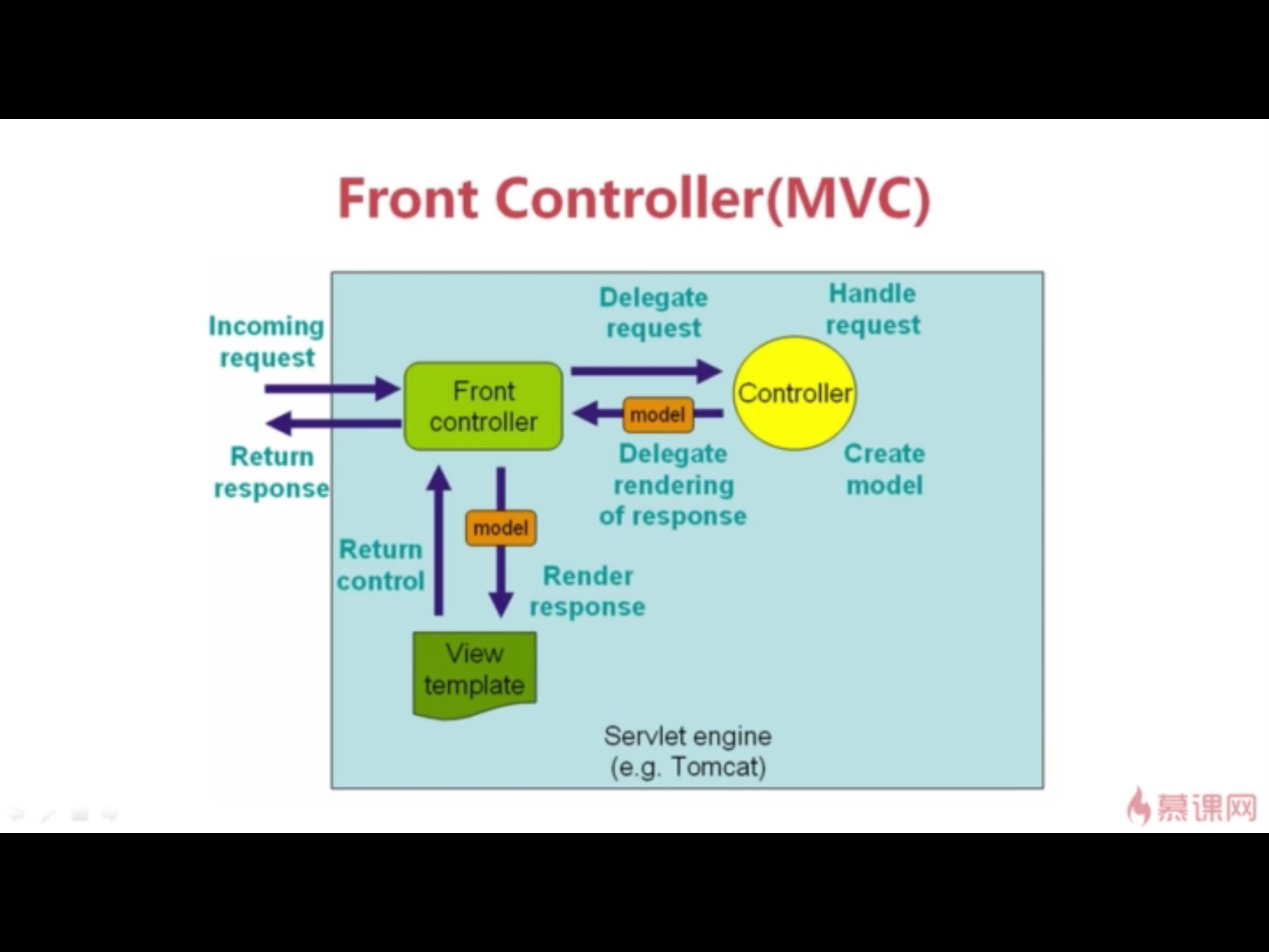
# Spring MVC简介

MVC是开发WEB应用程序的通用架构方式。

前端控制器：



Mvc的实现方式：用户通过http协议到达前端控制器，前端控制器（由谁处理请求）、控制器（了解业务逻辑细节）调用业务逻辑生成业务数据并返给前端控制器，前端控制器将业务数据分发给业务视图。由业务视图呈现最终的用户页面，将用户页面返回给前端控制器，前端控制器将用户页面返回给浏览器。

前端控制器：分发调度

控制器：业务数据抽取

试图模板：页面呈现

MVC的核心思想是业务数据抽取同业务数据呈现相分离。

# 什么是MVC？

MVC：Model、View、Controller

视图层（View）：为用户提供UI，重点关注数据的呈现。

模型层（Model）：业务数据的信息表示，关注支撑业务的信息构成，通常是多个业务实体的组合。

控制层（Controller）：调用业务逻辑产生合适的数据（Model）传递数据给视图层用于呈现。

MVC是一种架构模式：程序分层、分工合作，既相互独立，有协同工作。

MVC是一种思考方式：需要将什么信息展示给用户？如何布局？调用哪些业务逻辑？

# Spring MVC基本概念

DispatcherServlet

Controller

HandlerAdapter

HandlerInterceptor

HandlerMapping：

1.Help DispatcherServlet to get the right controller.

2.Wrap controller woth HandlerInterceptor.

HandlerExecutionChain

ModelAndView

ViewResolver:Help DispatcherServlet to Resolve the right View to render page.

View

# Maven

自动创建Web应用的目录结构命令：

mvn archetype:generate -DgroupId=imooc-arthur -DartifactId=spring-mvc-study -DarchetypeArtifactId=maven-archetype-webapp

# Spring配置

## context:component-scan标签

Eg.

<!-- 使Spring支持自动检测组件,只搜索@Controller 标注的类 -->

<context:component-scan base-package="com.imooc.mvcdemo">

<context:include-filter type="annotation" expression="org.springframework.stereotype.COntroller"/>

</context:component-scan>

下面是引用spring framework开发手册中的一段话:“Spring 2.5引入了更多典型化注解(stereotype annotations)： @Component、@Service和 @Controller。 @Component是所有受Spring管理组件的通用形式；而@Repository、@Service和 @Controller则是@Component的细化，用来表示更具体的用例(例如，分别对应了持久化层、服务层和表现层)。也就是说，你能用@Component来注解你的组件类，但如果用@Repository、@Service 或@Controller来注解它们，你的类也许能更好地被工具处理，或与切面进行关联。例如，这些典型化注解可以成为理想的切入点目标。当然，在Spring Framework以后的版本中， @Repository、@Service和 @Controller也许还能携带更多语义。如此一来，如果你正在考虑服务层中是该用 @Component还是@Service，那@Service显然是更好的选择。同样的，就像前面说的那样， @Repository已经能在持久化层中进行异常转换时被作为标记使用了。”

filter標籤在Spring3有五個type，如下：

|  |  |  |
| --- | --- | --- |
| ****Filter Type**** | ****Examples Expression**** | ****Description**** |
| annotation | org.example.SomeAnnotation | 符合SomeAnnoation的target class |
| assignable | org.example.SomeClass | 指定class或interface的全名 |
| aspectj | org.example..\*Service+ | AspectJ語法 |
| regex | org\.example\.Default.\* | Regelar Expression |
| custom | org.example.MyTypeFilter | Spring3新增自訂Type，實作org.springframework.core.type.TypeFilter |