MOOCS SEMINAR ON
Java Programming
SMC 101

Submitted by
Name:Sahil Rawat
STUDENT ID:2201334
ROLLNO.:28
COURSE: MCA 1 A
CAMPUS:Dehradun

Submitted to
Ms. Anupriya
(AP, SOC)

Graphic Era Hill University, Dehradun
School Of Computing
Department Of Computer Applications

CERTIFICATE

udemy

Certificate no: UC-6468de16-1d01-4c98-957f-b0874dec045e
Certificate url: ude.my/UC-6468de16-1d01-4c98-957f-b0874dec045e
Reference Number: 0004

CERTIFICATE OF COMPLETION

Java Programming: Complete Beginner to Advanced

Instructors **CodeIn Academy**

Sahil Rawat

Date **Jan. 8, 2023**
Length **7 total hours**

INDEX

Java Object Oriented

- Class
- Object
- Method
- Pillars of OOPs
    - Abstraction
    - Encapsulation
    - Inheritance
- Polymorphism
- Constructor In Java
- Interface
- Method Overloading
- Method Overriding

Exception Handling

- Hierarchy of Exception classes

INTRODUCTION

Java is one of the programming language or technology used for developing web applications.Java language developed at SUN Micro Systems in the year 1995 under the guidance of James

Gosling and there team. Originally SUN Micro Systems is one of the Academic university (Standford University Network)

Whatever the software developed in the year 1990, SUN Micro Systems has released on the name of oak, which is original name of java (scientifically oak is one of the tree name). The OAK has taken 18 months to develop.The oak is unable to fulfill all requirements of the industry. So James Gosling again reviews this oak and released with the name of java in the year 1995. Scientifically java is one of the coffee seed name.

The principles for creating java were simple, robust, secured, high performance, portable, multi-threaded, interpreted, dynamic, etc. In 1995 Java was developed by **James Gosling**, who is known as the Father of Java. Currently, Java is used in mobile devices, internet programming, games, e-business, etc.

BASICS OF JAVA

Overview Of Java

Java is a platform independent, more powerful, secure, high performance, multithreaded programming language. Here we discuss some points related to java.

Java Terminology

Before learning Java, one must be familiar with these common terms of Java.

1.Java Virtual Machine(JVM):

This is generally referred to as JVM. There are three execution phases of a program. They are written, compile and run the program.

Writing a program is done by a java programmer like you and me.

The compilation is done by the **JAVAC** compiler which is a primary Java compiler included in the Java development kit (JDK). It takes the Java program as input and generates bytecode as output.

In the Running phase of a program, **JVM** executes the bytecode generated by the compiler.

2.Java Development Kit(JDK):

While we were using the term JDK when we learn about bytecode and JVM. So, as the name suggests, it is a complete Java development kit that includes everything including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc. For the program to execute in java, we need to install JDK on our computer in order to create, compile and run the java program.

3.Java Runtime Environment (JRE):

JDK includes JRE. JRE installation on our computers allows the java program to run, however, we cannot compile it. JRE includes a browser, JVM, applet supports, and plugins. For running the java program, a computer needs JRE.

4. Garbage Collector:

In Java, programmers can't delete the objects. To delete or recollect that memory JVM has a program called Garbage Collector. Garbage Collectors can recollect the objects that are not referenced. So Java makes the life of a programmer easy by handling memory management. However, programmers should be careful about their code whether they are using objects that have been used for a long time. Because Garbage cannot recover the memory of objects being referenced.

5. ClassPath:

The classpath is the file path where the java runtime and Java compiler look for **.class** files to load. By default, JDK provides many libraries. If you want to include external libraries they should be added to the classpath.

6. Bytecode in the Development process:

As discussed, the Javac compiler of JDK compiles the java source code into bytecode so that it can be executed by JVM. It is saved as **.class** file by the compiler. To view the bytecode, a disassembler like javap can be used.

Features Of JAVA

1.Platform Independent:

Compiler converts source code to bytecode and then the JVM executes the bytecode generated by the compiler. This bytecode can run on any platform be it Windows, Linux, or macOS which means if we compile a program on Windows, then we can run it on Linux and vice versa. Each operating system has a different JVM, but the output produced by all the OS is the same after the execution of bytecode. That is why we call java a platform-independent language.

2. Object-Oriented Programming Language:

Organizing the program in the terms of collection of objects is a way of object-oriented programming, each of which represents an instance of the class.

The four main concepts of Object-Oriented programming are:

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism
-

3. Simple:

Java is one of the simple languages as it does not have complex features like pointers, operator overloading, multiple inheritances, and Explicit memory allocation.

4. Robust:

Java language is robust which means reliable. It is developed in such a way that it puts a lot of effort into checking errors as early as possible, that is why the java compiler is able to detect even those errors that are not easy to detect by another programming language. The main features of java that make it robust are garbage collection, Exception Handling, and memory allocation.

5. Secure:

In java, we don't have pointers, so we cannot access out-of-bound arrays i.e it shows **ArrayIndexOutOfBound Exception** if we try to do so. That's why several security flaws like stack corruption or buffer overflow are impossible to exploit in Java. Also java programs run in an environment that is independent of the os(operating system) environment which makes java programs more secure .

6. Distributed:

We can create distributed applications using the java programming language. Remote Method Invocation and Enterprise Java Beans are used for creating distributed applications in java. The java programs can be easily distributed on one or more systems that are connected to each other through an internet connection.

7. Multithreading:

Java supports multithreading. It is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of the CPU.

8. Portable:

As we know, java code written on one machine can be run on another machine. The platform-independent feature of java in which its platform-independent bytecode can be taken to any platform for execution makes java portable.

9. High Performance:

Java architecture is defined in such a way that it reduces overhead during the runtime and at some time java uses Just In Time (JIT) compiler where the compiler compiles code on-demand basics where it only compiles those methods that are called making applications to execute faster.

10. Dynamic flexibility:

Java being completely object-oriented gives us the flexibility to add classes, new methods to existing classes and even create new classes through sub-classes. Java even supports functions written in other languages such as C, C++ which are referred to as native methods.

11. Sandbox Execution:

Java programs run in a separate space that allows user to execute their applications without affecting the underlying system with help of a bytecode verifier. Bytecode verifier also provides additional security as its role is to check the code for any violation of access.

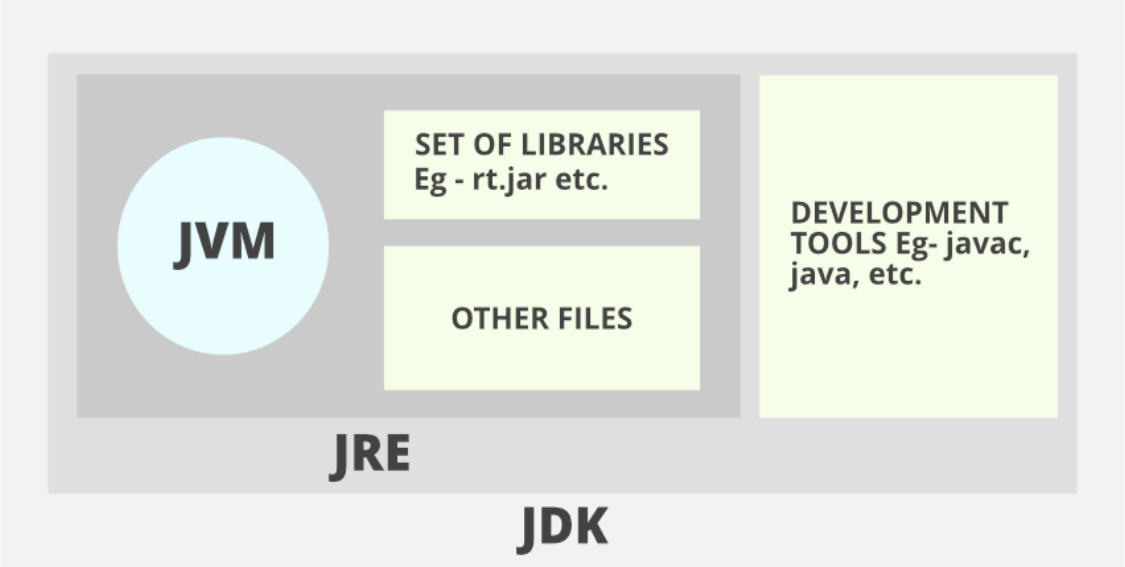
12. Write Once Run Anywhere:

As discussed above java application generates a '.class' file which corresponds to our applications(program) but contains code in binary format. It provides ease t architecture-neutral ease as bytecode is not dependent on any machine architecture. It is the primary reason java is used in the enterprising IT industry globally worldwide.

Differences between JDK, JRE and JVM

1.JDK (Java Development Kit) is a Kit that provides the environment to **develop and execute(run)** the Java program. JDK is a kit(or package) that includes two things

Development Tools(to provide an environment to develop your java programs)
JRE (to execute your java program).



2. JRE (Java Runtime Environment) is an installation package that provides an environment to **only run(not develop)** the java program(or application)onto your machine. JRE is only used by those who only want to run Java programs that are end-users of your system.

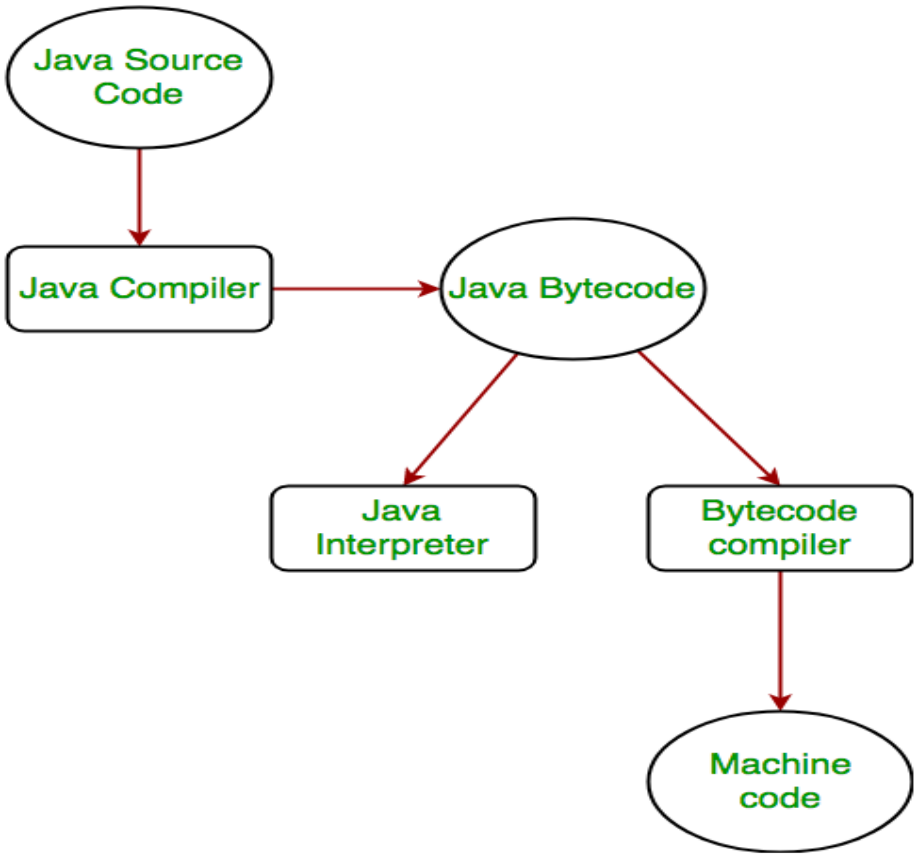
3. JVM (Java Virtual Machine) is a very important part of both JDK and JRE because it is contained or inbuilt in both. Whatever Java program you run using JRE or JDK goes into JVM and JVM is responsible for executing the java program line by line, hence it is also known as an

How is Java platform independent?

The meaning of platform-independent is that the java compiled code(byte code) can run on all operating systems.

Step by step Execution of Java Program:

- Whenever, a program is written in JAVA, the javac compiles it.
- The result of the JAVA compiler is the **.class file or the bytecode** and not the machine native code (unlike C compiler).
- The bytecode generated is a non-executable code and needs an interpreter to execute on a machine. This interpreter is the JVM and thus the Bytecode is executed by the JVM.
- And finally program runs to give the desired output.



Java Basic Syntax

Java program is a collection of objects, and these objects communicate through method calls to each other to work together. Here is a brief discussion on the Classes and Objects, Method, Instance variables, syntax, and semantics of Java.

Basic terminologies in Java

1.Class:

The class is a blueprint (plan) of the instance of a class (object). It can be defined as a template that describes the data and behavior associated with its instance.

- Example: Blueprint of the house is class.

2. Object:

The object is an instance of a class. It is an entity that has behavior and state.

- Example: A car is an object whose **states** are: brand, color, and number plate.

- **Behavior:** Running on the road.

3. Method: The behavior of an object is the method.

- **Example**: The fuel indicator indicates the amount of fuel left in the car.

4. Instance variables:

Every object has its own unique set of instance variables. The state of an object is generally created by the values that are assigned to these instance variables.

First Java Program

Let us look at a simple code that will print the words Hello World.

Example

```
public class MyFirstJavaProgram {

   /* This is my first java program.
    * This will print 'Hello World' as the output
    */

   public static void main(String []args) {

    // prints Hello World
      System.out.println("Hello World");
   }
}
```

Class definition

This line uses the keyword **class** to declare that a new class is being defined.

Example:

class HelloWorld {

//

//Statements

}

HelloWorld

It is an identifier that is the name of the class. The entire class definition, including all of its members, will be between the opening curly brace "**{**" and the closing curly brace "**}**".

main method:

In the Java programming language, every application must contain a main method. The main function(method) is the entry point of your Java application, and it's mandatory in a Java program. whose signature in Java is:

public static void main(String[] args)

public: So that JVM can execute the method from anywhere.

static: The main method is to be called without an object. The modifiers public and static can be written in either order.

void: The main method doesn't return anything.

main(): Name configured in the JVM. The main method must be inside the class definition. The compiler executes the codes starting always from the main function.

String[]: The main method accepts a single argument, i.e., an array of elements of type String.

Comments

They can either be multiline or single-line comments.

// This is a single line comment exampe

// Call this file "HelloWorld.java".

This is a single-line comment. This type of comment must begin with // as in C/C++. For multiline comments, they must begin from /* and end with */.

/ This is a example of*

of multiline comments/.*

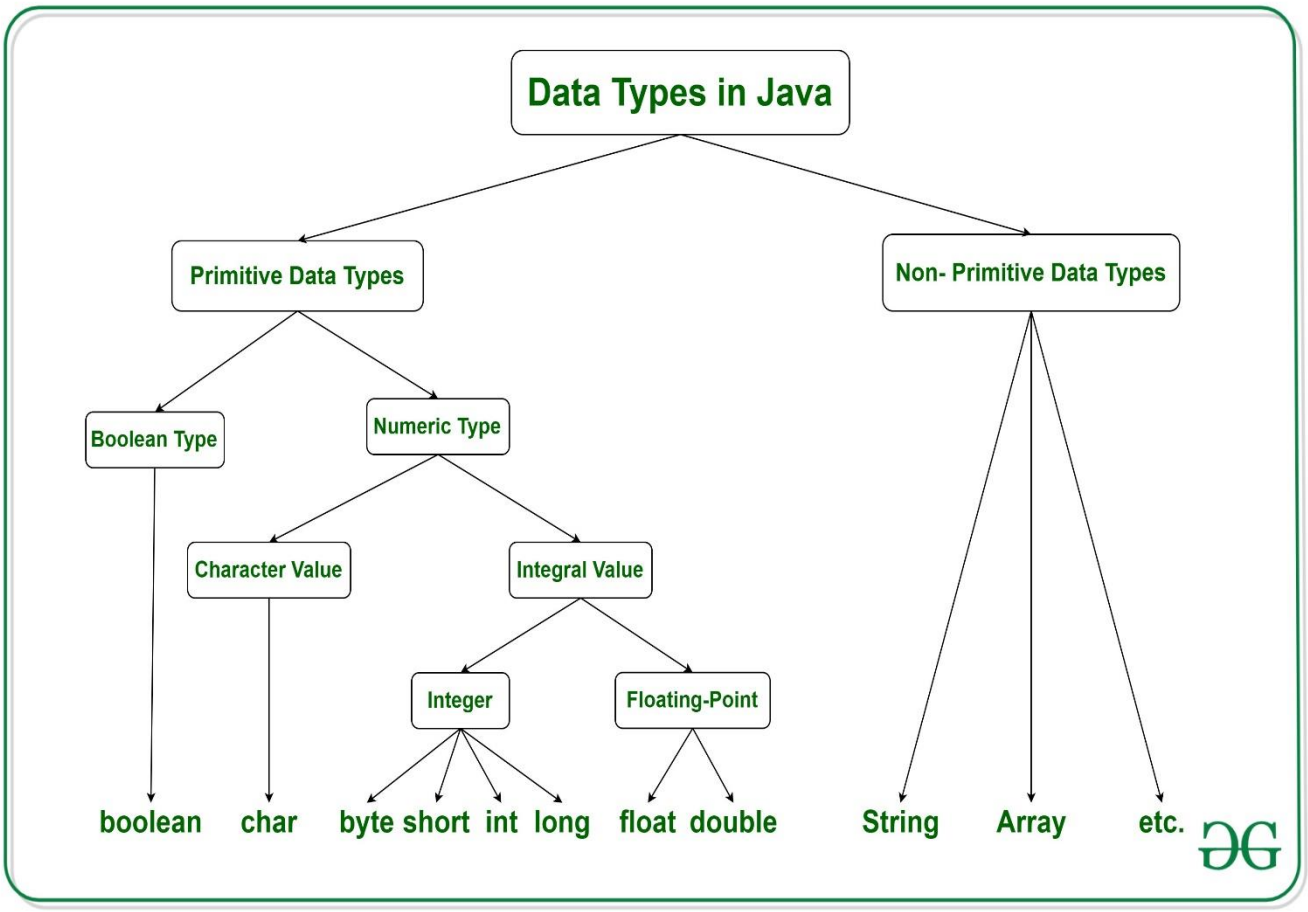
Data Types

Datatype is a spacial keyword used to allocate sufficient memory space for the data, in otherwords Data type is used for representing the data in main memory (RAM) of the computer.

Java has two categories in which data types are segregated

1. **Primitive Data Type:** such as boolean, char, int, short, byte, long, float, and double

2. **Non-Primitive Data Type or Object Data type:** such as String, Array, etc.



Data Type	Default Value	Default size
boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte
short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

Java Identifiers

In programming languages, identifiers are used for identification purposes. In Java, an identifier can be a class name, method name, variable name, or label. For example :

Variable

Variable is an identifier which holds data or another one variable is an identifier whose value can be changed at the execution time of program. Variable is an identifier which can be used to identifyinput data in a program.

Types of Variables in Java

Now let us discuss different types of variables which are listed as follows:

Local Variables

Instance Variables

Static Variables

Decision Making Statement

Decision making statement statements is also called selection statement. That is depending on thecondition block need to be executed or not which is decided by condition. If the condition is "true"statement block will be executed, if condition is "false" then statement block will not be executed. In java there are three types of decision making statement.

- if
- if-else
- switch

Looping Statement

Looping in programming languages is a feature which facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true.

- For loop
- While loop
- Do-while loop

Wrapper Classes

For each and every fundamental data type there exist a pre-defined class, Such predefined class is known as wrapper class. The purpose of wrapper class is to convert numeric string data into numerical or fundamental data.

Example:

Integer hundred = Integer.valueOf("100");

Boolean value = Boolean.valueOf("True");

Access modifiers

Access modifiers are those which are applied before data members or methods of a class. These areused to where to access and where not to access the data members or methods. In java programmingwe have four access modifiers they are

There are four types of access modifiers available in java:

- Default – No keyword required
- Private
- Protected
- Public

Access Modifier	within class	within package	outside package by subclass only	outside package
Private	Y	N	N	N
Default	Y	Y	N	N
Protected	Y	Y	Y	N
Public	Y	Y	Y	Y

Access Modifiers	Non-Access Modifiers
private default or No Modifier protected public	static final abstract synchronized transient volatile strictfp

JAVA USEFUL KEYWORDS

Final keyword

When a variable is declared with the ***final keyword***, its value can't be modified, essentially, a constant.

In java language final keyword can be used in following way.

- Final at variable level
- Final at method level
- Final at class level

Static keyword

The **static keyword** is used in java mainly for memory management. Static keyword are used withvariables, methods, blocks and nested class. Static is a keyword that are used for share the same variable or method of a given class. This is used for a constant variable or a method that is the same for every instance of a class. The main method of a class is generally labeled static.

In java language static keyword can be used for following

- variable (also known as class variable)
- method (also known as class method)
- block
- nested class

This keyword

this is a reference variable that refers to the current object. It is a keyword in java languagerepresents current class object

"**this**" keyword can be use in two ways.

- this . (this dot)
- this() (this off)

Super keyword

Super keyword in java is a reference variable that is used to refer parent class object. **Super** is an implicit keyword create by JVM and supply each and every java program for performing important role in three places.

- At variable level
- At method level
- At constructor level

Synchronized Keyword

Synchronized Keyword is used for when we want to allow only one thread at a time then use Synchronized modifier. If a method or block declared as a Synchronized then at a time only onethread is allowed to operate on the given object.

abstract

It is a non-access modifier applicable for classes and methods. It is used to achieve abstraction. For more, refer to abstract keyword in java

enum

It is used to define enum in Java

instanceof

It is used to know whether the object is an instance of the specified type (class or subclass or interface).

JAVA OBJECT ORIENTED

As the name suggests, Object-Oriented Programming or OOPs refers to languages that use objects in programming, they use objects as a primary source to implement what is to happen in the code. Objects are seen by the viewer or user, performing tasks assigned by you. Object-oriented programming aims to implement real-world entities like inheritance, hiding, polymorphism etc. in programming. The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function.

1. Class
2. Object
3. Method and method passing
4. **Pillars of OOPs**
    - Abstraction
    - Encapsulation
    - Inheritance
    - Polymorphism

1.Class

A class is a user-defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type. Using classes, you can create multiple objects with the same behavior instead of writing their code multiple times. This includes classes for objects occurring more than once in your code.

2.Object

An object is a basic unit of Object-Oriented Programming that represents real-life entities. A typical Java program creates many objects, which as you know, interact by invoking methods. The objects are what perform your code, they are the part of your code visible to the viewer/user.

3.Method

A method is a collection of statements that perform some specific task and return the result to the caller. A method can perform some specific task without returning anything. Methods allow us to **reuse** the code without retyping it, which is why they are considered **time savers**. In Java, every method must be part of some class, which is different from languages like C, C++, and Python

4. Pillars of OOPs

Abstraction

Data Abstraction is the property by virtue of which only the essential details are displayed to the user. The trivial or non-essential units are not displayed to the user. Ex: A car is viewed as a car rather than its individual components. Data Abstraction may also be defined as the process of identifying only the required characteristics of an object, ignoring the irrelevant details. The properties and behaviors of an object differentiate it from other objects of similar type and also help in classifying/grouping the object.

In Java, abstraction is achieved by interfaces and abstract classes. We can achieve 100% abstraction using interfaces.

Encapsulation

It is defined as the wrapping up of data under a single unit. It is the mechanism that binds together the code and the data it manipulates. Another way to think about encapsulation is that it is a protective shield that prevents the data from being accessed by the code outside this shield.

- Technically, in encapsulation, the variables or the data in a class is hidden from any other class and can be accessed only through any member function of the class in which they are declared.
- In encapsulation, the data in a class is hidden from other classes, which is similar to what **data-hiding** does. So, the terms “encapsulation” and “data-hiding” are used interchangeably.
- Encapsulation can be achieved by declaring all the variables in a class as private and writing public methods in the class to set and get the values of the variables.

Inheritance

Inheritance is an important pillar of OOP (Object Oriented Programming). It is the mechanism in Java by which one class is allowed to inherit the features (fields and methods) of another class.

- **Superclass:** The class whose features are inherited is known as superclass (also known as base or parent class).
- **Subclass:** The class that inherits the other class is known as subclass (also known as derived or extended or child class). The subclass can add its own fields and methods in addition to the superclass fields and methods.
- Inheritance supports the concept of "reusability", i.e. when we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class. By doing this, we are reusing the fields and methods of the existing class.

Types of Inheritance

- Single inheritance
- Multiple inheritance
- Hierarchical inheritance
- Multilevel inheritance
- Hybrid inheritance

Why use Inheritance ?

For Method Overriding (used for Runtime Polymorphism).

It's main uses are to enable polymorphism and to be able to reuse code for different classesby putting it in a common super class

For code Re-usability

Polymorphism

It refers to the ability of object-oriented programming languages to differentiate between entities with the same name efficiently. This is done by Java with the help of the signature and declaration of these entities

Constructor In Java

A **constructor** is a special member method which will be called implicitly (automatically) by the JVM whenever an object is created for placing user or programmer defined values in place of default values. In a single word constructor is a special member method which will be called automatically whenever object is created.

The purpose of constructor is to initialize an object called object initialization. Constructors are mainly create for initializing the object. Initialization is a process of assigning user defined values at the time of allocation of memory space.

Types of constructors

Based on creating objects in Java constructor are classified in two types. They are

- Default or no argument Constructor
- Parameterized constructor.

Interface

Interface is similar to class which is collection of public static final variables (constants) and abstract methods.The interface is a mechanism to achieve fully abstraction in java. There can be only abstract methods in the interface. It is used to achieve fully abstraction and multiple inheritance in Java.

Why we use Interface ?

It is used to achieve fully abstraction.

By using Interface, you can achieve multiple inheritance in java.

Method Overloading

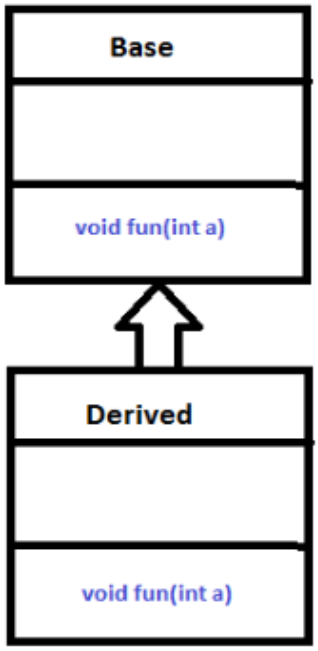
Whenever same method name is exiting multiple times in the same class with different number of parameter or different order of parameters or different types of parameters is known as **method overloading**.

Why method Overloading ?

Suppose we have to perform addition of given number but there can be any number of arguments, if we write method such as a(int, int)for two arguments, b(int, int, int) for three arguments then it is very difficult for you and other programmer to understand purpose or behaviors of method theycan not identify purpose of method. So we use method overloading to easily figure out the program. For example above two methods we can write sum(int, int) and sum(int, int, int) using method overloading concept.

Different ways to overload the method

- By changing number of arguments or parameters
- By changing the data type
- By changing the order of arguments.



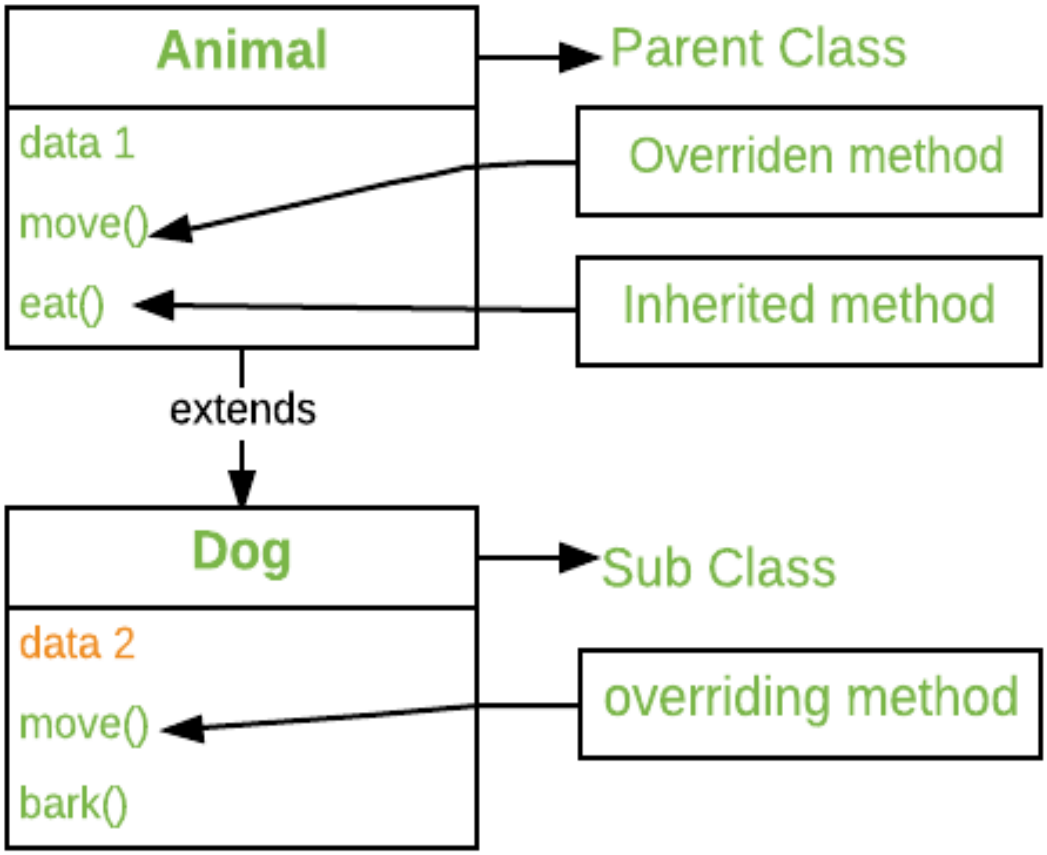
Overriding

Method Overriding

Whenever same method name is existing in both base class and derived class with same types ofparameters or same order of parameters is known as method Overriding.

Advantage of Java Method Overriding

- Method Overriding is used to provide specific implementation of a method that is alreadyprovided by its super class.
- Method Overriding is used for Runtime Polymorphism



Exception Handling

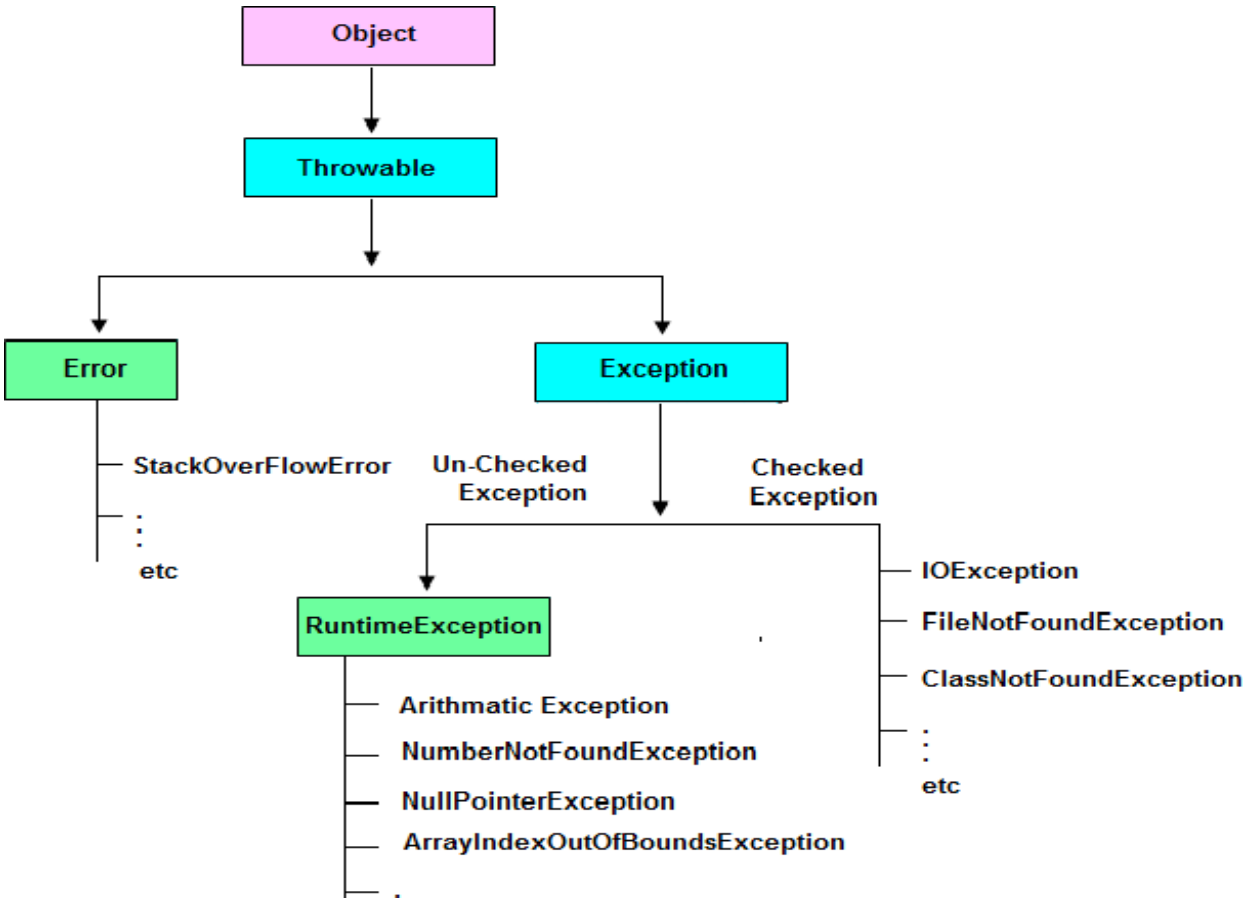
The process of converting system error messages into user friendly error message is known as **Exception handling**. This is one of the powerful feature of Java to handle run time error and maintain normal flow of java application.

An **Exception** is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's Instructions.

Type of Exception

- Checked Exception
- Un-Checked Exception

Hierarchy of Exception classes



References

- www.wikipedia.com
- www.geeksforgeeks.org
- docs.oracle.com/en/java/
- www.javatpoint.com/java-tutorial