What is Class:

Class is an entity which binds data member and member methods into single unit.

Data member🡪data types

Member method🡪 method name

Class is a best example for encapsulation

In other words:

Class is nothing but blueprint or templet for creating an object which defines properties and behavior.

Properties🡪data member like integer, float, long

Behavior🡪methods which you are writing.

Syntax:

Access-modifies Class keyword className()

{

----------

----------

}

Public class calculation(){

**Public void calculateMarks()**

**{**

**----------**

**----------**

**}**

}

What is method:

Method is block of java statements which perform a particular task.

Methods will increase the reusability of your program.

And it will avoid the code duplication. B

Methods can be called using (.) operator.

Syntax:

Access-modifies static/non-static return type methodName()

{

----------

----------

}

Note: if we did not mention static means its default to non-static

If you do not have any return type, we need to mention void

Main purpose of this return type is let’s suppose you want to use this method and this method will calculate some data and you want to use that data, will use return type.

Example:

Public void calculateMarks()

{

}

Public static void calculateMarks()

{

}

Object:

An entity that has a state and behavior is known as object.

State->data type

behavior-you will create certain methods that will be the behavior

Objects are created to access the content of the class(variables and methods)

Syntax:

ClassName objectName=new constructor()

calculateMarks object=new calculateMatrix

constructor

A constructor is a special method of a class.

A constructor is an instance method that usually has the same name as the class,

Constructor method is called when object is created.

Facts:

Non- static methods can be called through object using (.) operator

Static methods can be called using class name directly using (.) operator

**package** SoftwareTestingMaterial;

**public** **class** Learn\_Class\_Method\_Object\_Variable {

//Class is an entity which binds data member and member methods into single unit

// int x and y are data members in class

**int** x=90;

**int** y=100;

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

//create an object of class

Learn\_Class\_Method\_Object\_Variable obj1=**new** Learn\_Class\_Method\_Object\_Variable();

System.***out***.println("program started");

obj1.sum();

System.***out***.println("x value is "+obj1.x);

System.***out***.println("y value is "+obj1.y);

System.***out***.println("end program");

}

**public** **void** sum() {

//this behavior

**int** a=10;

**int** b=20;

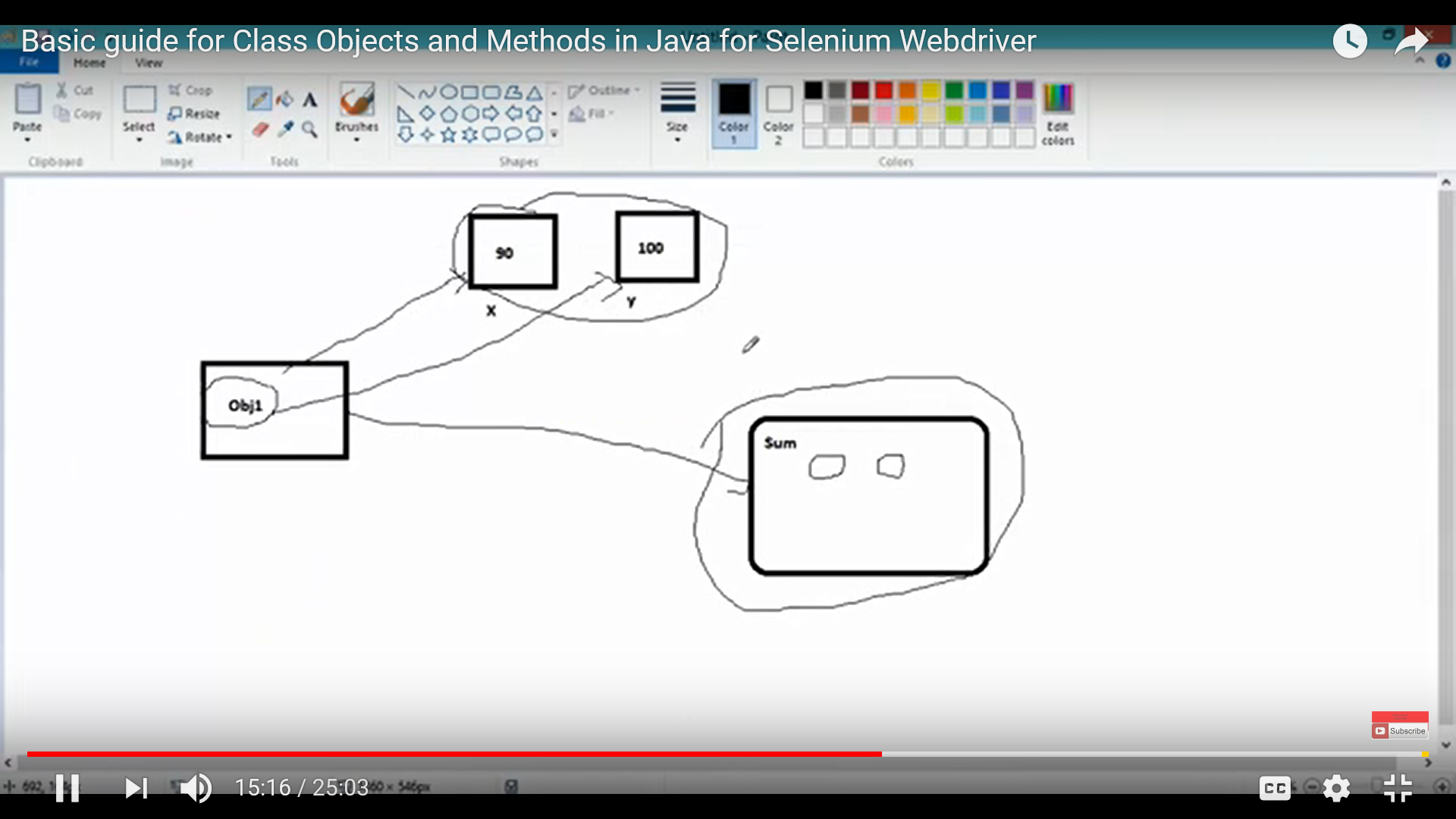
**int** c=a+b;

System.***out***.println("sum of a and b is "+c);

}

}

How obj works internally:



Multiple objects:

**package** SoftwareTestingMaterial;

**public** **class** Learn\_Class\_Method\_Object\_Variable {

//Class is an entity which binds data member and member methods into single unit

// int x and y are data members in class

**int** x=90;

**int** y=100;

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

//create an object of class

Learn\_Class\_Method\_Object\_Variable obj1=**new** Learn\_Class\_Method\_Object\_Variable();

System.***out***.println("program started");

obj1.sum();

System.***out***.println("x value is "+obj1.x);

System.***out***.println("y value is "+obj1.y);

System.***out***.println("end program");

Learn\_Class\_Method\_Object\_Variable obj2=**new** Learn\_Class\_Method\_Object\_Variable();

System.***out***.println("program started");

obj2.sum();

System.***out***.println("x value is "+obj2.x);

System.***out***.println("y value is "+obj2.y);

System.***out***.println("end program");

}

**public** **void** sum() {

//this behavior

**int** a=10;

**int** b=20;

**int** c=a+b;

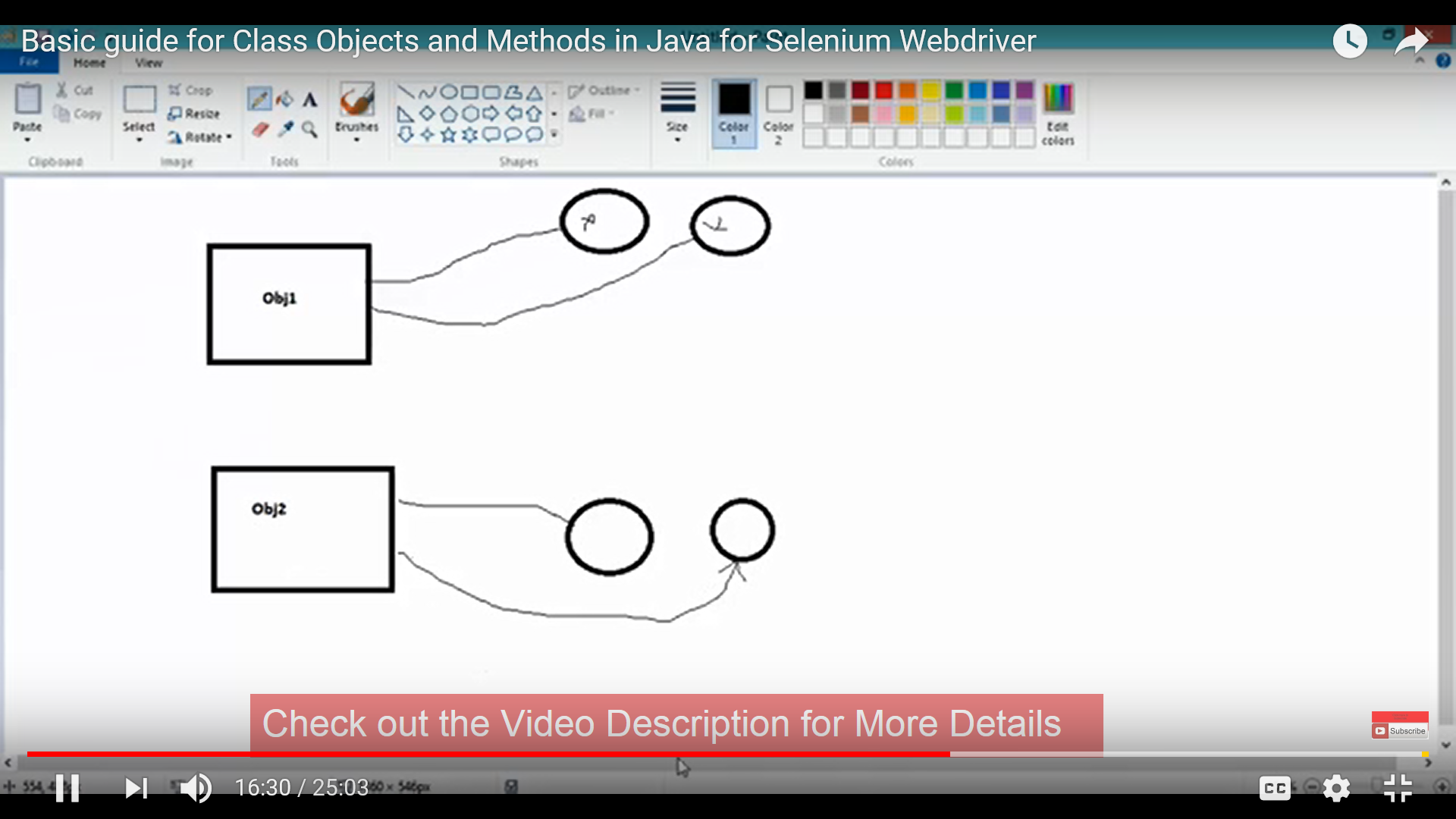
System.***out***.println("sum of a and b is "+c);

}

}

How object memory

If you have two objects each time it will create a separate memory for no static if static it will vary



Note:

We can create methods and variables in one class with out main method and we can call those methods and variables in second class using objects with main method.

We can create methods and variables in one class with main method and we can call those methods and variables in same class using objects under main method.

Static methods can be called using class name directly using (.) operator

**package** SoftwareTestingMaterial;

**public** **class** Static\_methods {

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

//Static methods can be called using class name directly using (.) operator

Static\_methods.*sum*();

Static\_methods.*mul*();

}

**public** **static** **void** sum() {

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

}

**public** **static** **void** mul() {

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

}

}

Variables:

Variables are devices that are used to store data, such as a number, or a **string** of **character** data.

Access Modifiers:

Access modifiers controls the access of the class or member variable or member method.

There are 4 types of access modifiers

1. Public
2. Default--blank
3. Private
4. Protected

Class Access Levels –public / nomodifier

•If a class is ‘public’, then it CAN be accessed fromANYWHERE.

•If a class has ‘no modifier’, then it CAN ONLY be accessed from ‘samepackage’.

•Member Access Levels –public / protected /no modifier /private

•public and no modifier –the same way as used in classlevel.

•private –members CAN ONLYaccess.

•protected –CAN be accessed from ‘same package’ and a subclass existing in any package canaccess.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Modifer | Same Class | Same Package | Sub Class | Other Package |
| Public | Y | Y | Y | Y |
| default | Y | Y | N | N |
| protected | Y | Y | Y | N |
| private | Y | N | N | N |

Main Method:

Main method is the starting point of Execution.

Methods are used in 3 concepts

1. Calling a method
2. Definition of the method
3. Signature of the method

Static variable means once you create a variable as static then that variable is common to all instance or object of class.

Static method :

If you declare method as static means we can call that method directly in main method without any object or instance.

Static block is used to initializing the static variables.

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

1. Why JVM is going to search first for main method only?

JVM is a software program.

Instructions return inside JVM is nothing but whenever is JVM is going to execute any program internally which method it must execute is nothing, but **Public static void main(String[] args)** is configured inside JVM.

1. Is it possible to change the name of the main?

Yes, we can update JVM configuration inside. We can replace main with some other thing what ever we want, but customization od JVM needs to be done once updated with new one.

1. Why main method is Public?

Who is going to call main method means JVM is going to call main method.

JVM may installed in inside C or any drive in out laptop.

So to call by JVM from anywhere main method should be public.

1. **Why main method is static?**

JVM will call main method first.

First calling main method means, there is no hope to find object in program.

Without existing any object also JVM has to call this main method and main method no way related to any particular object. That’s why main method should be static.

1. **Why main method is void?**

JVM is going to call main method.

If I am calling main method then I can expect some return type and will that return type value in rest of the program.

If I will return something to the main method what JVM is going to do with that return value.

Nothing will do JVM.

That’s why main method will not return anything to the JVM. That’s why return type for main method as void.

1. **what (String[] args)?**

(String[] args) are command line arguments

**Note:** If we will change anything in syntax of main method, will not get any compile error, but will get runtime error.

Error: NoSuchMethodError:main.

Below error getting in eclipse oxygen.

Error: Main method not found in class week1.day2.ErrorForMainMethod, please define the main method as:

public static void main(String[] args)

or a JavaFX application class must extend javafx.application.Application

Loops:

|  |
| --- |
| [while loop](https://www.tutorialspoint.com/java/java_while_loop.htm)  Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body. |
| 2 | [for loop](https://www.tutorialspoint.com/java/java_for_loop.htm)  Execute a sequence of statements multiple times and abbreviates the code that manages the loop variable. |
| Sr.No. | Control Statement & Description |
| 1 | [break statement](https://www.tutorialspoint.com/java/java_break_statement.htm)  Terminates the loop or switch statement and transfers execution to the statement immediately following the loop or switch. |
| 2 | [continue statement](https://www.tutorialspoint.com/java/java_continue_statement.htm)  Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating. |

**Java for Loop**

**Number of ieration fixed go for for loop**

The syntax of for Loop in Java is:

for (initialization; testExpression; update)

{

// codes inside for loop's body

}

If else

An **if** statement can be followed by an optional **else** statement, which executes when the Boolean expression is false.

## Syntax

Following is the syntax of an if...else statement −

if(Boolean\_expression) {

// Executes when the Boolean expression is true

}else {

// Executes when the Boolean expression is false

}

# Java While Loop

The Java while loop is used to iterate a part of the program several times. If the number of iteration is not fixed, it is recommended to use while loop.

**Syntax:**

1. while(condition){
2. //code to be executed
3. }

switch

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

## Syntax

The syntax of enhanced for loop is −

switch(expression) {

case value :

// Statements

break; // optional

case value :

// Statements

break; // optional

// You can have any number of case statements.

default : // Optional

// Statements

}

## Example

[Live Demo](http://tpcg.io/uqGH4y)

public class Test {

public static void main(String args[]) {

// char grade = args[0].charAt(0);

char grade = 'C';

switch(grade) {

case 'A' :

System.out.println("Excellent!");

break;

case 'B' :

case 'C' :

System.out.println("Well done");

break;

case 'D' :

System.out.println("You passed");

case 'F' :

System.out.println("Better try again");

break;

default :

System.out.println("Invalid grade");

}

System.out.println("Your grade is " + grade);

}

}

## Java Operator Precedence

|  |  |  |
| --- | --- | --- |
| **Operator Type** | **Category** | **Precedence** |
| Unary | postfix | expr++ expr-- |
| prefix | ++expr --expr +expr -expr ~ ! |
| Arithmetic | multiplicative | \* / % |
| additive | + - |
| Shift | shift | << >> >>> |
| Relational | comparison | < > <= >= instanceof |
| equality | == != |
| Bitwise | bitwise AND | & |
| bitwise exclusive OR | ^ |
| bitwise inclusive OR | | |
| Logical | logical AND | && |
| logical OR | || |
| Ternary | ternary | ? : |
| Assignment | assignment | = += -= \*= /= %= &= ^= |= <<= >>= >>>= |

### J

### ava Ternary Operator

Java Ternary operator is used as one liner replacement for if-then-else statement and used a lot in java programming. it is the only conditional operator which takes three operands.

### Java Ternary Operator Example

1. class OperatorExample{
2. public static void main(String args[]){
3. int a=2;
4. int b=5;
5. int min=(a<b)?a:b;
6. System.out.println(min);
7. }}

**package** week1.day2;

**public** **class** TernaryOperator {

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

/\*ternary operator syntax is

\*variable=(condition?true:false;\*/

**int** a=10;

**int** b=20;

**boolean** z=(a>b)?**true**:**false**;

System.***out***.println(z);

}

}