**Java**

**Basics of Java**

**What is Java?**

* Java is high level programming language
* Robust and Object-Oriented Language
* Secure language
* Developed by sun Microsystems which is subsidiary of Oracle in 1995
* Platform in-dependent (Ex: write once run anywhere)

**Types of Java Application**

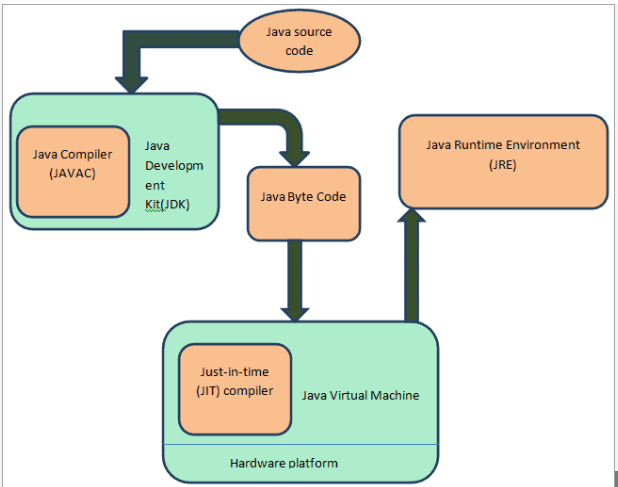
* Standalone – Desktop based applications (AWT, Swing)
* Web – Server and web browser based (Servlet, Spring, Hibernate)
* Mobile
* Enterprise – large distributed system and business solutions (Ex: Banking, CRM)

**Java Platform/Editions**

* Standard- (Oops, IO stream, Collections, Exception)
* Enterprise (Servlet, JSP, Web services, EJB)
* Micro (Mobile application like Android)

**JDK, JRE, JVM**

* JDK – Java development toolkit it’s provide platform to develop and compile the java Program
* JRE – Java Runtime Environment It’s provided library, JVM at run time
* JVM- Java Virtual Machine It’s done some operation like load code, verify code, execute code, provide runtime environment



**Types of Variables**

* Local
* Instance
* Static
* Argument
* Final

A screenshot of a computer

Description automatically generated

**Data Types**

A diagram of a data type

Description automatically generated

**Operators**

1.Airthmetic

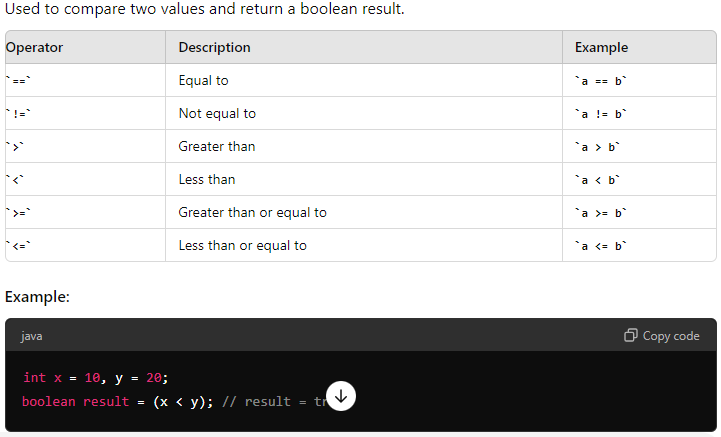
A screenshot of a computer

Description automatically generated

A black rectangular object with white text

Description automatically generated

2.Relational

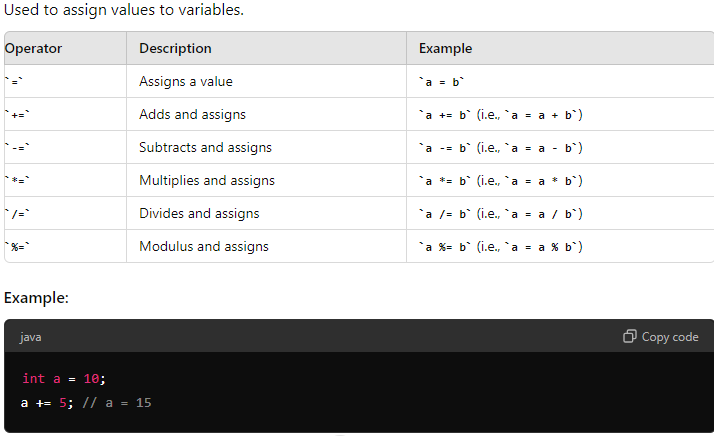


3.Logical

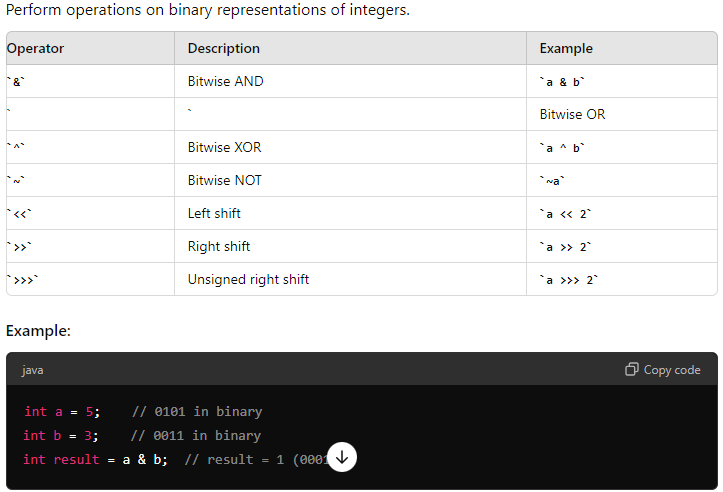
A screenshot of a computer

Description automatically generated

4.Assignment



5.Bitwise



6.Ternary

A screenshot of a computer

Description automatically generated

7.Type casting

A screenshot of a computer

Description automatically generated

9.Unary

A screenshot of a computer

Description automatically generated

10.Shift operator

A screenshot of a computer

Description automatically generated

**Access Modifier**

* Private – access only within the same class, can’t modify values from outside class
* Protected – accessible within the package to their subclass (even if they are in different package)
* Public – accessible with any class or package
* Default – accessible within the same package

**Types of Java Comments**

* Single line (// - specific code details)
* Multi-line (/\* ... \*/ - multiple line of code details
* Documentation (/\*\* ... \*/ - include classes, methods and fields)

**Constructors**

* Default
* Parameterized

Default

* If constructors not defined, then by default java will create default constructors

A screenshot of a computer program

Description automatically generated

Parameterized

* It allows to pass the values at the time of object creation, Initializing object with specific data



Example in selenium

A computer screen shot of a program

Description automatically generated

Object oriented Programing

**Topics in Oops**

* Polymorphism
* Inheritance
* Encapsulation
* Interface
* Abstraction

**Polymorphism**

* Compile time / static – Method overloading
* Run time / dynamic – method overriding

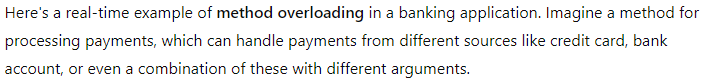
Method Overloading

* Different methods can share same name but different parameters

A computer screen with text on it

Description automatically generated

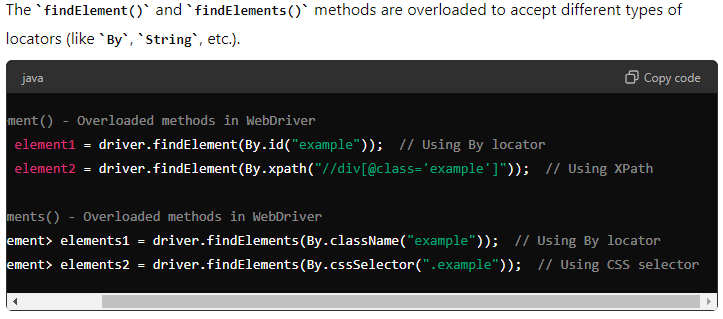
Example

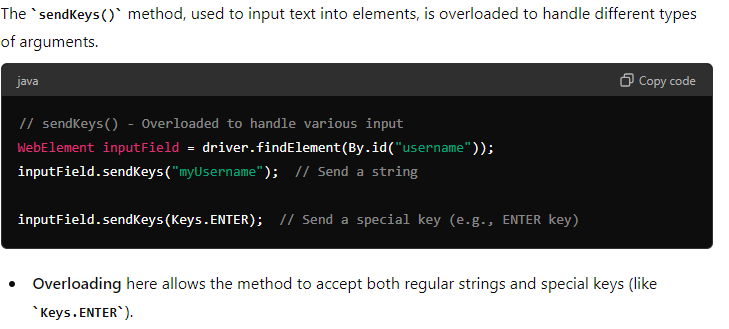


A screenshot of a computer program

Description automatically generated

Examples in Selenium





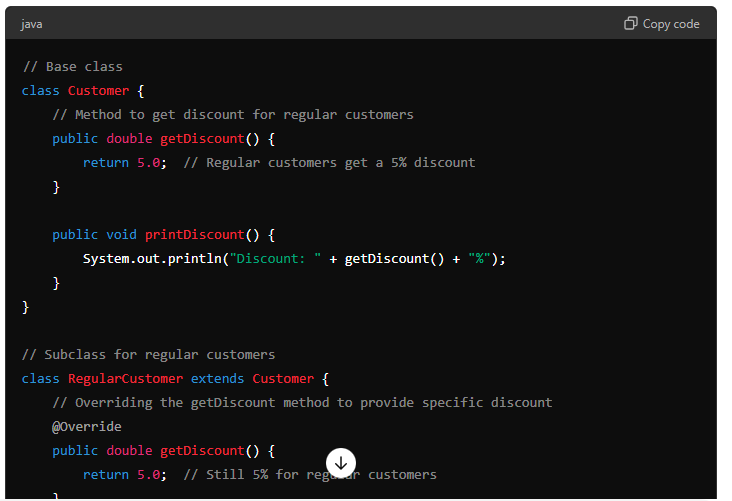
Method Overriding

* Subclass provides specific implementation of methods that is already defined by super class



A white background with black text

Description automatically generated



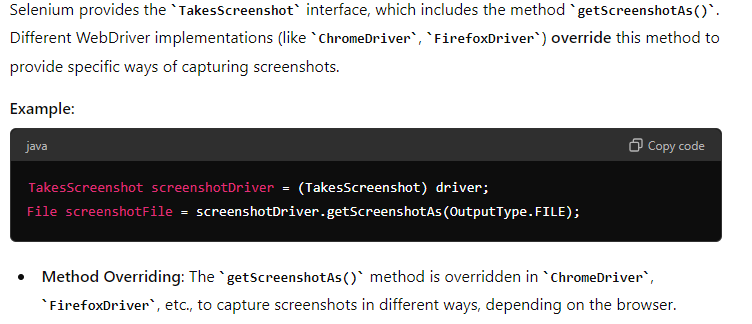
A screenshot of a computer program

Description automatically generated

Example in selenium

A screenshot of a computer

Description automatically generated



Key notes

* Main method we can overload but can’t override
* Static method we can overload but can’t override

A computer code with text

Description automatically generated with medium confidence

**Inheritance**

* Single
* Multiple
* Multilevel
* Hierarchical
* Hybrid

A diagram of a class

Description automatically generated

A diagram of a class

Description automatically generated

Key Notes

* Multiple inheritance does not support in class, supported in interface

**Encapsulation**

* Wrapping the data into single unit
* Variables are in private/protected with accessible public methods
* Restricted unauthorized access from external system



Examples

A screenshot of a computer screen

Description automatically generated

Example in selenium

A white background with black text

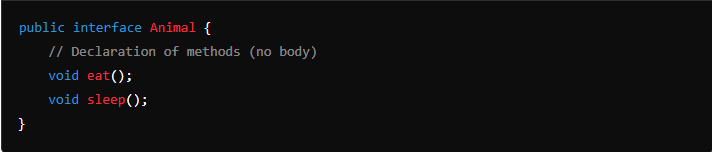
Description automatically generated

A computer screen shot of a program

Description automatically generated

**Interface**

* Interface promotes loose coupling and code flexibility by allowing different classes to implement the same set of methods in their own way
* Easy maintenance and scalability.



A computer screen with text on it

Description automatically generated

Examples

A screenshot of a computer

Description automatically generated

A screen shot of a computer code

Description automatically generated

Examples in selenium

A screenshot of a computer

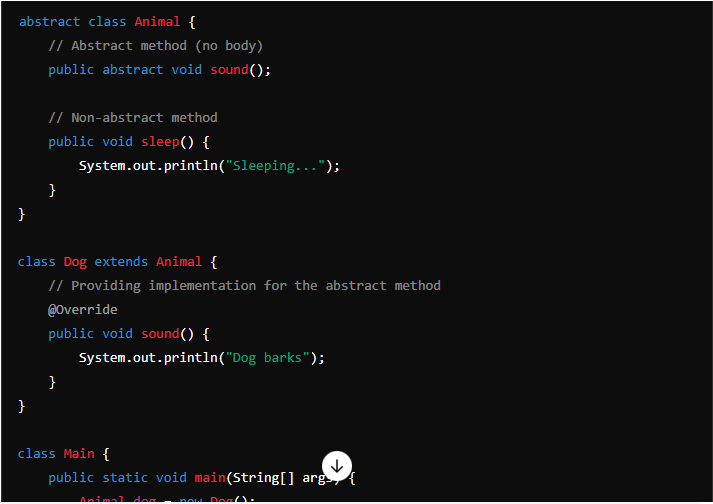
Description automatically generated

Key Notes

* No implementation (pure abstract)
* Java 8 and above interface have static and default method with implementation
* Support multiple inheritance
* By default, methods are “public abstract” void
* By default, variables are “public static final”
* Interface doesn’t have constructors and can’t instantiated directly
* If one abstract method is in interface, then it’s called as functional interface (@FunctionalInterface)

**Abstraction**

* It’s hiding the complex implementation details of a system and exposing only the essential features behaviors that are necessary to use.



A computer screen with white text

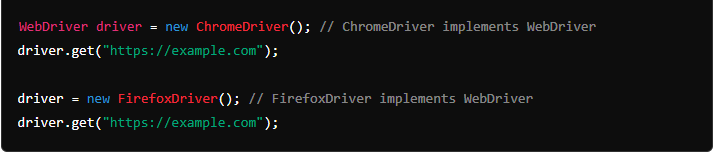
Description automatically generated

Examples

A white text on a white background

Description automatically generated

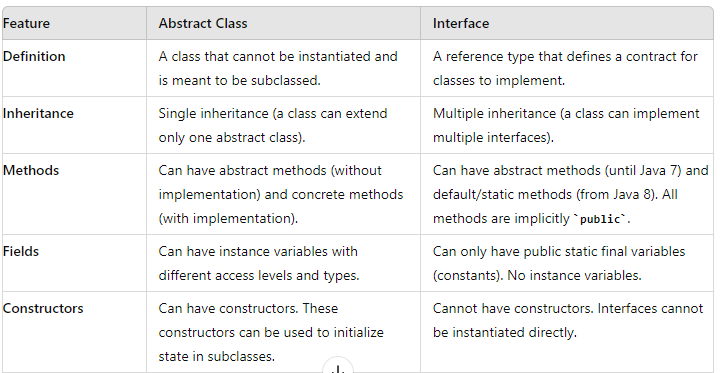
Examples in selenium



Key Notes

* Hiding the complex implementation
* Abstract class can have both abstract and non-abstract methods
* Abstraction achieved through abstract class and interface
* Provide security
* Encourage reusability
* Can’t extends multiple abstract class because on Multiple inheritance
* Can’t instantiate for abstract class

Diff b/w abstract class and interface



A white background with black text

Description automatically generated

**Type casting**

* Widening
* Narrowing
* Upcasting
* Down casting

Widening (Implicit)

* Small data type to large data type conversion
* Automatic conversion because small to large data type conversion

Example

A computer code on a black background

Description automatically generated

Narrowing (Explicit)

* Converting large data type to small data type conversion
* This will be done by manually

Example

A black rectangular object with white text

Description automatically generated

A computer code on a black background

Description automatically generated

Upcasting (Implicit)

* Casting to subclass object to its superclass type

Example

A computer screen shot of a program code

Description automatically generated

Down casting (Explicit)

* Casting superclass reference back to subclass reference
* Done manually, if not casting applied “classcastexception” will throw at run time.

Example

A computer screen with white text

Description automatically generated

**Keywords**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a phone

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Java Strings**

**What is String?**

* It’s a primitive data type
* Java.lang package
* String we can use as a variable and class

**String Creation**

* Literals
* New keyword

Literals

* String we can declare as literals(variables)

Example

A green and blue circle with text

Description automatically generated

New Keyword

* String objects are stored in heap memory

Example

A black background with white text

Description automatically generated

**String Methods**

* charAt
* compareTo
* concat
* contains
* endsWith
* equals
* equalsIgnoreCase
* indexOf
* intern
* isEmpty
* lastIndexOf
* length
* replace
* replaceAll
* split
* startsWith
* substring
* toCharArray
* toLowerCase
* toUpperCase
* trim
* valueOf

charAt



CompareTo

A screen shot of a computer program

Description automatically generated

Concat

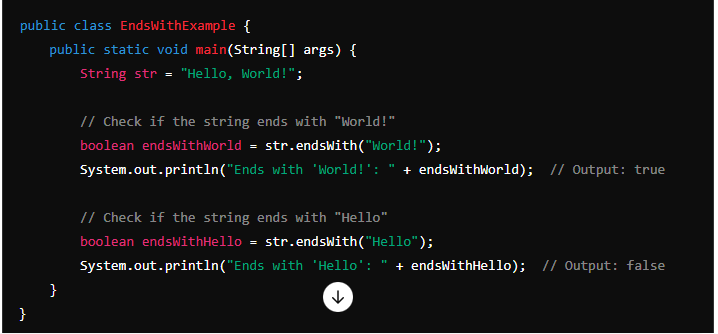
A computer screen with white text

Description automatically generated

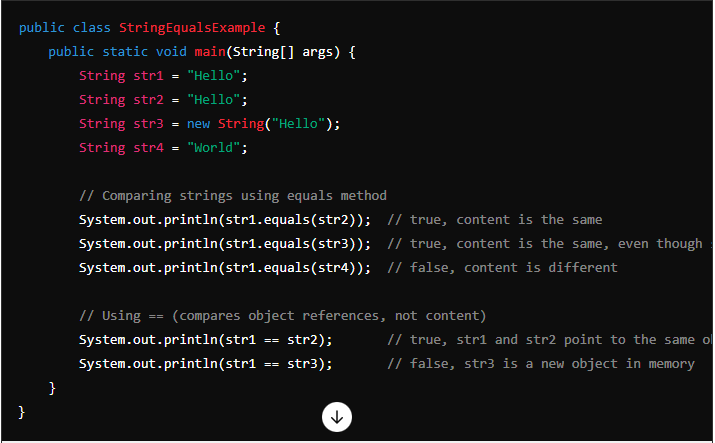
Contains



endsWith



Equals



equalsIgnoreCase

A screen shot of a computer

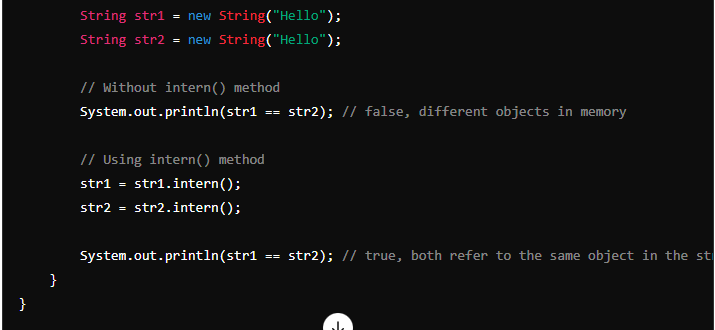
Description automatically generated

indexOf

A computer screen shot of a program

Description automatically generated

Intern



isEmpty

A screen shot of a computer program

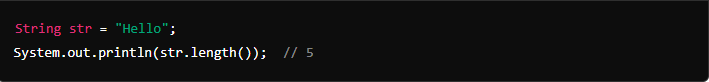
Description automatically generated

lastIndexOf

A screenshot of a computer program

Description automatically generated

Length



Replace

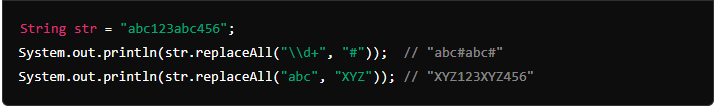
A black background with white text

Description automatically generated

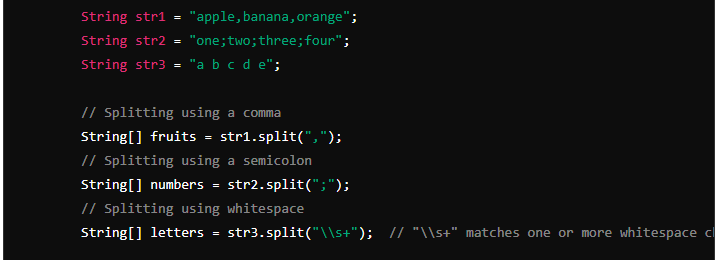
replaceAll

A screenshot of a computer program

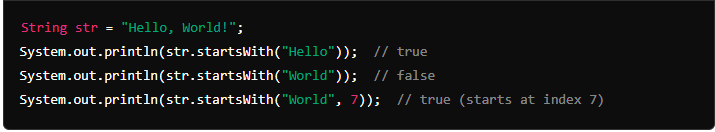
Description automatically generated



Split



startsWith



Substring

A screen shot of a computer program

Description automatically generated

toLowerCase

A black screen with white text

Description automatically generated

ToUpperCase

A black background with white text

Description automatically generated

Trim

A screen shot of a computer program

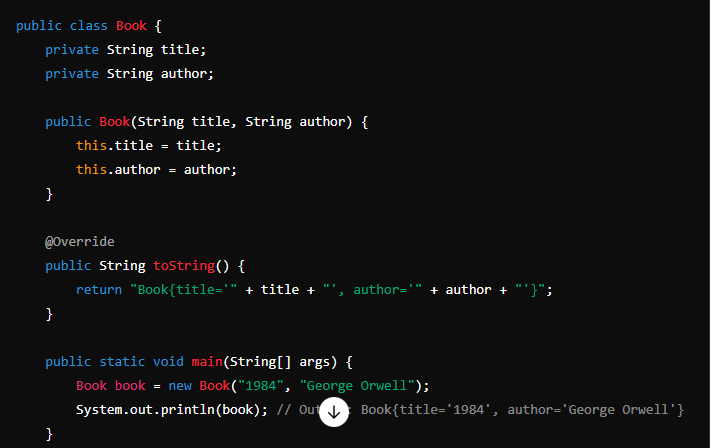
Description automatically generated

valueOf

A black screen with white text

Description automatically generated

toString



Key Notes

* Immutable - Once string object created can’t modified, if need to modify then create new string object
* Stored in string pool Memory & Heap Memory – String literals are stored in pool and objects are stored in heap memory