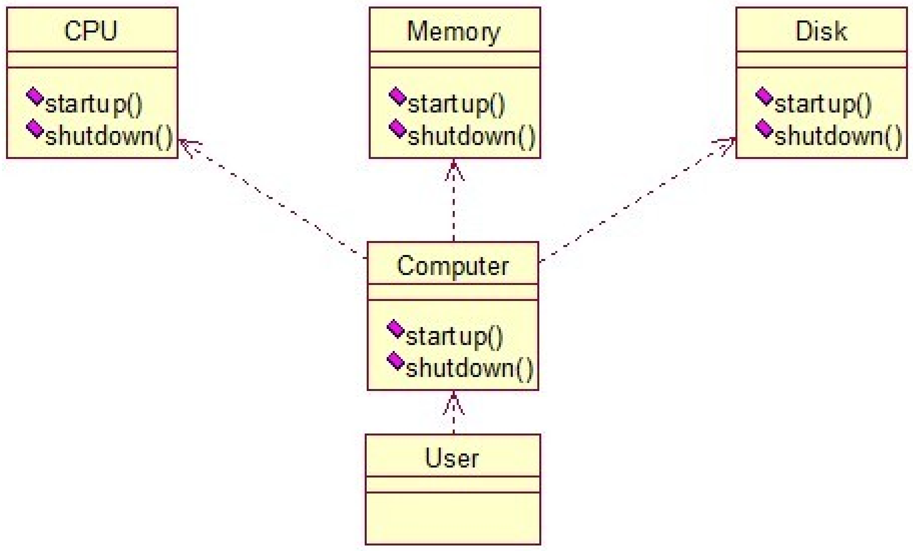
外观模式（Facade）

外观模式是为了解决类与类之家的依赖关系的，像spring一样，可以将类和类之间的关系配置到配置文件中，而外观模式就是将他们的关系放在一个Facade类中，降低了类类之间的耦合度，该模式中没有涉及到接口，看下类图：（我们以一个计算机的启动过程为例）



我们先看下实现类：

[java] view plaincopy

```java
public class CPU {
     
    public void startup(){
        System.out.println("cpu startup!");
    }
     
    public void shutdown(){
        System.out.println("cpu shutdown!");
    }
}
```

[java] view plaincopy

```java
public class Memory {
     
    public void startup(){
        System.out.println("memory startup!");
    }
     
    public void shutdown(){
        System.out.println("memory shutdown!");
    }
}
```

[java] view plaincopy

```java
public class Disk {
    
    public void startup(){
        System.out.println("disk startup!");
    }
    
    public void shutdown(){
        System.out.println("disk shutdown!");
    }
}
```

[java] view plaincopy

```java
public class Computer {
    private CPU cpu;
    private Memory memory;
    private Disk disk;

    public Computer(){
        cpu = new CPU();
        memory = new Memory();
        disk = new Disk();
     }

     public void startup(){
         System.out.println("start the computer!");
         cpu.startup();
         memory.startup();
         disk.startup();
         System.out.println("start computer finished!");
     }

     public void shutdown(){
         System.out.println("begin to close the computer!");
         cpu.shutdown();
         memory.shutdown();
         disk.shutdown();
         System.out.println("computer closed!");
     }
}
```

User类如下:

[java] view plaincopy

```java
public class User {

    public static void main(String[] args) {
        Computer computer = new Computer();
        computer.startup();
        computer.shutdown();
    }
}
```

输出：

start the computer!

cpu startup!

memory startup!

disk startup!

start computer finished!

begin to close the computer!

cpu shutdown!

memory shutdown!

disk shutdown!

computer closed!

如果我们没有Computer类，那么，CPU、Memory、Disk他们之间将会相互持有实例，产生关系，这样会造成严重的依赖，修改一个类，可能会带来其他类的修改，这不是我们想要看到的，有了Computer类，他们之间的关系被放在了Computer类里，这样就起到了解耦的作用，这，就是外观模式！