

1.springboot-helloworld

springboot-helloworld

1.新建一个maven springboot web工程；

1.1 编写maven pom.xml配置

1.1.1 配置项目继承spring-boot-starter-parent

1.1.2 pom配置 Spring Boot web工程依赖

1.1.3 pom配置war 打包

1.1.4 配置 兼容非springboot内置tomcat中运行

1.1.5 maven update project 更新配置中的lib到本地

1.2.配置application.yml

1.3 配置日志打印，默认logback日志配置

1.4. springboot程序入口 继承SpringBootServletInitializer 类

1.4.1 页面动态访问

1.5 编译运行

1.5.1 Run as maven install

1.5.2 Run as Java Application

1.6.访问web静态资源

1.7. 访问web动态网页

1.新建一个maven springboot web工程；

create a simple project

Artifact	
Group Id:	com.lance
Artifact Id:	springboot-helloworld
Version:	0.0.1-SNAPSHOT
Packaging:	war
Compiler Level:	1.8
Name:	springboot-helloworld
Description:	springboot-helloworld
Parent Project	
Group Id:	org.springframework.boot
Artifact Id:	spring-boot-starter-parent
Version:	2.0.1.RELEASE
Advanced	

继承parent的一些默认配置
当前springboot版本号

1.1 编写maven pom.xml配置

参考：

```

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema
-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <!-- 1.spring-boot-starter-parent -->
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.0.1.RELEASE</version>
  </parent>

  <groupId>com.lance</groupId>
  <artifactId>springboot-helloworld</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <!-- war 打包 配置1-->
  <packaging>war</packaging>

  <name>springboot-helloworld</name>
  <description>springboot-helloworld</description>

  <dependencies>
    <!--2. springboot web工程 -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <!--4. 兼容在外部tomcat中运行 -->
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-tomcat</artifactId>
      <scope>provided</scope>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
      <!-- 3.war 打包 配置2 -->
      <plugin>
        <artifactId>maven-war-plugin</artifactId>
        <configuration>
          <version>3.1</version>
          <!-- 忽略找不到web.xml的错误 -->
          <failOnMissingWebXml>false</failOnMissingWebXml>
        </configuration>
      </plugin>

    </plugins>
  </build>

</project>

```

1.1.1 配置项目继承spring-boot-starter-parent

Maven的用户可以通过继承spring-boot-starter-parent项目来获得一些合理的默认配置。这个parent提供了以下特性：

- 默认使用Java 8
- 使用UTF-8编码
- 一个引用管理的功能，在dependencies里的部分配置可以不用填写version信息，这些version信息会从spring-boot-dependencies里得到继承。
- 识别过来资源过滤 (Sensible resource filtering.)
- 识别插件的配置 (Sensible plugin configuration (exec plugin, surefire, Git commit ID, shade).)
- 能够识别application.properties和application.yml类型的文件，同时也能支持profile-specific类型的文件（如： application-foo.properties and application-foo.yml，这个功能可以更好的配置不同生产环境下的配置文件）。

<!-- 继承默认配置 -->

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.0.1.RELEASE</version>
</parent>
```

1.1.2 pom配置 Spring Boot web工程依赖

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

1.1.3 pom配置war打包

```
<packaging>war</packaging>
<plugin>
  <artifactId>maven-war-plugin</artifactId>
  <configuration>
    <version>3.1</version>
    <!-- 忽略找不到web.xml的错误 -->
    <failOnMissingWebXml>false</failOnMissingWebXml>
  </configuration>
</plugin>
```

1.1.4 配置 兼容非springboot内置tomcat中运行

<!--spring boot tomcat（默认可以不用配置，但当需要把当前web应用布置到外部servlet容器时就需要配置，并将scope配置为provided）-->

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

1.1.5 maven update project 更新配置中的lib到本地

❏ Maven Dependencies

- ▷ spring-boot-starter-web-2.0.1.RELEASE.jar - D:\maven\
- ▷ spring-boot-starter-2.0.1.RELEASE.jar - D:\maven\repo:
- ▷ spring-boot-2.0.1.RELEASE.jar - D:\maven\repository\o
- ▷ spring-boot-autoconfigure-2.0.1.RELEASE.jar - D:\mave
- ▷ spring-boot-starter-logging-2.0.1.RELEASE.jar - D:\mav
- ▷ logback-classic-1.2.3.jar - D:\maven\repository\ch\qos\
- ▷ logback-core-1.2.3.jar - D:\maven\repository\ch\qos\lc
- ▷ slf4j-api-1.7.25.jar - D:\maven\repository\org\slf4j\slf4j
- ▷ log4j-to-slf4j-2.10.0.jar - D:\maven\repository\org\apac
- ▷ log4j-api-2.10.0.jar - D:\maven\repository\org\apache\
- ▷ jul-to-slf4j-1.7.25.jar - D:\maven\repository\org\slf4j\jul
- ▷ spring-core-5.0.5.RELEASE.jar - D:\maven\repository\or
- ▷ spring-jcl-5.0.5.RELEASE.jar - D:\maven\repository\org\
- ▷ snakeyaml-1.19.jar - D:\maven\repository\org\yaml\sn
- ▷ spring-boot-starter-json-2.0.1.RELEASE.jar - D:\maven\
- ▷ jackson-databind-2.9.5.jar - D:\maven\repository\com\
- ▷ jackson-annotations-2.9.0.jar - D:\maven\repository\co
- ▷ jackson-core-2.9.5.jar - D:\maven\repository\com\faste
- ▷ jackson-datatype-jdk8-2.9.5.jar - D:\maven\repository\
- ▷ jackson-datatype-jsr310-2.9.5.jar - D:\maven\repositor
- ▷ jackson-module-parameter-names-2.9.5.jar - D:\maven
- ▷ hibernate-validator-6.0.9.Final.jar - D:\maven\repositor
- ▷ validation-api-2.0.1.Final.jar - D:\maven\repository\java
- ▷ jboss-logging-3.3.2.Final.jar - D:\maven\repository\org
- ▷ classmate-1.3.4.jar - D:\maven\repository\com\fastern
- ▷ spring-web-5.0.5.RELEASE.jar - D:\maven\repository\or
- ▷ spring-beans-5.0.5.RELEASE.jar - D:\maven\repository\

- ▷ spring-webmvc-5.0.5.RELEASE.jar - D:\maven\repository\org
- ▷ spring-aop-5.0.5.RELEASE.jar - D:\maven\repository\org\sp
- ▷ spring-context-5.0.5.RELEASE.jar - D:\maven\repository\org\
- ▷ spring-expression-5.0.5.RELEASE.jar - D:\maven\repository\
- ▷ spring-boot-starter-tomcat-2.0.1.RELEASE.jar - D:\maven\re
- ▷ javax.annotation-api-1.3.2.jar - D:\maven\repository\javax\ai
- ▷ tomcat-embed-core-8.5.29.jar - D:\maven\repository\org\ap
- ▷ tomcat-embed-el-8.5.29.jar - D:\maven\repository\org\apac
- ▷ tomcat-embed-websocket-8.5.29.jar - D:\maven\repository\

1.2.配置application.yml

application.yml

```
1 server: #打包成war，发布到tomcat中时，以tomcat配置为准，此处server配置可能失效
2   port: 8088
3   session.timeout: 30
4   tomcat.max-threads: 0
5   tomcat.uri-encoding: UTF-8
```

1.3 配置日志打印，默认logback日志配置

- ❏ springboot-helloworld
 - src/main/java
 - ❏ src/main/resources
 - application.yml
 - logback.xml
 - src/test/java

1.4. springboot程序入口 继承SpringBootServletInitializer 类

```

@SpringBootApplication //让spring boot自动给程序进行必要的配置
public class Application extends SpringBootServletInitializer{//1.继承SpringBootServletInitializer

//spring-boot-parent 默认使用logback 和slf4j 日志工具
private static Logger logger = LoggerFactory.getLogger(Application.class);

@Override //2.重写configure方法
protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
    return application.sources(Application.class);
}

public static void main(String[] args){//3.main方法入口，如果不使用内置tomcat，可以省略main函数
    logger.debug("enter main method");

    SpringApplication.run(Application.class, args);
}
}

```

1.4.1 页面动态访问

```

@Controller
public class HelloWorldController {

    private Logger logger = LoggerFactory.getLogger(getClass());

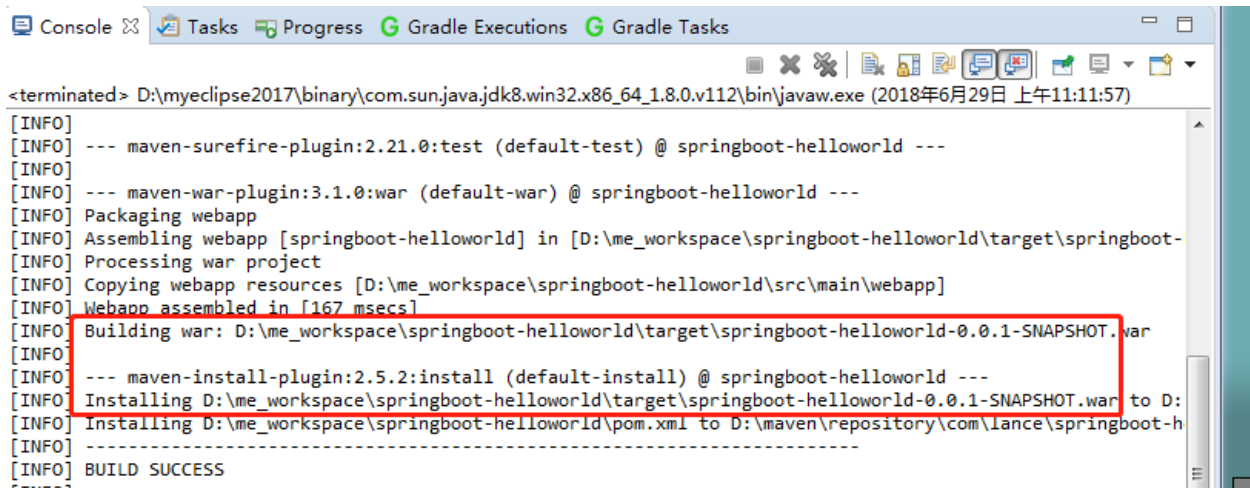
    @RequestMapping("/hello") //若"/"未配置，则默认指定为静态页面index.html
    @ResponseBody
    public String hello(){
        logger.debug("enter hello page");
        return "Hello World";
    }
}

```

1.5 编译运行

1.5.1 Run as maven install

打包成war包，拷贝到tomcat的webapps下运行



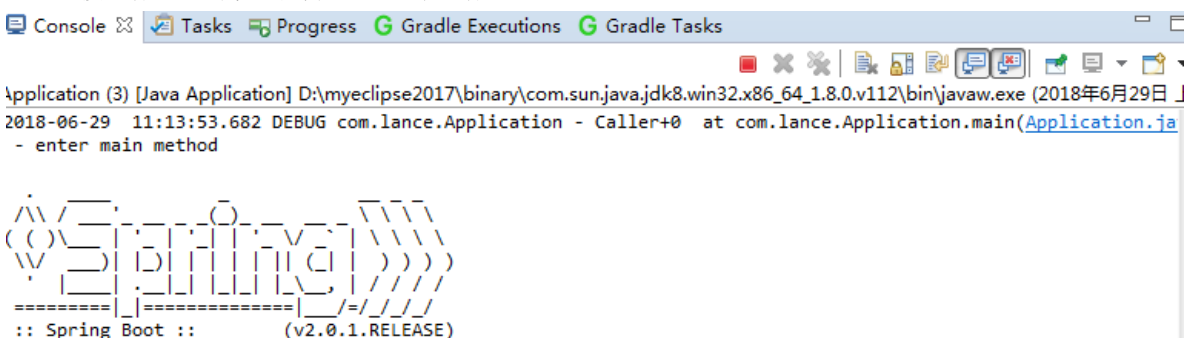
```

<terminated> D:\myeclipse2017\binary\com.sun.java.jdk8.win32.x86_64_1.8.0.v112\bin\javaw.exe (2018年6月29日 上午11:11:57)
[INFO] --- maven-surefire-plugin:2.21.0:test (default-test) @ springboot-helloworld ---
[INFO] --- maven-war-plugin:3.1.0:war (default-war) @ springboot-helloworld ---
[INFO] Packaging webapp
[INFO] Assembling webapp [springboot-helloworld] in [D:\me_workspace\springboot-helloworld\target\springboot-
[INFO] Processing war project
[INFO] Copying webapp resources [D:\me_workspace\springboot-helloworld\src\main\webapp]
[INFO] Webapp assembled in [167 msecs]
[INFO] Building war: D:\me_workspace\springboot-helloworld\target\springboot-helloworld-0.0.1-SNAPSHOT.war
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ springboot-helloworld ---
[INFO] Installing D:\me_workspace\springboot-helloworld\target\springboot-helloworld-0.0.1-SNAPSHOT.war to D:
[INFO] Installing D:\me_workspace\springboot-helloworld\pom.xml to D:\maven\repository\com\lance\springboot-h
[INFO] BUILD SUCCESS

```

1.5.2 Run as Java Application

直接运行main函数，在内置tomcat中运行



```

Application (3) [Java Application] D:\myeclipse2017\binary\com.sun.java.jdk8.win32.x86_64_1.8.0.v112\bin\javaw.exe (2018年6月29日
2018-06-29 11:13:53.682 DEBUG com.lance.Application - Caller+0 at com.lance.Application.main(Application.java
- enter main method

:: Spring Boot ::                (v2.0.1.RELEASE)

```

1.6. 访问web静态资源

默认静态资源放置在/src/main/resources/static目录下

<http://localhost:8080/>

1.7. 访问web动态网页

<http://localhost:8080/hello>
