**春天的杰作Spring源码分析**

主讲：Cat 老师

北京动力节点教育科技有限公司  
2020 • 北京

动力节点•版权所有•禁止传播

## Spring MVC 源码分析

Tomcat是运行Java EE规范下的web程序的服务器，它实现了Java EE 规范，Java EE本身是提出了一系列的技术标准和规范（接口），但是没有具体实现，具体的实现都是各个软件厂商实现的，在Java生态下，要实现web项目开发，切入点是JavaEE规范下的Servlet，Servlet是规范和接口，Servlet可以在Tomcat服务器上运行，像我们此次需要分析SpringMVC的源码，SpringMVC就是把JavaEE规范下的Servlet整合进来，这样就可以实现使用SpringMVC开发Java web项目，整个逻辑可以如下进行表述：

Oracle(原来sun公司) --> Web开发的Servlet规范 -->被SpringMVC整合 --> 部署到Tomcat中运行；

## 注解版SpringMVC如何启动

没有web.xml配置，SpringMVC如何通过继承于

AbstractAnnotationConfigDispatcherServletInitializer

的类来启动SpringMVC？

1. Tomcat在启动的过程中会查找所有实现

javax.servlet.ServletContainerInitializer接口的实现类；

2、javax.servlet.ServletContainerInitializer的实现类通过@HandlesTypes注解进行扩展；

3、Tomcat启动时会通过加载被@HandlesTypes注解的类，并读取@HandlesTypes注解的value值，将读取出来的值作为参数传给

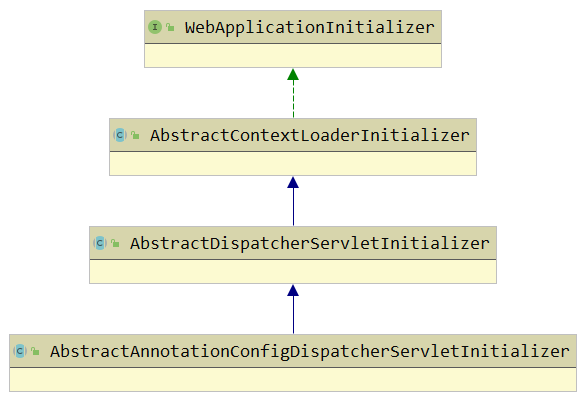
javax.servlet.ServletContainerInitializer实现类的onStartup ()方法；

4、Tomcat内部启动时调用 javax.servlet.ServletContainerInitializer的onStartup()方法，这个onStartup()方法的第一个参数就是@HandlesTypes注解的value 值指定的类的集合；

5、而在Spring MVC中，对javax.servlet.ServletContainerInitializer的实现就是org.springframework.web.SpringServletContainerInitializer, 而

org.springframework.web.SpringServletContainerInitializer类上面标注了@HandlesTypes注解，该注解指定的类就是org.springframework.web.WebApplicationInitializer接口，这样就将javax.servlet.ServletContainerInitializer和javax.servlet.WebApplicationInitializer关联起来了；

6、然后我们只需要在项目中写一个类实现org.springframework.web.WebApplicationInitializer接口，然后通过覆盖该接口里面的方法将我们的配置类告诉SpringMVC，即可完成对SpringMVC在Tomcat启动初始化过程中进行扩展，后续SpringMVC启动时就知道我们配置了什么信息，这就相当于把原来在web.xml里面做的工作移到了我们的配置类里面来完成，所以就可以废除web.xml文件；



org.springframework.web.WebApplicationInitializer接口

以上步骤进一步总结补充如下；

### 1、SPI机制

SPI即service privider interface，是JDK为提供的一种扩展和解耦机制。

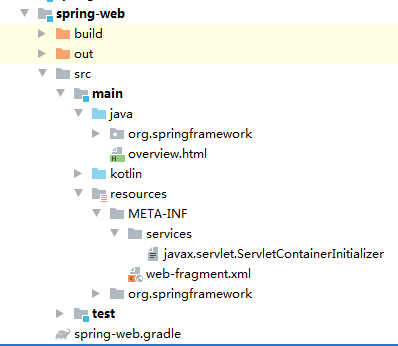
SPI的具体规范为：当服务的提供者提供了服务接口的一种实现后，在jar包的META-INF/services/目录下创建一个以包名+接口名命名的文件，该文件就是服务接口的具体实现类，而当外部程序加载这个jar包的时候就能通过META-INF/services/里的配置文件找到接口的具体实现类，基于这样一个约定规则就能避免在代码中指定和写死，并且jdk提供服务接口的实现类查找工具：java.util.ServiceLoader帮助我们完成查找；

### 2、spring-web中的具体应用

从servlet 3.0开始，web容器启动时为第三方组件提供了做一些初始化工作的机会，例如注册servlet或者filter等，servlet规范中通过ServletContainerInitializer实现此功能，每个框架要使用ServletContainerInitializer就必须在对应的jar包的META-INF/services 目录创建一个名为javax.servlet.ServletContainerInitializer的文件，文件内容指定具体的ServletContainerInitializer实现类，那么当web容器启动时就会运行这个初始化器做一些组件的初始化工作；

一般伴随着ServletContainerInitializer一起使用的还有@HandlesTypes注解，通过@HandlesTypes注解可以将感兴趣的一些类作为参数注入到ServletContainerInitializerde的onStartup方法中；

spring-web的jar定义了一个ServletContainerInitializerde的具体实现类，SpringServletContainerInitializer，并且在META-INF/services/目录下定义了如下文件：



文件的具体的内容为：

org.springframework.web.SpringServletContainerInitializer。

### 3、SpringServletContainerInitializer

配合注解@HandlesTypes它可以将其指定的Class对象作为参数传递到onStartup方法中，进而在onStartup方法中获取Class对象的具体实现类，进而调用实现类中的具体方法，SpringServletContainerInitializer类中@HandlesTypes指定的是Class对象是WebApplicationInitializer，

利用这个机制，只要实现WebApplicationInitializer这个接口，我们就可以自定义注入Servlet，或者Filter，也可以做其他扩展，即可以不再依赖web.xml的配置；

## SpringMVC注册版本启动源码分析

1. SpringServletContainerInitializer.onStartup(Set<?> WebApplicationInitializer, ServletContext)
2. WebApplicationInitializer.onStartup(ServletContext)
3. AbstractDispatcherServletInitializer.onStartup(ServletContext)
4. 注册ContextLoaderListener  
   registerContextLoaderListener(servletContext);

5、创建AnnotationConfigWebApplicationContext（Spring父容器）

6、把配置类RootConfig.class放在AnnotationConfigWebApplicationContext的annotatedClasses集合中；

1. 创建ContextLoaderListener监听器对象，把AnnotationConfigWebApplicationContext（Spring父容器）赋给ContextLoaderListener的context
2. 把ContextLoaderListener监听器添加到ServletContext中；

把监听器添加到servlet的上下文中，这是为后续启动埋下伏笔，到时候tomcat启动的某个步骤会调用监听器的方法；

9、注册DispatcherServletregisterDispatcherServlet(servletContext);

10、创建AnnotationConfigWebApplicationContext（spring子容器）

11、把配置类WebServletConfig.class放在AnnotationConfigWebApplicationContext的annotatedClasses

集合中；

1. 创建DispatcherServlet，把第11步创建的AnnotationConfigWebApplicationContext赋值给FrameworkServlet里面的webApplicationContext

13、把创建的DispatcherServlet加入到servletContext上下文中，

ServletRegistration.Dynamic registration = servletContext.addServlet(servletName, dispatcherServlet);

并设置servlet优先级为1，并且添加servlet的mapping映射为斜杠/

1. 进入到Tomcat源码，调用事件监听器ServletContextListener，而在第8步已经往ServletContext上下文中添加了ServletContextListener监听器，此时调用ServletContextListener监听器的contextInitialized(..)方法；
2. 执行initWebApplicationContext(event.getServletContext());

16、配置和刷新WebApplicationContext  
configureAndRefreshWebApplicationContext(cwac, servletContext);

17、刷新和创建Spring容器，里面就是spring ioc的12个步骤wac.refresh();

18、然后进入到tomcat源码，执行loadOnStartup(findChildren())

19、加载执行DispatchServlet的init方法，DispatchServlet没有init方法，执行父类HttpServletBean的init方法

20、执行FrameworkServlet的initServletBean()方法初始化WebApplicationContext容器（spring子容器），里面又是走spring ioc的12个步骤；

两个容器里面的对象是重复的，但是两个容器里面的每个单例对象不是同一个对象，不equals()，子容器里面有一个父容器parentBeanFactory，如果一个bean在子容器里面拿不到，那么就会去父容器获取，如果都获取不到，那就报错；

在实际开发中，也可以只配置一个容器，这样项目启动会更快，占用更少内存；