Programming in Java
Getting Started

Hua Huang, Ph.D.

Spring 2019

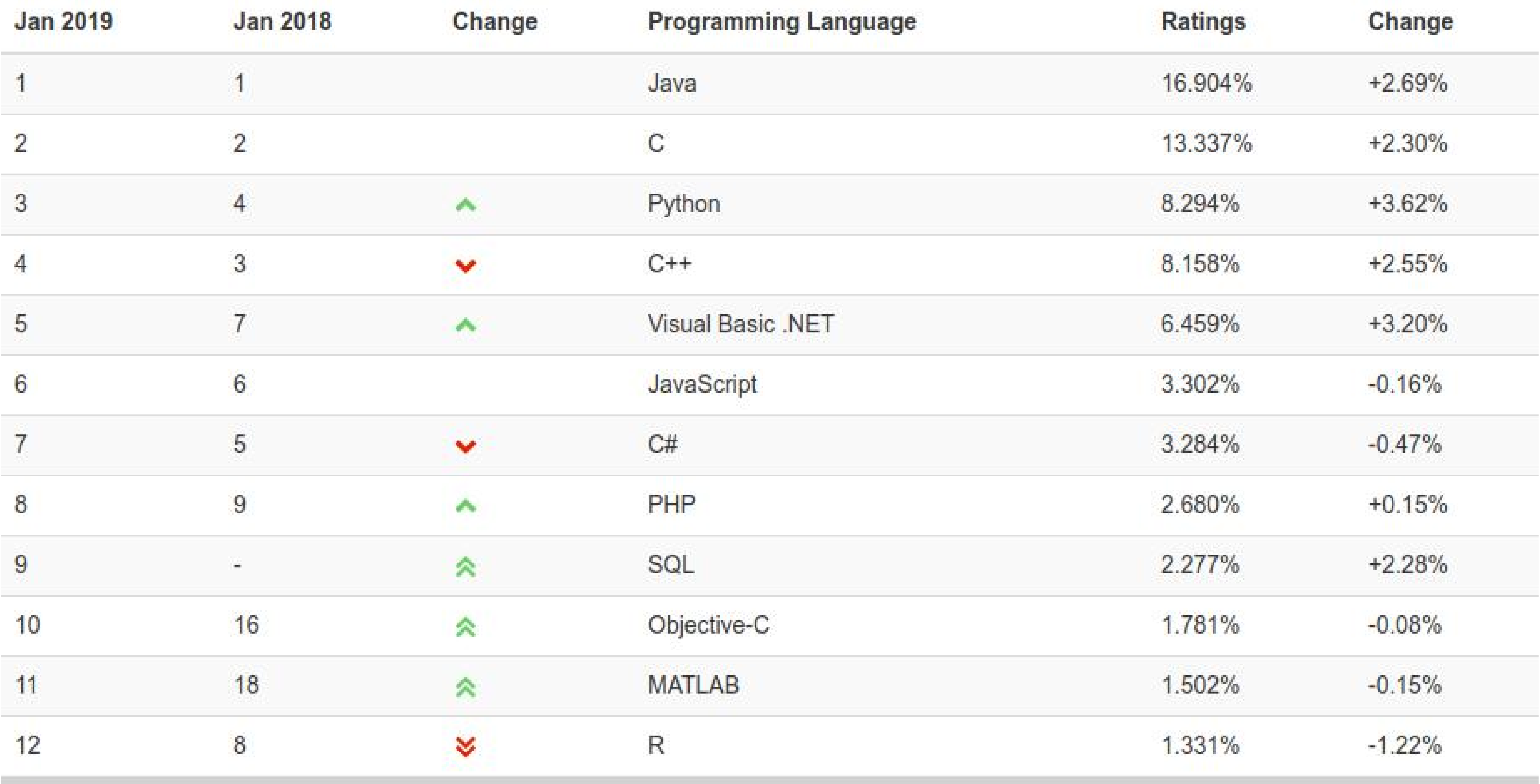
Objectives

- Describe the key features of Java technology
- Describe the function of the Java Virtual Machine (JVM™)
- Define garbage collection
- List the three tasks performed by the Java platform that handle code security
- Use the Java technology application programming interface (API) online documentation
- Examine the Java Development Kit (JDK™) software
- Examine Java application loading and executing
- Write, compile, and run a simple Java technology application and explore the compiling & running errors

Relevance

- Is the Java programming language a complete language or is it useful only for writing programs for the Web?
- Why do you need another programming language?
- How does the Java technology platform improve on other language platforms?

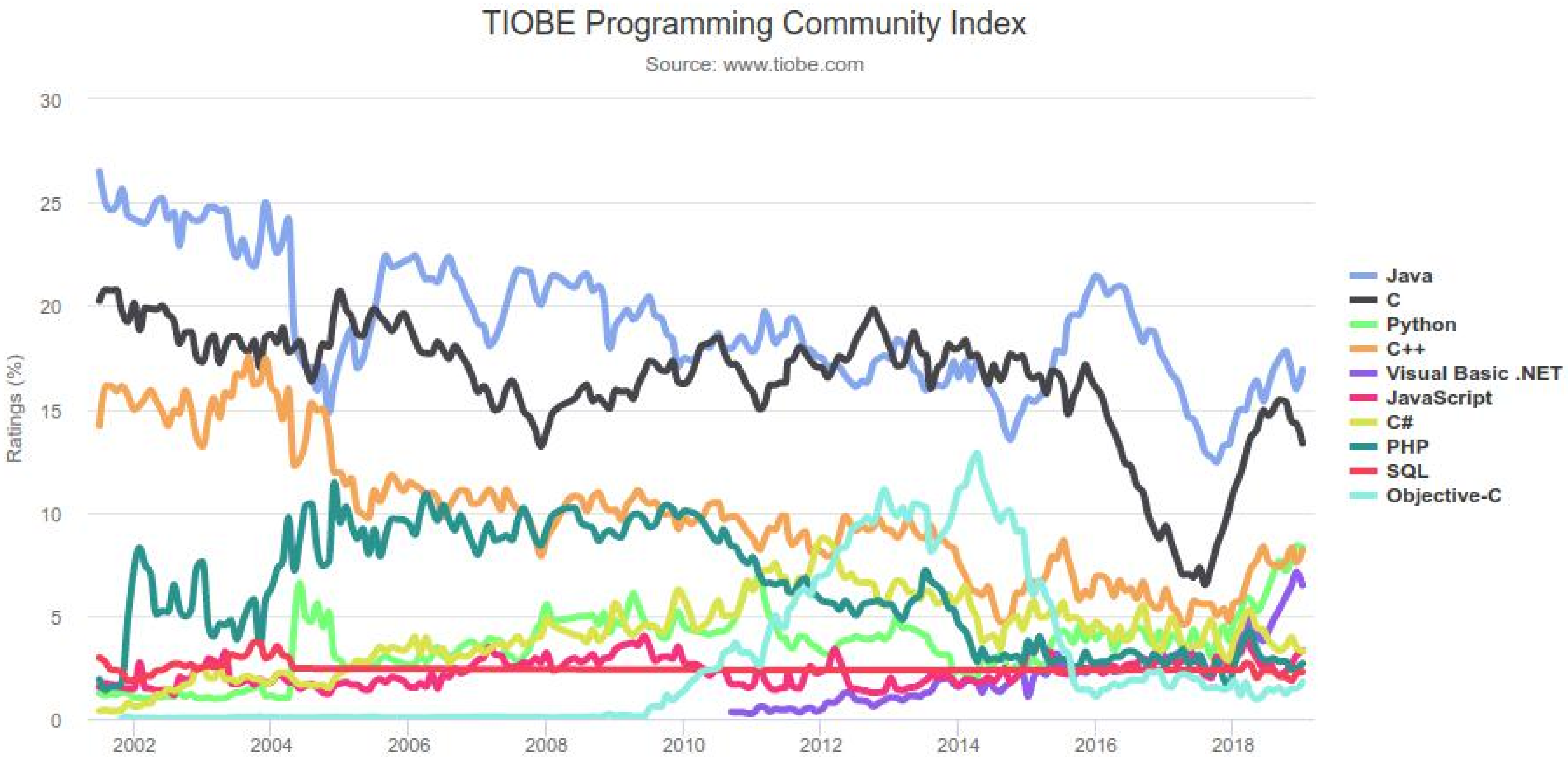
Popularity of programming languages*



Jan 2019	Jan 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.904%	+2.69%
2	2		C	13.337%	+2.30%
3	4	▲	Python	8.294%	+3.62%
4	3	▼	C++	8.158%	+2.55%
5	7	▲	Visual Basic .NET	6.459%	+3.20%
6	6		JavaScript	3.302%	-0.16%
7	5	▼	C#	3.284%	-0.47%
8	9	▲	PHP	2.680%	+0.15%
9	-	▲▲	SQL	2.277%	+2.28%
10	16	▲▲	Objective-C	1.781%	-0.08%
11	18	▲▲	MATLAB	1.502%	-0.15%
12	8	▼▼	R	1.331%	-1.22%

*Data from http://www.tiobe.com/tiobe-index/, January, 2019

Long Term Trends



What Is the Java™ Technology?

- Java technology is:
  - A programming language
  - A development environment
  - An application environment
  - A deployment environment
- It is similar in syntax to C++.
- It is used for developing both **applets** (Deprecated) and **applications** (Desktop and/or Web)

Key Features of Java Programming Language

- Simple
- Object oriented
- Distributed
- Multithreaded
- Dynamic
- Architecture neutral
- Portable
- High performance
- Robust
- Secure

Primary Goals of the Java Technology

- Provides an **easy-to-use** language by:
  - Avoiding many pitfalls of other languages
  - Being object-oriented
  - Enabling users to create streamlined and clear code
- Provides an **interpreted** environment for:
  - Improved speed of development
  - Code portability
  - Enables users to run more than one thread of activity
  - Loads classes dynamically; that is, at the time they are actually needed
  - Supports changing programs dynamically during runtime by loading classes from disparate sources
  - Furnishes better security

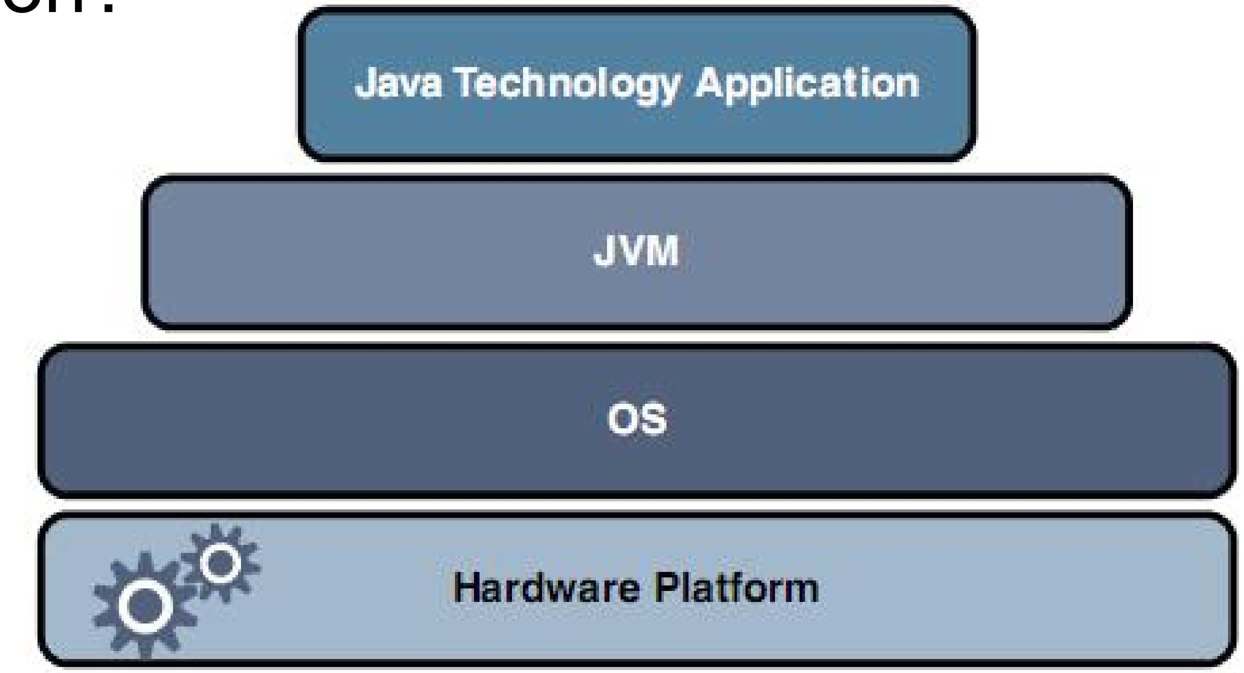
Primary Goals of the Java Technology (Cont')

- The following features fulfill these goals:
  - The Java Virtual Machine (JVM™)[1]
  - Garbage collection
  - The Java Runtime Environment (JRE)
  - JVM tool interface

[1] The terms "Java Virtual Machine" and "JVM" mean a Virtual Machine for the Java platform

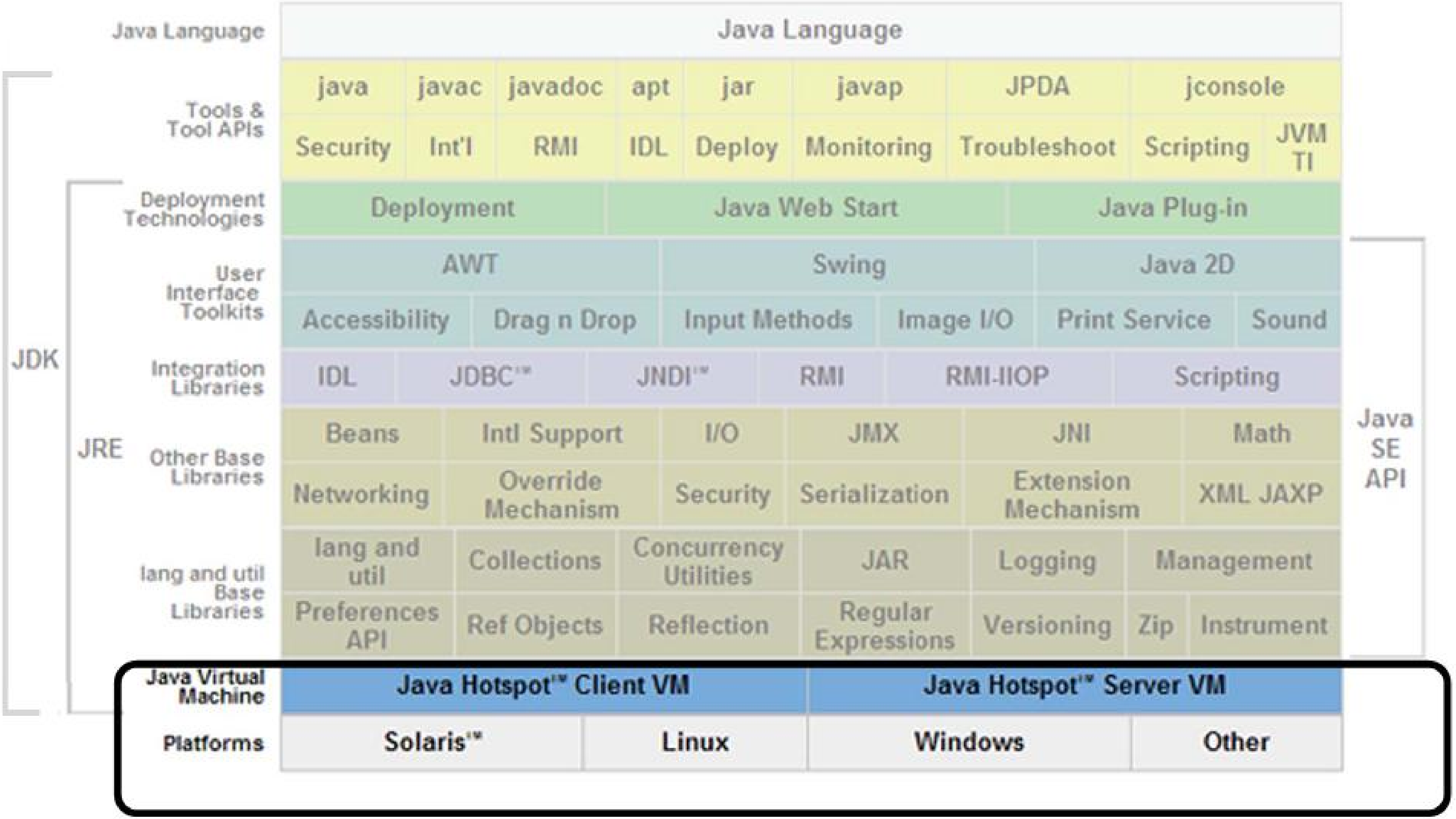
The Java Virtual Machine (JVM)

- What is a JVM implementation?



- Are JVM implementations platform dependent?
- Are Java technology applications platform dependent?
- What is a Java Hotspot™ (Client/Server) JVM implementation?

The JVM: Supported Platforms



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, Scripting

Other Base Libraries: Beans, Intl Support, I/O, 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, Instrument

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

Platforms: Solaris™, Linux, Windows, Other

The Java Virtual Machine

- Provides hardware platform specifications
- Reads compiled byte codes that are platform-independent
- Is implemented as **software** or **hardware**
- Is implemented in a Java technology development tool or a Web browser

The Java Virtual Machine (Cont')

- JVM provides definitions for the:
  - Instruction set (central processing unit [CPU])
  - Register set
  - Class file format
  - Stack
  - Garbage-collected heap
  - Memory area
  - Fatal error reporting
  - High-precision timing support
- The majority of **type checking** is done when the code is compiled.
- Implementation of the JVM approved by Oracle must be able to run any compliant class file.
- The JVM executes on multiple operating environments.

Garbage Collection

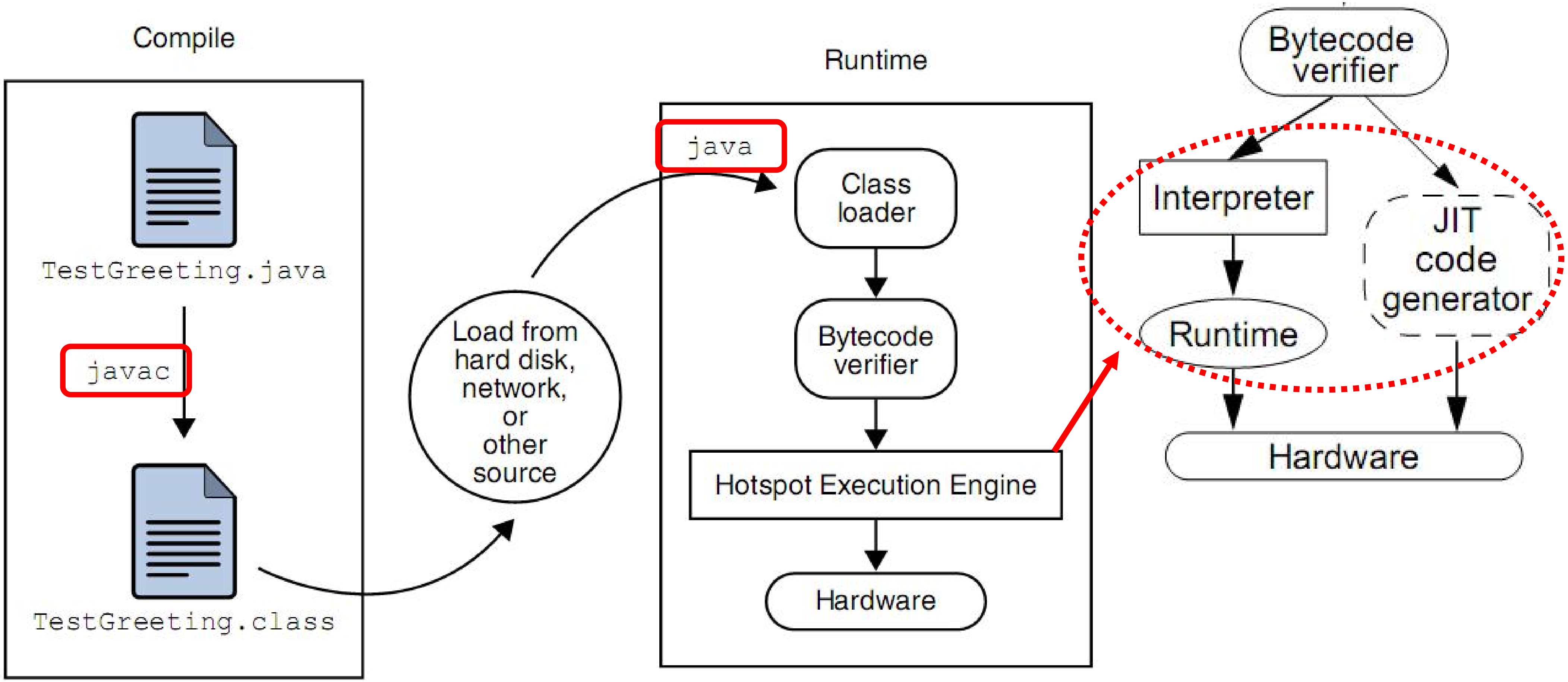
- Allocated memory that is no longer needed should be deallocated.
- In other languages, deallocation is the programmer's responsibility.

- The Java programming language provides a **system-level thread** to **track memory allocation**.

- Garbage collection has the following characteristics:
  - Checks for and frees memory no longer needed
  - Is done **automatically**
  - Can vary dramatically across JVM implementations

The Java Runtime Environment

- The Java application environment performs as follows:



JVM™ Tasks

- The JVM performs three main tasks:
  - **Loads** code
  - **Verifies** code
  - **Executes** code

The Class Loader

- **Loads all classes** necessary for the execution of a program

- Maintains classes of the local file system in separate **namespaces**

- Prevents spoofing(欺骗)

The Bytecode Verifier

- Ensures that:
  - The code adheres to the **JVM specification**.
  - The code does not violate **system integrity**.
  - The code causes **no operand stack overflows or underflows**.
  - The **parameter types** for all operational code are correct.
  - **No illegal data conversions** (the conversion of integers to pointers) have occurred.

A Simple Java Application

- The TestGreeting.java Application

```
//
// Sample "Hello World" application
//
public class TestGreeting{
    public static void main (String[] args)  {
        Greeting hello = new Greeting();
        hello.greet();
    }
}
```

- The Greeting.java Class

```
public class Greeting {
    public void greet() {
        System.out.println("hi");
    }
}
```

The **TestGreeting** Application

- The **TestGreeting** Application
  - Comment lines
  - Class declaration
  - The main method
  - Method body

- The **Greeting** Class
  - Class declaration
  - The greet method

Compiling and Running the **TestGreeting** Program

- Compile TestGreeting.java:

  ```
  javac TestGreeting.java
  ```

- The Greeting.java is compiled automatically.
- Run the application by using the following command:

  ```
  java TestGreeting
  ```

- Locate common compile and runtime errors.

Compile-Time Errors

- javac: Command not found
- ./Greeting.java:3: error: cannot find symbol

  ```
      System.out.printl("hi");
                ^
   symbol:   method printl(String)
   location: variable out of type PrintStream
  1 error
  ```

- TestGreet.java:4: error: class TestGreeting is public, should be declared in a file named TestGreeting.java

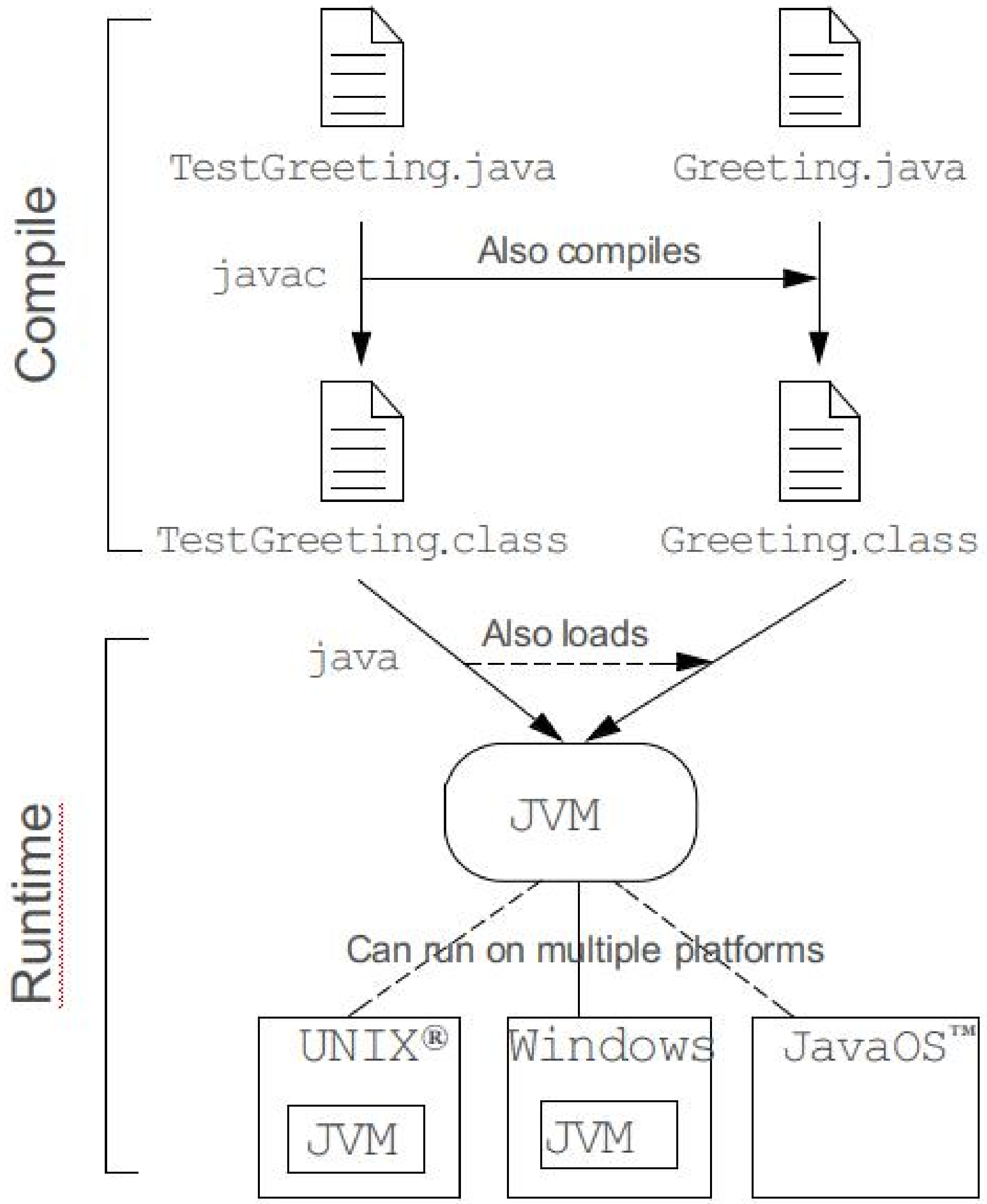
  ```
  public class TestGreeting{
         ^
  1 error
  ```

Runtime Errors

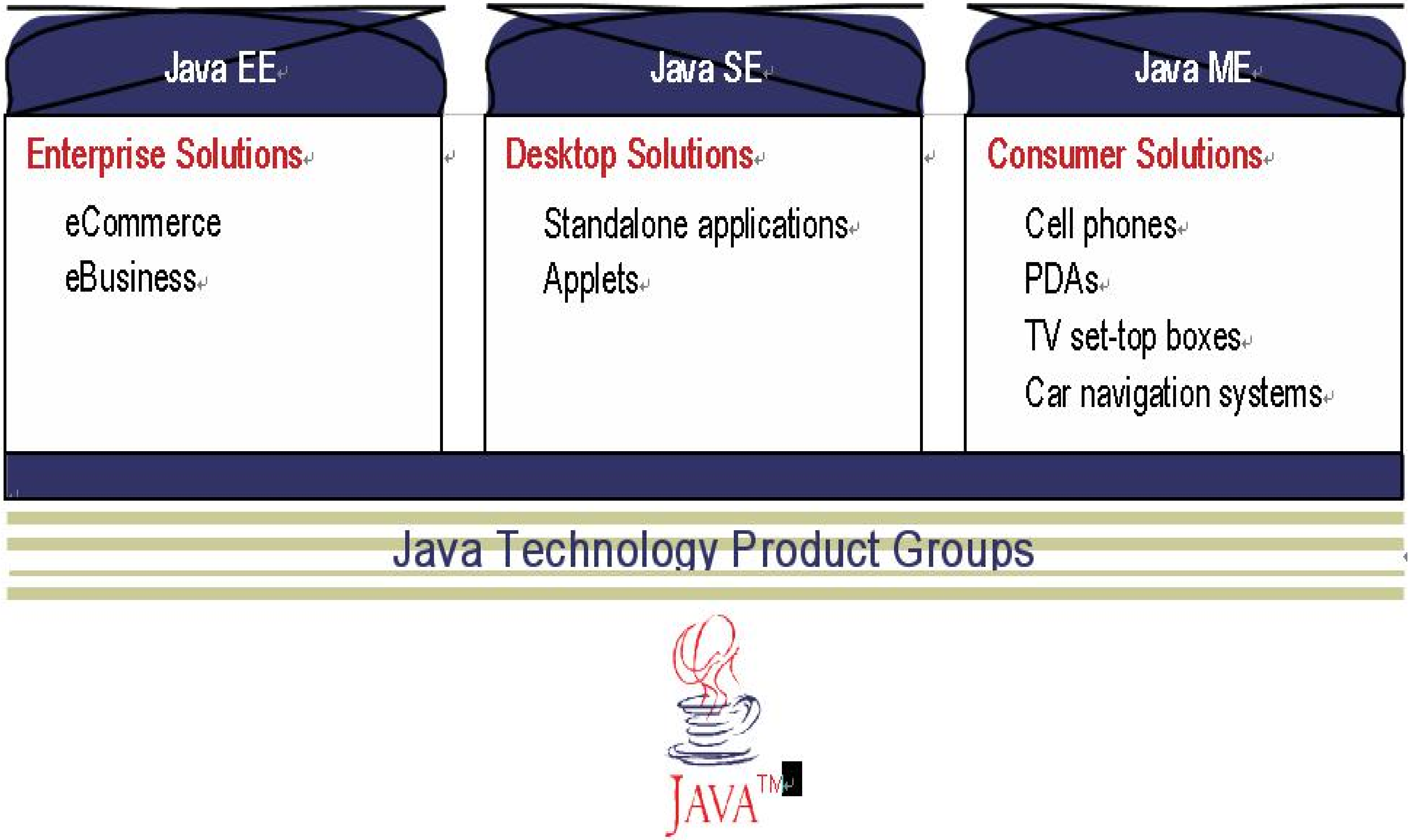
- Error: Could not find or load main class TestGreeting
  Caused by: java.lang.ClassNotFoundException: TestGreeting

- Error: Main method not found in class Greeting, please define the main method as:
  public static void main(String[] args)
  or a JavaFX application class must extend javafx.application.Application

Compiling and Running



Java Technology Product Groups



Java EE

Enterprise Solutions

eCommerce
eBusiness

Java SE

Desktop Solutions

Standalone applications
Applets

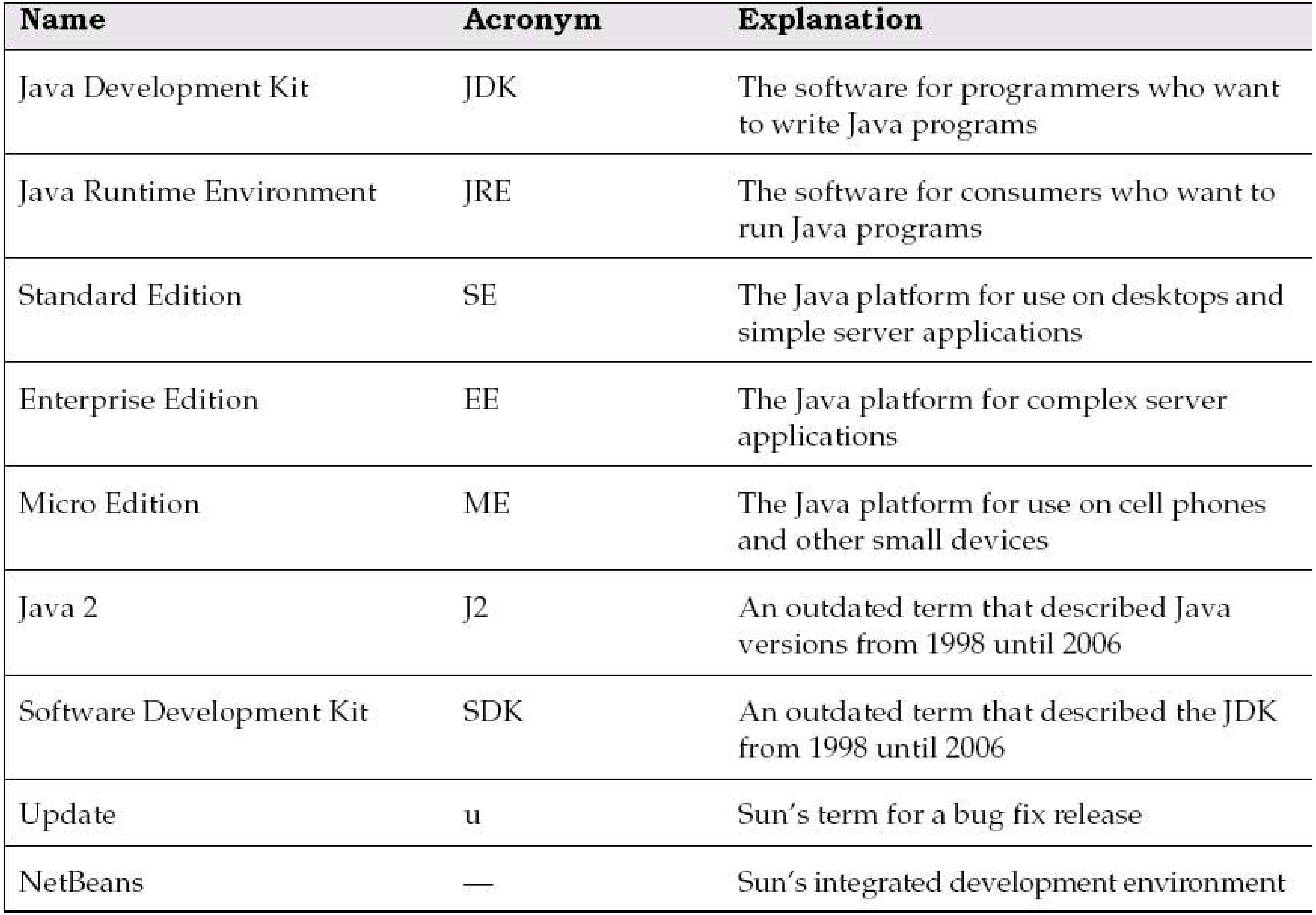
Java ME

Consumer Solutions

Cell phones
PDAs
TV set-top boxes
Car navigation systems

Java Technology Product Groups

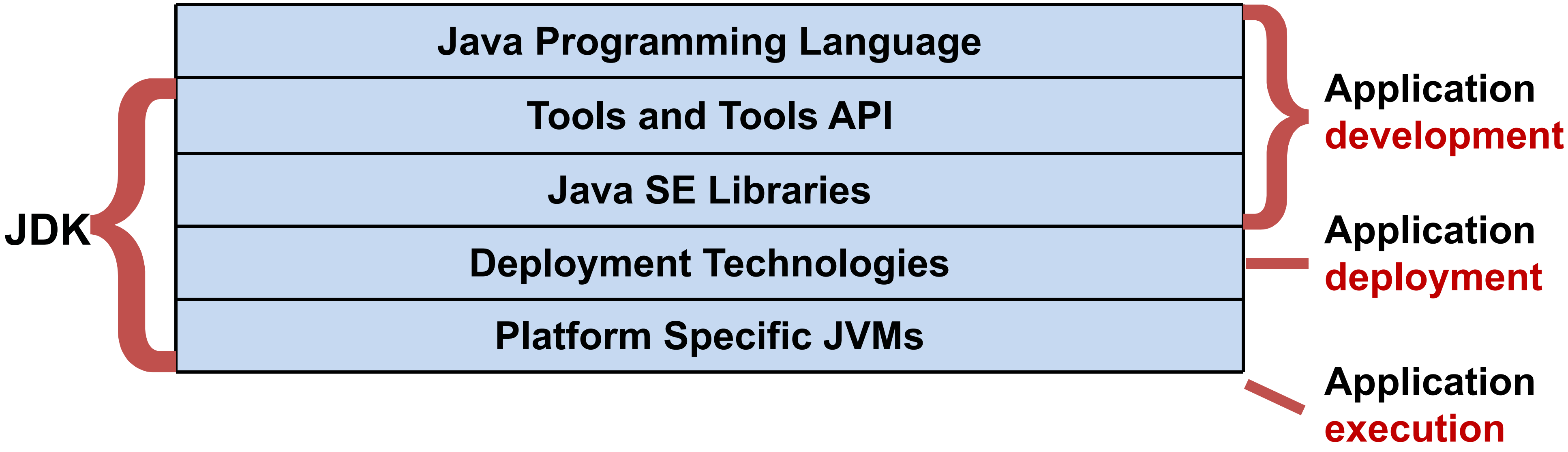
Java Technology Product Groups(Cont’)



Name	Acronym	Explanation
Java Development Kit	JDK	The software for programmers who want to write Java programs
Java Runtime Environment	JRE	The software for consumers who want to run Java programs
Standard Edition	SE	The Java platform for use on desktops and simple server applications
Enterprise Edition	EE	The Java platform for complex server applications
Micro Edition	ME	The Java platform for use on cell phones and other small devices
Java 2	J2	An outdated term that described Java versions from 1998 until 2006
Software Development Kit	SDK	An outdated term that described the JDK from 1998 until 2006
Update	u	Sun’s term for a bug fix release
NetBeans	—	Sun’s integrated development environment

Examining the JDK Software

- The JDK contains components to perform the following tasks:
  - **Develop** Java technology applications
  - **Deploy** Java technology applications
  - **Execute** Java technology applications



Components of the JDK

- The Java programming language
- **Tools and tools API**
- **Deployment** technologies
- Java Platform, Standard Edition (Java SE) **libraries**
- Java Virtual Machine (**JVM**™)

Examples of Java technology programs are also in bundle.

Strictly speaking the Java programming language is not a component of the JDK software. Nevertheless, for the purposes of providing a more complete discussion, it is treated as a pseudo component.

Download URL:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

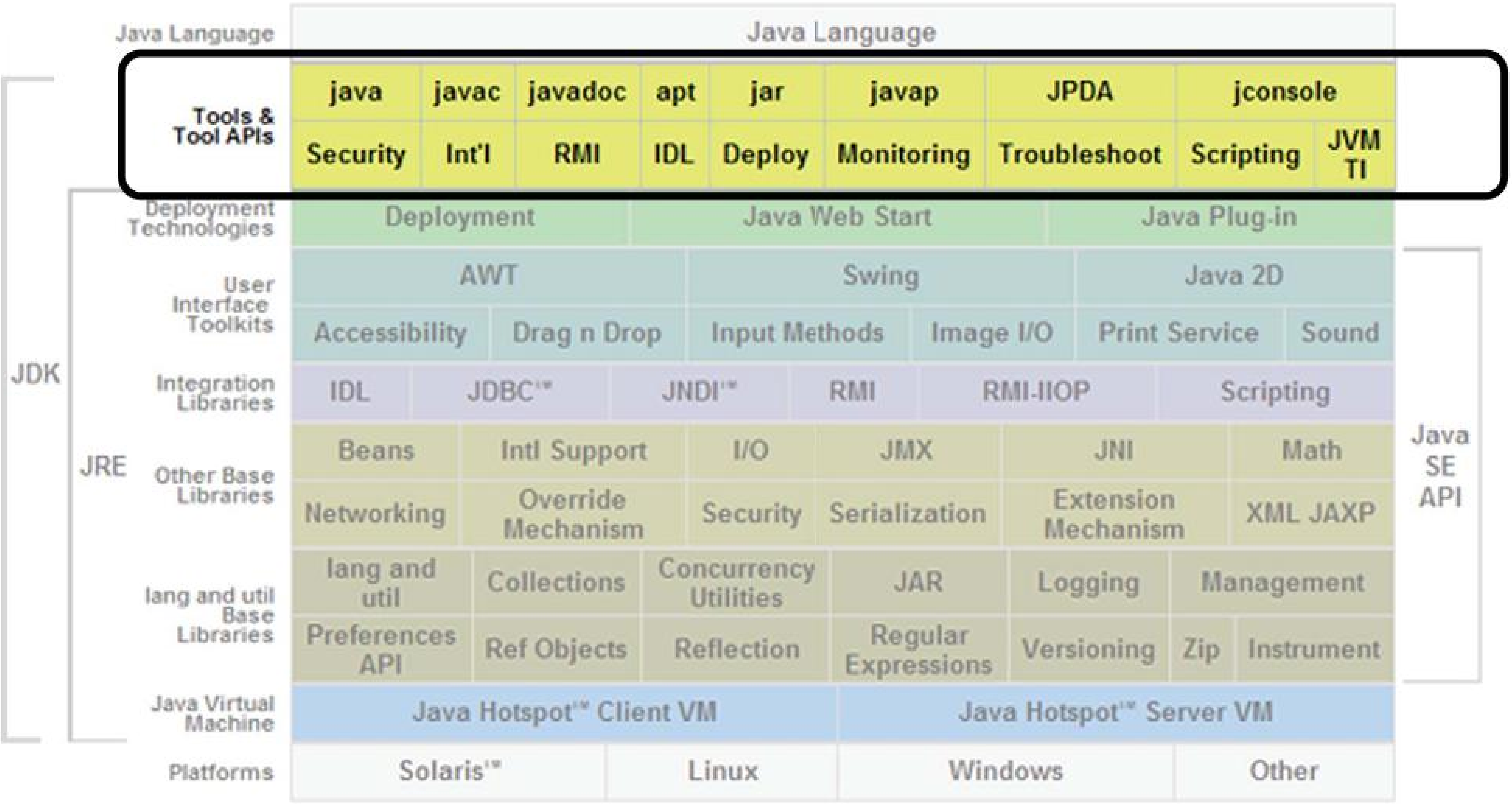
JDK Support for Developing Java Applications

- The Java programming language
- The JDK tools
- The JDK libraries

The Java Programming Language

- The Java programming language is a **general-purpose, concurrent, strongly typed, class-based object-oriented** language.
- The Java programming language is defined by the **Java language specification**.
- The primary building block of a Java technology application is a **class**.

The JDK Tools and Tools 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 | Scripting

Other Base Libraries — Beans | Intl Support | I/O | 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 | Instrument

Java Virtual Machine — Java Hotspot™ Client VM | Java Hotspot™ Server VM

Platforms — Solaris™ | Linux | Windows | Other

JDK

JRE

Java SE API

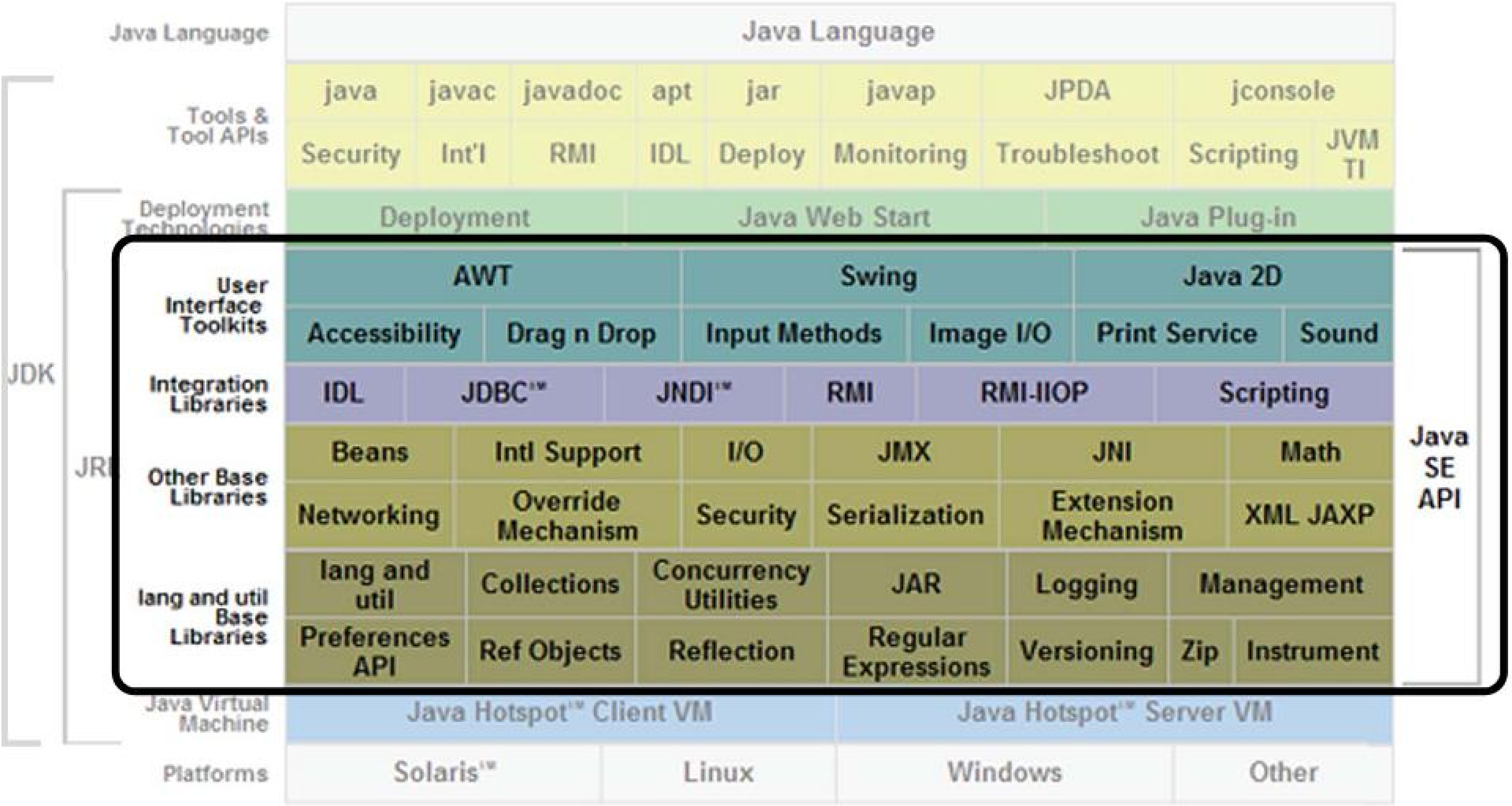
Basic Tools

Tool Name	**Function**
javac	The compiler for the Java programming language
java	The launcher for Java technology applications
jdb	The Java debugger
javadoc	The API document generator
jar	Java Archive (JAR) file creator and management tool

Advanced Tools

Tool Category	Comments
Security tools	Implement security policies in applications
Internationalization tools	Enable applications to be localized
Remote method invocation (RMI) tools	Create (network) distributed applications
Common object request broker architecture (CORBA) tools	Create network applications that are based on CORBATechnology
Java deployment tools	Support application deployment
Java Plug-in tools	Provide utilities for use with the Java Plug-in
Java web start tools	Used with Java web start technology
...	

JDK Libraries



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, Scripting

Other Base Libraries: Beans, Intl Support, I/O, 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, Instrument

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

Platforms: Solaris™, Linux, Windows, Other

JDK Libraries

Library Name	Sample Classes in Library
java.lang	Enum, Float, String, Object
java.util	ArrayList, Calendar, Date
java.io	File, Reader, Writer
java.math	BigDecimal, BigInteger
java.text	DateFormat, Collator
javax.crypto	Cipher, KeyGenerator
java.net	Socket, URL, InetAddress
java.sql	ResultSet, Date, Timestamp
javax.swing	JFrame, JPanel
javax.xml.parsers	DocumentBuilder, SAXParser

Java Technology API Documentation

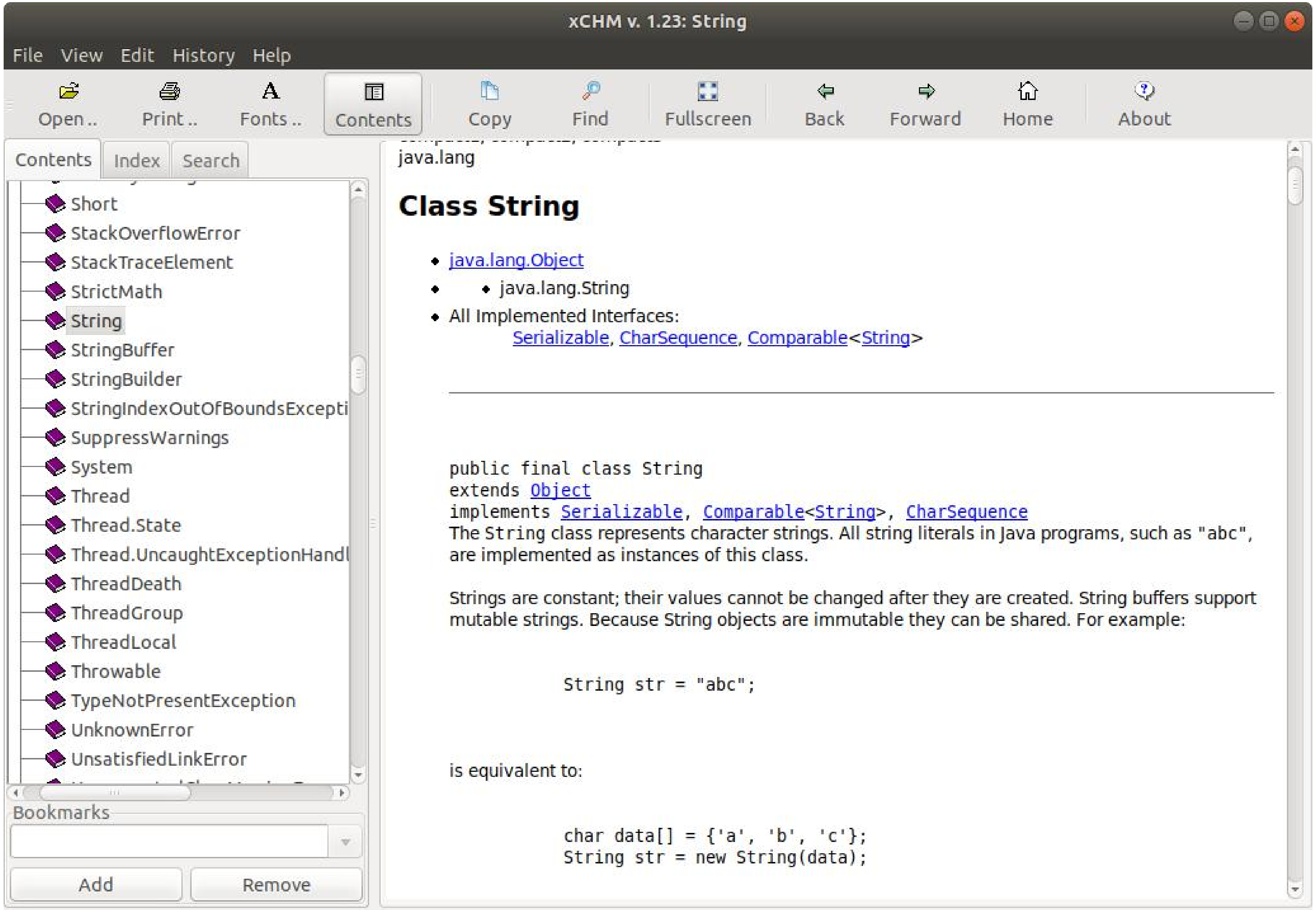
- A set of Hypertext Markup Language (**HTML**) files provides **information about the API**.

- A frame describes a package and contains hyperlinks to information describing each class in that package.

- A class document includes the class hierarchy, a description of the class, a list of member variables, a list of constructors, and so on.

- You can also get a single “chm” file instead of thousands of HTML files from 3rd parties.
  - https://javadoc.allimant.org/

Java Technology API Documentation



java.lang

Class String

- java.lang.Object
-     - java.lang.String
- All Implemented Interfaces:
  Serializable, CharSequence, Comparable<String>

```
public final class String
extends Object
implements Serializable, Comparable<String>, CharSequence
```

The String class represents character strings. All string literals in Java programs, such as "abc", are implemented as instances of this class.

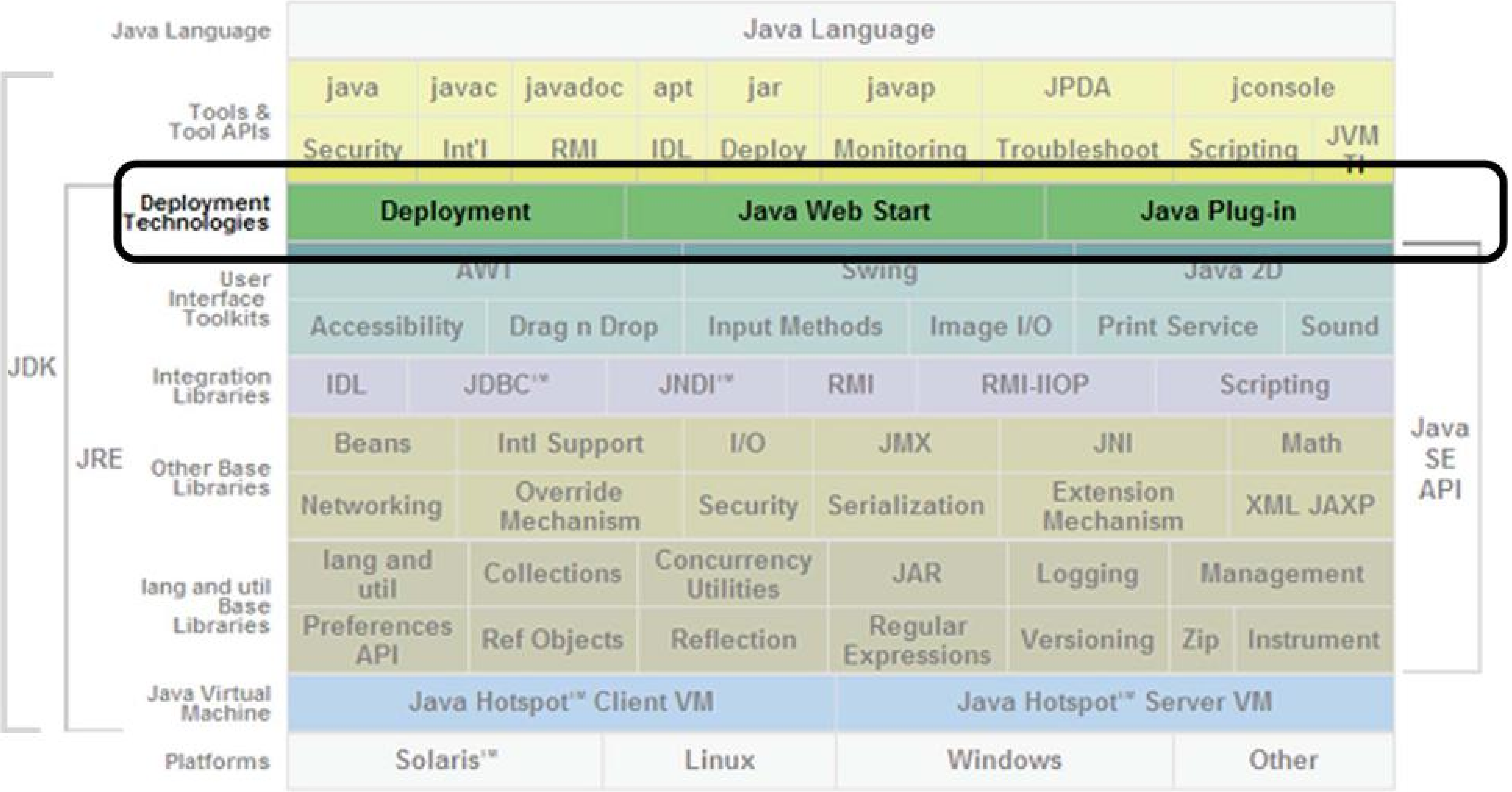
Strings are constant; their values cannot be changed after they are created. String buffers support mutable strings. Because String objects are immutable they can be shared. For example:

```
String str = "abc";
```

is equivalent to:

```
char data[] = {'a', 'b', 'c'};
String str = new String(data);
```

JDK Deployment Technologies



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, Scripting

Other Base Libraries — Beans, Intl Support, I/O, JMX, JNI, Math

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

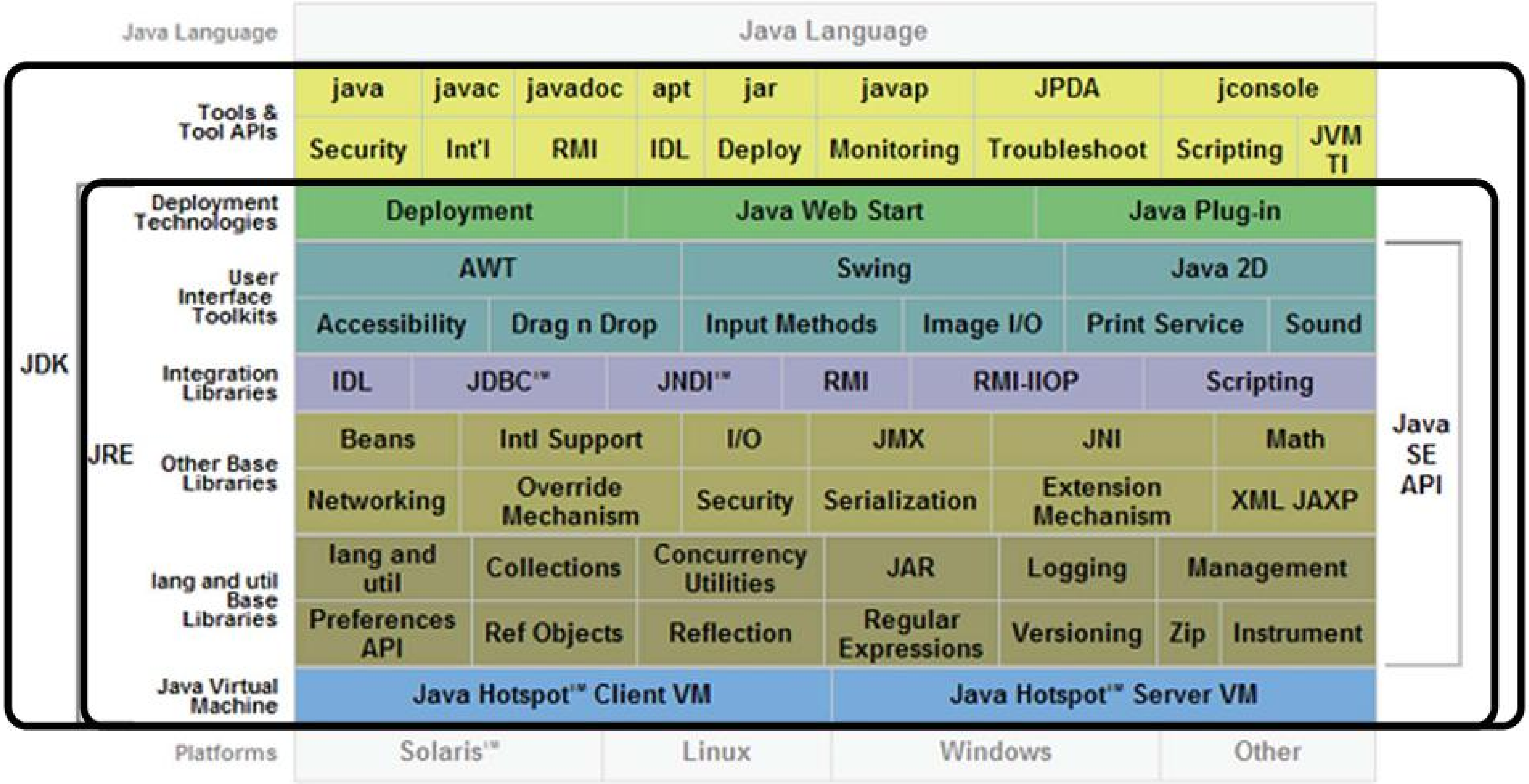
lang and util Base Libraries — lang and util, Collections, Concurrency Utilities, JAR, Logging, Management

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

Java Virtual Machine — Java Hotspot™ Client VM, Java Hotspot™ Server VM

Platforms — Solaris™, Linux, Windows, Other

The Java Runtime Environment (JRE™)



Java Language: Java Language

JDK

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

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

JRE

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, Scripting

Other Base Libraries: Beans, Intl Support, I/O, 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, Instrument

Java SE API

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

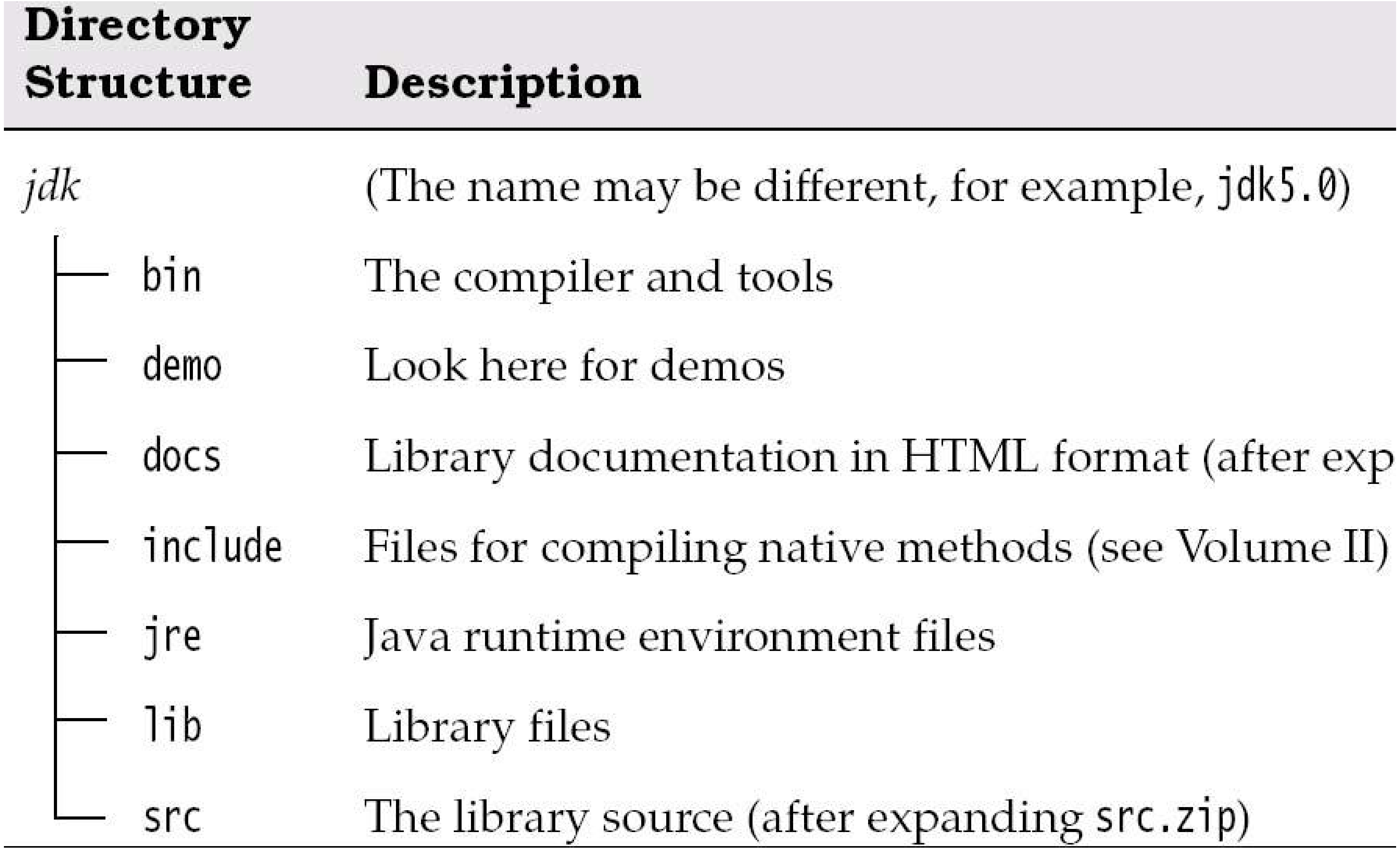
Platforms: Solaris™, Linux, Windows, Other

Prepare the Programming Environment

- Download and install the Java Development Kit, Available from Oracle for Solaris, Linux, Windows and et al.
  - Download URL: http://www.oracle.com/technetwork/java/javase/downloads/index.html

- Note:
  - Windows users are strongly recommend not to accept the default location with spaces in the path , such as c:\Program Files\jdk1.8.0. Just use c:\jdk1.8.0

Installation Directory structure



Directory Structure	Description
jdk	(The name may be different, for example, `jdk5.0`)
`bin`	The compiler and tools
`demo`	Look here for demos
`docs`	Library documentation in HTML format (after exp
`include`	Files for compiling native methods (see Volume II)
`jre`	Java runtime environment files
`lib`	Library files
`src`	The library source (after expanding `src.zip`)

Command-Line Tools & Env

- Cmd line tools
  - javac, java, appletviewer...
- Env Settings under Windows
  - system properties dialog->Advanced tab->Environment button->System Variables Path , Add the jdk\bin directory to the beginning of the path, such as: d:\jdk1.8\bin;other stuff
- Env Settings under UNIX/Linux
  - Bourne Again shell:~/.bashrc or ~/.bash_profile file: export PATH=/usr/local/jdk/bin:$PATH
  - Others, Google/Baidu
- java -version

Deal with source codes

- Text Editors
  - Visual Studio Code (Strongly Recommended), Notepad++...
  - Compile and run with commands
- Integrated Development Environment
  - Eclipse
    - http://eclipse.org, the most commonly used, origin from IBM
  - IntelliJ IDEA
    - http://www.jetbrains.com/idea/
  - Netbeans
    - http://netbeans.org
  - Others
    - Oracle JDeveloper, JCreator …

Questions or Comments?