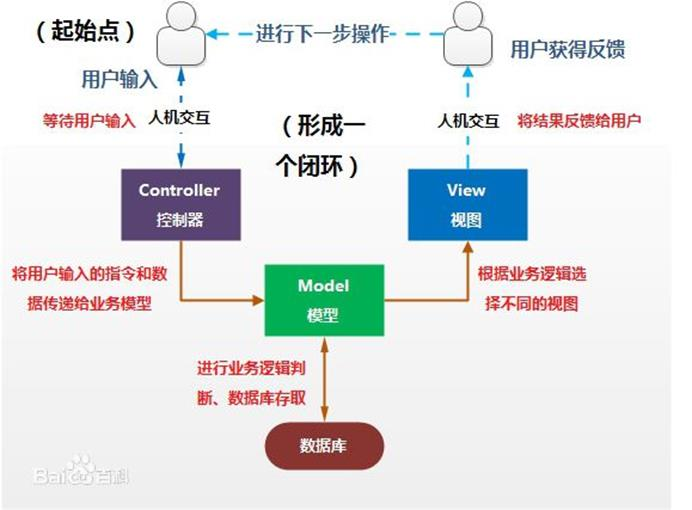
**今天完成的事情：**

首先，今天学习了RESTful风格的API设计，然后又学习了HTTP请求的动作和过程，最后学习了MVC的概念，以下为MVC模型中M（model）V（view）C（controller）之间关系：



1. **执行流程**
2. 用户首先在界面中进行人机交互
3. 然后请求发送到控制器，控制器根据请求类型和请求的指令发送到相应的模型
4. 模型可以与数据库进行交互，进行增删改查操作
5. 根据业务的逻辑选择相应的视图进行显示
6. **MVC模型举例一**

**最典型的MVC模型就是jsp+servlet+javabean模式**

1. JavaBean

模型层，可以作为数据模型来封装业务数据，也可以作为业务逻辑模型来包含应用操作。其中数据模型用来存储或传递业务数据；而业务逻辑模型接收到控制器传过来的模型更新请求后，执行特定的业务逻辑处理，然后返回相应的结果。

1. JSP

表现层，负责提供页面为用户展示数据，提供相应的表单（Form）来用于用户的请求，并在适当的时候（点击按钮）向控制器发出请求来请求模型进行更新。

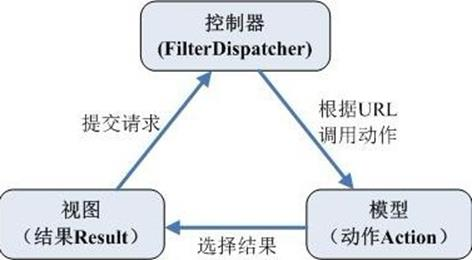
1. Servlet

控制层，用来接收用户提交的请求，然后获取请求中的数据，将之转换为业务模型需要的数据模型，然后调用业务模型相应的业务方法进行更新，同时根据业务执行结果来选择要返回的视图。

1. **MVC模型举例二**

**目前比较流行的模型是Struts2框架，Struts2是基于MVC的轻量级的web应用框架**

框架模型流程图为：



1. FilterDispatcher

控制器，负责根据用户提交的url和struts.xml中的配置，来选择合适的动作（action），让action来处理用户的请求。filterdipatcher同时也是过滤器（Filter，servlet规范中的一种web组件），只需要在项目的web.xml中配置一下即可。

1. Action

动作，负责把用户请求中的参数组装成合适的数据模型，并调用相应的业务逻辑进行真正的功能处理，获取下一个视图展示所需要的数据。

Struts2的Action，它实现了与Servlet API的解耦，使得Action里面不需要再直接去引用和使用HttpServletRequest与HttpServletResponse等接口。

1. Result

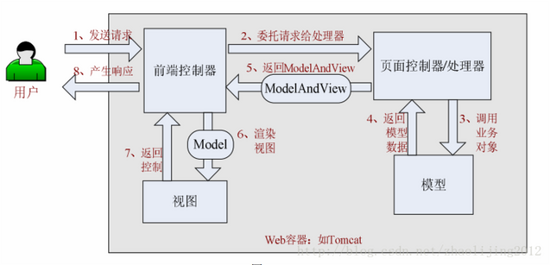
视图结果，负责把动作中获取的数据展现给用户。

在Struts2中有多种结果展示方式：常规jsp、模版freemarker、velocity、图表jfreechart、报表JasperReport、将xml转化成html的XSTL

1. **SpringMVC的请求过程**

由于需要搭建基本的MVC模型来实现页面的传参接参，所以直接选择SpringMVC框架来做入门理解，原因是之前搭建过Spring框架，对其的运作原理有一定理解，还一个原因是SpringMVC框架是成熟、简洁的框架模型，易于理解，易于构成。

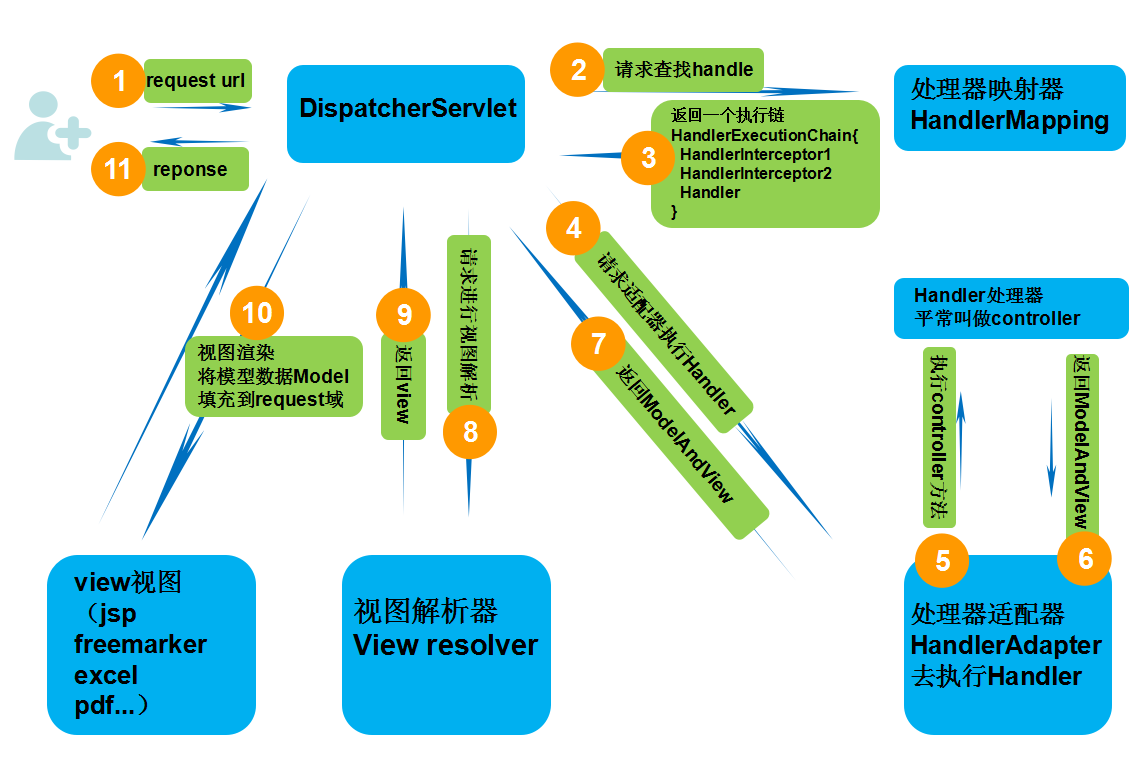
1. **SpringMVC的整体运行流程如图所示**：



**具体步骤为**：

* 首先用户发送请求到前端控制器，前端控制器根据请求信息（如 URL）来决定选择哪一个页面控制器进行处理并把请求委托给它，即以前的控制器的控制逻辑部分。图中的 1、2 步骤。
* 页面控制器接收到请求后，进行功能处理。首先需要收集和绑定请求参数到一个对象（这个对象在Spring Web MVC中叫命令对象），并进行验证，然后将命令对象委托给业务对象进行处理。处理完毕后返回一个 ModelAndView（模型数据和逻辑视图名）。图中的 3、4、5 步骤。
* 前端控制器收回控制权，然后根据返回的逻辑视图名，选择相应的视图进行渲染，并把模型数据传入以便视图渲染。图中的步骤 6、7。
* 前端控制器再次收回控制权，将响应返回给用户。图中的步骤 8。

1. **SpringMVC的核心流程如图所示**：



**具体步骤为**：

* 发起请求URI到前端控制器(DispatcherServlet)
* 前端控制器请求HandlerMapping查找 Handler映射器（能根据xml配置、注解进行查找）
* Handler映射器向前端控制器返回HandlerExecutionChain处理执行链（包含一个controller（页面控制器），多个HandlerInterceptor拦截器），通过这种策略模式，很容易添加新的映射策略
* 前端控制器调用处理器适配器去执行Handler
* 处理器适配器HandlerAdapter将会根据适配的结果去执行Handler
* Handler执行完成给适配器返回ModelAndView
* 处理器适配器向前端控制器返回ModelAndView（ModelAndView是springmvc框架的一个底层对象，包括 Model和view）
* 前端控制器请求视图解析器去进行视图解析（根据逻辑视图名解析成真正的视图(jsp)），通过这种策略很容易更换其他视图技术，只需要更改视图解析器即可
* 视图解析器向前端控制器返回View
* 前端控制器进行视图渲染（视图渲染将模型数据(在ModelAndView对象中)填充到request域）
* 前端控制器向用户响应结果

**明天计划完成的事情：**

完成SpringMVC表面层构建，完成Hello World的功能

**遇到的问题：**

学习完web基本概念后，直接用SpringMVC来搭建MVC基础，还是用别的构件来搭建MVC基础，不知道你们开始在这里是怎么开始搭建的

**收获：**

学习了RESTful的API接口风格设计，学习了MVC的请求流程，学习了HTTP的请求动作，初步了解SpringMVC的结构