Maven构建旅游项目聚合工程

代码下载

https://gitee.com/kekesam/kuangstudy-pug-parent.git

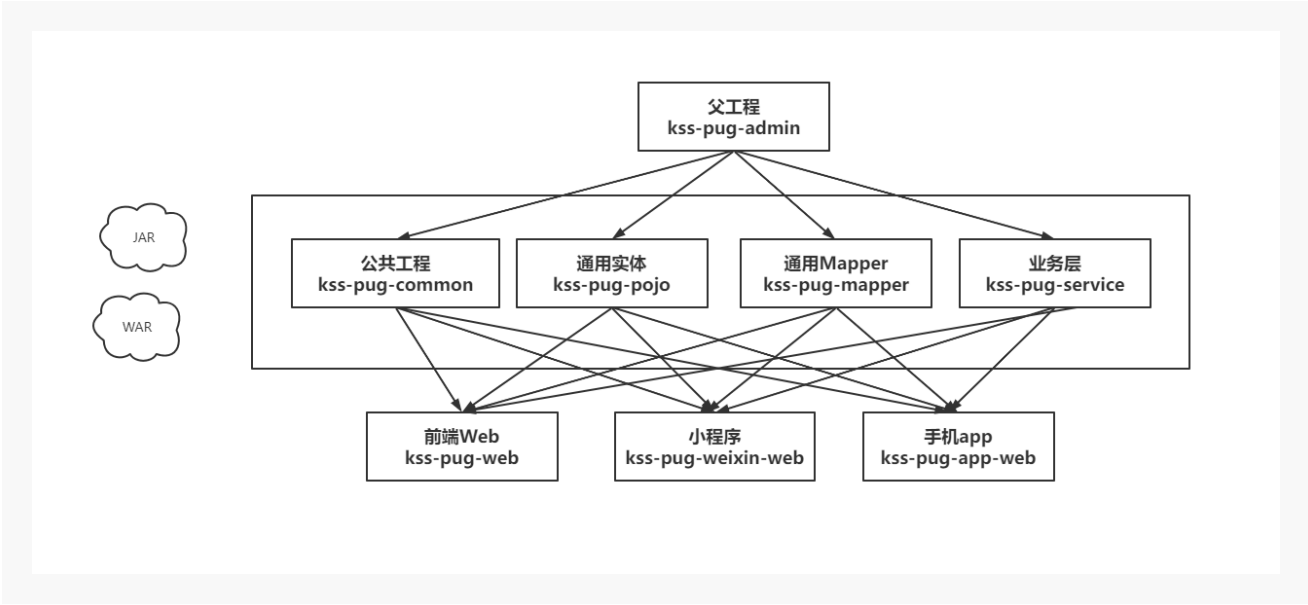
01、单体架构存在的缺陷

比如：个人资讯系统 存在的问题如下：

> 一句话： 集中式的单体架构往往会将多个功能放到一个应用中，经过日积月累，这个应用就会编程一个庞大而复杂的“怪物”， 随着项目团队成员的更替，时间一长就很少有人能够完全的搞懂这些代码之间的逻辑关系。有些人可能会因为担心修改遗留代码而出现不可预知的*BUG*，，而宁愿增加大量不必要的代码，这样会导致应用越来越庞大，越来越复杂，可读性越来越差，最终陷入恶性循环。对于整个团队来说，系统研发工作编程了一件及其痛苦的事情。

02、单体架构进行重构和拆分

整体的项目的聚合工程如下：



各模块职责定位：

- **kss-pug-admin**：父工程，打包方式为pom，统一锁定(管理)依赖的版本，同时聚合其他子模块便于统一执行maven命令打包，编译和部署的命令。
- **kss-pug-common**：通用模块，打包方式为jar，存放项目中使用到的一些工具类和常量类。
- **kss-pug-pojo**：==打包方式为jar==，存放实体类和返回结果类等
- **kss-pug-mapper**：持久层模块，打包方式为jar，存放Dao接口和Mapper映射文件等
- **kss-pug-service**：Dubbo服务模块，打包方式为war，存放服务实现类，作为服务提供方，需要部署到tomcat运行
- **kss-pug-web-api**：打包方式为jar/war，用于部署系统的后台接口。
- **kss-pug-weixin-web-api**： 打包方式为jar/war 提供给小程序使用的接口。

依赖架构好处是什么？

- 不需要重复造轮子，比如：pojo ,service,common,mapper重新开发。

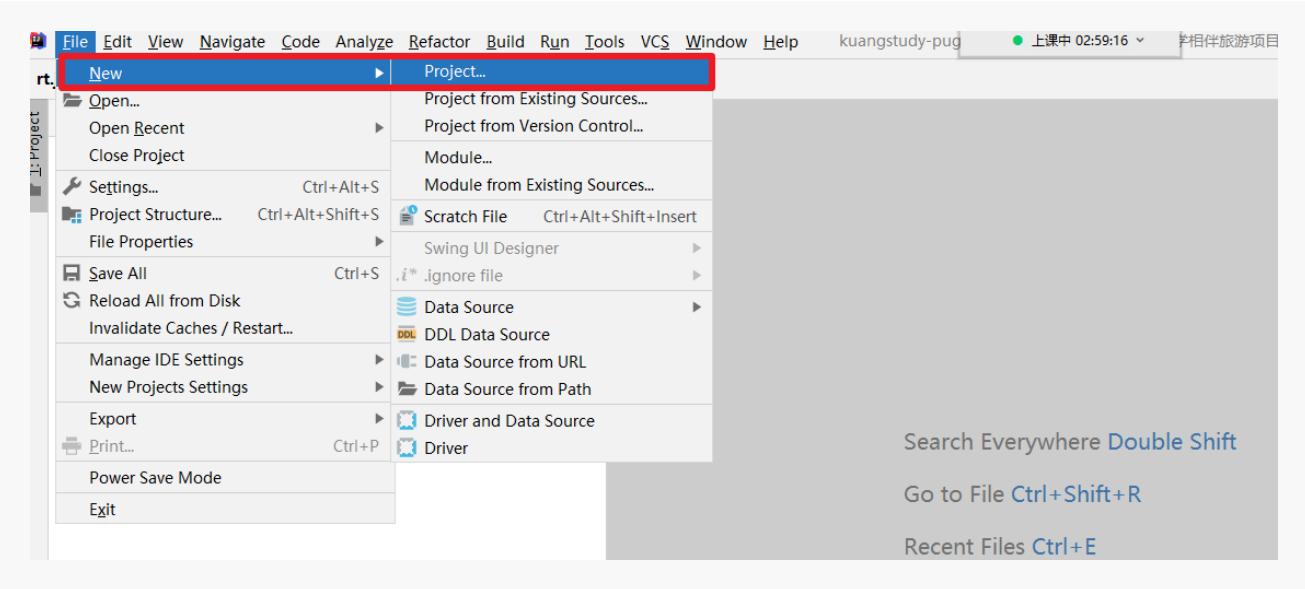
依赖架构问题和弊端是什么？

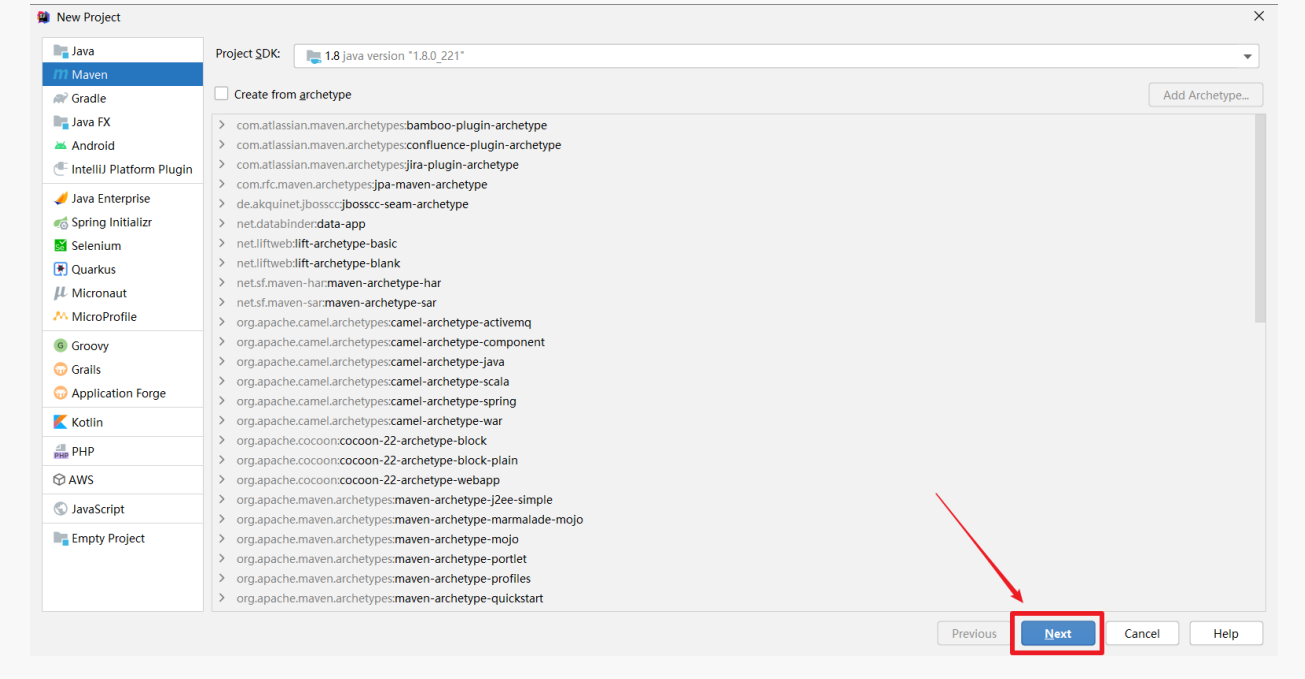
03、开发工具

- idea2020
- JDK1.8
- Maven3.6(不要用maven3.8)
- GIT
- MYSQL5.7

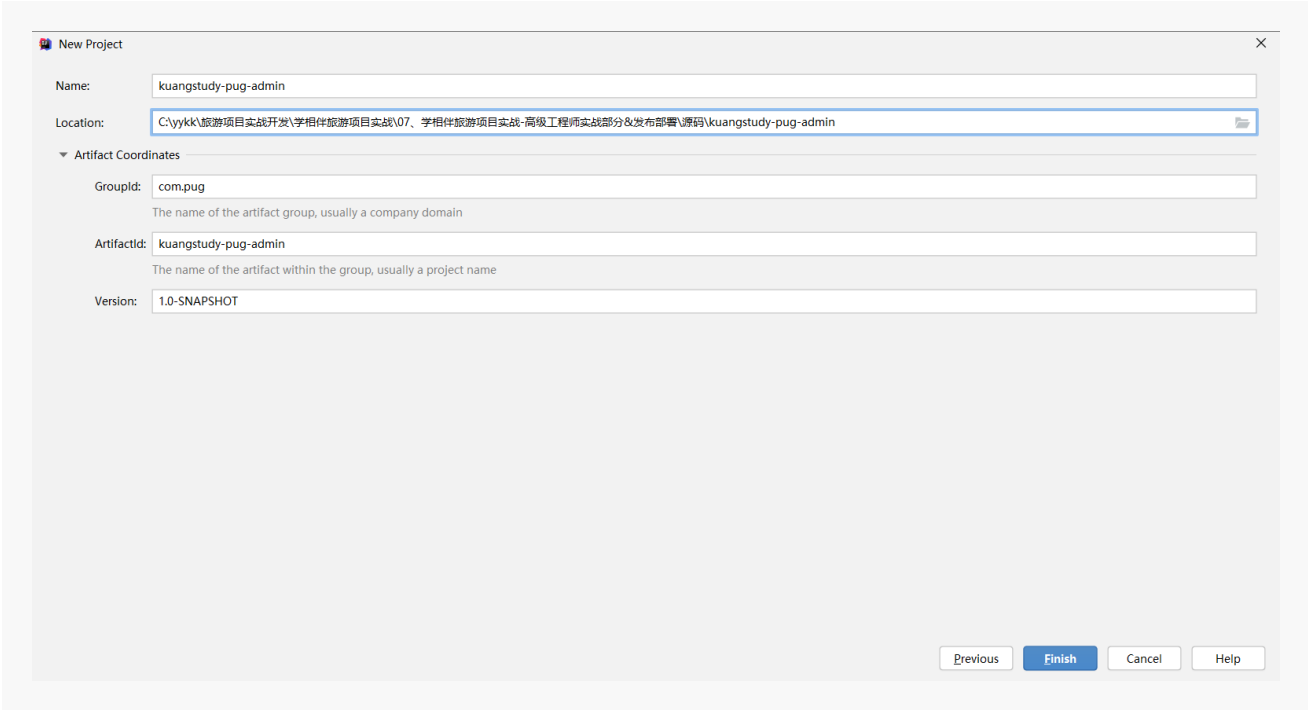
04 、具体搭建步骤

08-01、构建父工程



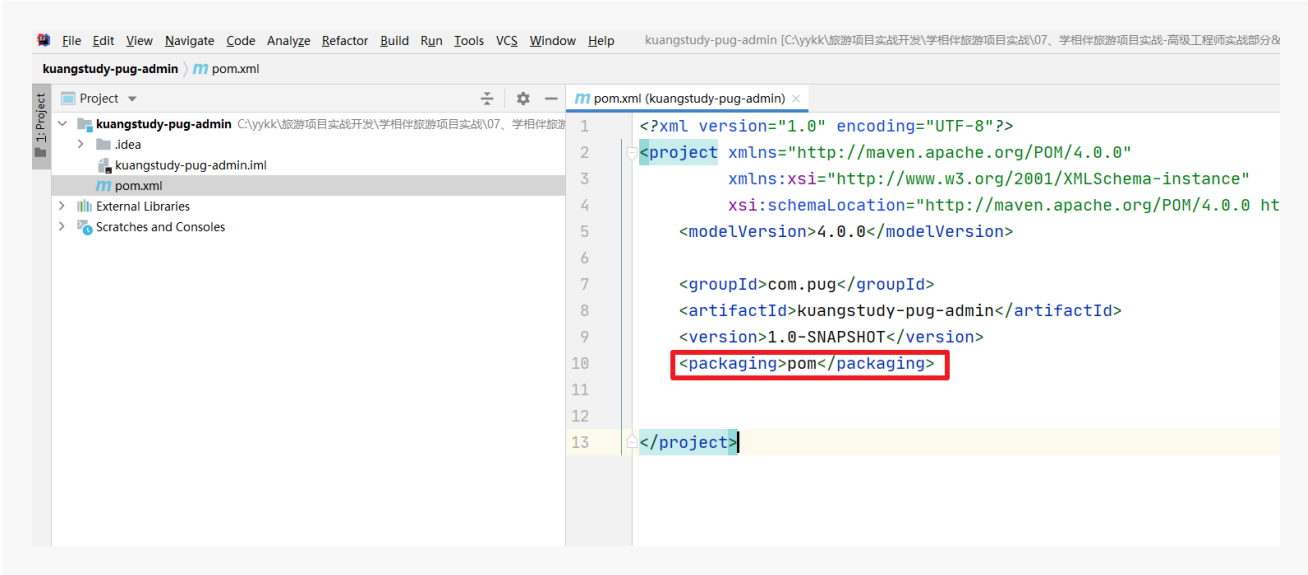


添加对应的工程名即可。



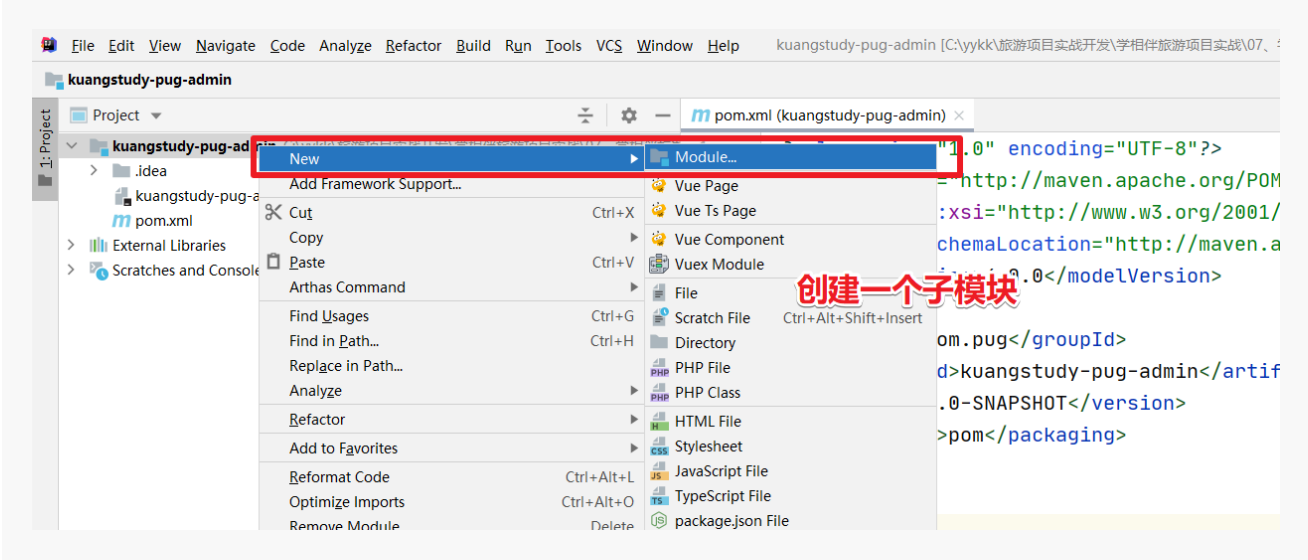
点击【finished】即可。

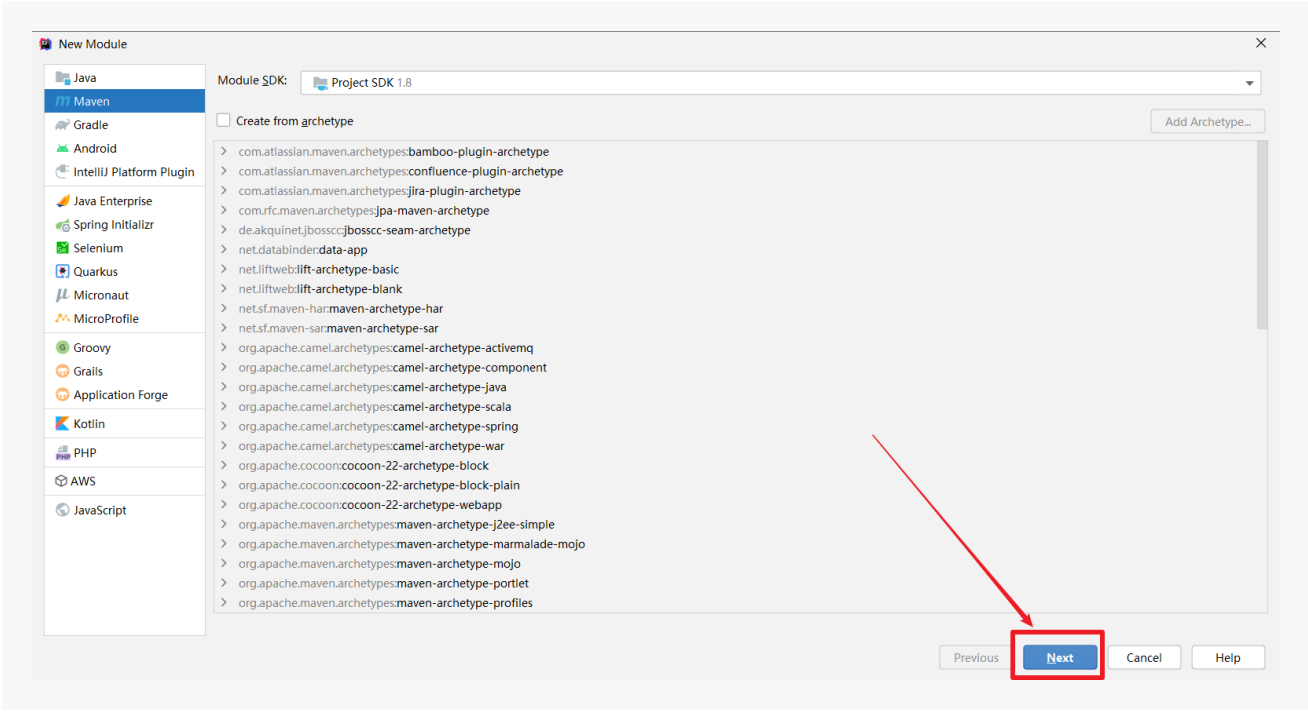
- 注意删除：*src*目录
- 然后把*pom.xml*中的*packaging*改成*pom*即可。

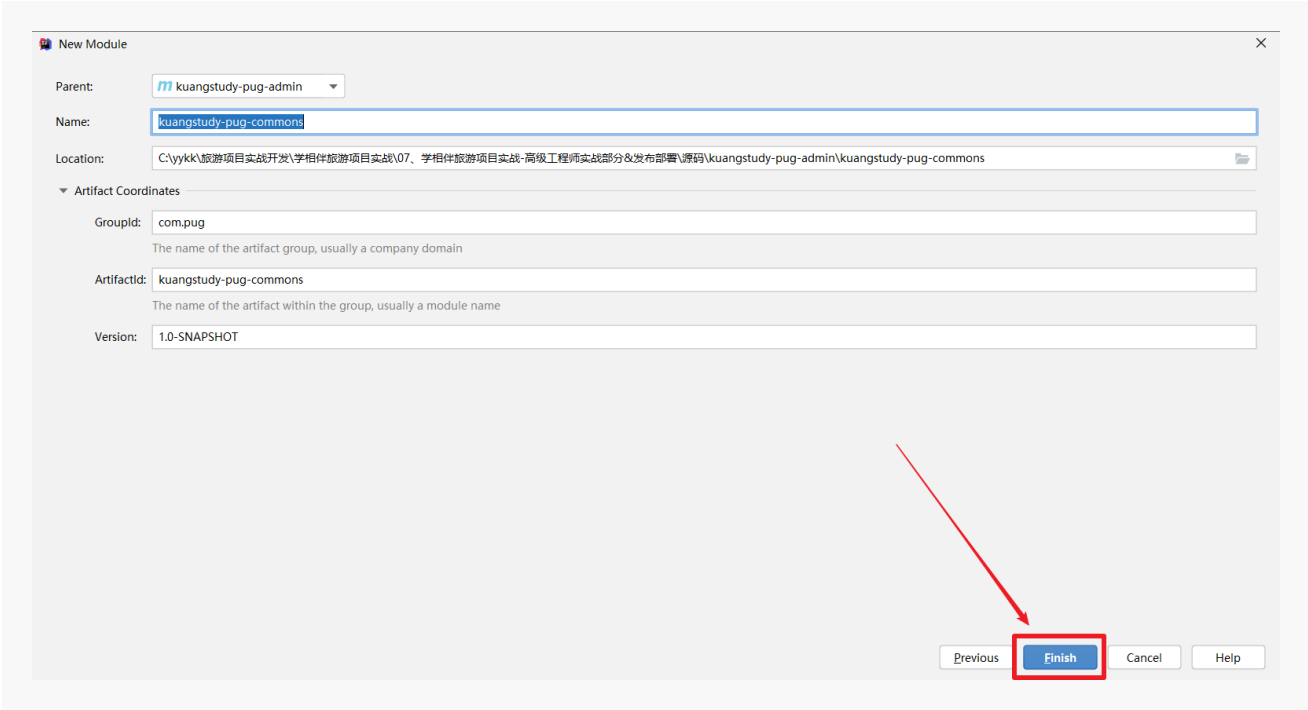


08-02、构建子工程

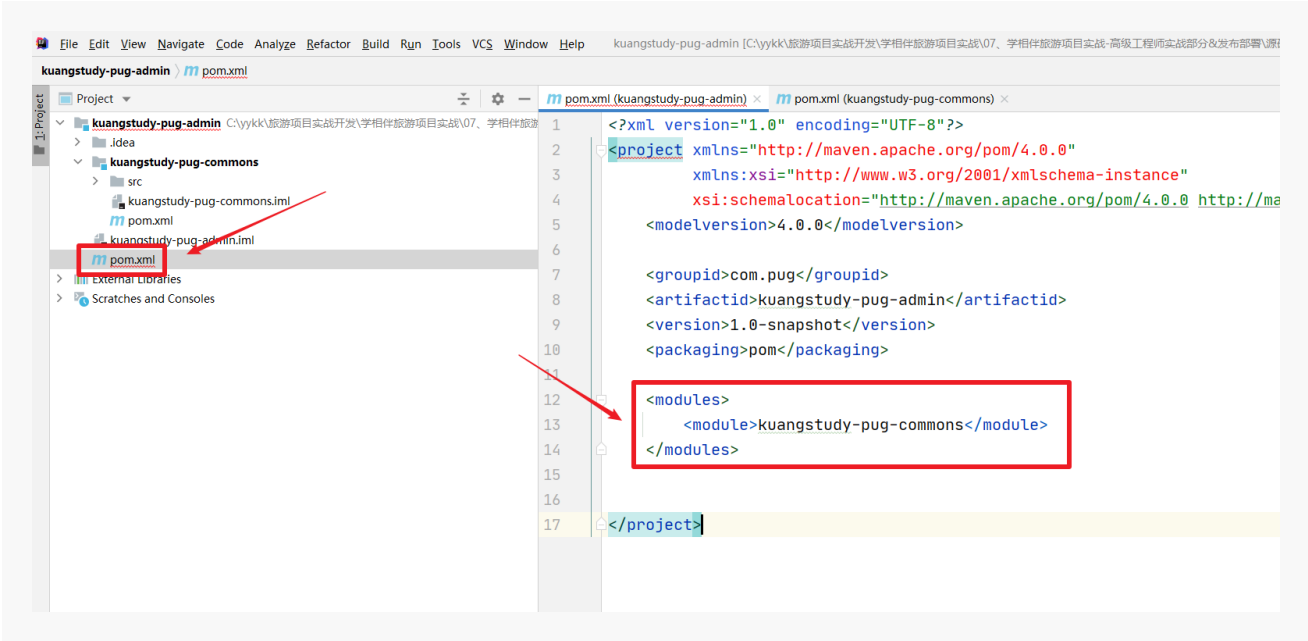
选中项目，【New】---【Module】如下：







创建好的子模块，会自动在父工程的pom.xml中注册一个module如下：

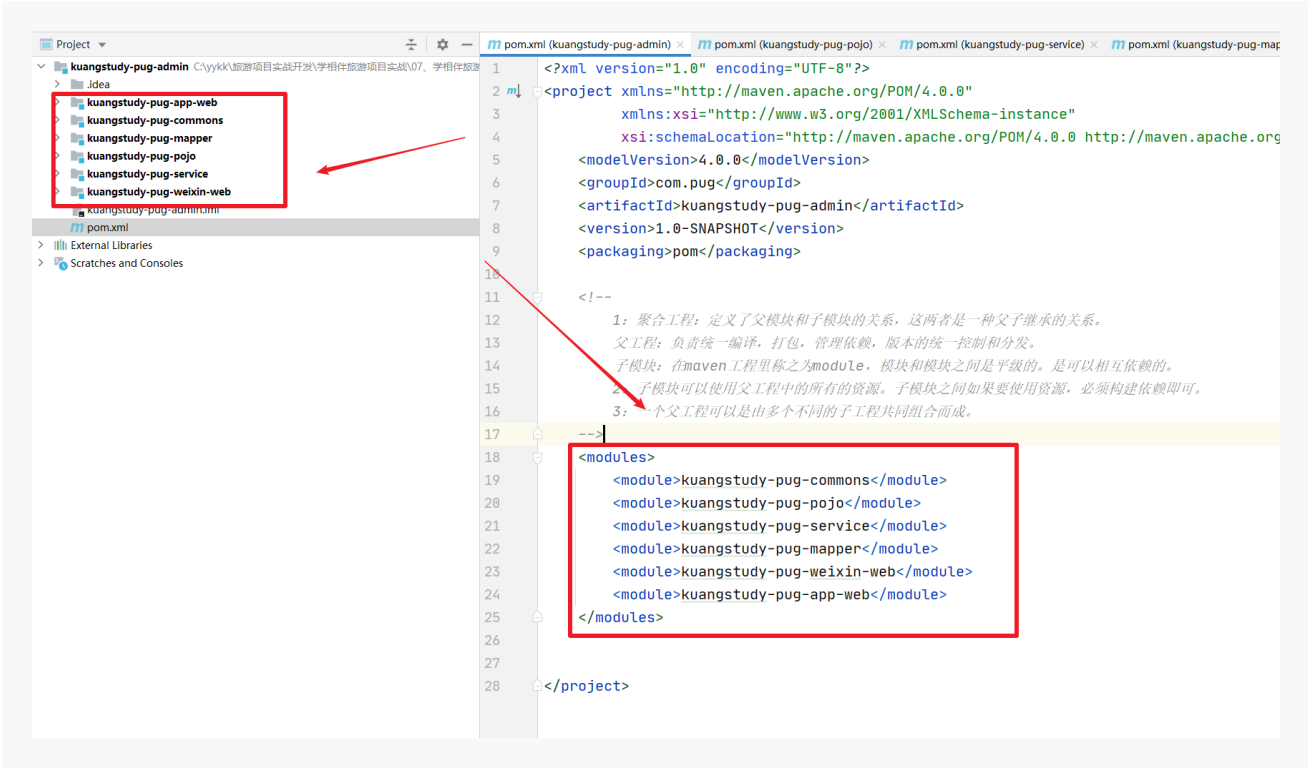


父pom.xml

```
<?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.pug</groupId>
    <artifactId>kuangstudy-pug-admin</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>pom</packaging>

    <!--
        1：聚合工程：定义了父模块和子模块的关系，这两者是一种父子继承的关系。
        父工程：负责统一编译，打包，管理依赖，版本的统一控制和分发。
        子模块：在maven工程里称之为module，模块和模块之间是平级的。是可以相互依赖的。
        2：子模块可以使用父工程中的所有的资源。子模块之间如果要使用资源，必须构建依赖即可。
        3：一个父工程可以是由多个不同的子工程共同组合而成。
    -->
    <modules>
        <module>kuangstudy-pug-commons</module>
    </modules>


</project>
```

以此类推其他的pojo、service、mapper、weixin-web、app-web都是一样的方式。如下：



08-03、整个的依赖关系

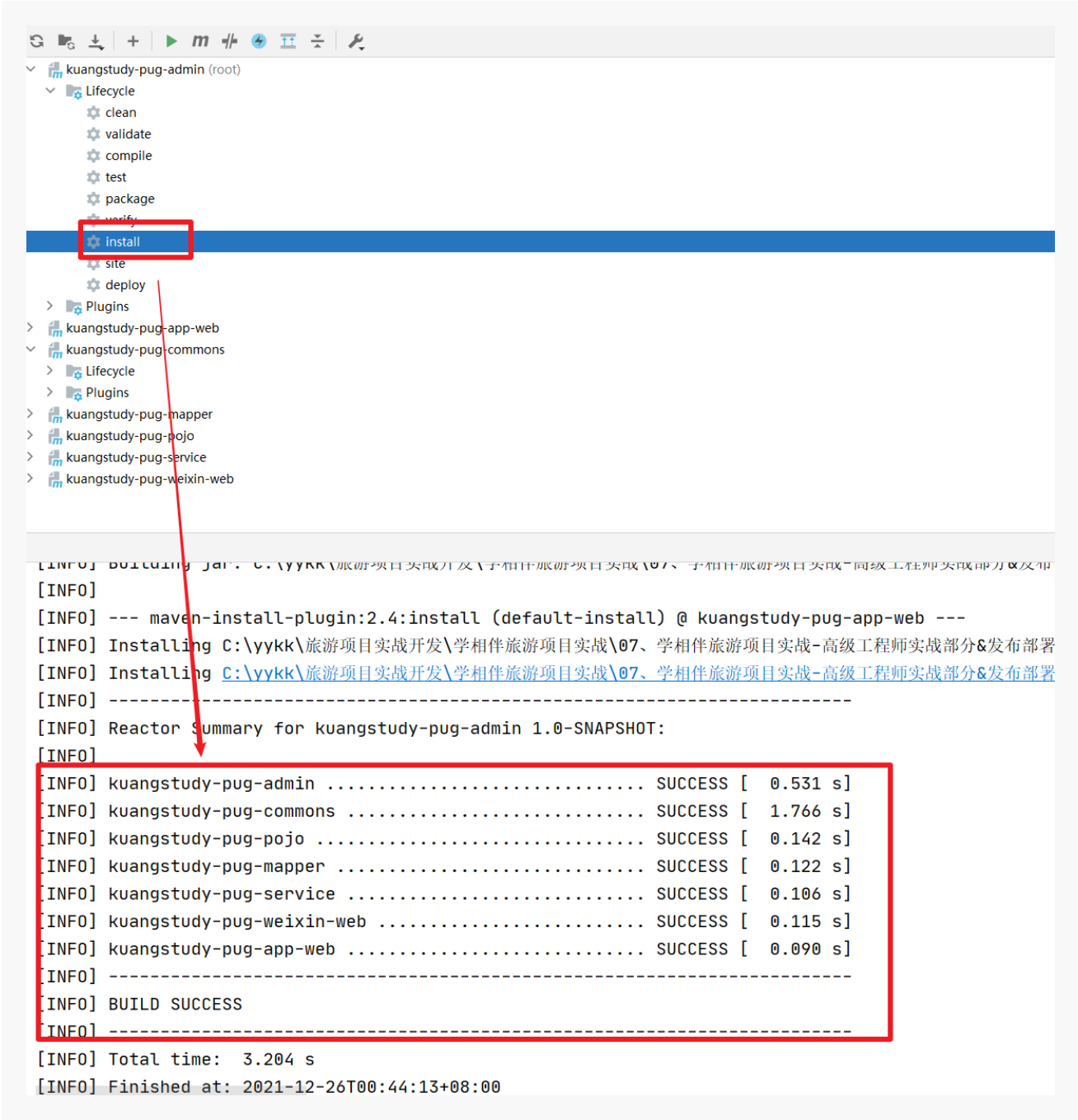
kuangstudy-pug-weixin-web -->kuangstudy-pug-service-->kuangstudy-pug-mapper-->kuangstudy-pug-pojo-->kuangstudy-pug-commons

模块之间是可以相互依赖的是平级的关系：

```
<?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">
    <parent>
        <artifactId>kuangstudy-pug-admin</artifactId>
        <groupId>com.pug</groupId>
        <version>1.0-SNAPSHOT</version>
    </parent>
    <modelVersion>4.0.0</modelVersion>
```

```
11
12      <artifactId>kuangstudy-pug-weixin-web</artifactId>
13      <dependencies>
14          <!--
15              web -> service -- commons --> mapper ->pojo ->
    commons
16              web -> common-redis 也可以去依赖其他的
17              web可以使用service、mapper、pojo和commons中所有的方法
    和类。
18          -->
19          <dependency>
20              <groupId>com.pug</groupId>
21              <artifactId>kuangstudy-pug-service</artifactId>
22              <version>1.0-SNAPSHOT</version>
23          </dependency>
24      </dependencies>
25
26  </project>
```

注意：现在模块之间的方法调用还暂时不可以，必须要执行install安装即可如下：



这样才才可以生效和使用模块之间的方法调用。

05、方案一：聚合springboot的依赖 -继承

继承解决方案 spring-boot-parent ----- kuangstudy-pug-parent 。

pom.xml文件如下：

```xml
<?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>
    <!--把springboot-作为父工程-->
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-dependencies</artifactId>
        <version>2.6.2</version>
    </parent>

    <groupId>com.ksd.pug</groupId>
    <artifactId>kuangstudy-pug-parent</artifactId>
    <version>1.0-SNAPSHOT</version>
    <!--父工程-->
    <packaging>pom</packaging>

    <modules>
        <module>kuangstudy-pug-commons</module>
        <module>kuangstudy-pug-pojo</module>
        <module>kuangstudy-pug-service</module>
        <module>kuangstudy-pug-mapper</module>
        <module>kuangstudy-pug-web</module>
        <module>kuangstudy-pug-web-app</module>
    </modules>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
        <java.version>1.8</java.version>
        <mysql.version>8.0.26</mysql.version>
        <lombok.version>1.18.20</lombok.version>
```

```
    </properties>

    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>mysql</groupId>
                <artifactId>mysql-connector-java</artifactId>
                <version>${mysql.version}</version>
            </dependency>
            <dependency>
                <groupId>org.projectlombok</groupId>
                <artifactId>lombok</artifactId>
                <version>${lombok.version}</version>
            </dependency>
        </dependencies>
    </dependencyManagement>
</project>
```

为什么上面的方式就可以把springboot继承过来呢？分析:

> 理解：如果一份*pom*父工程继承了另外一个*pom*的父工程。就会自动把另外一个*pom*的工程的依赖管理继承过来。

06、方案二：聚合springboot的依赖 -依赖管理继承

- 若依
- dubbo
- .....

如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
```

```
3           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5       <modelVersion>4.0.0</modelVersion>
6       <!--把springboot-作为父工程-->
7       <groupId>com.ksd.pug</groupId>
8       <artifactId>kuangstudy-pug-parent</artifactId>
9       <version>1.0-SNAPSHOT</version>
10      <!--父工程-->
11      <packaging>pom</packaging>
12
13      <modules>
14          <module>kuangstudy-pug-commons</module>
15          <module>kuangstudy-pug-pojo</module>
16          <module>kuangstudy-pug-service</module>
17          <module>kuangstudy-pug-mapper</module>
18          <module>kuangstudy-pug-web</module>
19          <module>kuangstudy-pug-web-app</module>
20      </modules>
21
22      <properties>
23          <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
24          <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
25          <java.version>1.8</java.version>
26          <mysql.version>8.0.26</mysql.version>
27          <lombok.version>1.18.20</lombok.version>
28      </properties>
29
30      <dependencyManagement>
31          <dependencies>
32
33              <dependency>
34                  <groupId>org.springframework.boot</groupId>
```

```
35                  <artifactId>spring-boot-
    dependencies</artifactId>
36                  <version>2.6.2</version>
37                  <scope>import</scope>
38                  <type>pom</type>
39              </dependency>
40
41              <dependency>
42                  <groupId>mysql</groupId>
43                  <artifactId>mysql-connector-java</artifactId>
44                  <version>${mysql.version}</version>
45              </dependency>
46              <dependency>
47                  <groupId>org.projectlombok</groupId>
48                  <artifactId>lombok</artifactId>
49                  <version>${lombok.version}</version>
50              </dependency>
51          </dependencies>
52      </dependencyManagement>
53  </project>
```

推荐用这种。

07、这种架构存在的问题是什么？

- common 是可以共用
- mapper service pojo 开发中是不可以共用

这种架构随着开发会出现如下问题：

- 前端的入口很多比如：小程序，APP，Tv, HTML5等。
- mapper service 很难区分哪些是小程序接口，哪些APP接口
- 如果APP有一些特殊的接口如何定制和区分。
- 如果一个service方法被多个共用的时候，可能引发解决一个问题，引发N个问题出现。
- 大家都在修改这个service层，会造成代码冲突。而且可能会造成误删。

解决方案

1： 改架构

2：冗余接口