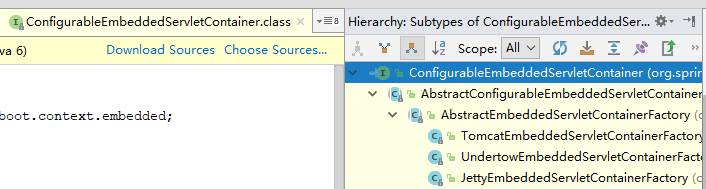
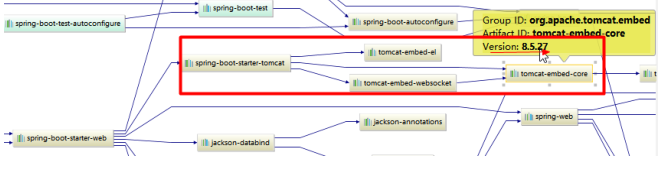
# 配置嵌入式Servlet容器

## 默认支持



## 默认使用servlet容器-tomcat

SpringBoot默认使用Tomcat作为嵌入式的Servlet容器；

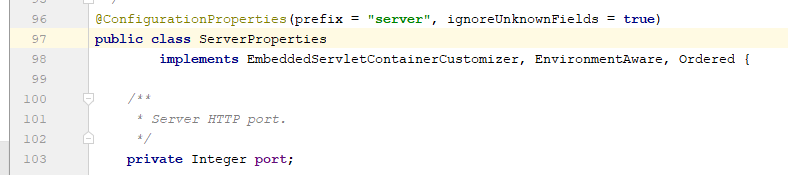


## 定制和修改Servlet容器的相关配置；

底层都是EmbeddedServletContainerCustomizer中customize（）方法

### 修改yml配置

修改和server有关的配置（ServerProperties【也是EmbeddedServletContainerCustomizer】）；





### 修改代码配置

2、编写一个EmbeddedServletContainerCustomizer：嵌入式的Servlet容器的定制器；来修改Servlet容器的配置



# 注册tomcat三大组件

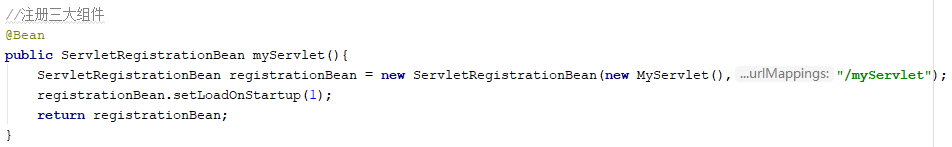
## 原因

由于SpringBoot默认是以jar包的方式启动嵌入式的Servlet容器来启动SpringBoot的web应用，没有web.xml文件。

## 注册组件

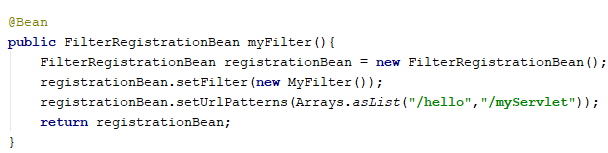
### 注册servlet

ServletRegistrationBean



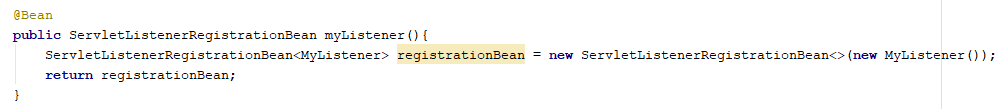
### 注册Filter

FilterRegistrationBean



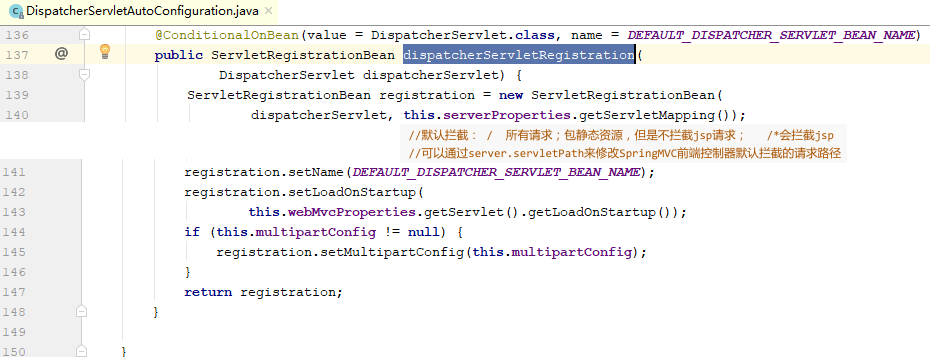
### 注册Listener

ServletListenerRegistrationBean



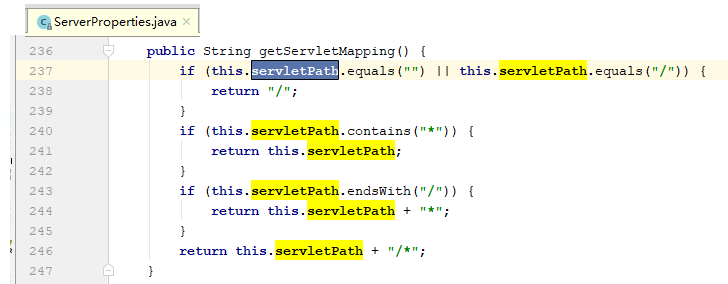
## 原理：注册前端控制器DIspatcherServlet

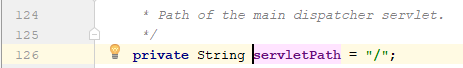
SpringBoot帮我们自动SpringMVC的时候，自动的注册SpringMVC的前端控制器；DIspatcherServlet；DispatcherServletAutoConfiguration中：



//默认拦截： /  所有请求；包静态资源，但是不拦截jsp请求；   /\*会拦截jsp

  //可以通过server.servletPath来修改SpringMVC前端控制器默认拦截的请求路径





# 配置其他嵌入式组件

## 默认支持：Tomcat（默认使用）

不必配置

<dependency>

   <groupId>org.springframework.boot</groupId>

   <artifactId>spring‐boot‐starter‐web</artifactId>

   引入web模块默认就是使用嵌入式的Tomcat作为Servlet容器；

</dependency>

## Jetty

先移除tomcat

<dependency>

   <groupId>org.springframework.boot</groupId>

   <artifactId>spring‐boot‐starter‐web</artifactId>

   <exclusions>

      <exclusion>

         <artifactId>spring‐boot‐starter‐tomcat</artifactId>

         <groupId>org.springframework.boot</groupId>

      </exclusion>

   </exclusions>

</dependency>

<!‐‐引入其他的Servlet容器‐‐>

<dependency>

   <artifactId>spring‐boot‐starter‐jetty</artifactId>

   <groupId>org.springframework.boot</groupId>

</dependency>

## Undertow

<dependency>

   <groupId>org.springframework.boot</groupId>

   <artifactId>spring‐boot‐starter‐web</artifactId>

   <exclusions>

      <exclusion>

         <artifactId>spring‐boot‐starter‐tomcat</artifactId>

         <groupId>org.springframework.boot</groupId>

      </exclusion>

   </exclusions>

</dependency>

<!‐‐引入其他的Servlet容器‐‐>

<dependency>

   <artifactId>spring‐boot‐starter‐undertow</artifactId>

   <groupId>org.springframework.boot</groupId>

</dependency>

# 嵌入式Servlet容器自动配置原理

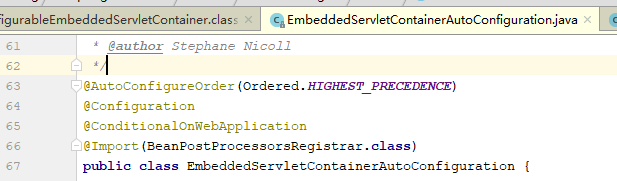
## EmbeddedServletContainerAutoConfiguration

EmbeddedServletContainerAutoConfiguration：嵌入式的Servlet容器自动配置？

//导入BeanPostProcessorsRegistrar：Spring注解版；给容器中导入一些组件

//导入了EmbeddedServletContainerCustomizerBeanPostProcessor：

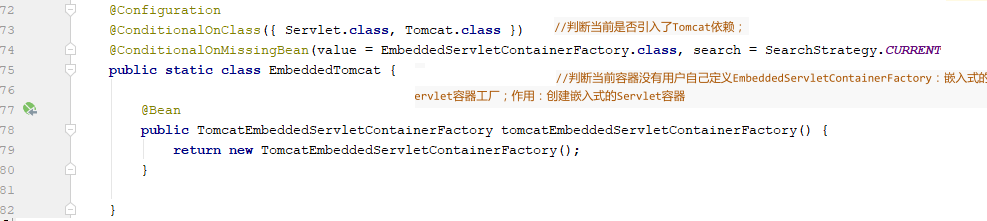
//后置处理器：bean初始化前后（创建完对象，还没赋值赋值）执行初始化工作



//判断当前是否引入了Tomcat依赖；

//判断当前容器没有用户自己定义EmbeddedServletContainerFactory：嵌入式的

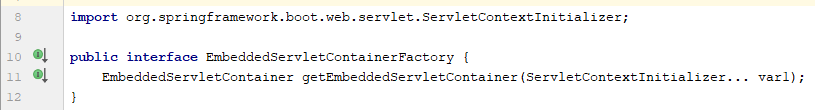
Servlet容器工厂；作用：创建嵌入式的Servlet容器

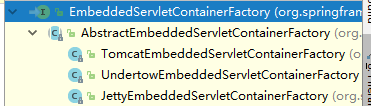


## 嵌入式servlet容器

### 容器工厂

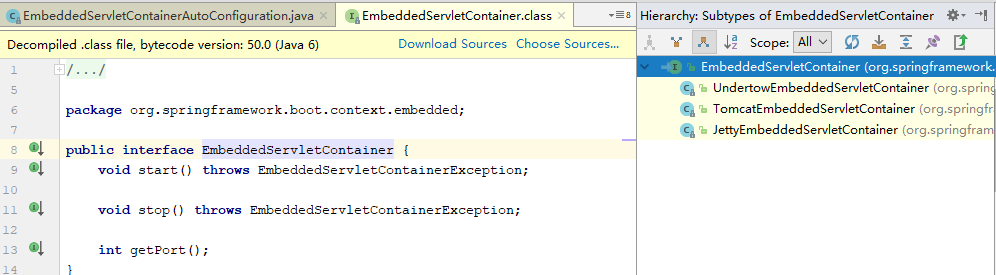
EmbeddedServletContainerFactory（嵌入式Servlet容器工厂）





### 容器

EmbeddedServletContainer：（嵌入式的Servlet容器

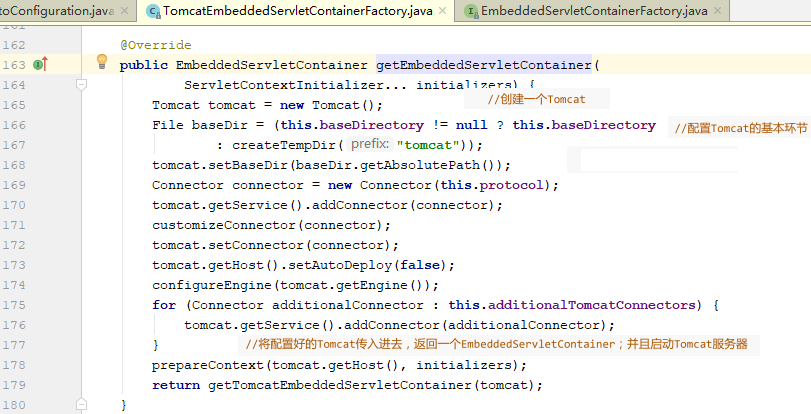


### 以TomcatEmbeddedServletContainerFactory为例

 //创建一个Tomcat

 //配置Tomcat的基本环节

 //将配置好的Tomcat传入进去，返回一个EmbeddedServletContainer；并且启动Tomcat服务器



## 定制和修改Servlet容器的相关配置原理

EmbeddedServletContainerCustomizer：定制器帮我们修改了Servlet容器的配置？

怎么修改的原理？

容器中导入了EmbeddedServletContainerCustomizerBeanPostProcessor

//如果当前初始化的是一个ConfigurableEmbeddedServletContainer类型的组件

//获取所有的定制器，调用每一个定制器的customize方法来给Servlet容器进行属性赋值；

 //从容器中获取所有这葛类型的组件：EmbeddedServletContainerCustomizer

 //定制Servlet容器，给容器中可以添加一个EmbeddedServletContainerCustomizer类型的组件



步骤：

1）、SpringBoot根据导入的依赖情况，给容器中添加相应的

EmbeddedServletContainerFactory【TomcatEmbeddedServletContainerFactory】

2）、容器中某个组件要创建对象就会惊动后置处理器；

EmbeddedServletContainerCustomizerBeanPostProcessor；

只要是嵌入式的Servlet容器工厂，后置处理器就工作；

3）、后置处理器，从容器中获取所有的EmbeddedServletContainerCustomizer，调用定制器的定制方法

# 嵌入式Servlet容器启动原理；

什么时候创建嵌入式的Servlet容器工厂？什么时候获取嵌入式的Servlet容器并启动Tomcat；

获取嵌入式的Servlet容器工厂：

1）、SpringBoot应用启动运行run方法

2）、refreshContext(context);SpringBoot刷新IOC容器【创建IOC容器对象，并初始化容器，创建容器中的每一

个组件】；如果是web应用创建AnnotationConfigEmbeddedWebApplicationContext，否则：

AnnotationConfigApplicationContext

3）、refresh(context);刷新刚才创建好的ioc容器；

4）、 onRefresh(); web的ioc容器重写了onRefresh方法

5）、webioc容器会创建嵌入式的Servlet容器；createEmbeddedServletContainer();

6）、获取嵌入式的Servlet容器工厂：

EmbeddedServletContainerFactory containerFactory = getEmbeddedServletContainerFactory();

从ioc容器中获取EmbeddedServletContainerFactory 组件；TomcatEmbeddedServletContainerFactory创建

对象，后置处理器一看是这个对象，就获取所有的定制器来先定制Servlet容器的相关配置；

7）、使用容器工厂获取嵌入式的Servlet容器：this.embeddedServletContainer = containerFactory

.getEmbeddedServletContainer(getSelfInitializer());

8）、嵌入式的Servlet容器创建对象并启动Servlet容器；

先启动嵌入式的Servlet容器，再将ioc容器中剩下没有创建出的对象获取出来；

IOC容器启动创建嵌入式的Servlet容器