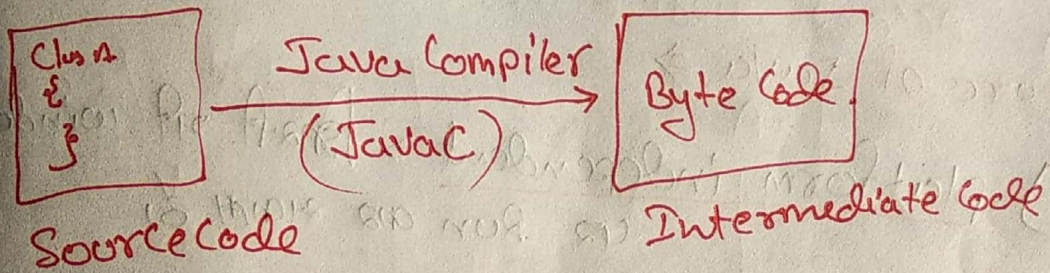
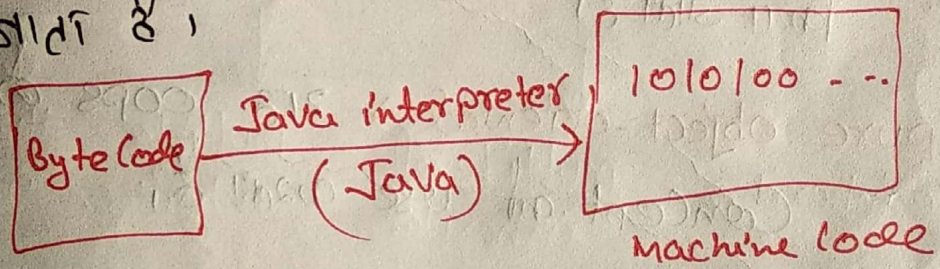
Java

JVM (Java Virtual machine) ⇒ यह एक technique है जिसकी सहायता से Source Code को Byte Code में तथा Byte Code को machine Code में Convert किया जाता है।

- इसमें Java Compiler की सहायता से पहले source Code को Byte Code (intermediate Code) में Convert किया जाता है।



- इस intermediate Code को Java interpreter की सहायता से machine Code में Convert किया जाता है।

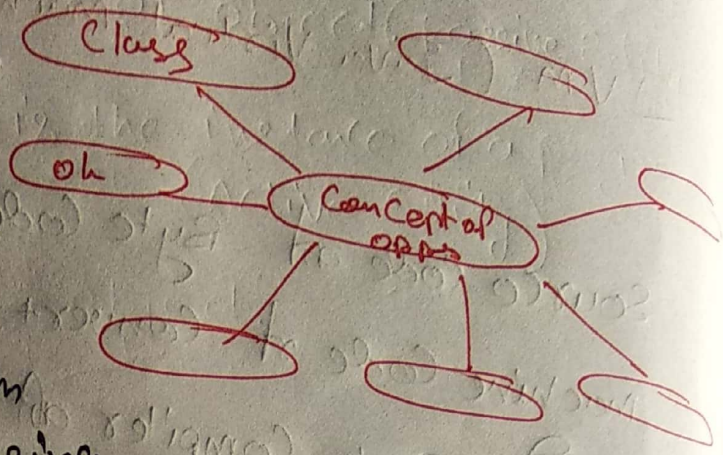


इस पूरे process को JVM कहते हैं।

JIT (Just in time Compiler) ⇒ जब source Code Byte Code में Convert होता है, Compiler की help से तो उसे JIT.

- Concept of oops ⇒

1) Class
2) object
3) Inheritance
4) polymorphism
5) abstraction
6) Encapsulation
7) message passing
8) Dynamic Binding



- feature of Java ⇒

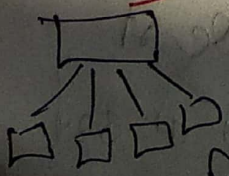
1) platform independent ⇒ किसी भी window पर Run कर सकता है,

2) portable ⇒ इसके program को एक System से दूसरे S/m पर Easily move किया जाता है,

3) pure object-oriented ⇒ यह oops के सभी Concept को use करता है,

4) Robust & Secure ⇒ यह more Secure language है,

5) multithreaded ⇒ इसमें एक main program को बहुत सारे छोटे-छोटे program में



divide किया जाता है, फिर उन सबको Execute किया जाता है ताकि Error Easily remove हो सके,

6) Compler & Interpreter ⇒ यह Compiler तथा Interpreter दोनो यह work perform करती है।

7) Simple & Small ⇒ Java Simple & Small Language है।

difference b/w java & C++.

Java	C++	C
1) यह compiler तथा interpreter base Language	2) Compiler base.	Compiler base
2) portable Language	3) Not portable	Not
3) pure object-oriented	3) Less purce	Not
4) It Not Support Global variable	Support	Support
5) platform independent	dependent	dependent

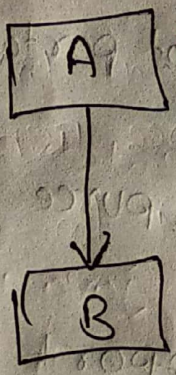
Class & object.

Class ⇒ Collection of object (passive Entity)

object ⇒ object is the instance of a program. (Active Entity)

WAP for Class & object.

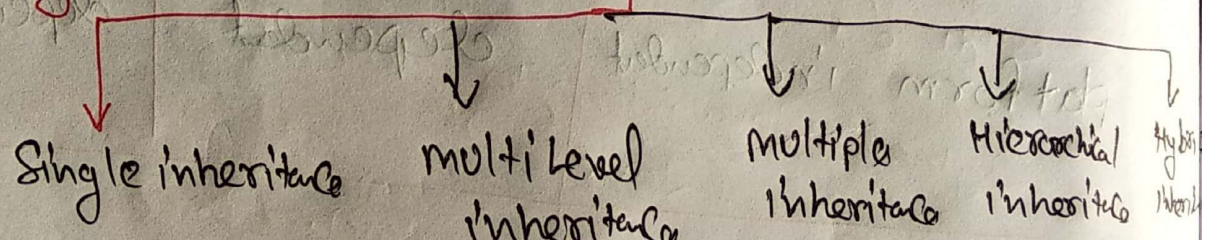
Inheritance ⇒ जब एक Class दूसरी Class की property (data member, memberfunction) और Access करने के लिए Inheritance करते हैं।



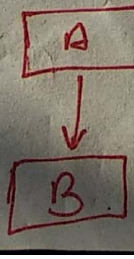
Superclass / parentclass / Baseclass

Subclass / Childclass / derivedclass

type of Inheritance



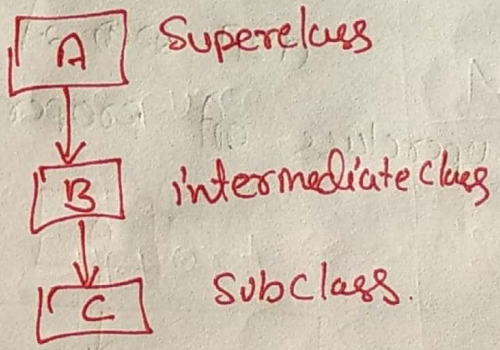
1) Single inheritance ⇒ इसमें एक SuperClass और एक SubClass होती है। SubClass, SuperClass की property और Access करती है।



Syntax ⇒

```
Class A
{
  data member;
  member function;
}
Class B Extends A
{
  data member;
  member function;
}
```

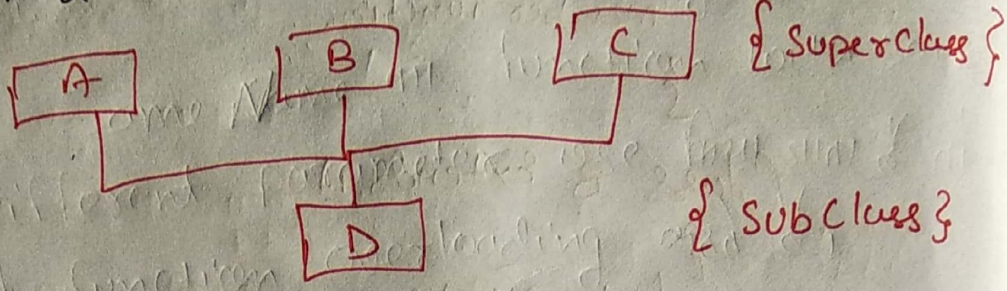
2) multiLevel inheritance ⇒ इसमें subclass का parent ~~का~~ ~~का~~ subclass create की जाती है;



Syntax ⇒

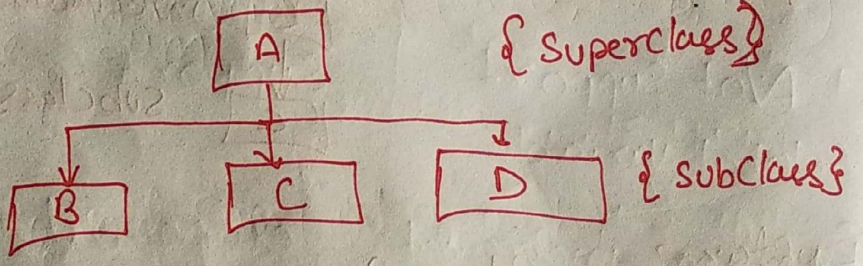
```
Class A
{ ---
  ---
}
Class B Extens A
{
  ---
  ---
}
Class C Extends B
{
```

3) Multiple inheritance ⇒ इसमें बहुत सारी Superclass होती हैं इन सभी की property को एक subclass Access करती है,



NOTE: → Java does not support multiple inheritance.

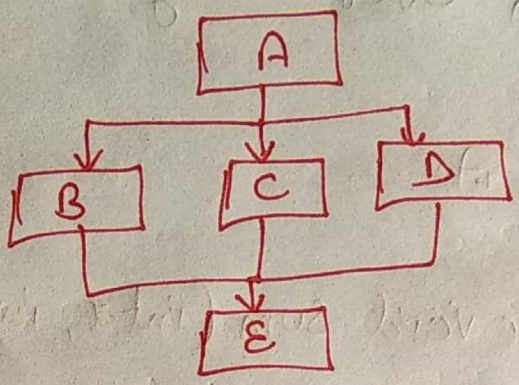
4) Hierarchical inheritance ⇒ इसमें एक Superclass तथा एक से ज्यादा Subclass होती हैं, Subclass Superclass की property को Access करती हैं,




```
Class A
{ ---
}
Class B Extends A
{ ---
}
Class C Extends B
{ ---
}
```

```
Class D Extens A
{
}
```

5) Hybrid inheritance ⇒ यह multiple तथा Hierachical का Combination होता है।



} Hierarchical inheritance

} multiple inheritance.

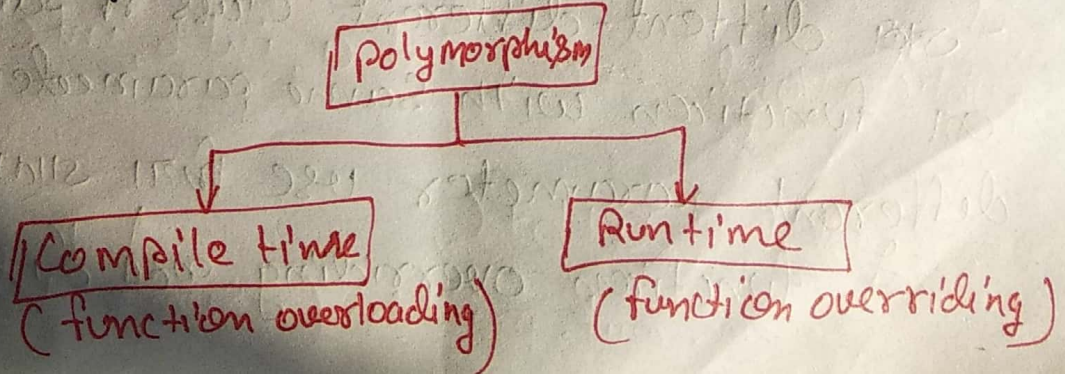
NOTE: → Java doesn't support Hybrid inheritance.

polymorphism ⇒ One name many forms is called polymorphism.

यह दो word से मिलकर बना है poly + morphism.

poly means बहुत सारे

morphism means forms.



Compile time ⇒ इसमें function compile time पर overload होता है,

function overloading ⇒ जब single class में Same name का function with different parameter use किया जाता है तो function overloading कहते है,

Ex ⇒

```
Class A
{
  public void sum (int x, int y)
  {
    ---
  }
  public void sum (int a, int b, int c)
  {
    ---
  }
}
```

Run time polymorphism ⇒ (function overriding) ⇒ जब diffent-different class में same name का function with same parameter or different parameter use किया जाता है तो function overriding कहते है,

Ex⇒
```
Class A
{
public void sum (int x, int y)
{
- -
- -
}
}
Class B Extends A
{
public void sum (int a, int b, int c)
{
- -
- -
}
}
```

Constructor ⇒ • यह एक special member function होता है, जिसका same name Class name के समान होता है।

• Constructor automatic call होता है, यह object create करते समय ही call हो जाता है।

Syntax⇒
```
Class Sum
{
public void sum ( )
{ - -
- -
}
}
```

1) WAP for Class & object
2) WAP for Constructor / parameterize Constructor
or
3) WAP for function overloading.
or
4) WAP for object & function Creation.
or
5) WAP for more then one Class Create.
6) WAP to add two & three no. using function.

```
Class Sum
{
public void Sum(int a, int b)
{
  int c = a+b;
System.out.println("sum="+c);
}
public void Sum(int a, int b, int c)
  {
    int d = a+b+c;
    System.out.println("sum="+d);
```

```
}
}
Class main
  {
  public static void main (String args[])
  {
Sum S = new Sum (2, 3, 4);

}
}
```

Error ⇒ mistakes in a program.

1) Compile time Error
2) Runtime Error

1) Compile time Error ⇒ All the Error that Can form to Compile the program, it's called Compile time Error.

- → missing semicolons.
- → missing brackets in classes and methods.
- ⇒ missing double quotes in strings.
- → Use of undeclared variables.

```
Class A
{p.s.v.m (String args[])
{
S.o.println ("Hello")
```

Run time Error ⇒ program Compile होने के बाद Run time पर जो error form होते है उन्हे Run time Error कहते है।

Ex ⇒ * divided by zero.
* Array index out of bound.
* Converting invalid string to ano.

Ex ⇒
```
Class A
{
public static void main(String args[])
{
  int a=10, b=5, c=5, d;
  d = a/(b-c);  = 10/(5-5) = 10/0 = ∞
  System.out.println("division = " + d);
}}
```

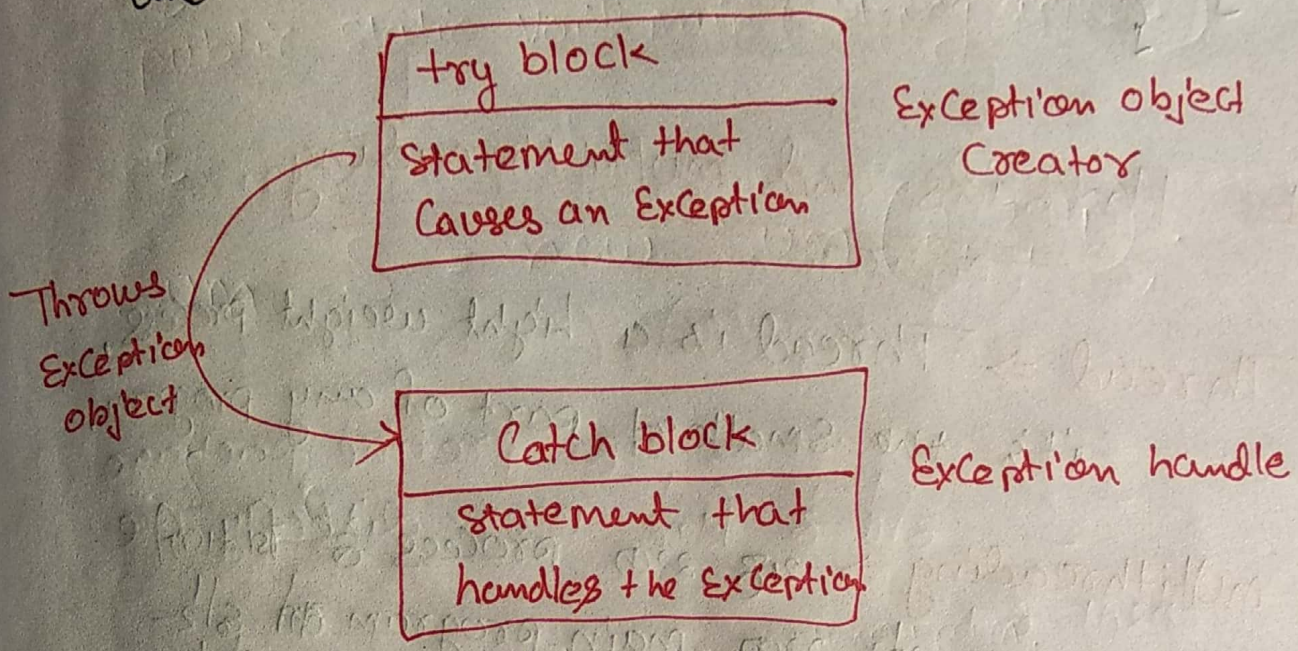
Exception ⇒ Run time पर जो Error generate होते है, उन्हे Exception कहते है।

Exception handling ⇒ Runtime पर जो Error generate होते है उन्हे find करना पर remove करना Exception handling कहलाता है, इसे perform करने के लिए हमे task perform करने पड़ते है, जैसे.

1) find the problem
2) Inform Error has occurred
3) Receive the Error information

4) take Corrective Action.

Exception handling Code ⇒ सारा try, Catch, finally statement की help से error को find and resolve किया जाता है।



Syntax ⇒

```
try
{
 statement;
}
Catch (Exception type e)
 {
  statement;
  }
```

Ex ⇒

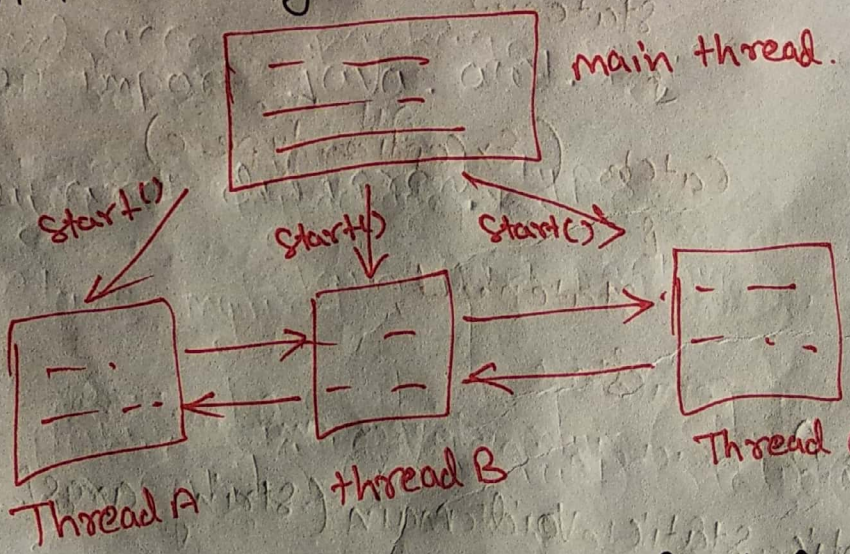
```
Class A
{ public static void main (String args[])
{ int a=10; b=5, c=5, d;
try {
      d = a/b-c;
    }
```

```
Catch (ArithmeticException e)
{
    System.out.println("Divided by zero");
}
}
```

Threads and multithreading

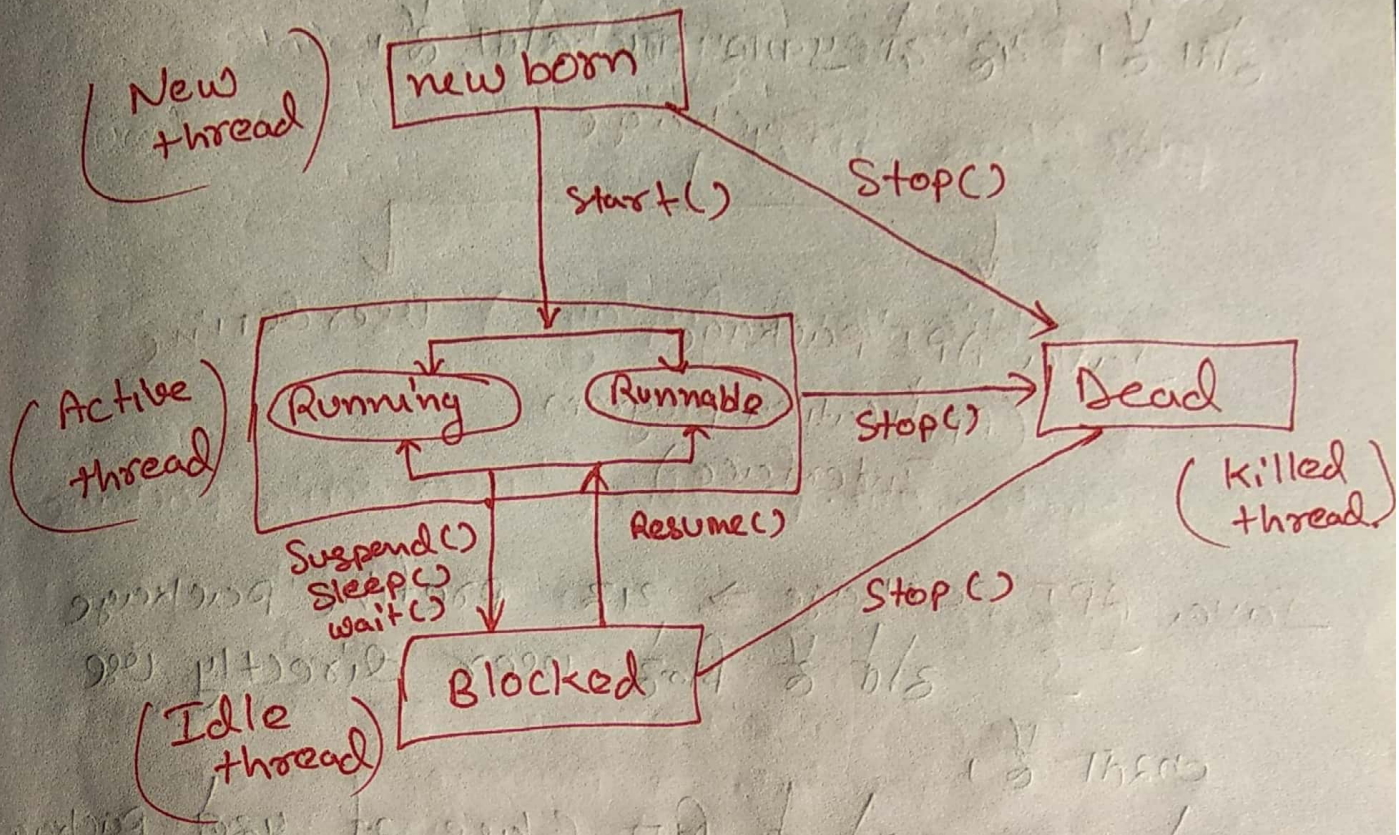
Thread ⇒ Thread is a light weight process, it's the smallest part of any program.

multithreading ⇒ यह 2 या process से ज्यादा 2 या main program को एक-एक program में divide करता है उसे multithreading कहते हैं।



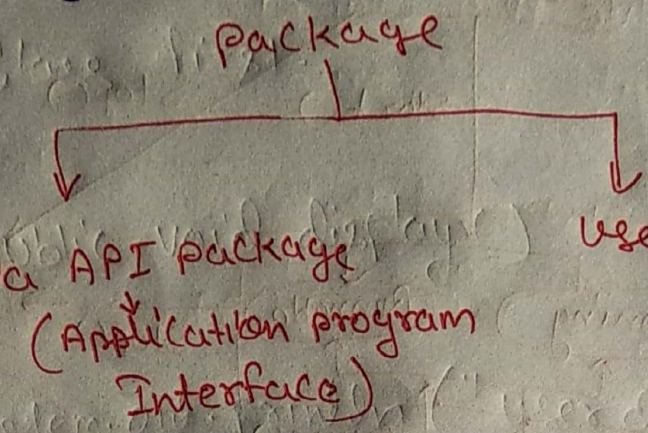
thread में run() method की help से program को execute कराया जाता है।
run() method है heart and soule of thread.

Life cycle of thread.



Java package ⇒ Bunch of Classes.

Java package बहुत सारी classes का collection होता है। यह दो प्रकार का होता है।

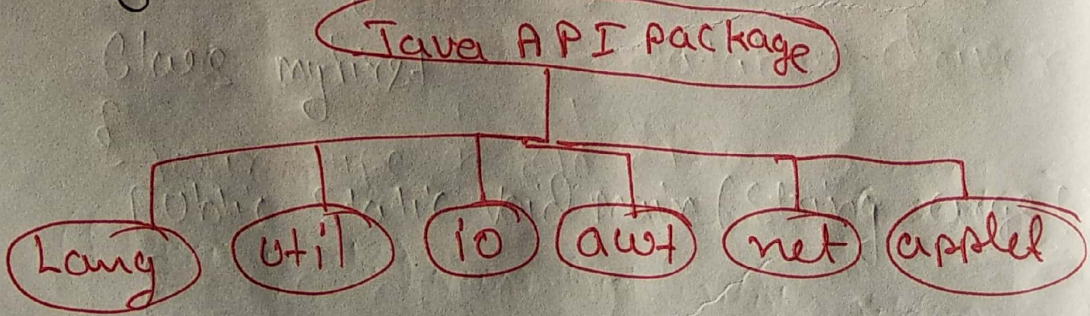


Java API package ⇒ यह predine package होते हैं जिन्हे user directly use करता है।

इन्हे use करने के लिए Class से पहले package को define किया जाता है।

Ex⇒ 'import java.awt.*;'

they are different types.



1) Lang ⇒ import java.lang.*; ⇒ यह language Support classes होती है, जैसे math, String function etc.

2) util ⇒ utility classes, hash table, vectors, date etc.

Ex ⇒ date d = new date();

import java.util.*;

3) io ⇒ Input/output Classes. यह input & output को define करता है।

4) awt ⇒ abstract window toolkit.
Label, textBox, CheckBox, Button.

import java.awt.*;
import java.awt.Button;

5) Net ⇒ Classes of networking.

6) Applet ⇒ Clases for Creating and implementing Applets.

User define package ⇒ इसे user अपने use के लिए Create करता है, first package keyword से define करता है।

1) पहले एक folder package name का Create करते है।

2) 32+ folder ⇒ package, Class ⇒ save

Ex ⇒ package mypackage;

public Class first

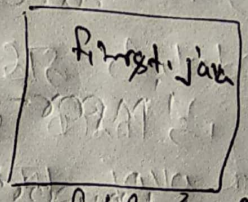
{

public void display()

{

System.out.println("user define package");

}}



3) 2nd new program outside these floder Create and access package mypackage.

import mypackage.*; (package user define)

Class myfirst

{

public static void main(String args[])

{

first ob = new first();

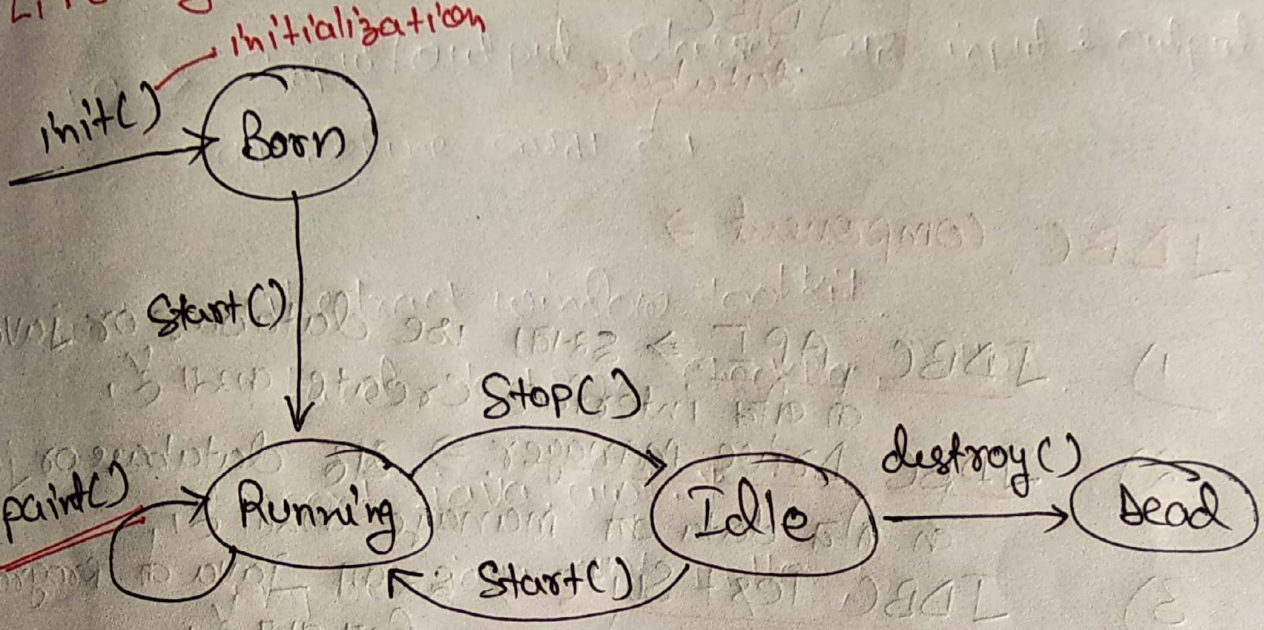
ob.display();

}}

Applet ⇒ इसकी सहायता से java पर web page Access किये जाते है।

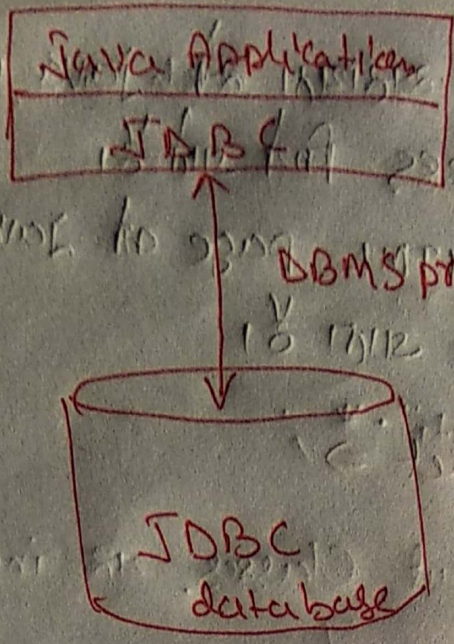
Applet का use HTML page को Java पर Access करने के लिए किया जाता है।

Life cycle of Applet ⇒



JDBC ⇒ Java database connectivity

- इसका use java program को database से Connect करने के लिए किया जाता है।
- इसका सहायता से data को database में store तथा data को database से Access किया जाता है।
- इसमें SQL की query के through database Access or store किया जाता है।



JDBC component ⇒

1) JDBC API ⇒ इसका use database or Java के बीच interface create करता है,
2) JDBC Drive manager ⇒ यह database or Java के बीच data को manage करता है,
3) JDBC Test Suite ⇒ इसका Java के program द्वारा database को test किया जाता है,
4) JDBC-ODBC Bridge ⇒ यह JDBC Command को ODBC (Open database connectivity) में Convert करता है,

JDBC Drivers ⇒ database driver four type के होते है,

1) JDBC-ODBC Bridge driver ⇒ यह JDBC function को ODBC function में convert करता है, जिसे हमें easily use कर सके,

2) Native API Driver ⇒ यह Client side Library को manage करता है।

3) Network Protocol Driver ⇒ इसकी सहायता से database or java के बीच Network Create किया जाता है।

4) Native Protocol Driver ⇒ इसकी सहायता से JDBC Call को direct access किया जाता है।

Steps for JDBC ⇒

1) Loading driver
2) Establishing Connection
3) Creat a statement Object
4) Executing statements
5) Getting result
6) closing database connection