一 什么是 Spring boot

Springboot 是一个基于 spring 所有功能的工具框架.能够让一个 spring 框架的 开发过程简化,再简化,能应对非常多的开发场景实现自动配置.

例如:开发 web 应用, springboot 帮你完成了 web 容器的配置, SpringMvc, Spring 的配置

二特点

1 独立运行的 spring 容器

Spring 开发的项目的运行一般都需要别的容器的支持,比如开发一个 web 应用,要使用第三方的 web 容器。

但 Springboot 可以独立运行,一个 main 方法加载 springboot 运行代码

2 内嵌 Servlet 容器

Springboot 为 web 工程提供了内嵌的 web 容器,默认使用 tomcat(可以配置为 jetty 和 undertow),并且按照习惯进行默认配置,例如端口号 8080 项目访问路径 /.结合第一个特点,web 工程就可以不用封装 war 包访问外部的 web 容器运行,直接使用 jar 包运行.

3 简化依赖

Springboot 能够实现它的独有的特点,是因为它在 Spring 基础之上扩展了非常庞大的量的代码.导致要想使用 Spring 必须依赖大量的资源.这样极其不方便的.所以 Springboot 为开发者准备来的丰富环境的简化依赖。

例如:我们要开发一个 web 应用(spring-context, spring-webmvc, jackson, loggings 等等),在 springboot 只要想开发一个 web 应用,只需要依赖一个 spring-boot-starter-web,实现依赖的传递。

4 自动配置(核心特点)

springboot 最核心的特点,就是自动配置.它为大多数开发场景准备对应的配置逻辑.在使用 springboot 开发时,可以不考虑如何配置这些技术,环境,只需要按照 springboot 提供的逻辑准备你的技术,环境的各种尚需经

例如:自动完成 web 容器的配置

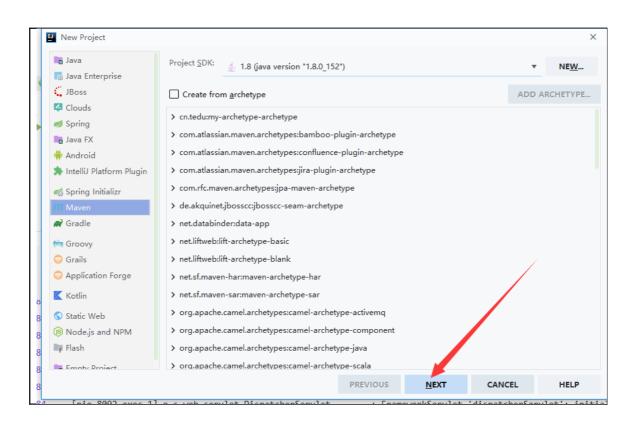
简化了端口,程序访问根目录,静态资源访问,前后缀拼接等等配置内容自动完成了 datasource 数据源配置

你一旦需要持久层数据源连接数据库,只需要提供对应属性就可以了

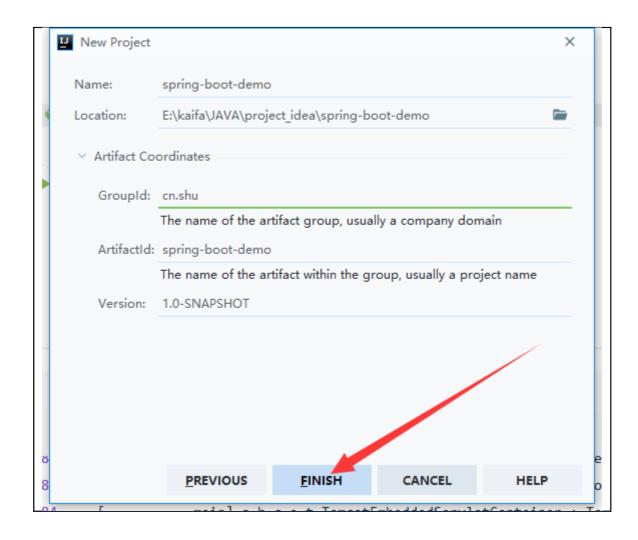
三 Spring Boot 入门(快速开发 SSM 项目)

1 创建一个 maven 项目

1.A 选择骨架(默认不选)



1.B 填写信息(groupId, ArtifactId)



1.C Finish 创建完成

```
spring-boot-demo > V pom.xml >
                   <?xml version="1.0" encoding="UTF-8"?>
> 🗀 spring-boot-demo E:\kaifa\JAVA\project_i
                                      project xmlns="http://maven.apache.org/POM/4.0.0"
> 🏥 External Libraries
                                               \verb|xmlns:xsi="http://www.w3.org/2001/XMLSchema-instar|\\
> E Scratches and Consoles
                                               xsi:schemaLocation="http://maven.apache.org/POM/4.
                                 4
                                 5
                                          <modelVersion>4.0.0</modelVersion>
                                 6
                                 7
                                          <groupId>cn.shu
                                          <artifactId>spring-boot-demo</artifactId>
                                 8
                                          <version>1.0-SNAPSHOT
                                 9
                                10
                                      </project>
```

2 配置 Spring 相关依赖(pom.xml)

2.A 继承 spring boot 资源

2.B 添加 spring web 应用的资源

3 开发一个 Controller 类(TestController)

```
package cn.shu;

▼ □ spring-boot-demo E:\kaifa\JAVA\project_i

 > 📷 .idea
                                           import org.springframework.boot.SpringApplication;

√ □ src

                                           import org.springframework.boot.autoconfigure.SpringBootApplication;

∨ □ main

                                   5
                                   6
     🗸 🗀 java
                                   7
                                            # @作者 舒新胜
       🗸 🗀 cn.shu
                                             * @项目 spring-boot-demo
                                   8
          ∨ toontroller
                                             * @创建时间 2020-5-22 17:06
                                   9
            DemoMain
                                  11
                                  12 🔕
                                            @SpringBootApplication
       nesources
                                  13 6
                                            public class DemoMain {
   > 📑 test
                                                public static void main(String[] args) {
                                  14
   V pom.xml
                                  15
   31 spring-boot-demo.iml
                                                     SpringApplication.run(DemoMain.class,args);
                                  16
> 🏥 External Libraries
                                   17
                                            }
> 🗏 Scratches and Consoles
```

4 开发启动类(唯一入口)

```
package cn.shu;

▼ □ spring-boot-demo E:\kaifa\JAVA\project_i

 > 📷 .idea
                                          bimport org.springframework.boot.SpringApplication;

✓ 
☐ src

                                   4
                                          himport org.springframework.boot.autoconfigure.SpringBootApplication;
   🗸 🗀 main
                                   5

→ ijava

                                            * @作者 舒新胜
       🗸 🗀 cn.shu
                                             * @项目 spring-boot-demo
                                   8
          ∨ 🛅 controller
                                   9
                                             * @创建时间 2020-5-22 17:06
              ■ TestController
                                   10
            12 🔕
                                           @SpringBootApplication
       nesources
                                           public class DemoMain {
                                  13 6
   > 📑 test
                                                public static void main(String[] args) {
                                  14
    V pom.xml
                                  15
    31 spring-boot-demo.iml
                                                    SpringApplication.run(DemoMain.class,args);
> || External Libraries
                                  17
                                           }
                                  18
> 

Scratches and Consoles
                                  19
```

5 测试

5.A 启动 main 方法

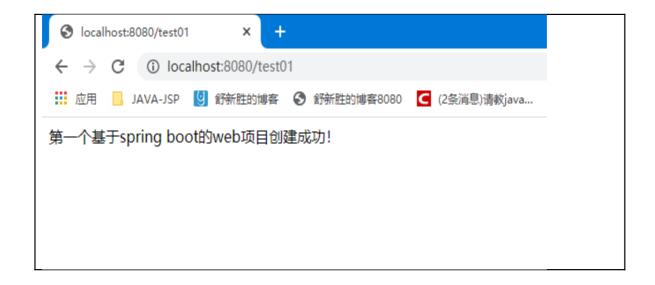
```
✓ □ spring-boot-demo E:\kaifa\JAVA\project i

                                                                          import org.springframework.boot.SpringApplication:
   ∨ 🔯 src
                                                                          import org.springframework.boot.autoconfigure.SpringBootApplication;
     ∨ 🗀 main
           ∨ 🗀 java
                                                                             * @作者 舒新胜
            ∨ 🗀 cn.shu
                                                                             * @项目 spring-boot-demo
              > 🛅 controller
                                                                             * @創建时间 2020-5-22 17:06

    ⊕ DemoMain

           resources
                                                           11
                                                            12 å ♥ @SpringBootApplication
                                                           13 ♠ ▶ | public class DemoMain {
   > matarget
                                                                                public static void main(String[] args) {
                                                           14
      31 spring-boot-demo.iml
                                                                                         SpringApplication.run(DemoMain.class,args);
> III External Libraries
                                                                            DemoMain
2020-05-22 17:14:40.139 INFO 15108 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'hiddenHttpMethodFilter' to: [/*
     2820-05-22 17:14:40.139 INFO 15108 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'httpPutFormContentFilter' to: [,
      1 2020-05-22 17:14:40.139 INFO 15108 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'requestContextFilter' to: [/*]
       🔱 2020-05-22 17:14:40.462 INFO 15108 --- [ main] s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking for @ControllerAdvice: org.springframewor
      = 2020-05-22 17:14:40.524 INFO 15108 --- [
                                                                                                 main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/test01]}" onto public java.lang.String
 To 2020-05-22 17:14:40.528 INFO 15108 --- [
                                                                                                  main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error]}" onto public org.springframewo
      5 2020-05-22 17:14:40.528 INFO 15108 --- [
                                                                                                main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error],produces=[text/html]}" onto pul
      2020-05-22 17:14:40.563 INFO 15108 --- [
                                                                                                 main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/webjars/**] onto handler of ty
            2020-05-22 17:14:40.563 INFO 15108 --- [
                                                                                                 main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/**] onto handler of type [class of the content of the content of type [class of the content of type content of type [class of type content 
             2020-05-22 17:14:40.597 INFO 15108 --- [
                                                                                                 main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/**/favicon.ico] onto handler or
             2020-05-22 17:14:40.752 INFO 15108 --- [
                                                                                                  main] o.s.j.e.a.AnnotationMBeanExporter
                                                                                                                                                                                : Registering beans for JMX exposure on startup
            2020-05-22 17:14:40.810 INFO 15108 --- [
                                                                                                 main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8080 (http)
             2020-05-22 17:14:40.816 INFO 15108 --- [
                                                                                                   main] cn.shu.DemoMain
                                                                                                                                                                                 : Started DemoMain in 2.747 seconds (JVM running fo
             2020-05-22 17:14:59.533 INFO 15108 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring FrameworkServlet 'dispatcher
```

5.B 网页测试

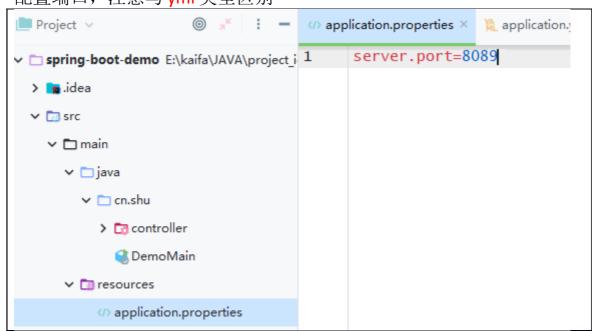


四 Spring Boot 的配置文件

配置文件需要以 application 命名,支持二种格式,需要放到 resource 目录下

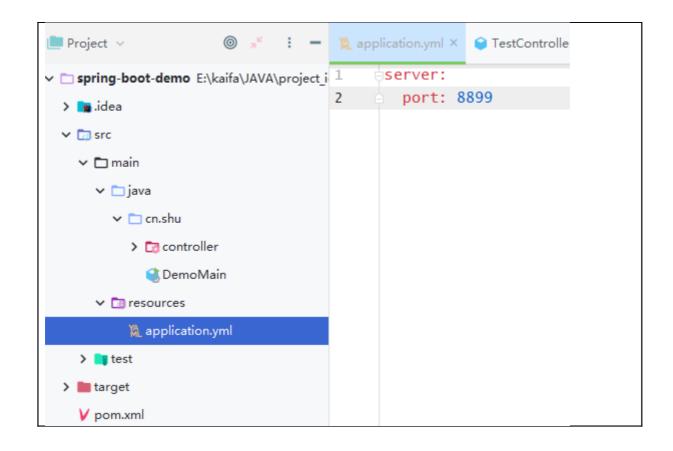
1 application.properties 类型

配置端口,注意与 yml 类型区别



2 application.yml 类型

配置端口,注意与 properties 类型区别一级添加一个空格,再下一级添加二个空格



五 Spring Boot 整合 Mybatis

作为 springboot 框架,为持久层 mybatis 整合准备自动配置的逻辑,其中 dataSource 这个对象不需要我们自行配置 bean 标签,bean 注解等内容,是通过 DataSourceAutoConfiguration 的配置类实现自动配置.我们只需要给它准备必要的四个属性,提供 mybatis 的依赖就能是实现持久层整合 Mybatis 不属于 spring 的一员,使用配置稍微复杂一些

1 导入相关依赖

2 配置 MyBatis(spring-boot 的配置文件中)

```
server:
  port: 8099
spring:
  datasource:
    #数据源类型
 type: com.alibaba.druid.pool.DruidDataSource
    url: jdbc:mysql:///microtest
    username: root
    password: admin
    driver-class-name: com.mysql.jdbc.Driver
#mvbatis
mybatis:
  #mapper文件
mapper-locations: classpath:mappers/*.xml
  configuration:
    #关闭二级缓存
 cache-enabled: false
    #开启驼峰命名
 map-underscore-to-camel-case: true
```

3 配置 mapper 扫描包

在我们的入口类中,添加@MapperScan 注解,指向我们的 maper 接口所在包即可

六 打包独立运行(jar)

工程的静态资源需要放到 resource->static 下,没有 web-inf 作为一个 web 应用,使用 springboot 开发完毕之后,无需按照 war 包形式打包放到第三方 web 容器运行.只需要在一个 JDK 环境下,运行 java 命令,来使当前项目 jar 包生效。

1 添加打包插件(pom.xml)

2 可指定打包名称

3 运行

JDK 环境运行:

java -jar jar 包名称