

# 乐字节教育高级架构课程

正所谓“授人以鱼不如授人以渔”，你们想要的 **Java 学习资料** 来啦！

不管你是学生，还是已经步入职场的同行，希望你们都要珍惜眼前的学习机会，奋斗没有终点，知识永不过时。

扫描下方二维码即可领取



乐字节晓啡



乐字节官方交流群



# 后台管理系统——项目框架搭建

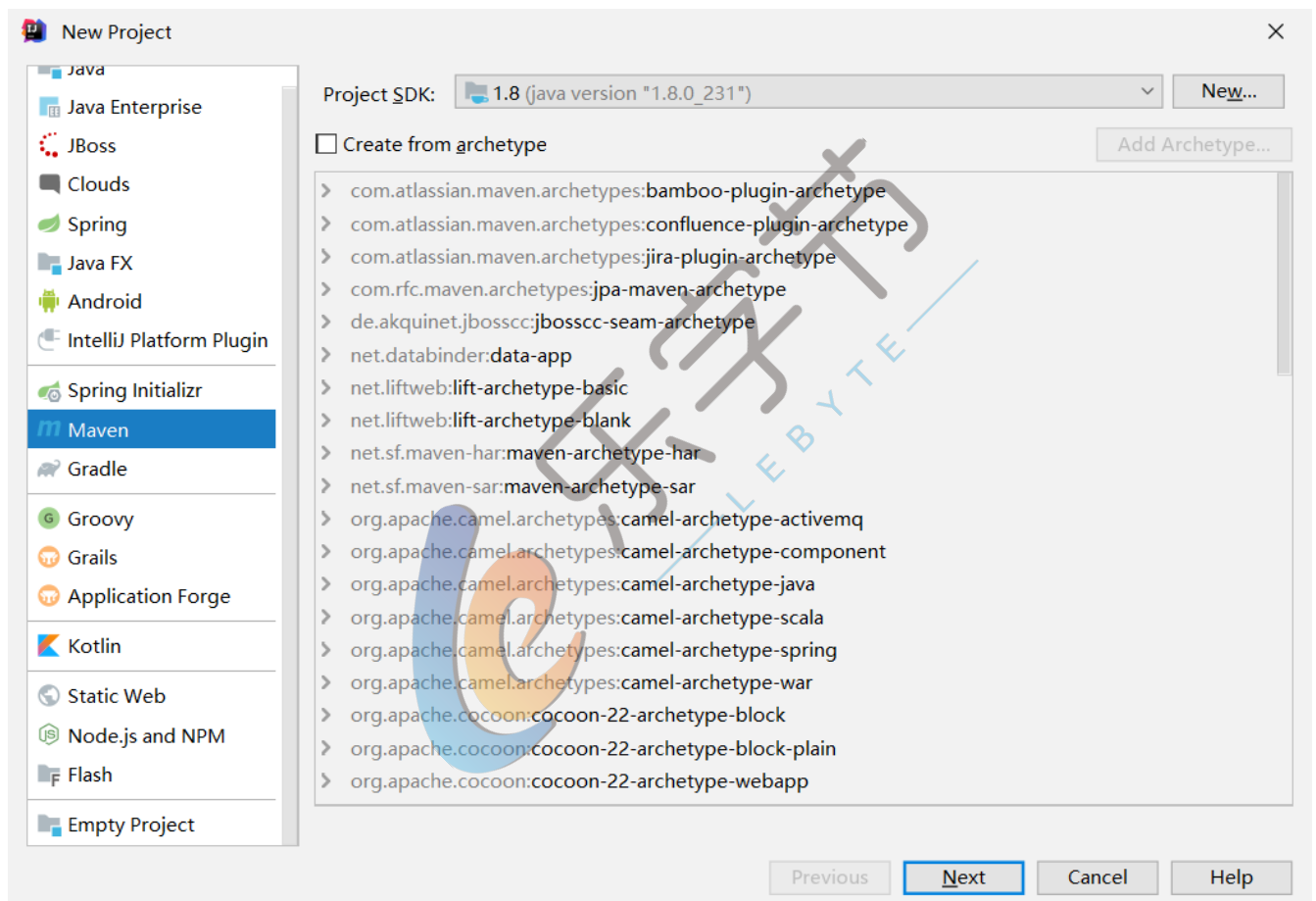
## shop父模块

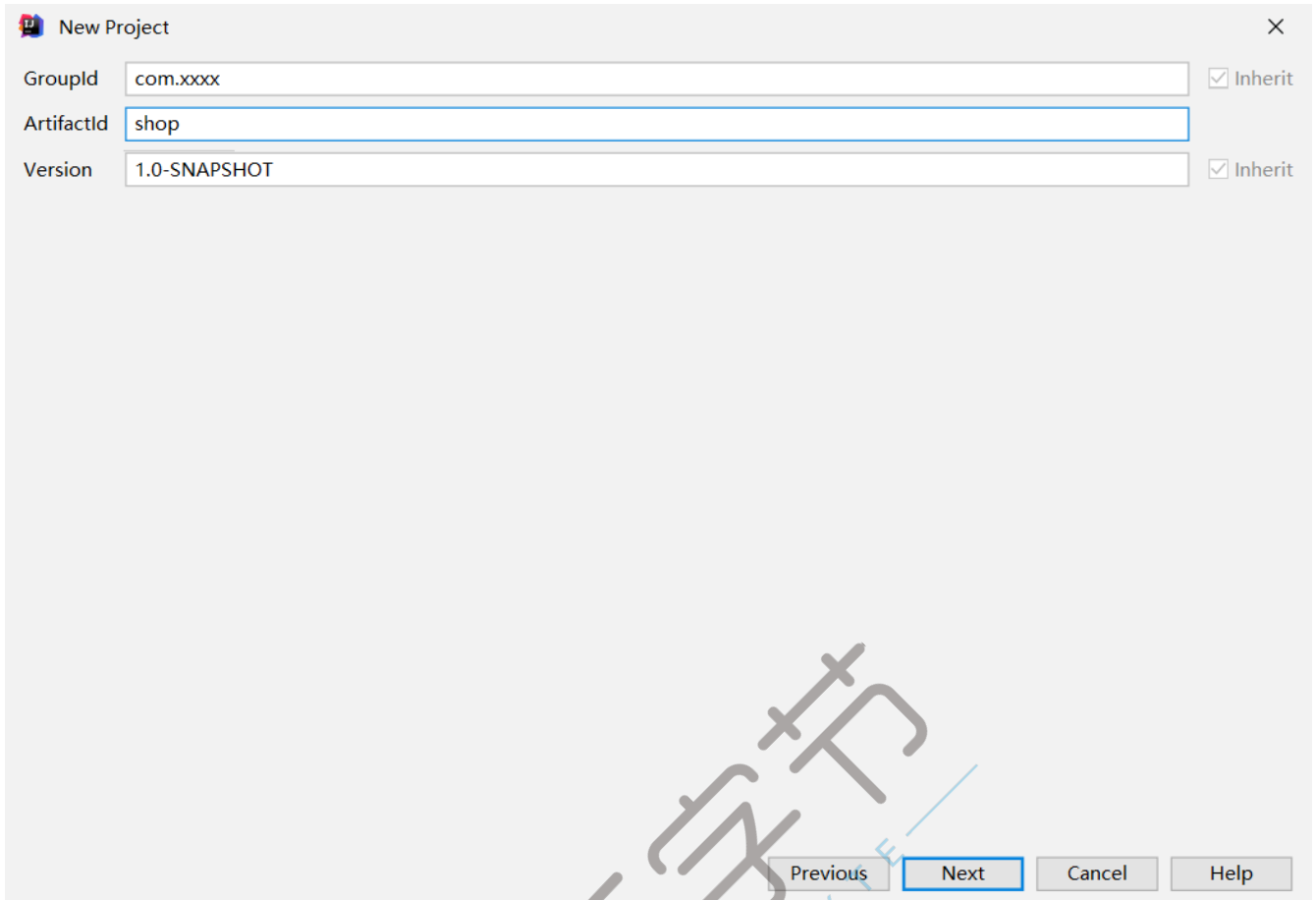
### 创建pom项目

Maven提供了聚合类型的项目，其本质就是一个分布式架构。

Maven分布式架构-聚合类型，需要一个parent项目，shop-parent就是parent项目。

创建父项目





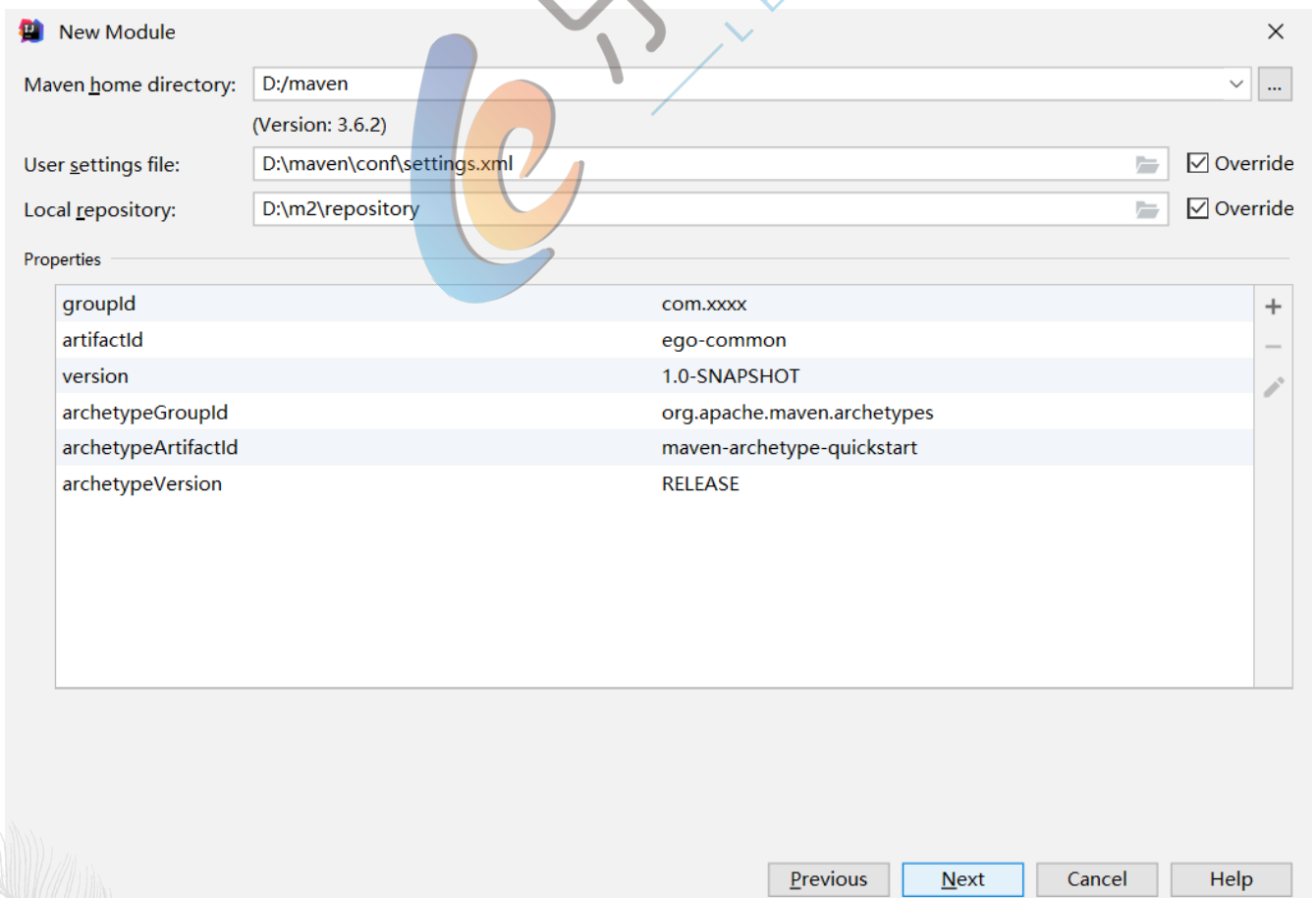
New Project

GroupId:  ☒ Inherit

ArtifactId:

Version:  ☒ Inherit

Previous Next Cancel Help



New Module

Maven home directory:  (Version: 3.6.2)

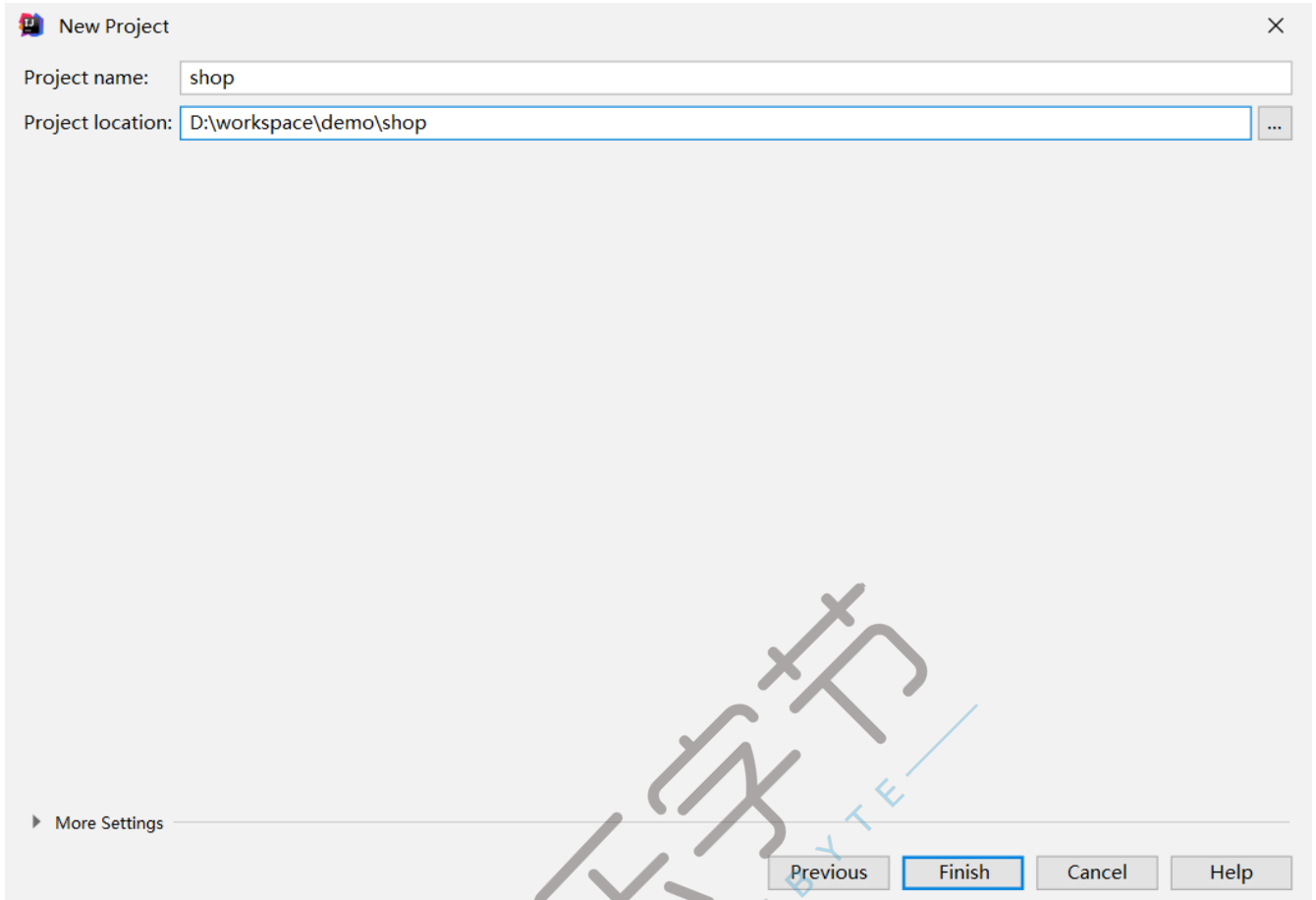
User settings file:  ☒ Override

Local repository:  ☒ Override

Properties

groupId	com.xxxx
artifactId	ego-common
version	1.0-SNAPSHOT
archetypeGroupId	org.apache.maven.archetypes
archetypeArtifactId	maven-archetype-quickstart
archetypeVersion	RELEASE

Previous Next Cancel Help



## 编辑pom.xml配置文件

```
<?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
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <!-- 项目坐标地址 -->
  <groupId>com.xxxx</groupId>
  <!-- 项目模块名称 -->
  <artifactId>shop</artifactId>
  <!-- 项目打包类型 -->
  <packaging>pom</packaging>
  <!-- 项目版本名称 快照版本SNAPSHOT、正式版本RELEASE -->
  <version>1.0-SNAPSHOT</version>

  <!--
    模块管理，实现 pom 项目之间的聚合关系，
    聚合关系下对父项目使用 mvn 命令会对其他子项目产生同样效果
  -->
  <modules>
```

```

<module>shop-common</module>
<module>shop-generator</module>
<module>shop-manager</module>
<module>shop-order</module>
<module>shop-portal</module>
<module>shop-rpc</module>
<module>shop-sso</module>
</modules>

<!-- 继承 spring-boot-starter-parent 依赖 -->
<!-- 使用继承方式，实现复用，符合继承的都可以被使用 -->
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.2.2.RELEASE</version>
</parent>

<!--
  集中定义依赖组件版本号，但不引入，
  在子工程中用到声明的依赖时，可以不加依赖的版本号，
  这样可以统一管理工程中用到的依赖版本
-->
<properties>
  <!-- JDK 版本定义 -->
  <java.version>1.8</java.version>
  <!-- mybatis 依赖 -->
  <mybatis.version>2.1.1</mybatis.version>
  <!-- pagehelper 分页依赖 -->
  <pagehelper.version>1.2.13</pagehelper.version>
  <!-- mysql 数据库依赖 -->
  <mysql.version>8.0.18</mysql.version>
  <!-- druid 连接池依赖 -->
  <druid.version>1.1.20</druid.version>
</properties>

<!-- 项目依赖管理 父项目只是声明依赖，子项目需要写明需要的依赖（可以省略版本信息） -->
<dependencyManagement>
  <dependencies>
    <!-- mybatis 依赖 -->
    <dependency>
      <groupId>org.mybatis.spring.boot</groupId>
      <artifactId>mybatis-spring-boot-starter</artifactId>
      <version>${mybatis.version}</version>
    </dependency>
    <!-- pagehelper 分页依赖 -->
    <dependency>
      <groupId>com.github.pagehelper</groupId>
      <artifactId>pagehelper-spring-boot-starter</artifactId>
      <version>${pagehelper.version}</version>
    </dependency>
    <!-- mysql 数据库依赖 -->
    <dependency>
      <groupId>mysql</groupId>

```

```

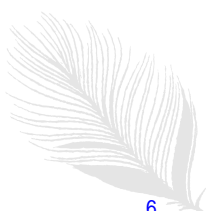
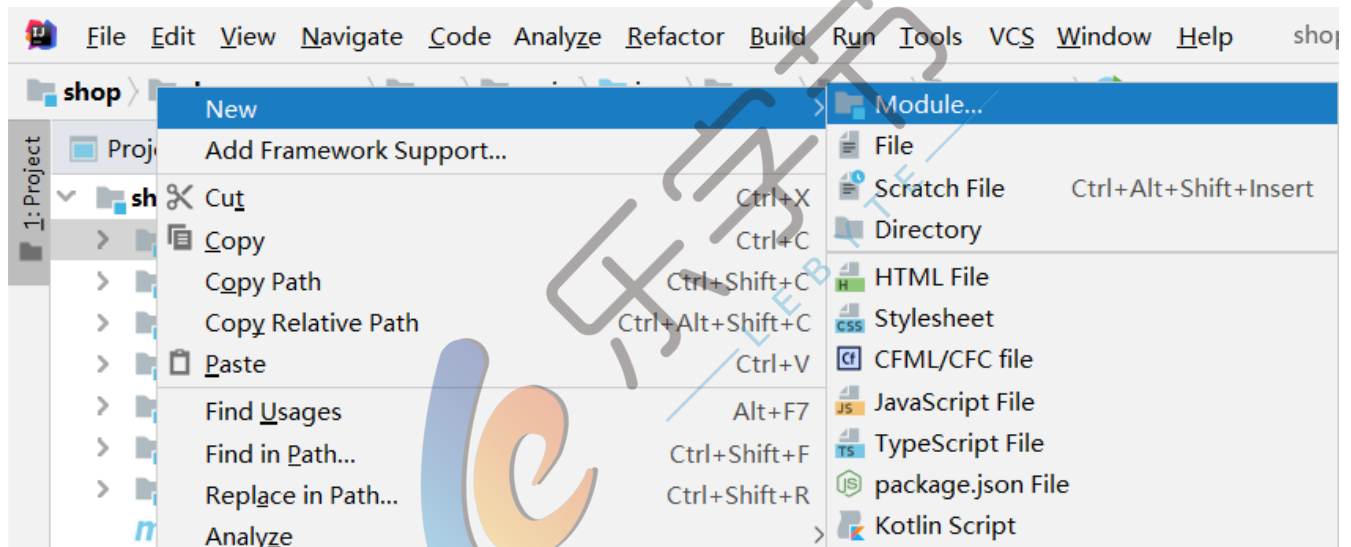
        <artifactId>mysql-connector-java</artifactId>
        <version>${mysql.version}</version>
    </dependency>
    <!-- druid 连接池依赖 -->
    <dependency>
        <groupId>com.alibaba</groupId>
        <artifactId>druid</artifactId>
        <version>${druid.version}</version>
    </dependency>
</dependencies>
</dependencyManagement>
</project>

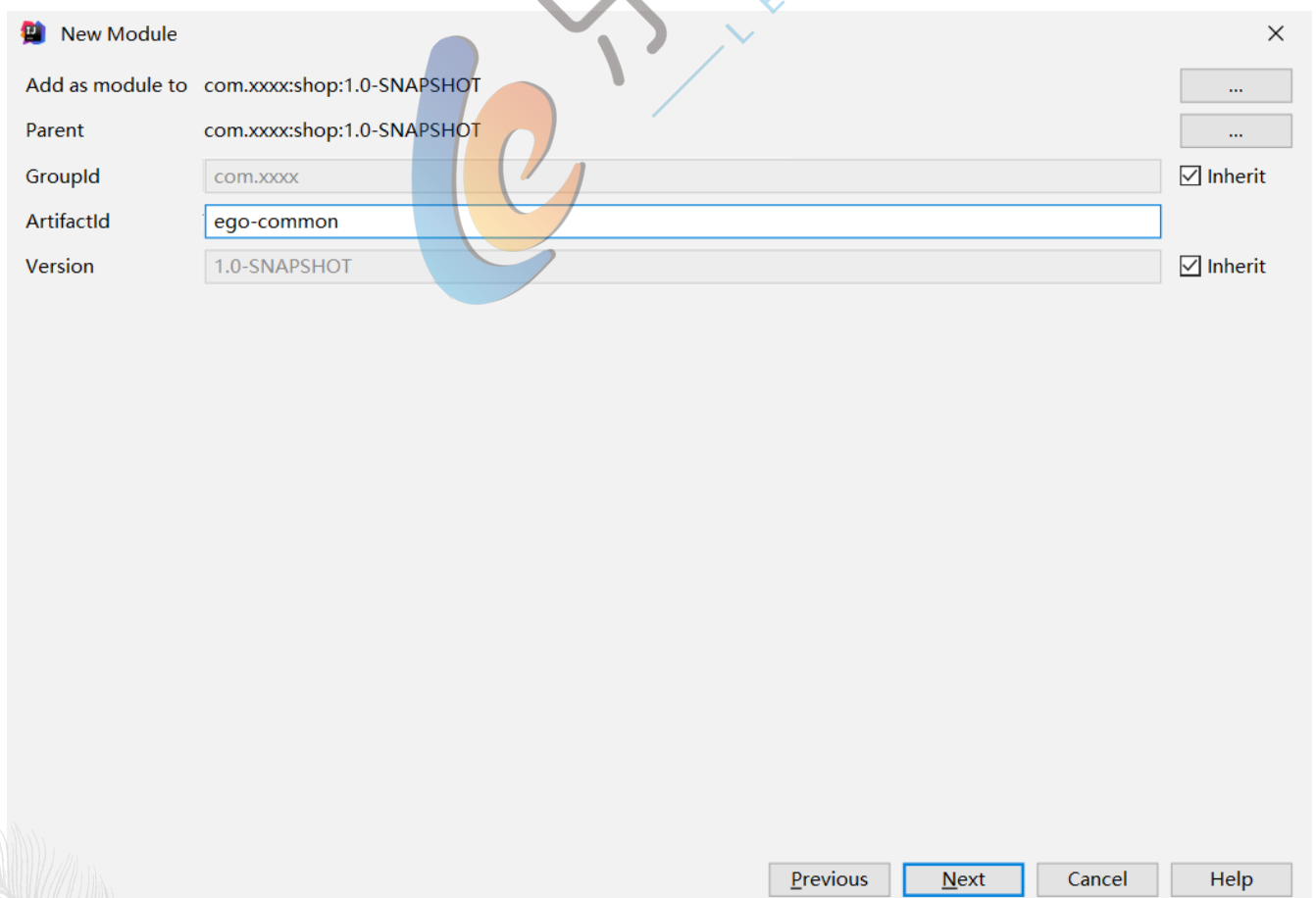
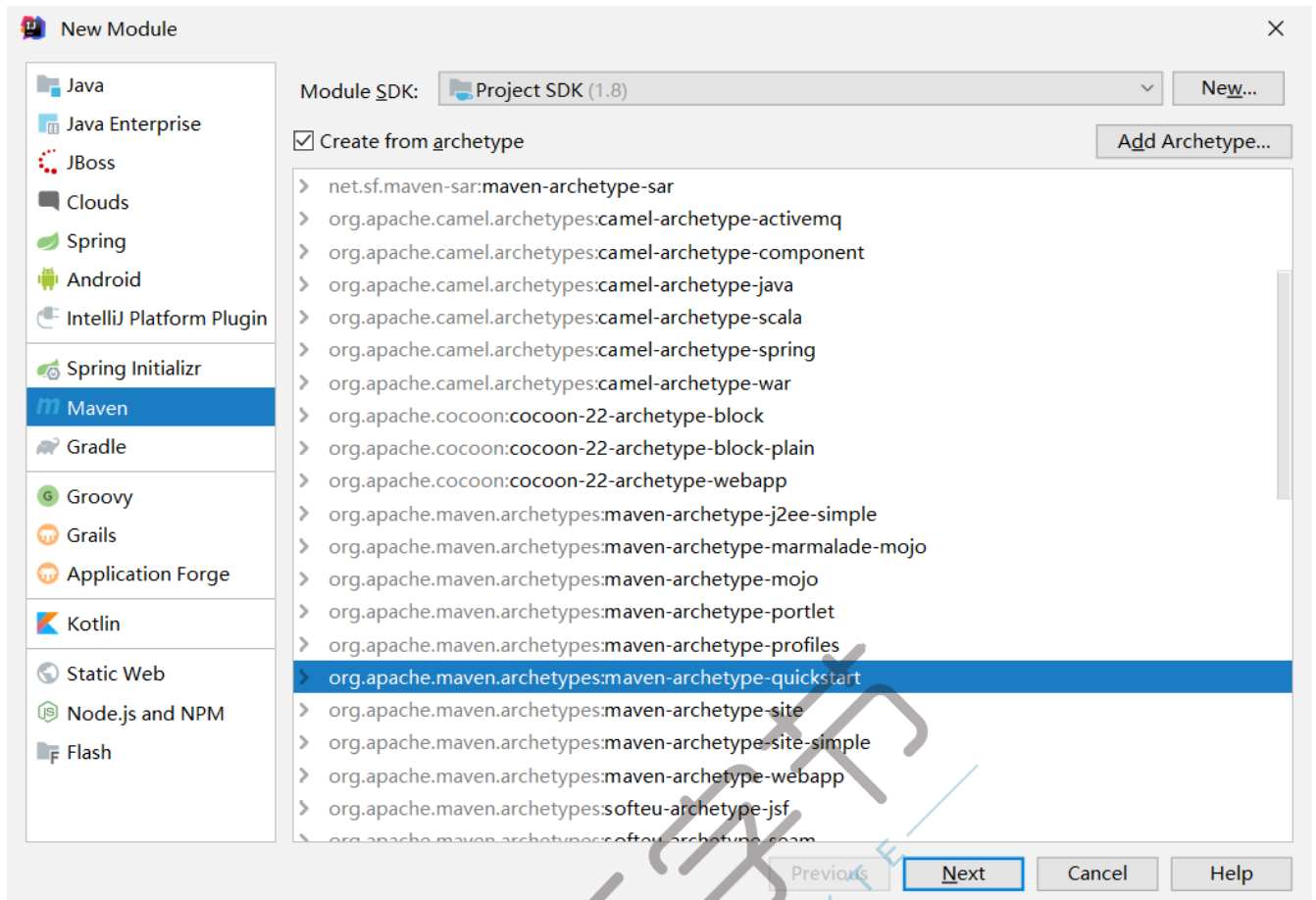
```

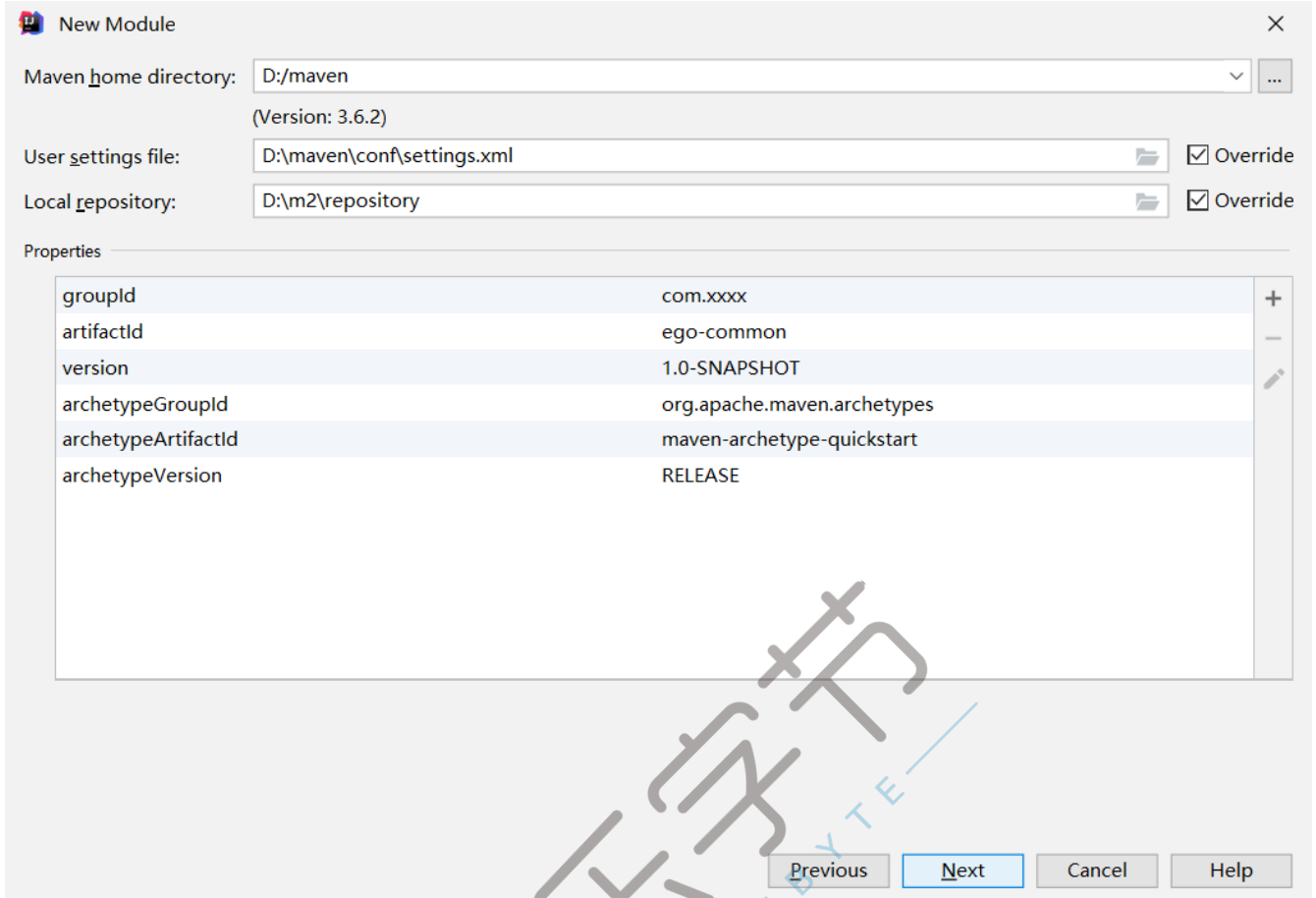
## shop-common子模块

用于添加公共的工具类、枚举类、拦截器等。

## 创建shop-manager项目







New Module

Maven home directory: D:/maven (Version: 3.6.2)

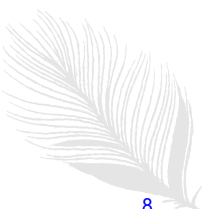
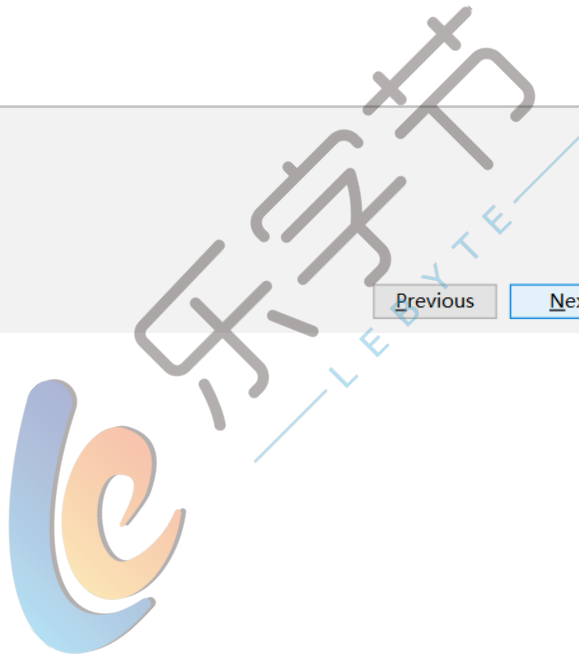
User settings file: D:\maven\conf\settings.xml ☒ Override

Local repository: D:\m2\repository ☒ Override

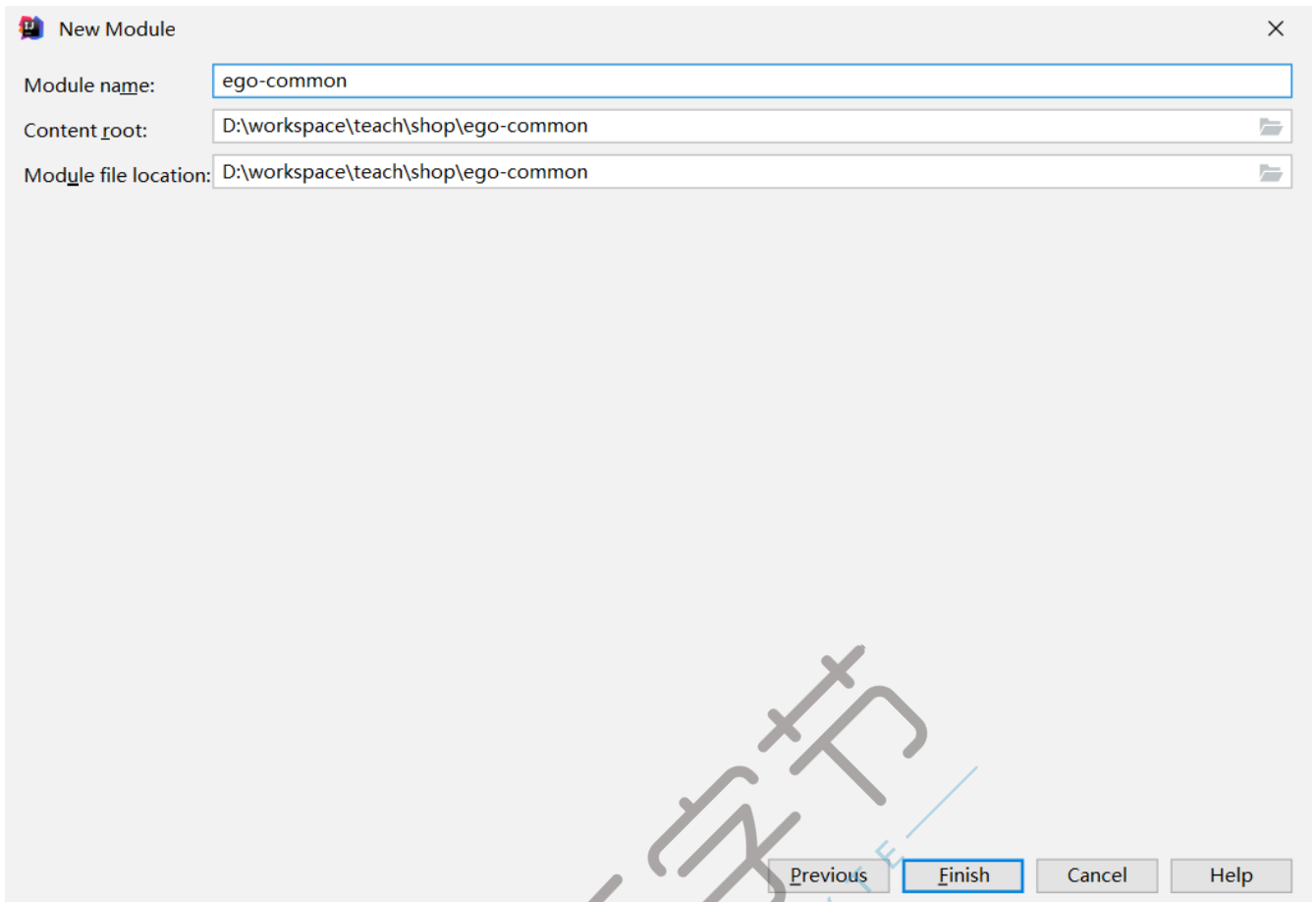
Properties

groupId	com.xxxx	+
artifactId	ego-common	-
version	1.0-SNAPSHOT	✎
archetypeGroupId	org.apache.maven.archetypes	
archetypeArtifactId	maven-archetype-quickstart	
archetypeVersion	RELEASE	

Previous Next Cancel Help







New Module

Module name: ego-common

Content root: D:\workspace\teach\shop\ego-common

Module file location: D:\workspace\teach\shop\ego-common

Previous Finish Cancel Help

## 编辑pom.xml配置文件

```
<?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 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.xxxx</groupId>
  <artifactId>shop-common</artifactId>
  <version>1.0-SNAPSHOT</version>

  <!-- 继承 shop-parent 依赖 -->
  <parent>
    <groupId>com.xxxx</groupId>
    <artifactId>shop</artifactId>
    <version>1.0-SNAPSHOT</version>
  </parent>

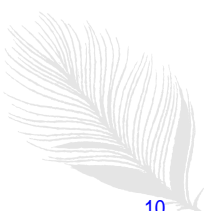
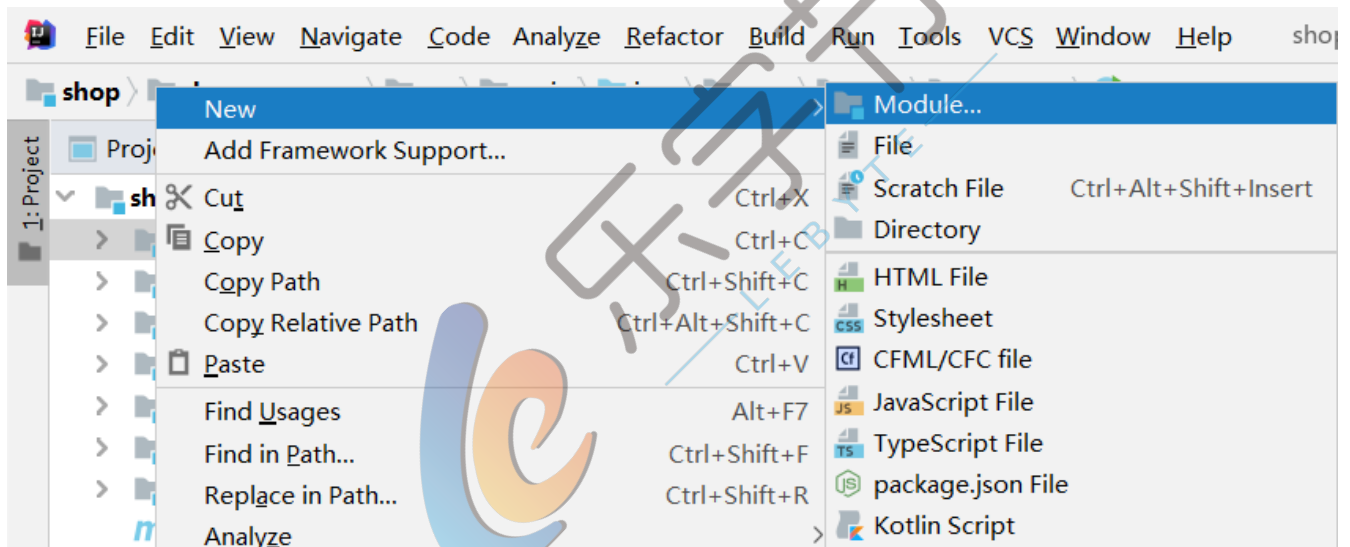
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
```

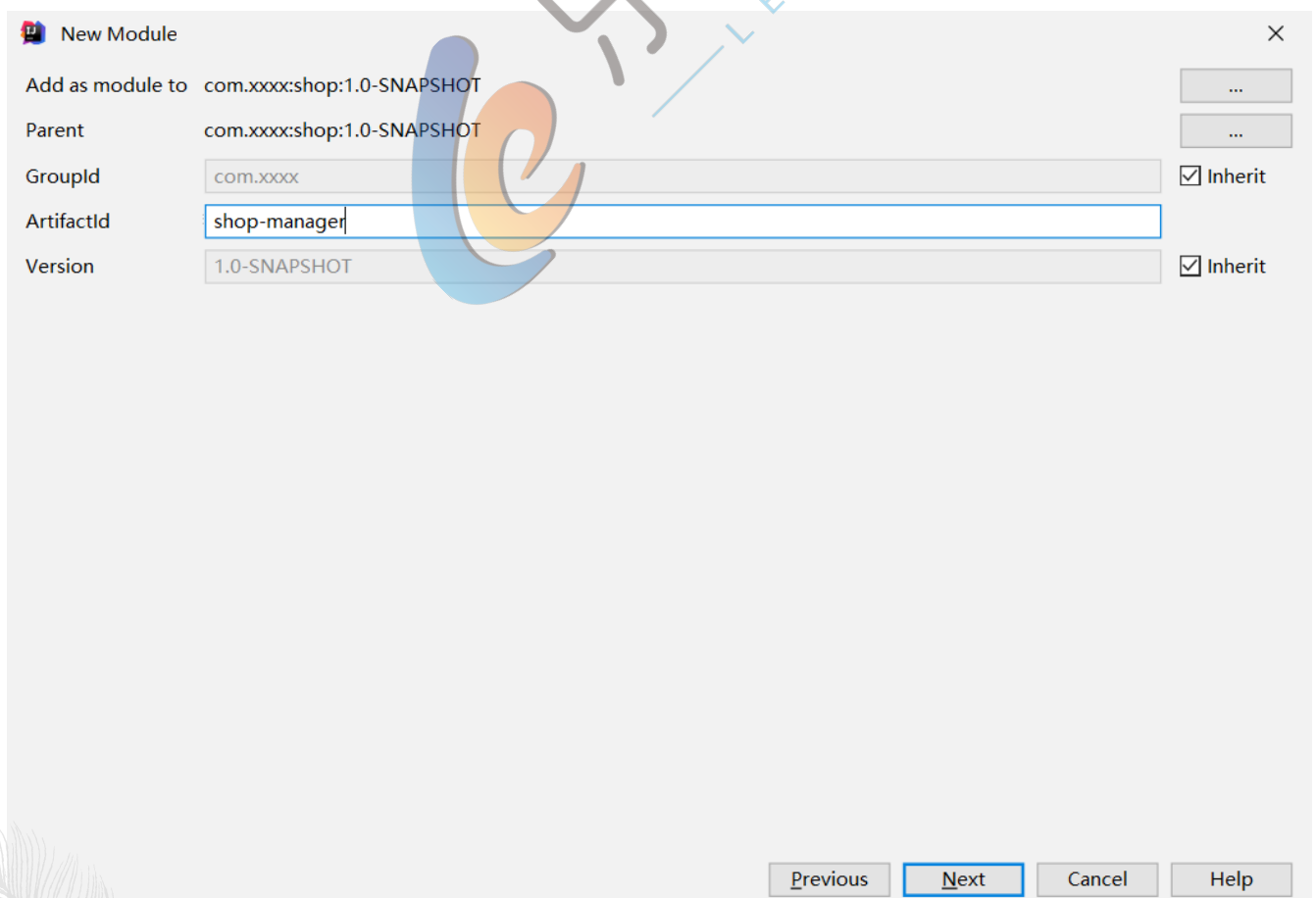
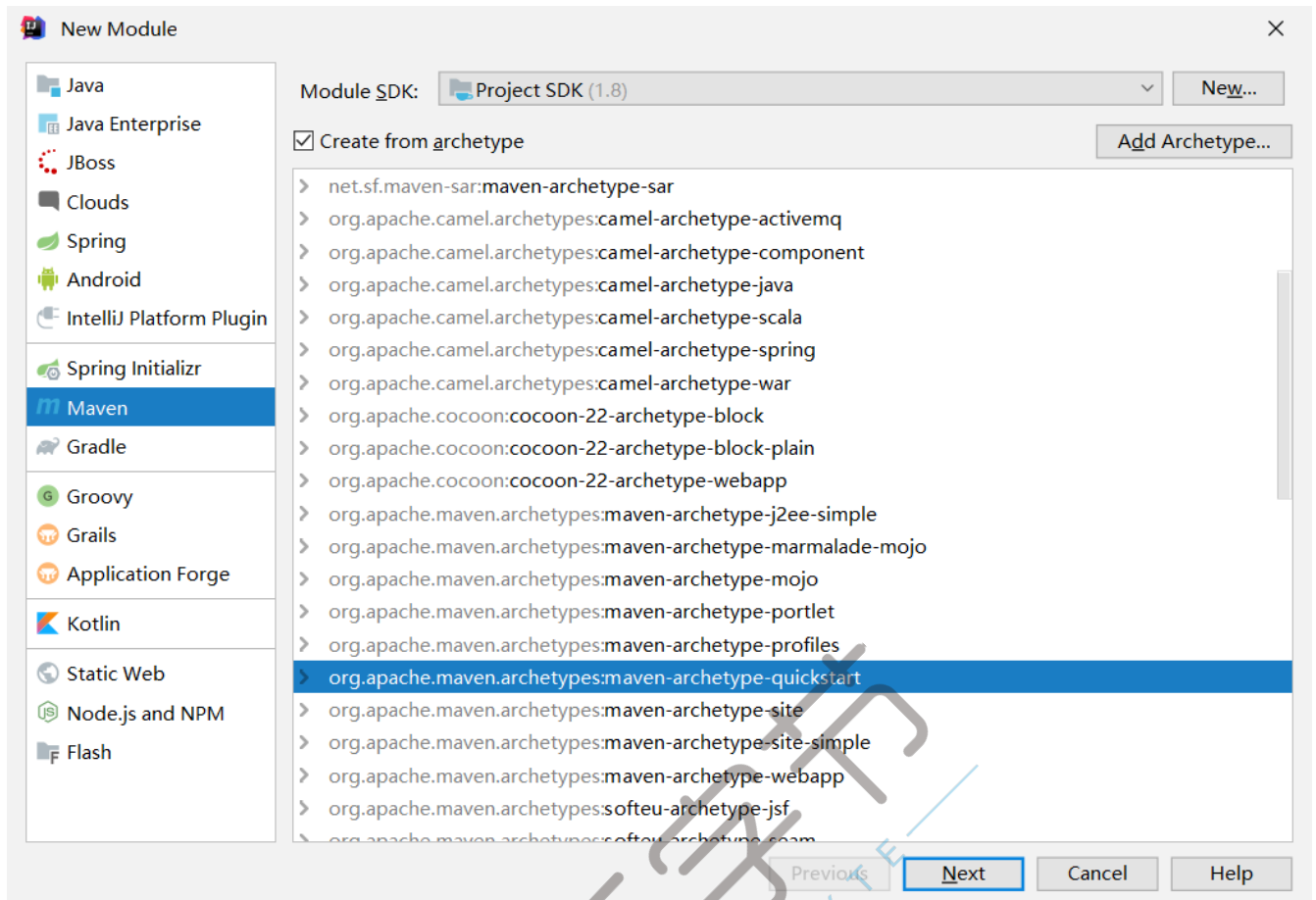
```
<dependencies>
  <!-- spring boot web 依赖 -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>
  <!-- pagehelper 分页依赖 -->
  <dependency>
    <groupId>com.github.pagehelper</groupId>
    <artifactId>pagehelper-spring-boot-starter</artifactId>
  </dependency>
</dependencies>
</project>
```

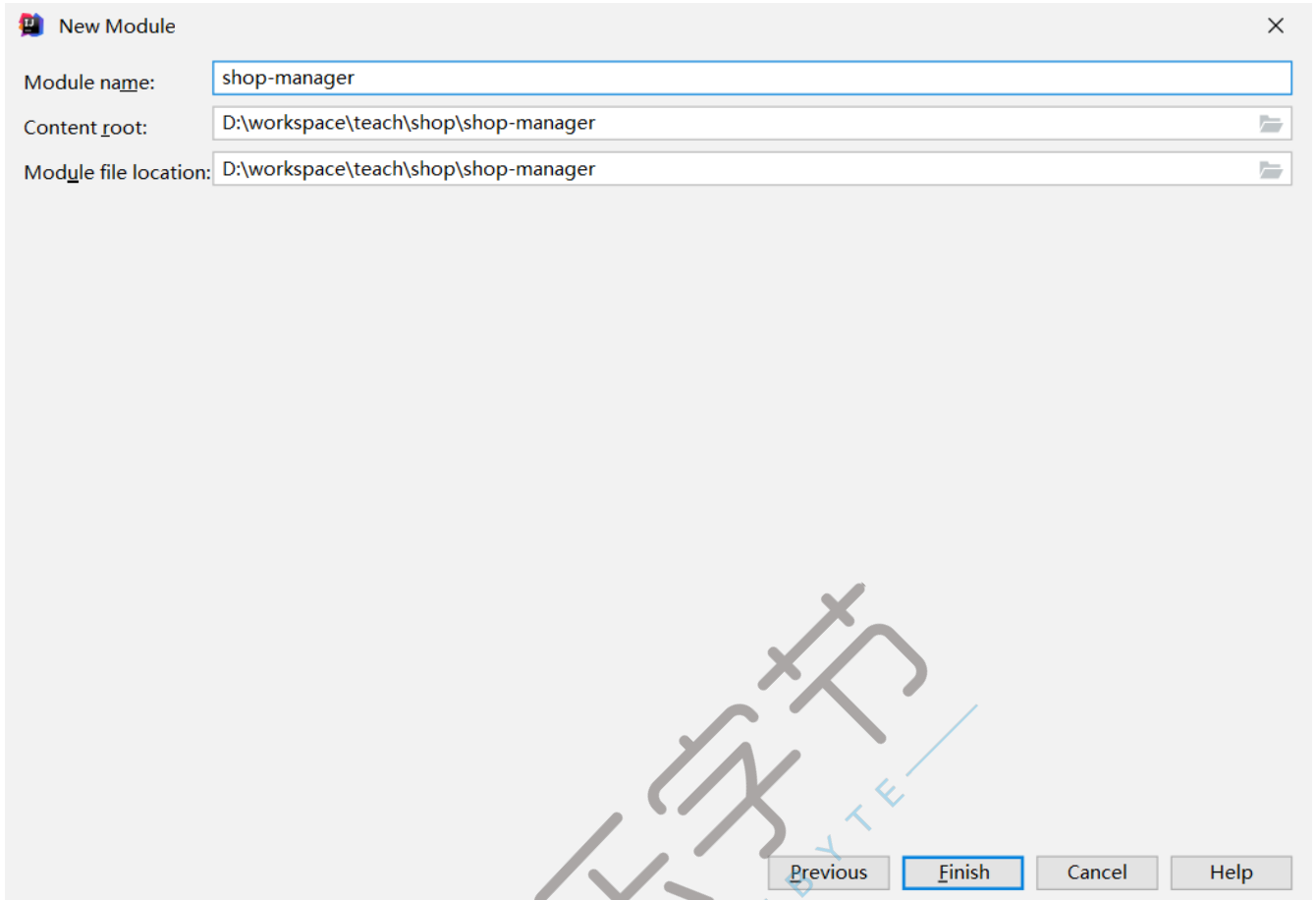
## shop-manager子模块

后台管理系统

### 创建shop-manager项目







## 编辑pom.xml配置文件

```
<?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 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.xxxx</groupId>
  <artifactId>shop-manager</artifactId>
  <version>1.0-SNAPSHOT</version>

  <!-- 继承 shop-parent 依赖 -->
  <parent>
    <groupId>com.xxxx</groupId>
    <artifactId>shop</artifactId>
    <version>1.0-SNAPSHOT</version>
  </parent>

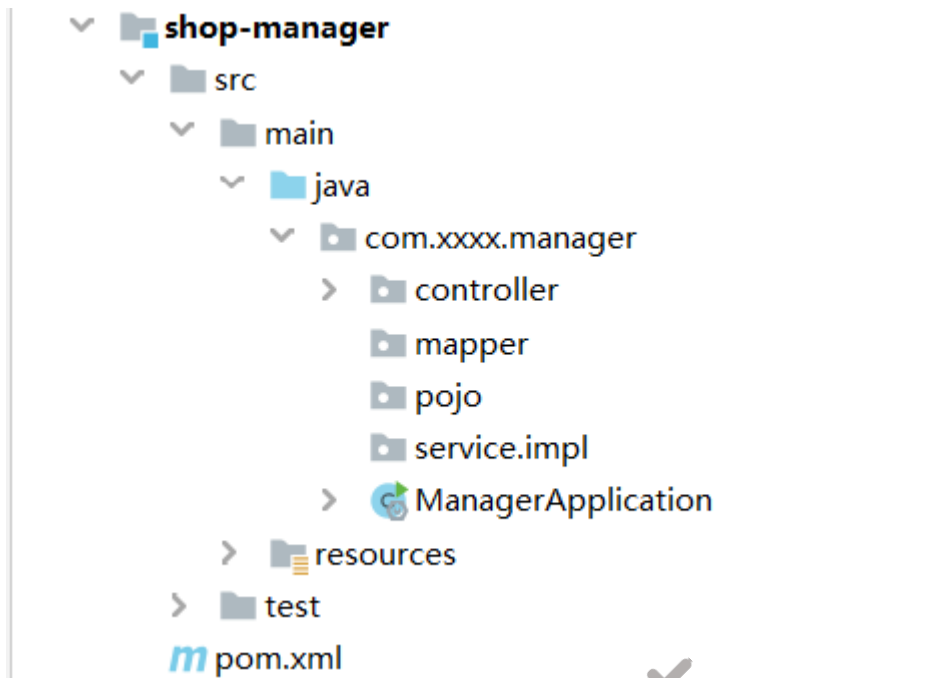
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
</project>
```

```
</properties>

<dependencies>
  <!-- shop common 依赖 -->
  <dependency>
    <groupId>com.xxxx</groupId>
    <artifactId>shop-common</artifactId>
    <version>1.0-SNAPSHOT</version>
  </dependency>
  <!-- spring boot freemarker 依赖 -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-freemarker</artifactId>
  </dependency>
  <!-- mybatis 依赖 -->
  <dependency>
    <groupId>org.mybatis.spring.boot</groupId>
    <artifactId>mybatis-spring-boot-starter</artifactId>
  </dependency>
  <!-- mysql 数据库依赖 -->
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
  </dependency>
  <!-- druid 连接池依赖 -->
  <dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>druid</artifactId>
  </dependency>
  <!-- spring boot test 依赖 -->
  <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>
</dependencies>
</project>
```

## 其他配置文件





application-dev.yml

```

server:
  port: 9090
  servlet:
    context-path: /shop-manager

# Spring
spring:
  # 数据源
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/shop?useUnicode=true&characterEncoding=UTF-8&serverTimezone=Asia/Shanghai
    username: root
    password: root
    # 指定 druid 连接池以及 druid 连接池配置
    type: com.alibaba.druid.pool.DruidDataSource
    druid:
      initial-size: 1
      max-active: 20
      max-idle: 20
      min-idle: 1
      max-wait: 60000

# freemarker 模板引擎
freemarker:
  cache: false
  charset: UTF-8
  content-type: text/html;charset=UTF-8
  enabled: true
  suffix: .ftl
  template-loader-path: classpath:/views/
  # 配置模板里是否可以直接取request的属性 request是别名
  request-context-attribute: request

```

```
# 配置将request和session中的键值添加到
# AbstractTemplateView类的renderMergedOutputModel方法中的model这个Map参数中
expose-request-attributes: true
expose-spring-macro-helpers: true
# 配置模板里是否可以直接取session的属性 true 是允许
expose-session-attributes: true
settings:
    tag-syntax: auto_detect # 配置标签语法为自动, 页面可以将 <> 改为 [], 为了区别
html 标签
    template_update_delay: 0 # 模板更新时间, 单位秒
    default_encoding: UTF-8 # 默认编码字符集
    output_encoding: UTF-8 # 模板输出编码字符集
    locale: zh_CN # 本地化配置
    date_format: yyyy-MM-dd # 日期格式化
    time_format: HH:mm:ss # 时间格式化
    datetime_format: yyyy-MM-dd HH:mm:ss # 日期时间格式化
    number_format: #.## # 数字格式化
    boolean_format: true,false # boolean格式化
    # ignore, debug, html_debug, rethrow
    # 1.TemplateExceptionHandler.IGNORE_HANDLER简单地压制所有异常
    # 它对处理异常没有任何作用, 也不会重新抛出异常, 页面可以正常渲染, 后台抛异常
    # 2.TemplateExceptionHandler.DEBUG_HANDLER打印堆栈信息和重新抛出异常。这是默认的异常控制器
    # 3.TemplateExceptionHandler.HTML_DEBUG_HANDLER和DEBUG_HANDLER相同
    # 但是可以格式化堆栈跟踪信息, HTML页面, 建议使用它而不是DEBUG_HANDLER
    # 4.TemplateExceptionHandler.RETHROW_HANDLER简单重新抛出所有异常而不会做其他的事情
    # 5.使用自定义异常类实现TemplateExceptionHandler重写handleTemplateExceptionHandler方法
    template_exception_handler: html_debug
# MyBatis
mybatis:
    # 配置 MyBatis数据返回类型别名(默认别名是类名)
    type-aliases-package: com.xxxx.manager.pojo
    # 配置 MyBatis Mapper 映射文件
    mapper-locations: classpath:/mapper/*.xml
    # Mybatis SQL 打印(方法接口所在的包, 不是 Mapper.xml 所在的包)
logging:
    level:
        com.xxxx.manager.mapper: debug
```

## 项目运行测试

PageController.java

```
package com.xxxx.manager.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;

/**
 * 跳转页面
 *
 * @author zhoubin
 * @since 1.0.0
```

```

*/
@Controller
public class PageController {
    /**
     * 公共页面跳转 restful风格
     * 比如：前台传welcome就跳转至/WEB-INF/pages/welcome.ftl
     * 比如：前台传login就跳转至/WEB-INF/pages/login.ftl
     * ...
     * @param page
     * @return
     */
    @RequestMapping("/{page}")
    public String page(@PathVariable String page) {
        System.out.println(page);
        return page;
    }
}

```

ManagerApplication.java

```

package com.xxxx.manager;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

/**
 * 启动类
 */
@SpringBootApplication
public class ManagerApplication {
    public static void main(String[] args) {
        SpringApplication.run(ManagerApplication.class,args);
    }
}

```

Welcome.ftl

```








<!-- 设置项目根路径全局变量 -->
<#assign ctx=request.contextPath/>
<!DOCTYPE html>
<html>
<head>
    <title>shop商城后台管理系统</title>
</head>
<body>
<h1>${ctx}欢迎使用shop商城后台管理系统</h1>
</body>
</html>

```

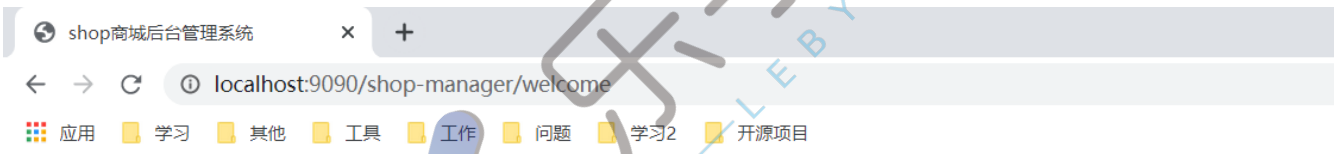
程序运行前先把parent项目install一下





- >  shop (root)
- >  shop-common
- >  shop-generator
- ▼  shop-manager
  - ▼  Lifecycle
    - ⚙️ clean
    - ⚙️ validate
    - ⚙️ compile
    - ⚙️ test
    - ⚙️ package
    - ⚙️ verify
    - ⚙️ install**
    - ⚙️ site
    - ⚙️ deploy
  - >  Plugins
  - >  Dependencies

#### 测试结果



**/shop-manager欢迎使用shop商城后台管理系统**

