链滴

《Head First 设计模式》：单件模式

作者：jingqueyimu

原文链接：https://ld246.com/article/1596289989567

来源网站：链滴

正文

一、定义

单件模式确保一个类只有一个实例，并提供一个全局访问点。

要点：

- 定义持有唯一单件实例的类变量。
- 私有化构造，避免其他类产生实例。
- 对外提供获取单件实例的静态方法。

二、实现步骤

1、创建单件类

（1）方式一：懒汉式

延迟创建单件实例。

线程不安全：

```
/**
 * 单件类（懒汉式、线程不安全）
 */
public class Singleton {
```

```
    /**
     * 唯一单件实例
     */
    private static Singleton uniqueInstance;

    /**
     * 私有构造
     */
    private Singleton() {}

    /**
     * 获取单件实例
     */
    public static Singleton getInstance() {
        if (uniqueInstance == null) {
            // 延迟创建单件实例
            uniqueInstance = new Singleton();
        }
        return uniqueInstance;
    }
}
```

线程安全：

```
/**
 * 单件类（懒汉式、线程安全）
 */
public class Singleton {

    /**
     * 唯一单件实例
     */
    private static Singleton uniqueInstance;

    /**
     * 私有构造
     */
    private Singleton() {}

    /**
     * 获取单件实例（同步方法）
     */
    public static synchronized Singleton getInstance() {
        if (uniqueInstance == null) {
            // 延迟创建单件实例
            uniqueInstance = new Singleton();
        }
        return uniqueInstance;
    }
}
```

（2）方式二：饿汉式

“急切”创建单件实例。

```
/**
 * 单件类（饿汉式）
 */
public class Singleton {

    /**
     * 唯一单件实例（“急切”创建单件实例）
     */
    private static Singleton uniqueInstance = new Singleton();

    /**
     * 私有构造
     */
    private Singleton() {}

    /**
     * 获取单件实例
     */
    public static Singleton getInstance() {
        return uniqueInstance;
    }
}
```

（3）方式三：双检锁

```
/**
 * 单件类（双重检查加锁）
 */
public class Singleton {

    /**
     * 唯一单件实例
     */
    private volatile static Singleton uniqueInstance;

    /**
     * 私有构造
     */
    private Singleton() {}

    /**
     * 获取单件实例
     */
    public static Singleton getInstance() {
        if (uniqueInstance == null) {
            synchronized (Singleton.class) {
                if (uniqueInstance == null) {
                    uniqueInstance = new Singleton();
                }
            }
        }
        return uniqueInstance;
    }
}
```

2、使用单件类获取唯一单件实例

```
public class Test {

    public static void main(String[] args) {
        // 获取单件实例
        Singleton singleton = Singleton.getInstance();
        System.out.println(singleton);
        Singleton singleton2 = Singleton.getInstance();
        System.out.println(singleton2);
    }
}
```