Day 10

Created	@Mar 19, 2021
Created by	Ⓥ VIGNESH J
Tags	Java

Factory Patter

- Factories creates objects
- Don't sell
- Gives to the builder (Assembler)
-

```java
package ga.veee.day10;

public class BuilderDemo {
    public static void main(String[] args) {
        Computer myComputer = new Computer.ComputerBuilder("i7 core processor", "12gb").build();

        System.out.println(myComputer);

        Computer myCom2 = new Computer.ComputerBuilder("18 core proccsss", "16gb ram").setGraphicsCard("graphics card")
                .setHdd("my new hdd").build();

        System.out.println(myCom2);
    }
}

class Computer {
    //fixed properties
    private String motherBoard;
    private String ram;

    public String getMotherBoard() {
        return motherBoard;
    }

    public void setMotherBoard(String motherBoard) {
        this.motherBoard = motherBoard;
    }

    public String getRam() {
        return ram;
    }

    public void setRam(String ram) {
        this.ram = ram;
    }

    public String getGraphicsCard() {
        return graphicsCard;
    }
    public void setGraphicsCard(String graphicsCard) {
        this.graphicsCard = graphicsCard;
    }

    public String getHdd() {
        return hdd;
    }

    public void setHdd(String hdd) {
        this.hdd = hdd;
    }

    //optional Properties
    private String graphicsCard;
    private String hdd;

    @Override
    public String toString() {
        return "Computer [motherBoard=" + motherBoard + ", ram=" + ram + ", graphicsCard=" + graphicsCard + ", hdd="
                + hdd + "]";
    }

    public Computer(ComputerBuilder builder) {
        this.motherBoard = builder.motherBoard;
        this.ram = builder.ram;
        this.graphicsCard = builder.getGraphicsCard();
        this.hdd = builder.getHdd();
    }

    public static class ComputerBuilder {
        //fixed properties
        private String motherBoard;
        private String ram;
        //optional Properties
        private String graphicsCard;
        private String hdd;

        public ComputerBuilder(String motherBoard, String ram) {
            this.motherBoard = motherBoard;
            this.ram = ram;
        }

        public String getGraphicsCard() {
            return graphicsCard;
        }

        public ComputerBuilder setGraphicsCard(String graphicsCard) {
            this.graphicsCard = graphicsCard;
            return this;
        }

        public String getHdd() {
            return hdd;
        }

        public ComputerBuilder setHdd(String hdd) {
            this.hdd = hdd;
            return this;
        }

        public Computer build() {
            return new Computer(this);
        }
    }

}
```

Best Free Online Video Recording & Screen Capture Software | Fluvid

Fluvid is the best all-in-one online screen capture & video recording software that is available for free. Fluvid helps you record, edit, communicate & share your video messages in the most simplest way.

https://fluvid.com/videos/detail/n8QdniZOq3hXdkP22#.YFRyE0eGdsk.link

Creational Pattern

Interface

- To create component
- Dynamic binding through interface
-

```java
package ga.veee.day10;

import java.lang.reflect.InvocationHandler;
import java.lang.reflect.Method;
import java.lang.reflect.Proxy;

public class InterfaceDemo {
    public static void main(String[] args) {
        Object obj = new AImpl();

        Importer importObj = (Importer) obj;
        importObj.testA();

        Exporter exportImplObj = new ExporterImpl();
        obj = Proxy.newProxyInstance(importObj.getClass().getClassLoader(),
                new Class[]{Importer.class, Exporter.class, MegaExporter.class},
                new MyInvocationHandler(new Object[]{importObj, exportImplObj}));

        Importer iObj = (Importer) obj;

        Exporter eObj = (Exporter) obj;

        MegaExporter mObj = (MegaExporter) obj;

        iObj.testA();
        eObj.doExport();
        String result = mObj.doMegaExport("mega export...");
        System.out.println("The result....is..:" + result);

    }
}

class MyInvocationHandler implements InvocationHandler {
    Object obj[];

    public MyInvocationHandler(Object obj[]) {
        this.obj = obj;
    }

    @Override
    public Object invoke(Object proxy, Method method, Object[] args) throws Throwable {
        Object returnObj = null;
        for (Object o : obj) {
            Method m[] = o.getClass().getDeclaredMethods();
            for (Method met : m) {
                if (met.getName().equals(method.getName())) {
                    met.setAccessible(true);
                    returnObj = method.invoke(o, args);
                }
            }
        }
        return returnObj;
    }
}

interface Importer {
    public void testA();
}

class AImpl implements Importer {
    @Override
    public void testA() {
        System.out.println("test a is called....");
    }
}

interface Exporter {
    public void doExport();
}

interface MegaExporter {
    public String doMegaExport(String data);
}

class ExporterImpl implements Exporter, MegaExporter {
    @Override
    public void doExport() {
        System.out.println("do export method called....");
    }

    @Override
    public String doMegaExport(String data) {
        return "return value is..................:" + data;
    }
}
```