# Object-Oriented Programming

Any programming language that supports Object-oriented principles like,

* Data Encapsulation
* Data Abstraction
* Inheritance
* polymorphism

Is called an Object-Oriented programming language.

**Object-Based Languages:** JavaScript, VB Script, Visual Basic, etc.

**Object-Oriented** = Object Based + inheritance + runtime polymorphism

**Object-Oriented Languages**: C++, JAVA, C#.NET etc.

**Class**: Class is a collection of data members and member functions.

**Object**: it is an instance of a class.

**Syntax to create an object**:

ClassName ObjectReference = new Constructor();

**Example**:

Emp e = new Emp();

**new**: new is a keyword and it is a Dynamic memory allocation operator. It allocates memory to instance variables at runtime.

**Constructor():** it constructs the object.

Diagram

Description automatically generated

Every object has an address, an address mapped into another code, and that code is implemented with hashing techniques, that code is called hash code.

Both object and object reference occupy memory. The object contains data whereas the object reference contains hash code.

**new Emp();**  => This is called Anonymous object (or) Unreferenced object.

**Emp e**; => Object Reference is created without Object; Here Object Reference will store null.

## Variables

**Variables** => A variable is a container that contains data.

**Types of Variables**:

* **Instance variables**:  A variable that is defined as a member of a class is known as an instance variable; memory is allocated to instance variables whenever the object is created. Instance variables are stored in the **heap area.**
* **Class variables**: A variable that is defined as a static member of a class is known as a class variable. Memory is allocated to class variables whenever a class is loaded. Class variables are stored in the **method area**.
* **Local variables**: A variable that is defined inside a method is known as local variable. Memory is allocated to local variables whenever a method is called. Local variables are stored in the **stack area**.

**Note1**: Local variables cannot be static in java.

**Note2**: No global variables in Java.

Text

Description automatically generated

## **Memory Allocations:**

There are 5 memory blocks in JVM.

* Heap Area
* Method Area
* Stack Area
* Native Stack Area
* PC(Program Counter) Registers
* Class can also be called a reference data type. An object reference can also be called a reference variable.
* Primitive type variables contain data whereas reference type variables contain hash code.
* Separate copy of instance variable exists forever object.
* Only one copy of the class variable exists for all objects.
* Graphical user interface, text, application

  Description automatically generated  
  Use instance variables if the values are different for objects.
* Use class variables if the value is the same for all objects.
* Use local variables to perform the task.

## **Execution Priority:**

* Class Variables (Static variables)
* Static Blocks
* Main() method

## Ways to Access Variables

### Instance Variables

**There are 2 ways to access an instance variable.**

* By using an object
* By using an object reference.

Application

Description automatically generated with low confidence

* Use an object to access an instance variable if it is required only one time.
* Use object reference to access instance variable if it is required more than one time.

### Class Variables

**There are 4 ways to access class variables.**

1. Directly: Access class variables directly if it is present in the same class.
2. By using class name: if the variable is present in a different class use class name.
3. By using an object: not recommended. Waste of memory.
4. By using object reference: not recommended, waste of memory

Graphical user interface, text, application

Description automatically generated

### Local Variable

**There is only one way to access local variable. That is Directly.**

Graphical user interface, application, Word

Description automatically generated

### Examples

Example1

Text, table

Description automatically generated  
**//OUTPUT 12 22 10 22**

Example2

Graphical user interface, text, application

Description automatically generated  
Example3

Graphical user interface, text, application, Word

Description automatically generated

## **Method Parameters and Block Variables:**

Graphical user interface, application, table, Word, Excel

Description automatically generated  
Method Parameters and Block Variables are also called local variables.

|  |  |
| --- | --- |
| **Variable** | **Life** |
| **Instance Variable** | **Object. As long as Object is allocated with memory.** |
| **Class Variable** | **Class. As long as Class is loaded.** |
| **local Variable** | **Method. As long as method is called.** |
| **Method variable** | **Method. As long as method is called.** |
| **Block Variable** | **Block. Block variable scope is limited to Block.** |

* Block variable scope is limited to block.
* Method parameters and local variables scope is limited to method only.
* Instance variables and class variables scope is based on access modifiers.

## **Enhanced for loop or for each loop:**

**It is used to iterate elements of an array.**

**Syntax:**

**For(Datatype variable : Arrayreferencevariable)**

**{**

**}**

Graphical user interface, text, application, Word

Description automatically generated

//output:

14

53

59

14

53

## **Methods**

A group of statements into a single logical unit is called a method. Methods are used to perform the task.

**In java, methods are divided into 4 categories.**

* Methods with arguments and with return values.
* Methods with arguments without return values.
* Methods without arguments with return values
* Methods without arguments and without return values.

**Advantages of Methods:**

* Modularity:
* Reusability:

Instance Method:

A method that is defined as a member of a class is called an instance method.

There are 2 ways to access an instance method.

* By using an object
* By using an object reference.

Use object to access an instance method if it is required only one time

Use object reference to access an instance method if it is required more than one time.

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

//OUTPUT

125

512

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, Word

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## **Method Overloading**

If 2 or more methods with the same name and with different parameters list then it is said to be method overloading.

Parameters can be different in the number of arguments, data types, or order of an argument.

In method overloading return can be same or different.

Graphical user interface, application

Description automatically generated

## **Method Overriding**

If 2 or more methods with the same name and with the same parameters list, then it is said to be method overriding.

Methods cannot be overridden in the same class because of ambiguity to call.

## **Blocks**

* A group of statements between { and } is called as a block.
* A block is also called an initialization block because a block also can be used to initialize instance variables.
* A class can have any number of Blocks, they will get executed in the order they are written.
* Blocks are executed even before constructors.

**public** **class** test

{

test()

{

System.***out***.println("Constructor");

}

{

System.***out***.println("Block 1");

}

**public** **static** **void** main(String[] args)

{

test t = **new** test();

System.***out***.println("Main");

}

{

System.***out***.println("Block 2");

}

}

//Output

Block 1

Block 2

Constructor

Main

## **Static Blocks**

* A Block with a static keyword is called a static Block.
* A Static Block is also called a Static initialization Block because a static Block can be used to initialize static variables.
* A class can have any number of static Blocks and all those static blocks are executed from top to bottom order whenever class is loaded.
* Static blocks are executed even before the main method.

**public** **class** test

{

test()

{

System.***out***.println("Constructor");

}

**static**

{

**int** a = 1;

System.***out***.println("Block 1");

}

**public** **static** **void** main(String[] args)

{

test t = **new** test();

System.***out***.println("Main");

}

**static**

{

System.***out***.println("Block 2");

}

}

//output

Block 1

Block 2

Constructor

Main

Static blocks are used to load libraries.