学号 E21714067 专业 软件工程 姓名 石月

实验日期 2020/5/11 教师签字 成绩

实验报告

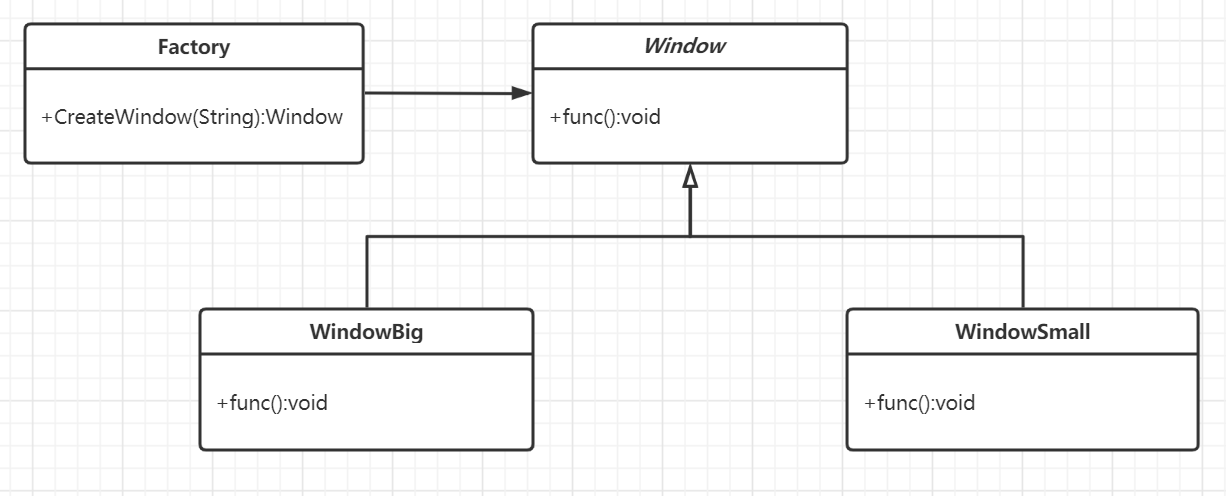
【实验名称】 设计模式

【实验内容】

1. 工厂模式

当使用new时，一般是在实例化一个具体类，因此是面向实现编程，但是这就违背了一条设计原则：针对接口编程，而不是针对实现编程。代码如果绑定着具体类，会导致我们的代码缺乏弹性，变得脆弱，这样的代码，一旦需求有变化或扩展，就必须重新打开这段代码进行检查和修改，而这样直接修改现有代码将很容易出错，并且给代码的更新与维护带来困难。因此，我们需要把代码中变与不变的部分分离出来。

实验代码使用了简单工厂，当然简单工厂并不算是一个真正的设计模式，更像是一种编程习惯，下面是实验代码的UML图。



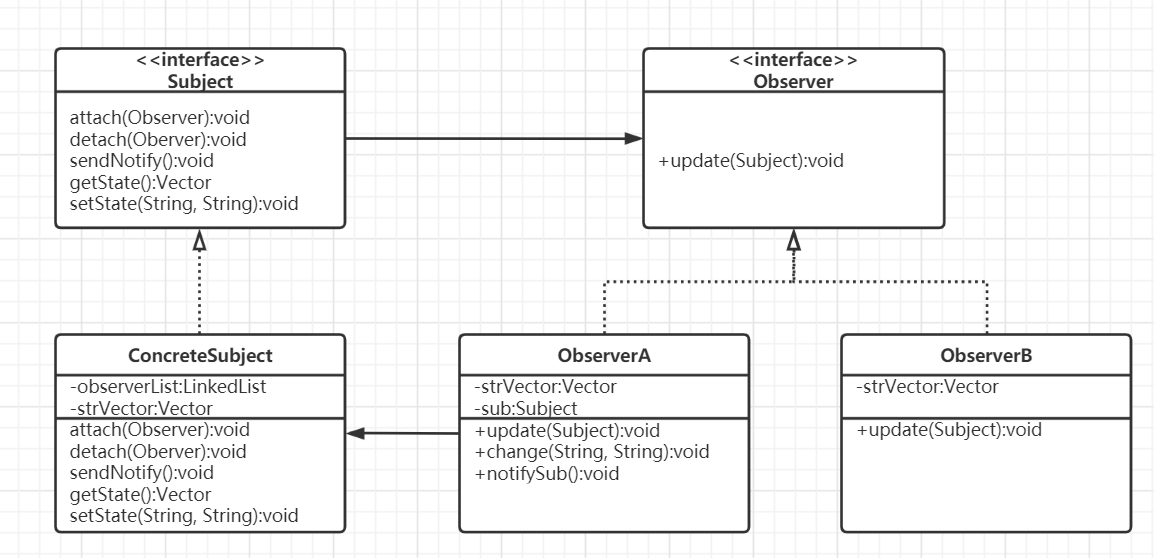
客户端（也就是代码冲的main()方法）通过Factory取得Window的实例，Factory是创建Windows的工厂，从代码中可以看到，Factory的CreateWindow()方法是唯一用到具体窗口类的地方；而Window为抽象类，具有func()方法，其实现可以被覆盖；WindowBig和WindowSmall是工厂的具体产品，继承自Window类，这样他们就可以被工厂创建并返回给用户了。

1. 观察者模式

观察者模式是JDK中使用最多的模式之一，简单地理解这个设计模式，可以表述为：观察者模式可以帮助对象知悉现况，不会错过该对象“感兴趣”的事情，对象甚至可以在运行时决定是否要继续被通知。

可以将观察者模式类比为报纸的订阅：出版社+订阅者=观察者模式。观察者模式定义了一系列对象之间的一对多关系：主题对象管理某些数据，当数据改变时，就会通知观察者；观察者对象需要注册主题以便在该主题数据改变时能够收到更新。观察者模式的实现方法不止一种，但是以包含Subject和Observer接口的设计最常见。

实验代码的UML图如下：



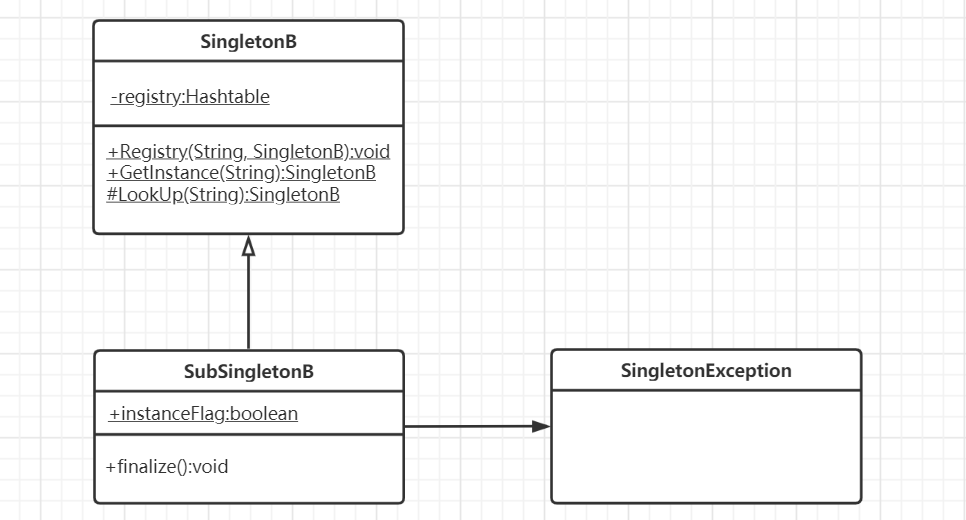
Subject是主题接口，ConcreteSubject实现该接口，能够注册、解绑观察者、向观察者发送数据更新通知，并获取和更新状态字符串；Observer是观察者接口，ObserverA和ObserverB都实现此接口，这样主题在需要通知观察者时，有了一个共同的接口；ObserverA可以通过调用更改主题，ObserverB不能更改主题，但是可以按字母序打印内容向量。

1. 单例模式

有一些对象在程序中其实只需要一个并且也只能有一个，例如线程池、缓存

等，如果制造出多个实例，就会导致许多问题产生。单件模式可以确保只有一个实例会被创建，也提供了一个全局访问点，具有全局变量的方便性还剔除了全局变量的缺点。经典的实现单件模式的方法是：提供私有的构造器、一个静态方法获取实例，一个静态类变量来表示这个唯一实例。实验代码并没有采用这种方法，而是采用了注册表（Registry）的方法，因为涉及到了继承的问题，由于单件的实现是利用静态变量，直接继承会导致所有的派生类共享一个实例变量，可能会带来麻烦，因此想要子类工作顺利，基类SingletonB必须实现注册表功能。

实验代码的UML类图如下：



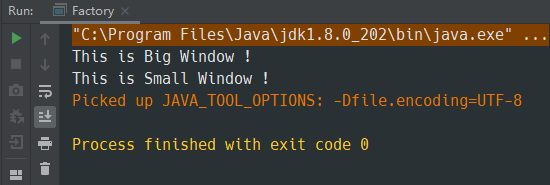
SingletonB中有一个Hashtable，用来存放实例的名字和实例的映射，因此可以避免多个实例的产生，GetInstance()可以获取到指定名字的实例对象，即去Hashtable中查找对应key的value并返回。SubSingletonB继承自SingletonB，有一个静态变量instanceFlag，用来标记是否产生了实例，其构造函数中将进行实例化，如果当前还没创造实例，就会实例化一个名为Sub1的对象，并修改instanceFlag，否则会抛出异常。SingletonException为自定义的一个异常类，继承自RuntimeException。

1. KWIC

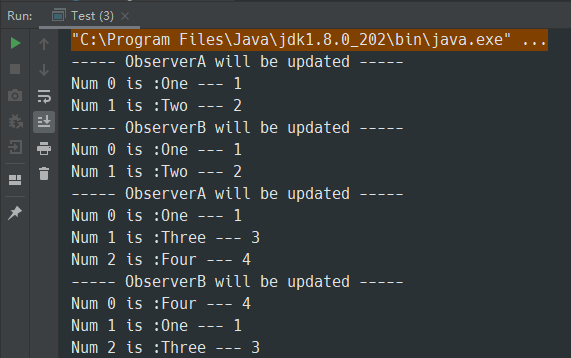
KWIC的主要模块如下：两个行存储模块，第一个行存储模块负责存储所有原先的行，第二个行存储模块负责存储所以循环移位后的模块。 输入模块负责设定路径并且从输入文件中读入并且存储到第一个行存储模块中。 循环移位模块负责循环移位并且存储在第二个行存储模块。输出模块负责输出到文件，主控模块负责主控流程。添加一行到第一个行存储模块就触发了一个事件向循环移位模块发送，循环移位模块会根据这个事件对这一行就行处理，存储到第二个行存储模块，这就会另一个事件发送到排序模块，模块会把新加进来的行与旧行进行混合然后排序。实现的是Observable/Observers模式，其中的事件互动有两部分：移位循环模块和第一个行存储模块注册，移位模块是第一个行存储模块的观察者；排序模块注册和第二个行存储模块注册。排序模块是第二个行存储模块的观察者。KWIC的实现除了源代码提供的ES风格，还有MS（主/子程序体系结构）风格、OO（面向对象体系结构）风格、PF（管道过滤体系结构）风格。

【实验结果】

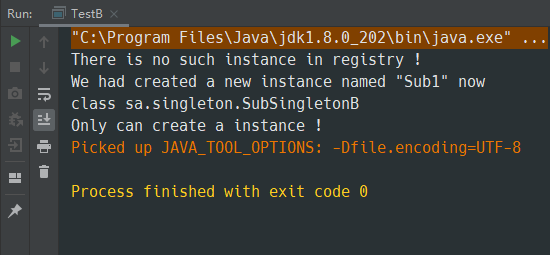
【工厂模式】



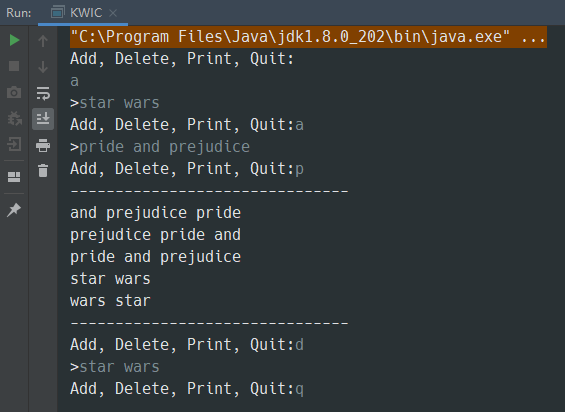
【观察者模式】



【单例模式】



【KWIC】



【小结与体会】

通过阅读基于事件过程调用体系结构的KWIC检索系统，对软件体系结构有了深入的理解。单例模式、工厂模式都在选修课软件构造进行了学习，通过实验代码进一步加深了理解，观察者模式也通过第一次MVC实验进行了学习和实践，有了一定的理解。