Backend and Database Development

13 classes

Day 2 : 23 Apr 2024

Java Technologies

OOPs concept

Object : object is any real world entity.

Class : blue print of object or template of object

Wheel, colour, price etc

Car object

Start() appliedGear() moving() stop()

Syntax to create the object

className objectRefName = new ClassName();

objectRefName.methodName();

types of variable or fields.

In java variable are divided into 3 types.

1. Instance variable
   1. The variable which declared inside a class but outside a method is known as instance variable.
   2. Instance variable hold default value respective their data types like int family 0 float family 0.0, char space, Boolean false, String null
   3. Instance variable we can access within a same class directly inside non static method.
2. Local variable
   1. The variable which declared inside a method is known as local variable.
   2. Local variable doesn’t hold default value.
   3. The scope of the variable within that method where it declared.
3. Static variable

This keyword : when instance variable and local variable have same name then local variable hide the visibility of instance variable.

this.variable name consider as instance and variable name local.

Constructor : constructor is a type of special method which help to create memory or object.

Pts

1. Constructor have same name as class itself.
2. Constructor no need to call it will call automatically when we create the object.
3. Constructor doesn’t contain any return type not even void also.

In the life of the object if we want to perform any task only one time that type of task we need to do inside constructor.

In the life of the object if we want to perform any task more than one time that type task we need to inside a methods.

Encapsulation : binding or wrapping data(variables) and code(methods) in a single unit is known as Encapsulation.

Example : class. by default in all oops language class internally follow Encapsulation.

Inheritance : Inheritance is use to inherits the properties and behaviour of old class to new class.

class OldClass { super class or base class or parent class.

property

behaviour

}

class NewClass extends OldClass{ sub class or child class or derived class.

property

behaviour

}

With help of sub class object we can access its own property/behaviour as well as super class property/behaviour. But with help of super class we can access only its own property/behaviour