
G
O
D
O
O
O
O
D

OOPs in Java

Object oriented Programming →
Classes + Object ⟹ Design Program

* Class →
  - user defined datatype
  - Define its properties & functions

* Action ⟶ Functions

* class does not occupy any memory space till the time an object is instantiated.

* Object ⟶
  - Run time entity
  - Instance of the class
  - Represent any Person, Place, Item.

* Objects → data members + member functions.

Explain example →

→ Object creation - new

→ 'this' → Current instance of the class

* Constructor →

- invoked automatically at the time of object creation
- Initilize the data members of new objects generally

- Same name as class
- don't have return type
- Only called once, at object creation

→ Default

→ Parameterized

→ Copy

OOPs concept →

- Polymorphism →
-
-
-

* Polymorphism →

→ Allow object of different classes to be treated as same class

→ More Flexible & extensible code

→ Achieved through method Overriding.

- 2 types
  - Compile time polymorphism - (static)
  - Runtime Polymorphism - (Dynamic)

Compile time → Implemented

eg. method overloading.

Method overloading →

→ Allow you to have more than one function.
  └→ same function name but with different functionality

* Runtime Polymorphism →
  * dynamic Polymorphism.
→ Function Overriding

→ What is function overriding?
  → When child class contains a method which is already present in parent class
  └→ child class overrides the method of the parent class.

→ Parent & child contain same function with different definition
→ It is determine at runtime

Inheritance →

- Object acquires Properties & behaviours of Parent Object automatically

→ you can reuse, extend or modify the attribute & behaviour defined in Other classes

→ Derived Class - Inherite member of another class

→ Base class → Whose members are Inherited

Types of Inheritance →

1) Single ⇒ One class Inherit another
2) Hierarchical ⇒ Derive more than one Class from base class

3) Multilerel → Deriving class from another derived class
4) Hybrid → Combination Single, mul, Hi

Encapsulation →

- Process of Combining data & function into Single unit

- Data is not accessed directly →
  access through function present in Class
  * Model class - getters & setters.
  → attributes are kept private

- Encapsulation make concept of "Data hiding" possible

Abstraction →

- It is hiding un-necessory details
- Show only essential functionalities to user.

Data binding :-

process of binding Application UI + Business Logic

- Any change in BL → Appl^n UI

Achieved in 2 ways →

1) Abstract class
2) Interface * Pure abstraction

Abstract class :-

→ Abstract Keyword

→ Can have abstract & non-abstract method

→ Cannot be instantiated

→ Can have Constructors & static methods

→ Can have final methods

Force subclass not change body

Interfaces Example :-

→ All fields Public, Static & final by default

→ methods are public & abstract by :-

→ Class must implement all methods declared in interfaces

→ ex →