**基于注解的控制器：**

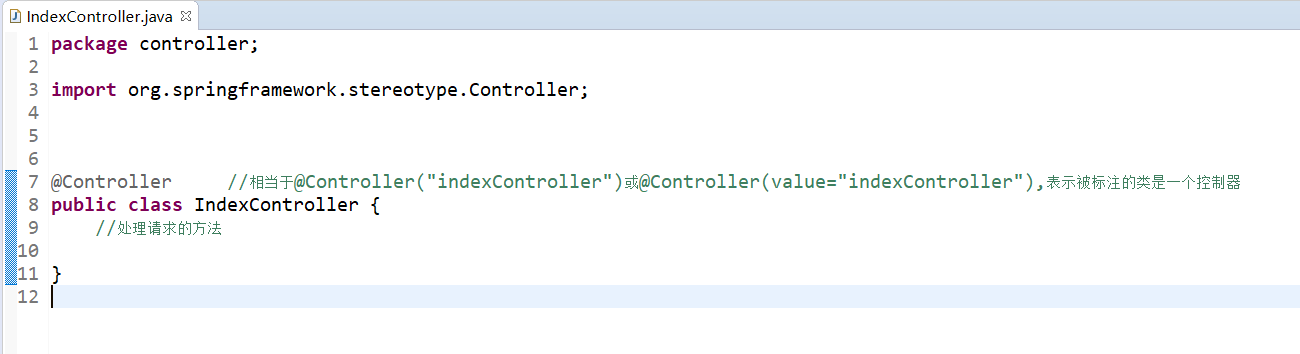
传统控制器（实现Controller的类）不仅需要在Spring MVC配置文件中进行映射，而且 只能编写一个处理方法，不够灵活。为此，在开发中我们使用基于注解的控制器，它具 有以下优点：

1. 在基于注解的控制器类中可以编写多个处理方法，进而可以处理多个请求（动作）， 这就允许了将相关的操作编写在同一个控制类中，从而减少控制器类的数量，方便 维护。
2. 基于注解的控制器类不需要实现Controller接口，不需要在配置文件中进行映射 ，只需要使用ReequestMapping注解类型注解一个方法进行请求处理即可。

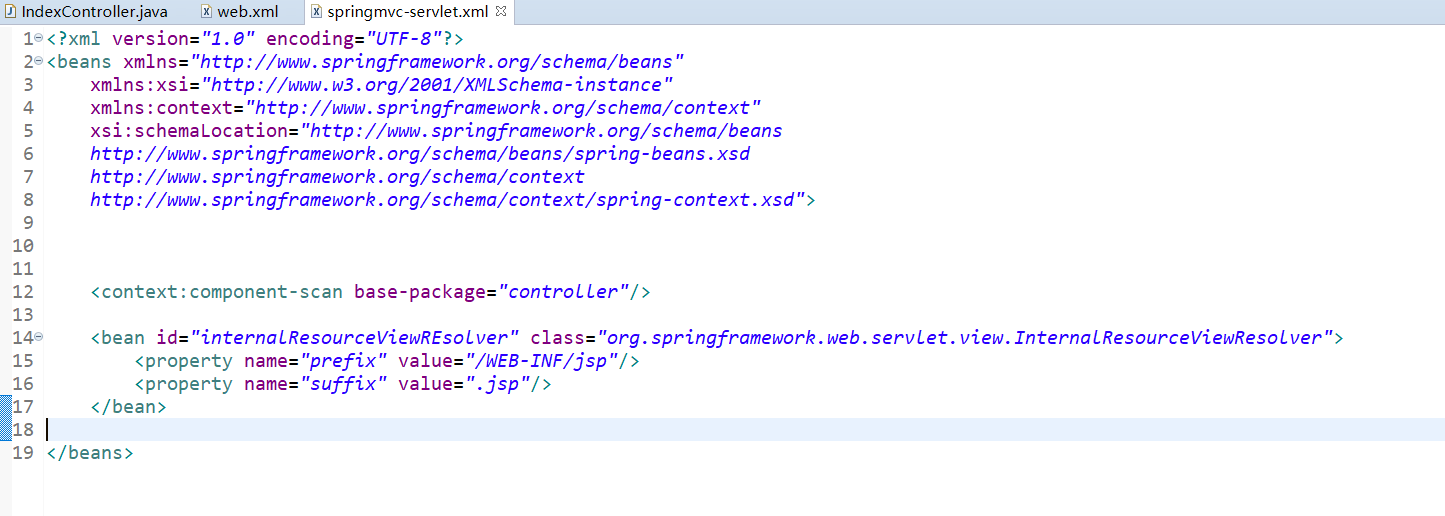
而要想实现基于注解的控制器，则需要使用到Controller和RequestMapping这两个类 型的注解。

**Controller注解类型：**

在Spring MVC中使用org.springframework.stereotype.Controller注解类型来声明一个类 ，表示该类是一个控制器。



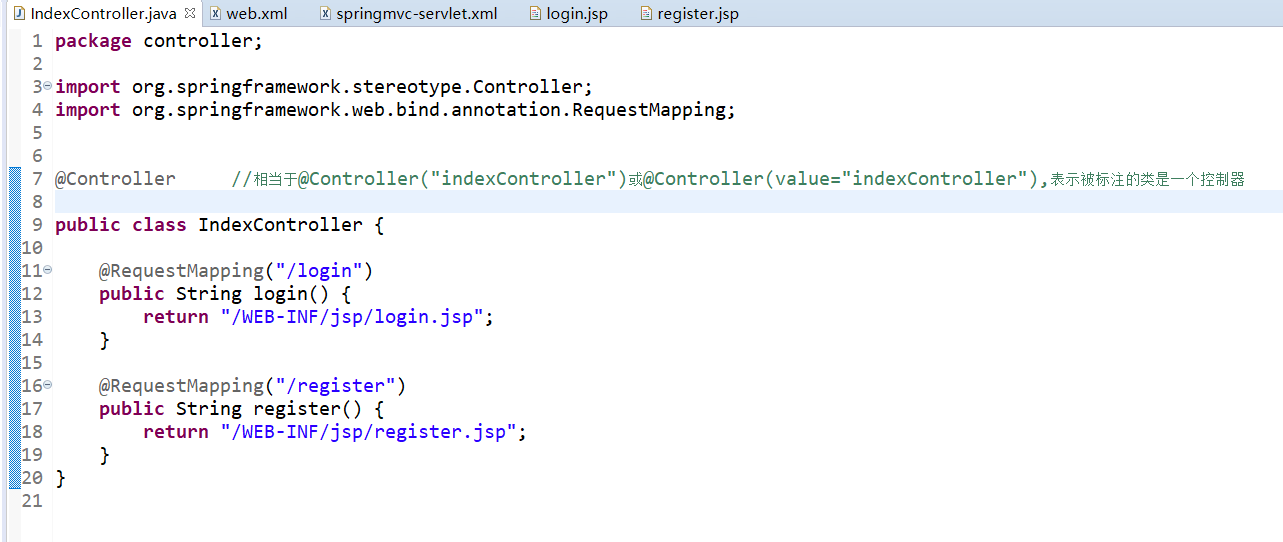
但为了让注解生效，还需要在Spring MVC的配置文件中扫描控制器所在的包，



**RequestMapping注解类型：**

在基于注解的控制类中可以为每一个请求编写对应的处理方法。此时就必须使用 RequestMapping类型的注解来实现。

1. 当RequestMapping标注在方法上



在上面的控制类中有两个请求处理方法中，

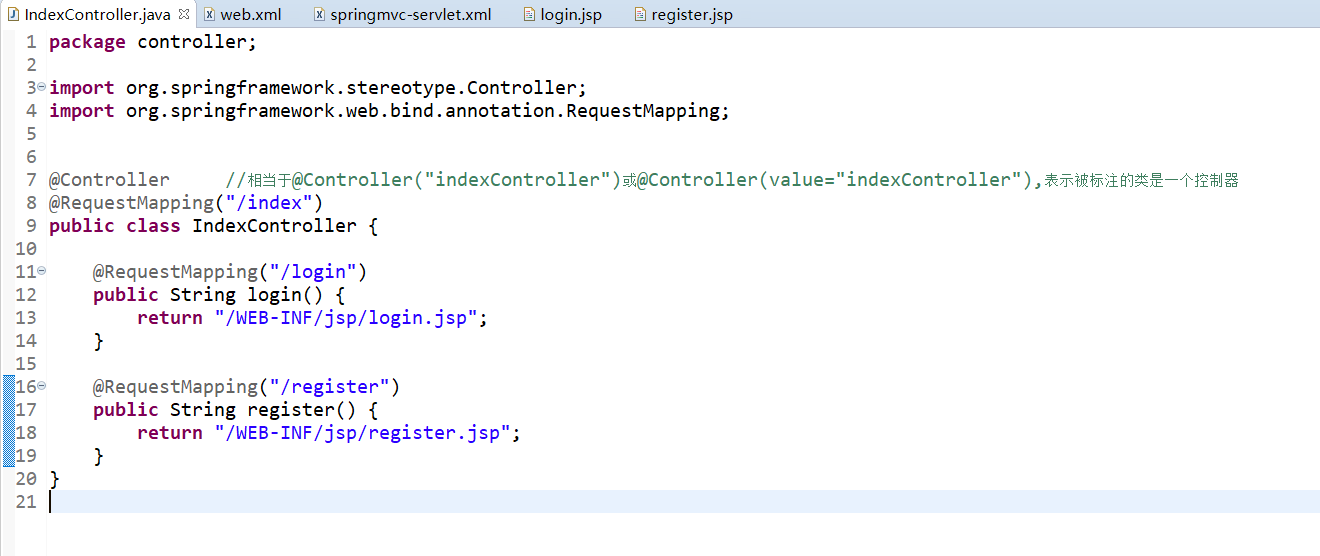
·RequestMapping注解的value属性将请求URI映射到方法

·value属性是RequestMapping注解的默认属性，如果只有一个value属性，则可 忽略。

·用户可以使用URL访问login方法，如

<http://localhost/ch10/index/login>

1. 当RequestMapping标注在方法上



将RequestMapping标注在类上后，访问该控制器类中的处理方法时都必须在其前面加上标注在类的路径。如上，如果要访问以上两个处理方法，则为：

http://localhost/ch10/index/login

<http://localhost/ch10/index/resister>

在开发中采用类级别的注解，可以将相关处理放在同一控制器类中，方便维护。

**编写请求处理方法：**

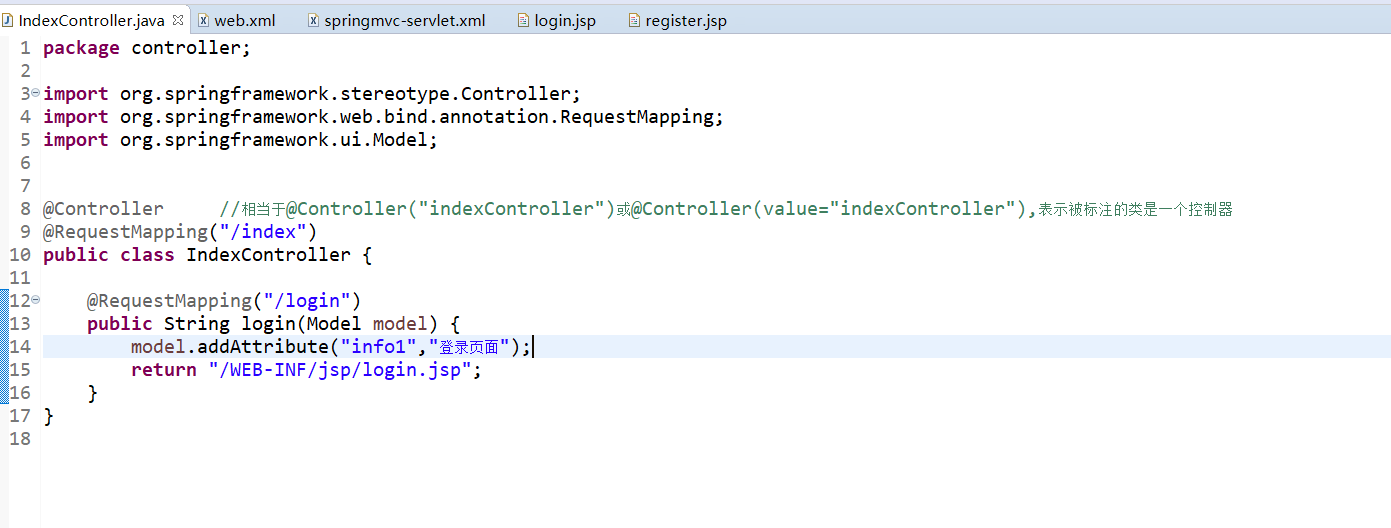
在控制器类中每一个请求处理方法可以由一个或多个任意类型的参数，以及一个任意类 型的返回值。这些都可以由自己进行定义。

1. 请求处理方法中常见的参数类型

（1）包含Servlet API类型参数的请求处理方法



（2）包含Model类型的请求处理方法



model.addAttribute(“info”,”信息”)等同于 request.setAttribute(“info”,” 信息”) ;

1. 请求处理方法常见的返回值类型

对于请求处理方法而言，最常见的返回值类型是代表逻辑视图名称的字符串。

**Controller接收请求参数的常见方式：**

在Controller中请求处理方法接收请求参数的方式有很多种，有的适合get请求方式， 有的适合post请求方式，有的两者都适合。大致一共有6种：

1. 通过实体Bean接收请求参数
2. 通过同名参数接收请求参数
3. 通过HttpServletRequest接收请求参数
4. 通过@PathVariable接收URL中的请求参数
5. 通过@RequestParam接收请求参数
6. 通过@ModelAttribute接收请求参数

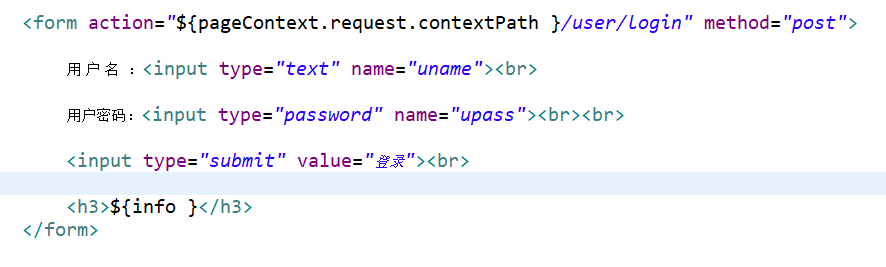
开发者应根据实际情况来选择适合的方式。

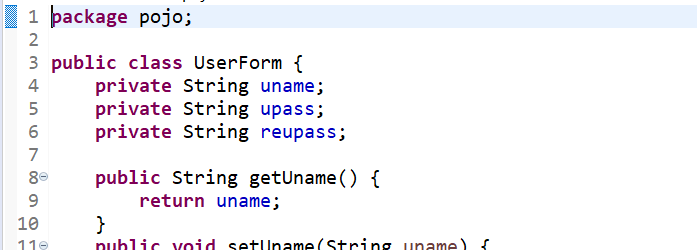
**通过实体Bean接收请求参数：**

通过是实体Bean接收请求参数，适用于get和post提交请求方式。

需要注意的是，Bean的属性名必须与请求参数名称相同。

如下：



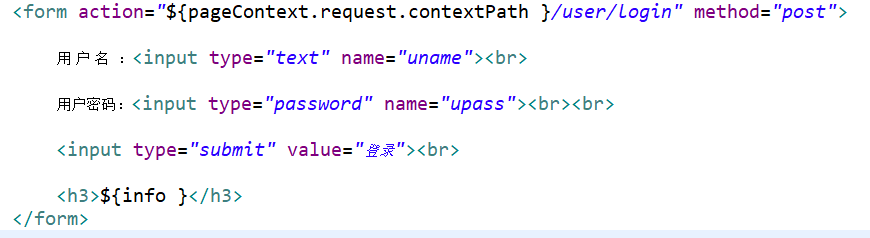




**通过同名参数接收请求参数：**

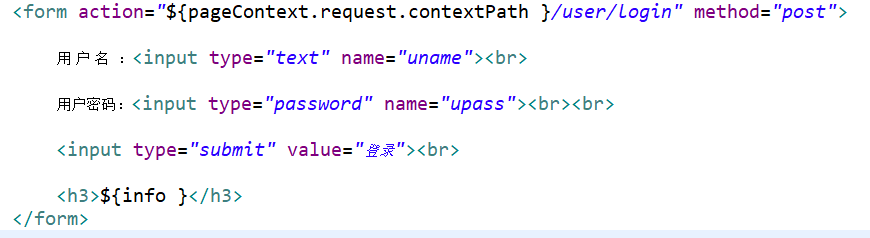
通过同名参数接收请求参数指的是直接把请求参数名作为控制器中请求处理方法的形 参，即形参名称与请求参数名称一致。适用于get和post提交方式。

如下：





**通过HttpServletRequest接收请求参数：**  
 在没有学框架之前，都是使用HttpServletRequest（request对象）的一系列方法， 来接收请求参数。适用于get和post提交方式。





**重定向与转发：**

重定向指的是客户端跳转，属于客户端行为；转发指的是服务端跳转，属于服务端 行为。

1. 实现客户端跳转：

实现客户端跳转只需要在发出请求后，经过请求处理方法，由请求处理方 法返回一个带有逻辑视图的String字符串并加上前缀“redirect：”表示客户端 跳转。



1. 实现服务端跳转：

实现客户端跳转只需要在发出请求后，经过请求处理方法，由请求处理方 法返回一个带有逻辑视图的String字符串并加上“forward：”表示服务端跳转。



不论是重定向（客户端跳转）还是转发（服务端跳转），都必须准守视图解析器的规则， 并且，它们跳转的路径必须是已存在的请求处理方法路径（DispatcherServlet路径），而 不能是某个具体的jsp文件。

如果需要转发到不需要DispatcherServlet的资源，例如

Return “ forward : /html/my.html ” ;

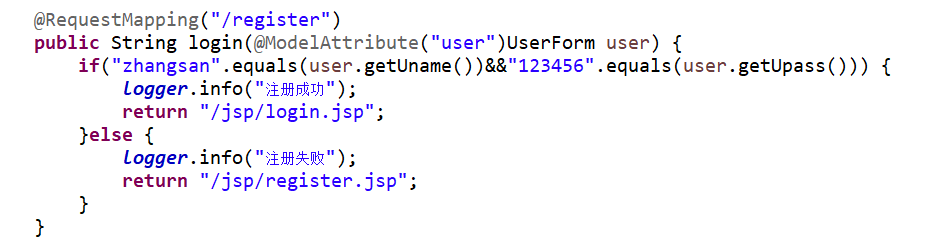
则需要在mvc.resources配置：

<mvc:resources location=”/html/” mapping=”/html/\*\*”></mvc:resources>

**@ModelAttribute注解：**

通过org.springframework.web.bind.annotation.ModelAttribute注解类型根据其标注的位置，从而有两个常用的功能。

1. 接收请求参数并绑定到Bean对象（标注在方法形参上）



在以上请求处理方法中，“@ModelAttribute(“user”) UserForm user”有两个含义：

·一是使用user对象通过实体Bean的方式来接收请求参数，实体Bean的属 性需要与请求参数名一致。

·二是当user对象接收请求参数后，将user对象存储在Model对象中，键值 名为”user”，和“model.addAttribute(“user”，user)”语句功能一样；当然你 也可以不指定其键值名，那么将会以userForm为键值名，与 “model.addAttribute(“userForm”,user)”语句相同。

1. 注解非控制器中的非请求处理方法（标注在方法上）

当@ModelAttribute标注在控制器类中的非请求处理方法上，那么在调用这个 控制器类的处理请求方法时，会先调用标注了@ModelAttribute的方法。利用 这种特性可以用来控制登录权限。



当然，控制登录权限的方法有很多，如过滤器、拦截器等。