Day10---设计模式

一、什么是设计模式

设计模式（Design pattern）是一套被反复使用、多数人知晓的、经过分类编目的、代码设计经验的总结。使用设计模式是为了可重用代码、让代码更容易被他人理解、保证代码可靠性。 毫无疑问，设计模式于己于他人于系统都是多赢的；设计模式使代码编制真正工程化；设计模式是软件工程的基石脉络，如同大厦的结构一样。

设计模式代表了最佳的实践，通常被有经验的面向对象的软件开发人员所采用。设计模式是软件开发人员在软件开发过程中面临的一般问题的解决方案。这些解决方案是众多软件开发人员经过相当长的一段时间的试验和错误总结出来的。

设计模式有六大原则：

1. 开闭原则。就是说模块应对扩展开放，而对修改关闭。
2. 里氏代换原则。如果调用的是父类的话，那么换成子类也完全可以运行。
3. 依赖倒转原则。把父类都替换成它的子类，程序的行为没有变化。
4. 接口隔离原则，每一个接口应该是一种角色，不多不少，不干不该干的事，该干的事都要干。
5. 单一职责原则。
6. 迪米特法则。 最少知识原则。

二、单例设计模式

单例模式是一种常用的软件设计模式。在它的核心结构中只包含一个被称为单例类的特殊类。通过单例模式可以保证系统中一个类只有一个实例而且该实例易于外界访问，从而方便对实例个数的控制并节约系统资源。

如果希望在系统中某个类的对象只能存在一个，单例模式是最好的解决方案。

2.1 最简单的单例：使用对象字面量

```
var mySingleton = {
    name: "张三",
    age: 20,
    //其实方法应该封装到原型内，此处省略
    eat: function () {
        console.log('太好吃了');
    }
};
```

2.2 能创建单例的函数

```
function Person(name){
  this.name = name;
}
Person.prototype.eat = function  () {
  alert("吃的挺好的");
}

var createSingle = (function () {
  var person = null;
  return function(name){
    if(!person){
      person = new Person(name);
    }
    return person;
  }
})();
var person = createSingle("zhangsan");
alert(person.name);
```

3.3 单例应用：使用单例创建模态对话框

```
```

三、工厂模式

工厂模式：通过工厂函数获取想要的内容

利：简化创造对象的操作，只需要调用函数就可以获取对象

弊：无法判断出对象的归属

四、适配器模式

五、观察者模式