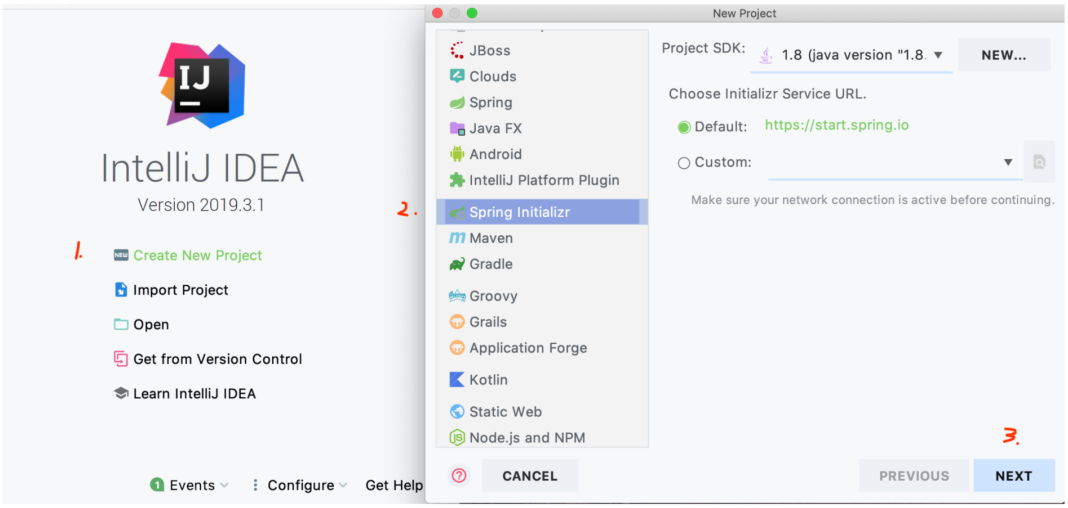
手把手带写项目代码

手把手带写项目

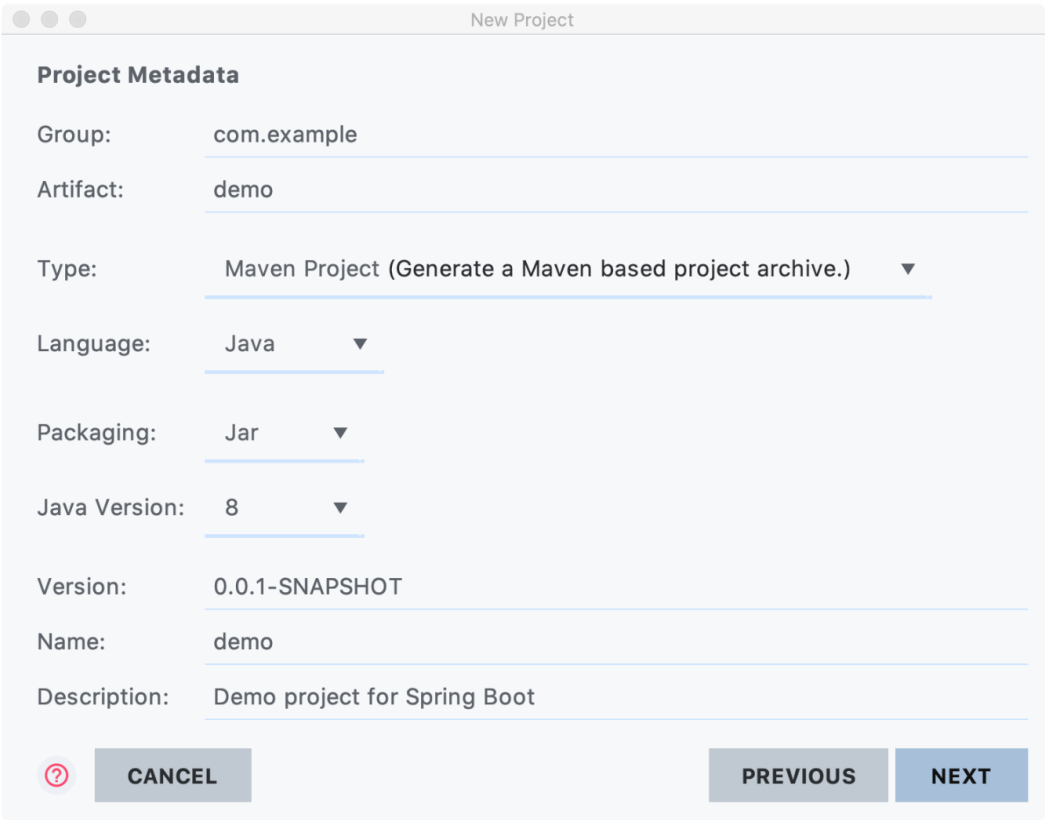
项目初始化

下载安装IDEA，使用 IDEA 中集成的 Spring Initializr 工具，初始化新项目

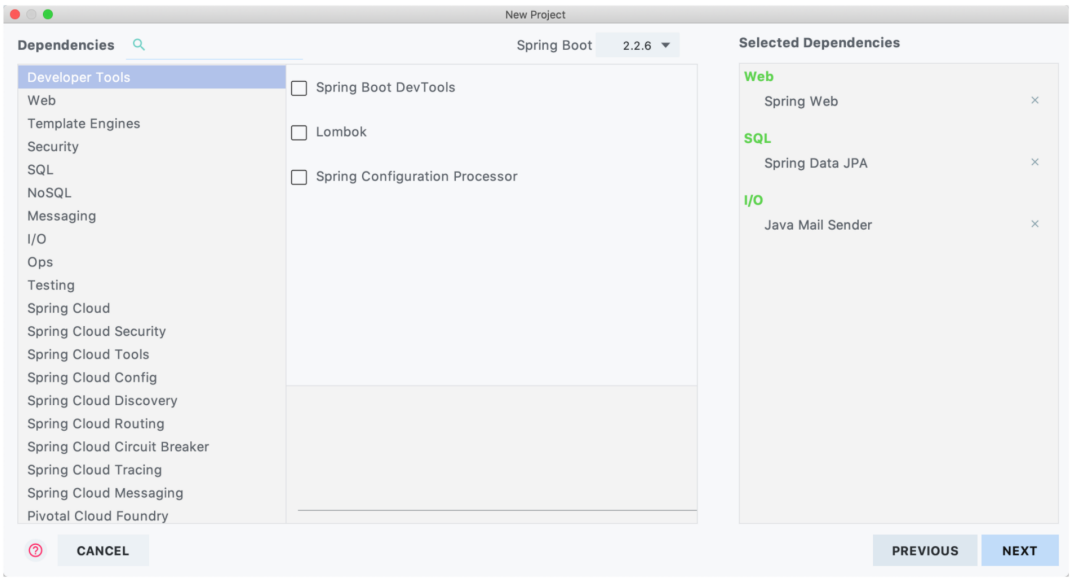
1. 新建项目



2. 项目信息初始化，默认即可



3. 选择 SpringBoot 集成的三个技术模块



4. 手动在pom.xml中添加其它依赖

```
<!-- 导入Mysql数据库链接jar包 -->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <!--
    <version>5.1.8</version>
    -->
  <version>8.0.19</version>
</dependency>
<!--   JWT认证    -->
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt</artifactId>
  <version>0.9.0</version>
</dependency>
<!--   工具包   -->
<dependency>
  <groupId>org.apache.commons</groupId>
  <artifactId>commons-lang3</artifactId>
</dependency>
```

最终pom.xml文件内容如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<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
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.2.6.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
```

```
    </parent>
    <groupId>com.example</groupId>
    <artifactId>demo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>demo</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-mail</artifactId>
        </dependency>
        <dependency>
              <groupId>org.springframework.boot</groupId>
              <artifactId>spring-boot-starter-test</artifactId>
              <scope>test</scope>
              <exclusions>
                  <exclusion>
                      <groupId>org.junit.vintage</groupId>
                      <artifactId>junit-vintage-engine</artifactId>
                  </exclusion>
              </exclusions>
          </dependency>

        <!-- 导入Mysql数据库链接jar包 -->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <!--
                        <version>5.1.8</version>
                        -->
            <version>8.0.19</version>
        </dependency>

        <!--   JWT认证     -->
        <dependency>
            <groupId>io.jsonwebtoken</groupId>
            <artifactId>jjwt</artifactId>
```

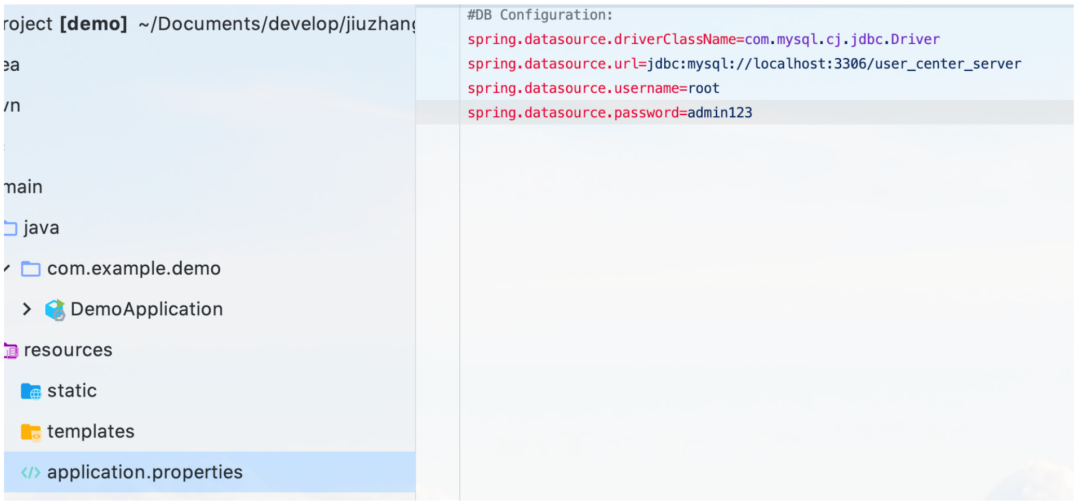
```
            <version>0.9.0</version>
        </dependency>
        <!--    工具包    -->
        <dependency>
            <groupId>org.apache.commons</groupId>
            <artifactId>commons-lang3</artifactId>
        </dependency>
    </dependencies>
    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
</project>
```

配置 MySQL 数据库

1. 安装见：
   a. Windows 系统安装 MySQL
   b. Mac OS 安装 MySQL
   c. MySQL 国内下载镜像
   d. 创建一个数据库，作为项目的数据库，之后声明到配置文件中
   e. 在项目配置文件 application.properties 中添加数据库信息

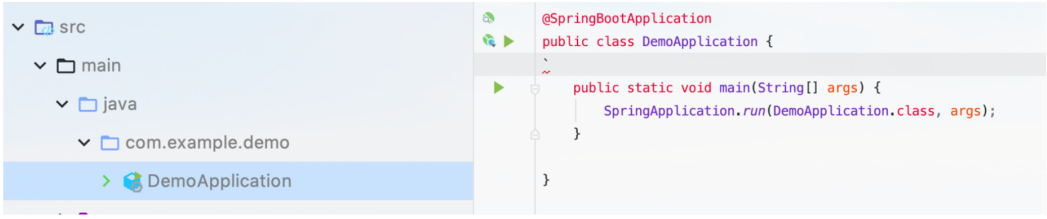
```
#DB Configuration:
#MySQL5.0+没有cj包，MySQL8.0+才有cj包
spring.datasource.driverClassName=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/user_center_server
spring.datasource.username=root
spring.datasource.password=admin123
```

如下图所示



6. 启动 SpringBoot 项目

运行 DemoApplication 主类



控制台输出 Log，说明项目搭建成功

```
2020-05-06 15:42:38.192  INFO 48755 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExecutor'
2020-05-06 15:42:38.417  INFO 48755 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2020-05-06 15:42:38.420  INFO 48755 --- [           main] com.example.demo.DemoApplication         : Started DemoApplication in 3.463 seconds (JVM running for 4.089)
```

Maven镜像

当国内同学的 Maven 下载依赖较慢时，可以添加国内的 Maven 仓库镜像，声明依赖的资源位置在国内，提高下载速度。
将下列标签添加到 pom.xml 文件中。

```
<!-- 使用aliyun镜像 -->
<repositories>
  <repository>
    <id>aliyun</id>
    <name>aliyun</name>
    <url>http://maven.aliyun.com/nexus/content/groups/public</url>
  </repository>
</repositories>
```

位置处 project 下一级子标签即可，如下图所示

```
<?xml version="1.0" encoding="UTF-8"?>
<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

    <!-- 使用aliyun镜像 -->
    <repositories>
        <repository>
            <id>aliyun</id>
            <name>aliyun</name>
            <url>http://maven.aliyun.com/nexus/content/groups/public</url>
        </repository>
    </repositories>
```

前端地址

API设计

HOST：**http://localhost:8099/jzsf**

注册

获取验证码

API：user/getCaptcha
请求方式：POST
参数类型：JSON（请添加请求头：Content-Type:application/json）
携带字段：
{
”username”:”your_username”,

”password”:”your_password”,
”email”:”youremail@mail.com”
}

响应参数类型：JSON
响应信息：
{
”code”: 20018,
”message”: ”获取验证码成功”,
”data”: {
”captcha”: ”rdaP”
}
}

注册

API：user/register

请求方式：POST
参数类型：JSON
携带字段：{
”username”:”your_username”,
”password”:”your_password”,
”email”:”youremail@mail.com”,
”captcha:”captcha”
}
响应信息：{
”code”: 20023,
”message”: ”注册成功”
}

登录

API：user/login
请求方式：POST
参数类型：JSON
请求参数：{
username：”your_username”,
password：”your_password”
}
响应数据
{
”code”: 0,
”message”: ”操作成功”,
”data”: {
”token”: ”longTokenStrings”
}
}

个人中心

API：account/center

请求方式：POST
参数类型：JSON
Auth 认证：添加 HTTP 自定义头，其中 key 为”Authorization”， value 为 ”Bearer ”拼接登录成功接口响应的 token字符串值(注意Bearer后面有空格)
eg.{
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJlYTUwZTIzNC1kYTI4LTRkNjUtYWU2ZS1mMjQ0YjU4NDM2Y2QiLCJzdWIiOiJhZG1pbilslmlzcyl6IjA5OGY2YmNkNDYyMWQzNzNjYWRlNGU4MzI2MjdiNGY2IiwiaWF0IjoxNTkwNjcwMTU2LCJhdWQiOiJqenNmIiwiZXhwIjoxNTkwNjcxOTU2LCJuYmYiOjE1OTA2NzAxNTZ9.QPnlhgDCdoufXxtlfihvEjlcBjos8SLvJESrcbgnR5g
}
前置条件：已登录
响应数据格式：JSON
响应字段：
{
”code”: 0,
”message”: ”操作成功”,
”data”: {
”username”: ”admin”,
”personalProfile”: ”personalProfile”,
”provinceAndCity”: ”provinceAndCity”,
”userTagList”: [],
”articleList”: []
}
}

个人设置 — 基本设置

查看个人设置

API：account/settings/profile/show
请求方式：POST
前置条件：已登录
Auth 认证：同上
响应参数类型：JSON
响应参数：{
”code”: 0,
”message”: ”操作成功”,
”data”: {
”username”: ”username”,
”email”: ”somemail@mail.com”,
”personalProfile”: ”personalProfile”,
”country”: ”China”,
”province”: ”浙江省”,
”city”: ”杭州市”,
”streetAddress”: ”崇仁街”,
”areaNumber”: ”+86”,
”phoneNumber”: ”123123”

}
}

更新个人设置

API：account/settings/profile/update
请求方式：POST
前置条件：已登录
Auth 认证：同上
请求参数：{
”username”: ”admin”,
”email”: ”somemail@mail.com”,
”personalProfile”: ”Oh Yeah”,
”country”: ”China”,
”province”: ”浙江省”,
”city”: ”杭州市”,
”streetAddress”: ”崇仁桥”,
”areaNumber”: ”+86”,
”phoneNumber”: ”123123”
}
响应参数类型：JSON
响应参数：{
”code”: 0,
”message”: ”操作成功”
}

个人设置 — 新消息通知设置

查看通知设置

API：account/settings/notice/show
请求方式：POST

前置条件：已登录
Auth 认证：同上
响应参数类型：JSON
响应参数：{
”code”: 0,
”message”: ”操作成功”,
”data”: {
”userId”: ”48039dd5-d8f0-46d1-b0dd-77ad5ee7ab7d”,
”todoNotice”: ”1”,
”sysMessageNotice”: ”0”,
”otherUserMessageNotice”: ”0”
}
}

更新通知设置

API：account/settings/notice/update
请求方式：POST

前置条件：已登录
Auth 认证：同上
参数类型：JSON
请求参数：{
    "todoNotice": "1",
    "sysMessageNotice": "1",
    "otherUserMessageNotice": "1"
}
响应参数类型：JSON
响应参数：{
  "code": 0,
  "message": "操作成功",
  "data": null
}

文章操作模块

发布文章

API：article/publish
请求方式：POST
参数类型：JSON
前置条件：已登录
Auth 认证：同上
请求参数：{
    "publishTime":"2020-04-15 10:20:03",
    "title":"user-center的后端开发",
    "content":"该项目是前后端分离，基于restful风格。",
    "articleTagList":[
        "Java",
        "后端编程",
        "Spring"
    ]
}
响应参数类型：JSON
响应参数：{
  "code": 0,
  "message": "操作成功",
  "data": "ea6f6f92-b564-41bc-ab71-12c26e02255d"
}

更新文章

API：article/update
请求方式：POST
参数类型：JSON
前置条件：已登录
Auth 认证：同上
请求参数：{
    "id":"ea6f6f92-b564-41bc-ab71-12c26e02255d",
    "publishTime":"2020-04-15 10:20:03",

```
    "title":"user-center的后端开发",
    "content":"该项目是前后端分离，基于restful风格。",
    "articleTagList":[
        "Java",
        "后端编程",
        "Spring"
    ]
}
```


响应参数类型：JSON

响应参数：{


```
  "code": 0,
  "message": "操作成功"
}
```


删除文章

API：article/delete
请求方式：POST
参数类型：JSON
前置条件：已登录
Auth 认证：同上
请求参数：{


```
    "id":"ea6f6f92-b564-41bc-ab71-12c26e02255d"
}
```


响应参数类型：JSON

响应参数：{


```
  "code": 0,
  "message": "操作成功"
}
```


查询文章

单个文章查询
API：article/show/detail
请求方式：POST
参数类型：JSON
前置条件：已登录
Auth 认证：同上
请求参数：{


```
    "id":"文章 id"
}
```


响应参数类型：JSON

响应参数：{


```
  "code": 0,
  "message": "操作成功",
  "data": {
      {
        "id": "1",
        "userId": "48039dd5-d8f0-46d1-b0dd-77ad5ee7ab7d",
        "publishTime": "2020-04-14T20:23:25.000+0000",
```

```
        "title": "t1",
        "content": "strs"
      }

}
```

数据库表字段设计

user表

filed_name	type	comment
id	varchar(48)	主键id
username	varchar(16)	用户名
password	varchar(64)	密码(md5)
is_verified	char(1)	是否激活
phone_number	varchar(16)	手机号
area_number	char(4)	区号

register_record表

field_name	type	comment
id	varchar(48)	PK id
user_id	varchar(48)	用户id
email	varchar(48)	邮箱
captcha	char(4)	验证码
sent_time	datetime	发送时间

user_profile表

filed_name	type	comment
user_id	varchar(48)	PK 用户id
photo	varchar(32)	头像图片的路径
personal_profile	varchar(1024)	个人简介

adress表

user_id	varchar(48)	PK 用户id
country	char(8)	国家
province	char(8)	省
street_address	varchar(16)	街道地址
city	char(8)	市

user_tag表

filed_name	type	comment
id	varchar(48)	主键id
user_id	varchar(48)	关联用户id
tag_name	varchar(16)	标签名

user_preference表

filed_name	type	comment
user_id	varchar(48)	PK 用户id
todo_notice	char(1)	是否接收待办通知
sys_message_notice	char(1)	是否接收系统通知
other_user_message_notice	char(1)	是否接收其他用户信息通知

article表

filed_name	type	comment
id	varchar(48)	PK 文章id
user_id	varchar(48)	发布人
publish_time	datetime	发布时间
title	varchar(16)	文章标题
content	blob	文章内容

article_tag表

filed_name	type	comment
id	varchar(48)	主键id
articlet_id	varchar(48)	关联文章id
tag_name	varchar(1024)	标签内容