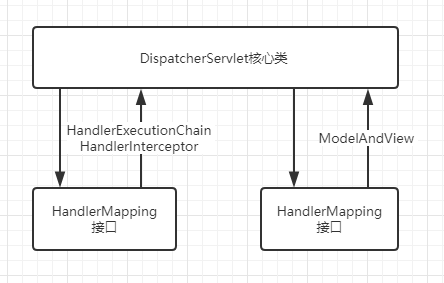
# 完整的web请求核心类和接口

核心接口定义了框架的骨架，也在最抽象的意义上表达了框架的设计思想。



## DispatcherServlet核心类

org.springframework.web.servlet.DispatcherServlet一个标准的servlet,主要作用是接受和转发web请求到内部框架处理单元。

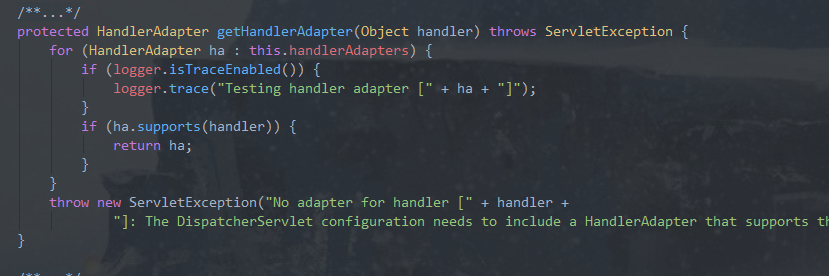
### 1.1 DI → HandlerMapping处理器映射器

1. 当 DispatcherServlet 接收到 web 请求后,由标准 Servlet 类处理方法 doGet 或者 doPost,经过几次转发后,最终注册在 DispatcherServlet 类中的一个 List循环中被遍历,该List是由HandlerMapping 实现类组成的。以该 web 请求的 HttpServletRequest 对象为参数，依次调用其 getHandler 方法，第一个不为 null 的调用结果，将被返回。



1. 一个 web 请求经过处理后，会得到一个 HandlerExecutionChain 对象，这就是 SpringMVC 对 URl 映射给出的回答。
2. HandlerMapping 接口的 getHandler 方法参数是 HttpServletRequest，这意味着，HandlerMapping 的实现类可以利用 HttpServletRequest 中的 所有信息来做出这个 HandlerExecutionChain 对象的生成” 决策 “。这包括，请求头、url 路径、cookie、session、参数等等一切你从一个 web 请求中可以得到的任何东西（最常用的是 url 路径）。

### 1.2 DI → HandlerAdapter适配器



1. HandlerExecutionChain 中的 handler 对象会被作为参数传递进去
2. 遍历在 DispatcherServlet 类中注册的 HandlerAdapter 实现类列表，参数为handler。返回第一个 supports 方法返回 true 的 HandlerAdapter 对象
3. 用这个HandlerAdapter 实现类中的 handle 方法处理 handler 对象，并返回 ModelAndView 这个包含了视图和数据的对象。

### 1.3 DI处理基本流程

1. 在一个典型的 SpringMVC 调用中，HandlerExecutionChain 中封装 handler 对象就是用 @Controller 注解标识的类的一个实例。
2. 根据类级别和方法级别的 @RequestMapping 注解，由默认注册的 DefaultAnnotationHandlerMapping（3.1.3 中更新为 RequestMappingHandlerMapping 类，但是为了向后兼容，DefaultAnnotationHandlerMapping 也可以使用）生成 HandlerExecutionChain 对象
3. 再由 AnnotationMethodHandlerAdapter（3.1.3 中更新为 RequestMappingHandlerAdapter 类，但是为了向后兼容，AnnotationMethodHandlerAdapter 也可以使用）来执行这个 HandlerExecutionChain 对象，
4. 生成最终的 ModelAndView 对象后，再由具体的 View 对象的 render 方法渲染视图。

## HandlerMapping接口

org.springframework.web.servlet.HandlerMapping位于spring-webmvc



### 核心方法

HandlerExecutionChain getHandler(HttpServletRequest request) throws Exception;

1. 请求参数为：HttpServletRequest

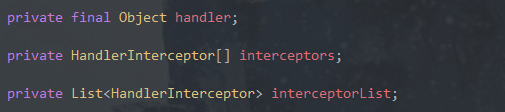
返回参数为：HandlerExecutionChain

不处理任何类型的异常。

## HandlerExecutionChain返回类

org.springframework.web.servlet.HandlerExecutionChain位于spring-webmvc包.

1. HandlerExecutionChain这个类是一个执行链的封装，主要是一个实质的执行对象以及一堆拦截器。

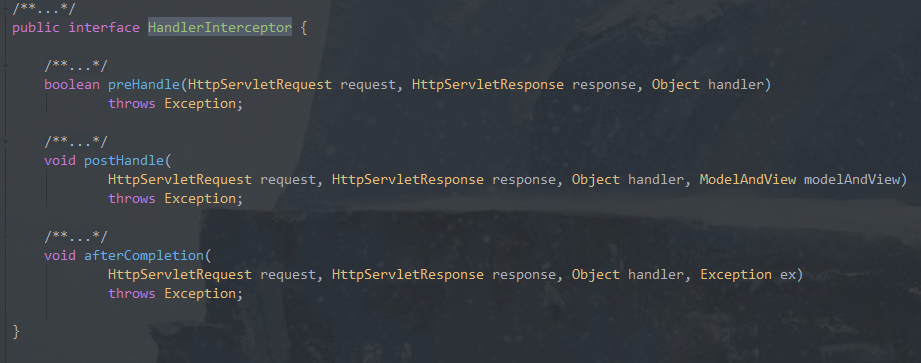


### 执行流程

1. 在真正调用其 handler 对象前，HandlerInterceptor 接口实现类组成的数组将会被遍历，其 preHandle 方法会被依次调用，然后真正的 handler 对象将被调用。
2. handler 对象被调用后，就生成了需要的响应数据，在将处理结果写到 HttpServletResponse 对象之前（SpringMVC 称为渲染视图），其 postHandle 方法会被依次调用。
3. 视图渲染完成后，最后 afterCompletion 方法会被依次调用，整个 web 请求的处理过程就结束了。

## HandlerInterceptor拦截器接口

org.springframework.web.servlet.HandlerInterceptor位于spring-webmvc包。



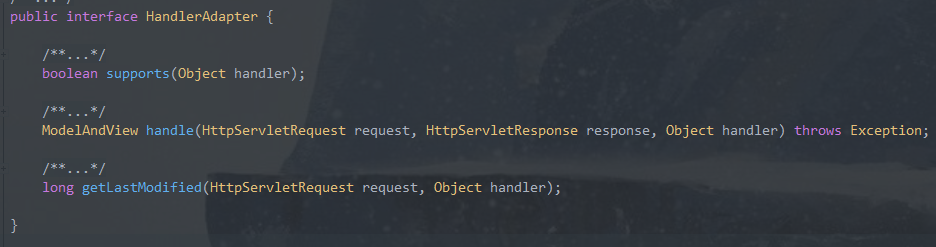
preHandler在**一个请求被真正处理之前**执行，

postHandler在**请求被处理但还没输出到响应中**执行，

afterCompletion在**请求已经被输出到响应中**执行。

## HandlerAdapter适配器接口

org.springframework.web.servlet.HandlerAdapter位于spring-webmvc包



1. support方法用来接收HandlerExecutionChain中的Handler对象，处理后并返回结果，DI会取结果为true的HandlerAdapter。
2. handle方法接收HttpServletRequest/Respose和Handler,并返回一个ModelAndView对象。
3. ModelAndView对象由一个Map数据模型和View对象组成，核心是View对象。

## View接口

org.springframework.web.servlet.View位于spring-webmvc包



1. 所有的数据，最后会作为一个 Map 对象传递到 View 实现类中的 render 方法，调用这个 render 方法，就完成了视图到响应的渲染。
2. 当然从 ModelAndView 到真正的 View 实现类有一个解析的过程，ModelAndView 中可以有真正的视图对象，也可以只是有一个视图的名字，SpringMVC 会负责将视图名称解析为真正的视图对象。