Object Oriented Programming – Java

# Required Software

* Java SDK: <https://www.oracle.com/in/java/technologies/javase-downloads.html>
* IntelliJ IDEA: <https://www.jetbrains.com/idea/>
* Git: <https://git-scm.com/downloads>

# Class and Object

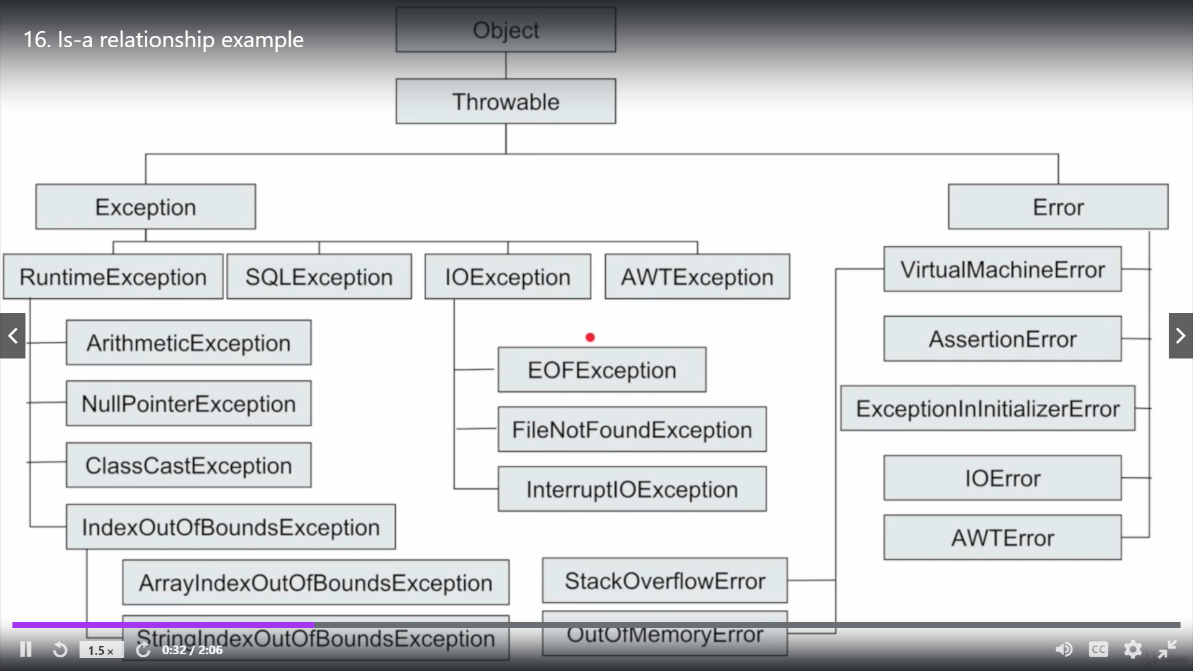
* Object have **State** and **behavior**.
* Classes: Blueprint used to define objects, User Define Data type.
* Class can contain Variable, Methods, Interfaces, Member class, Constructors

# Inheritance

* Is-A Relationship OR Parent-Child Relationship
* Super Class, Parent Class, Base Class – Sub Class, Child Class, Derived Class.
* Parent reference can be used to hold child object(known as Polymorphism) and can’t call child specific methods, but child reference cannot be used to hold parent object
* Every Class in Java is Child of Object class

## Need of Inheritance

* Re usability and Simplicity



## Types of Inheritance

1. **Single Inheritance**: One class inherit only by one class
2. **Hierarchical Inheritance**: One class inherited by two or more classes
3. **Multilevel Inheritance**: One class inherit another class which is also inherit from third class
4. **Multiple Inheritance**: One class inherit from two or more class [**Not supported in Java**]
5. **Hybrid Inheritance**: Combination of above Inheritance types

**Has-A Relationship**

**Composition & Aggregation:**

In Composition Container & Contained Object strongly associated, And in Aggregation Container & Contained Object weakly/loosely associated.

# Polymorphism

Two Types

1. Compile time polymorphism/ static binding/ early binding. Eg. Overloading-Same name, different argument types.
2. Runtime Polymorphism/ dynamic binding/ late binding. E.g. Overriding

## Rules for Overloading

* Method can be overloaded by changing signature of methods
* Return type of methods is not part of method signature , so just changing the return type will not overload methods in java.
* Java performs **automatic promotion** when it can’t find a matching method

Char 🡪 int 🡪 long 🡪 float 🡪 double 🡪 thrown compile-time error

Byte 🡪 short 🡪 int 🡪 long 🡪 float 🡪 double 🡪 thrown compile-time error

* While resolving overloaded methods compiler will always give precedence for child type argument , then it will check for parent type argument

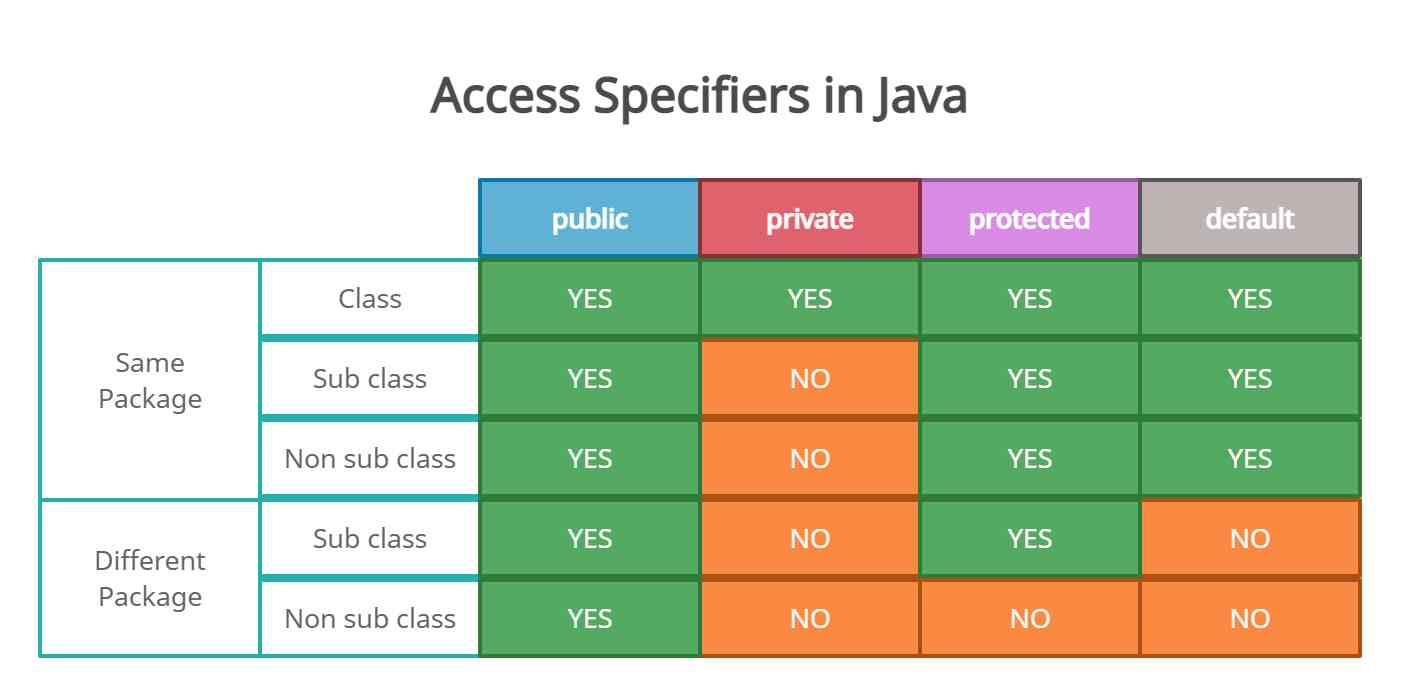
String 🡪 Object

* While resolving overloaded methods, if 2 parameters are at same level then there will be compile time error

String and StringBuffer are at same level. If ‘null’ is pass, it can be both so ambiguous argument

* In overloading, methods resolution always take care by compiler base on reference type.
* Var-argument methods will get least priority. i.e. If no other method matched then only var -arg method will get change (e.g. int … marks)

# Data Hiding



# Abstraction

* Hiding implementation level details of system from user
* Security
* Ease of implementation
* Improve maintainability
* Ease of use
* By using interfaces and abstract classes we can implement Abstraction in Java

# Encapsulation

Encapsulation = Data Hiding + Abstraction