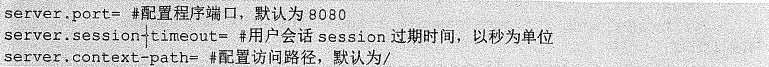
# Spring Boot核心-Tomcat配置

在SpringBoot中Tomcat 的所有属性都在org. springframework.boot.autoconfigure.web. ServerProperties配置类中做了定义，我们只需在application.properties 配置属性做配置即可。通用的Servlet 容器配置都以“server”作为前缀，而Tomcat 特有配置都以“ server.tomcat ＂作为前缀。下面是常用示例。

配置Servlet 容器：



配置Tomcat:



详细配置可见：<http://docs.spring.io/spring-boot/docs/1.5.6.RELEASE/reference/htmlsingle/#common-application-properties>

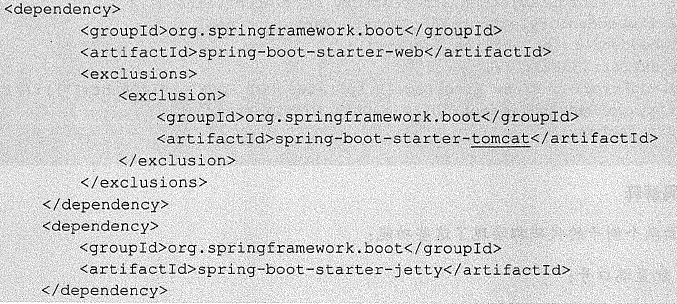
## 代码配置Tomcat

如果需要使用代码方式配置Servlet 容器，可以注册实现EmbeddedServletContainerCustomizer 接口的Bean。如果需要直接配置Tomcat 、Jetty 、Undertow，可以直接定义TomcatEmbeddedServletContainerFactory、JettyEmbeddedServletContainerFactory 、UndertowEmbeddedServletContainerFactory。

|  |
| --- |
| @Component **public class** CustomServletContainer **implements** EmbeddedServletContainerCustomizer {  @Override  **public void** customize(ConfigurableEmbeddedServletContainer container) {  container.setPort(**8888**);  container.addErrorPages(**new** ErrorPage(HttpStatus.***NOT\_FOUND***, **"/404.html"**));  container.setSessionTimeout(**10**, TimeUnit.***MINUTES***);  } } |

## 替换Tomcat

替换为Jetty：在pom.xml 中，将spring-boot-starter-web 的依由spring-boot-start-tomcat 替换为spring-boot-start-jetty。如果是替换成Undertow，则改成spring-boot-start-undertow。



## SSL配置

SSL (Secure Sockets Layer，安企套接层）是为网络通信提供安全及数据完整性的一种安全协议， SSL 在网络传输层对网络连接进行加密。SSL 协议位于TCP/IP 协议与各种应用层协议之间，为数据通信提供安全支持。SSL 协议可分为两层： SSL 记录协议（ SSL Record Protocol ），它建立在可靠的传输协议（如TCP ）之上，为高层协议提供数据封装、压缩、加密等基本功能的支持。SSL 握手协议（ SSL Handshake Protocol ），它建立在SSL 记录协议之上，用于实际数据传输开始前，通信双方进行身份认证、协商加密算法、交换加密密钥等。

而在基于B/S的 Web 应用中，是通过HTTPS 来实现SSL的。SpringBoot内嵌了Tomcat，使用SSL需要做如下操作。

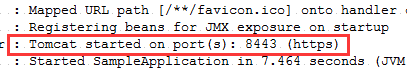
### 生成证书

使用SSL 首先需要一个证书，这个证书既可以是自签名的，也可以是从SSL 证书授权。每一个JDK 或者JRE 都有一个工具 keytool ，它是一叫个证书管理工具，可以用来生成自签名的证书。在控制台输入如下命令：keytool -genkey -alias tomcat，然后按照提示操作。会在用户目录下生成.keystore文件

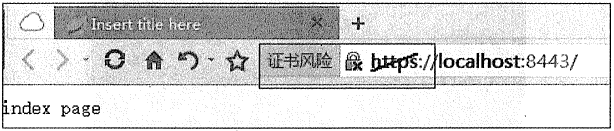
### SpringBoot配置SSL

|  |
| --- |
| **server.port**=**8443 server.ssl.key-store**=**classpath:.keystore server.ssl.key-store-password**=**123456 server.ssl.key-store-type**=**JKS server.ssl.key-alias**=**tomcat** |

启动Spring Boot ，控制台输出效果



访问：<https://localhost:8443>



### http 转向https

很多时候我们在地址栏输入的是http ，但是会自动转向到https，要实现这个功能，我们需配置TomcatEmbeddedServletContainerF actory，并且添加Tomcat的connector 来实现。

|  |
| --- |
| @Bean **public** EmbeddedServletContainerFactory containerFactory () {  TomcatEmbeddedServletContainerFactory factory =  **new** TomcatEmbeddedServletContainerFactory() {  @Override  **protected void** postProcessContext(Context context) {  SecurityConstraint constraint = **new** SecurityConstraint();  constraint.setUserConstraint(**"CONFIDENTIAL"**);  SecurityCollection collection = **new** SecurityCollection();  collection.addPattern(**"/\*"**);  constraint.addCollection(collection);  context.addConstraint(constraint);  }  };   factory.addAdditionalTomcatConnectors(httpConnector());   **return** factory; }  **private** Connector httpConnector() {  System.***out***.println(Http11NioProtocol.**class**.getName());  Connector connector = **new** Connector(Http11NioProtocol.**class**.getName());  connector.setScheme(**"http"**);  connector.setPort(**8080**);  connector.setSecure(**false**);  connector.setRedirectPort(**8443**);   **return** connector; } |

运行结果

