# 44 Spring控制器Controller如何设置AOP?

更新时间：2020-08-31 10:52:22

> 学习这件事不在乎有没有人教你，最重要的是在于你自己有没有觉悟和恒心。 —— 法布尔

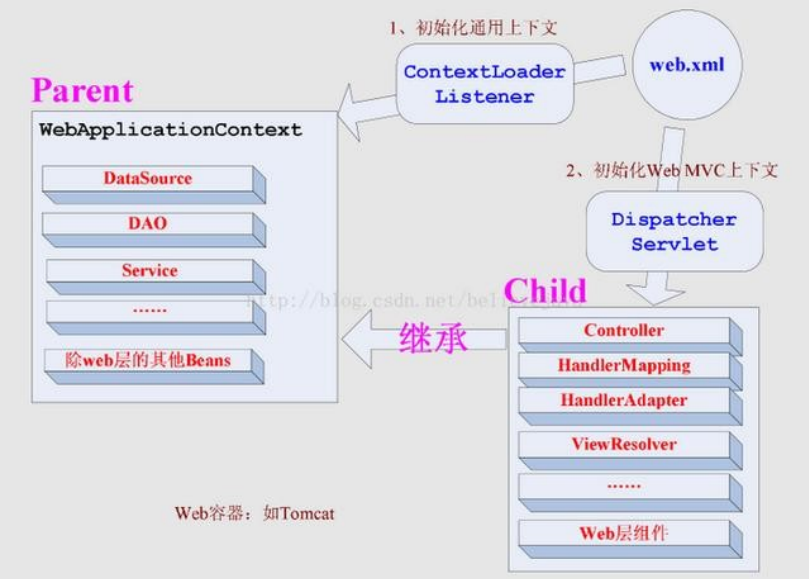
## 背景

做项目时碰到 Controller 不能使用 AOP 进行拦截，从网上搜索得知：使用 Spring MVC 启动了两个 context：ApplicationContext 和 WebApplicationContext 。

## 排查分析

- ContextLoaderListener 创建基于 Web 的应用根 ApplicationContext 并将它放入到 ServletContext。ApplicationContext 加载或者卸载 Spring 管理的 beans。在 Structs 和 Spring MVC 的控制层都是这样使用的；
- DispatcherServlet 创建自己的 WebApplicationContext 并管理这个 WebApplicationContext 里面的 handlers/controllers/view-resolvers；
- 当 ContextLoaderListener 和 DispatcherServlet 一起使用时, ContextLoaderListener 先创建一个根 ApplicationContext，然后 DispatcherSerlvet 创建一个子 ApplicationContext 并且绑定到根 ApplicationContext。

首先来看看 web.xml 的定义：




一般的 Web 应用，通过 ContextLoaderListener 监听，ContextLoaderListener 中加载的 context 成功后，Spring 将 ApplicationContext 存放在 ServletContext 中 key 值为"org.springframework.web.context.WebApplicationContext.ROOT"的 attribute 中。

```
<listener>
   <listener-class>
      org.springframework.web.context.ContextLoaderListener
   </listener-class>
</listener>
<context-param>
   <param-name>contextConfigLocation</param-name>
   <param-value>classpath*:conf/applicationContext*.xml</param-value>
</context-param>
```

DispatcherServlet 加载的 context 成功后，会将 ApplicationContext 存放在 org.springframework.web.servlet.FrameworkServlet.CONTEXT. + (servletName)的 attribute 中。

```
<servlet>
 <servlet-name>mvcServlet</servlet-name>
 <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
 <init-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>classpath:conf/spring-dispatcher-servlet.xml</param-value>
   </init-param>
   <load-on-startup>1</load-on-startup>
</servlet>
```

当然，如果没有指定 *servlet.xml 配置，则默认使用 DispatcherServlet 的默认配置 DispatcherServlet.properties。

```
# Default implementation classes for DispatcherServlet's strategy interfaces.
# Used as fallback when no matching beans are found in the DispatcherServlet context.
# Not meant to be customized by application developers.
org.springframework.web.servlet.LocaleResolver=org.springframework.web.servlet.i18n.AcceptHeaderLocaleResolver
org.springframework.web.servlet.ThemeResolver=org.springframework.web.servlet.theme.FixedThemeResolver
org.springframework.web.servlet.HandlerMapping=org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping,\ org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping
org.springframework.web.servlet.HandlerAdapter=org.springframework.web.servlet.mvc.HttpRequestHandlerAdapter,\ org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter,\ org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter
org.springframework.web.servlet.HandlerExceptionResolver=org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerExceptionResolver,\
```

简单的来说：Spring Bean 的管理在 ApplicationContext 中，ContextLoaderListener 的作用：

将 ApplicationContext 的生命周期和 servletContext 的生命周期联系到一起；

自动管理 ApplicationContext 的创建，ContextLoaderListener 给我们提供了一个便利，不用显式的去创建 ApplicationContext。

DispatcherServlet 相关 bean 的管理在 WebApplicationContext，ServletContextListener 创建 WebApplicationContext，WebApplicationContext 可以访问 ServletContext/ServletContextAware 这些 bean，还可以访问 getServletContext 方法。

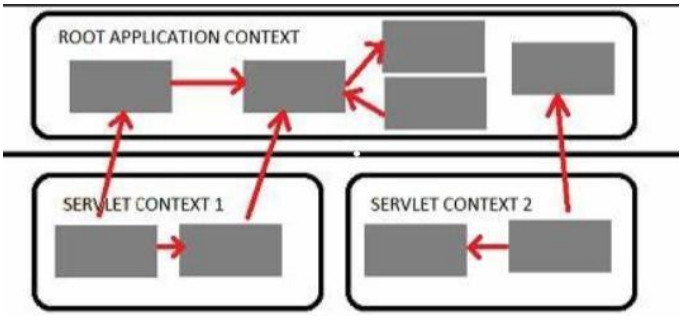
正式的官方文档：

> A web application can define any number of DispatcherServlets. Each servlet will operate in its own namespace, loading its own application context with mappings, handlers, etc. Only the root application context as loaded by ContextLoaderListener, if any, will be shared. As of Spring 3.1, DispatcherServlet may now be injected with a web application context, rather than creating its own internally. This is useful in Servlet 3.0+ environments, which support programmatic registration of servlet instances.

在一个 Web 应用中使用多个 DispatcherServlet，每个 Servlet 通过自己的命名空间来获取自己的 WebApplicationContext，然后加载此 ApplicationContext 里面的 HangdlerMapping，HandlerAdapter 等等。ContextLoaderListener 加载根 application，所有子 ApplicationContext 共享根 ApplicationContext。

从版本 3.1 后，Spring 支持将 DispatcherServlet 注入到根 ApplicationContext，而不用创建自己的 WebApplicationContext，这主要为支持 Servlet 3.0 以上版本环境的要求，因为 Servlet 3.0 以上版本支持使用编程的方式来注册 Servlet 实例。

Spring 支持使用多层层次 ApplicationContext，通常我们使用两层结构就够了。



## 解决问题

当 Controller 和 Service 不在同一个 ApplicationContext 中，如一个配置在 Root ApplicationContext，一个配置在 WebApplicationContext 中，虽然在 WebApplicationContext 使用了 AOP 配置，并且也生效了，但在 Controller 时却不能生效，原因就在于上面所说的两个不同 ApplicationContext，如果想要解决这个问题，可以使用两种方式：

1.在 root ApplicationContext 同样配置 AOP 开关；

2.从版本 3.1 后，Spring 支持将 DispatcherServlet 注入到根 ApplicationContext，可以使用同一套 AOP 注解。

## 总结

因 Spring 继承 Spring MVC 时，生成了两个 ApplicationContext，ContextLoaderListener 创建基于 Web 的应用根 ApplicationContext 并将它放入到 ServletContext。DispatcherServlet 创建自己的 WebApplicationContext 并管理这个 WebApplicationContext 里面的 bean。当需要在 Controller 层设置 AOP 时：

在低版本（3.1 之前）的 Spring 中，那么需要将 `<aop:aspectj-autoproxy proxy-target-class="true"/>` 配置到 dispatcher-servlet.xml（MVC 文件）当中并且设置包扫描规则为 **"use-default-filters="true"**。 将横切关注点封装为切面，切面中方法的注入是根据切点表达式来决定的。

高版本的 Spring 中，Spring 支持将 DispatcherServlet 注入到根 ApplicationContext，而不用创建自己的 WebApplicationContext，这时使用同一套 ApplicationContext 就不会出现上面的问题了。

}

43 离开了 Spring AOP，我们如何切面编程？

45 Spring中多个AOP如何协调执行？