

Java多线程基础（四）——Immutable模式



Ressimix
发布于 2018-07-07

一、定义

Immutable是“永恒的”“不会改变”的意思。在Immutable Patttern中，有着能够保证实例状态绝不会改变的类（immutable 类）。因为访问这个实例时，可以省去使用共享互斥机制所会浪费的时间，提高系统性能。java.lang.String就是一个Immutable的类。

二、模式案例

案例：

Person类，具有姓名（name）、地址（address）等字段。字段都是私有的，只能通过构造器来设置，且只有get方法，没有set方法。这时，即使有多个线程同时访问相同实例，Person类也是安全的，它的所有方法都不需要定义成synchronized。

Person定义：

```

public final class Person {
    private final String name;
    private final String address;
    public Person(String name, String address) {
        this.name = name;
        this.address = address;
    }
    public String getName() {
        return name;
    }
    public String getAddress() {
        return address;
    }
    public String toString() {
        return "[ Person: name = " + name + ", address = " + address + " ]";
    }
}

```

线程定义：

```

public class PrintPersonThread extends Thread {
    private Person person;
    public PrintPersonThread(Person person) {
        this.person = person;
    }
    public void run() {
        while (true) {
            System.out.println(Thread.currentThread().getName() + " prints " + person);
        }
    }
}

```

执行：

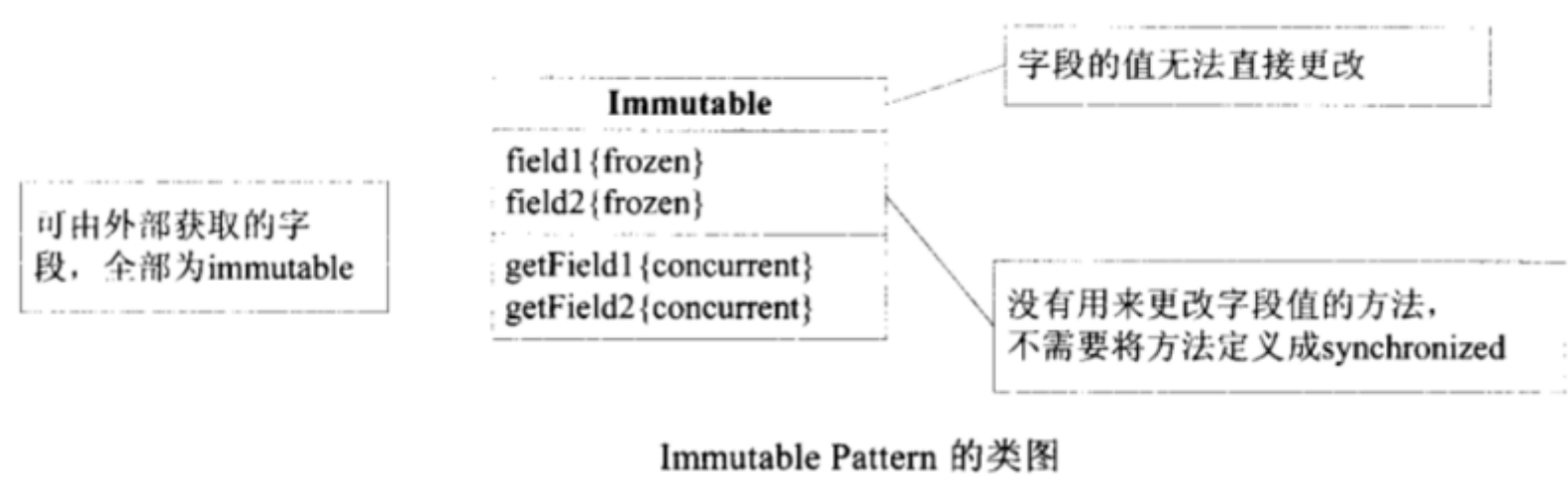
```
public class Main {
    public static void main(String[] args) {
        Person alice = new Person("Alice", "Alaska");
        new PrintPersonThread(alice).start();
        new PrintPersonThread(alice).start();
        new PrintPersonThread(alice).start();
    }
}
```

三、模式讲解

Immutable模式的角色如下：

- Immutable(不变的)参与者

Immutable参与者是一个字段值无法更改的类，没有任何用来更改字段值的方法。当Immutable参与者的实例建立后，状态就完全不再变化。



适用场景：

Immutable模式的优点在于，“不需要使用synchronized保护”。而“不需要使用synchronized保护”的最大优点就是可在不丧失安全性与生命性的前提下，提高程序的执行性能。若实例由多数线程所共享，且访问非常频繁，Immutable模式就能发挥极大的优点。

[多线程](#) [java](#)


阅读 4.5k • 更新于 2018-08-02

👍 赞 7

🔖 收藏 1

🔗 分享

本作品系原创，采用《署名-非商业性使用-禁止演绎 4.0 国际》许可协议



透彻理解Java并发编程

Java并发编程是整个Java开发体系中最难以理解但也是最重要的知识点，也是各类开源分布式框架中各...

关注专栏



Ressmix
1.2k 声望 1.3k 粉丝

关注作者

6 条评论

得票数

最新



撰写评论 ...