源码阅读

——Spring MVC

汇报人：乔怡

2021.12.31

Spring MVC的核心工作流程（核心组件）

前情提要

核心组件

核心工作流程

类图及类之间的关系

高级设计思想

一

二

三

四

五

前情提要

01 有关Spring MVC

Spring MVC 是 Spring 提供的一个基于 MVC 设计模式的轻量级 Web 开发框架，本质上相当于 Servlet。

Servlet 是用Java编写的服务端程序，可交互式的浏览和生成数据。

02 MVC

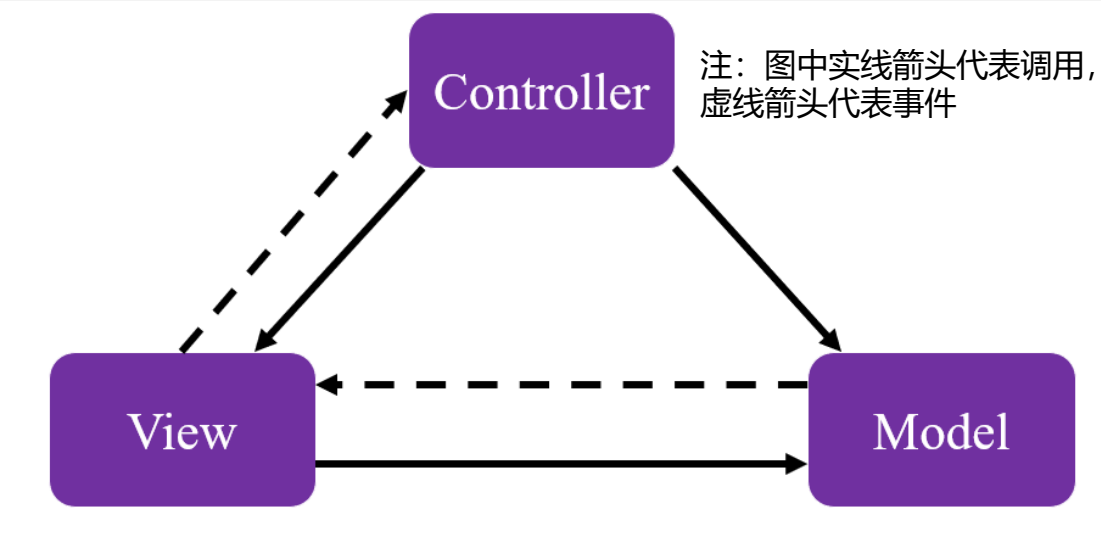
M：Model(模型)——V：View(视图)——C：Controller(控制器)

Model：模型，所有的用户数据、状态以及程序逻辑

View：视图，呈现模型，类似于Web程序中的界面

Controller：控制器，**作用于Model和View上**，负责获取用户的输入信息，进行解析并反馈给模型

02 MVC



Model：模型，所有的用户数据、状态以及程序逻辑

View：视图，呈现模型，类似于Web程序中的界面

Controller：控制器，作用于Model和View上，负责获取用户的输入信息，进行解析并反馈给模型

核心组件

DispatcherServlet：中央控制器或前端控制器，是整个Spring MVC的核心，负责接收HTTP请求组织协调Spring MVC的各个组成部分，把请求给转发到具体的控制类进行响应。

DispatcherServlet：中央控制器或前端控制器，是整个Spring MVC的核心，责接收HTTP请求组织协调Spring MVC的各个组成部分，把请求给转发到具体的控制类进行响应。

Controller：具体处理请求的控制器，处理用户请求。

HandlerMapping：映射处理器，用于查找handler，负责映射中央处理器站发给Controller时的映射策略。

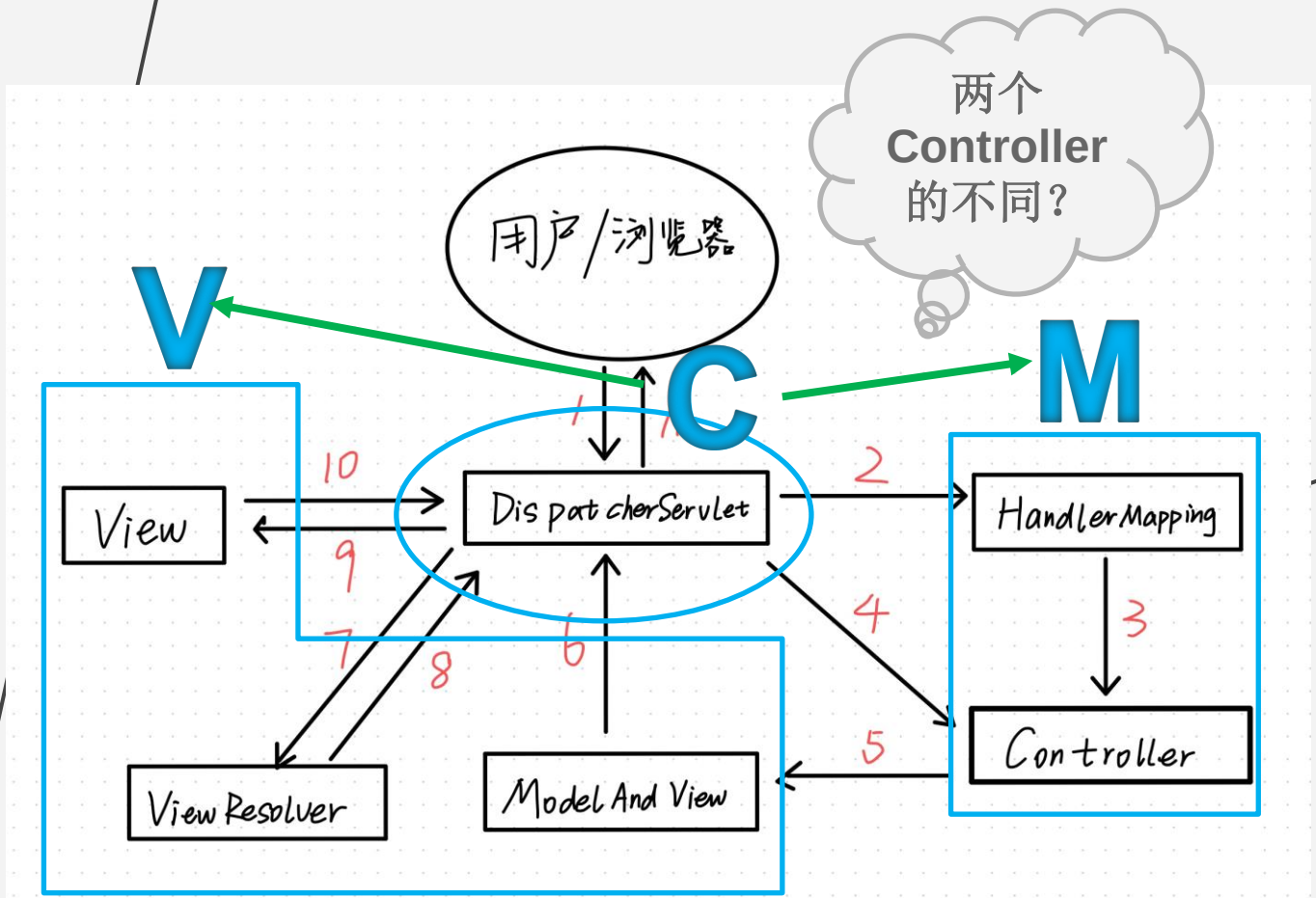
HandlerAdapter：处理器适配器，用让固定的Servlet处理方法调用Handler来进行处理。

ModelAndView：包括模型（Model）和视图（View），服务层返回的数据和视图层的封装类。

ViewResolver：视图解析器，解析具体的视图，从而将相应结果渲染给客户。

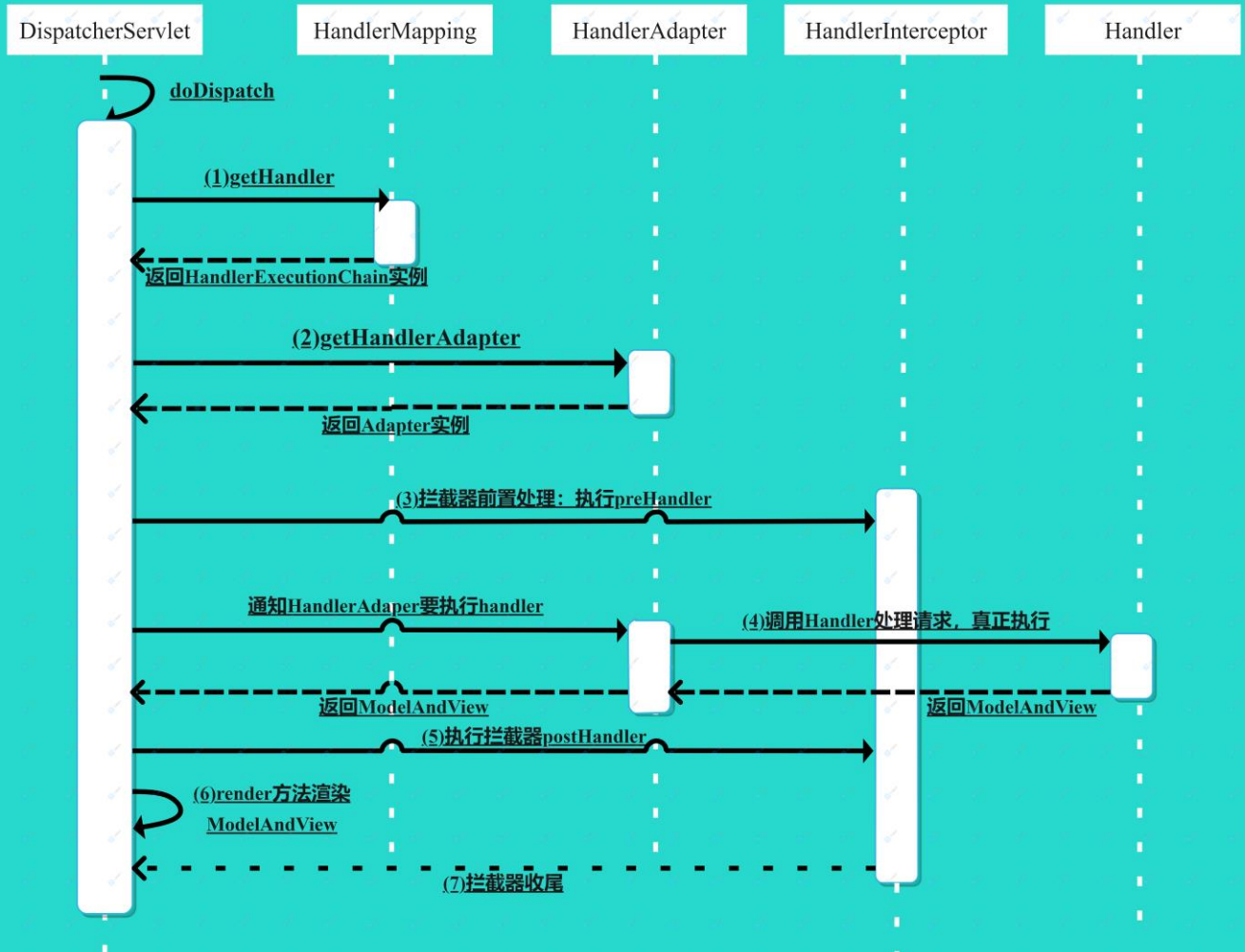
Interceptor：拦截器，负责拦截我们定义的请求然后做处理工作

核心工作流程


两个
Controller
的不同？
用户/浏览器
V
C
M
View
DispatcherServlet
HandlerMapping
Controller
ModelAndView
ViewResolver
1
2
3
4
5
6
7
8
9
10

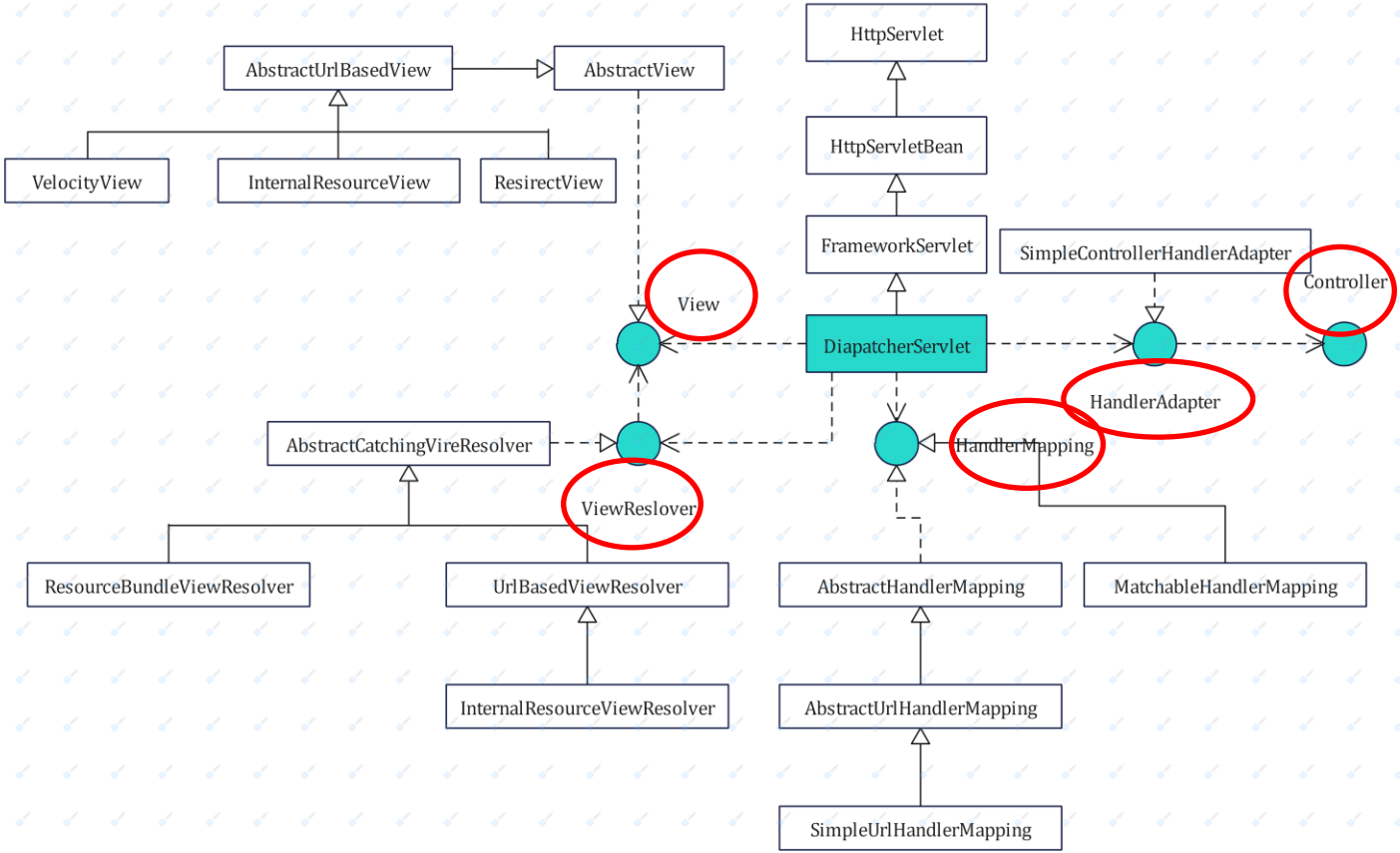
处理请求的核心函数：

核心组件**DispatcherServlet**中的 **doDispatch**函数

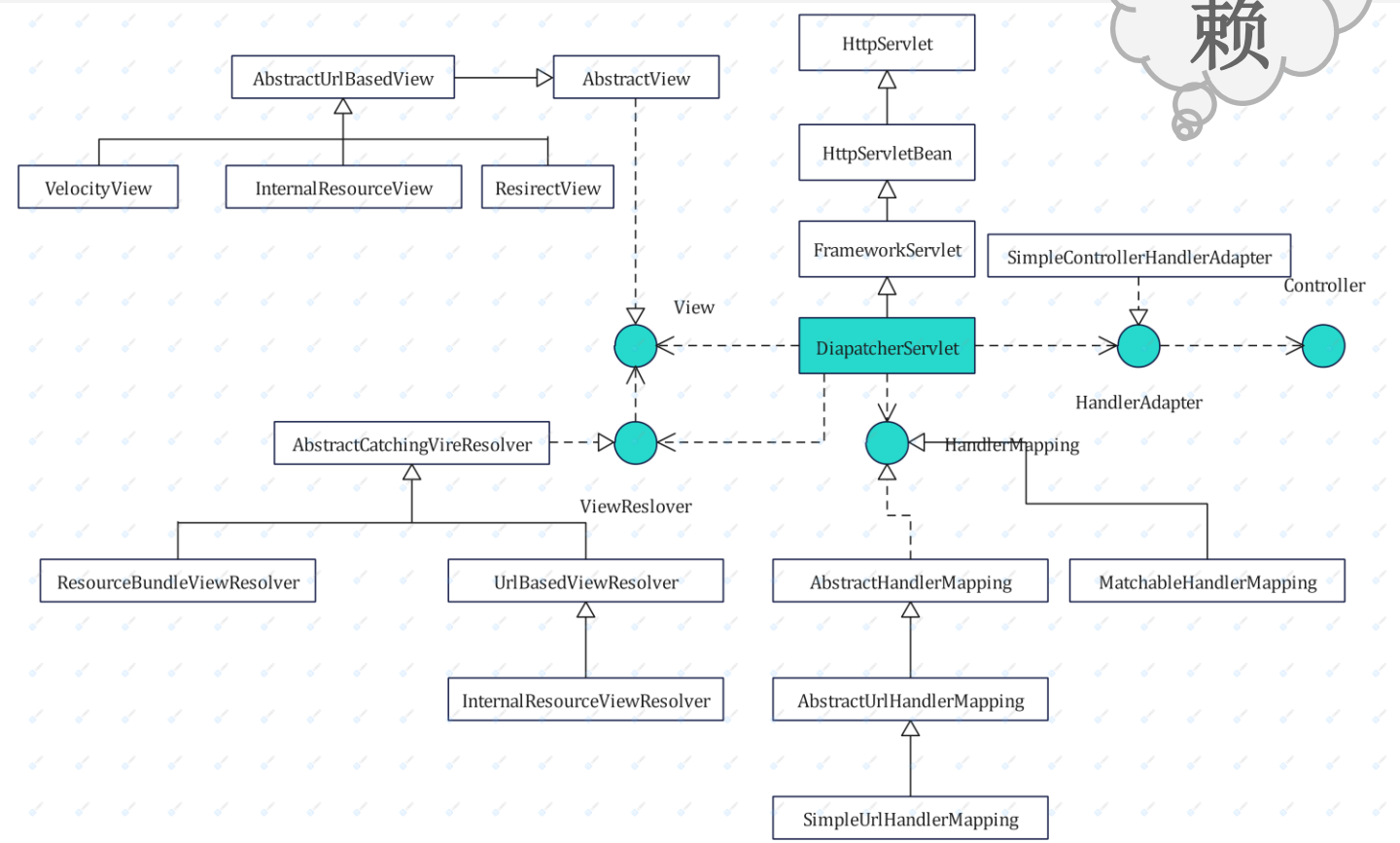

DispatcherServlet
HandlerMapping
HandlerAdapter
HandlerInterceptor
Handler
doDispatch
(1)getHandler
返回HandlerExecutionChain实例
(2)getHandlerAdapter
返回Adapter实例
(3)拦截器前置处理：执行preHandler
通知HandlerAdaper要执行handler
(4)调用Handler处理请求，真正执行
返回ModelAndView
返回ModelAndView
(5)执行拦截器postHandler
(6)render方法渲染
ModelAndView
(7)拦截器收尾

类图&类之间的关系

01 类图

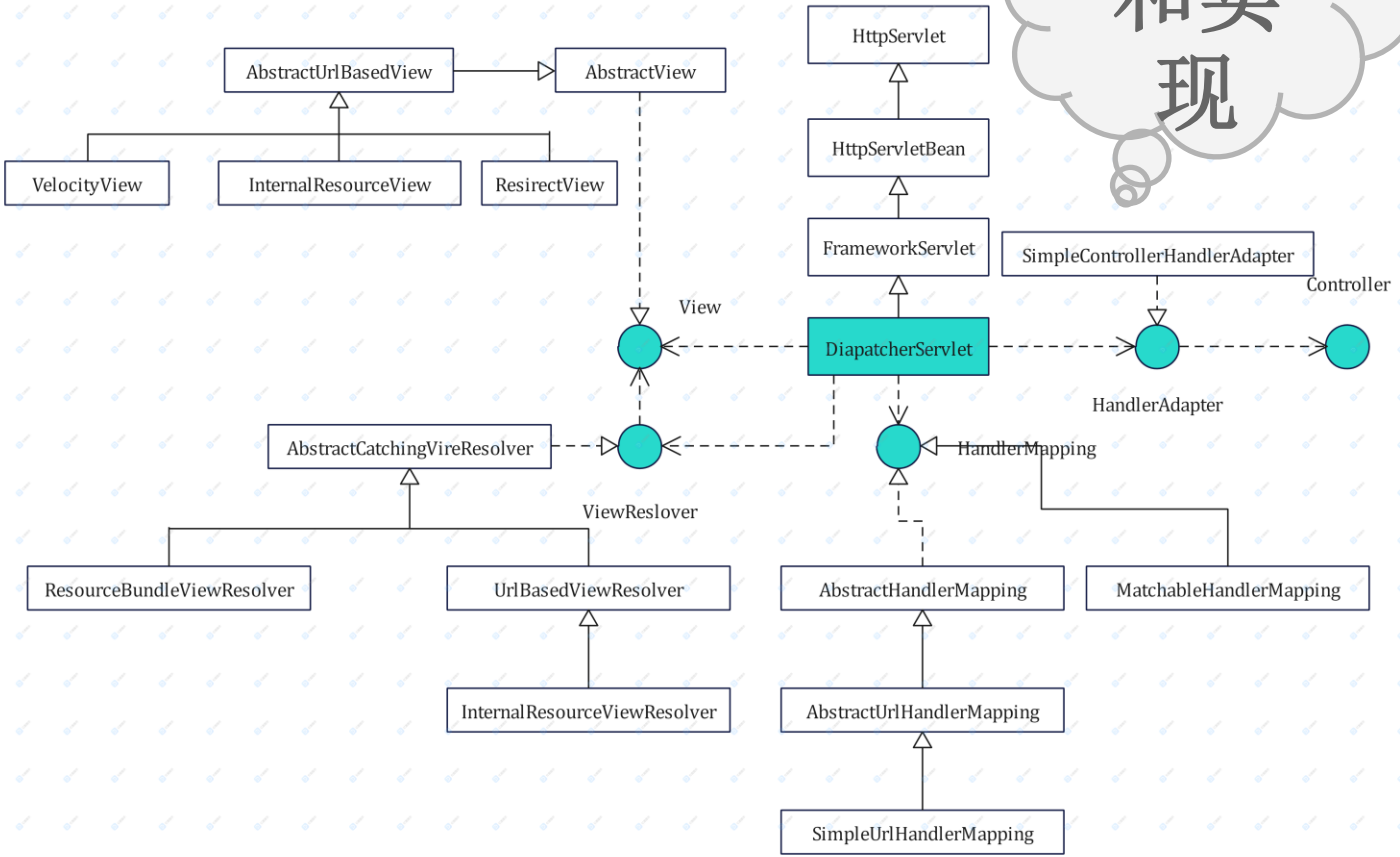

HttpServlet
AbstractUrlBasedView
AbstractView
HttpServletBean
VelocityView
InternalResourceView
ResirectView
FrameworkServlet
SimpleControllerHandlerAdapter
Controller
View
DiapatcherServlet
HandlerAdapter
AbstractCatchingVireResolver
HandlerMapping
ViewReslover
ResourceBundleViewResolver
UrlBasedViewResolver
AbstractHandlerMapping
MatchableHandlerMapping
InternalResourceViewResolver
AbstractUrlHandlerMapping
SimpleUrlHandlerMapping

02 类之间的关系


依赖
HttpServlet
AbstractUrlBasedView
AbstractView
HttpServletBean
VelocityView
InternalResourceView
ResirectView
FrameworkServlet
SimpleControllerHandlerAdapter
Controller
View
DiapatcherServlet
HandlerAdapter
AbstractCatchingVireResolver
HandlerMapping
ViewReslover
ResourceBundleViewResolver
UrlBasedViewResolver
AbstractHandlerMapping
MatchableHandlerMapping
InternalResourceViewResolver
AbstractUrlHandlerMapping
SimpleUrlHandlerMapping

02 类之间的关系

继承和实现



高级设计思想

01 策略模式

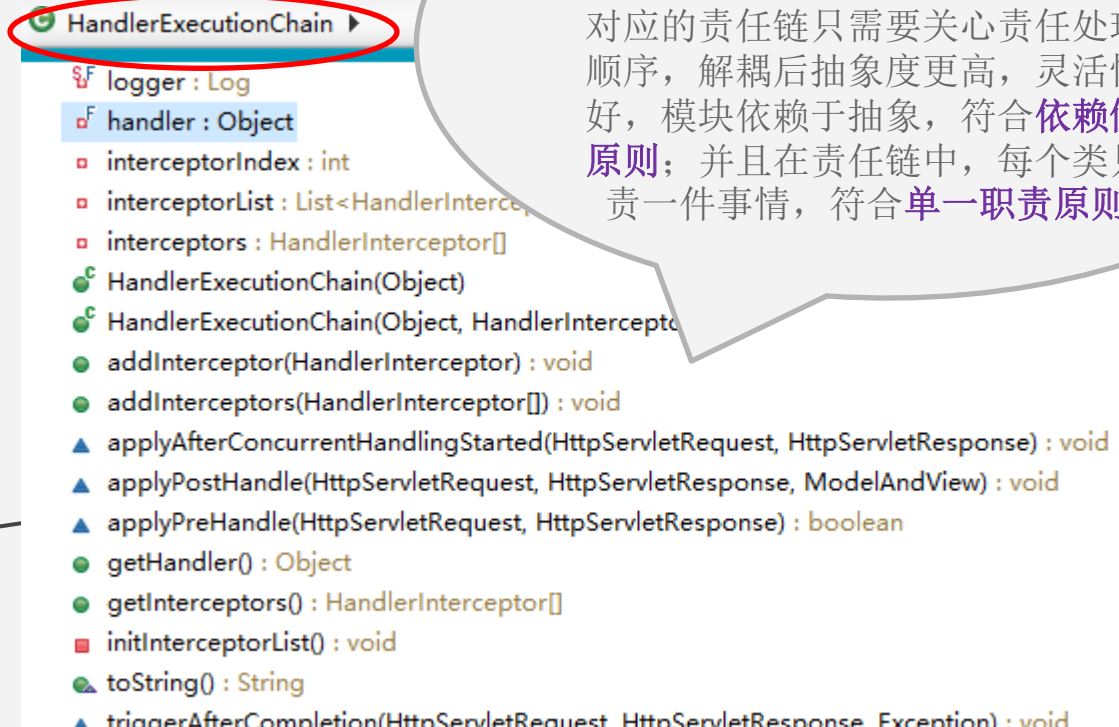
- 在有多种算法相似的情
  护
- - 定义一系列的算法
  使它们可相互替换，使

LocaleResolver.class，而这个类有多个实现类，对应多种不同的处理方式，就利用**接口、继承和多态**来替代原来的if-else实现方法，增加了代码的可读性和可维护性；类是可扩展的，符合面向对象的**开放封闭原则**

```
private void initLocaleResolver(ApplicationContext cont
    try {
        this.localeResolver = context.getBean(LOCALE_RESOLVER_BEAN_NAME, LocaleResolver.class);
        if (logger.isTraceEnabled()) {
            logger.trace("Detected " + this.localeResolver);
        }
        else if (logger.isDebugEnabled()) {
            logger.debug("Detected " + this.localeResolver.getClass().getSimpleName());
        }
    }
    catch (NoSuchBeanDefinitionException ex) {
        // We need to use the default.
        this.localeResolver = getDefaultStrategy(context, LocaleResolver.class);
        if (logger.isTraceEnabled()) {
            logger.trace("No LocaleResolver '" + LOCALE_RESOLVER_BEAN_NAME +
                    "': using default [" + this.localeResolver.getClass().getSimpleName() + "]");
        }
    }
}
```

02 责任链模式

- 定义多个可以处理请求（承担责任）的
沿着该链向下传递请求（责任）
的类处理之。
- 使得请求者和处理者之

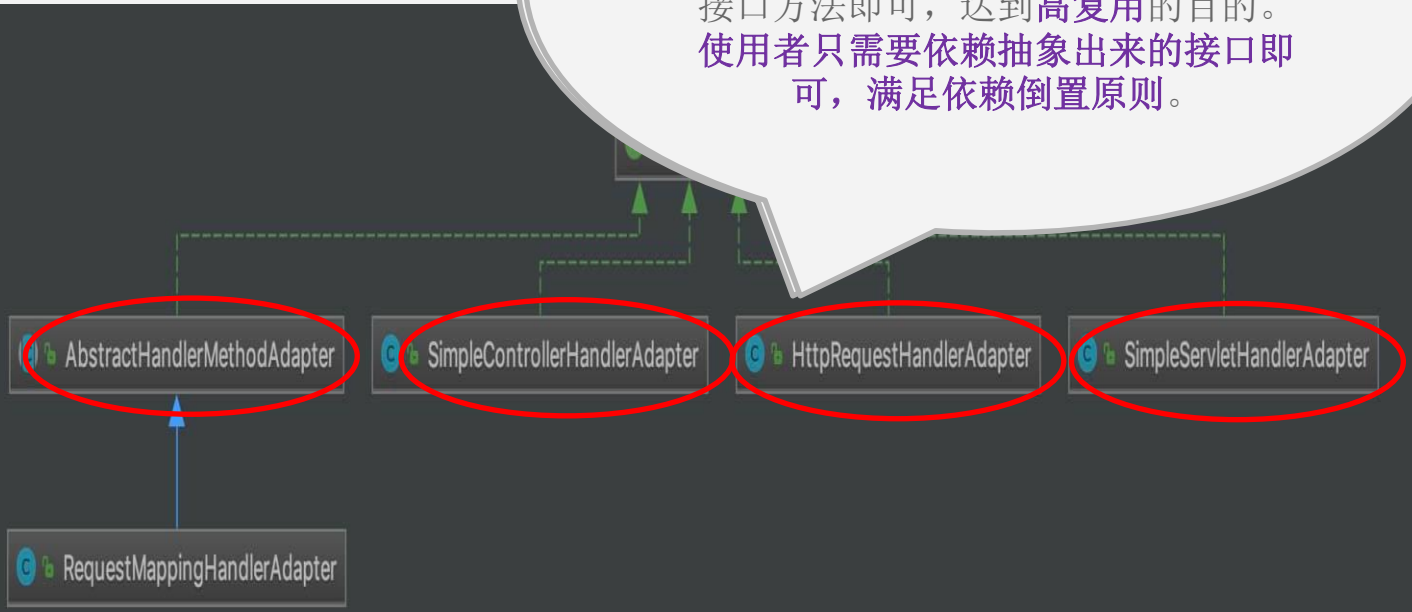


责任链将顺序过程处理的代码和逻辑，**解耦**成执行的顺序逻辑和对应的负责人，**请求者和处理者之间关系解耦，**对应的责任链只需要关心责任处理的顺序，解耦后抽象度更高，灵活性更好，模块依赖于抽象，符合**依赖倒置原则**；并且在责任链中，每个类只负责一件事情，符合**单一职责原则**。

03 适配器模式

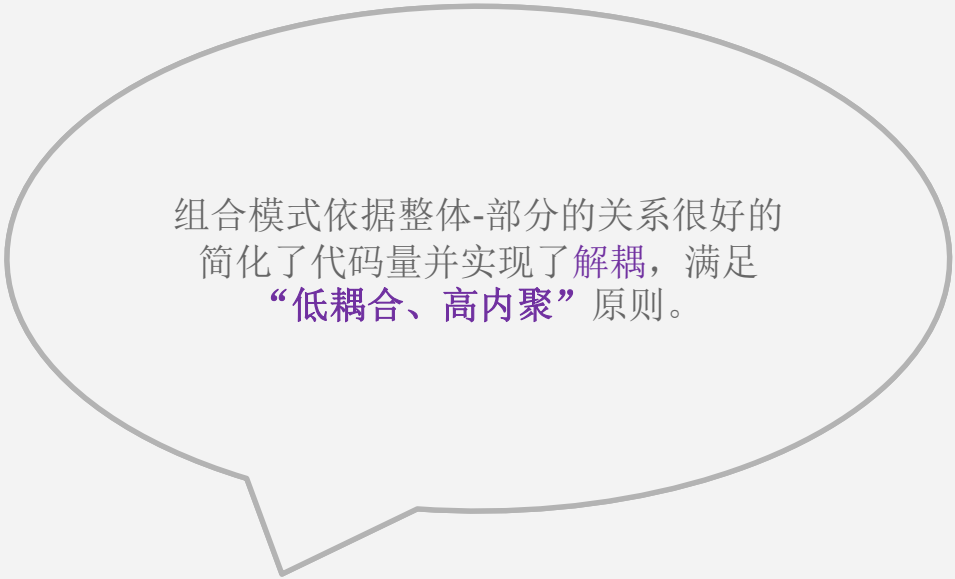
- 把一个类的接口扩展成客户
- 使得原来不能一起使用的
一起工作。

在HandlerAdapter中，适配器模式将**所需要用的类与使用者相解耦**，使用者只需关注相应的适配器接口提供是接口方法即可，达到**高复用**的目的。**使用者只需要依赖抽象出来的接口即可，满足依赖倒置原则**。



04 组合模式

- 将对象组合成树形结构以表示“部分-整体”的层次结构，用户对单个对象和组合对象的使用具有一致性。
- 使高层调用简单。



THANKS!

参　考

- ✓ 郝佳：Spring源码深度分析
- ✓ CSDN：[强烈推荐]SpringMVC源码分析郝佳：Spring源码深度分析
- ✓ 韩路彪：《看透Spring MVC：源代码分析与实践》
- ✓ Spring MVC源码分析(五)（HandlerMethod执行过程解析)https://blog.csdn.net/qq_38975553/article/details/103704036