✅ Core Java Learning Path for Selenium Automation Testers

# 1. Java Basics

## 1.1 Introduction to Java

* **Java** is an object-oriented, class-based, high-level programming language.
* Originally developed by *Sun Microsystems*, now owned by *Oracle*.
* **Write Once, Run Anywhere (WORA)** – thanks to Java Virtual Machine (JVM).
* Widely used in **enterprise applications**, **mobile development**, and **test automation (Selenium)**.

**🔧 Real-world Use Case in Selenium:** Java powers the **Selenium WebDriver** API, enabling browser automation in a platform-independent way.

## 1.2 JDK, JRE, JVM

|  |  |  |
| --- | --- | --- |
| Component | Full Form | Description |
| JDK | Java Development Kit | Includes JRE + compilers + tools to develop Java apps |
| JRE | Java Runtime Environment | Runs Java programs (JVM + libraries) |
| JVM | Java Virtual Machine | Converts .class bytecode into OS-specific machine code |

## 1.3 How Java works

## 1.4 Java Syntax and Structure

## 1.5 Data Types and Variables

## 1.6 Operators

# 2. Conditional Statements

## 2.1 if, if-else, if-else-if

## 2.2 switch

## 2.3 Best practices

# 3. Loops

## 3.1 for, while, do-while

## 3.2 for-each

## 3.3 Loop control: break, continue

# 4. Arrays

## 4.1 One-dimensional arrays

## 4.2 Two-dimensional arrays

## 4.3 Iteration and manipulation

## 4.4 Arrays vs Collections

# 5. Strings

## 5.1 String creation & comparison

## 5.2 String methods

## 5.3 StringBuilder vs StringBuffer

## 5.4 String use in validations

## 5.5 Regular Expressions

# 6. Type Casting

## 6.1 Implicit (Widening) and Explicit (Narrowing)

## 6.2 Object type casting

## 6.3 Upcasting and Downcasting

## 6.4 Selenium use cases (TakesScreenshot, JavascriptExecutor)

# 7. Object-Oriented Programming (OOPs)

## 7.1 Classes and Objects

## 7.2 Inheritance

## 7.3 Polymorphism

### 7.3.1 Method Overloading

### 7.3.2 Method Overriding

## 7.4 Encapsulation

## 7.5 Abstraction

### 7.5.1 Abstract classes

### 7.5.2 Interfaces (e.g., WebDriver)

# 8. Constructors

## 8.1 Default and Parameterized constructors

## 8.2 Constructor overloading

## 8.3 this() and super()

## 8.4 Usage in POM (Page Object Model)

# 9. Access Modifiers

## 9.1 private, default, protected, public

## 9.2 Scope of access and impact

# 10. Static & Final Keywords

## 10.1 Static methods/variables

## 10.2 Static blocks

## 10.3 Final variables, methods, classes

# 11. Collections Framework

## 11.1 List: ArrayList, LinkedList

## 11.2 Set: HashSet, TreeSet

## 11.3 Map (Optional but helpful): HashMap

## 11.4 Iterator and for-each

## 11.5 Conversions: Arrays ↔ List

## 11.6 Real-time usage with Selenium WebElements

# 12. Exception Handling

## 12.1 try-catch-finally

## 12.2 throw vs throws

## 12.3 Checked vs Unchecked

## 12.4 Custom exceptions

## 12.5 Selenium Exception types

# 13. File Handling

## 13.1 File, FileReader, BufferedReader

## 13.2 FileWriter, BufferedWriter

## 13.3 Working with .properties files

## 13.4 Apache POI for Excel automation

# 14. Java 8 Features ⭐ Important for Frameworks

## 14.1 Lambda Expressions

## 14.2 Streams API

## 14.3 Functional Interfaces

## 14.4 Method references

## 14.5 Real-time Selenium usage

# 15. Multithreading (Optional but Bonus)

## 15.1 Basics of threads

## 15.2 Runnable and Thread

## 15.3 synchronized keyword

## 15.4 Thread safety in parallel test execution

# 16. Code Optimization

## 16.1 Clean code practices

## 16.2 Utility class design

## 16.3 DRY principle

## 16.4 Logging strategies (Log4j, ExtentReports)