Backend and Database Development

13 classes

Day 3 : 24 Apr 2024

Java Technologies

Run time polymorphism :

Method Overriding : the method have same name and same method signature(number of parameter list, type of parameter list and return type must be same). To achieve method overriding we need inheritance concept.

annotation : annotation is known as meta-data. Meta -data means data about data. Java provided lot of pre defined annotation as well as we can create custom annotation.

Some annotation we can use on class level, method level or property level etc.

All annotation start with pre fix @ followed by name of annotation.

@Override : this annotation we can use on method level of sub class. if that method override super class method. we doesn’t get any error else we get the error.

abstract keyword

1. abstract keyword we can use with method and class but not with variable.
2. Abstract method means incomplete method or method without body or without curly braces.

abstract returntype methodName(parameterList);

abstract void speed();

1. If class contains one or more abstract method we need to declare the class as abstract class.

abstract class className {}

1. Whichever class extends abstract class that class must be provide the body for all abstract method mandatory.
2. abstract class can contains normal as well as abstract methods. Means it can contains zero or 1 or many abstract methods. Class is abstract but not mandatory all methods or any method must be abstract.
3. We can’t create the object of abstract class.

final keyword

1. Final keyword we can use with class, method and variable.
2. Final variable : to declare a constant value in java we use final keyword with variable.

final int A=10;

1. Final method : if method is final we can’t override that method.
2. Final class : we can’t inherits or extends the final class.

static keyword :

1. Static keyword we can use with variable and method but not with class.
2. Static variable : if variable is static we can access or assign the value for that variable using class name.
3. Static method : if method is static we can that method with help of class name object not required.
4. Static variable as well as static method we can access or call using object also.
5. Inside a static method we can access only static method/variable directly.
6. Inside non static method we can access both type of variable ie static as well as non static.

Static memory vs instance memory

Whenever class get loaded for each class one static memory present.

If that class contains one or more than one variable those variable is part of that static memory ie only one copy.

interface : interface is a type of reference data type. It is also known as 100% pure abstract class.

syntax

interface interfaceName {

property

behaviour

}

By default all variable or property in interface are public static and final.

By default all behaviour or methods are public and abstract.

Interface contains only final or constant variable and abstract methods.

interface Abc {

public static final int A=10;

static final int B=20;

final int C=30;

int D=40;

public abstract void dis1();

abstract void dis2();

void dis3();

}

interface Abc {

int A=10;

void dis1();

}

interface Mno {

int B=20;

void dis2();

}

interface Xyz extends Abc,Xyz{

int C=30;

void dis3();

}

class Demo implements Abc,Mno {

provide the body for dis1 and dis2

}

One interface can extends more than one interface. But class can extends only one class.

Using interface we can achieve multiple inheritance. Abc and Mno are super interface and Xyz are sub interface. In Xyz it contains 3 method ie dis1, dis2 and dis3 but all these three methods are incomplete. Class only provide the implementation for interface. Class can implements one or more than one interface. Which ever class implements one or more than one interface that class must be provide the body for all methods belongs to that interface.

Abstract class Vs interface

1. Abstract class can contains normal variable as well as final variable. Interface contains only final or constant variable.
2. Abstract class can contains normal as well as abstract method. but interface contains only abstract methods.
3. Normal class can extends only one abstract class but normal class can implements more than one interface.

Common point between two.

1. We can create object of interface as well as abstract class.
2. Using interface and abstract class we can achieve abstraction. But using abstract class we can achieve partial abstraction and using interface we can achieve 100% pure abstraction.

Package and access specifiers

package : package is a collection of classes and interface which have same name but different functionality or different purpose.

It mainly divided into two types

1. User defined package
2. Pre defined packages or built in package.

User defined package.

education

school college pg

Attendance Attendance Attendance

Java provided totally 4 types of access specifiers. Using access specifiers we can expose visibility or accessibility of classes, variable and methods between same package or other package.

1. private
   1. use with what : we can use with instance variable, static variable, non static method, static method, constructor but we can’t use with local variable and class.
   2. scope : with in a same class.
2. protected
   1. use with what : we can use with instance variable, static variable, non static method, static method, constructor but we can’t use with local variable and class.
   2. scope : within a same package
3. default (nothing or no access specifiers)
   1. use with what : we can use with all.
   2. Scope : within in a same package as well as other package but class must be sub class.
4. public
   1. use with what : we can use with instance variable, static variable, non static method, static method, constructor, class but we can’t use with local variable.
   2. Scope : same as well other package.