement of the use of works and their fragments is carried out with the authors and other right owners. Materials from this document can be used only with resource link.

Liability for unauthorized copying and commercial use of materials is defined according to the current legislation of Ukraine.