Java 核心技术

第六章　static、final和常量设计

第二节 单例模式

华东师范大学 陈良育

单例模式(1)

- 单例模式，又名单态模式, Singleton。
- 限定某一个类在整个程序运行过程中，只能保留一个实例对象在内存空间。
- 单例模式是GoF的23种设计模式(Design Pattern)中经典的一种，属于创建型模式类型。

单例模式(2)

- 设计模式：在软件开发过程中，经过验证的，用于解决在特定环境下的、重复出现的、特定问题的解决方案。
- 设计模式起源于建筑领域。Alexander总结了建筑行业的设计模式。

- 1995年Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides(GoF)合著的《设计模式—可复用面向对象软件基础》总结了常见的23种设计模式，包括：创建型、结构型和行为型。

单例模式(3)

- 单例模式：保证一个类有且只有一个对象
  - 采用static 来共享对象实例
  - 采用private 构造函数，防止外界new操作
  - 查看Singleton例子

单例模式(4)

- 总结
  - 设计模式：是经过验证的、用于某些特定场合的解决方案
  - GoF提出23种设计模式：创建型、结构型和行为型
  - 单例模式保证一个类在内存空间中只有一个对象

代码(1)Singleton.java


```java
public class Singleton {
    private static Singleton obj = new Singleton(); //共享同一个对象
    private String content;

    private Singleton()  //确保只能在类内部调用构造函数
    {
        this.content = "abc";
    }

    public String getContent()  {
        return content;
    }
    public void setContent(String content) {
        this.content = content;
    }

    public static Singleton getInstance()   {
        //静态方法使用静态变量
        //另外可以使用方法内的临时变量，但是不能引用非静态的成员变量
        return obj;
    }
```

```java
    public static void main(String[] args) {
        Singleton obj1 = Singleton.getInstance();
        System.out.println(obj1.getContent());  //abc

        Singleton obj2 = Singleton.getInstance();
        System.out.println(obj2.getContent());  //abc

        obj2.setContent("def");
        System.out.println(obj1.getContent());
        System.out.println(obj2.getContent());

        System.out.println(obj1 == obj2); //true, obj1和obj2指向同一个
    }

}
```

谢 谢！