

بررسی الگوهای طراحی

Builder

توسط : محمد حسین زارع

@zare88r: twitter

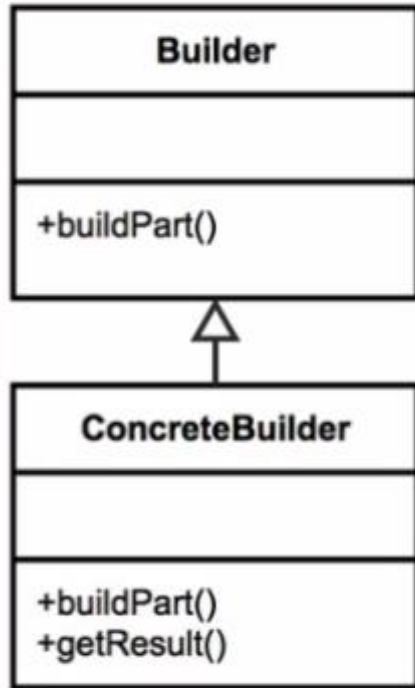
telegram

Join Us : @JavaLandCH

مفاهیم

- **handle کردن constructor های پیچیده**
- **تعداد پارامترهای بالا**
- **Immutability**
- **نمونه ها :**
- **StringBuilder**
- **DocumentBuilder**
- **Locale.Builder**

طراحی



- انعطاف پذیری در مقابل constructor ها
- استفاده از inner class ها
- متوذهایی برای جایگزینی setter ها
- ایجاد کلاسی برای Build کردن

نمونه موجود

```
StringBuilder builder = new StringBuilder();  
builder.append("This is an example of Builder Pattern")  
        .append(" , You can use this pattern ")  
        .append("instead of calling setters ")  
        .append(" or calling complex constructors");  
String result = builder.toString();
```

یک نمونه

```
public class X264Properties {  
    private int keyInt;  
    private int minKeyInt;  
  
    public X264Properties(int keyInt, int minKeyInt) {  
        this.keyInt = keyInt;  
        this.minKeyInt = minKeyInt;  
    }  
  
    public int getKeyInt() {  
        return keyInt;  
    }  
    public void setKeyInt(int keyInt) {  
        this.keyInt = keyInt;  
    }  
    public int getMinKeyInt() {  
        return minKeyInt;  
    }  
    public void setMinKeyInt(int minKeyInt) {  
        this.minKeyInt = minKeyInt;  
    }  
}
```

یک نمونه

```
public class X264PropertiesBuilder {  
    private int keyInt;  
    private int minKeyInt;  
  
    public X264PropertiesBuilder keyInt(int keyInt) {  
        this.keyInt = keyInt;  
        return this;  
    }  
  
    public X264PropertiesBuilder minKeyInt(int minKeyInt) {  
        this.minKeyInt = minKeyInt;  
        return this;  
    }  
  
    public X264Properties build() {  
        return new X264Properties(keyInt, minKeyInt);  
    }  
}
```

جمع بندی

- راهی خلاقانه برای رفع پیچیدگی های ساخت یک کلاس
- پیاده سازی ساده
- امکان پیاده سازی هم به شکل `inner class` و هم یک کلاس مستقل را داراست
- **Designed First** میباشد