Spring Boot Basic

简单的用户登录注册

张洪胤

项目Github地址: https://github.com/stormbroken/Spring-Basic

操作环境

使用软件：

IntelliJ IDEA 2020.3.3
Navicat Premium 12
Postman
Xshell、Xftp(Windows连接服务器)

操作环境：

Windows 10
Java 1.8.0_251(最好是Java 8)
Maven 3.6.3
MySQL 5.7(大版本一定要是5)
Docker 20.10.5(服务器部署使用)

服务器配置：

搭建服务器(写的比较早，谨慎使用)
建议使用Docker的方式进行部署配置

下载指路：

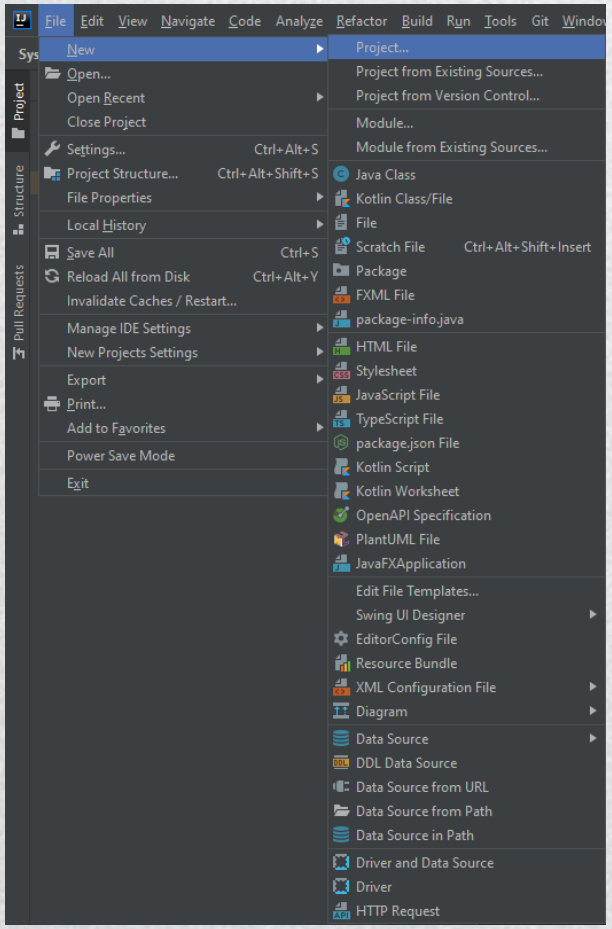
https://www.jetbrains.com/idea/
https://www.navicat.com.cn/products
https://www.netsarang.com/en/xshell/
https://www.netsarang.com/en/xftp/
https://www.java.com/zh-CN/download/
https://maven.apache.org/download.cgi
https://downloads.mysql.com/archives/community/
https://www.postman.com/downloads/

建议Maven换源：

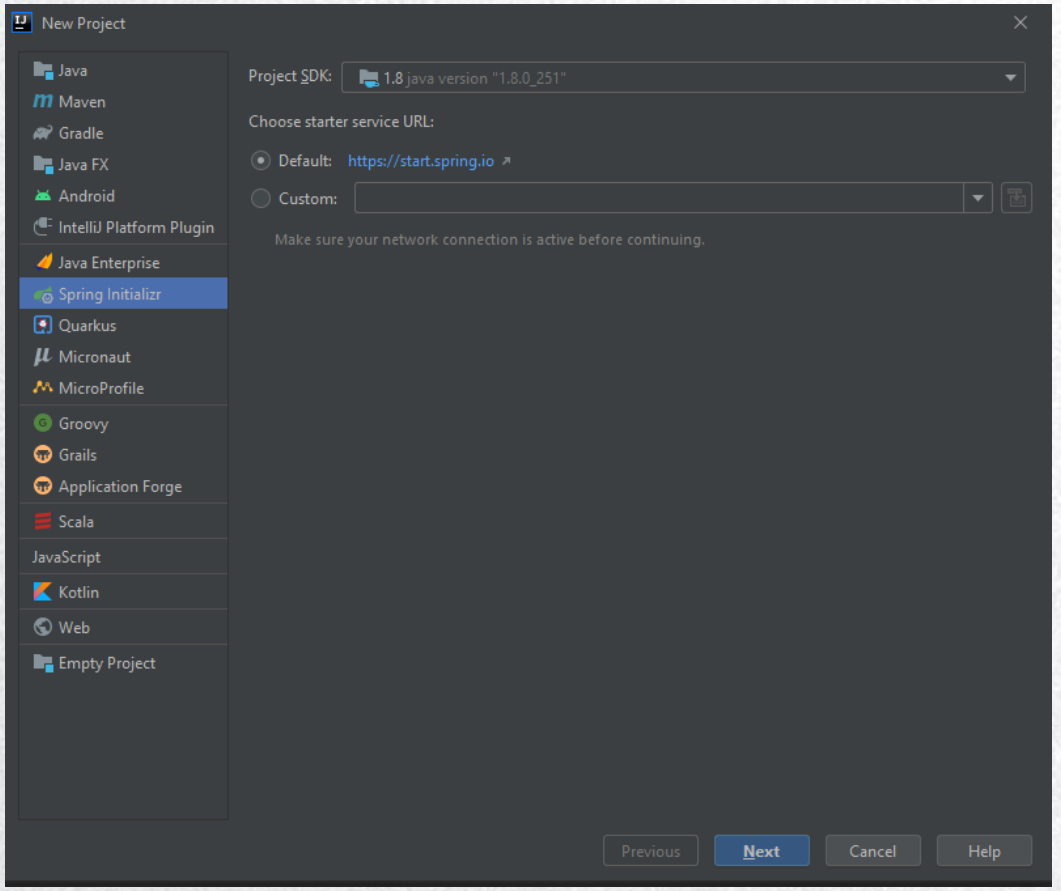
```
<mirror>
    <id>alimaven</id>
    <name>aliyun maven</name>
    <url>http://maven.aliyun.com/nexus/content/groups/public/</url>
    <mirrorOf>central</mirrorOf>
</mirror>
```

创建项目

选择创建Project:

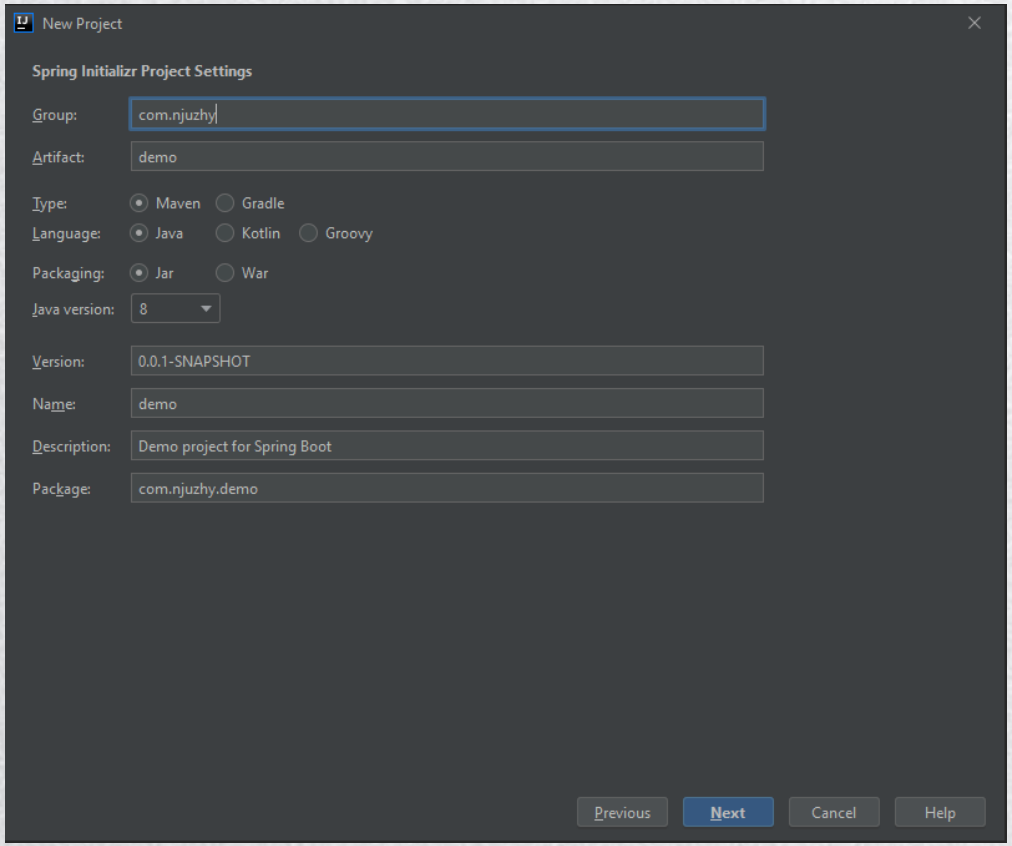


选择Spring Initializer后点击next:

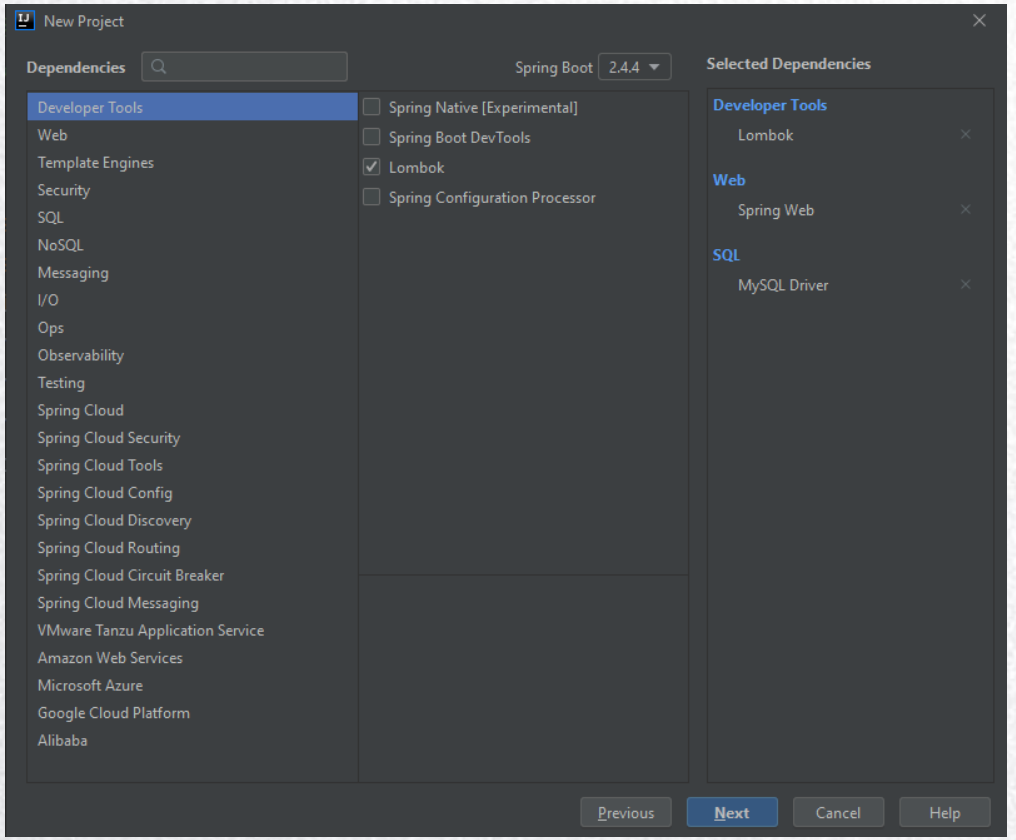


创建项目

编辑项目设置:

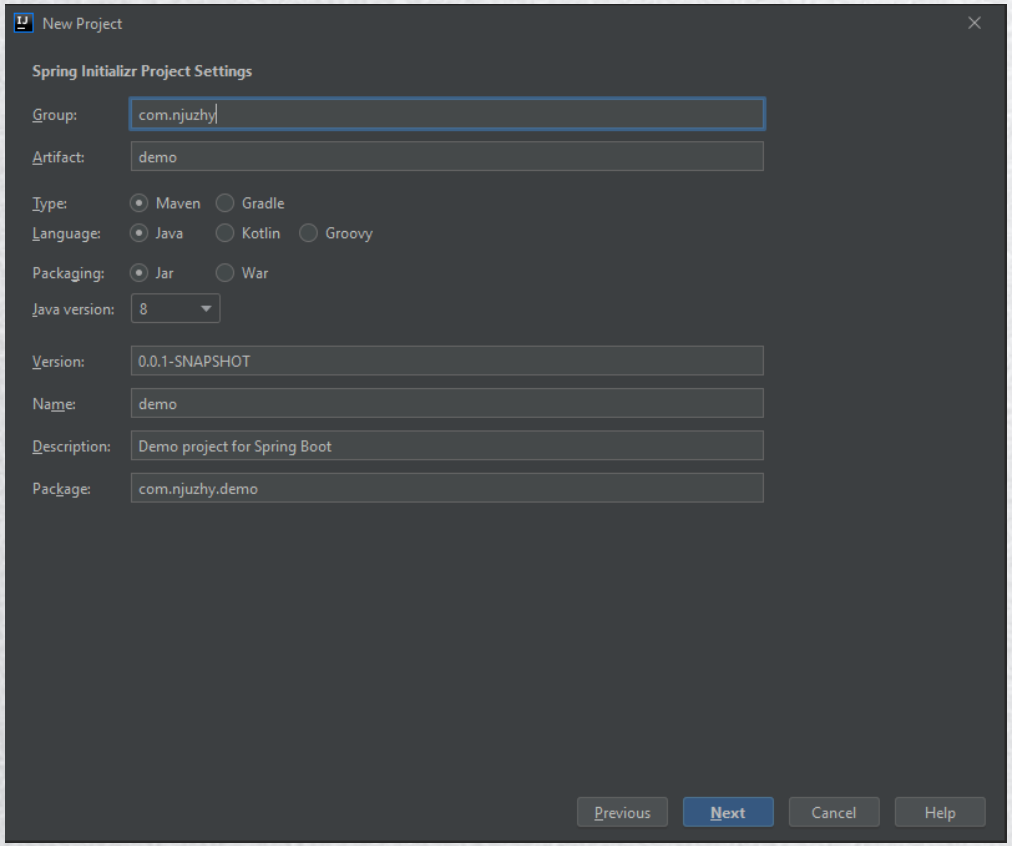


初始化项目依赖(之后选择存储位置即可):

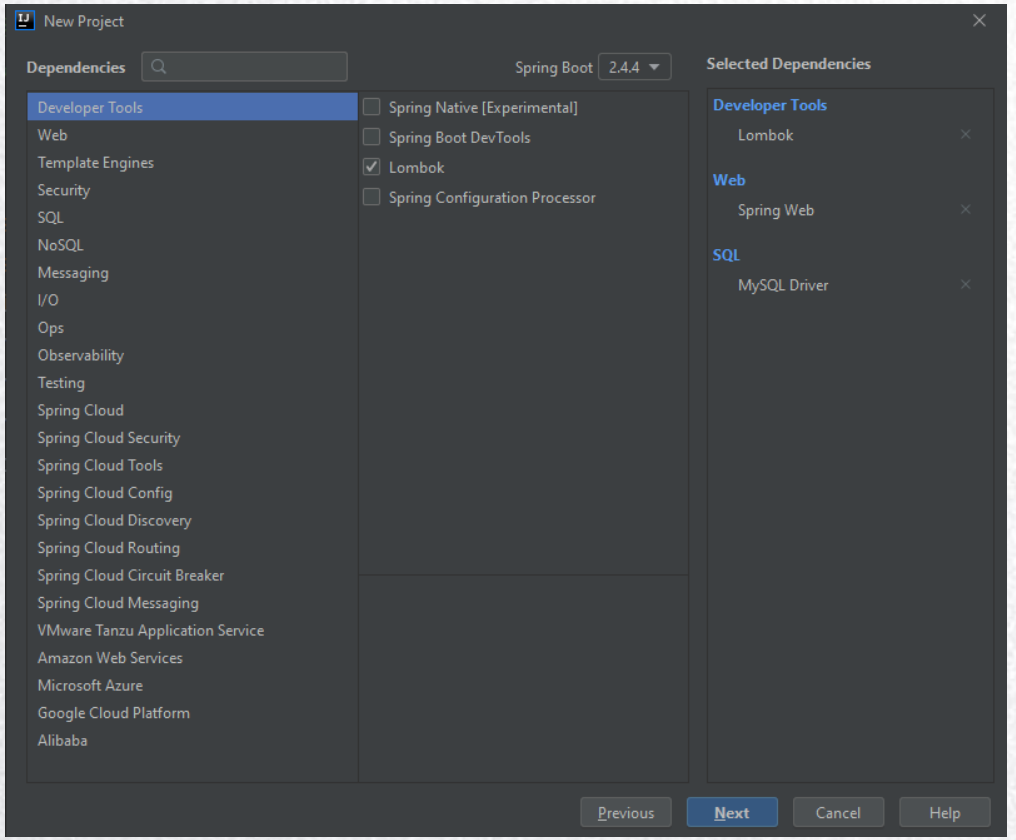


创建项目

编辑项目设置:

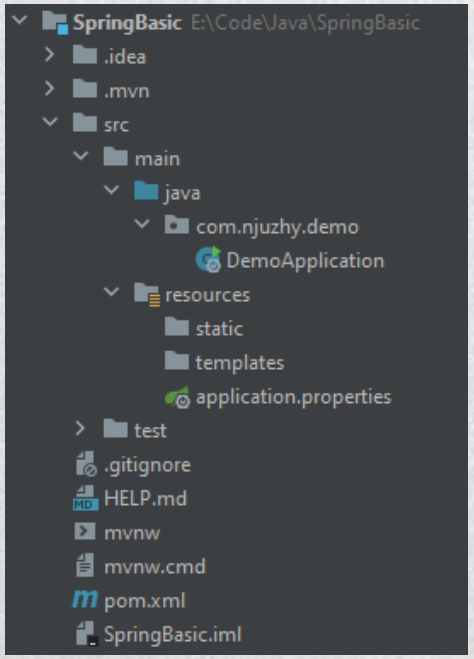


初始化项目依赖(之后选择存储位置即可):



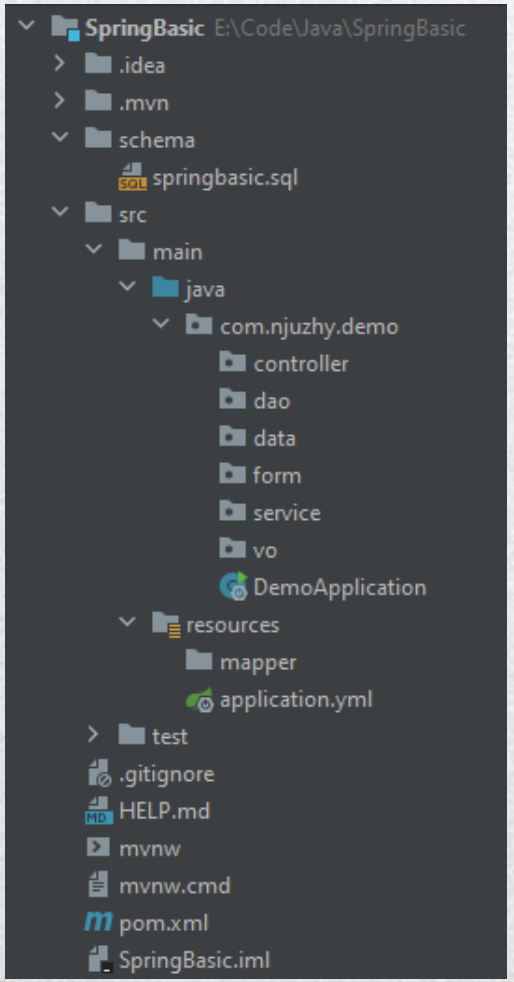
初始化项目——目录结构

创建后：



略作修改

修改后：



简要介绍目录结构：

schema:放置数据库脚本文件
src.main.java.com.njuzhy.demo:
　controller:本层主要放置了控制器
　service:本层主要放置了业务逻辑
　dao:本层主要是数据库接口
　data:本系统涉及到的数据实体
　form:本系统从前端接受到的格式
　vo:本系统返回给前端的格式
src.resources.mapper:放置xml配置文件
src.resources:application.yml是Spring配置文件

初始化项目——配置mybatis

补充mybatis配置(pom.xml中):

```xml
<!-- mybatis -->
<dependency>
    <groupId>org.mybatis.spring.boot</groupId>
    <artifactId>mybatis-spring-boot-starter</artifactId>
    <version>2.1.1</version>
</dependency>
```

项目的配置(application.yml中):

```yaml
server:
  port: 8080

spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/springbasic?serverTimezone=GMT%2B8&useUnicode=true&characterEncoding=utf-8
    username: root
    password: root #改为自己数据库的密码
    bySearch:
      testWhileIdle: true
      validationQuery: SELECT 1
      timeBestweenEvictionRunsMillis: 3600000 #每个小时确认连接是否可用

mybatis:
  mapper-locations: classpath:mapper/*.xml
  type-aliases-package: com.njuzhy.demo.data
```

server.port:运行后监听端口
datasource:配置本系统的数据库连接
mybatis:配置了实体位置和映射xml位置

初始化项目——初始化MySQL

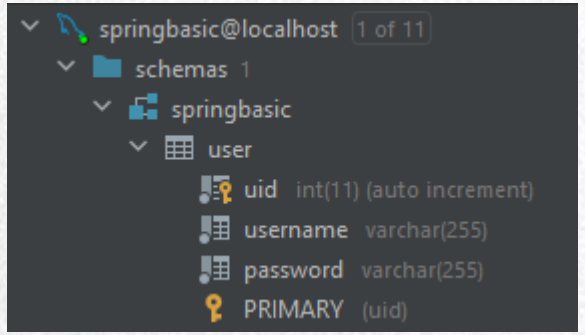
先创建springbasic数据库：

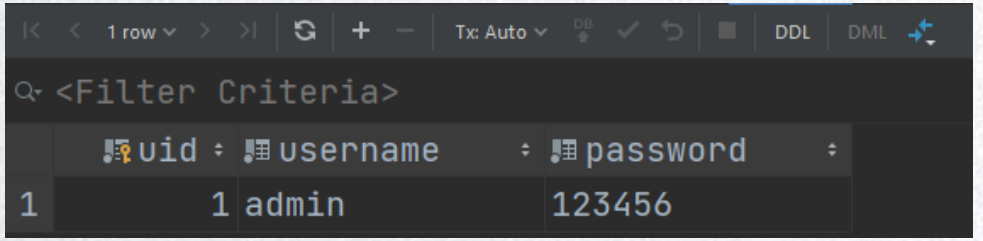
- 进入数据库命令行界面
- 在命令行执行：**create database springbasic**命令

在数据库中运行schema/springbasic.sql

```sql
SET NAMES utf8mb4;
SET FOREIGN_KEY_CHECKS = 0;

DROP DATABASE IF EXISTS springbasic;
CREATE DATABASE springbasic DEFAULT CHARACTER SET utf8;
USE springbasic;

-- ----------------------------
-- Table structure for user
-- ----------------------------
DROP TABLE IF EXISTS `user`;
CREATE TABLE `user`  (
  `uid` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NOT NULL,
  `password` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NOT NULL,
  PRIMARY KEY (`uid`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 2 CHARACTER SET = utf8mb4 COLLATE = utf8mb4_general_ci ROW_FORMAT = Dynamic;

INSERT INTO user values (1, "admin", "123456");

SET FOREIGN_KEY_CHECKS = 1;
```

查看结果：





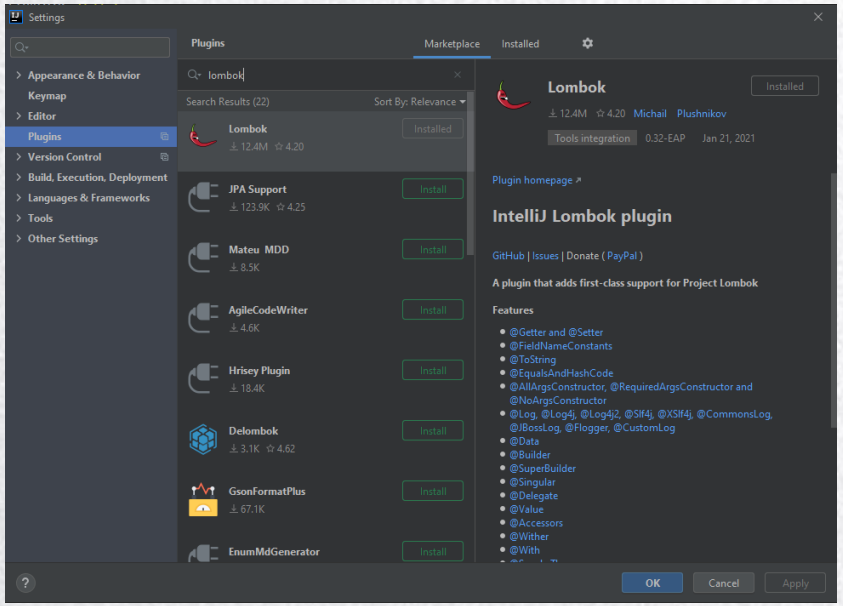
构建项目——配置data

配置User实体类：

```java
package com.njuzhy.demo.data;

import lombok.Data;

/**
 * @Author stormbroken
 * Create by 2021/04/05
 * @Version 1.0
 **/

@Data
public class User {
    private Integer uid;
    private String username;
    private String password;
}
```

@Data:

- @Data是Lombok插件的一部分，但是lombok确实有侵入性，结合自己情况使用。
- Lombok在IDEA中使用需要安装Plugin
  - 进入File，找到settings
  - 选择plugin，在marketplace中搜索Lombok
  - 点击install即可



构建项目——配置dao层

创建UserDao接口：

```java
package com.njuzhy.demo.dao;

import com.njuzhy.demo.data.User;
import org.apache.ibatis.annotations.Mapper;

/**
 * @Author stormbroken
 * Create by 2021/04/05
 * @Version 1.0
 **/

@Mapper
public interface UserDao {
    /** 存储一个用户 ...*/
    boolean save(User user);

    /** 根据用户名查找用户 ...*/
    User getUserByUsername(String username);
}
```

@Mapper 是Mybatis框架的注解

配置UserMapper.xml(src.main.resources.mapper下)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.njuzhy.demo.dao.UserDao">
    <insert id="save" parameterType="com.njuzhy.demo.data.User" useGeneratedKeys="true" keyProperty="uid">
        insert into `user` (username, password) VALUES (#{username}, #{password})
    </insert>

    <select id="getUserByUsername" resultMap="User">
        select * from `user` where username = #{username}
    </select>


    <resultMap id="User" type="com.njuzhy.demo.data.User">
        <id column="uid" property="uid"></id>
        <result column="username" property="username"></result>
        <result column="password" property="password"></result>
    </resultMap>
</mapper>
```

Mybatis说明（个人编写，谨慎使用）

构建项目——配置service层

创建UserVO:

```java
@Data
public class UserVO {
    private Integer uid;
    private String username;
}
```

创建UserService接口:

```java
public interface UserService {
    /** 用户注册 ...*/
    boolean save(User user);

    /** 用户登录 ...*/
    UserVO login(String username, String password);
}
```

创建UserServiceImpl实现类:

```java
@Service
public class UserServiceImpl implements UserService{
    @Autowired
    UserDao userDao;

    /** 用户注册 ...*/
    @Override
    public boolean save(User user) {
        boolean result = false;
        try{
            result = userDao.save(user);
        }catch (Exception e){
            e.printStackTrace();
            return false;
        }
        return result;
    }

    /** 用户登录 ...*/
    @Override
    public UserVO login(String username, String password) {
        User user = userDao.getUserByUsername(username);
        if(user == null || !password.equals(user.getPassword())){
            return null;
        }
        UserVO userVO = new UserVO();
        BeanUtils.copyProperties(userVO, user);
        return userVO;
    }
}
```

Service说明（个人编写，谨慎使用）

构建项目——配置controller层

创建UserForm:

```java
@Data
public class UserForm {
    private String username;
    private String password;
}
```

创建LoginForm:

```java
@Data
public class LoginForm {
    private String username;
    private String password;
}
```

创建UserController:

```java
@RestController
@RequestMapping("/user")
public class UserController {
    @Autowired
    UserService userService;

    @PostMapping("/register")
    public boolean register(@RequestBody UserForm userForm){
        System.out.println(userForm.toString());
        User user = new User();
        BeanUtils.copyProperties(userForm, user);
        return userService.save(user);
    }

    @PostMapping("/login")
    public UserVO login(@RequestBody LoginForm loginForm){
        System.out.println(loginForm.toString());
        return userService.login(loginForm.getUsername(), loginForm.getPassword());
    }
}
```

Controller说明（个人编写，谨慎使用）

运行项目

在DemoApplication中运行

```java
package com.njuzhy.demo;

import ...

@SpringBootApplication
public class DemoApplication {

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

}
```

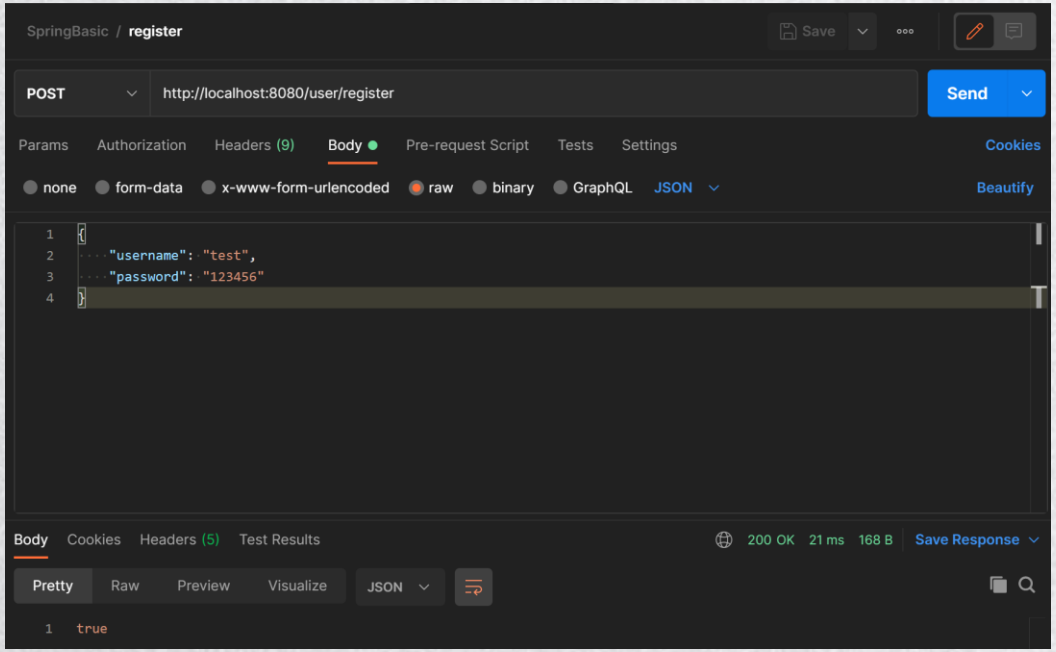
运行结果 (此时就可以请求了)

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.4.4)

2021-04-05 14:05:18.737  INFO 9624 --- [           main] com.njuzhy.demo.DemoApplication          : Starting DemoApplication using Java 1.8.0_251 on
2021-04-05 14:05:18.741  INFO 9624 --- [           main] com.njuzhy.demo.DemoApplication          : No active profile set, falling back to default pr
2021-04-05 14:05:19.791  INFO 9624 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2021-04-05 14:05:19.802  INFO 9624 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2021-04-05 14:05:19.802  INFO 9624 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.44]
2021-04-05 14:05:19.888  INFO 9624 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContex
2021-04-05 14:05:19.888  INFO 9624 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization comple
2021-04-05 14:05:20.277  INFO 9624 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExec
2021-04-05 14:05:20.475  INFO 9624 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with conte
2021-04-05 14:05:20.487  INFO 9624 --- [           main] com.njuzhy.demo.DemoApplication          : Started DemoApplication in 2.168 seconds (JVM run
```

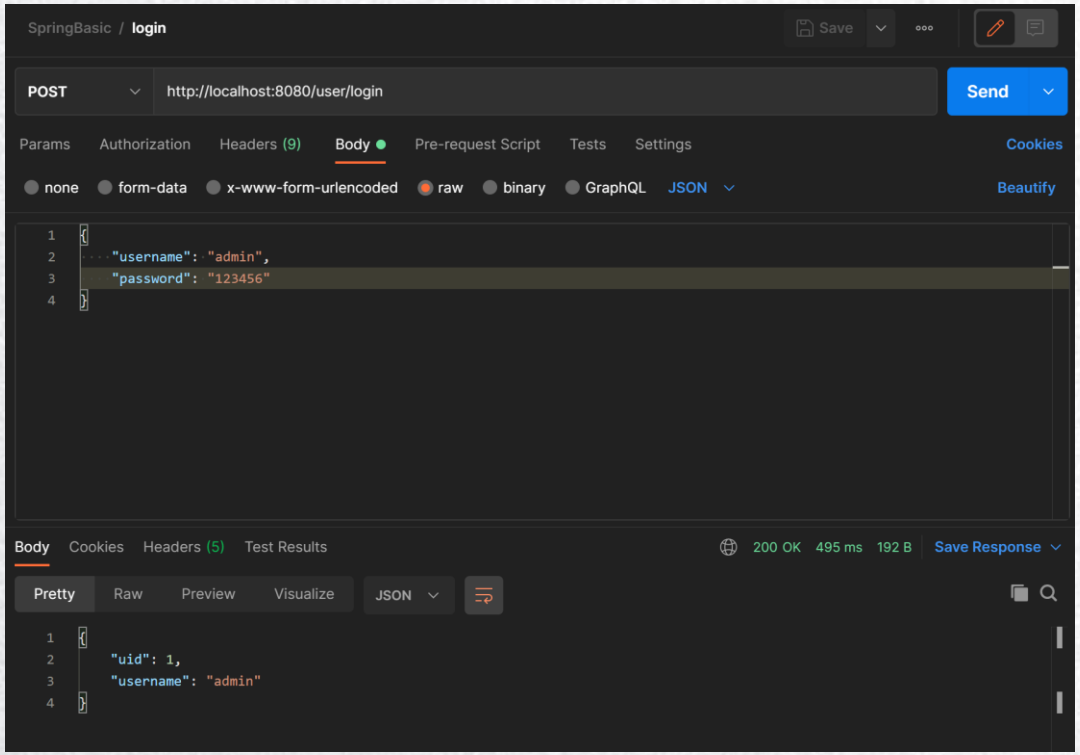
没有报错，并且能看到Started DemoApplication xxx则启动成功

Postman请求本项目API

请求/user/register

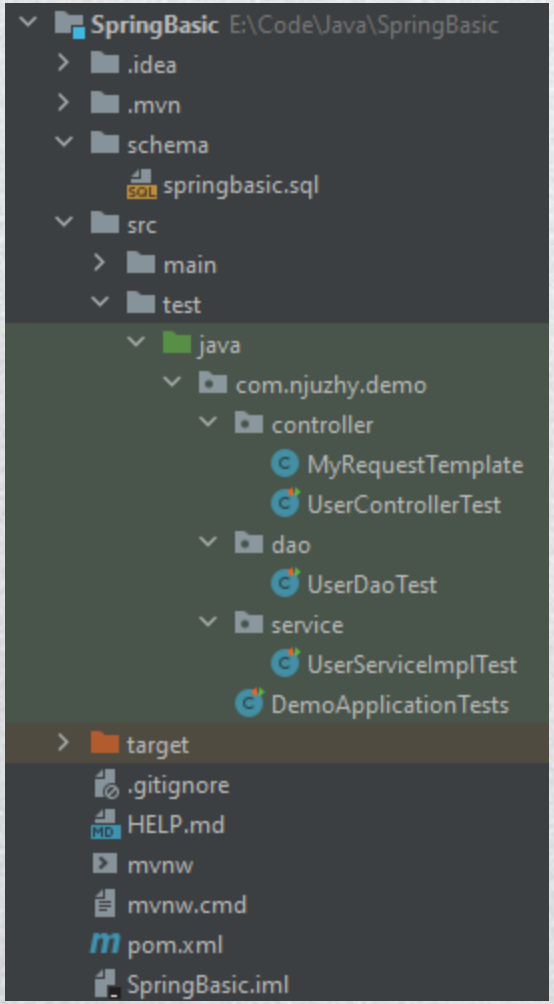


请求/user/login

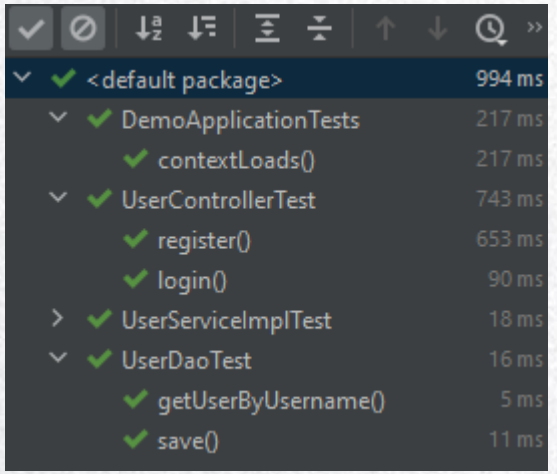


项目测试

测试目录：



测试结果：



测试工具：

- Junit 组件，个人编写资料
- MockMvc组件，推荐阅读

推荐阅读

到目前为止，仅仅讲述了登录注册功能，距离用好Spring Boot还有很大的差距，但是这部分工作难以一一描述，建议自己用好Spring官方文档、StackOverflow、CSDN和简书等等，自己按需学习使用。

- 我学习Spring Boot过程中留下的笔记：谨慎阅读
- 可能会比较使用的与微信服务器的交互：谨慎阅读
- 其他比较推荐了解的Spring Boot的内容：
  - Spring Boot的启动过程
  - Spring Boot的一个请求的分发过程（从Servlet开始）
  - Spring Boot的特性：IoC、DI、AOP
  - Spring Boot的Bean的生命周期和使用
  - Spring Boot的Java Config配置方式和XML配置方式
  - Spring Boot中的事务和其他的数据库框架（JDBC、JPA等等）

项目Github地址: https://github.com/stormbroken/Spring-Basic