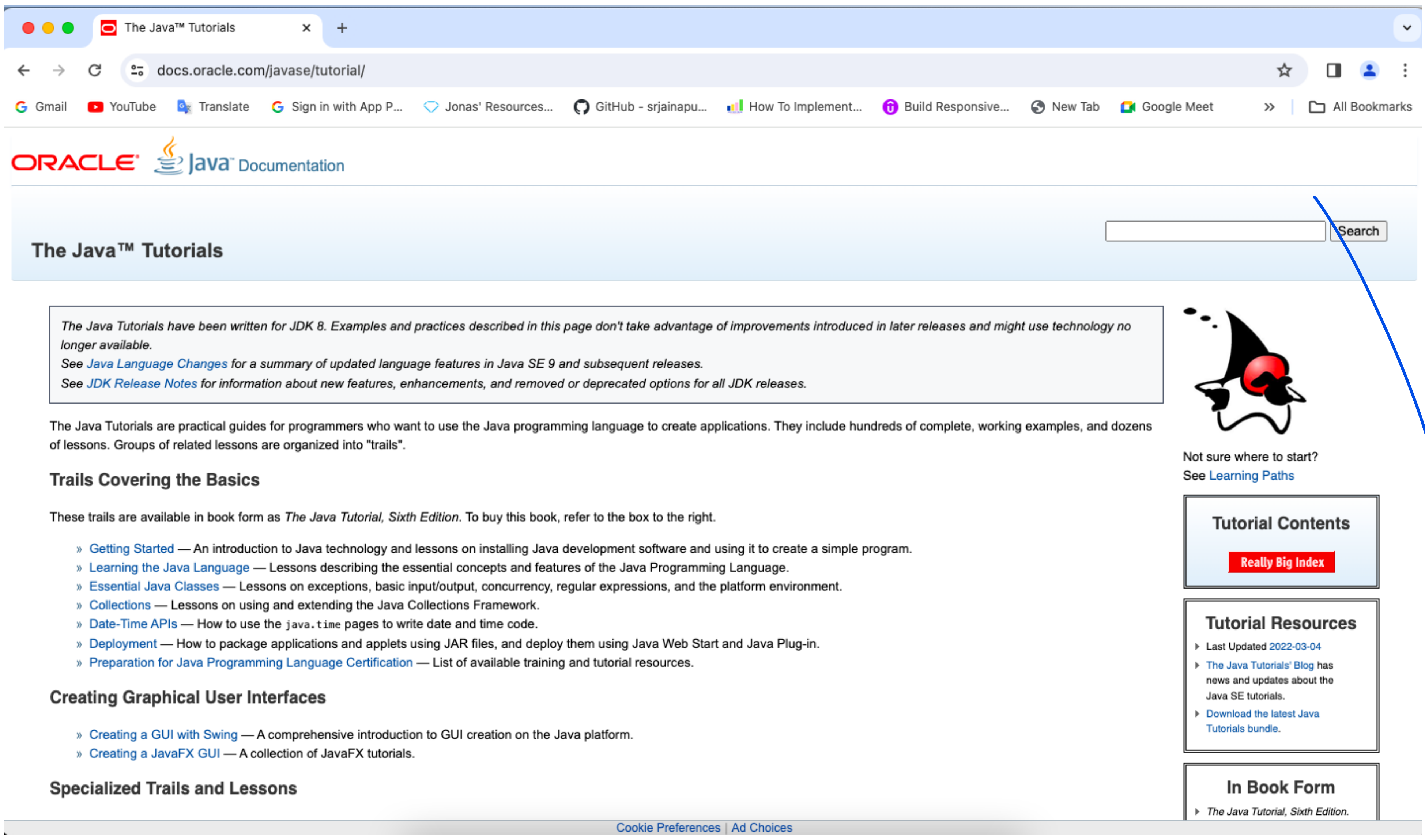
Java Jumpstart

Module Details

- **Module Name**: Object Oriented Programming with Java
- **Duration**: 112 hours
  - **Theory Hours**: 50
  - **Lab Hours**: 50
  - **Revision/Practice Hours**: 12
- **Evaluation**: Total 100 Marks
- **Weightage**:
  - **Theory Exam**: 40%
  - **Lab Exam**: 40%
  - **Internal**: 20%

References

- URL: https://docs.oracle.com/javase/tutorial/



The Java™ Tutorials

The Java Tutorials have been written for JDK 8. Examples and practices described in this page don't take advantage of improvements introduced in later releases and might use technology no longer available.
See Java Language Changes for a summary of updated language features in Java SE 9 and subsequent releases.
See JDK Release Notes for information about new features, enhancements, and removed or deprecated options for all JDK releases.

The Java Tutorials are practical guides for programmers who want to use the Java programming language to create applications. They include hundreds of complete, working examples, and dozens of lessons. Groups of related lessons are organized into "trails".

Trails Covering the Basics

These trails are available in book form as *The Java Tutorial, Sixth Edition*. To buy this book, refer to the box to the right.

» Getting Started — An introduction to Java technology and lessons on installing Java development software and using it to create a simple program.
» Learning the Java Language — Lessons describing the essential concepts and features of the Java Programming Language.
» Essential Java Classes — Lessons on exceptions, basic input/output, concurrency, regular expressions, and the platform environment.
» Collections — Lessons on using and extending the Java Collections Framework.
» Date-Time APIs — How to use the `java.time` pages to write date and time code.
» Deployment — How to package applications and applets using JAR files, and deploy them using Java Web Start and Java Plug-in.
» Preparation for Java Programming Language Certification — List of available training and tutorial resources.

Creating Graphical User Interfaces

» Creating a GUI with Swing — A comprehensive introduction to GUI creation on the Java platform.
» Creating a JavaFX GUI — A collection of JavaFX tutorials.

Specialized Trails and Lessons

Not sure where to start?
See Learning Paths

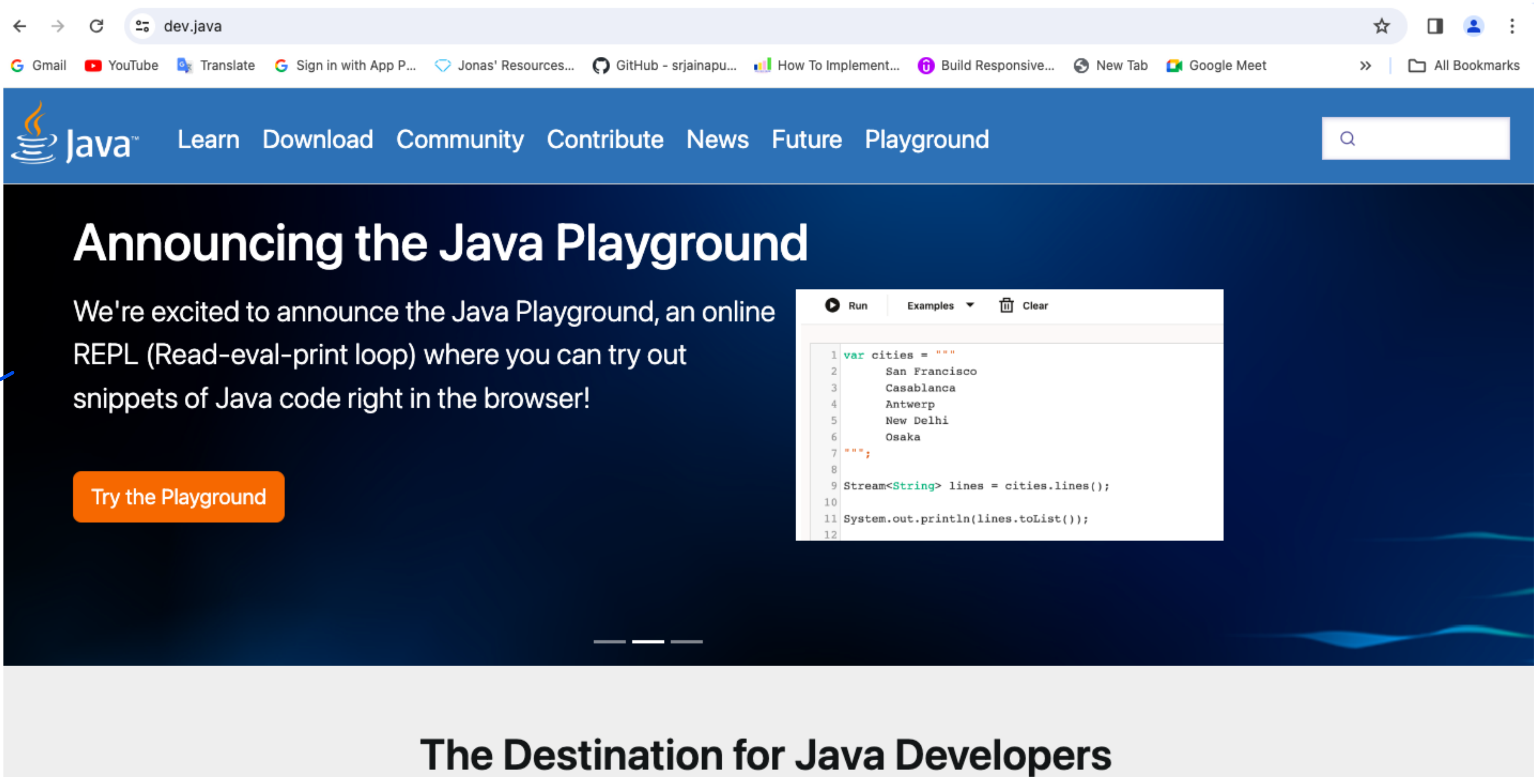
Tutorial Contents
Really Big Index

Tutorial Resources
- Last Updated 2022-03-04
- The Java Tutorials' Blog has news and updates about the Java SE tutorials.
- Download the latest Java Tutorials bundle.

In Book Form
- *The Java Tutorial, Sixth Edition.*

Cookie Preferences | Ad Choices

- URL: https://dev.java/



Announcing the Java Playground

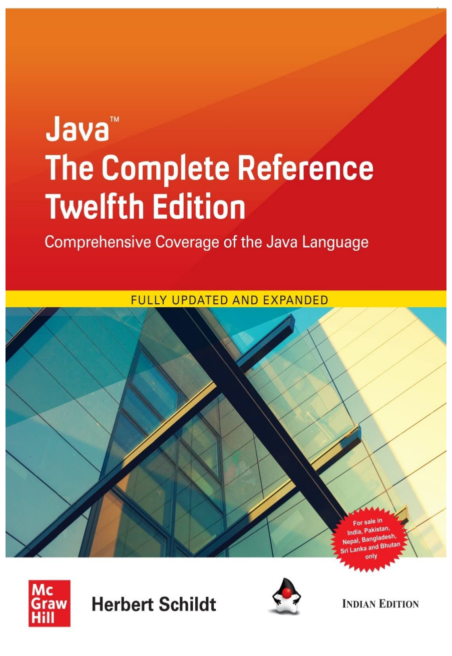
We're excited to announce the Java Playground, an online REPL (Read-eval-print loop) where you can try out snippets of Java code right in the browser!

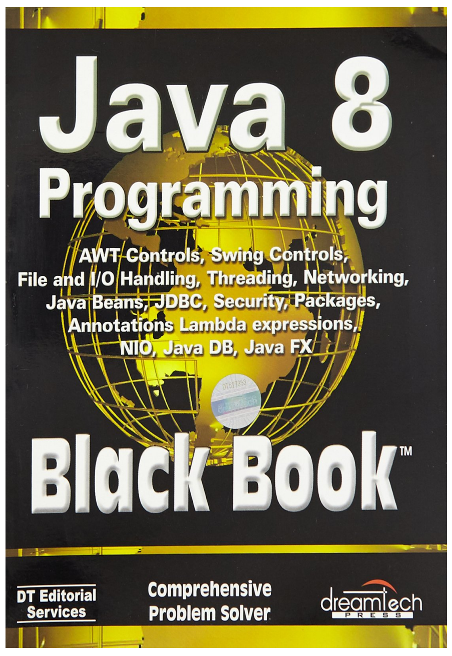
Try the Playground

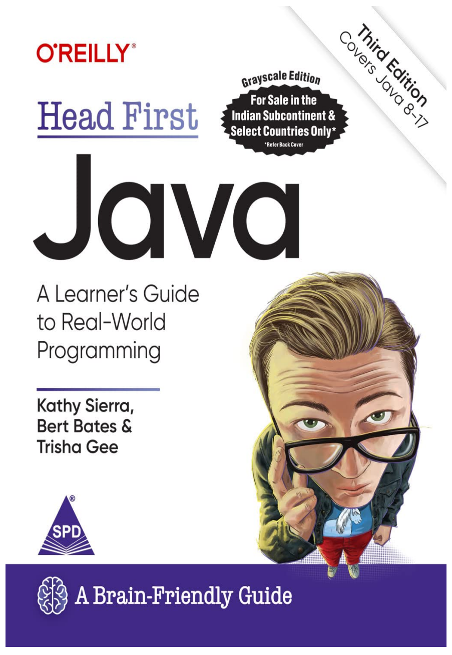
```
var cities = """
      San Francisco
      Casablanca
      Antwerp
      New Delhi
      Osaka
""";

Stream<String> lines = cities.lines();

System.out.println(lines.toList());
```

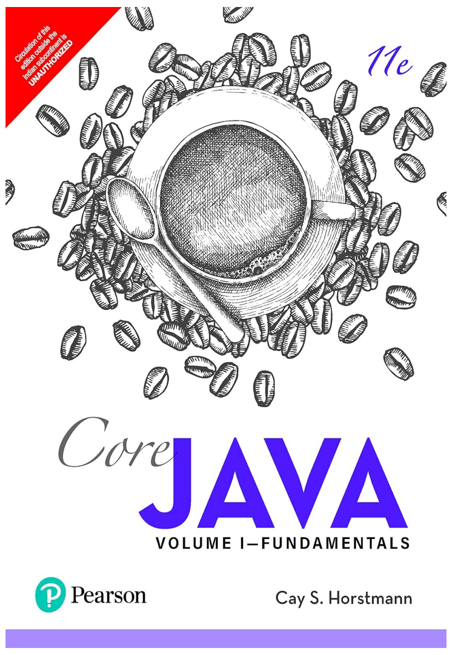
The Destination for Java Developers

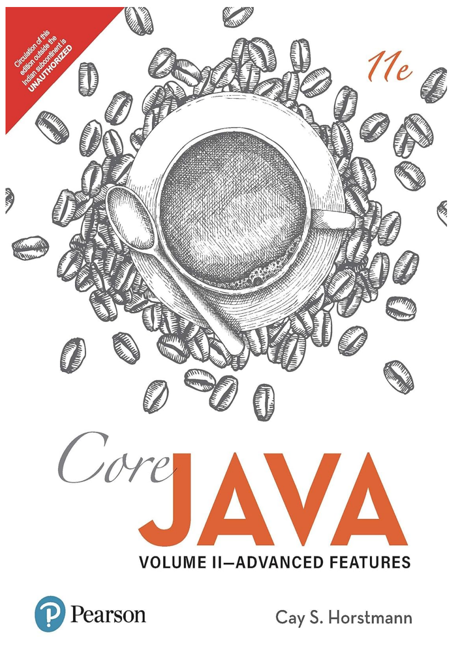
- **Books:**

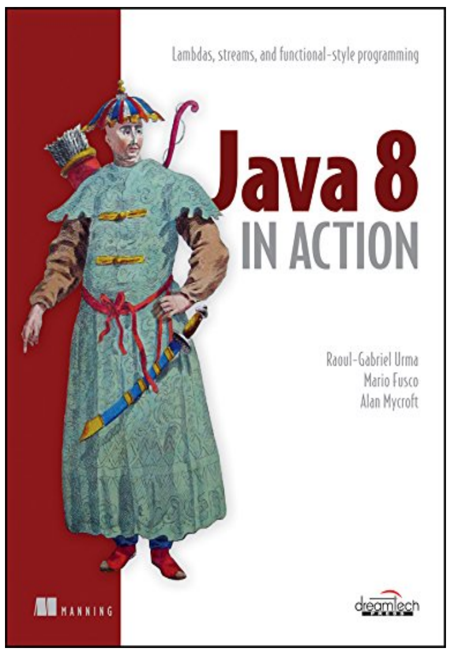

Java™ The Complete Reference Twelfth Edition
Comprehensive Coverage of the Java Language
Herbert Schildt

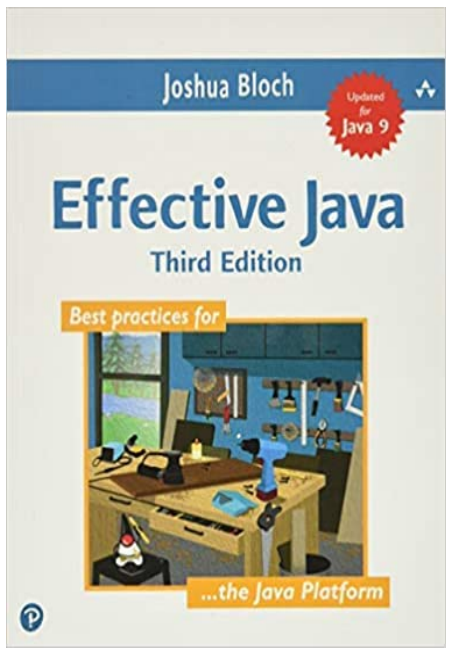


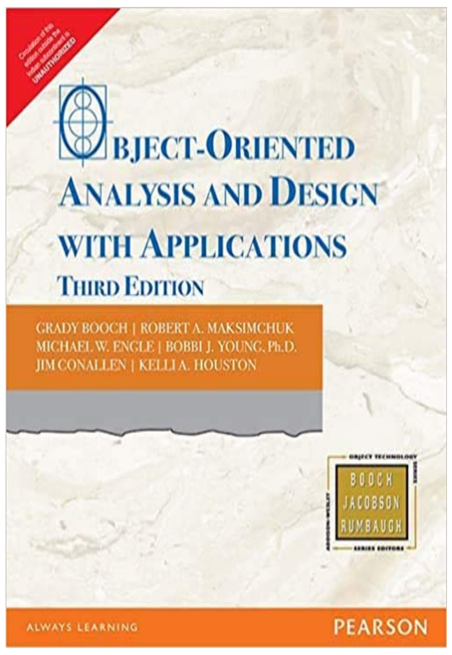

O'REILLY
Head First Java
A Learner's Guide to Real-World Programming
Kathy Sierra, Bert Bates & Trisha Gee
Third Edition Covers Java 8-17

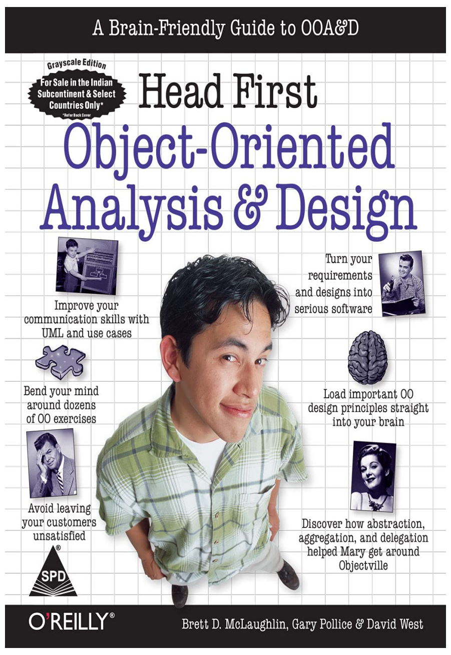









OBJECT-ORIENTED ANALYSIS AND DESIGN WITH APPLICATIONS
THIRD EDITION
GRADY BOOCH | ROBERT A. MAKSIMCHUK
MICHAEL W. ENGLE | BOBBI J. YOUNG, Ph.D.
JIM CONALLEN | KELLI A. HOUSTON
PEARSON


A Brain-Friendly Guide to OOA&D
Head First Object-Oriented Analysis & Design
O'REILLY
Brett D. McLaughlin, Gary Pollice & David West

Language versus Technology versus Platform versus Framework

Language

- A programming language is a system of notation for writing computer programs.
- A programming language is described by its syntax (form) and semantics (meaning).
- Consider the arithmetic expression: 3 + 5:
  - The syntax dictates how the expression 3 + 5 should be written to be valid in the programming language.
  - The semantics determine what the expression 3 + 5 actually means and what should be done when it is executed.
- If we want to implement business logic then we should use language.
- Using language, we can develop application too. Hence every language can be considered as a technology.
  - Command Line Interface(CLI) / Console User Interface(CUI) application.
  - Graphical User Interface(GUI) application.
  - Library application
  - Network application
- Example(s):
  - C, C++, Java, C#, Ruby, Python, GO, Swift, Scala, Java Script, R etc.

Technology

- In computing, technology refers to the tools, techniques, and methodologies used to design, develop, deploy, and maintain software applications.
- Using technology we can create application. It doesn't mean every technology is a language.
- Example: Java, Remote Method Invocation(RMI), Servlet etc.

Platform

- A platform is the hardware or software environment in which a program runs.
- Most platforms can be described as a combination of the operating system and underlying hardware.
- Example(s)
  - Hardware based platforms
    - Microsoft Windows
      - Windows 98, Windows Me, Windows 2000, Windows XP, Windows Vista,Windows 7,Windows 8, Windows 8.1,Windows 10, Windows 11
    - Unix/Linux
      - Ubuntu, Fedora, openSUSE, CentOS, Linux Mint, RHEL etc.
    - Mac OS
      - Sierra, High Sierra, Mojave, Catalina, Big Sur, Monterey, Ventura, Sonoma
    - Mobile OS
      - Android, IOS
    - Cloud Computing Platforms
      - Amazon Web Services(AWS), Google Cloud Platform(GCP), MS Azure, Oracle Cloud Infrastructure (OCI), IBM Cloud, Alibaba Cloud etc.
- Some platforms are different. They run on the top of hardware based platform.
- Example:
  - Java, Microsoft .NET
- Java language is considered as technology as well as platform.

Framework

- Well structured and well designed set of tools, libraries, and best practices that provide a foundation for building software application more efficiently, is called as framework.
- Example(s)
  - JNI, Struts, Spring, Hibernate, JUnit, Log4j, Django etc

Classification of languages

- Machine Level Programming Language
- Low Level Programming Language
- High Level Programming Language

Programming Paradigms.

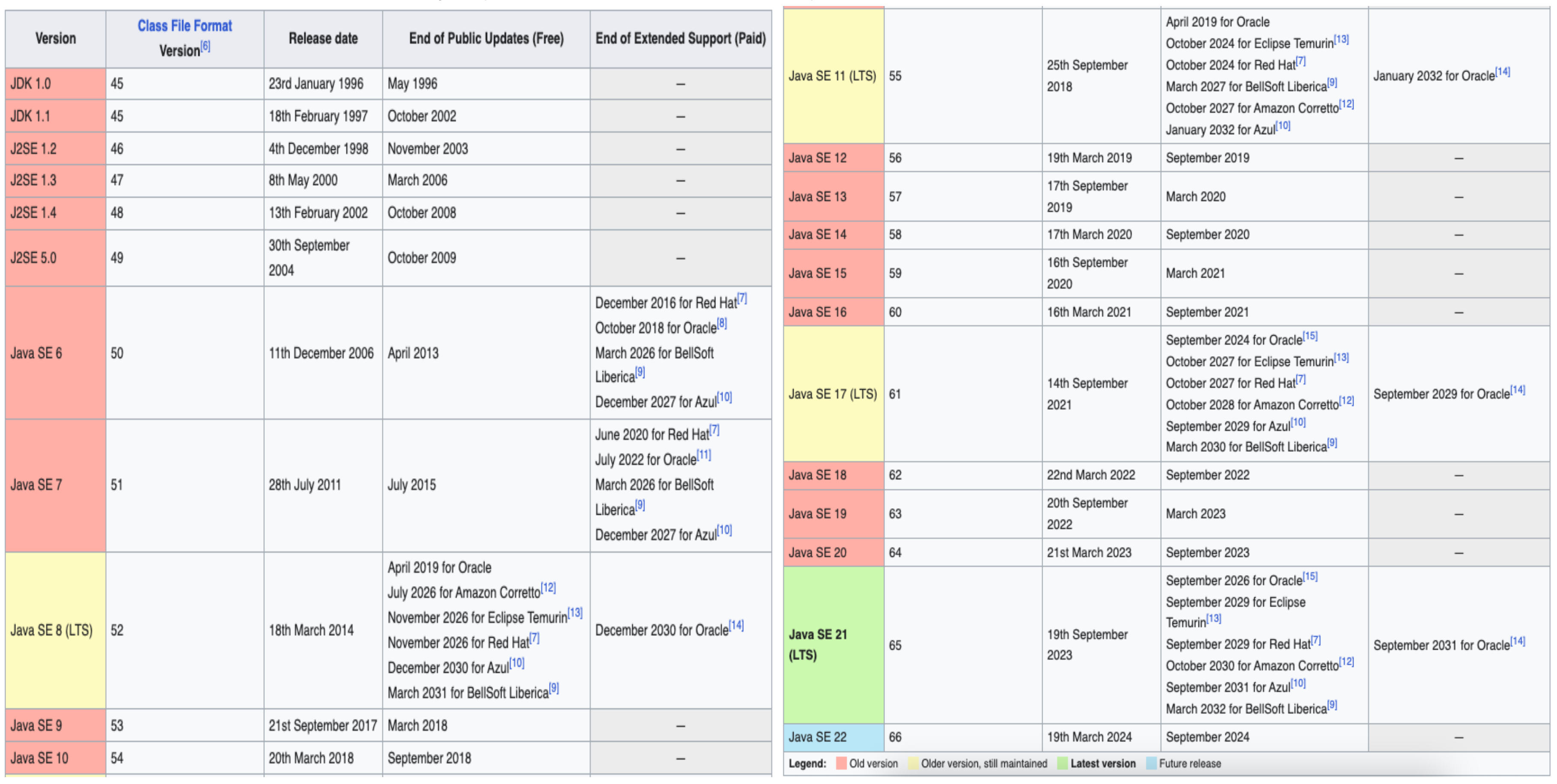
- Procedure Oriented Programming
- Object Oriented Programming
- Functional Programming

A Brief History of the Java Programming Language

- **Origins**: Back in 1991, engineers at Sun Microsystems wanted to create a simple computer language for devices like cable TV boxes. They needed a language that was small, efficient, and could work on different types of devices.
- **The Green Project**: They started a project called **"Green"** and were inspired by earlier attempts with Pascal, a computer language designed for portability. They used a similar approach of creating a virtual machine, which could run code on any device with the right interpreter.
- **Development of Oak**: Instead of Pascal, they based their language on C++ and made it object-oriented. The lead engineer, **James Gosling**, named the language "Oak" because of an oak tree outside his window. But later, they changed the name to Java.
- **Early Attempts**: In 1992, they created their first product called "*7," a smart remote control. Unfortunately, it didn't attract much interest. They tried to market their technology to other companies but didn't find success.
- **Internet Growth**: Meanwhile, the internet was expanding rapidly, and browsers were becoming crucial. In 1994, a browser called **Mosaic** was popular, but there was room for innovation.
- **The Birth of HotJava**: Realizing the potential, the Java team decided to create their own browser, called **HotJava**. It was not only a browser but also capable of running small programs called **applets** directly in web pages.
- **Sun Releases Java**: The success of HotJava led Sun to release the first version of Java in 1996.
- **Oracle Corporation**: Sun Microsystems was the original creator of Java. In 2010, It was acquired by Oracle Corporation.
- **"Write Once, Run Anywhere(WORA)"**: This slogan highlights Java's platform independence, meaning that Java programs can run on any device or operating system that has a Java Virtual Machine (JVM) installed.

The Java Version History

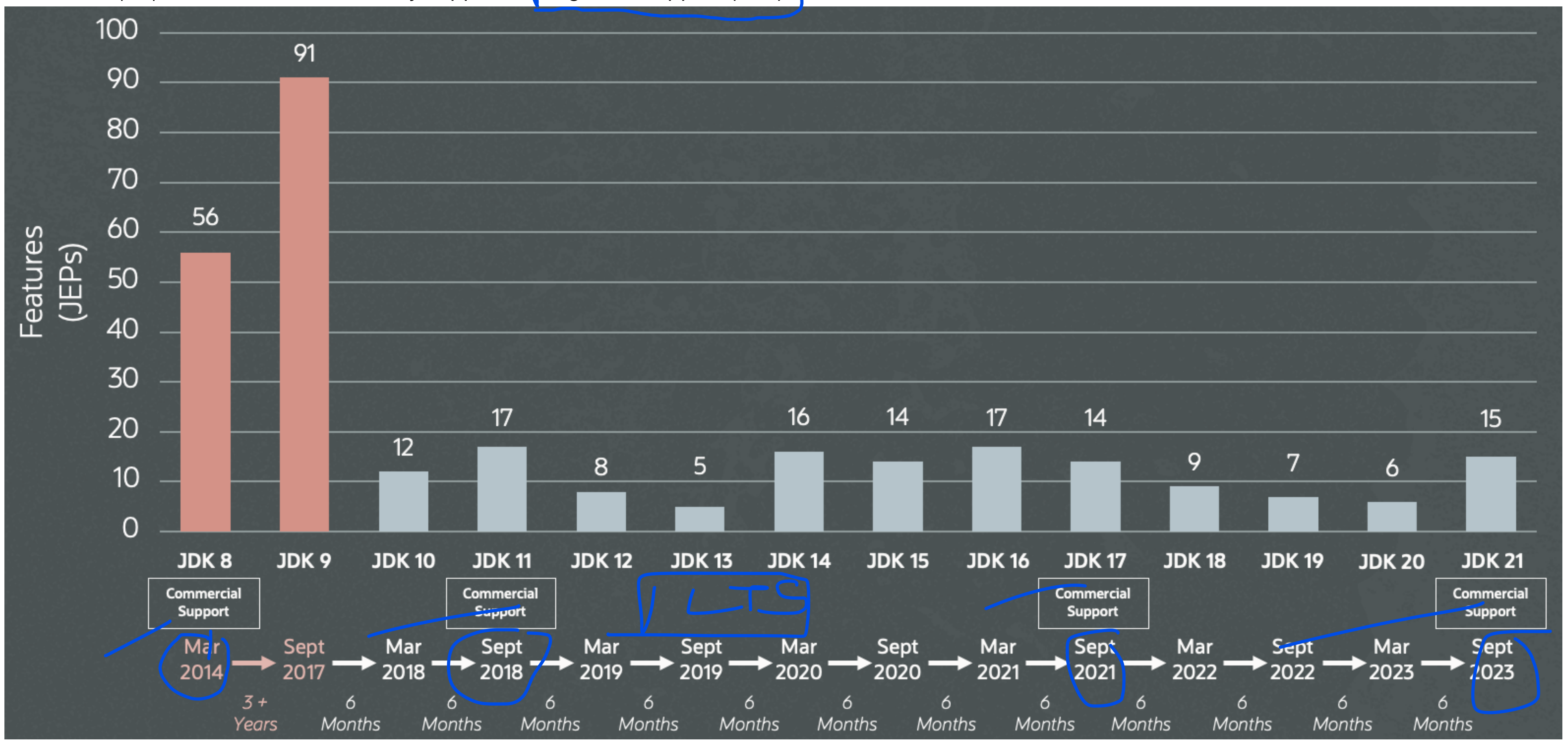
- The first version was released on January 23, 1996. The first stable version, JDK 1.0.2 is called Java 1.



Version	Class File Format Version[6]	Release date	End of Public Updates (Free)	End of Extended Support (Paid)
JDK 1.0	45	23rd January 1996	May 1996	–
JDK 1.1	45	18th February 1997	October 2002	–
J2SE 1.2	46	4th December 1998	November 2003	–
J2SE 1.3	47	8th May 2000	March 2006	–
J2SE 1.4	48	13th February 2002	October 2008	–
J2SE 5.0	49	30th September 2004	October 2009	–
Java SE 6	50	11th December 2006	April 2013	December 2016 for Red Hat[7] October 2018 for Oracle[8] March 2026 for BellSoft Liberica[9] December 2027 for Azul[10]
Java SE 7	51	28th July 2011	July 2015	June 2020 for Red Hat[7] July 2022 for Oracle[11] March 2026 for BellSoft Liberica[9] December 2027 for Azul[10]
Java SE 8 (LTS)	52	18th March 2014	April 2019 for Oracle July 2026 for Amazon Corretto[12] November 2026 for Eclipse Temurin[13] November 2026 for Red Hat[7] December 2030 for Azul[10] March 2031 for BellSoft Liberica[9]	December 2030 for Oracle[14]
Java SE 9	53	21st September 2017	March 2018	–
Java SE 10	54	20th March 2018	September 2018	–
Java SE 11 (LTS)	55	25th September 2018	April 2019 for Oracle October 2024 for Eclipse Temurin[13] October 2024 for Red Hat[7] March 2027 for BellSoft Liberica[9] October 2027 for Amazon Corretto[12] January 2032 for Azul[10]	January 2032 for Oracle[14]
Java SE 12	56	19th March 2019	September 2019	–
Java SE 13	57	17th September 2019	March 2020	–
Java SE 14	58	17th March 2020	September 2020	–
Java SE 15	59	16th September 2020	March 2021	–
Java SE 16	60	16th March 2021	September 2021	–
Java SE 17 (LTS)	61	14th September 2021	September 2024 for Oracle[15] October 2027 for Eclipse Temurin[13] October 2027 for Red Hat[7] October 2028 for Amazon Corretto[12] September 2029 for Azul[10] March 2030 for BellSoft Liberica[9]	September 2029 for Oracle[14]
Java SE 18	62	22nd March 2022	September 2022	–
Java SE 19	63	20th September 2022	March 2023	–
Java SE 20	64	21st March 2023	September 2023	–
Java SE 21 (LTS)	65	19th September 2023	September 2026 for Oracle[15] September 2029 for Eclipse Temurin[13] September 2029 for Red Hat[7] October 2030 for Amazon Corretto[12] September 2031 for Azul[10] March 2032 for BellSoft Liberica[9]	September 2031 for Oracle[14]
Java SE 22	66	19th March 2024	September 2024	–

Legend: 🟥 Old version 🟨 Older version, still maintained 🟩 **Latest version** 🟦 Future release

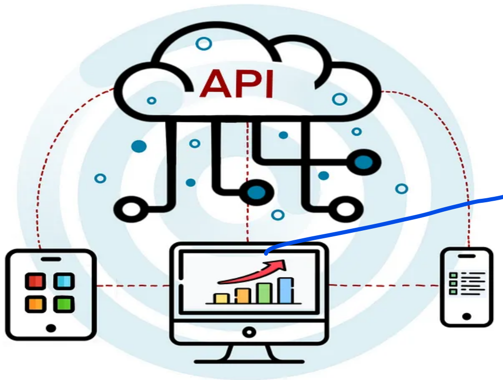
- In September 2017, Mark Reinhold, chief Architect of the Java Platform, proposed to change the release train to **"one feature release every six months"** rather than the then-current two-year schedule.
- Versions 21, 17, 11 and 8 are the currently supported long-term support (**LTS**) versions.



- In 1997, Sun Microsystems approached the ISO/IEC JTC 1 standards body and later the ECMA International to formalize Java, but it soon withdrew from the process. Java remains a de facto standard, controlled through the Java Community Process(**JCP**).

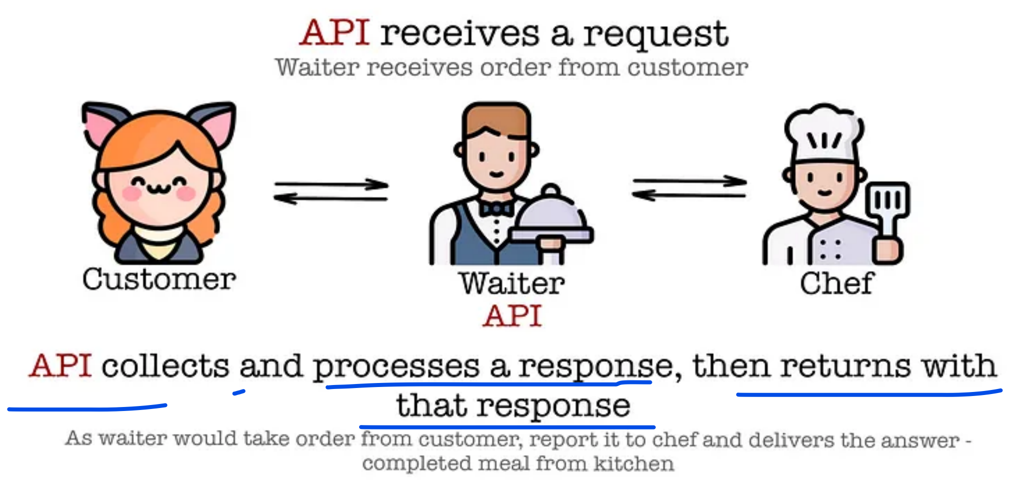
API

- An application programming interface(API) is a connection between computers or between computer programs.

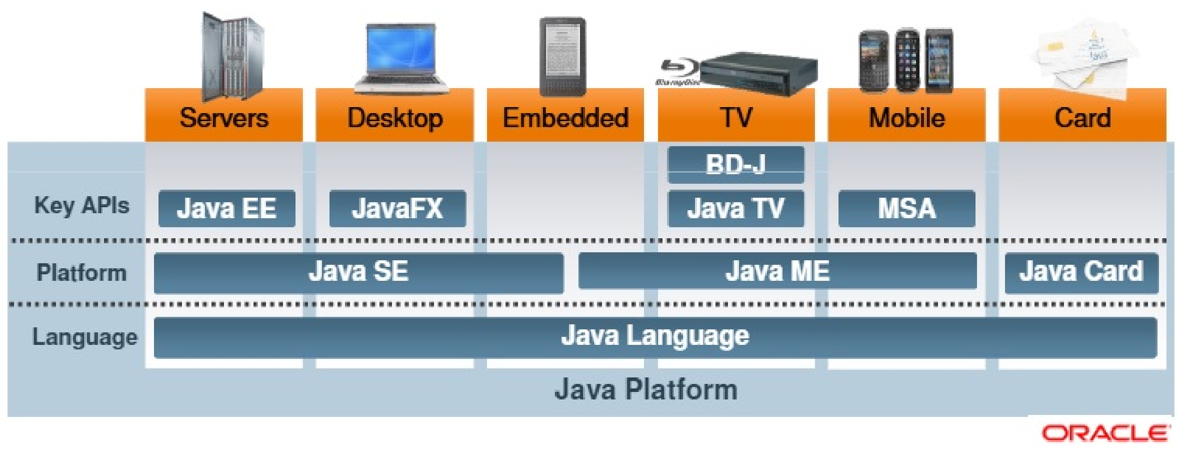


API.

- In a more simple way — API is a set of ways and rules for interaction and data exchange between different programs and computers.
- Example: You are in a restaurant. You are a client, you choose an order from the menu. The kitchen is the executor of your order. You need an intermediary who will report the order to the kitchen and delivery your food to your table. It could not be a chef because he is busy cooking in the kitchen. You need someone to connect the client and the chef. And here the waiter could help us — API.
- The waiter takes your order, report it to the kitchen and then delivers the answer, or food to you.


API receives a request
Waiter receives order from customer
Customer
Waiter
API
Chef
API collects and processes a response, then returns with that response
As waiter would take order from customer, report it to chef and delivers the answer - completed meal from kitchen

The Java Development Platforms



- There are four platforms of the Java programming language:
  - Java Platform, Standard Edition (Java SE)
  - Java Platform, Enterprise Edition (Java EE)
  - Java Platform, Micro Edition (Java ME)
  - JavaFX
- All Java platforms consist of a Java Virtual Machine (VM) and an application programming interface (API).
- The Java Virtual Machine is a program, for a particular hardware and software platform, that runs Java technology applications.
- An API is a collection of software components that you can use to create other software components or applications.

Java SE

- Java Platform, Standard Edition. It is also called as Core Java
- When most people think of the Java programming language, they think of the Java SE API.
- Java SE's API provides the core functionality of the Java programming language. It defines everything from the basic types and objects of the Java programming language to high-level classes that are used for networking, security, database access, graphical user interface (GUI) development, and XML parsing.

Java EE

- Java Platform, Enterprise Edition. It is also called as Advanced Java/Web Java/Java EE/ JEE.
- Formerly it is called as Java 2 Platform, Enterprise Edition (J2EE).
- The Java EE platform is built on top of the Java SE platform. In other words, The Java EE API is a superset of the Java SE API.
- The Java EE platform provides an API and runtime environment for developing and running large-scale, multi-tiered, scalable, reliable, and secure network applications.
- Now days it is called as Jakarta EE.

Java ME

- Java Platform, Micro Edition.
- The Java ME platform provides an API and a small-footprint virtual machine for running Java programming language applications on small devices, like mobile phones.
- The API is a subset of the Java SE API, along with special class libraries useful for small device application development.
- Java ME applications are often clients of Java EE platform services.

JavaFX

[GUI]

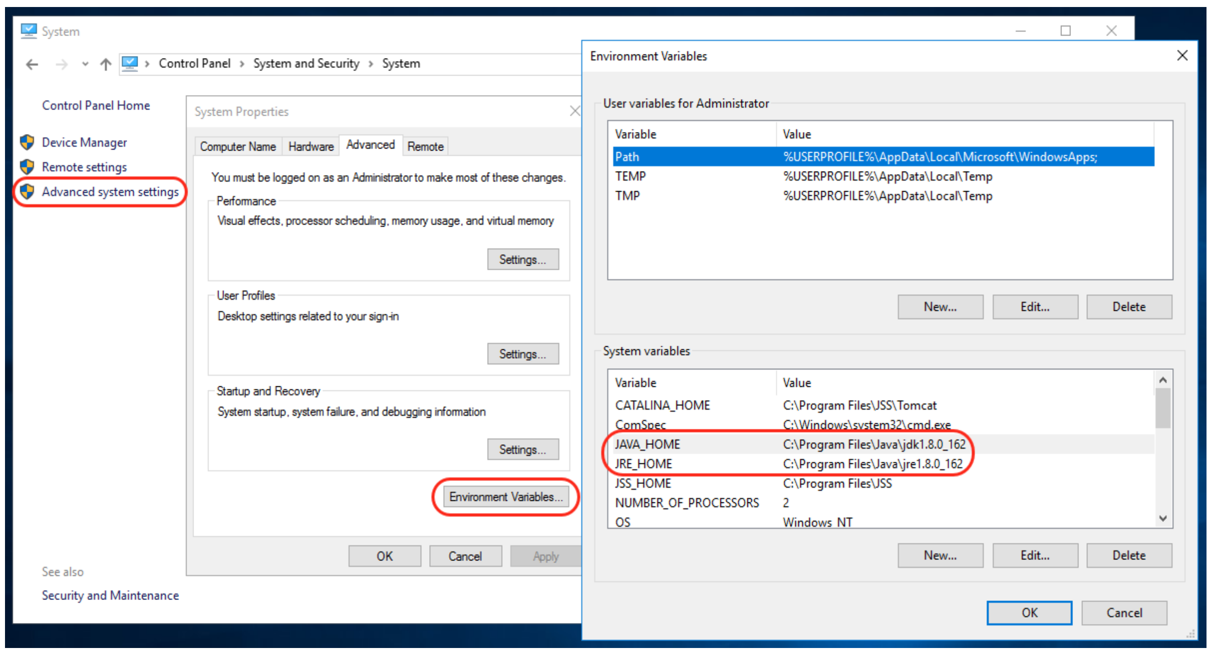
- JavaFX is a platform for creating rich internet applications using a lightweight user-interface API.
- JavaFX applications use hardware-accelerated graphics and media engines to take advantage of higher-performance, modern-looking clients and high-level APIs to connect to networked data sources.
- JavaFX applications may be clients of Java EE platform services.

JDK installation

- If you have other JDK version installed on your machine, keep as it is.
- Download "Java SE Development Kit 8u401" from below location and install it.
  - https://www.oracle.com/in/java/technologies/downloads/#java8
    - Java SE subscribers have more choices
      - Java 8
        - Windows
          - jdk-8u401-windows-x64.exe
- **Note:**Don't change default path.
- During installation, observe Java home and JRE home installation path.
- Once installation is done explore home directories.

Java Home, JDK Home and JRE Home

- **Java Home**
  - The "Java Home" directory refers to the root directory where your Java installation is located.
  - For Java 8 on Windows, it's often set to something like C:\Program Files\Java\jdk1.8.0.
- **JDK Home**
  - The "JDK Home" directory specifically refers to the directory where the Java Development Kit (JDK) is installed.
  - In Java 8 on Windows, the JDK home directory is usually the same as the Java home directory mentioned above (C:\Program Files\Java\jdk1.8.0)
- **JRE Home**
  - The "JRE Home" directory refers to directory that contains the Java Runtime Environment (JRE).
  - In Java 8 on Windows, it's often set to something like C:\Program Files\Java\jre1.8.0.
- JAVA_HOME and JRE_HOME are environment variables used to specify the location of the Java Development Kit (JDK) and the Java Runtime Environment (JRE) respectively. These variables are commonly used in various Java applications and tools to locate the Java installation directory.
- How can we set it?



You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles
Desktop settings related to your sign-in

Startup and Recovery
System startup, system failure, and debugging information

User variables for Administrator

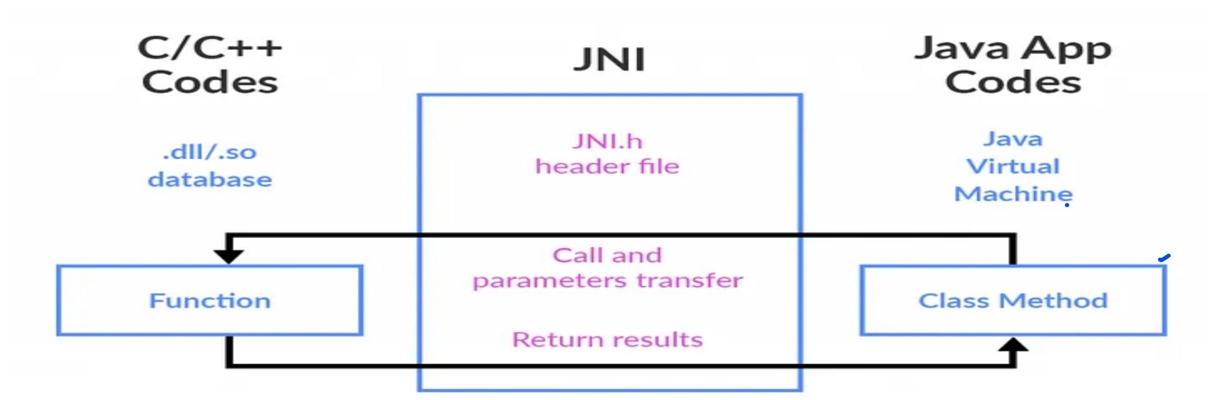
Variable	Value
Path	%USERPROFILE%\AppData\Local\Microsoft\WindowsApps;
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
CATALINA_HOME	C:\Program Files\JSS\Tomcat
ComSpec	C:\Windows\system32\cmd.exe
JAVA_HOME	C:\Program Files\Java\jdk1.8.0_162
JRE_HOME	C:\Program Files\Java\jre1.8.0_162
JSS_HOME	C:\Program Files\JSS
NUMBER_OF_PROCESSORS	2
OS	Windows_NT

Java Home Installation Directory Structure

- **bin:**
  - This folder contains executable files of Java development tools.
  - Some of the key tools are:
    - javac
    - java
    - jar
    - javap
- **include:**
  - JNI stands for Java Native Interface.
  - It's a framework that allows Java code running in the Java Virtual Machine (JVM) to interact with code written in other programming languages, particularly C and C++.
  - This directory contains header files required for developing native methods using the Java Native Interface (JNI).
  - Example:
    - jni.h
    - jvmti.h



- **lib:**
  - It contains files used by the Java development tools which is available in jdk/bin.
  - Example:
    - tools.jar
    - dt.jar
    - packager.jar
- **jre:**
  - Root directory of the Java Runtime Environment (JRE) used by the JDK development tools.
  - This directory is typically present in the JDK installation for convenience, especially when testing Java applications.
  - It includes only the essential components needed to run Java applications:
    - rt.jar
      - File which contains all core Java API.
    - Java Virtual Machine
      - A runtime environment that executes Java bytecode, enabling Java programs to run on any device or operating system.
- **src.zip:**
  - Java is open source technology. It means that source code of Java API is available for free. We can use it for customization or understanding the internals.
  - src.zip file contains source code of Java API. We can extract it and read the source code of any API using text editor.
- **docs:**
  - This directory contains Java API documentation files.
  - By default this directory does not exist. We need to download and extract it from below location:
    - https://www.oracle.com/in/java/technologies/javase-jdk8-doc-downloads.html
  - We can access Java API Docs online too so no need to download:
    - https://docs.oracle.com/javase/8/docs/api/

Simple Hello Application

- Create a file "Program.java" on Desktop and write below code inside it. Note: Code is case sensitive.

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

- Open terminal/command prompt and set path to get access of all Java development tools:
  - Note you can set path from anywhere.
  - In Windows:
    - set path="C:\Program Files\Java\jdk1.8\bin" (check path of bin in C drive) or
    - set path=drag and drop bin folder here.
  - Linux/Mac OS:
    - export PATH=/usr/bin
- If you get below error then you should set path properly:
  - javac is not recognized as an internal or external command, operable program or batch file.
- Compile the code

```
  C:\Users\Profile_Name\Desktop> javac Program.java  //Output: Program.class
```

- Execute the code

```
C:\Users\Profile_Name\Desktop> java Program  //Output: Hello World!!
```

Java Language Changes

- https://docs.oracle.com/en/java/javase/21/language/java-language-changes.html