尚硅谷大数据项目之电商实时数仓（电商

数据仓库系统）

（作者：尚硅谷研究院）

版本：V5.0

1. IDEA开发环境准备
   1. 项目架构与代码管理



* 1. 项目架构设计

本项目准备一个父工程，父工程下创建四个module，如下：

realtime-common：用于引入公共的第三方依赖，编写工具类、实体类等。

realtime-dim：用于编写DIM层业务代码。

realtime-dwd：用于编写DWD层业务代码。

realtime-dws：用于编写DWS层业务代码。

其中，后三个module统称为业务模块，业务模块都要将realtime-common模块作为依赖引入。

* + 1. 父工程
       - 1. 知识储备

dependencyManagement是Maven中的一个元素，它用于集中管理项目中所有依赖项的版本，而不是真正地声明这些依赖。当项目中有多个模块时，可以在父POM中集中管理所有模块的依赖版本。这确保了所有模块都使用相同的依赖版本，从而避免了版本冲突。当子模块添加已被管理的依赖时不需要声明版本，因为它已经在父工程定义了。

* + - * 1. POM文件

创建父工程，将打包时不必包含在内的依赖添加到父工程的pom文件中，这部分依赖的scope均为provided，主要有三类：

* Flink相关：集群Flink的lib目录下存在的依赖，包括Flink运行环境及日志框架Log4j有关的依赖。
* Hadoop相关：hadoop-client和hadoop-auth。Flink的检查点保存在HDFS，hadoop-client用于和HDFS交互。本项目还要读写HBase，操作HBase需要引入hadoop-auth。
* Lombok依赖：Lombok是一个Java库，它提供了一系列注解，旨在通过自动化某些常见的样板代码生成来简化Java代码。通过使用Lombok，开发者可以减少大量重复和冗长的代码，从而使得代码更加简洁、易读和易维护。

前两类都是集群已有的依赖，因而打包时不必包含，而Lombok只在编译时生效，编译完成后代码的运行和它再无关系，打包时也不需要包含在内。

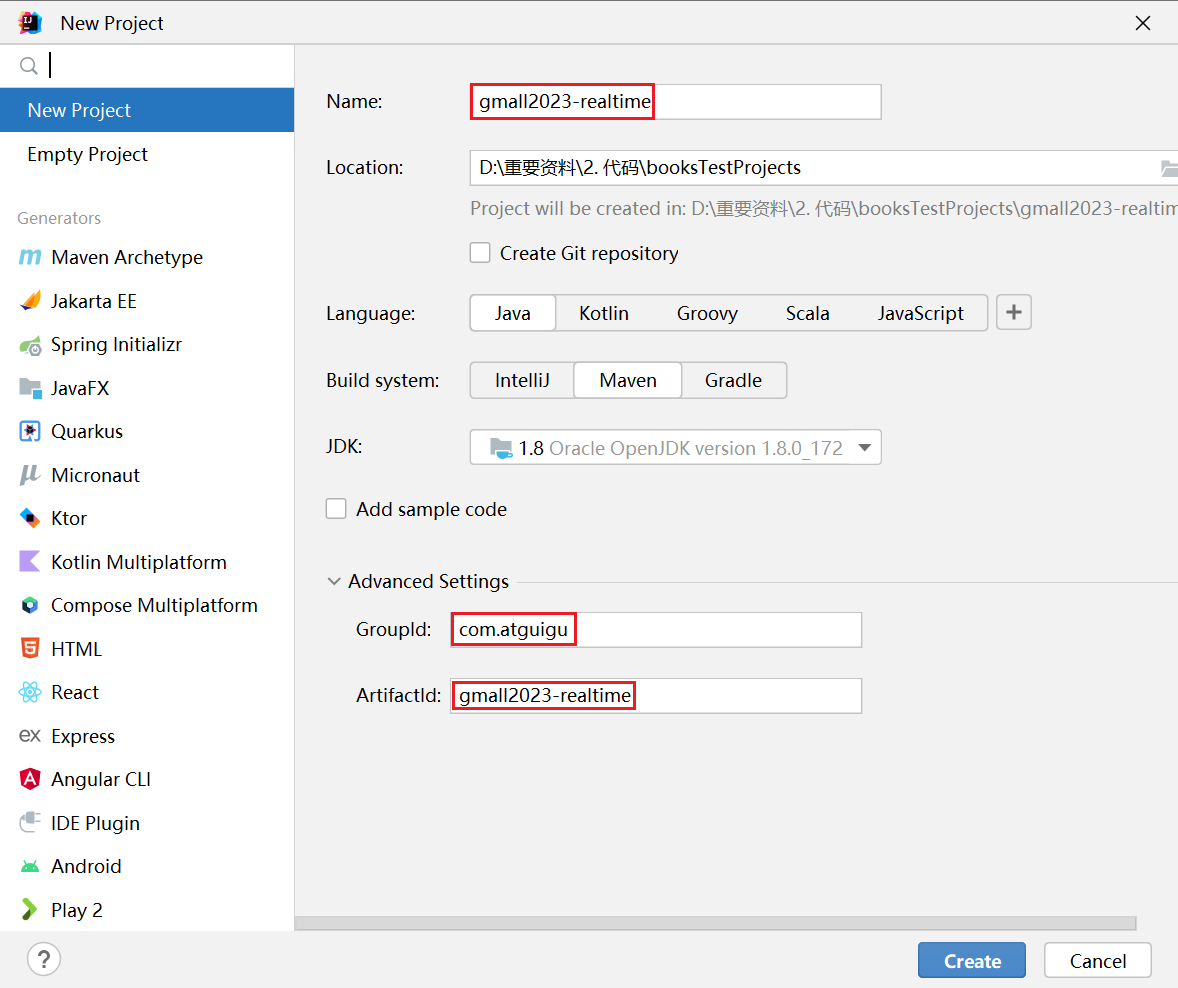
在父工程的POM文件中通过dependencyManagement标签声明所有依赖的版本。

* + 1. realtime-common

本项目需要引入大量依赖，除去父类中引入的部分（集群已有的依赖），还有很多集群没有的依赖（下文称为第三方依赖），打包时都要包含在jar包里，这样就会导致业务模块打包时jar包很大，打包时间很长，部署效率不高。为了改善这一情况，我们构建公共模块realtime-common，将第三方依赖都在公共模块的POM文件中引入，业务模块只需要引入realtime-common依赖即可。但要注意，如果realtime-common的scope为默认的compile，业务模块打包时依然会包含所有的第三方依赖，不能解决问题，所以我们将scope更改为provided，并将打好的realtime-common包放到集群Flink的lib目录下。

为了精简子模块的代码，我们将基类、工具类、实体类、常量类、通用的自定义函数全部放到realtime-common中。

* 1. 创建gmall2023-realtime
     1. 创建项目



* + 1. 修改pom文件

在pom.xml中添加以下内容

<**properties**>

<**java.version**>1.8</**java.version**>

<**maven.compiler.source**>${java.version}</**maven.compiler.source**>

<**maven.compiler.target**>${java.version}</**maven.compiler.target**>

<**project.build.sourceEncoding**>UTF-8</**project.build.sourceEncoding**>

<**flink.version**>1.17.1</**flink.version**>

<**scala.version**>2.12</**scala.version**>

<**hadoop.version**>3.3.4</**hadoop.version**>

<**flink-cdc.vesion**>2.4.2</**flink-cdc.vesion**>

<**fastjson.version**>1.2.83</**fastjson.version**>

<**hbase.version**>2.4.11</**hbase.version**>

</**properties**>

<**dependencies**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-streaming-java</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-clients</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**><!--在 idea 运行的时候,可以打开 web 页面-->

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-runtime-web</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-json</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-csv</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<!--如果保存检查点到hdfs上，需要引入此依赖-->

<**dependency**>

<**groupId**>org.apache.hadoop</**groupId**>

<**artifactId**>hadoop-client</**artifactId**>

<**version**>${hadoop.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.hadoop</**groupId**>

<**artifactId**>hadoop-auth</**artifactId**>

<**scope**>provided</**scope**>

<**exclusions**>

<**exclusion**>

<**groupId**>org.slf4j</**groupId**>

<**artifactId**>slf4j-reload4j</**artifactId**>

</**exclusion**>

</**exclusions**>

</**dependency**>

<**dependency**>

<**groupId**>org.projectlombok</**groupId**>

<**artifactId**>lombok</**artifactId**>

<**version**>1.18.20</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<!--Flink默认使用的是slf4j记录日志，相当于一个日志的接口,我们这里使用log4j作为具体的日志实现-->

<**dependency**>

<**groupId**>org.slf4j</**groupId**>

<**artifactId**>slf4j-api</**artifactId**>

<**version**>1.7.25</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.logging.log4j</**groupId**>

<**artifactId**>log4j-to-slf4j</**artifactId**>

<**version**>2.14.0</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-table-api-java-bridge</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-table-planner-loader</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-table-runtime</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-connector-files</**artifactId**>

<**version**>${flink.version}</**version**>

<**scope**>provided</**scope**>

</**dependency**>

</**dependencies**>

<**dependencyManagement**>

<**dependencies**>

<**dependency**>

<**groupId**>com.alibaba</**groupId**>

<**artifactId**>fastjson</**artifactId**>

<**version**>${fastjson.version}</**version**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-connector-kafka</**artifactId**>

<**version**>${flink.version}</**version**>

</**dependency**>

<**dependency**>

<**groupId**>com.ververica</**groupId**>

<**artifactId**>flink-connector-mysql-cdc</**artifactId**>

<**version**>${flink-cdc.vesion}</**version**>

</**dependency**>

<!-- hbase 依赖-->

<**dependency**>

<**groupId**>org.apache.hbase</**groupId**>

<**artifactId**>hbase-client</**artifactId**>

<**version**>${hbase.version}</**version**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.hadoop</**groupId**>

<**artifactId**>hadoop-auth</**artifactId**>

<**version**>${hadoop.version}</**version**>

<**scope**>provided</**scope**>

<**exclusions**>

<**exclusion**>

<**groupId**>org.slf4j</**groupId**>

<**artifactId**>slf4j-reload4j</**artifactId**>

</**exclusion**>

</**exclusions**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-connector-hbase-2.2</**artifactId**>

<**version**>${flink.version}</**version**>

</**dependency**>

<!-- https://mvnrepository.com/artifact/org.apache.doris/flink-doris-connector-1.17 -->

<**dependency**>

<**groupId**>org.apache.doris</**groupId**>

<**artifactId**>flink-doris-connector-1.17</**artifactId**>

<**version**>1.5.2</**version**>

</**dependency**>

<**dependency**>

<**groupId**>commons-beanutils</**groupId**>

<**artifactId**>commons-beanutils</**artifactId**>

<**version**>1.9.4</**version**>

</**dependency**>

<**dependency**>

<**groupId**>com.janeluo</**groupId**>

<**artifactId**>ikanalyzer</**artifactId**>

<**version**>2012\_u6</**version**>

</**dependency**>

<**dependency**>

<**groupId**>redis.clients</**groupId**>

<**artifactId**>jedis</**artifactId**>

<**version**>3.3.0</**version**>

</**dependency**>

<**dependency**>

<**groupId**>io.lettuce</**groupId**>

<**artifactId**>lettuce-core</**artifactId**>

<**version**>6.2.4.RELEASE</**version**>

</**dependency**>

</**dependencies**>

</**dependencyManagement**>

<**build**>

<**plugins**>

<**plugin**>

<**groupId**>org.apache.maven.plugins</**groupId**>

<**artifactId**>maven-shade-plugin</**artifactId**>

<**version**>3.1.1</**version**>

<**executions**>

<**execution**>

<**phase**>package</**phase**>

<**goals**>

<**goal**>shade</**goal**>

</**goals**>

<**configuration**>

<**artifactSet**>

<**excludes**>

<**exclude**>com.google.code.findbugs:jsr305</**exclude**>

<**exclude**>org.slf4j:\*</**exclude**>

<**exclude**>log4j:\*</**exclude**>

<**exclude**>org.apache.hadoop:\*</**exclude**>

</**excludes**>

</**artifactSet**>

<**filters**>

<**filter**>

<!-- Do not copy the signatures in the META-INF folder.Otherwise, this might cause SecurityExceptions when using the JAR. -->

<!-- 打包时不复制META-INF下的签名文件，避免报非法签名文件的SecurityExceptions异常-->

<**artifact**>\*:\*</**artifact**>

<**excludes**>

<**exclude**>META-INF/\*.SF</**exclude**>

<**exclude**>META-INF/\*.DSA</**exclude**>

<**exclude**>META-INF/\*.RSA</**exclude**>

</**excludes**>

</**filter**>

</**filters**>

<**transformers** combine.children="append">

<!-- The service transformer is needed to merge META-INF/services files -->

<!-- connector和format依赖的工厂类打包时会相互覆盖，需要使用ServicesResourceTransformer解决-->

<**transformer**

implementation="org.apache.maven.plugins.shade.resource.ServicesResourceTransformer"/>

</**transformers**>

</**configuration**>

</**execution**>

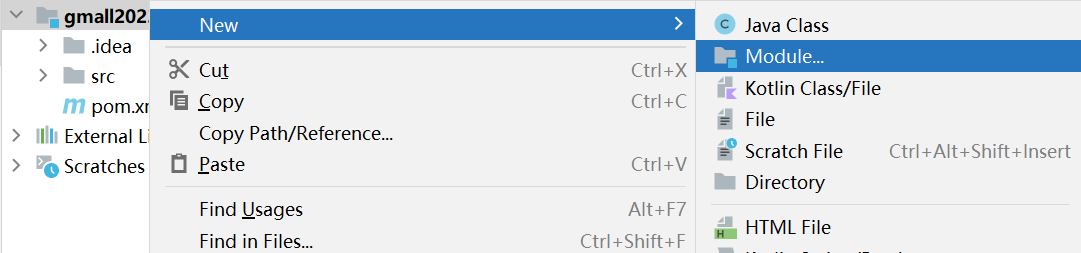
</**executions**>

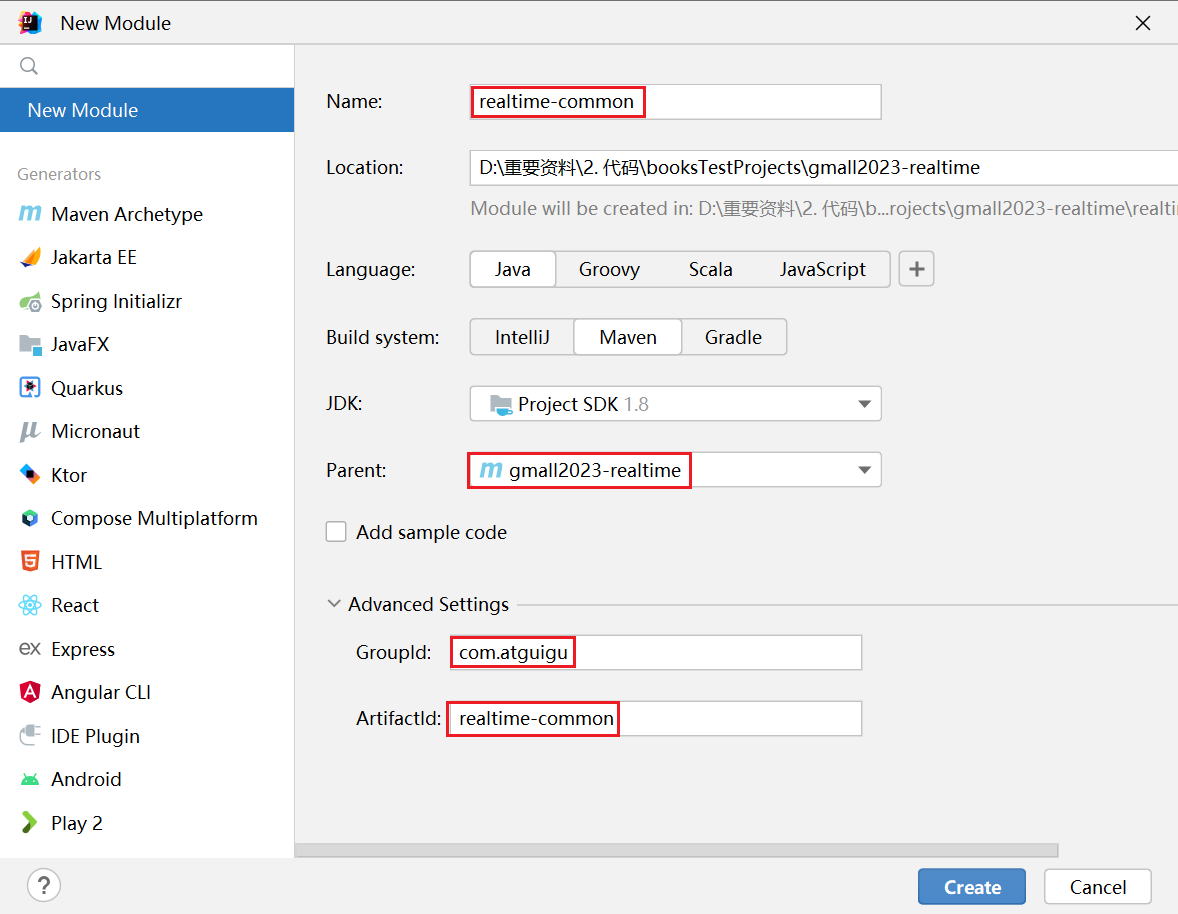
</**plugin**>

</**plugins**>

</**build**>

* 1. 在gmall2023-realtime下创建子模块realtime-common
     1. 创建module





* + 1. 引入依赖

<**dependencies**>

<**dependency**>

<**groupId**>com.alibaba</**groupId**>

<**artifactId**>fastjson</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-connector-kafka</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>com.ververica</**groupId**>

<**artifactId**>flink-connector-mysql-cdc</**artifactId**>

</**dependency**>

<!-- hbase 依赖-->

<**dependency**>

<**groupId**>org.apache.hbase</**groupId**>

<**artifactId**>hbase-client</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>org.apache.flink</**groupId**>

<**artifactId**>flink-connector-hbase-2.2</**artifactId**>

</**dependency**>

<!-- https://mvnrepository.com/artifact/org.apache.doris/flink-doris-connector-1.17 -->

<**dependency**>

<**groupId**>org.apache.doris</**groupId**>

<**artifactId**>flink-doris-connector-1.17</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>commons-beanutils</**groupId**>

<**artifactId**>commons-beanutils</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>com.janeluo</**groupId**>

<**artifactId**>ikanalyzer</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>redis.clients</**groupId**>

<**artifactId**>jedis</**artifactId**>

</**dependency**>

<**dependency**>

<**groupId**>io.lettuce</**groupId**>

<**artifactId**>lettuce-core</**artifactId**>

</**dependency**>

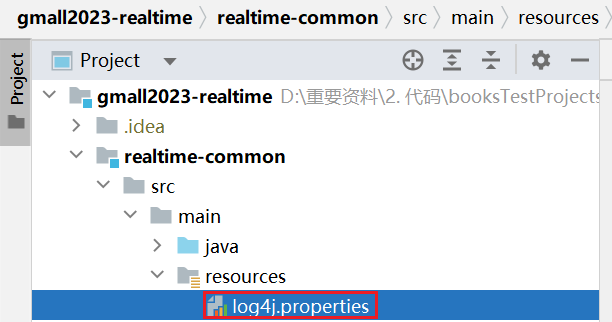
</**dependencies**>

* + 1. 在resources目录下创建log4j.properties文件

Flink使用log4j作为默认的日志框架，启动应用程序，控制台会打印日志，但有时默认的日志级别并不能满足我们的需求，此时就要在module的resources目录下创建log4j.properties文件，在其中对日志级别等做出调整。

本项目有数十个module，如果每个module下都准备一份log4j.properties文件，将会非常繁琐。实际上，本项目的module只有两类：common模块和业务模块，而所有的业务模块都依赖的common模块，只需要在common模块的resources目录下准备一份log4j配置文件即可对所有模块生效。

* + - * 1. 在realtime-common模块的resources目录下创建log4j.properties



* + - * 1. 写入以下内容

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.target=System.out

log4j.appender.stdout.layout=org.apache.log4j.PatternLayout

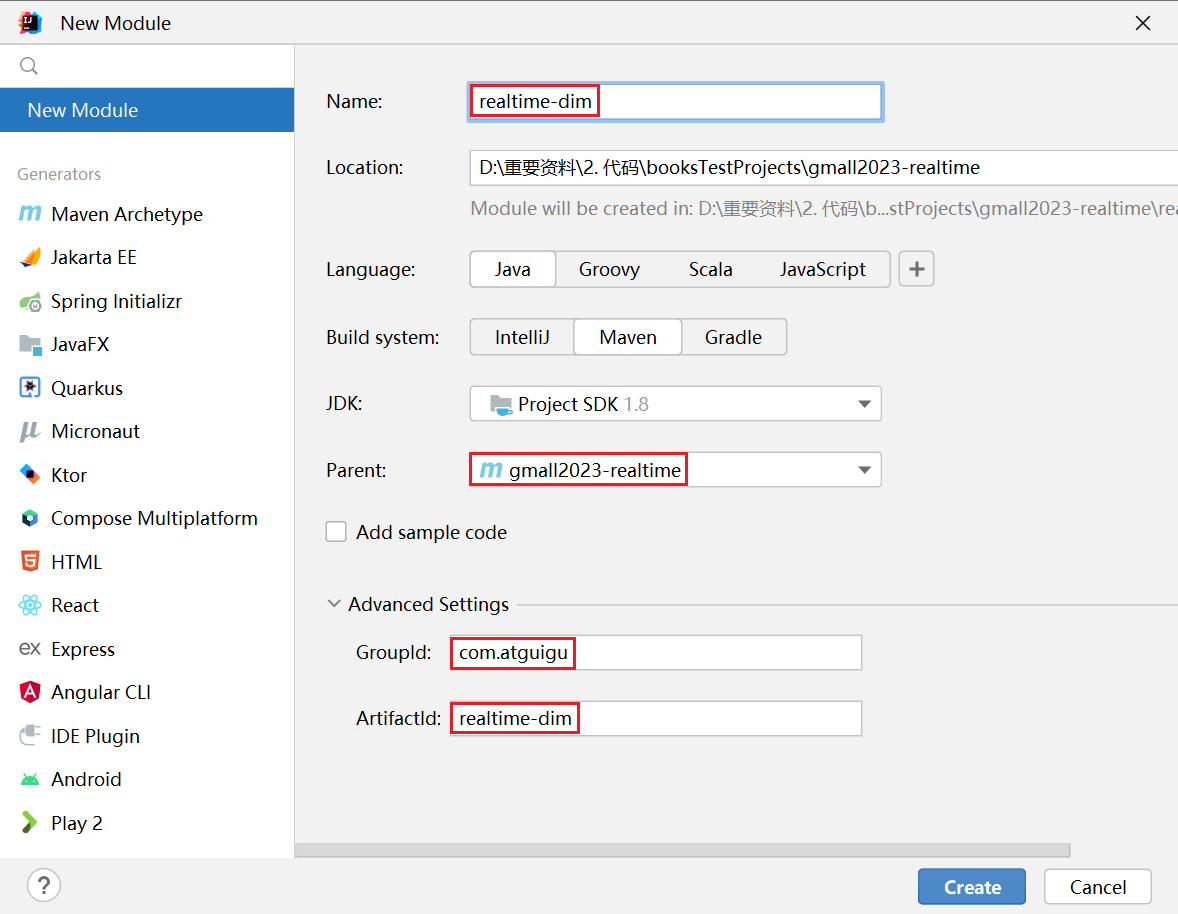
log4j.appender.stdout.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %10p (%c:%M) - %m%n

log4j.rootLogger=error,stdout

* 1. 创建子模块realtime-dim

如前所述，业务模块只需要依赖realtime-common即可，下文不再赘述。

* + 1. 创建module



* + 1. 导入依赖

<**dependencies**>

<**dependency**>

<**groupId**>com.atguigu</**groupId**>

<**artifactId**>realtime-common</**artifactId**>

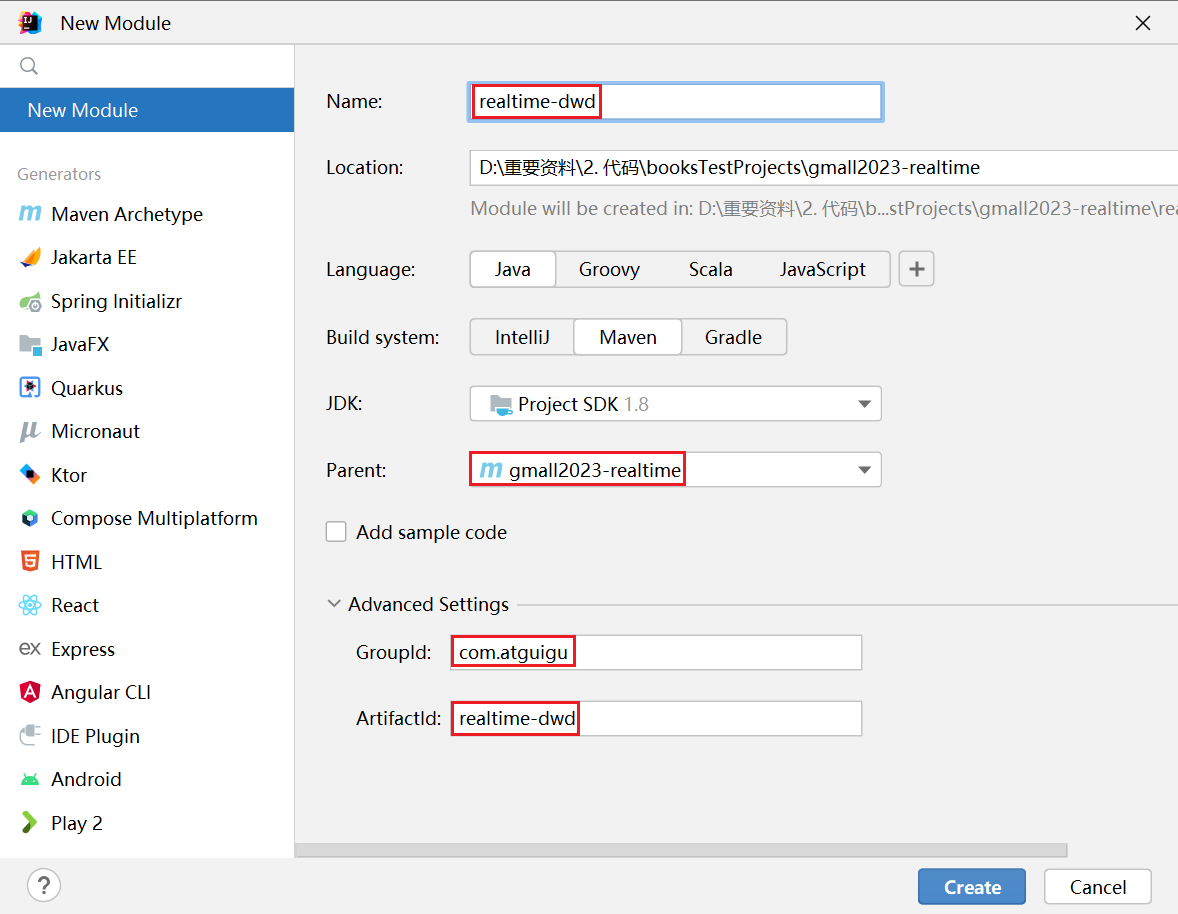
<**version**>1.0-SNAPSHOT</**version**>

<**scope**>provided</**scope**>

</**dependency**>

</**dependencies**>

* 1. 创建子模块realtime-dwd
     1. 创建module



* + 1. 导入依赖

<**dependencies**>

<**dependency**>

<**groupId**>com.atguigu</**groupId**>

<**artifactId**>realtime-common</**artifactId**>

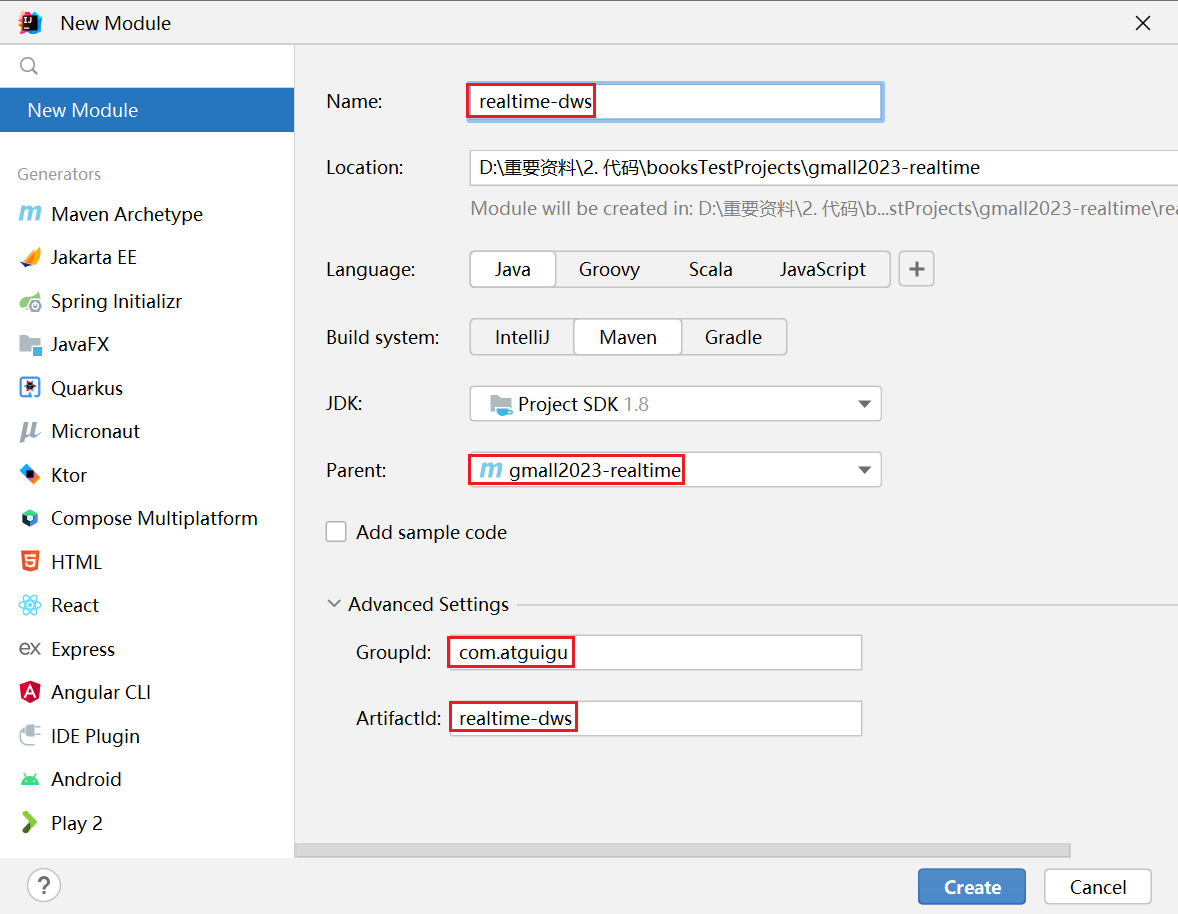
<**version**>1.0-SNAPSHOT</**version**>

<**scope**>provided</**scope**>

</**dependency**>

</**dependencies**>

* 1. 创建子模块realtime-dws
     1. 创建module



* + 1. 导入依赖

<**dependencies**>

<**dependency**>

<**groupId**>com.atguigu</**groupId**>

<**artifactId**>realtime-common</**artifactId**>

<**version**>1.0-SNAPSHOT</**version**>

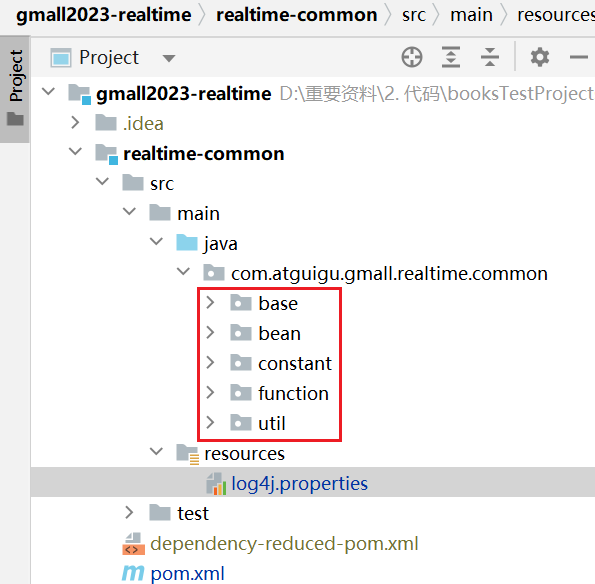
<**scope**>provided</**scope**>

</**dependency**>

</**dependencies**>

* 1. package设计

在realtime-common包下创建如下包结构



* + - * 1. base包

包含所有业务代码的基类。

* + - * 1. bean包

包含所有实体类。

* + - * 1. constant包

包含所有常量类。

* + - * 1. function包

包含所有函数类。

* + - * 1. util包

包含所有工具类。

1. GitLab环境准备
   1. GitLab集成
      1. GitLab部署和IDEA集成
         * 1. 部署和集成

参见以下文档。

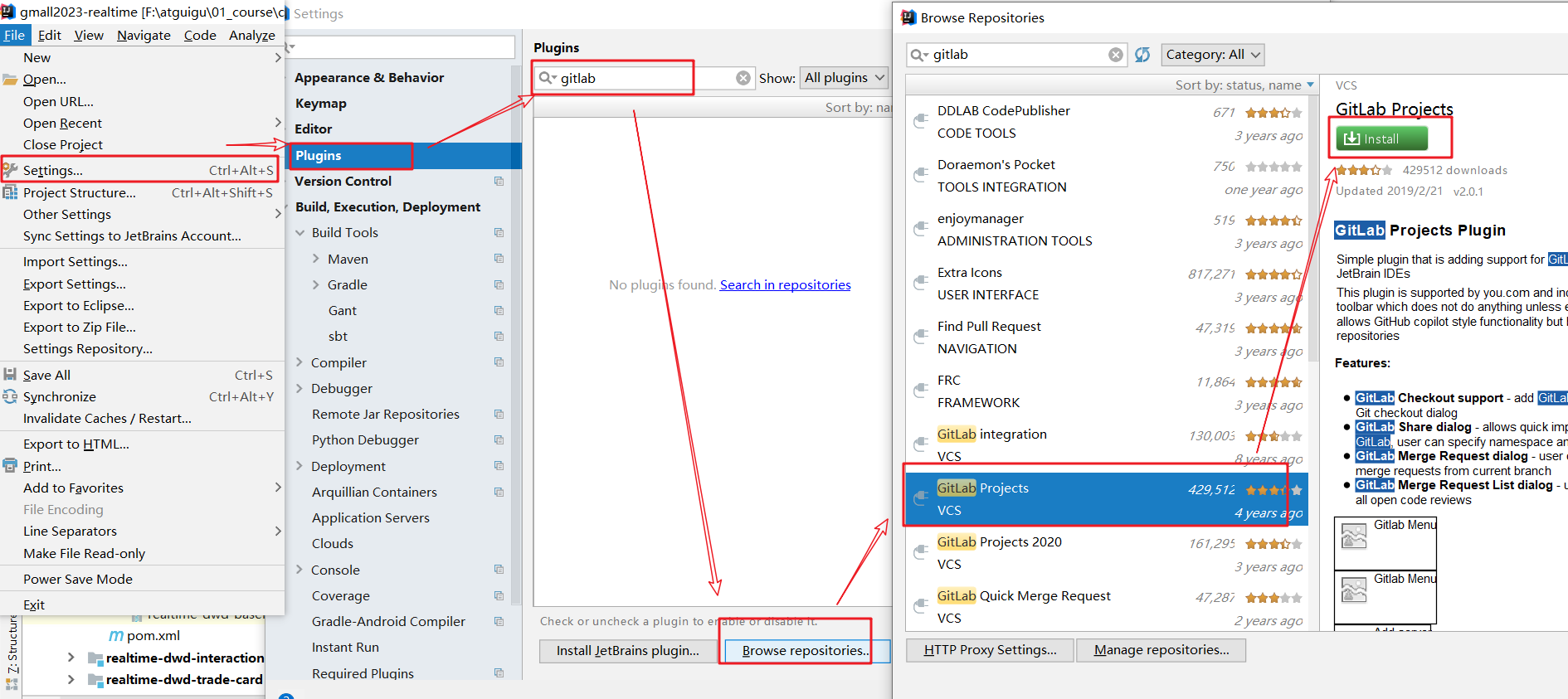


* + 1. 初始化项目

确保已完成GitLab部署及IDEA和GitLab的集成。

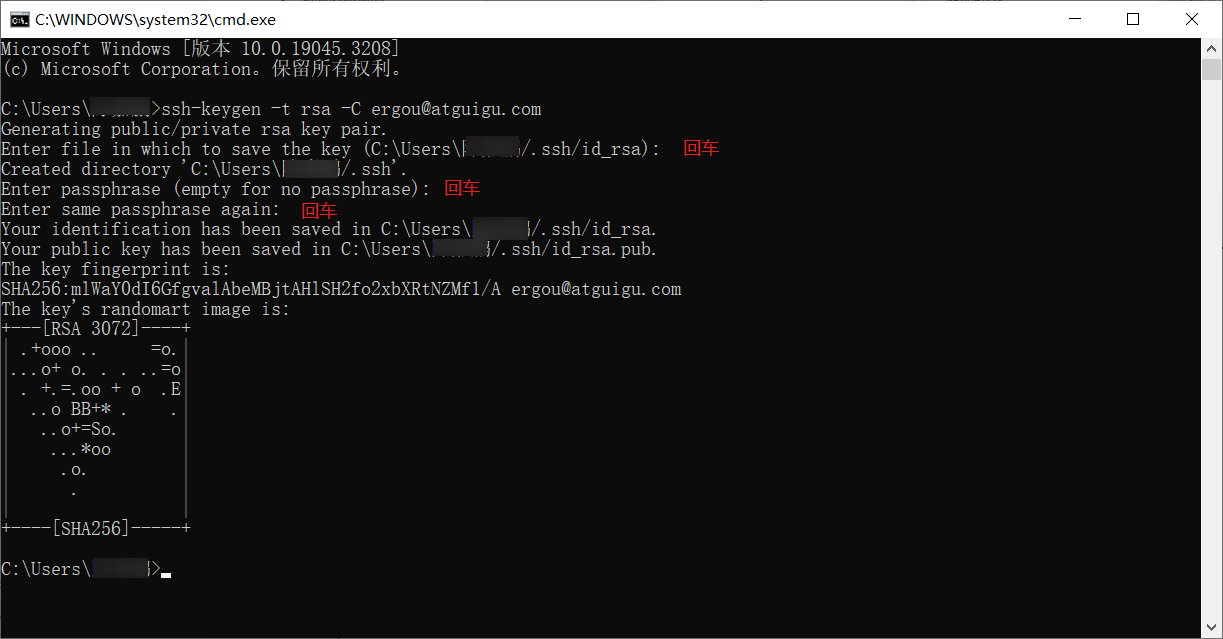
我们将实时数仓项目分享到GitLab上：

* + - * 1. 安装 GitLab 插件



* + - * 1. 配置 SSH 免密登录

ssh-keygen -t rsa -C ergou@atguigu.com



到用户目录下.ssh查看id\_rsa.pub文件

图形用户界面, 应用程序

描述已自动生成

图片包含 文本

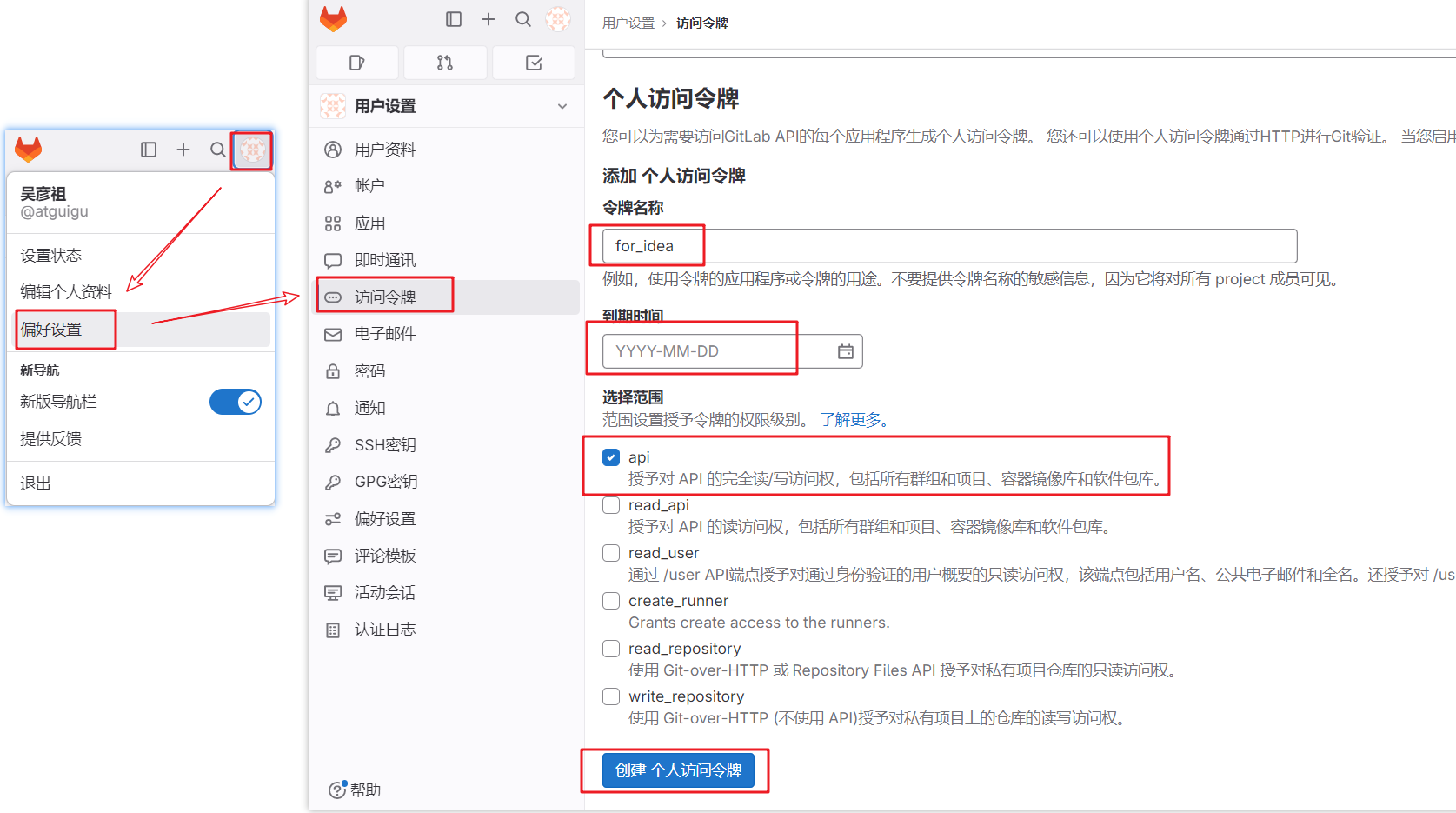
描述已自动生成

添加到GitLab中：

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

* + - * 1. 获取 GitLab 个人令牌



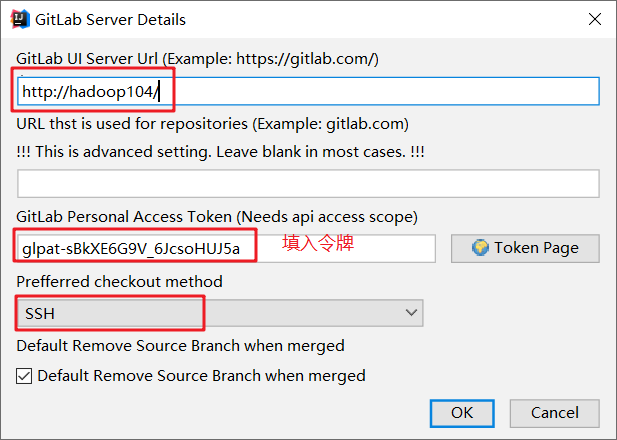
创建后，可以查看和复制生成的token：



* + - * 1. 添加 GitLab 服务

图形用户界面, 应用程序

描述已自动生成

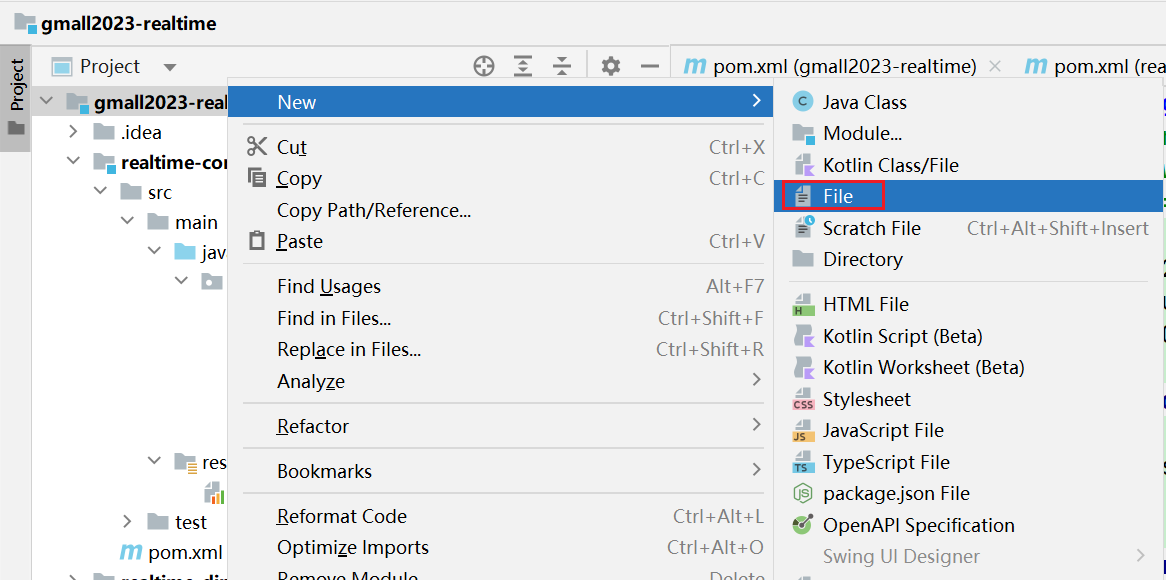


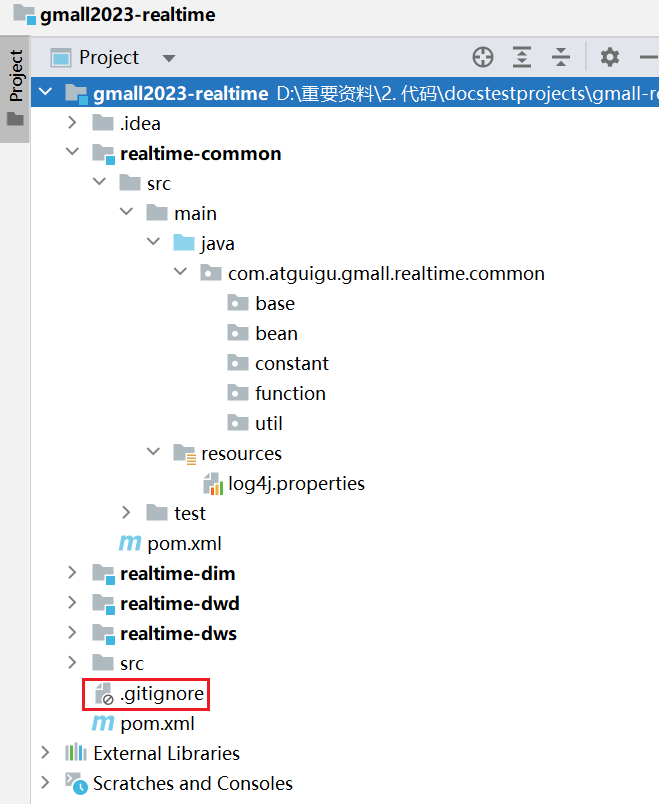
* + - * 1. 修改默认分支的保护策略



* + - * 1. 创建.gitignore

在父工程根目录下创建.gitignore文件





写入以下内容

/\*\*/target

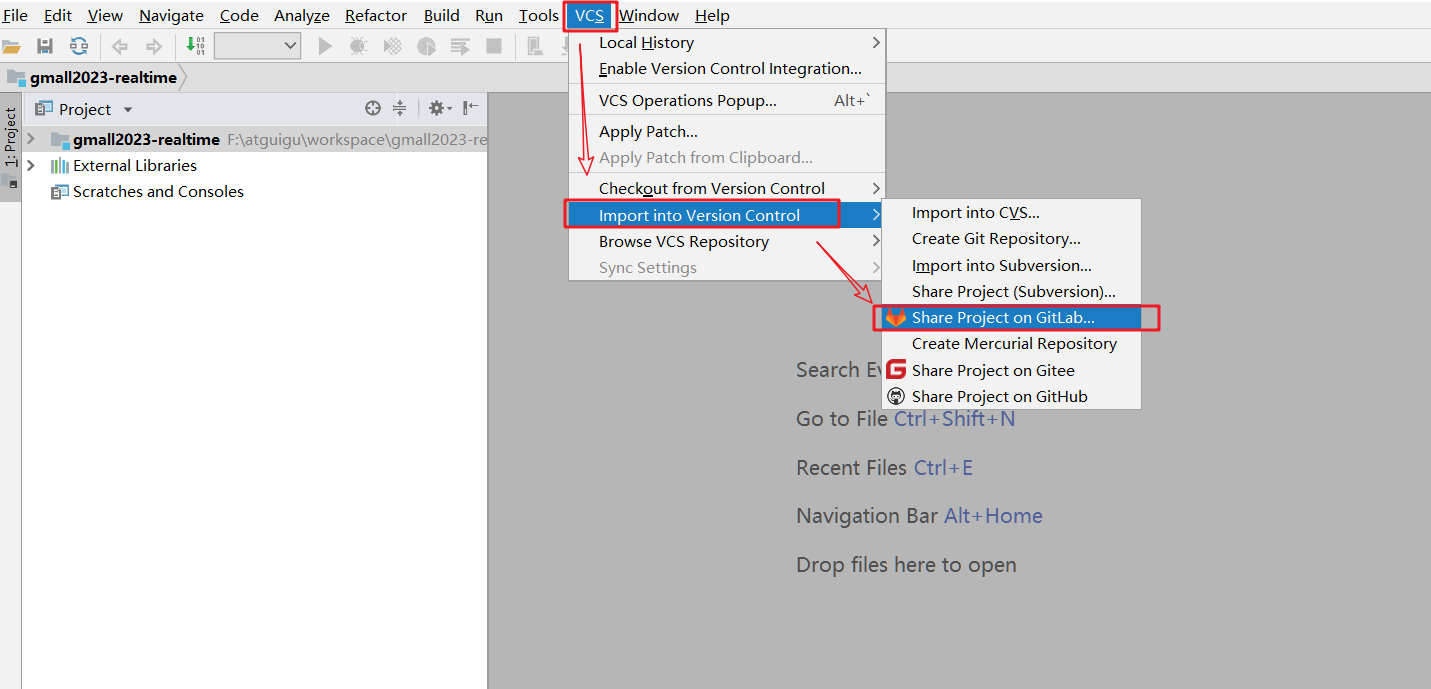
/\*\*/.idea

\*.iml

\*.class

\*\*/\*/dependency-reduced-pom.xml

* + - * 1. 分享到GitLab



有了token，才能获取group列表，才能选择group：



GitLab的bigdata群组下，可以看到分享的项目：

表格

描述已自动生成

团队其他人员，克隆项目可以通过GitLab页面快速操作：

图形用户界面, 应用程序

描述已自动生成

1. 数据仓库涉及的组件环境准备
   1. Flink环境搭建

Flink集群部署请参考如下文档。



* 1. Hbase环境搭建

Hbase 集群部署请参考如下文档。



* 1. Redis环境搭建
     1. Redis 安装

Redis 安装请参考如下文档



* + 1. Redis配置文件修改
       - 1. 切换到 atguigu 用户，拷贝 redis.conf 配置文件，置于 atguigu 家目录下

若已按照 1 中文档操作，则这一步可以省略

[root@hadoop102 atguigu]# su atguigu

[atguigu@hadoop102 redis-6.0.8]$ cp /opt/module/redis-6.0.8/redis.conf ~/my\_redis.conf

* + - * 1. 注销 bind 配置项

# bind 127.0.0.1

* + - * 1. 修改 protected-mode

protected-mode no

* 1. Doris环境搭建

Doris安装请参考如下文档

