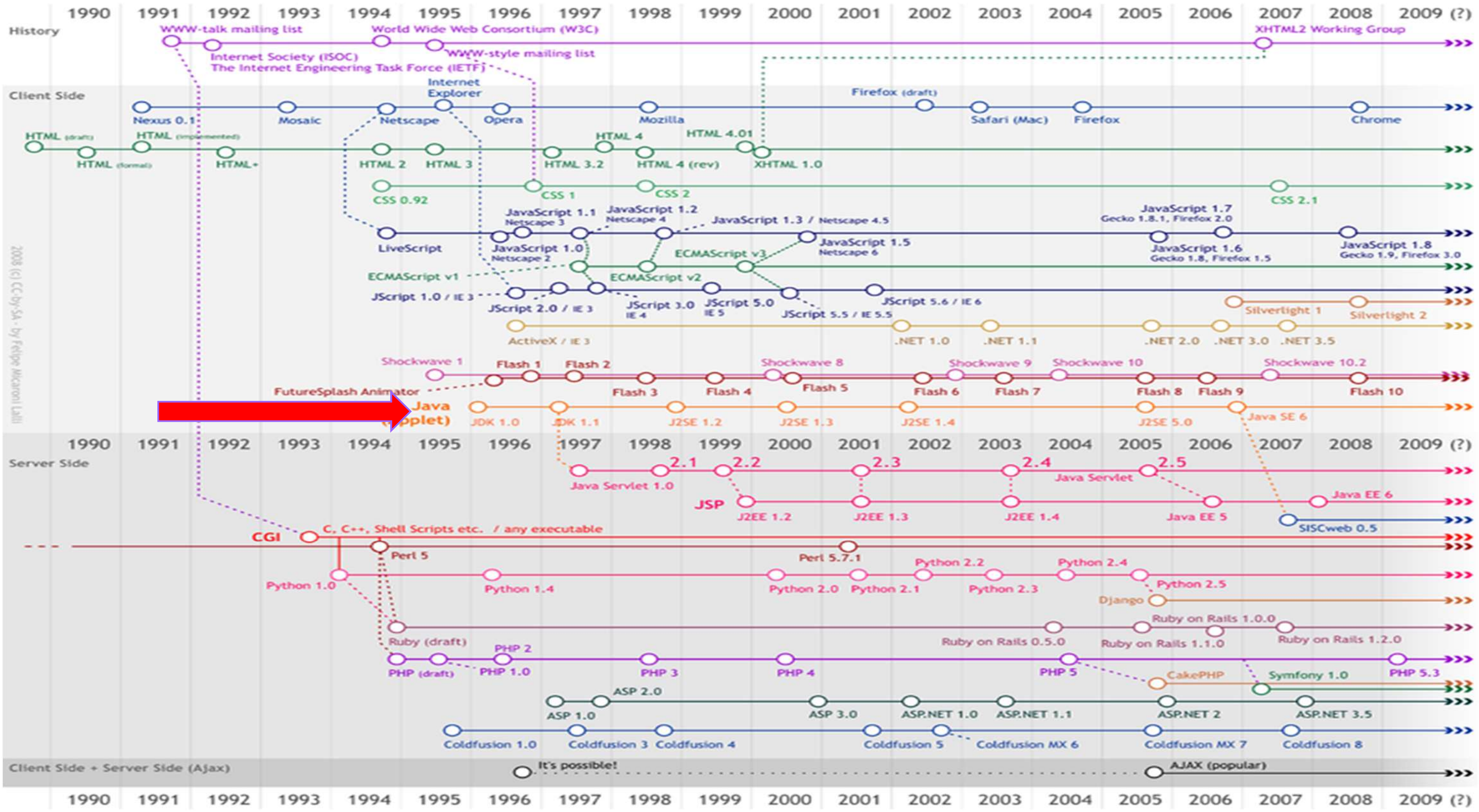
Java Programming Language Overview

Bok, Jong Soon
javaexpert@nate.com
https://github.com/swacademy/Core-Java

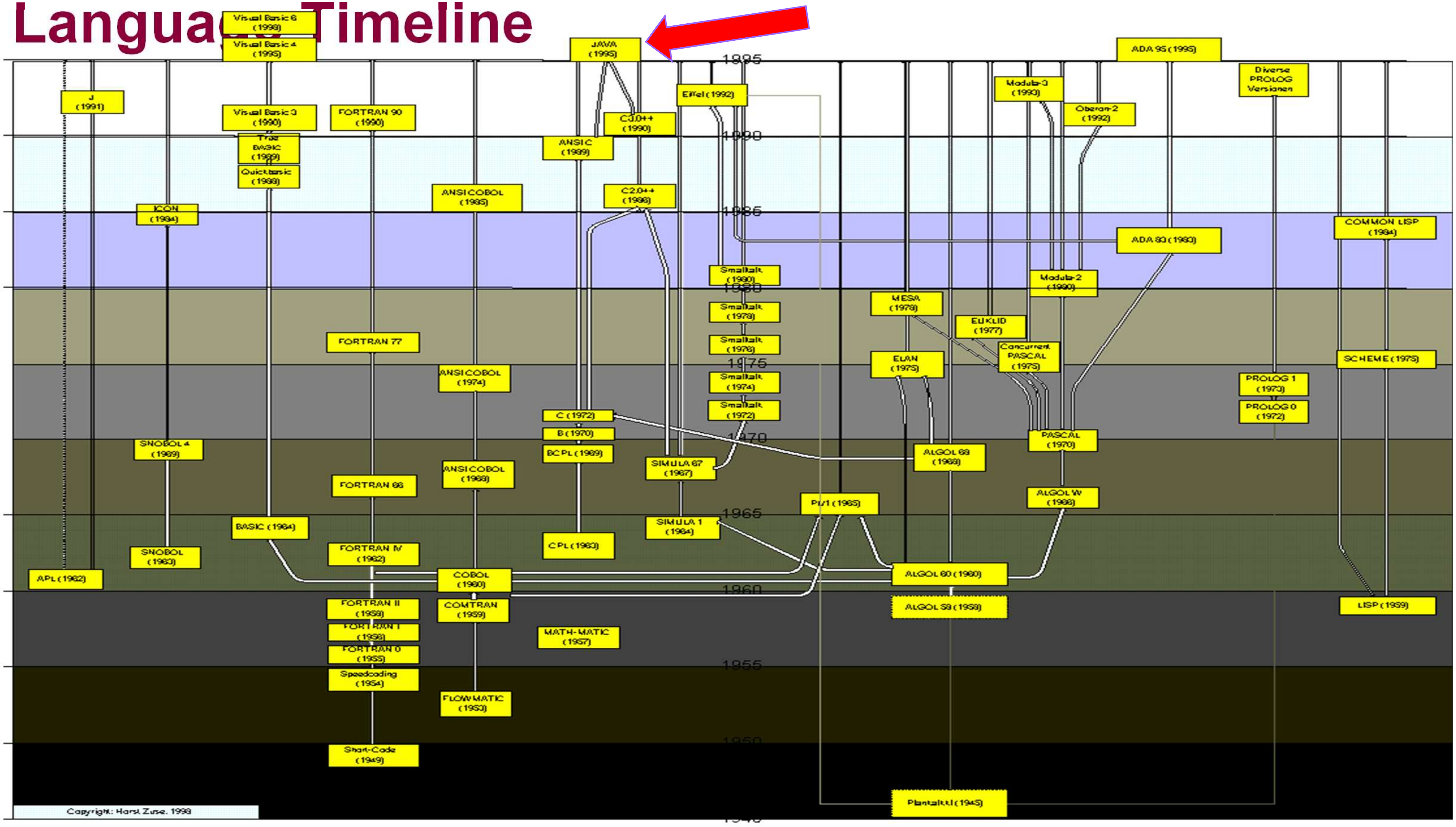
What is Java Technology?

- Is a programming language.
- Is a platform.

Web Timeline

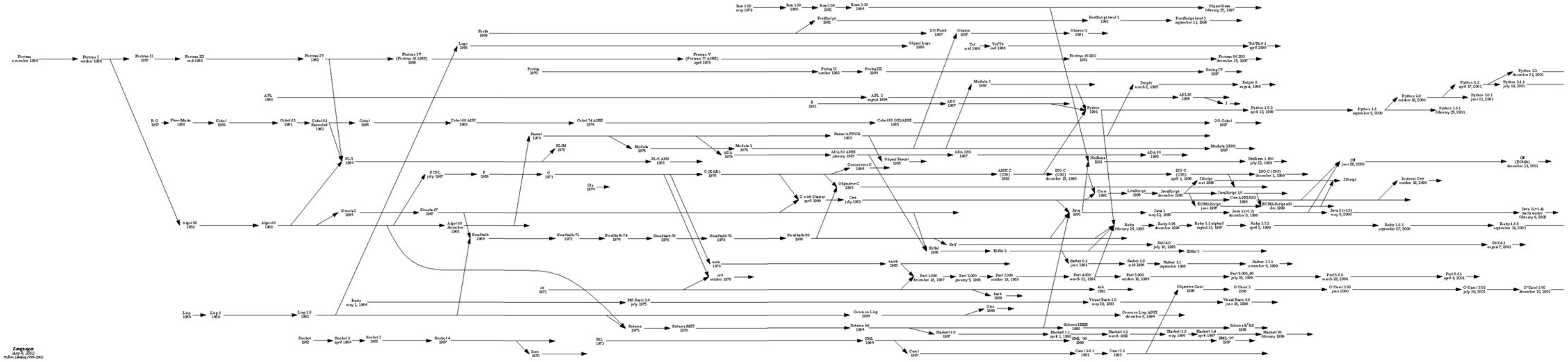


Language Timeline



Copyright: Horst Zuse, 1998

Language Timeline (Cont.)

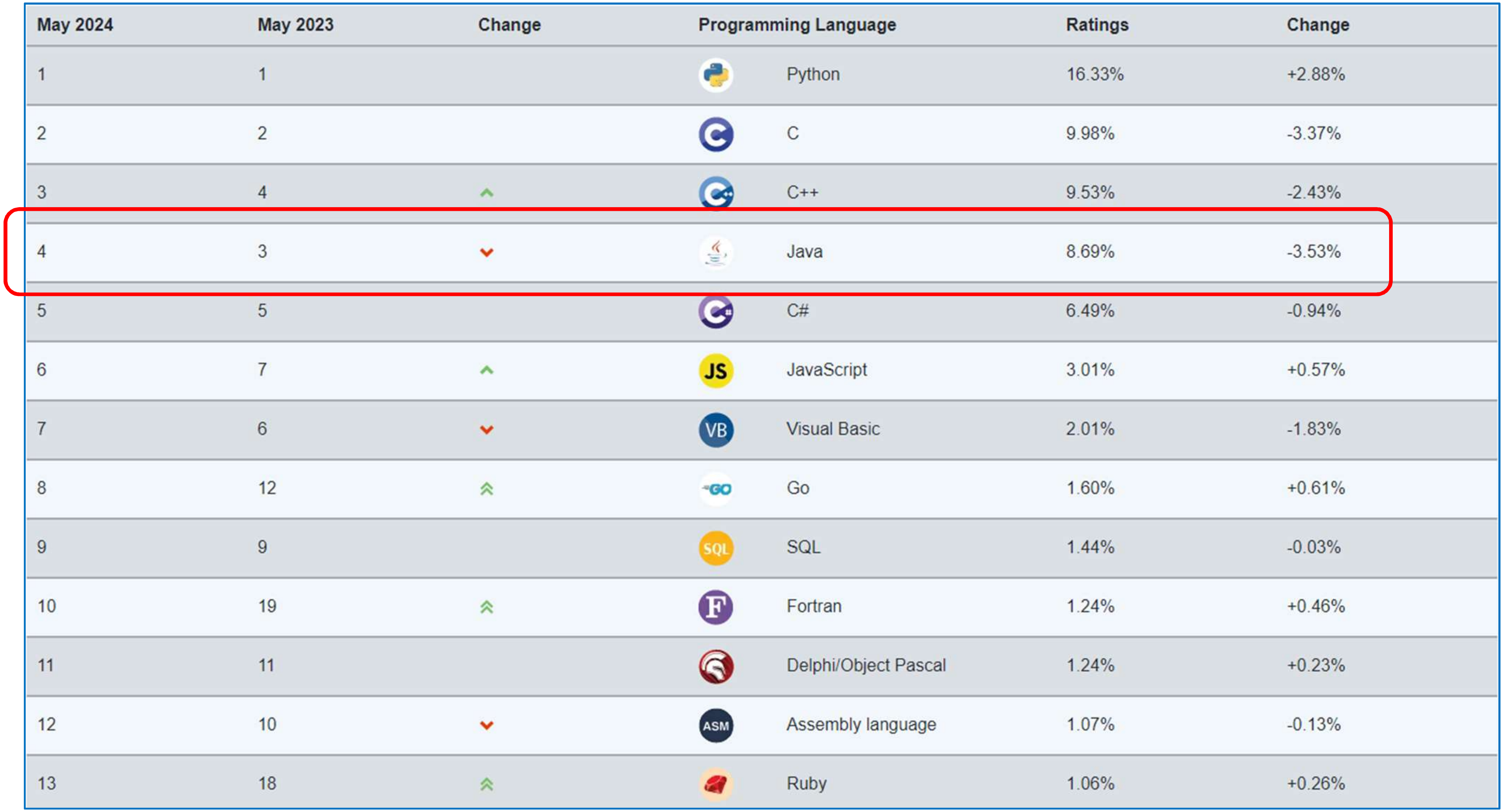


http://www.cs.umd.edu/class/spring2002/cmsc434-0101/MUIseum/applications/langtl2.html

Java Programming Language

- Is platform independent programming language.
- Similar to C++ in syntax.
- Similar to SmallTalk in mental paradigm.
- Is one of today's most popular software-development languages.
- Is used for Web programming
- Is used for developing standalone applications across platforms on servers, desktops, and mobile devices.
- Is a high-level language.

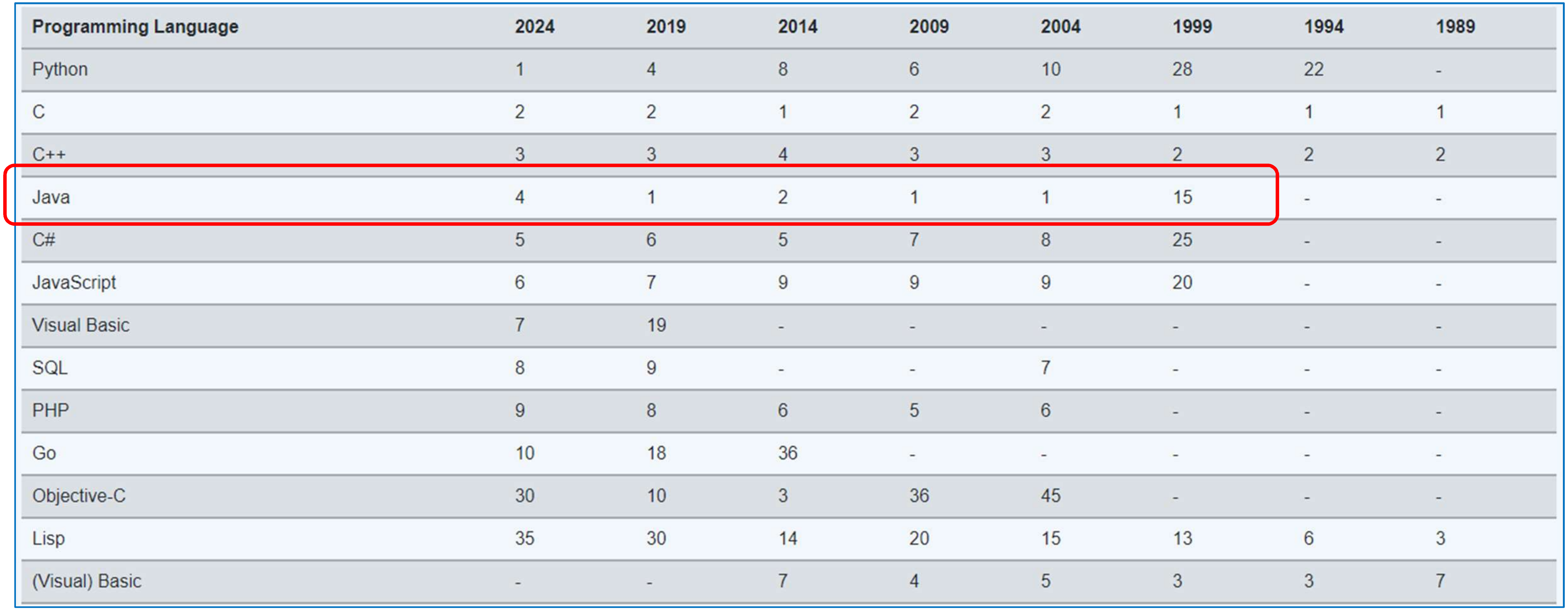
Java Programming Language (Cont.)



May 2024	May 2023	Change	Programming Language	Ratings	Change
1	1		Python	16.33%	+2.88%
2	2		C	9.98%	-3.37%
3	4	▲	C++	9.53%	-2.43%
4	3	▼	Java	8.69%	-3.53%
5	5		C#	6.49%	-0.94%
6	7	▲	JavaScript	3.01%	+0.57%
7	6	▼	Visual Basic	2.01%	-1.83%
8	12	▲▲	Go	1.60%	+0.61%
9	9		SQL	1.44%	-0.03%
10	19	▲▲	Fortran	1.24%	+0.46%
11	11		Delphi/Object Pascal	1.24%	+0.23%
12	10	▼	Assembly language	1.07%	-0.13%
13	18	▲▲	Ruby	1.06%	+0.26%

Source : https://www.tiobe.com/tiobe-index/

Java Programming Language (Cont.)

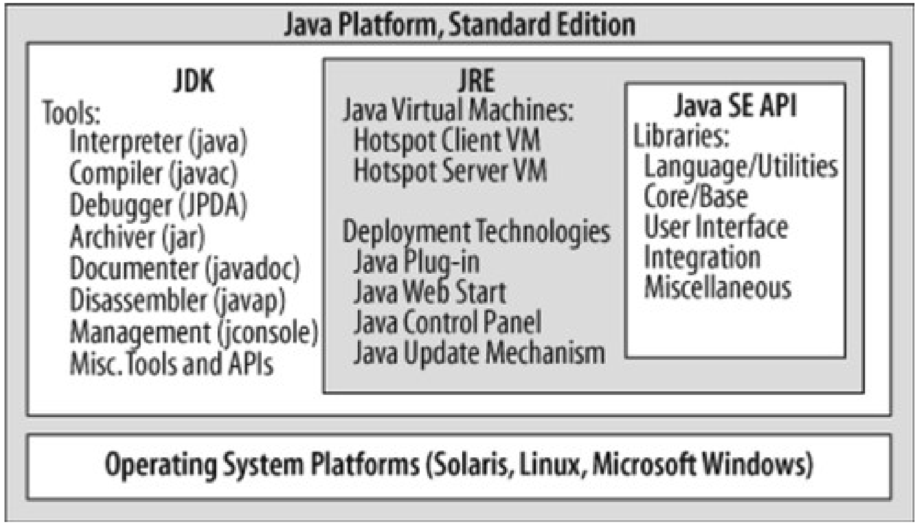


Programming Language	2024	2019	2014	2009	2004	1999	1994	1989
Python	1	4	8	6	10	28	22	-
C	2	2	1	2	2	1	1	1
C++	3	3	4	3	3	2	2	2
Java	4	1	2	1	1	15	-	-
C#	5	6	5	7	8	25	-	-
JavaScript	6	7	9	9	9	20	-	-
Visual Basic	7	19	-	-	-	-	-	-
SQL	8	9	-	-	7	-	-	-
PHP	9	8	6	5	6	-	-	-
Go	10	18	36	-	-	-	-	-
Objective-C	30	10	3	36	45	-	-	-
Lisp	35	30	14	20	15	13	6	3
(Visual) Basic	-	-	7	4	5	3	3	7

Source : https://www.tiobe.com/tiobe-index/

The Java Platform

- Platform : The hardware or software environment in which a program runs.
- Has two components:
  - The Java Virtual Machine
  - The Java Application Programming Interface (API)



Java Platform, Standard Edition

JDK

Tools:
Interpreter (java)
Compiler (javac)
Debugger (JPDA)
Archiver (jar)
Documenter (javadoc)
Disassembler (javap)
Management (jconsole)
Misc. Tools and APIs

JRE

Java Virtual Machines:
Hotspot Client VM
Hotspot Server VM

Deployment Technologies
Java Plug-in
Java Web Start
Java Control Panel
Java Update Mechanism

Java SE API

Libraries:
Language/Utilities
Core/Base
User Interface
Integration
Miscellaneous

Operating System Platforms (Solaris, Linux, Microsoft Windows)

From : [Java Pocket Guide], Robert Liguori ; Patricia Liguori, O'Reilly, 2008, 978-0-59-651419-8, p191

The Java Platform (Cont.) - JRE

- Java Runtime Environment
- Provides the backbone for running Java application.
- Is a collection of software.
- Allows a computer system to run a Java application.
- Consists of
  - JVMs, Java Virtual Machines, interpret Java ***bytecode*** into machine code.
  - Standard class libraries
  - User interface toolkits
  - A variety of utilities.

The Java Platform (Cont.) - JDK

- Java Development Kit
- Provides all of the components and necessary resources to develop Java applications.
- Is a programming environment for compiling, debugging, and running Java applets, applications, and Java Beans.
- Includes the JRE, Java Programming language, development tools and tool APIs.
- Refer to http://java-virtual-machine.net/other.html

The Java Platform (Cont.) - JDK

- Download the most recent version at https://www.oracle.com/java/technologies/
- Download older versions at https://www.oracle.com/java/technologies/downloads/archive/

Java Development Kits	Codename	Release
Java SE 8 with JDK 1.8.0	Spider	2014
Java SE 7 with JDK 1.7.0	Dolphin	2011
Java SE 6 with JDK 1.6.0	Mustang	2006
Java 2 SE 5.0 with JDK 1.5.0	Tiger	2004
Java 2 SE with SDK 1.4.0	Merlin	2002
Java 2 SE with SDK 1.3	Kestrel	2000
Java 2 with SDK 1.2	Playground	1998

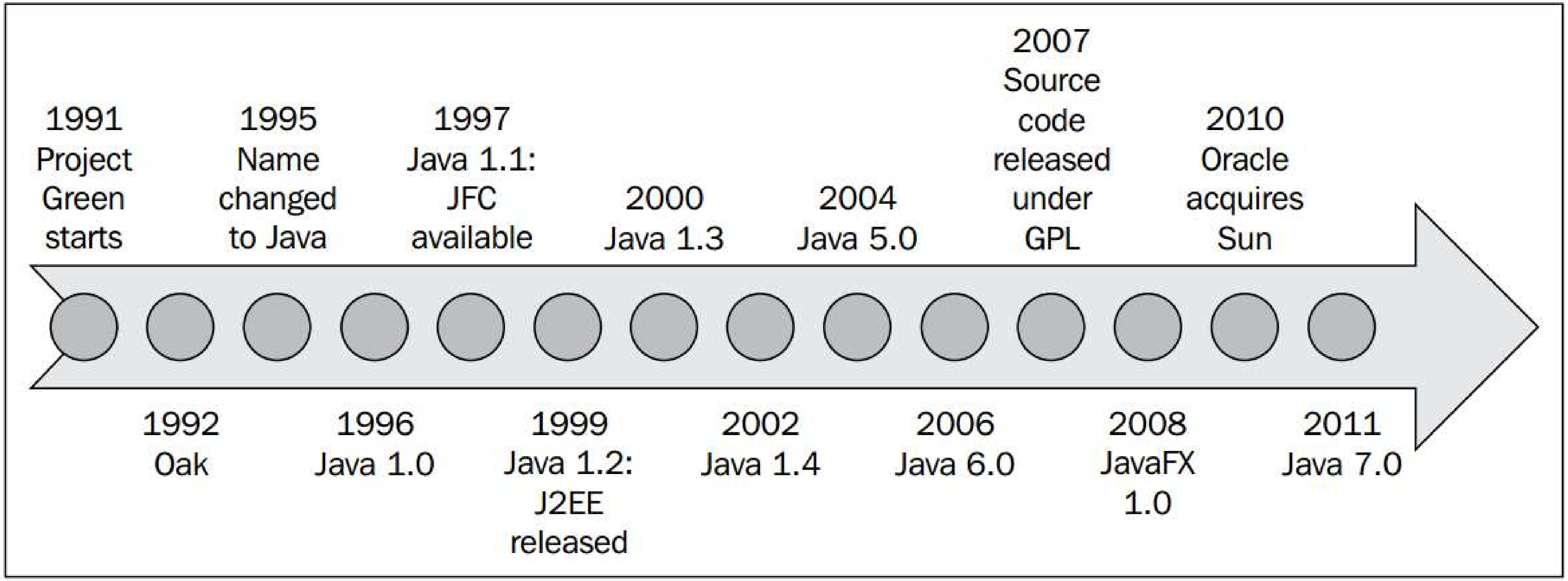
Java SE Code Names

Version	Code Name	Release Date
JDK 1.1.4	Sparkler	1997. 10. 11
JDK 1.1.5	Pumpkin	1997. 11. 03
JDK 1.1.6	Abigail	1998. 04. 24
JDK 1.1.7	Brutus	1998. 09. 28
JDK 1.1.8	Chelsea	1999. 04. 08
J2SE 1.2	Playground	1998. 11 04
J2SE 1.2.1	(none)	1999. 03. 30
J2SE 1.2.2	Cricket	1999. 07. 08
J2SE 1.3	Kestrel	2000. 08. 05
J2SE 1.3.1	Ladybird	2001. 05. 17
J2SE 1.4.0	Merlin	2002. 02. 13
J2SE 1.4.1	Hopper	2002. 09. 16
J2SE 1.4.2	Mantis	2003. 06. 26
Java SE 5.0(1.5.0)	Tiger	2004. 09. 29
Java SE 6.0(1.6.0)	Mustang	2005. 11. 20
Java SE 7.0(1.7.0)	Dolphin	2011. 07. 28
Java SE 8.0(1.8.0)	Spider	2014. 03. 18

History

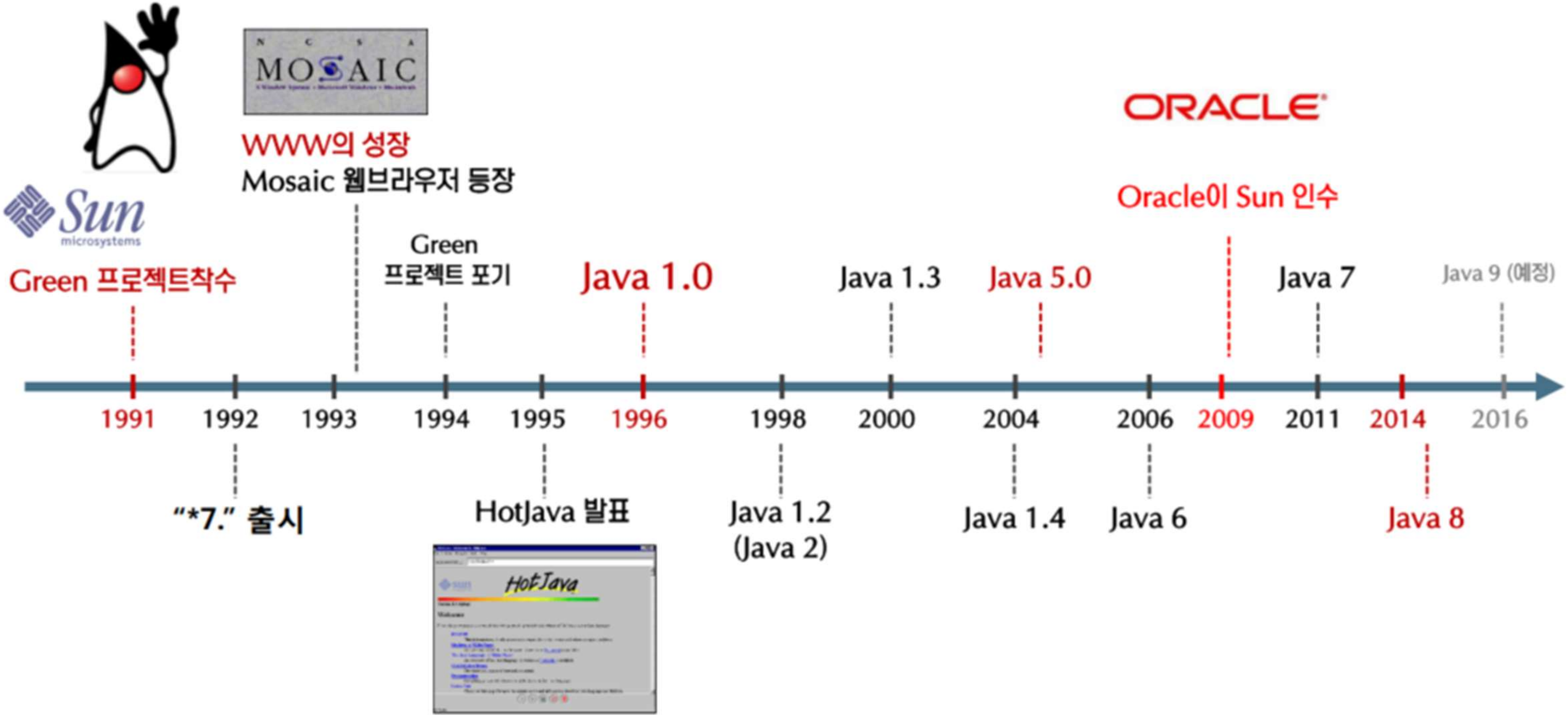
- Originally named ***Oak***, designed in 1991.
- Main team members : Bill Joy, Patrick Naughton, Mike Sheridan, James Gosling.
- Original goal : use in embedded consumer electronic appliances.
- In 1994, team realized Oak was perfect for ***Internet***.
- In 1995, renamed Java, was redesigned for developing Internet applications.
- Announced ***in May 23 in 1995*** at SunWorld'95.
- First non-beta release January 23 in 1996.
- Refer to http://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html

History (Cont.)


1991 Project Green starts
1992 Oak
1995 Name changed to Java
1996 Java 1.0
1997 Java 1.1: JFC available
1999 Java 1.2: J2EE released
2000 Java 1.3
2002 Java 1.4
2004 Java 5.0
2006 Java 6.0
2007 Source code released under GPL
2008 JavaFX 1.0
2010 Oracle acquires Sun
2011 Java 7.0

Source : Oracle Certified Associate Java SE 7 Programmer Study Guide 2012, PACKT Press, p8

History (Cont.)



1991 – Green 프로젝트착수

1992 – "*7." 출시

1993 – WWW의 성장
Mosaic 웹브라우저 등장

1994 – Green
프로젝트 포기

1995 – HotJava 발표

1996 – Java 1.0

1998 – Java 1.2
(Java 2)

2000 – Java 1.3

2004 – Java 5.0

2004 – Java 1.4

2006 – Java 6

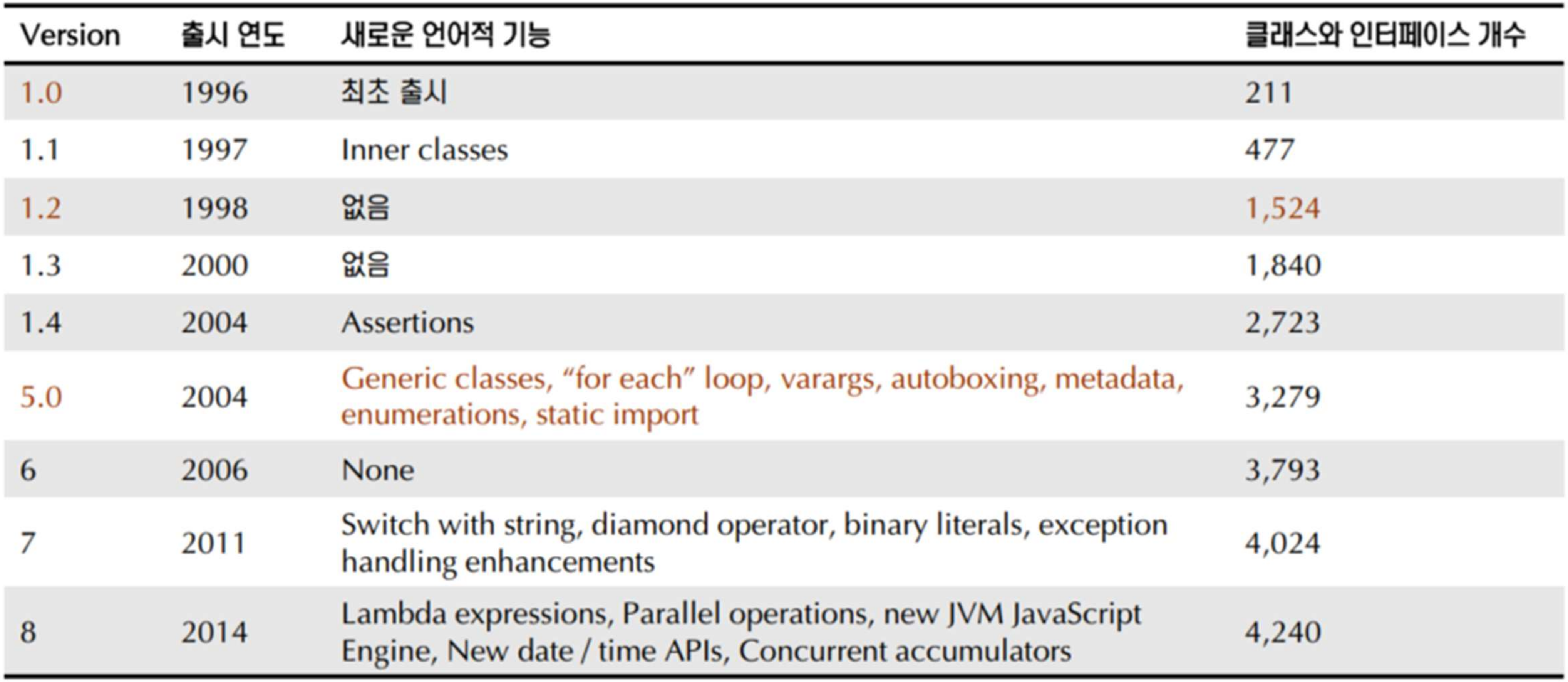
2009 – Oracle이 Sun 인수

2011 – Java 7

2014 – Java 8

2016 – Java 9 (예정)

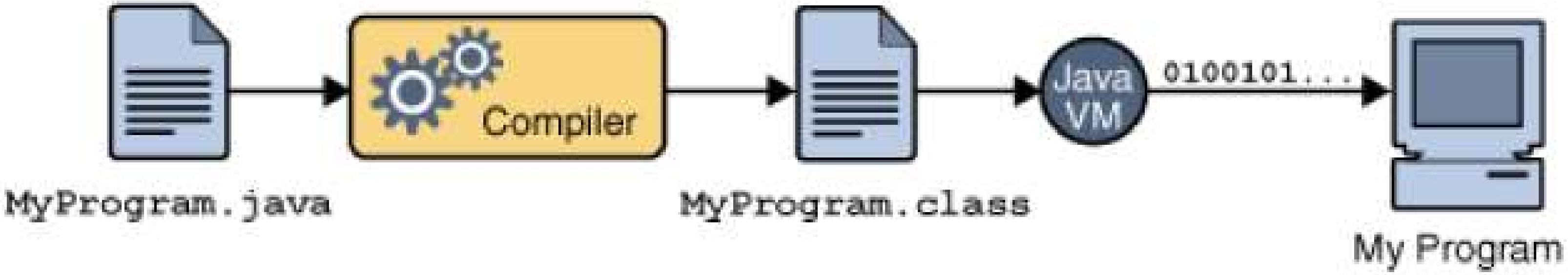
History (Cont.)



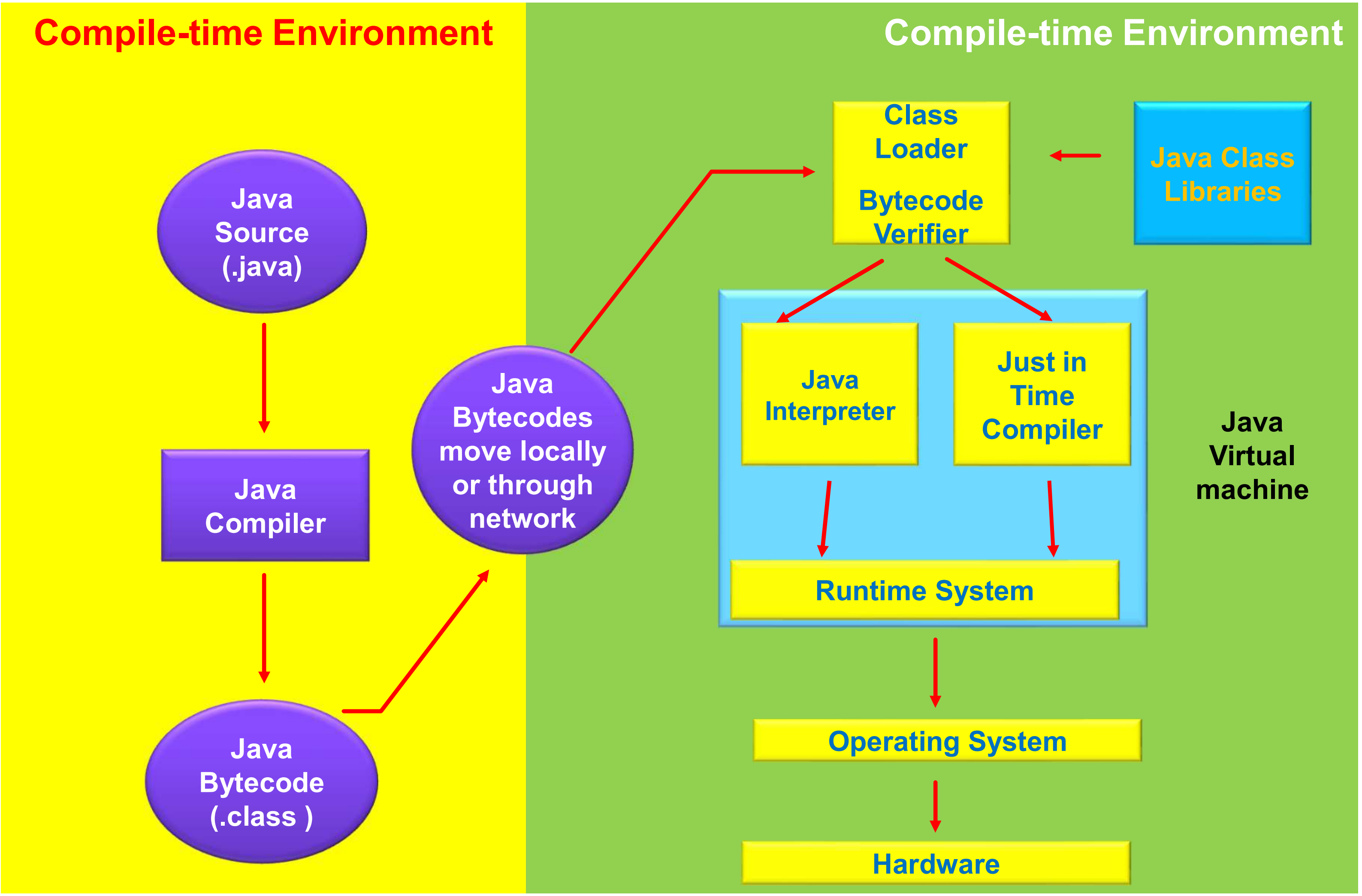
Version	출시 연도	새로운 언어적 기능	클래스와 인터페이스 개수
1.0	1996	최초 출시	211
1.1	1997	Inner classes	477
1.2	1998	없음	1,524
1.3	2000	없음	1,840
1.4	2004	Assertions	2,723
5.0	2004	Generic classes, “for each” loop, varargs, autoboxing, metadata, enumerations, static import	3,279
6	2006	None	3,793
7	2011	Switch with string, diamond operator, binary literals, exception handling enhancements	4,024
8	2014	Lambda expressions, Parallel operations, new JVM JavaScript Engine, New date / time APIs, Concurrent accumulators	4,240

Features

- Simple
- Object-Oriented
- Distributed
- Multithreaded
- Dynamic
- Architecture neutral
- Portable
- High performance
- Robust
- Secure
- Write Once, Run Anywhere™
- http://java.sun.com/docs/white/langenv/



How it works…!



Source : Java tutorial PPT, by Intelligo Technologies on Mar 08, 2011

Figure 1.1 J2SE vs. J2EE vs. J2ME

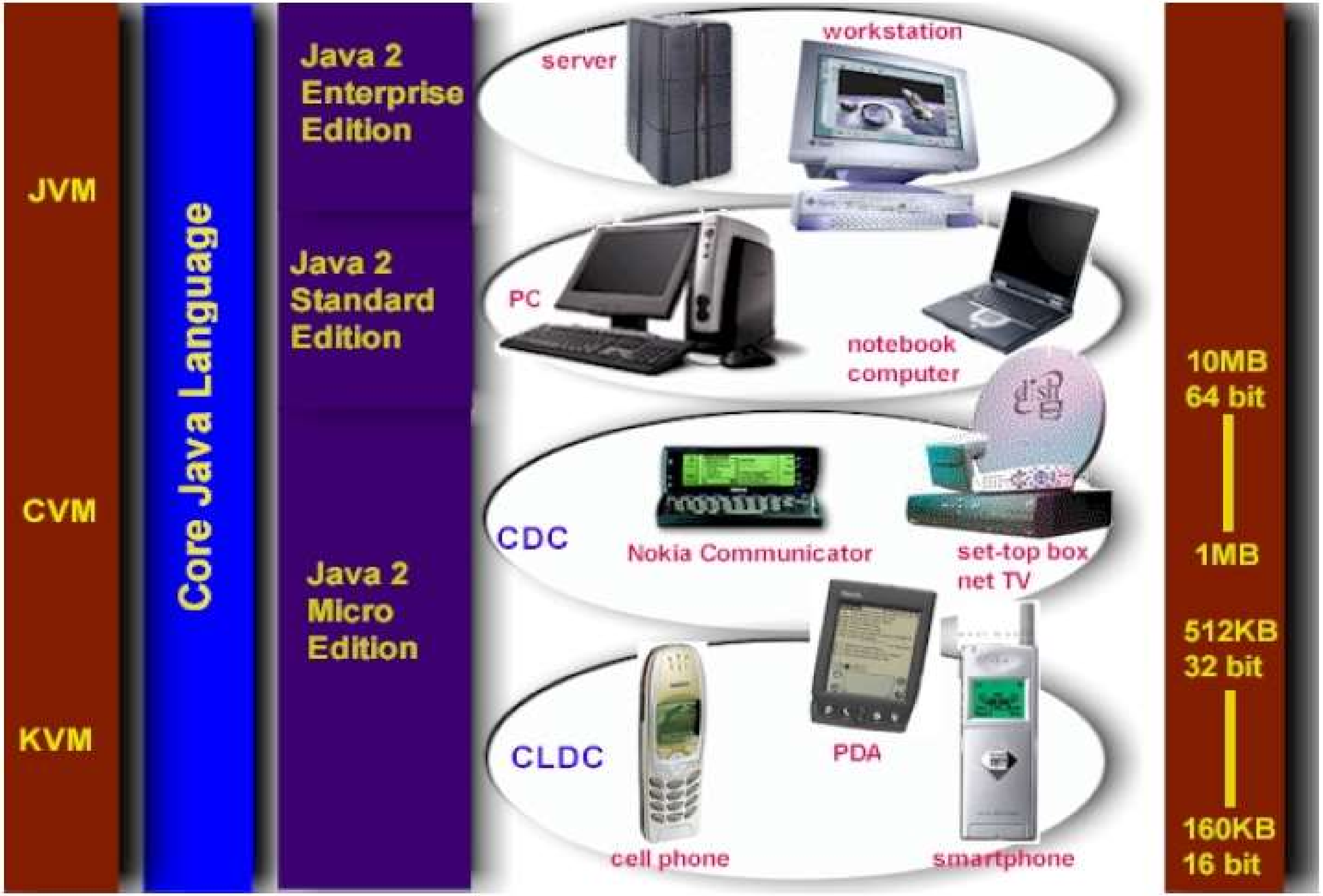
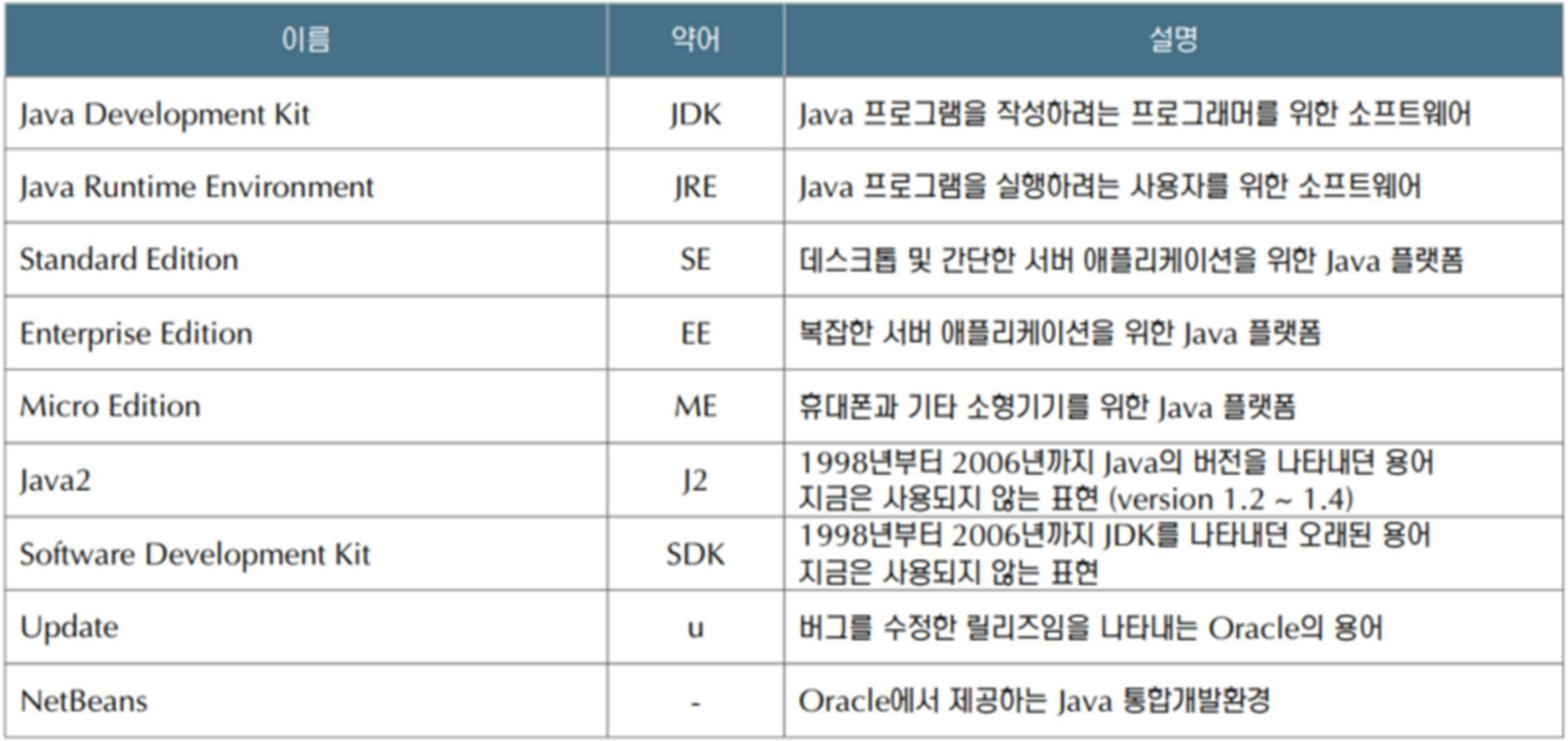
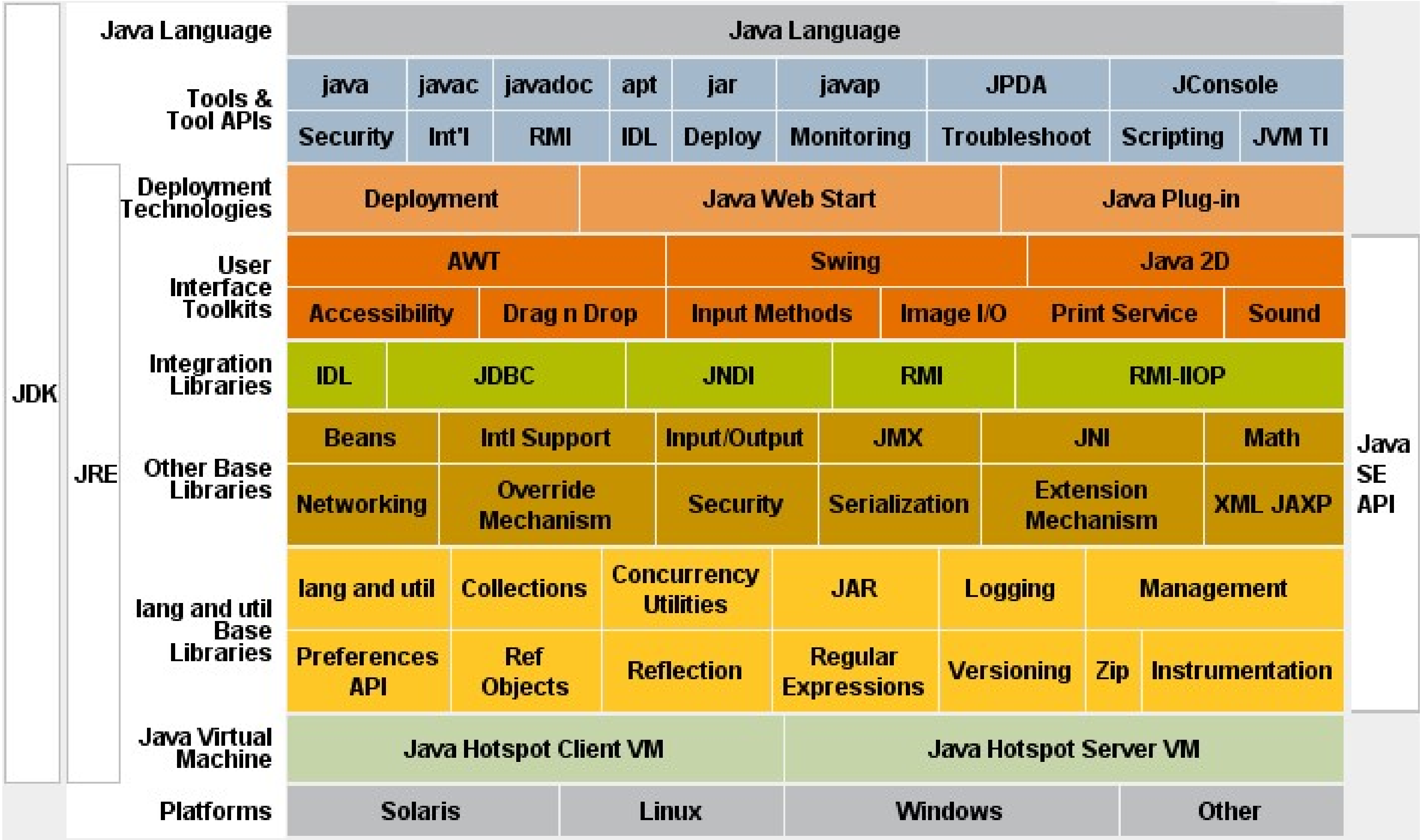


Figure 1.1 J2SE vs. J2EE vs. J2ME



이름	약어	설명
Java Development Kit	JDK	Java 프로그램을 작성하려는 프로그래머를 위한 소프트웨어
Java Runtime Environment	JRE	Java 프로그램을 실행하려는 사용자를 위한 소프트웨어
Standard Edition	SE	데스크톱 및 간단한 서버 애플리케이션을 위한 Java 플랫폼
Enterprise Edition	EE	복잡한 서버 애플리케이션을 위한 Java 플랫폼
Micro Edition	ME	휴대폰과 기타 소형기기를 위한 Java 플랫폼
Java2	J2	1998년부터 2006년까지 Java의 버전을 나타내던 용어 지금은 사용되지 않는 표현 (version 1.2 ~ 1.4)
Software Development Kit	SDK	1998년부터 2006년까지 JDK를 나타내던 오래된 용어 지금은 사용되지 않는 표현
Update	u	버그를 수정한 릴리즈임을 나타내는 Oracle의 용어
NetBeans	-	Oracle에서 제공하는 Java 통합개발환경

Figure 1.2 Java SE 6 Platform at a Glance



JDK

JRE

Java SE API

Java Language: Java Language

Tools & Tool APIs: java, javac, javadoc, apt, jar, javap, JPDA, JConsole

Security, Int'l, RMI, IDL, Deploy, Monitoring, Troubleshoot, Scripting, JVM TI

Deployment Technologies: Deployment, Java Web Start, Java Plug-in

User Interface Toolkits: AWT, Swing, Java 2D

Accessibility, Drag n Drop, Input Methods, Image I/O, Print Service, Sound

Integration Libraries: IDL, JDBC, JNDI, RMI, RMI-IIOP

Other Base Libraries: Beans, Intl Support, Input/Output, JMX, JNI, Math

Networking, Override Mechanism, Security, Serialization, Extension Mechanism, XML JAXP

lang and util Base Libraries: lang and util, Collections, Concurrency Utilities, JAR, Logging, Management

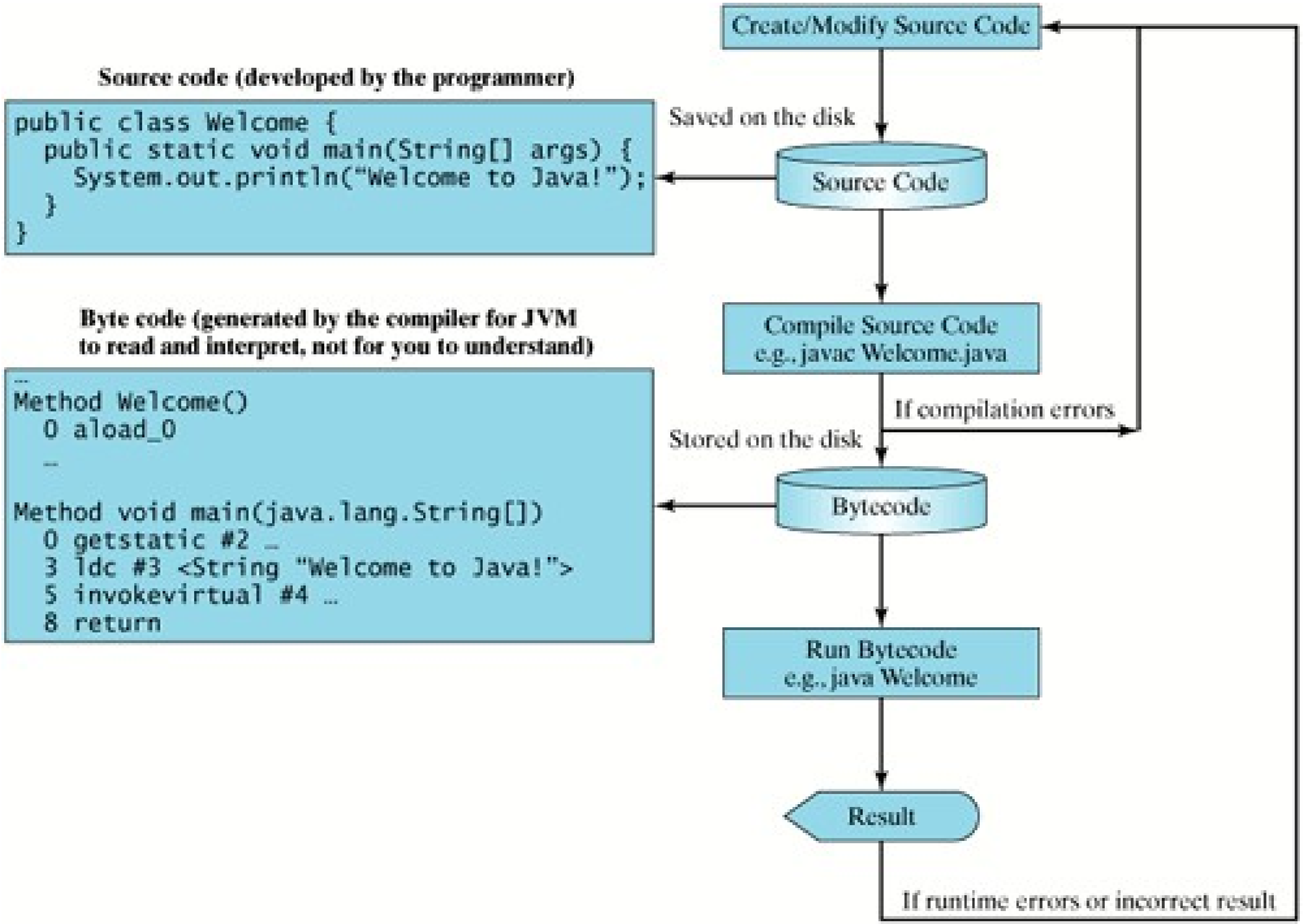
Preferences API, Ref Objects, Reflection, Regular Expressions, Versioning, Zip, Instrumentation

Java Virtual Machine: Java Hotspot Client VM, Java Hotspot Server VM

Platforms: Solaris, Linux, Windows, Other

※ http://java.sun.com/javase/technologies/index.jsp

Development Process



Source code (developed by the programmer)

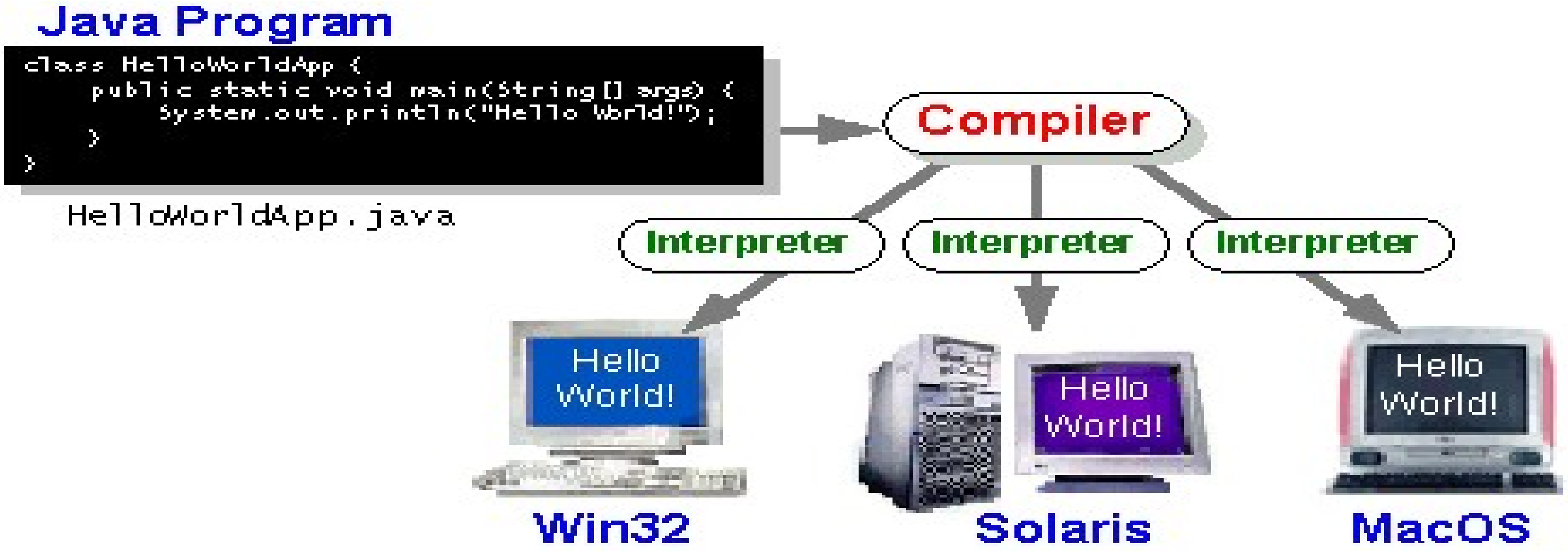
```
public class Welcome {
  public static void main(String[] args) {
    System.out.println(“Welcome to Java!”);
  }
}
```

Byte code (generated by the compiler for JVM to read and interpret, not for you to understand)

```
…
Method Welcome()
  0 aload_0
  …

Method void main(java.lang.String[])
  0 getstatic #2 …
  3 ldc #3 <String "Welcome to Java!">
  5 invokevirtual #4 …
  8 return
```

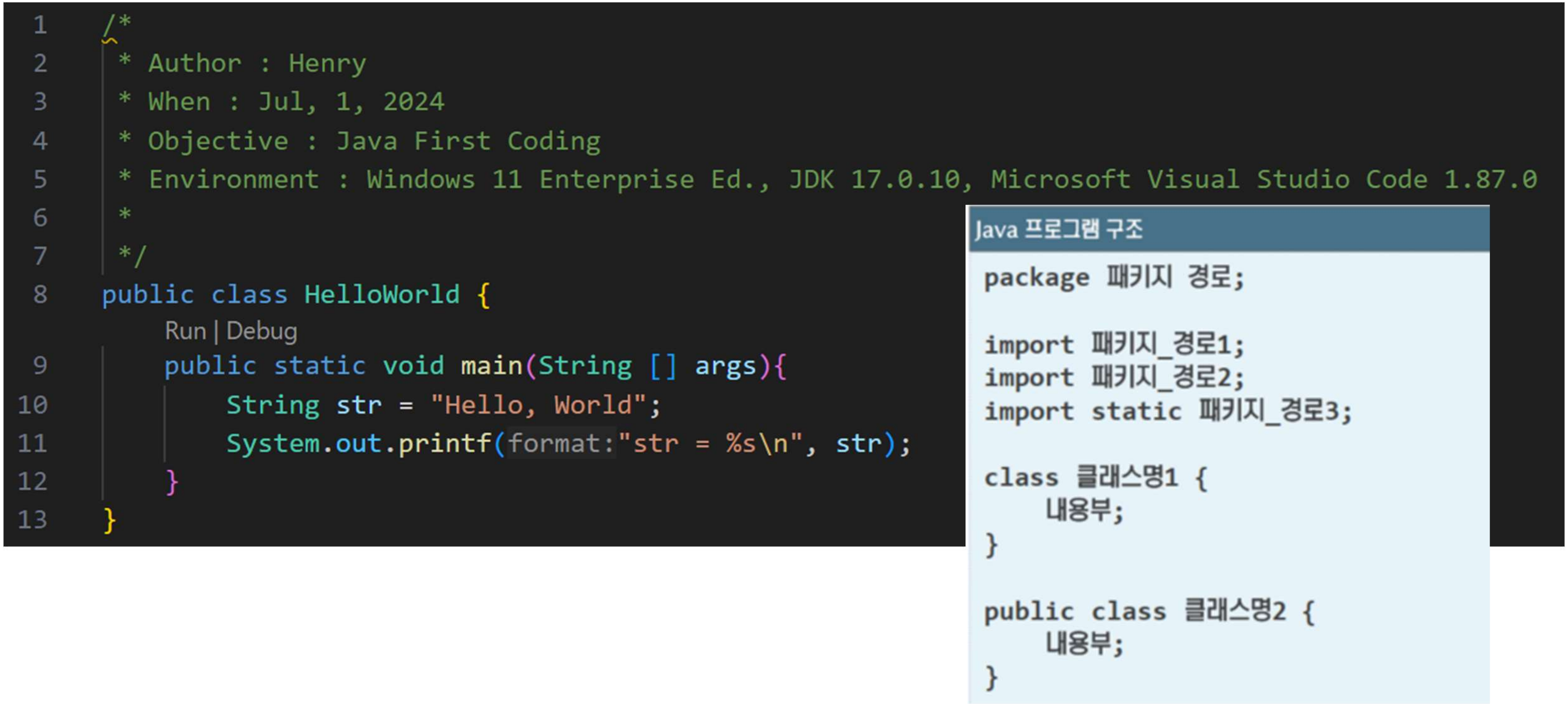
Development Process (Cont.)

1. Create a ***source file***
2. Compile the source file into a ***bytecode*** file
3. Run the program contained in the ***bytecode*** file



```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

Creating a Source Code – HelloWorld.java



```
/*
 * Author : Henry
 * When : Jul, 1, 2024
 * Objective : Java First Coding
 * Environment : Windows 11 Enterprise Ed., JDK 17.0.10, Microsoft Visual Studio Code 1.87.0
 *
 */
public class HelloWorld {
    public static void main(String [] args){
        String str = "Hello, World";
        System.out.printf(format:"str = %s\n", str);
    }
}
```

Java 프로그램 구조

```
package 패키지 경로;

import 패키지_경로1;
import 패키지_경로2;
import static 패키지_경로3;

class 클래스명1 {
    내용부;
}

public class 클래스명2 {
    내용부;
}
```

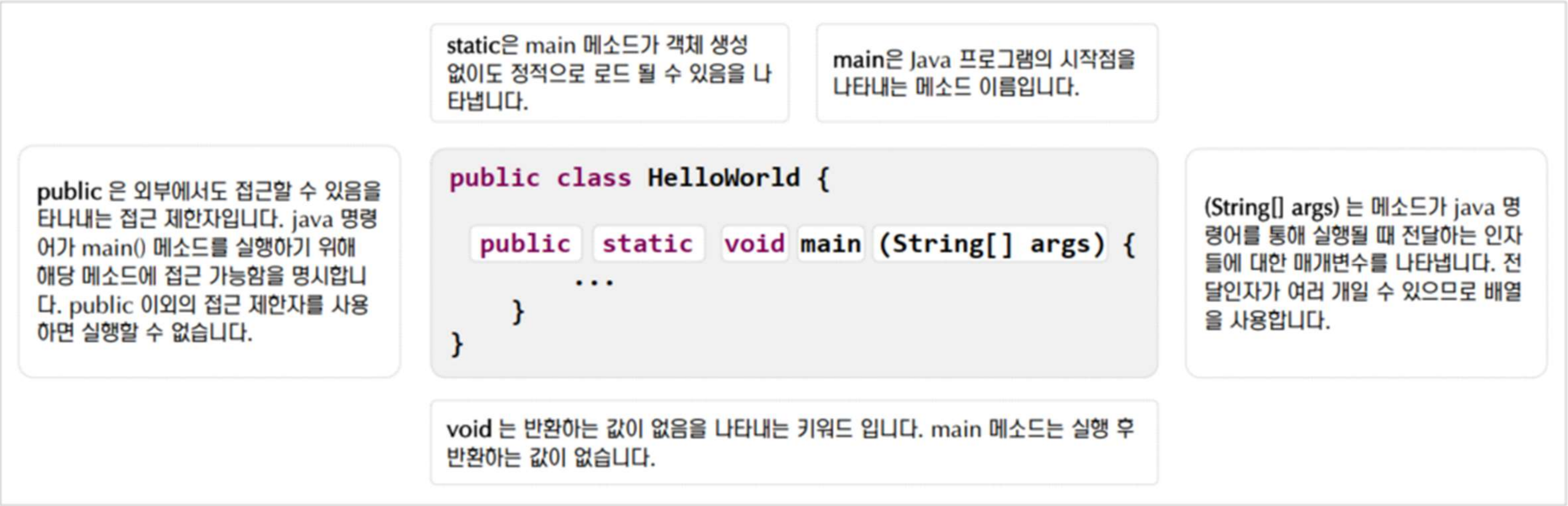
Creating a Source Code – HelloWorld.java

✓ Java 클래스의 main() 메소드는 java 명령어를 통해 처음 실행되는 메소드(method)입니다.

✓ 예를 들어, java HelloWorld 를 실행하면, HelloWorld 클래스의 main 메소드가 실행됩니다.

✓ main() 메소드는 객체를 생성하지 않고도, 외부에서도 접근할 수 있어야 합니다.

✓ main() 메소드는 반환 값이 없으며, java 명령어의 전달인자를 받는 매개변수(parameter)를 갖습니다.



static은 main 메소드가 객체 생성 없이도 정적으로 로드 될 수 있음을 나타냅니다.

main은 Java 프로그램의 시작점을 나타내는 메소드 이름입니다.

public 은 외부에서도 접근할 수 있음을 타나내는 접근 제한자입니다. java 명령어가 main() 메소드를 실행하기 위해 해당 메소드에 접근 가능함을 명시합니다. public 이외의 접근 제한자를 사용하면 실행할 수 없습니다.

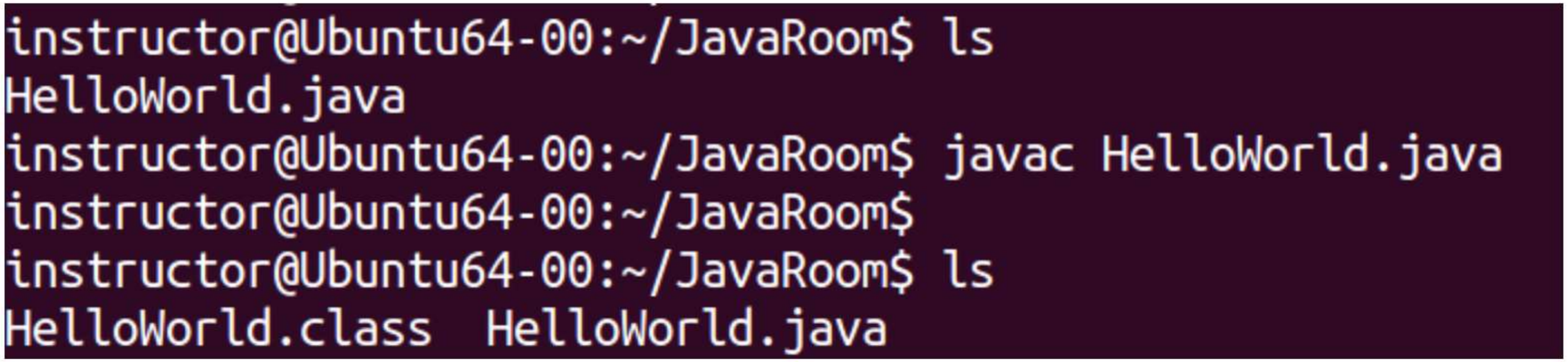
```
public class HelloWorld {

  public static void main (String[] args) {
        ...
    }
}
```

(String[] args) 는 메소드가 java 명령어를 통해 실행될 때 전달하는 인자들에 대한 매개변수를 나타냅니다. 전달인자가 여러 개일 수 있으므로 배열을 사용합니다.

void 는 반환하는 값이 없음을 나타내는 키워드 입니다. main 메소드는 실행 후 반환하는 값이 없습니다.

Compiling the Source Code – HelloWorld.java

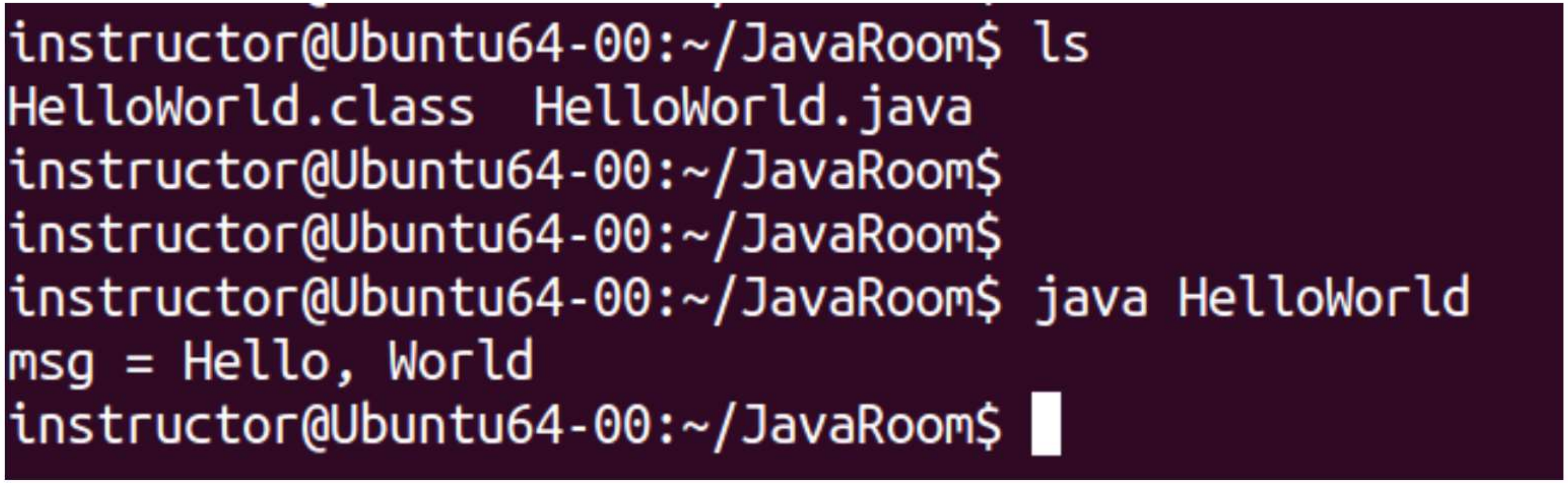
- Java Compiler – `javac.exe`



```
instructor@Ubuntu64-00:~/JavaRoom$ ls
HelloWorld.java
instructor@Ubuntu64-00:~/JavaRoom$ javac HelloWorld.java
instructor@Ubuntu64-00:~/JavaRoom$
instructor@Ubuntu64-00:~/JavaRoom$ ls
HelloWorld.class  HelloWorld.java
```

Interpreting the *bytecode* – HelloWorld.class

- Java Interpreter – **`java.exe`**



```
instructor@Ubuntu64-00:~/JavaRoom$ ls
HelloWorld.class  HelloWorld.java
instructor@Ubuntu64-00:~/JavaRoom$
instructor@Ubuntu64-00:~/JavaRoom$
instructor@Ubuntu64-00:~/JavaRoom$ java HelloWorld
msg = Hello, World
instructor@Ubuntu64-00:~/JavaRoom$ █
```

Command Line Tools

- JDK provides several command-line tools.
- Commonly used tools is a compiler, launcher/interpreter, archiver, documenter.
- Refer to https://docs.oracle.com/en/java/javase/17/docs/specs/man/index.html

Command Line Tools - Compiler

- Translates Java source files into Java ***bytecode***.
- Creates a ***bytecode*** file with the same name as the source file but with the `.class` extension.
- `javac [-options] [source files]`
  - `javac` HelloWorld.java
  - `javac –cp` ./dir/classes/ HelloWorld.java
  - `javac –d` ./opt/hwapp/classes HelloWorld.java
  - `javac –source` 1.4 HelloWorld.java
  - `javac –version`
  - `javac –help`
- Refer to https://bluemond.tistory.com/entry/22

Command Line Tools - Interpreter

- Handles the program execution, including launching the application.
- **`java [-options] class [arguments…] or java [-options] -jar jarfile [arguments…]`**
  - **`java`** HelloWorld
  - **`java -cp`** .:./dir/Classes HelloWorld
  - **`java -ea`** HelloWorld
  - **`java -version`**
  - **`java -help`**
  - **`javaw <classname>`**
- Refer to https://bluemond.tistory.com/entry/23

Command Line Tools - Packager

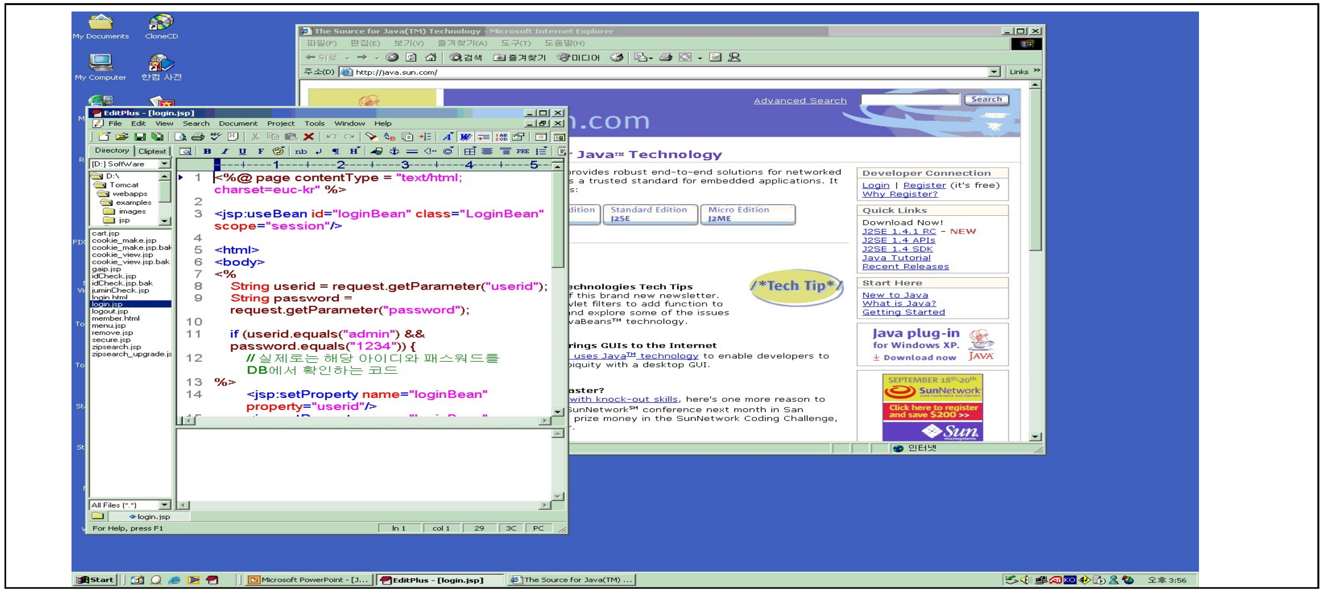
- JAR, Java Archive, utility is an archiving and compression tool.
- Used to combine multiple files into a single file called a JAR file.
- JAR consists of a ZIP archive containing a manifest file (JAR content describer) and optional signature files (for security).
- **`jar [options] [jar-file] [manifest-files] [entry-point] [-C dir] files…`**
  - **`jar cf`** files.jar HelloWorld.java kr/co/javaexpert/HelloWorld.class
  - **`jar tfv`** files.jar
  - **`jar xf`** files.jar
- Refer to https://bluemond.tistory.com/entry/24

Command Line Tools – JAR Execution

- Can be created to be executable.
- Specifies the file within the JAR where the **main** class resides.
- Refer to https://bluemond.tistory.com/entry/25

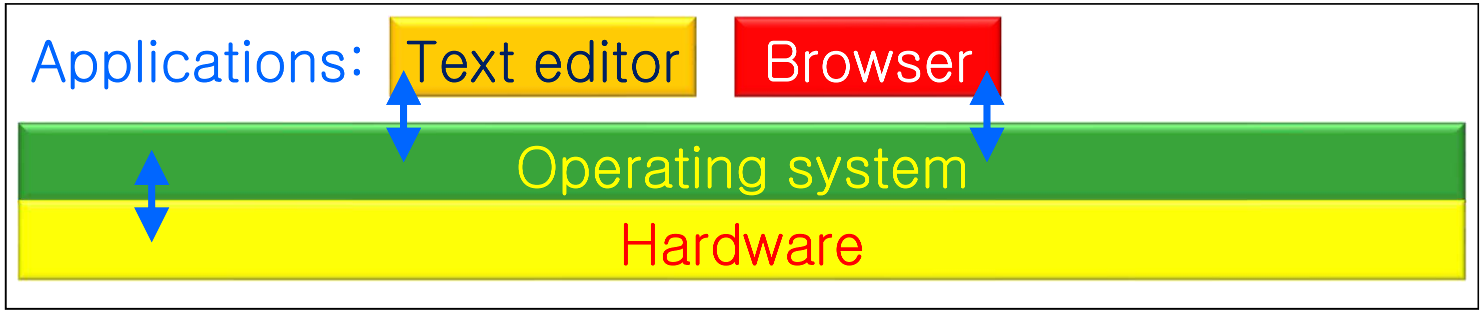
1. Compile **.java** file with package option.
2. Create a file **Manifest.txt** using editor.
3. Create a JAR file that adds the Manifest.txt contents to the manifest file, **MANIFEST.MF**.
4. Display the contents of the JAR file.
5. Execute the JAR file using **java –jar** option.

Figure 1.3. shows an examples of this basic communication.



EditPlus - [login.jsp]
File Edit View Search Document Project Tools Window Help
The Source for Java(TM) Technology - Microsoft Internet Explorer
http://java.sun.com/
Java™ Technology
Start
Microsoft PowerPoint - [D...
EditPlus - [login.jsp]
The Source for Java(TM) ...

```
<%@ page contentType = "text/html; charset=euc-kr" %>

<jsp:useBean id="loginBean" class="LoginBean" scope="session"/>

<html>
<body>
<%
   String userid = request.getParameter("userid");
   String password = request.getParameter("password");

   if (userid.equals("admin") && password.equals("1234")) {
      // 실제로는 해당 아이디와 패스워드를 DB에서 확인하는 코드
%>
      <jsp:setProperty name="loginBean" property="userid"/>
```

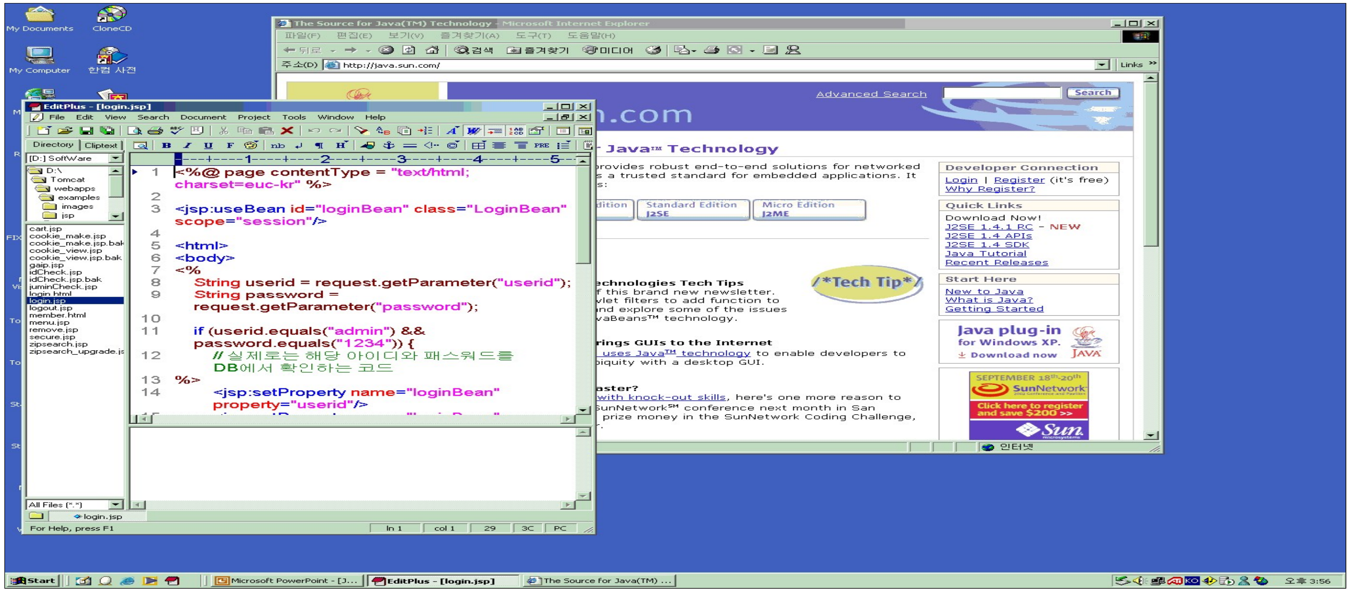


Applications: Text editor Browser

Operating system

Hardware

Figure 1.4. Computer Communication Problem



```
<%@ page contentType = "text/html; charset=euc-kr" %>

<jsp:useBean id="loginBean" class="LoginBean" scope="session"/>

<html>
<body>
<%
   String userid = request.getParameter("userid");
   String password = request.getParameter("password");

   if (userid.equals("admin") && password.equals("1234")) {
      // 실제로는 해당 아이디와 패스워드를 DB에서 확인하는 코드
%>
      <jsp:setProperty name="loginBean" property="userid"/>
```

n.com

Java™ Technology

provides robust end-to-end solutions for networked
s a trusted standard for embedded applications. It
s:

dition | Standard Edition J2SE | Micro Edition J2ME

echnologies Tech Tips
f this brand new newsletter.
vlet filters to add function to
nd explore some of the issues
vaBeans™ technology.

/*Tech Tip*/

rings GUIs to the Internet
uses Java™ technology to enable developers to
biquity with a desktop GUI.

aster?
with knock-out skills, here's one more reason to
SunNetwork℠ conference next month in San
prize money in the SunNetwork Coding Challenge,

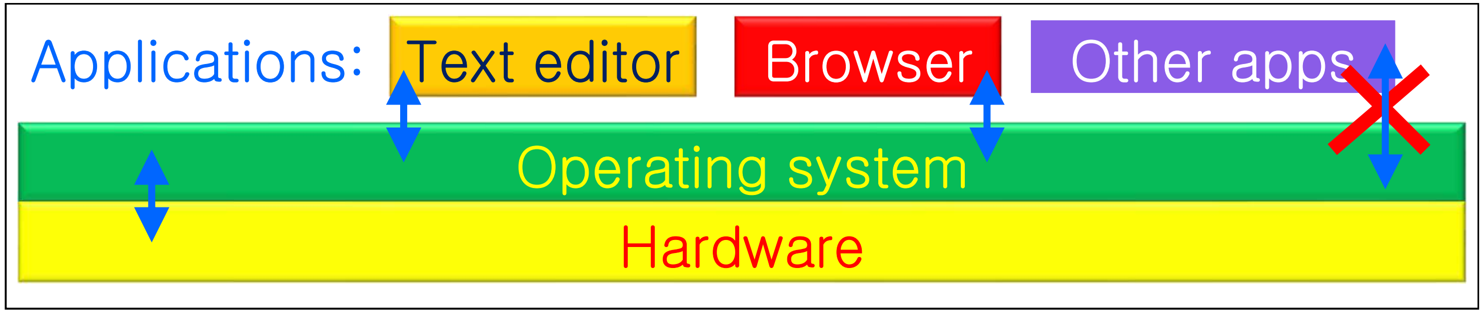
Developer Connection
Login | Register (it's free)
Why Register?

Quick Links
Download Now!
J2SE 1.4.1 RC - NEW
J2SE 1.4 APIs
J2SE 1.4 SDK
Java Tutorial
Recent Releases

Start Here
New to Java
What is Java?
Getting Started

Java plug-in for Windows XP.
Download now

SEPTEMBER 18th-20th SunNetwork
Click here to register and save $200 >>
Sun


Applications:

Text editor

Browser

Other apps

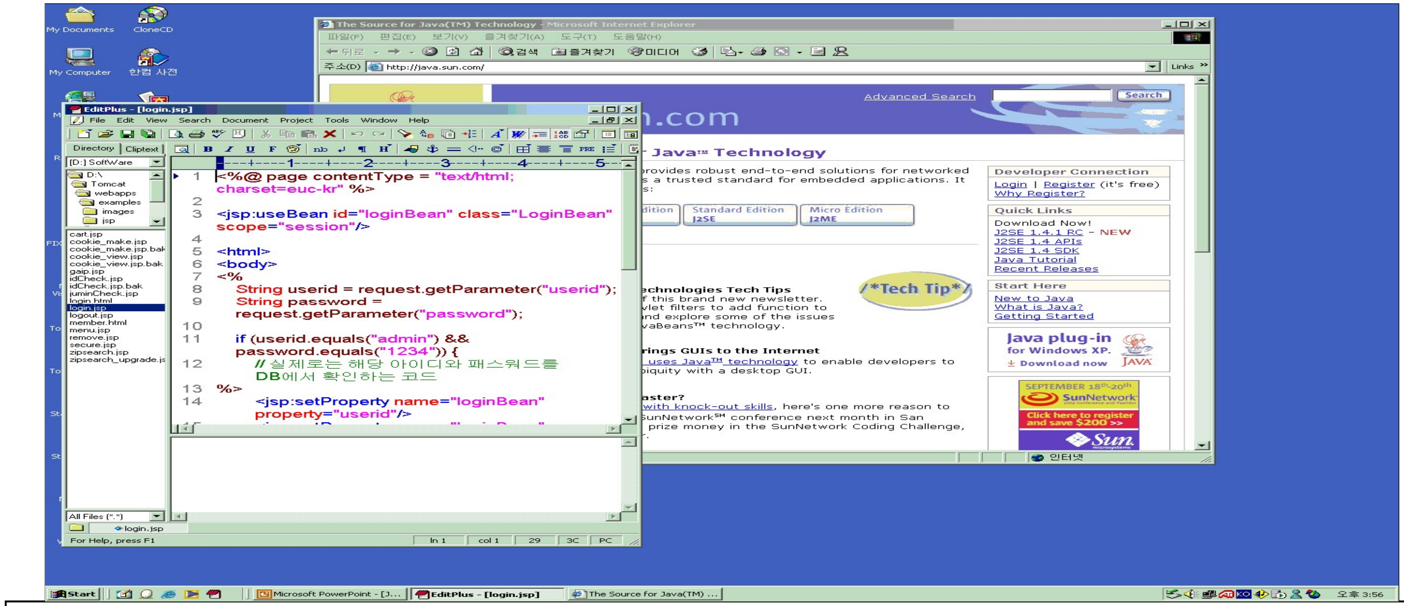
Operating system

Hardware

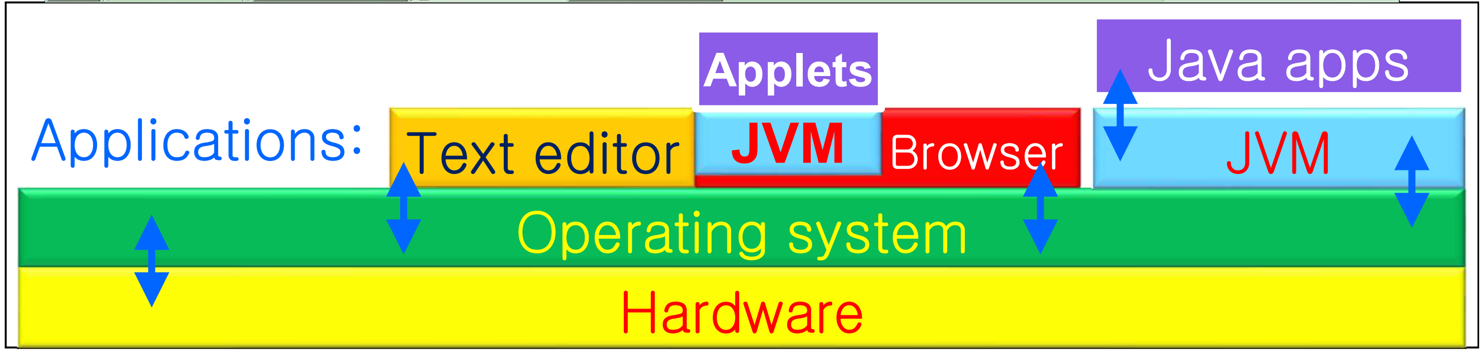
How Java Technology Solves the Communication Problem

- Uses compiling and interpretation.
- A little slower than compiled programs, but runs on any operating system.
- Compiles source code to ***bytecode***.
- Uses Java virtual machine (JVM^{TM}), interprets ***bytecode***.
- Uses a different JVM for every operating system.

Figure 1.5. How the JVM Interacts With the Operating System and Java Applets



```
<%@ page contentType = "text/html;
charset=euc-kr" %>

<jsp:useBean id="loginBean" class="LoginBean"
scope="session"/>

<html>
<body>
<%
   String userid = request.getParameter("userid");
   String password =
   request.getParameter("password");

   if (userid.equals("admin") &&
   password.equals("1234")) {
      // 실제로는 해당 아이디와 패스워드를
      DB에서 확인하는 코드
%>
      <jsp:setProperty name="loginBean"
      property="userid"/>
```

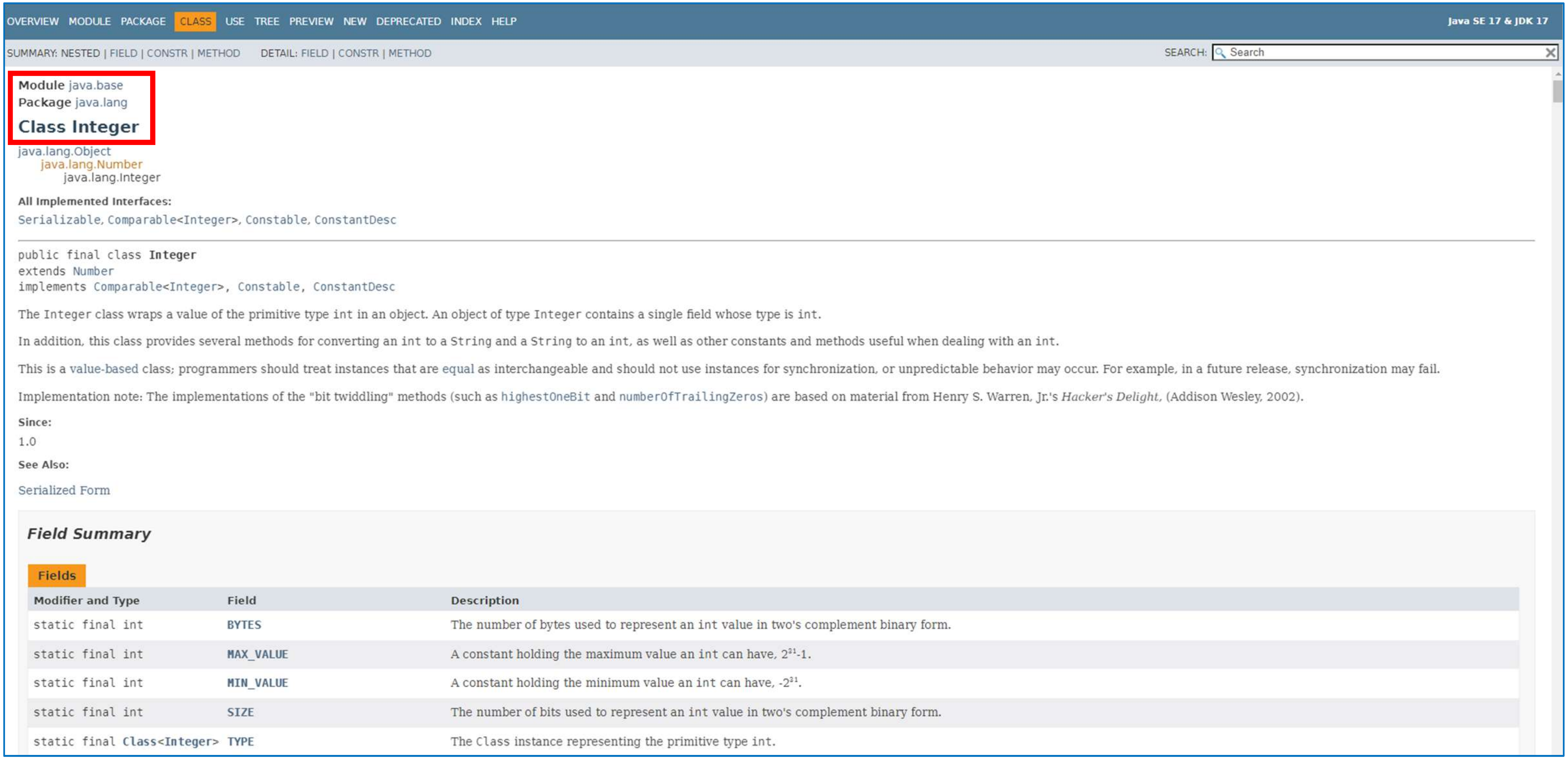


Java API Documentation

- Detailed information API
- Very valuable resource: download, or view online at:

https://docs.oracle.com/en/java/javase/17/docs/api/index.html

Figure 1.6. Java 2 Platform Specification, `java.lang` *Package,* `Integer` *Class*



Module java.base
Package java.lang

Class Integer

```
java.lang.Object
   java.lang.Number
      java.lang.Integer
```

All Implemented Interfaces:

`Serializable, Comparable<Integer>, Constable, ConstantDesc`

```
public final class Integer
extends Number
implements Comparable<Integer>, Constable, ConstantDesc
```

The `Integer` class wraps a value of the primitive type `int` in an object. An object of type `Integer` contains a single field whose type is `int`.

In addition, this class provides several methods for converting an `int` to a `String` and a `String` to an `int`, as well as other constants and methods useful when dealing with an `int`.

This is a value-based class; programmers should treat instances that are equal as interchangeable and should not use instances for synchronization, or unpredictable behavior may occur. For example, in a future release, synchronization may fail.

Implementation note: The implementations of the "bit twiddling" methods (such as `highestOneBit` and `numberOfTrailingZeros`) are based on material from Henry S. Warren, Jr.'s *Hacker's Delight*, (Addison Wesley, 2002).

Since:

1.0

See Also:

Serialized Form

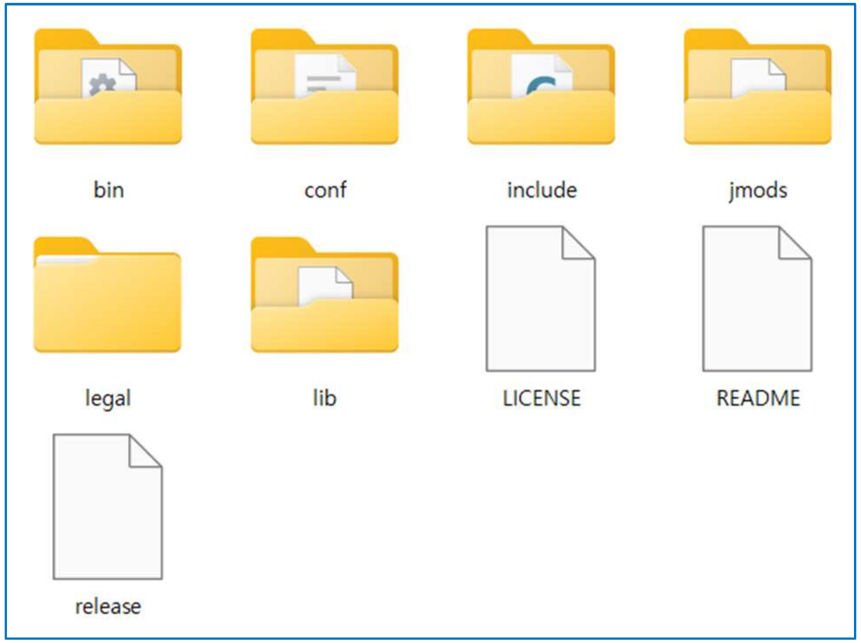
Field Summary

Fields

Modifier and Type	Field	Description
`static final int`	`BYTES`	The number of bytes used to represent an `int` value in two's complement binary form.
`static final int`	`MAX_VALUE`	A constant holding the maximum value an `int` can have, 2^{31}-1.
`static final int`	`MIN_VALUE`	A constant holding the minimum value an `int` can have, -2^{31}.
`static final int`	`SIZE`	The number of bits used to represent an `int` value in two's complement binary form.
`static final Class<Integer>`	`TYPE`	The `Class` instance representing the primitive type `int`.

JDK File Structure

- bin : contains the tools used for developing a Java application including the compiler and JVM.
- include : contains header files used to interact with C applications.
- lib/src.zip : includes the actual code for the core classes, called the SDK.



Additional Resource

- Java Technology : An Early History
  - http://oracle.com.edgesuite.net/timeline/java/
  - http://www.cs.umd.edu/class/spring2002/cmsc434-0101/MUIseum/applications/index.html
- Java Language and Virtual Machine Specifications
  - http://docs.oracle.com/javase/specs/
- Java SE6 API Hangul Documentation
  - http://docs.xrath.com/java/se/6/docs/ko/
  - http://docs.xrath.com/java/se/6/docs/ko/api/index.html
- Comparison of Java and C++
  - http://en.wikipedia.org/wiki/Comparison_of_Java_and_C%2B%2B
  - http://verify.stanford.edu/uli/java_cpp.html