8、Web-配置嵌入式Servlet容器

2020年6月9日 11:05

SpringBoot默认使用Tomcat作为嵌入式的Servlet容器

- 1）如何定制和修改Servlet容器相关配置
  - 1、修改和server有关的配置（ServerProperties【也是EmbeddedServletContainerCustomizer】）；

```
server.port=8080
server.tomcat.uri-encoding=utf-8
#配置静态资源路径spring.resources.static-locations=
server.servlet.context-path=/crud
```

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

```
publicclassMyServerConfig{
//配置嵌入式servlet容器
@Bean
publicWebServerFactoryCustomizer<ConfigurableWebServerFactory>
webServerFactoryCustomizer(){
returnnewWebServerFactoryCustomizer<ConfigurableWebServerFactory>(){
//定制嵌入式的servlet容器相关的规则
@Override
publicvoidcustomize(ConfigurableWebServerFactoryfactory){
factory.setPort(8082);
}
};
}
)
```

- 2）注册Servlet三大组件【Servlet、Filter、Listener】

  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。

  注册三大组件用以下方式

  ServletRegistrationBean

```
//注册三大组件
@Bean
publicServletRegistrationBeanmyServlet(){
ServletRegistrationBeanregistrationBean=newServletRegistrationBean(newMyServlet(),"/myServlet");
returnregistrationBean;
}
@Bean
publicFilterRegistrationBeanmyFilter(){
FilterRegistrationBeanfilterRegistrationBean=newFilterRegistrationBean();
filterRegistrationBean.setFilter(newMyfilter());
filterRegistrationBean.setUrlPatterns(Arrays.asList("/hello","/myServlet"));
returnfilterRegistrationBean;
}
@Bean
publicServletListenerRegistrationBeanmyListener(){
returnnewServletListenerRegistrationBean(newMyListener());
```

```
}
```

- 3）替换为其他嵌入式servlet容器

  默认支持：tomcat（默认使用）

  ```
  <dependency>
  <artifactId>spring-boot-starter-tomcat</artifactId>
  <groupId>org.springframework.boot</groupId>
  引入web模块默认就是使用嵌入式的Tomcat作为Servlet容器；
  </dependency>
  ```

  Undertow

  ```
  <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-undertow</artifactId>
  </dependency>
  ```

- 4、使用外置的Servlet容器

  嵌入式Servlet容器：应用打成可执行的jar

  优点：简单、便携；

  缺点：默认不支持JSP、优化定制比较复杂（使用定制器）

  外置的Servlet容器：外面安装Tomcat---应用war包的方式打包；

  步骤：

  - 1）、必须创建一个war项目；（利用idea创建好目录结构）
  - 2）、将嵌入式的Tomcat指定为provided；

    ```
    <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-tomcat</artifactId>
    <scope>provided</scope>
    </dependency>
    ```

  - 3）、必须编写一个SpringBootServletInitializer的子类，并调用configure方法

    ```
    publicclassServletInitializerextendsSpringBootServletInitializer{
    @Override
    protectedSpringApplicationBuilderconfigure(SpringApplicationBuilderapplication){
    returnapplication.sources(SpringBoot04WebJspApplication.class);
    }
    }
    ```

  - 4）、启动服务器就可以使用；