Agenda

Builder Design Pattern

Builder Design Pattern :

class student {

- name
- batch
- id
- year
- gradyear
- course
- age
- email
- psp

}

class car {

- engine
- # doors
- # wheels
- color
- reg no
- groundClearance
- priceOnRoad
- launch year
- # of seats

}

```
class TradeOrder{
    - tokenfrom
    - token To
    - tokenValueBTC
    - date
    - user
    - stopLossAdded
    - tokenValueINR
    - walletAdd
    - Exchange
}
```

① Setters

object already created

```
Student st = new Student();
```

```
st.setName();
st.setAge();
   :
st.setBatch();
```

```
class Student {
    void setAge(int x){
        if(x<18){
            throw new Exceptio
        }
        this.x = x;
}
```

1. How to validate the parameters without creating an object.

```
class Authoz {
    boolean login(Studenl st) {
        //some checks on st
    }
}
```

② Constructors :

```
                                  name       age.
Student st = new Student("Akash", 15,
                          "SeptLLD", 1);
```

↳ ① order of params can go wrong

② Number of params which you want to initialise.

③ Too many Consructors

④ No constructor possible

```
Student(id)
student(id nam)
student(id name batch)
student(-- --)
student(- -- - -)
```

Telescopic Constructors

```
Student(String name, double psp);
Student(String batch, double age);
```

③ Map <String, Object>

```
Student st = new Student( ______ )
```

one data structure storing all param

```
class Student {
  student (Map<-,-> m) {
     if (m.containsKey("id"))
    {   this.id = m.get("id");
    }
       ≡
}
```

```
Map<String, Object> m = new HashMap();
 "ids" ←
m.set("id", 123); "xyz"
m.set("name", "Akash");
   "namee"         123
```

① No type validation (compile time) ←

② Param name mismatch

4 Solution → Map like Properties
&
Can also type check in compile time.

```
class Student {
    String name;
    int id;
     :

  Student(StudentMap smap){
   this.name = smap.name;
   this.id = smap.id;

  }
}
```

```
Class StudentMap {
    String name;
    String batch;
    int age;
     int psp;
    // Setters
}
```

```
StudentMap smap = new StudentMap();
    smap.setAge(19);
    smap.setName("Akash");
     :
StudentMap st = new Student(smap);
```

```
class Authorize {
✗ void login(StudentMap smap){
}

class Student {

    public Student (StudentMap
                            smap){
        if (s.map.id = = null){
            throw new Exception();
        }
        this.id = smap.id.

        if (smap.age != null &
                  smap.age > 20){
            this. age = age;
        }
}
```

This StudentMap is a helper class to validate & build Student Object.

→ Builder Class

StudentMap → StudentBuilder

( ) Builder Design Pattern

Break time : 8:15