

بررسی الگوهای طراحی -

Singleton

توسط : محمد حسین زارع

@HoseinZare67: twitter

telegram

Join Us : @JavaLandCH

مفاهیم

- ساخت فقط و فقط یک نمونه از یک شی
- تضمین کنترل منابع
- بارگذاری Lazy
- نمونه ها :

- .1 Runtime
- .2 Logger
- .3 Spring Beans

طراحی

Singleton
- <u>singleton : Singleton</u>
- Singleton()
+ <u>getInstance() : Singleton</u>

- اصل static بودن
- نیاز به Thread Safe بودن
- نمونه موجود ولی private
- سازنده private
- سازنده بدون پارامتر

نمونه موجود

```
Runtime firstInstance = Runtime.getRuntime();  
firstInstance.gc();  
System.out.println(firstInstance);  
  
Runtime anotherInstance = Runtime.getRuntime();  
System.out.println(anotherInstance);  
  
if(firstInstance == anotherInstance){  
    System.out.println("Two instane are equal");  
}
```

یک نمونه

```
public class DBConnection {  
    private static DBConnection dbConnection = new DBConnection();  
    private DBConnection() {  
    }  
    public static DBConnection getDbConnection() {  
        return dbConnection;  
    }  
}
```

- Singleton میتواند از روشهایی مثل **Lazy loading** استفاده کرد
- همچنین در برنامه های **concurrent** باید اینگونه کلاس ها را **Thread safe** نوشت

جمع بندی

با پیاده سازی این الگو قادر خواهیم بود از یک کلاس همیشه یک نمونه بسازیم ، اگر شرایطی داریم که ممکن است نمونه های مختلفی با حالات مختلفی داشته باشیم باید به سراغ **factory** برویم

این الگویی از ساده ترین الگو ها از مجموعه GOF میباشد

Singleton انعطاف پذیری ندارد لذا باید با شناخت کافی از آن استفاده کرد