**Day-5 Basics of Java**

1.Singleton Design Pattern:

//BillingServices

**package** com.mycom.designpattern.singleton;

**public** **class** BillingServices {

**private** **static** BillingServices *instance*;

**public** String processpayment;

**public** String OrderDetails;

**public** **static** BillingServices getInstance() {

**if**(*instance* == **null**) {

*instance* = **new** BillingServices();

}

**return** *instance*;

}

**public** **void** processPayment(String paymentDetails) {

System.***out***.println("Processing payment : " + paymentDetails);

}

**public** **void** generateInvoice(String OrderDetails) {

System.***out***.println("Generate Invoice for order : " + OrderDetails);

}

}

//Main Class

**package** com.mycom.designpattern.singleton;

**public** **class** MainBillingServices {

**public** **static** **void** main(String[] args) {

BillingServices singleton = BillingServices.*getInstance*();

singleton.processPayment("Payment details: 100$" );

singleton.generateInvoice("Order Details: Product:1 ,Quantity:2");

}

}

2. Factory Design Factory

//Vehicle class

**package** com.mycom.designpattern.factory;

**public** **interface** Vehicle {

**void** start();

**void** accelerate();

**void** brake();

}

// Car Class

**public** **class** Car **implements** Vehicle{

@Override

**public** **void** start() {

// **TODO** Auto-generated method stub

System.***out***.println("Car is Starting");

}

@Override

**public** **void** accelerate() {

// **TODO** Auto-generated method stub

System.***out***.println("Car is accelerate ");

}

@Override

**public** **void** brake() {

// **TODO** Auto-generated method stub

System.***out***.println("Car is braking");

}

}

// Motor Cycle

**public** **class** MotorCycle **implements** Vehicle {

@Override

**public** **void** start() {

// **TODO** Auto-generated method stub

System.***out***.println("MotorCycle is Starting");

}

@Override

**public** **void** accelerate() {

// **TODO** Auto-generated method stub

System.***out***.println("MotorCycle is accelerate ");

}

@Override

**public** **void** brake() {

// **TODO** Auto-generated method stub

System.***out***.println("MotorCycle is braking");

}

}

// Truck Class

**public** **class** Truck **implements** Vehicle{

@Override

**public** **void** start() {

// **TODO** Auto-generated method stub

System.***out***.println("Truck is Starting");

}

@Override

**public** **void** accelerate() {

// **TODO** Auto-generated method stub

System.***out***.println("Truck is accelerate ");

}

@Override

**public** **void** brake() {

// **TODO** Auto-generated method stub

System.***out***.println("Truck is braking");

}

}

// Vehicle

**public** **class** VehicleFactory {

**public** **static** Vehicle createVehicle(String type) {

**switch** (type.toLowerCase()) {

**case** "car":

**return** **new** Car();

**case** "motorcycle":

**return** **new** MotorCycle();

**case** "truck":

**return** **new** Truck();

**default**:

**throw** **new** IllegalArgumentException("Unknown vehicle type: " + type);

}

}

}

// Main class

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Vehicle Car = VechicleFactory.*createVehicle*("car");

Car.start();

Car.accelerate();

Car.brake();

Vehicle MotorCycle = VechicleFactory.*createVehicle*("MotorCycle");

MotorCycle.start();

MotorCycle.accelerate();

MotorCycle.brake();

Vehicle Truck = VechicleFactory.*createVehicle*("Truck");

Truck.start();

Truck.accelerate();

Truck.brake();

}

}

3. Abstract factory Design

// Shape interface

**package** com.mycom.designpattern.abstractfactory;

**public** **interface** Shape {

**void** draw();

}

// Circle class

**public** **class** Circle **implements** Shape {

@Override

**public** **void** draw() {

System.***out***.println("Drawing a Circle");

}

}

//rectangle class

**public** **class** Rectangle **implements** Shape {

@Override

**public** **void** draw() {

System.***out***.println("Drawing a Rectangle");

}

}

// Square class

**public** **class** Square **implements** Shape {

@Override

**public** **void** draw() {

System.***out***.println("Drawing a Square");

}

}

//Abstract class

**public** **abstract** **class** AbstractFactory {

**abstract** Shape getShape(String shapeType);

}

//ShapeFactory class

**public** **class** ShapeFactory **extends** AbstractFactory{

@Override

Shape getShape(String shapeType) {

**if** (shapeType == **null**) {

**return** **null**;

}

**if** (shapeType.equalsIgnoreCase("CIRCLE")) {

**return** **new** Circle();

} **else** **if** (shapeType.equalsIgnoreCase("RECTANGLE")) {

**return** **new** Rectangle();

} **else** **if** (shapeType.equalsIgnoreCase("SQUARE")) {

**return** **new** Square();

}

**return** **null**;

}

}

//Main class

**public** **class** AbstractFactoryPatternDemo {

**public** **static** **void** main(String[] args) {

AbstractFactory shapeFactory = **new** ShapeFactory();

// Get an object of Circle and call its draw method.

Shape shape1 = shapeFactory.getShape("CIRCLE");

shape1.draw();

// Get an object of Rectangle and call its draw method.

Shape shape2 = shapeFactory.getShape("RECTANGLE");

shape2.draw();

// Get an object of Square and call its draw method.

Shape shape3 = shapeFactory.getShape("SQUARE");

shape3.draw();

}

}

4. Immutable Design Pattern

// Employee class

**package** com.mycom.designpattern.immutable;

**public** **final** **class** Employee {

**private** **final** String firstname;

**private** **final** String lastname;

**private** **final** String DateOfBirth;

**private** **final** **int** employeeId;

**private** **final** String joiningDate;

**private** **final** **double** salary;

**public** Employee(String firstname, String lastname, String dateOfBirth, **int** employeeId, String joiningDate,

**double** salary) {

**super**();

**this**.firstname = firstname;

**this**.lastname = lastname;

DateOfBirth = dateOfBirth;

**this**.employeeId = employeeId;

**this**.joiningDate = joiningDate;

**this**.salary = salary;

}

**public** String getFirstname() {

**return** firstname;

}

**public** String getLastname() {

**return** lastname;

}

**public** String getDateOfBirth() {

**return** DateOfBirth;

}

**public** **int** getEmployeeId() {

**return** employeeId;

}

**public** String getJoiningDate() {

**return** joiningDate;

}

**public** **double** getSalary() {

**return** salary;

}

}

// Main Class

**public** **class** MainEmployee {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Employee emp = **new** Employee("Joe", "Deo", "1989-12-13", 123456, "2022-07-04", 30000.00);

System.***out***.println("Employee Details:");

System.***out***.println("First Name: " + emp.getFirstName());

System.***out***.println("Last Name: " + emp.getLastName());

System.***out***.println("Date of Birth: " + emp.getDateOfBirth());

System.***out***.println("Employee ID: " + emp.getEmployeeId());

System.***out***.println("Joining Date: " + emp.getJoiningDate());

System.***out***.println("Salary: " + emp.getSalary());

}

}