Academia Xideral Java Monterrey

Daniel Ivan Anaya Alvarez

Activity Builder

In this activity we are going to use a method named builder as for the case I use it was basically a Laptop Factory that can make office or gaming setups. This method establish an external builder from an interface that sets the parameter. The order I work with this builder was the following:

1. First I made the object in this case was the Laptop, there I defined everything its going to have
2. Then I made the Builder, that established all the methods it needed in order to get the Laptop
3. Then we got the LaptopBuilder that implements all the method from the interface that program is in charge to actually build the object.
4. The Director is the one who tell the details about what type of Laptop it needed to build
5. Lastly we have the Client that is the one who asks for the Laptop.

Laptop.java

```
package ActBuilder;

public class Laptop {

	//Create the object instances

    private String cpu;
    private String screen;
    private int keys;
    private String gpu;
    private Brand brand;

    // Constructors for each component
    public void setCpu(String cpu) { this.cpu = cpu; }
    public void setScreen(String screen) { this.screen = screen; }
    public void setKeys(int keys) { this.keys = keys; }
    public void setGpu(String gpu) { this.gpu = gpu; }
    public void setBrand(Brand brand) { this.brand = brand; }

    @Override
    //Override how will it prints
    public String toString() {
        return "Laptop [CPU=" + cpu + ", Screen=" + screen + ", Keys=" + keys +
", GPU=" + gpu + ", Brand=" + brand + "]";
    }
}
```

Builder.java

```java
package ActBuilder;

public interface Builder {
	void reset();
	void setCpu(String cputype);
	void setScreen(String screentype);
	void setKeys(int numKeys);
	void setGpu(String gpuModel);
	void setBrand(Brand brand);

}
```

LaptopBuilder.java

```java
package ActBuilder;

public class LaptopBuilder implements Builder {

	  //Encapsulate
    private Laptop laptop;

    @Override
    public void reset() {
        this.laptop = new Laptop(); // Start with a new Laptop instance
    }
    //Set every parameter from the laptop

    @Override
    public void setCpu(String cpuType) {
        laptop.setCpu(cpuType);
    }

    @Override
    public void setScreen(String screenType) {
        laptop.setScreen(screenType);
    }

    @Override
    public void setKeys(int numKeys) {
        laptop.setKeys(numKeys);
    }

    @Override
    public void setGpu(String gpuModel) {
        laptop.setGpu(gpuModel);
    }

    @Override
    public void setBrand(Brand brand) {
        laptop.setBrand(brand);
```

```
    }

    //Return the build Laptop result
    public Laptop getResult() {
        return this.laptop; //
    }
}
```

Director.java

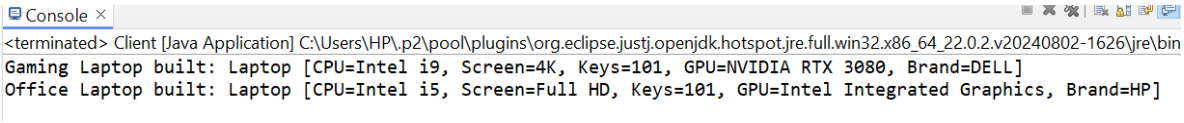
```
package ActBuilder;

public class Director {

    // Method to build a Gaming Laptop
    public void makeGamingLaptop(Builder builder) {
        builder.reset();
        builder.setCpu("Intel i9");
        builder.setScreen("4K");
        builder.setKeys(101);
        builder.setGpu("NVIDIA RTX 3080");
        builder.setBrand(Brand.DELL);
    }

    // Method to build an Office Laptop
    public void makeOfficeLaptop(Builder builder) {
        builder.reset();
        builder.setCpu("Intel i5");
        builder.setScreen("Full HD");
        builder.setKeys(101);
        builder.setGpu("Intel Integrated Graphics");
        builder.setBrand(Brand.HP);
    }
}
```

Client.java

```
package ActBuilder;

public class Client {
    public static void main(String[] args) {
        Director director = new Director();
        LaptopBuilder builder = new LaptopBuilder();

        // Build a Gaming Laptop
        director.makeGamingLaptop(builder);
        Laptop gamingLaptop = builder.getResult();
        System.out.println("Gaming Laptop built: " + gamingLaptop);

        // Build an Office Laptop
        director.makeOfficeLaptop(builder);
        Laptop officeLaptop = builder.getResult();
```

```
        System.out.println("Office Laptop built: " + officeLaptop);
    }
}
```

Print



Conclusion

Builder is a very useful method that can encapsulate a lot of information and settings to a point that you just ask what you want on a single class and the builder will just do it.