<http://www.cnblogs.com/suncj/p/4065589.html>

最近公司运用springboot构建项目，确实比ssh搭建要快很多。[springboot官方学习网站](http://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/#_working_with_spring_boot)

1.首先要下载maven，用maven管理项目很方便，下载完maven配置好环境，maven我就不细说了。

2.创建一个maven项目，pom.xml文件里面写这些:

[复制代码](javascript:void(0);)

<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>

<groupId>springboot</groupId>

<artifactId>testSpringBoot</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>testSpringBoot</name>

<packaging>jar</packaging>

<!-- 继承父包 -->

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>1.1.3.RELEASE</version>

<relativePath></relativePath>

</parent>

<!-- spring-boot的web启动的jar包 -->

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!--jpa的jar包 ，操作数据库的，类似hibernate-->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<!--mysql驱动-->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

<!--thymeleaf模板jar，是很不错的html数据传递取值，类似jsp的jstl-->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

</dependencies>

<!--maven的插件-->

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

<!-- 配置java版本 不配置的话默认父类配置的是1.6-->

<pluginManagement>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>1.7</source>

<target>1.7</target>

</configuration>

</plugin>

</plugins>

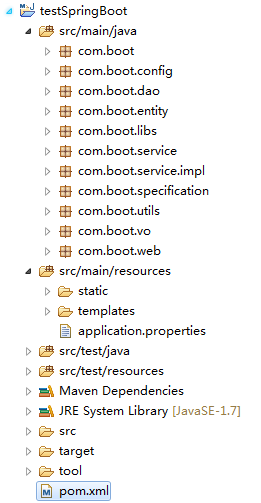
</pluginManagement>

</build>

</project>

[复制代码](javascript:void(0);)

3.创建的文件目录如图:



4.在com.boot(即最外层目录文件)下写一个如下main方法:

[复制代码](javascript:void(0);)

package com.boot;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.EnableAutoConfiguration;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

@Configuration

@EnableAutoConfiguration

@ComponentScan

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

[复制代码](javascript:void(0);)

5.在com.boot.web下创建一个类如下:

[复制代码](javascript:void(0);)

package com.boot.web;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

@Controller

public class MainController {

// @RequestMapping("")

// public String index(){

// return "examples/index";

// }

@RequestMapping("/")

@ResponseBody

String home() {

return "Hello World!";

}

}

[复制代码](javascript:void(0);)

@RequestMapping     @ResponseBody

这两个注解都是springMVC的,不懂得可以看springMVC

6.在resources下增加一个application.properties文件

文件内容如下配置:

[复制代码](javascript:void(0);)

spring.datasource.platform=mysql

spring.datasource.url=jdbc:mysql://localhost/springboot?useUnicode=true&characterEncoding=utf-8&zeroDateTimeBehavior=convertToNull&transformedBitIsBoolean=true&autoReconnect=true&failOverReadOnly=false

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.driverClassName=com.mysql.jdbc.Driver

# Advanced configuration...

spring.datasource.max-active=50

spring.datasource.max-idle=6

spring.datasource.min-idle=2

spring.datasource.initial-size=6

#create table

spring.jpa.hibernate.ddl-auto=validate

server.port=8080

server.session-timeout=30

server.tomcat.uri-encoding=UTF-8

spring.thymeleaf.prefix=classpath:templates/

spring.thymeleaf.suffix=.html

spring.thymeleaf.mode=HTML5

spring.thymeleaf.encoding=UTF-8

spring.thymeleaf.content-type=text/html

spring.thymeleaf.cache=false

[复制代码](javascript:void(0);)

该文件的配置可以参考[springboot的官网](http://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/#_working_with_spring_boot)

该文件的全部参数配置；如下(摘自官网)

[复制代码](javascript:void(0);)

# ===================================================================

# COMMON SPRING BOOT PROPERTIES

#

# This sample file is provided as a guideline. Do NOT copy it in its

# entirety to your own application. ^^^

# ===================================================================

# ----------------------------------------

# CORE PROPERTIES

# ----------------------------------------

# SPRING CONFIG (ConfigFileApplicationListener)

spring.config.name= # config file name (default to 'application')

spring.config.location= # location of config file

# PROFILES

spring.profiles.active= # comma list of active profiles

# APPLICATION SETTINGS (SpringApplication)

spring.main.sources=

spring.main.web-environment= # detect by default

spring.main.show-banner=true

spring.main....= # see class for all properties

# LOGGING

logging.path=/var/logs

logging.file=myapp.log

logging.config= # location of config file (default classpath:logback.xml for logback)

logging.level.\*= # levels for loggers, e.g. "logging.level.org.springframework=DEBUG" (TRACE, DEBUG, INFO, WARN, ERROR, FATAL, OFF)

# IDENTITY (ContextIdApplicationContextInitializer)

spring.application.name=

spring.application.index=

# EMBEDDED SERVER CONFIGURATION (ServerProperties)

server.port=8080

server.address= # bind to a specific NIC

server.session-timeout= # session timeout in seconds

server.context-path= # the context path, defaults to '/'

server.servlet-path= # the servlet path, defaults to '/'

server.ssl.client-auth= # want or need

server.ssl.key-alias=

server.ssl.key-password=

server.ssl.key-store=

server.ssl.key-store-password=

server.ssl.key-store-provider=

server.ssl.key-store-type=

server.ssl.protocol=TLS

server.ssl.trust-store=

server.ssl.trust-store-password=

server.ssl.trust-store-provider=

server.ssl.trust-store-type=

server.tomcat.access-log-pattern= # log pattern of the access log

server.tomcat.access-log-enabled=false # is access logging enabled

server.tomcat.internal-proxies=10\.\d{1,3}\.\d{1,3}\.\d{1,3}|\

192\.168\.\d{1,3}\.\d{1,3}|\

169\.254\.\d{1,3}\.\d{1,3}|\

127\.\d{1,3}\.\d{1,3}\.\d{1,3} # regular expression matching trusted IP addresses

server.tomcat.protocol-header=x-forwarded-proto # front end proxy forward header

server.tomcat.port-header= # front end proxy port header

server.tomcat.remote-ip-header=x-forwarded-for

server.tomcat.basedir=/tmp # base dir (usually not needed, defaults to tmp)

server.tomcat.background-processor-delay=30; # in seconds

server.tomcat.max-threads = 0 # number of threads in protocol handler

server.tomcat.uri-encoding = UTF-8 # character encoding to use for URL decoding

# SPRING MVC (HttpMapperProperties)

http.mappers.json-pretty-print=false # pretty print JSON

http.mappers.json-sort-keys=false # sort keys

spring.mvc.locale= # set fixed locale, e.g. en\_UK

spring.mvc.date-format= # set fixed date format, e.g. dd/MM/yyyy

spring.mvc.message-codes-resolver-format= # PREFIX\_ERROR\_CODE / POSTFIX\_ERROR\_CODE

spring.view.prefix= # MVC view prefix

spring.view.suffix= # ... and suffix

spring.resources.cache-period= # cache timeouts in headers sent to browser

spring.resources.add-mappings=true # if default mappings should be added

# JACKSON (JacksonProperties)

spring.jackson.date-format= # Date format string (e.g. yyyy-MM-dd HH:mm:ss), or a fully-qualified date format class name (e.g. com.fasterxml.jackson.databind.util.ISO8601DateFormat)

spring.jackson.property-naming-strategy= # One of the constants on Jackson's PropertyNamingStrategy (e.g. CAMEL\_CASE\_TO\_LOWER\_CASE\_WITH\_UNDERSCORES) or the fully-qualified class name of a PropertyNamingStrategy subclass

spring.jackson.deserialization.\*= # see Jackson's DeserializationFeature

spring.jackson.generator.\*= # see Jackson's JsonGenerator.Feature

spring.jackson.mapper.\*= # see Jackson's MapperFeature

spring.jackson.parser.\*= # see Jackson's JsonParser.Feature

spring.jackson.serialization.\*= # see Jackson's SerializationFeature

# THYMELEAF (ThymeleafAutoConfiguration)

spring.thymeleaf.prefix=classpath:/templates/

spring.thymeleaf.suffix=.html

spring.thymeleaf.mode=HTML5

spring.thymeleaf.encoding=UTF-8

spring.thymeleaf.content-type=text/html # ;charset=<encoding> is added

spring.thymeleaf.cache=true # set to false for hot refresh

# FREEMARKER (FreeMarkerAutoConfiguration)

spring.freemarker.allowRequestOverride=false

spring.freemarker.cache=true

spring.freemarker.checkTemplateLocation=true

spring.freemarker.charSet=UTF-8

spring.freemarker.contentType=text/html

spring.freemarker.exposeRequestAttributes=false

spring.freemarker.exposeSessionAttributes=false

spring.freemarker.exposeSpringMacroHelpers=false

spring.freemarker.prefix=

spring.freemarker.requestContextAttribute=

spring.freemarker.settings.\*=

spring.freemarker.suffix=.ftl

spring.freemarker.templateLoaderPath=classpath:/templates/ # comma-separated list

spring.freemarker.viewNames= # whitelist of view names that can be resolved

# GROOVY TEMPLATES (GroovyTemplateAutoConfiguration)

spring.groovy.template.cache=true

spring.groovy.template.charSet=UTF-8

spring.groovy.template.configuration.\*= # See Groovy's TemplateConfiguration

spring.groovy.template.contentType=text/html

spring.groovy.template.prefix=classpath:/templates/

spring.groovy.template.suffix=.tpl

spring.groovy.template.viewNames= # whitelist of view names that can be resolved

# VELOCITY TEMPLATES (VelocityAutoConfiguration)

spring.velocity.allowRequestOverride=false

spring.velocity.cache=true

spring.velocity.checkTemplateLocation=true

spring.velocity.charSet=UTF-8

spring.velocity.contentType=text/html

spring.velocity.dateToolAttribute=

spring.velocity.exposeRequestAttributes=false

spring.velocity.exposeSessionAttributes=false

spring.velocity.exposeSpringMacroHelpers=false

spring.velocity.numberToolAttribute=

spring.velocity.prefix=

spring.velocity.properties.\*=

spring.velocity.requestContextAttribute=

spring.velocity.resourceLoaderPath=classpath:/templates/

spring.velocity.suffix=.vm

spring.velocity.viewNames= # whitelist of view names that can be resolved

# INTERNATIONALIZATION (MessageSourceAutoConfiguration)

spring.messages.basename=messages

spring.messages.cacheSeconds=-1

spring.messages.encoding=UTF-8

# SECURITY (SecurityProperties)

security.user.name=user # login username

security.user.password= # login password

security.user.role=USER # role assigned to the user

security.require-ssl=false # advanced settings ...

security.enable-csrf=false

security.basic.enabled=true

security.basic.realm=Spring

security.basic.path= # /\*\*

security.filter-order=0

security.headers.xss=false

security.headers.cache=false

security.headers.frame=false

security.headers.contentType=false

security.headers.hsts=all # none / domain / all

security.sessions=stateless # always / never / if\_required / stateless

security.ignored=false

# DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)

spring.datasource.name= # name of the data source

spring.datasource.initialize=true # populate using data.sql

spring.datasource.schema= # a schema (DDL) script resource reference

spring.datasource.data= # a data (DML) script resource reference

spring.datasource.sqlScriptEncoding= # a charset for reading SQL scripts

spring.datasource.platform= # the platform to use in the schema resource (schema-${platform}.sql)

spring.datasource.continueOnError=false # continue even if can't be initialized

spring.datasource.separator=; # statement separator in SQL initialization scripts

spring.datasource.driver-class-name= # JDBC Settings...

spring.datasource.url=

spring.datasource.username=

spring.datasource.password=

spring.datasource.jndi-name # For JNDI lookup (class, url, username & password are ignored when set)

spring.datasource.max-active=100 # Advanced configuration...

spring.datasource.max-idle=8

spring.datasource.min-idle=8

spring.datasource.initial-size=10

spring.datasource.validation-query=

spring.datasource.test-on-borrow=false

spring.datasource.test-on-return=false

spring.datasource.test-while-idle=

spring.datasource.time-between-eviction-runs-millis=

spring.datasource.min-evictable-idle-time-millis=

spring.datasource.max-wait=

# MONGODB (MongoProperties)

spring.data.mongodb.host= # the db host

spring.data.mongodb.port=27017 # the connection port (defaults to 27107)

spring.data.mongodb.uri=mongodb://localhost/test # connection URL

spring.data.mongo.repositories.enabled=true # if spring data repository support is enabled

# JPA (JpaBaseConfiguration, HibernateJpaAutoConfiguration)

spring.jpa.properties.\*= # properties to set on the JPA connection

spring.jpa.openInView=true

spring.jpa.show-sql=true

spring.jpa.database-platform=

spring.jpa.database=

spring.jpa.generate-ddl=false # ignored by Hibernate, might be useful for other vendors

spring.jpa.hibernate.naming-strategy= # naming classname

spring.jpa.hibernate.ddl-auto= # defaults to create-drop for embedded dbs

spring.data.jpa.repositories.enabled=true # if spring data repository support is enabled

# JTA (JtaAutoConfiguration)

spring.jta.log-dir= # transaction log dir

spring.jta.\*= # technology specific configuration

# SOLR (SolrProperties})

spring.data.solr.host=http://127.0.0.1:8983/solr

spring.data.solr.zkHost=

spring.data.solr.repositories.enabled=true # if spring data repository support is enabled

# ELASTICSEARCH (ElasticsearchProperties})

spring.data.elasticsearch.cluster-name= # The cluster name (defaults to elasticsearch)

spring.data.elasticsearch.cluster-nodes= # The address(es) of the server node (comma-separated; if not specified starts a client node)

spring.data.elasticsearch.repositories.enabled=true # if spring data repository support is enabled

# DATA RESET (RepositoryRestConfiguration})

spring.data.rest.baseUri=foo # base URI against which the exporter should calculate its links

# FLYWAY (FlywayProperties)

flyway.locations=classpath:db/migrations # locations of migrations scripts

flyway.schemas= # schemas to update

flyway.initVersion= 1 # version to start migration

flyway.sql-migration-prefix=V

flyway.sql-migration-suffix=.sql

flyway.enabled=true

flyway.url= # JDBC url if you want Flyway to create its own DataSource

flyway.user= # JDBC username if you want Flyway to create its own DataSource

flyway.password= # JDBC password if you want Flyway to create its own DataSource

# LIQUIBASE (LiquibaseProperties)

liquibase.change-log=classpath:/db/changelog/db.changelog-master.yaml

liquibase.contexts= # runtime contexts to use

liquibase.default-schema= # default database schema to use

liquibase.drop-first=false

liquibase.enabled=true

liquibase.url= # specific JDBC url (if not set the default datasource is used)

liquibase.user= # user name for liquibase.url

liquibase.password= # password for liquibase.url

# JMX

spring.jmx.enabled=true # Expose MBeans from Spring

# RABBIT (RabbitProperties)

spring.rabbitmq.host= # connection host

spring.rabbitmq.port= # connection port

spring.rabbitmq.addresses= # connection addresses (e.g. myhost:9999,otherhost:1111)

spring.rabbitmq.username= # login user

spring.rabbitmq.password= # login password

spring.rabbitmq.virtualHost=

spring.rabbitmq.dynamic=

# REDIS (RedisProperties)

spring.redis.host=localhost # server host

spring.redis.password= # server password

spring.redis.port=6379 # connection port

spring.redis.pool.max-idle=8 # pool settings ...

spring.redis.pool.min-idle=0

spring.redis.pool.max-active=8

spring.redis.pool.max-wait=-1

spring.redis.sentinel.master= # name of Redis server

spring.redis.sentinel.nodes= # comma-separated list of host:port pairs

# ACTIVEMQ (ActiveMQProperties)

spring.activemq.broker-url=tcp://localhost:61616 # connection URL

spring.activemq.user=

spring.activemq.password=

spring.activemq.in-memory=true # broker kind to create if no broker-url is specified

spring.activemq.pooled=false

# HornetQ (HornetQProperties)

spring.hornetq.mode= # connection mode (native, embedded)

spring.hornetq.host=localhost # hornetQ host (native mode)

spring.hornetq.port=5445 # hornetQ port (native mode)

spring.hornetq.embedded.enabled=true # if the embedded server is enabled (needs hornetq-jms-server.jar)

spring.hornetq.embedded.serverId= # auto-generated id of the embedded server (integer)

spring.hornetq.embedded.persistent=false # message persistence

spring.hornetq.embedded.data-directory= # location of data content (when persistence is enabled)

spring.hornetq.embedded.queues= # comma-separated queues to create on startup

spring.hornetq.embedded.topics= # comma-separated topics to create on startup

spring.hornetq.embedded.cluster-password= # customer password (randomly generated by default)

# JMS (JmsProperties)

spring.datasource.jndi-name= # JNDI location of a JMS ConnectionFactory

spring.jms.pub-sub-domain= # false for queue (default), true for topic

# SPRING BATCH (BatchDatabaseInitializer)

spring.batch.job.names=job1,job2

spring.batch.job.enabled=true

spring.batch.initializer.enabled=true

spring.batch.schema= # batch schema to load

# AOP

spring.aop.auto=

spring.aop.proxy-target-class=

# FILE ENCODING (FileEncodingApplicationListener)

spring.mandatory-file-encoding=false

# SPRING SOCIAL (SocialWebAutoConfiguration)

spring.social.auto-connection-views=true # Set to true for default connection views or false if you provide your own

# SPRING SOCIAL FACEBOOK (FacebookAutoConfiguration)

spring.social.facebook.app-id= # your application's Facebook App ID

spring.social.facebook.app-secret= # your application's Facebook App Secret

# SPRING SOCIAL LINKEDIN (LinkedInAutoConfiguration)

spring.social.linkedin.app-id= # your application's LinkedIn App ID

spring.social.linkedin.app-secret= # your application's LinkedIn App Secret

# SPRING SOCIAL TWITTER (TwitterAutoConfiguration)

spring.social.twitter.app-id= # your application's Twitter App ID

spring.social.twitter.app-secret= # your application's Twitter App Secret

# SPRING MOBILE SITE PREFERENCE (SitePreferenceAutoConfiguration)

spring.mobile.sitepreference.enabled=true # enabled by default

# SPRING MOBILE DEVICE VIEWS (DeviceDelegatingViewResolverAutoConfiguration)

spring.mobile.devicedelegatingviewresolver.enabled=true # disabled by default

spring.mobile.devicedelegatingviewresolver.normalPrefix=

spring.mobile.devicedelegatingviewresolver.normalSuffix=

spring.mobile.devicedelegatingviewresolver.mobilePrefix=mobile/

spring.mobile.devicedelegatingviewresolver.mobileSuffix=

spring.mobile.devicedelegatingviewresolver.tabletPrefix=tablet/

spring.mobile.devicedelegatingviewresolver.tabletSuffix=

# ----------------------------------------

# ACTUATOR PROPERTIES

# ----------------------------------------

# MANAGEMENT HTTP SERVER (ManagementServerProperties)

management.port= # defaults to 'server.port'

management.address= # bind to a specific NIC

management.contextPath= # default to '/'

management.add-application-context-header= # default to true

# ENDPOINTS (AbstractEndpoint subclasses)

endpoints.autoconfig.id=autoconfig

endpoints.autoconfig.sensitive=true

endpoints.autoconfig.enabled=true

endpoints.beans.id=beans

endpoints.beans.sensitive=true

endpoints.beans.enabled=true

endpoints.configprops.id=configprops

endpoints.configprops.sensitive=true

endpoints.configprops.enabled=true

endpoints.configprops.keys-to-sanitize=password,secret,key # suffix or regex

endpoints.dump.id=dump

endpoints.dump.sensitive=true

endpoints.dump.enabled=true

endpoints.env.id=env

endpoints.env.sensitive=true

endpoints.env.enabled=true

endpoints.env.keys-to-sanitize=password,secret,key # suffix or regex

endpoints.health.id=health

endpoints.health.sensitive=false

endpoints.health.enabled=true

endpoints.health.time-to-live=1000

endpoints.info.id=info

endpoints.info.sensitive=false

endpoints.info.enabled=true

endpoints.metrics.id=metrics

endpoints.metrics.sensitive=true

endpoints.metrics.enabled=true

endpoints.shutdown.id=shutdown

endpoints.shutdown.sensitive=true

endpoints.shutdown.enabled=false

endpoints.trace.id=trace

endpoints.trace.sensitive=true

endpoints.trace.enabled=true

# HEALTH INDICATORS

health.diskspace.path=.

health.diskspace.threshold=10485760

# MVC ONLY ENDPOINTS

endpoints.jolokia.path=jolokia

endpoints.jolokia.sensitive=true

endpoints.jolokia.enabled=true # when using Jolokia

# JMX ENDPOINT (EndpointMBeanExportProperties)

endpoints.jmx.enabled=true

endpoints.jmx.domain= # the JMX domain, defaults to 'org.springboot'

endpoints.jmx.unique-names=false

endpoints.jmx.staticNames=

# JOLOKIA (JolokiaProperties)

jolokia.config.\*= # See Jolokia manual

# REMOTE SHELL

shell.auth=simple # jaas, key, simple, spring

shell.command-refresh-interval=-1

shell.command-path-patterns= # classpath\*:/commands/\*\*, classpath\*:/crash/commands/\*\*

shell.config-path-patterns= # classpath\*:/crash/\*

shell.disabled-plugins=false # don't expose plugins

shell.ssh.enabled= # ssh settings ...

shell.ssh.keyPath=

shell.ssh.port=

shell.telnet.enabled= # telnet settings ...

shell.telnet.port=

shell.auth.jaas.domain= # authentication settings ...

shell.auth.key.path=

shell.auth.simple.user.name=

shell.auth.simple.user.password=

shell.auth.spring.roles=

# GIT INFO

spring.git.properties= # resource ref to generated git info properties file

[复制代码](javascript:void(0);)

7.找到com.boot下的Application以java Application方式启动，然后打开浏览器输入localhost:8080就会出现Hello World!

这样一个简单的web开发就搭建好了。

8.启动图片:



分类: [springboot](http://www.cnblogs.com/suncj/category/620148.html)