Name - Papon

ID - IT-22019

Imagine a vehicle factor where different kinds of vehicles and produced are produced - cars, electric scooters

Characters :

→ abstract class : vehicle (common blueprint, shared logic like

→ Interface : Electricpowered (some vehicles can be charged

Why the difference ?

- All vehicles have a base structure → use an abstract class vehicle .
- Not all vehicles are electric, but some can be → use an interface Electricpowered .

When to use what ?

use abstract class when :

- You want to share common behavior class across multiple related classes .
- You want to prove base implementation

→ Use Interface when?

=> You want to define a contract without enforcing structure.

• You want multiple inheritance to type (Java doesn't allow multiple class Inheritance).

* Java application example:

code:

```
Interface Electricpowered {
    void chargeBattery();
}
abstract class vehicle {
    string brand;
    vehicle (string brand) {
        this.brand = brand;
    }
    abstract void startEngine();
    void stopEngine() {
System.out.println(brand+"; Engine stopped.");
    }
```

```
class Electriccar extends vehicle implements Electricpower{
    Electriccar (String brand){
        super (brand);
    }
    @void startEngine () {
        system.out.println (brand + ": Electric Engine started");
    }

    public void changeBattery (){
        System.out.println (brand + ": charging battery...");
    }
}
class petrolBike extends vehicle{
    petrolBike (String brand){
        super (brand);
    }
    void startEngine (){
        System.out.println (brand + ": petrol engine started.");
    }
}
```

Main class :

```
public class vehicleFactory {

    public static void main (String[] args){

     ElectricCar tesla = new ElectricCar("Tesla");

     petrolBike yamaha = new petrolBike("Yamaha");

     tesla.startEngine();

     tesla: changEngine();

     tesla.stopEngine();

    System.out.println("----------");

    Yamaha.startEngine();

    Yamaha.stopEngine();

    }

}
```

Conclusion :

- Use abstract class when building a family on related object that share code.
- Use interface when you need to support multiple capabilities across unrated classes.

To summarize the differences between abstract class and interface - in - Java

Feature	Abstract class	Interface
Inheritance	A class can Extend only one abstract class	A class implement multiple interfaces
Method Implementation	can have implemented and abstract method	All method are abstract by default
constructor	can have constructor	can not constructo
use case	used for objects that are closely related	used to define a contract or capabilit
Example Scenario	vehicle (base for cars, truck, etc)	Drivable, Flyable, Sepializable
performance overhead	Slightly more overhead due to possible base logic	Usally lighter as t contain no logic