

Java嵌入式tomcat整合SpringMVC

原创

陈虎_63

于 2021-12-02 12:12:48 发布

阅读量1.1k

收藏 3

点赞数2

版权

分类专栏:

spring

Java

文章标签:

tomcat

java

spring boot



Java 同时被 2 个专栏收录

1 订阅

30 篇文章

订阅专栏

章

订阅专栏

前言：

本文将介绍如何使用Java的方式启动tomcat，并整合Spring MVC，做到就像Springboot使用main方法启动，就可以访问controller资源的效果；

首先导入依赖：

```
1  <properties>
2      <embed.tomcat.version>9.0.21</embed.tomcat.version>
3  </properties>
4
5  <dependencies>
6
7      <dependency>
8          <groupId>org.apache.tomcat.embed</groupId>
9          <artifactId>tomcat-embed-core</artifactId>
10         <version>${embed.tomcat.version}</version>
11     </dependency>
12
13     <dependency>
14         <groupId>org.apache.tomcat.embed</groupId>
15         <artifactId>tomcat-embed-jasper</artifactId>
16         <version>${embed.tomcat.version}</version>
17         <!--<scope>provided</scope>-->
18     </dependency>
19
20
21     <dependency>
22         <groupId>org.springframework</groupId>
23         <artifactId>spring-context</artifactId>
24         <version>5.0.8.RELEASE</version>
25     </dependency>
26     <dependency>
27         <groupId>org.springframework</groupId>
28         <artifactId>spring-webmvc</artifactId>
29         <version>5.0.8.RELEASE</version>
30     </dependency>
31 </dependencies>
```

创建AppConfig配置类：

```
1  package com.hu.config;
2
3  import org.springframework.context.annotation.ComponentScan;
4  import org.springframework.context.annotation.Configuration;
5
6  /**
7   * @program: zdy-spring-boot
8   * @description:
9   * @author: hu.chen
10  * @createDate: 2021年12月01日 22:02
11  */
12 @Configuration
13 //添加包扫描路径
14 @ComponentScan({"com.hu"})
15 public class AppConfig {
16
```

```
17 | }
```

创建 MyWebApplicationInitializer 类实现 WebApplicationInitializer接口，重写onstartup方法

```
1  package com.hu.config;
2
3  import org.springframework.web.WebApplicationInitializer;
4  import org.springframework.web.context.support.AnnotationConfigWebApplicationContext;
5  import org.springframework.web.servlet.DispatcherServlet;
6
7  import javax.servlet.ServletContext;
8  import javax.servlet.ServletException;
9  import javax.servlet.ServletRegistration;
10
11 /**
12  * @program: zdy-spring-boot
13  * @description:
14  * @author: hu.chen
15  * @createDate: 2021年12月01日 21:59
16  */
17 public class MyWebApplicationInitializer implements WebApplicationInitializer {
18
19
20     public void onStartUp(ServletContext servletContext) throws ServletException {
21         //通过注解的方式初始化Spring的上下文
22         AnnotationConfigWebApplicationContext ac = new AnnotationConfigWebApplicationContext();
23
24         //注册spring的配置类（替代传统项目中xml的configuration）
25         ac.register(AppConfig.class);
26         ac.refresh();
27
28
29         //基于java代码的方式初始化DispatcherServlet
30         DispatcherServlet servlet = new DispatcherServlet(ac);
31         //绑定servlet
32         ServletRegistration.Dynamic registration = servletContext.addServlet("dispatcherServlet", servlet);
33
34         //设置tomcat启动立即加载 servlet
35         registration.setLoadOnStartup(1);
36         //浏览器访问uri
37         registration.addMapping("/app/*");
38     }
39 }
40
41
```

创建类MySpringServletContainerInitializer 并实现 ServletContainerInitializer 重写onstartup方法

```
1  package com.hu.config;
2
3
4  import javax.servlet.ServletContainerInitializer;
5  import javax.servlet.ServletContext;
6  import javax.servlet.ServletException;
7  import java.util.Set;
8
9  /**
10  * @program: zdy-spring-boot
11  * @description:
12  * @author: hu.chen
13  * @createDate: 2021年12月01日 21:46
14  */
15 public class MyServletContainerInitializer implements ServletContainerInitializer {
16
17     /**
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

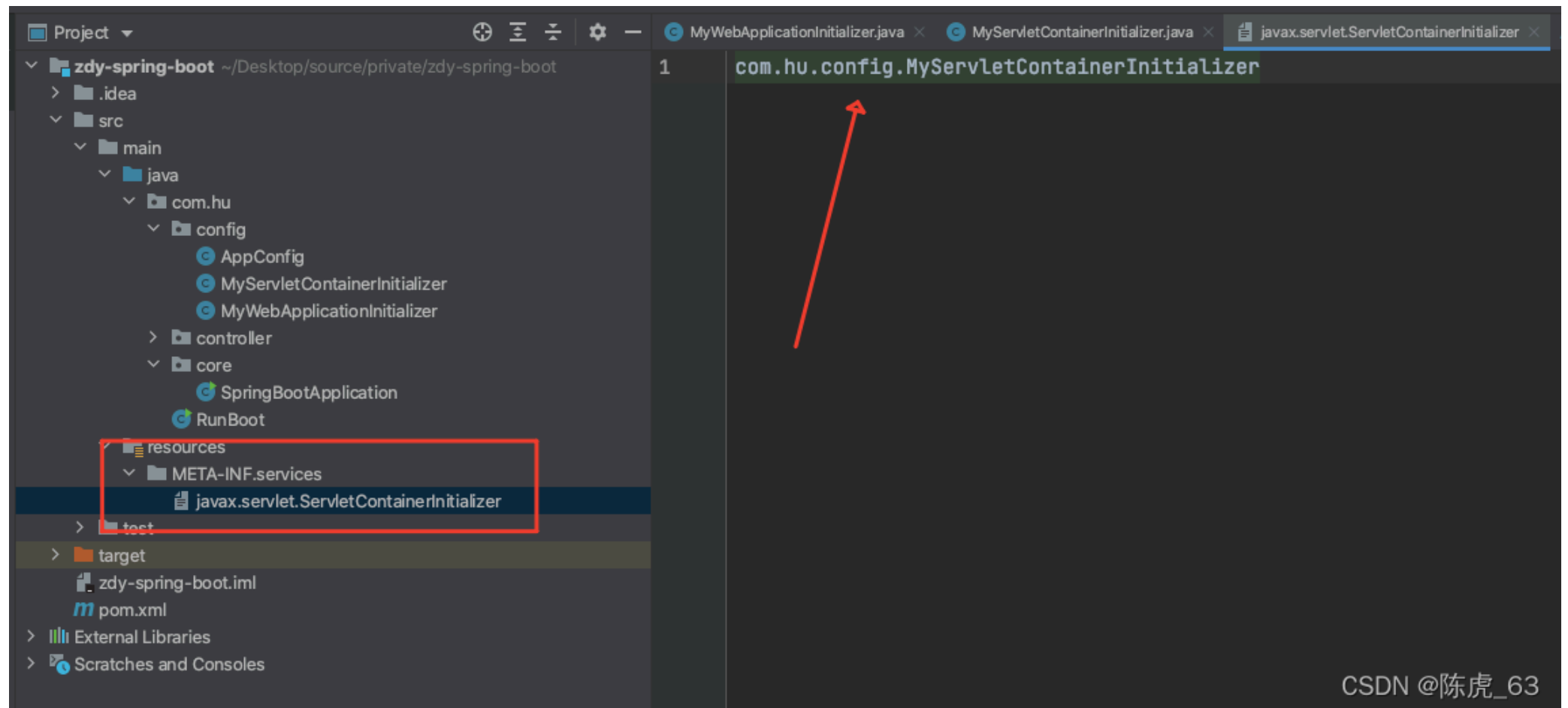
```

19     * @param set Servlet 3.0+ 容器启动时将自动扫描类路径以查找实现Spring的Webapplicationinitializer接口的所有实现,
20     * 将其放进一个Set集合中, 提供给ServletContainerInitializer中onStartup方法的第一个参数。
21     * @param servletContext
22     * @throws ServletException
23     */
24     public void onStartup(Set<Class<?>> set, ServletContext servletContext) throws ServletException {
25         System.out.println("加载.....");
26     }
27 }
28
29

```

在resources目录下, 创建META-INF/services/javax.servlet.ServletContainerInitializer

在文件中添加 MyServletContainerInitializer 类的全限定名, servlet3.0规范, 规定了tomcat在启动时会去扫描项目包括项目的jar包下这个 (META-INF/services) 目录下的 javax.servlet.ServletContainerInitializer 这个文件, 加载这个文件中配置的类的全限定名, 实例化并调用这个类中的 onstartup方法 (所以这个类必须实现ServletContainerInitializer这个接口)



创建启动类在main方法中实例化tomcat并启动

```

1  package com.hu.core;
2
3  import org.apache.catalina.LifecycleException;
4  import org.apache.catalina.connector.Connector;
5  import org.apache.catalina.startup.Tomcat;
6
7  /**
8   * @program: zdy-spring-boot
9   * @description:
10   * @author: hu.chen
11   * @createDate: 2021年12月01日 22:26
12   */
13  public class SpringBootApplication {
14
15      private static int port = 8080;
16      private static String contextPath = "/";
17
18      public static void run(){
19          Tomcat tomcat = new Tomcat();
20          String baseDir = Thread.currentThread().getContextClassLoader().getResource("").getPath();
21          //设置tomcat启动后的工作目录
22          tomcat.setBaseDir(baseDir);
23          //设置端口
24          tomcat.setPort(port);
25          //获取执行器, 并设置io协议
26          Connector connector = new Connector("org.apache.coyote.http11.Http11NioProtocol");
27          //设置端口

```

```

28         connector.setPort(port);
29         //设置执行器
30         tomcat.setConnector(connector);
31
32         tomcat.addWebapp(contextPath, baseDir);
33         tomcat.enableNaming();
34         try {
35             tomcat.start();
36         } catch (LifecycleException e) {
37             System.err.println("tomcat 启动失败");
38         }
39         //tomcat启动后, 让其阻塞, 不让当前线程结束, 等待处理请求,
40         tomcat.getServer().await();
41     }
42
43     public static void main(String[] args) {
44         run();
45     }
46 }
47
48

```

创建controller:

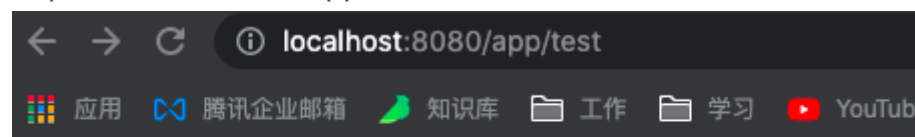
```

1     package com.hu.controller;
2
3     import org.springframework.web.bind.annotation.GetMapping;
4     import org.springframework.web.bind.annotation.RestController;
5
6     /**
7      * @program: zdy-spring-boot
8      * @description:
9      * @author: hu.chen
10     * @createDate: 2021年12月01日 22:23
11     **/
12     @RestController
13     public class TestController {
14
15         @GetMapping("/test")
16         public String test(){
17
18             return "hello";
19         }
20     }
21

```

浏览器访问:

http://localhost:8080/app/test



hello