University of Information Technology and Sciences (UITS)
Department of Information Technology

Lab Report : 5

IT-324 : Design Pattern Lab

Builder Pattern

Submitted To:

Sifat Nawrin Nova
Lecturer,
Department of IT, UITS
Email:sifat.nawrin@uits.edu.bd

Submitted By:

Name:Nazmul Zaman
Student ID:2014755055
Department of IT, UITS

Contents

1 Abstraction

In this lab report we learn about builder pattern method and how it's work and why we used this method.

The builder pattern is a design pattern designed to provide a flexible solution to various object creation problems in object-oriented programming. The intent of the Builder design pattern is to separate the construction of a complex object from its representation..

2 Theory

Builder pattern aims to "Separate the construction of a complex object from its representation so that the same construction process can create different representations." It is used to construct a complex object step by step and the final step will return the object. The process of constructing an object should be generic so that it can be used to create different representations of the same object.

3 Objective

In this section we Discuss the remarks of using this pattern.Discuss the remarks of using this pattern.

Consider we want to implement a Car Manufacturing through carbuilder,car configuration,shop. Let's try to implement this with the help of builder pattern method.We will also learn advantages and disadvantages of builder pattern and also real life example through the working procedure same like real life example.

4 When would you use a builder design pattern

The builder pattern, as the name implies, is an alternative way to construct complex objects. This pattern should be used when we want to build different immutable objects using the same object building process.

5 Advantage and disadvantage

Advantage : It provides clear separation between the construction and representation of an object. It provides better control over construction process. It supports to change the internal representation of objects.

Disadvantage : verbose and code duplication as Builder needs to copy all fields from Original or Item class.

6 Source Code

6.1 Shop class/Main Class)

```
1  package builderpattern;
2
3  public class Shop {
4
5
6      public static void main(String  args [])
7      {
8
9
10
11         //  Car information (Car name: Toyota,seats: 7,
       Horsepower : 3000cc,Weight : 2000kg , Battery: 1600 )
12
13
14         Car p = new Car("Car name : Toyota", 7, "3000CC", 2000,
       1600);
15
16         System.out.println(p);
17     }
18
19 }
```

?

6.2 (Car Class)

```
1  package builderpattern;
2
```

```
3  public class Car {
4
5      private String name;
6      private int seats;
7      private String horsepower;
8      private double weight;
9      private int battery;
10
11     public Car(String name ,int seats, String horsepower,
       double weight, int battery)
12     {
13         super();
14
15         this.name = name;
16         this.seats = seats;
17
18         this.horsepower = horsepower;
19
20         this.weight = weight;
21         this.battery = battery;
22     }
23
24
25     public String toString ()
26     {
27         return name+" "+seats+" "+horsepower+" "+weight+" "+
       battery;
28     }
29 }
```

?

6.3 (CarbuilderClass)

```
1  package builderpattern;
2
3  public class Carbuilder {
4
5      private String name;
6      private int  seats;
7      private String horsepower;
8      private double weight;
9      private int  battery;
10
11     public Carbuilder setos(String name)
12     {
13         this.name = name;
14         return this;
15
16
17     }
18
19      public Carbuilder setram(int seats)
20     {
21         this.seats = seats;
22         return this;
23     }
24
25     public Carbuilder setprocessor(String horsepower)
26     {
27         this.horsepower = horsepower;
28
29         return this;
30     }
31         public Carbuilder setscreensize(double weight)
32     {
33         this.weight = weight;
34         return this;
35     }
36     public Carbuilder setbattery(double setbattery)
37     {
38         this.battery = battery;
39
40         return this;
41     }
42
43     public Car getPhone()
44     {
45         return new Car(name,seats,horsepower,weight,battery);
46     }
47
48 }
```

?

6.4 Output

```
run:
Car name : Toyota 7 3000CC 2000.0 1600
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 1: Output

7 Conclusion

In this lab report we learn about builder pattern in practically. How we used builder pattern design pattern with real life and also getting deeply knowledge about builder pattern method and how does it works.

8 References

1.https://www.tutorialspoint.com/design$_p$*attern*/*singleton*$_p$*attern.htm*
2.*https* : *//www.geeksforgeeks.org/singleton−design−pattern−introduction*
3.*https* : *//www.javatpoint.com/singleton* − *design* − *pattern* − *in* − *java*