软件体系结构 作业17

22920212204392 黄勖

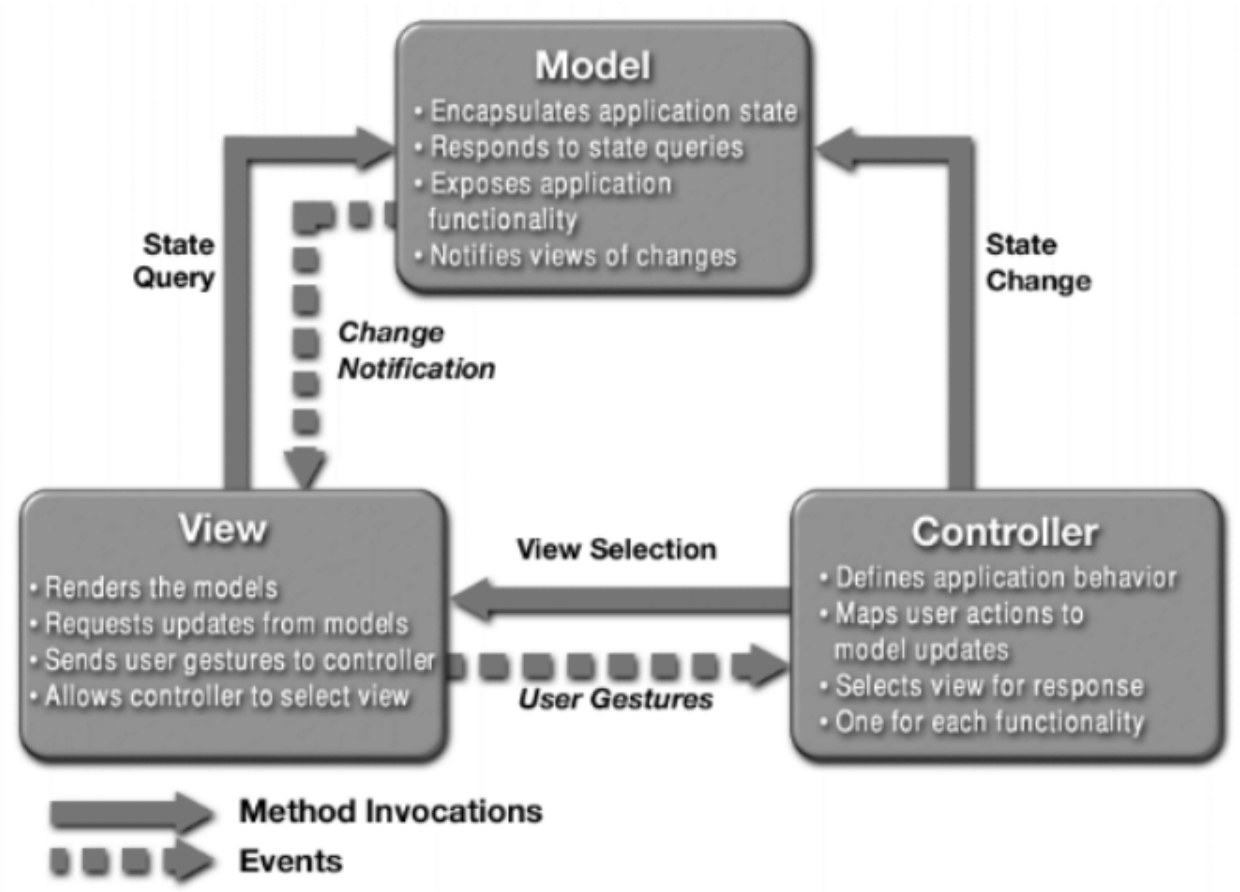
1 阅读：Java SE Application Design With MVC

模型：表示数据以及控制对此数据的访问和更新的规则。在企业软件中，模型通常用作真实世界过程的软件近似值。

视图：呈现模型的内容。它准确指定应如何呈现模型数据。如果模型数据发生更改，视图必须根据需要更新其表示形式。这可以通过使用推送模型或拉取模型来实现。

控制器：将用户与视图的交互转换为模型将执行的操作。在独立的GUI客户端中，用户交互可以是按钮单击或菜单选择，而在企业Web应用程序中，它们显示为HTTP请求。根据上下文，控制器还可以选择新视图以呈现给用户。

常见的MVC实现如下所示：



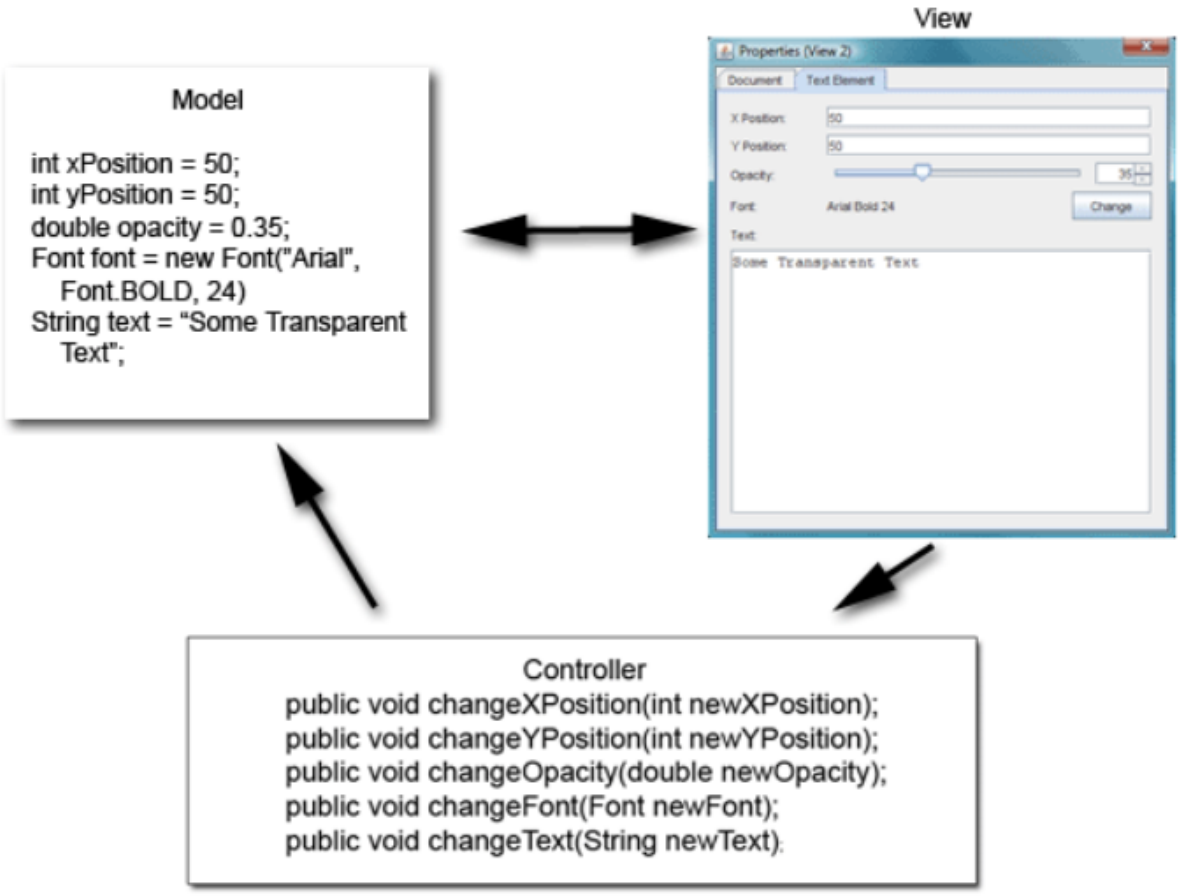
实例化模型、视图和控制器后发生的情况如下：

- 视图注册为模型上的侦听器。对模型基础数据的任何更改都会立即导致广播更改通知，视图会收到该通知。这是前面描述的推送模型的一个示例。但是模型不知道视图或控制器，它只是向所有感兴趣的侦听器广播更改通知。

- 控制器绑定到视图。这通常意味着对视图执行的任何用户操作都将调用控制器类中的已注册侦听器方法。
- 控制器被赋予对基础模型的引用。

用户和视图交互后的操作如下：

- 视图在控制器上调用相应的方法。
- 控制器访问模型，可能以适合用户操作的方式更新模型。
- 如果模型已更改，它会将更改通知感兴趣的侦听器，例如视图。在某些体系结构中，控制器可能还负责更新视图。这在基于Java技术的企业应用程序中很常见。



2　LoD原则强调“只和朋友通信，不和陌生人说话”。请举例说明“朋友圈”认定依据是啥？

迪米特法则(Law of Demeter,LoD)又叫做最少知识原则(LKP)，只与你的直接朋友交谈，不跟“陌生人”说话。其含义是：如果两个软件实体无需直接通信，那么就不应当发生直接的相互调用，可以通过第三方转发该调用。其目的是降低类之间的耦合度，提高模块的相对独立性。

认定依据：“朋友圈”包括对象本身，该对象方法能够创建的对象，该对象方法中以传参形式进入的对象，对象内本身存在的对象（及其引用的对象），同当前对象存在关联、聚合或组合关系的对象，可以直接访问这个对象的对象。

举例：

Apple类：

```
public class Apple {
    private String type;

    public Apple(String type) {
        this.type=type;
    }

    public void printName() {
        System.out.println("苹果的品牌是："+this.type);
    }
}
```

Pear类：

```
public class Pear {
    private String type;

    public Pear(String type) {
        this.type=type;
    }

    public void printName() {
        System.out.println("梨子的品牌是："+this.type);
    }
}
```

Example类：

```
public class Example {
    private static Apple apple;

    public static void main(String[] args) {
        apple=new Apple("红富士");
        apple.printName();

        Pear pear=new Pear("京鲜生秋月梨");
        pear.printName();
    }
}
```

在这个例子中，Apple是Example的成员变量，是Example的“朋友”；而Pear是出现在方法内部的类，不属于Example的“朋友”。

运行结果：

```
苹果的品牌是：红富士
梨子的品牌是：京鲜生秋月梨
```