3.Web相关配置

3.1 配置tomcat

通用的Servlet容器配置都以"server"作为前缀

配置servlet容器:

```
server.port = 9090 #配置程序端口,默认为8080
server.servlet.session.timeout= #用户会话过期时间,以秒为单位,默认是30分钟
server.servlet.context-path= # 配置访问路径,默认是/
```

配置Tomcat:

```
server.tomcat.uri-encoding= #配置tomcat编码,默认是UTF-8
server.compression.enabled= #是否开启压缩,默认为关闭off
```

3.2 代码配置tomcat

1.通用配置

(1) 新建类的配置 ,spring boot 自动扫描

```
@Component
public class CustomServletContainer implements
WebServerFactoryCustomizer<ConfigurableServletWebServerFactory> {
   @Override
   public void customize(ConfigurableServletWebServerFactory factory) {
      factory.setPort(8888);
      factory.addErrorPages(new ErrorPage(HttpStatus.NOT_FOUND,"/404.html"));
      Session session = new Session();
      session.setTimeout(Duration.ofMinutes(30));
      factory.setSession(session);
   }
}
```

(2) 当前配置文件内配置.若要在当前已有的配置文件内添加类的Bean的话,则在Spring配置中,注意当前类要声明为static:

```
@SpringBootApplication
public class Ch7Application {

   public static void main(String[] args) {
      SpringApplication.run(Ch7Application.class, args);
   }

   @Component
   public static class CustomServletContainer implements
WebServerFactoryCustomizer<ConfigurableWebServerFactory> {
      @Override
      public void customize(ConfigurableWebServerFactory factory) {
         factory.setPort(8989);
      }
```

```
    }
}
```

或者:

```
@SpringBootApplication
public class Ch7Application {

    public static void main(String[] args) {
        SpringApplication.run(Ch7Application.class, args);
    }
    @Bean
    public WebServerFactoryCustomizer<ConfigurableWebServerFactory>
webServerFactoryCustomizer(){
        return new WebServerFactoryCustomizer<ConfigurableWebServerFactory>() {
            @Override
            public void customize(ConfigurableWebServerFactory factory) {
                factory.setPort(9999);
            }
        };
    }
}
```

2.特定配置

以tomcat为例

```
@Bean
public TomcatServletWebServerFactory  servletContainer(){
    TomcatServletWebServerFactory factory = new
TomcatServletWebServerFactory();
    factory.setPort(7777);
    return factory;
}
```

3.3 SSL配置

1.spring boot配置ssl

SSL(Secure Sockets Layer,安全套接字层),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.

```
keytool -genkeypair -alias tomcat -keyalg RSA -keystore ./tomcat.key
```

配置application.properties

```
server.port = 8443
logging.level.org.springframework.web = DEBUG
server.ssl.key-store = classpath:tomcat.key
server.ssl.key-store-password = 123456
server.ssl.keyStoreType = JKS
server.ssl.keyAlias: tomcat
```

2.http转向https

配置类中,或启动类中添加如下代码:

```
@Bean
public ServletWebServerFactory servletContainer() {
    TomcatServletWebServerFactory tomcat = new TomcatServletWebServerFactory()
{
        @Override
        protected void postProcessContext(Context context) {
            SecurityConstraint securityConstraint = new SecurityConstraint();
            securityConstraint.setUserConstraint("CONFIDENTIAL");
            SecurityCollection collection = new SecurityCollection();
            collection.addPattern("/*");
            securityConstraint.addCollection(collection);
            context.addConstraint(securityConstraint);
        }
    };
    tomcat.addAdditionalTomcatConnectors(createHTTPConnector());
    return tomcat;
}


@Bean
public Connector createHTTPConnector() {
    Connector connector = new
Connector("org.apache.coyote.http11.Http11NioProtocol");
    //同时启用http（8080）、https（8443）两个端口
    connector.setScheme("http");
    connector.setSecure(false);
    connector.setPort(8080);
    connector.setRedirectPort(8443);
    return connector;
}
```