***帆软报表插件开发步骤***

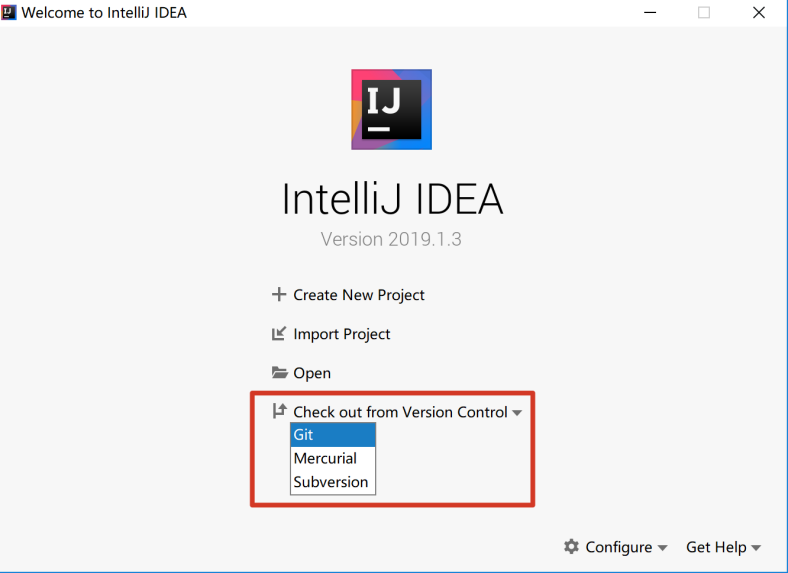
### **帆软报表设计器介绍**

FineReport是一款企业级web报表软件，易学易用，功能强大，简单拖拽操作便可制作中国式复杂报表，轻松实现报表的多样展示、交互分析、数据录入、权限管理、定时调度、打印输出、门户管理和移动应用等

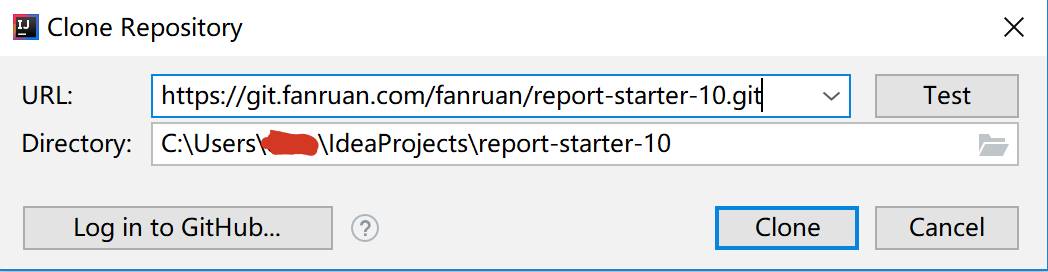
### 下载报表设计器

#### 1）工程下载

直接打开已经安装好的IntelliJ IDEA，启动好之后，可以看到如下图的界面，点击其中的 Checkout from Version Control ，选择Git



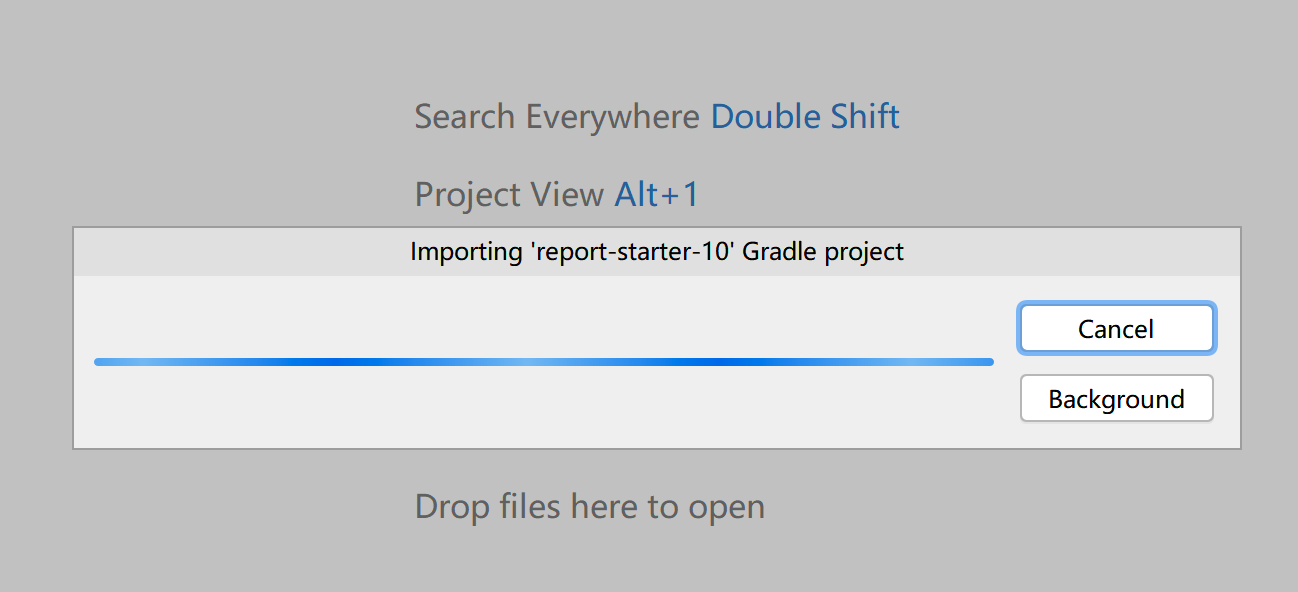
在弹出框中输入插件开发工程的地址：<https://code.fanruan.com/fanruan/report-starter-10.git>



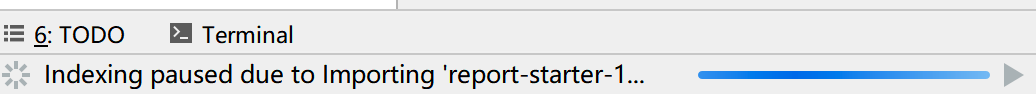
点击Clone按钮下载插件开发工程即可。

#### 2）打开工程

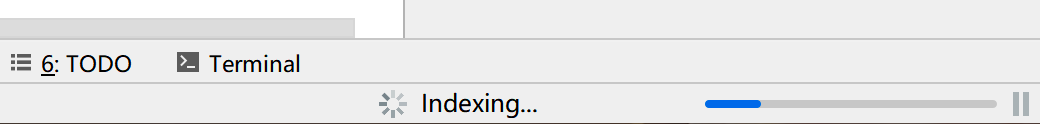
在等待一段时间后，插件开发工程下载完毕，会自动打开和下载依赖jar，这个时间可能比较长，可以去泡杯茶



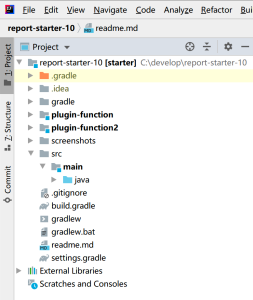
可以点击Background，让导入任务进入后台运行



等待IntelliJ IDEA自动下载依赖JAR包，之后再等待索引完毕，就可以开始使用插件开发工程了

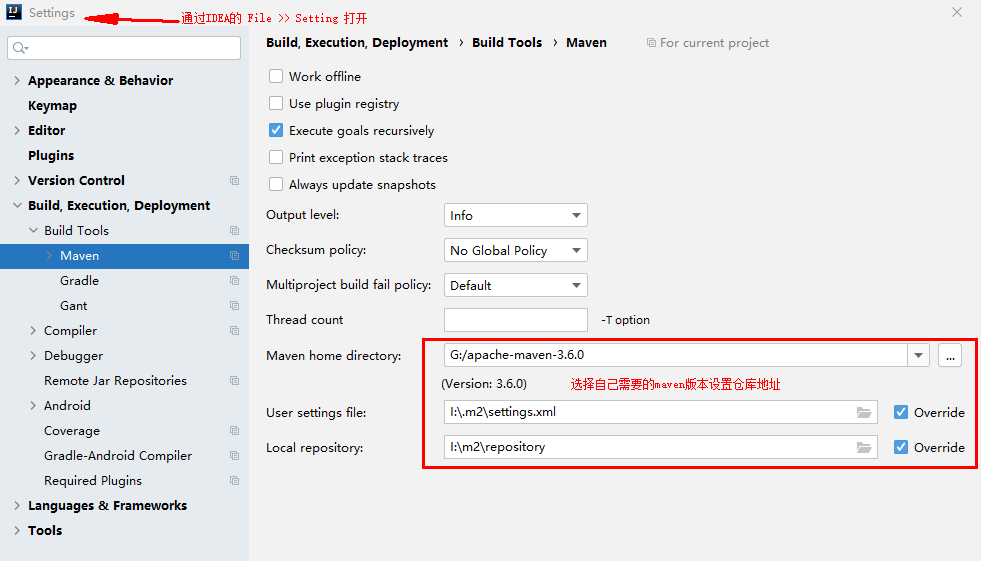


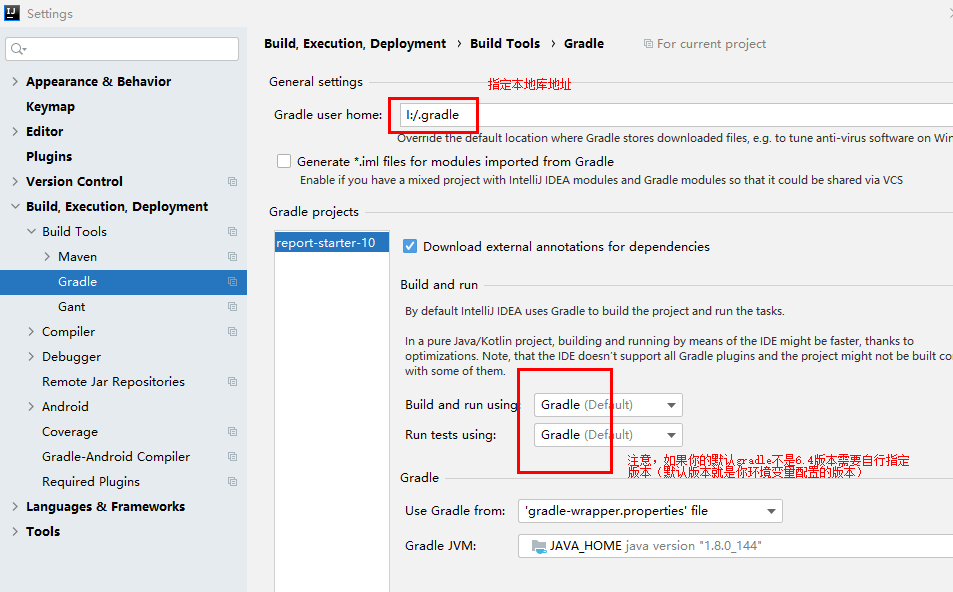
已经准备就绪的开发工程如下图所示



 若无法自动依赖tools.jar，需手动配置

上面的步骤完成后，根据我们的需要自行决定是否需要修改 maven/gradle的本地库地址（因为windows默认是保存在C盘的，我的C盘空间不够大，所以我都单独配置了响应的地址）





### 安装gradle工具

安装Gradle

Gradle用于管理插件开发工程的配置、依赖jar包的下载以及构建插件安装包，参考[Gradle安装教程](https://wiki.jikexueyuan.com/project/gradle-2-user-guide/installing.html)完成安装。

下载地址1：[https://fine-doc.oss-cn-shanghai.aliyuncs.com/tools/gradle-6.4-bin.zip（帆软提供）](https://fine-doc.oss-cn-shanghai.aliyuncs.com/tools/gradle-6.4-bin.zip)

下载地址2：[https://services.gradle.org/distributions（官网）](https://services.gradle.org/distributions)

完成安装后，可以在控制台执行命令

|  |
| --- |
| gradle -v |

输出大致如下

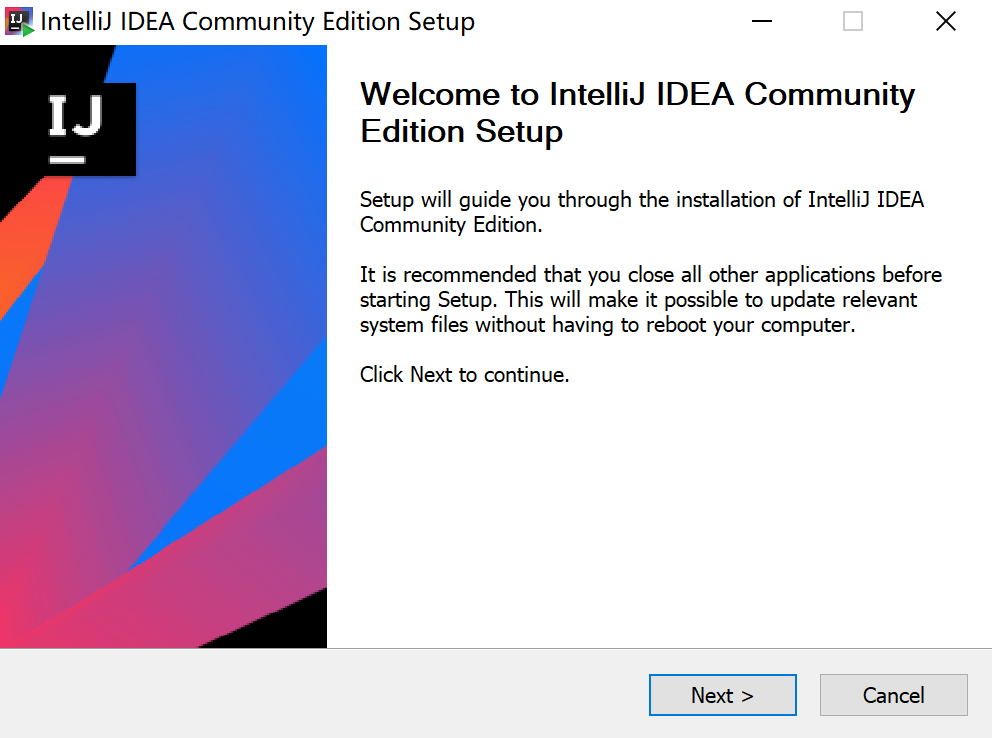
|  |
| --- |
| ------------------------------------------------------------  Gradle 4.7  ------------------------------------------------------------    Build time:   2018-04-18 09:09:12 UTC  Revision:     b9a962bf70638332300e7f810689cb2febbd4a6c    Groovy:       2.4.12  Ant:          Apache Ant(TM) version 1.9.9 compiled on February 2 2017  JVM:          1.8.0\_192 (Oracle Corporation 25.192-b12)  OS:           Mac OS X 10.14.6 x86\_64 |

### 配置idea开发环境

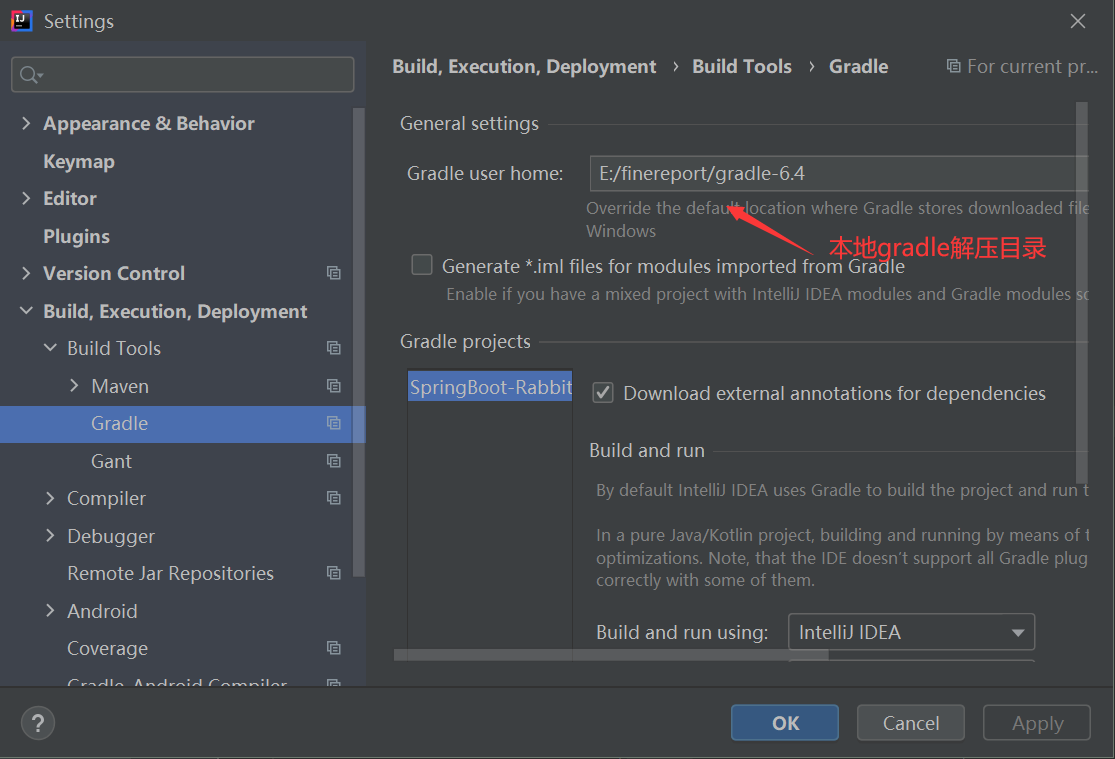
安装IntelliJ IDEA

IntelliJ IDEA是作为插件开发的IDE工具，直接到[Jetbrains官方网站](https://www.jetbrains.com/idea/download)，下载社区版本即可，也可以直接到下表中的地址下载，下载完毕直接安装即可

|  |  |
| --- | --- |
| windows平台 | [ideaIC-2020.2.exe](https://fine-doc.oss-cn-shanghai.aliyuncs.com/tools/ideaIC-2020.2.exe) |
| macOS平台 | [ideaIC-2020.2.dmg](https://fine-doc.oss-cn-shanghai.aliyuncs.com/tools/ideaIC-2020.2.dmg) |



安装完idea2020版本后，需要配置前面gradle安装目录，工程默认采用的gradle进行编译和打包，配置界面如下：



Gradle基本命令:

* 帮助命令  
  gradle --help
* 查看版本  
  gradle -v
* 执行特定的任务  
  gradle [taskName]
* 构建  
  gradle build
* 跳过测试构建构建  
  gradle build -x test
* 继续执行任务而忽略前面失败的任务  
  gradle build --continue
* 试运行build  
  gradle -m build
* 产生build运行时间的报告  
  gradle build --profile  
  结果存储在build/report/profile目录，名称为build运行的时间。
* 显示任务间的依赖关系  
  gradlle tasks --all
* 查看testCompile的依赖关系  
  gradle -q dependencies --configuration testCompile
* 清空所有编译、打包生成的文件(即：清空build目录)  
  gradle clean
* 使用指定的Gradle文件调用任务  
  gradle -b [file\_path] [task]
* 使用指定的目录调用任务  
  gradle -q -p [dir] helloWorld
* Gradle的图形界面  
  gradle --gui

### 开发插件包

#### 1）新增插件包

新建插件

1、在report-starter-10目录下，新建一个目录，这里假设为plugin-hello-world；

2、将plugin-function中的**plugin.xml**（用于描述插件接入点信息）以及**build.gradle**（用于管理插件jar包依赖）、**encrypt.xml**复制到plugin-hello-world目录下，然后对plugin.xml的内容稍作修改：

修改后的文件内容如下

**plugin.xml** 展开源码

3、在plugin-hello-world下新建一个用于存放java源码的目录和一个用于存放java源码以外的其他资源文件的目录，分别为

src/main/java

和

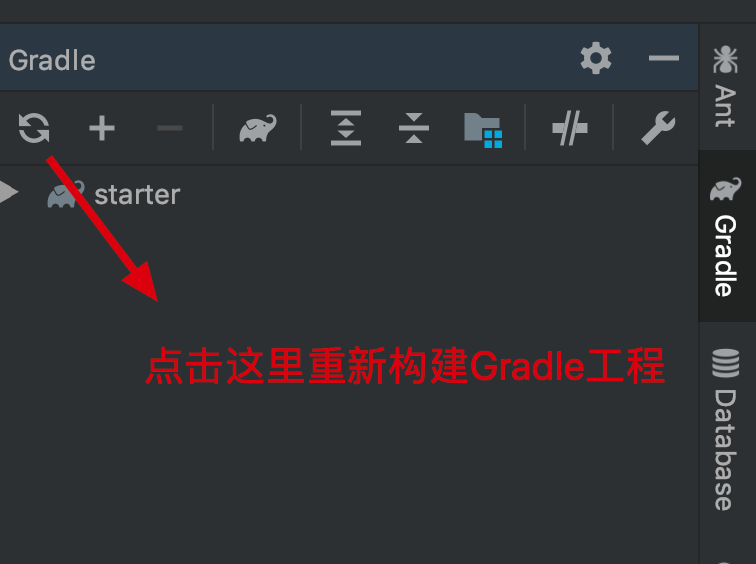
src/main/resources

插件依赖管理

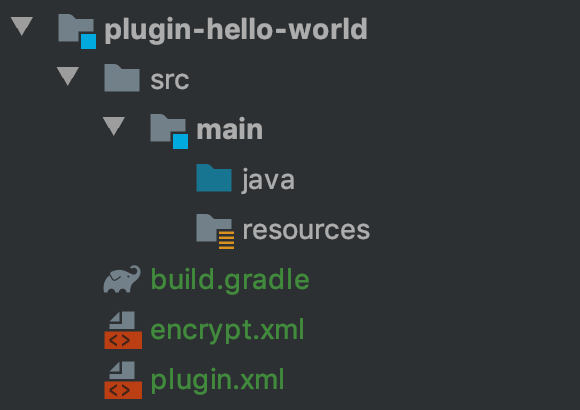
在report-starter-10/settings.gradle中增加一行

|  |
| --- |
| include(':plugin-hello-world') |

完成之后，在IntelliJ IDEA右侧边栏打开Gradle面板，点击刷新



等待IntelliJ IDEA解析完Gradle配置之后，就可以看到java目录和resources目录都变样式了



这里就可以直接在java目录下增加插件的实现类就可以。

第三方依赖

插件开发过程中，通常还会依赖一些非FineReport/FineBI内置的jar，这个时候，只需要把这些jar拷贝到插件工程的lib根目录下即可（没有就新建一个）。

#### **2）构建插件包**

构建加密插件包

进入插件源码目录，注意查看ext.guard的值，默认为false，需要构建加密插件包，要将这个值改为true；

执行命令

|  |
| --- |
| gradle zip |

执行完毕后，可以在插件源码目录/build/install目录下看到构建好的插件安装包(一个zip文件)，使用这种配置构建出来的插件代码，无法反编译。

关键构建配置

ext {

    /\*\*

     \* 项目中依赖的jar的路径

     \* 1.如果依赖的jar需要打包到zip中,放置在lib根目录下

     \* 2.如果依赖的jar仅仅是编译时需要，防止在lib下子目录下即可

     \*/

    libPath = "$projectDir/../webroot/WEB-INF/lib"

    /\*\*

     \* 是否对插件的class进行加密保护，防止反编译

     \*/

    guard = true

    def pluginInfo = getPluginInfo()

    pluginPre = "fine-plugin"

    pluginName = pluginInfo.id

    pluginVersion = pluginInfo.version

    outputPath = "$projectDir/../webroot/WEB-INF/plugins/plugin-" + pluginName + "-1.0/classes"

}

构建不加密插件包

进入插件源码目录，注意查看ext.guard的值，默认为false即可；

执行命令

|  |
| --- |
| gradle zip |

#### 3）插件接口开发

接口开发请参考官方文档：

https://wiki.fanruan.com/pages/viewpage.action?pageId=12746909

#### **4）发布插件包**

* 前言

在前面的章节中，我们已经学会了如何开发、调试和打包一下插件，但还无法把自己开发的插件展示和售卖给大量的FineReport/FineBI用户。

* 注册账号

访问https://market.fanruan.com（帆软市场），如果还没有账号，点击右上角的登录/注册，如下图所示



如果没有账号，就注册一个，如果已经有账号，就直接输入账号密码登录即可。

* 插件上传

成功进入帆软市场后，从右上角进入个人中心



切换到添加新商品页面，直接将自己构建好的插件包拖拽到上传区域上传即可。

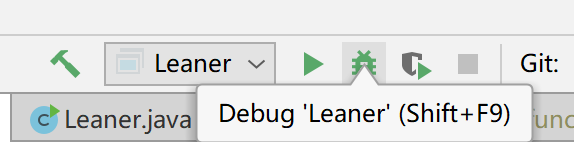
* 等待审核

成功上传后，等待帆软市场审核员审核通过，即可在帆软市场查看到自己的插件了。

### 在设计器中进行调试

#### 1）启动调试模式

因为在插件的Gradle配置文件中，我们已经配置好了插件的class编译目录，因此在启动设计器的时候，我们选择Debug模式即可进行插件调试



关键配置：在plugin-function的Gradle配置中，包含了class编译的目录描述

|  |
| --- |
| ext {    outputPath = "$projectDir/../webroot/WEB-INF/plugins/plugin-" + pluginName + "-1.0/classes"  } |

需要注意的是

1、plugin-com.fr.plugin.function.test-1.0中间的部分，必须保持和插件id相同，才能在开发时让插件正确的生效；

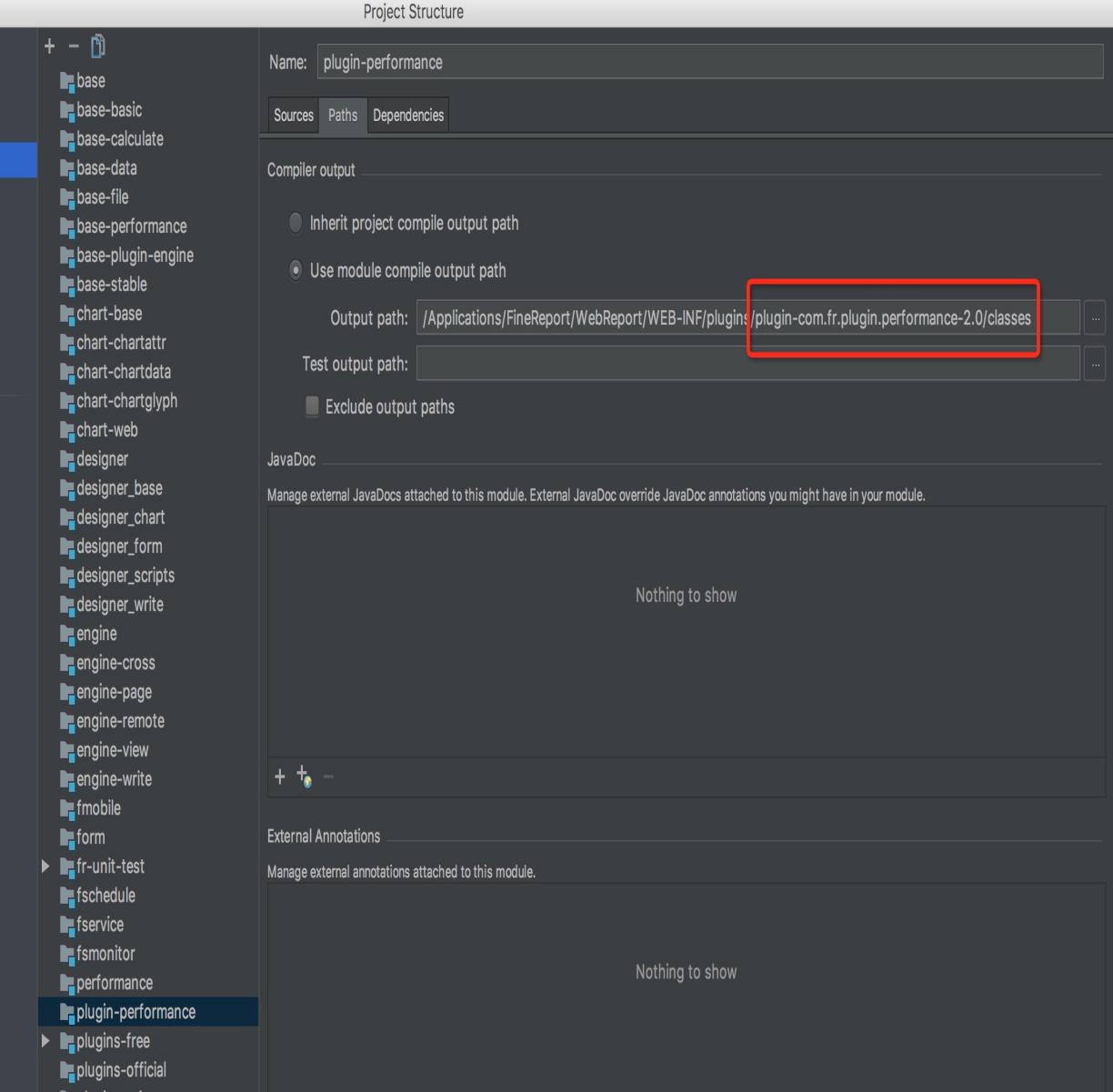
2、如果插件依赖了其他第三方jar包（包括finekit），需要将这些依赖的jar包也拷贝到plugin-com.fr.plugin.function.test-1.0目录下。

#### 2）调试环境搭建

**插件home**指环境下plugins文件夹中以【"plugin-" + 插件ID + "-" +插件版本号】命名的文件夹，如：plugin-com.fr.plugin.chart.vancharts-8.5.3

调试时，需要将插件的plugin.xml复制到插件home中，并将插件的编译output path设置为插件home/classes文件夹。如插件有自定义的lib，也要复制到插件home下。

如：

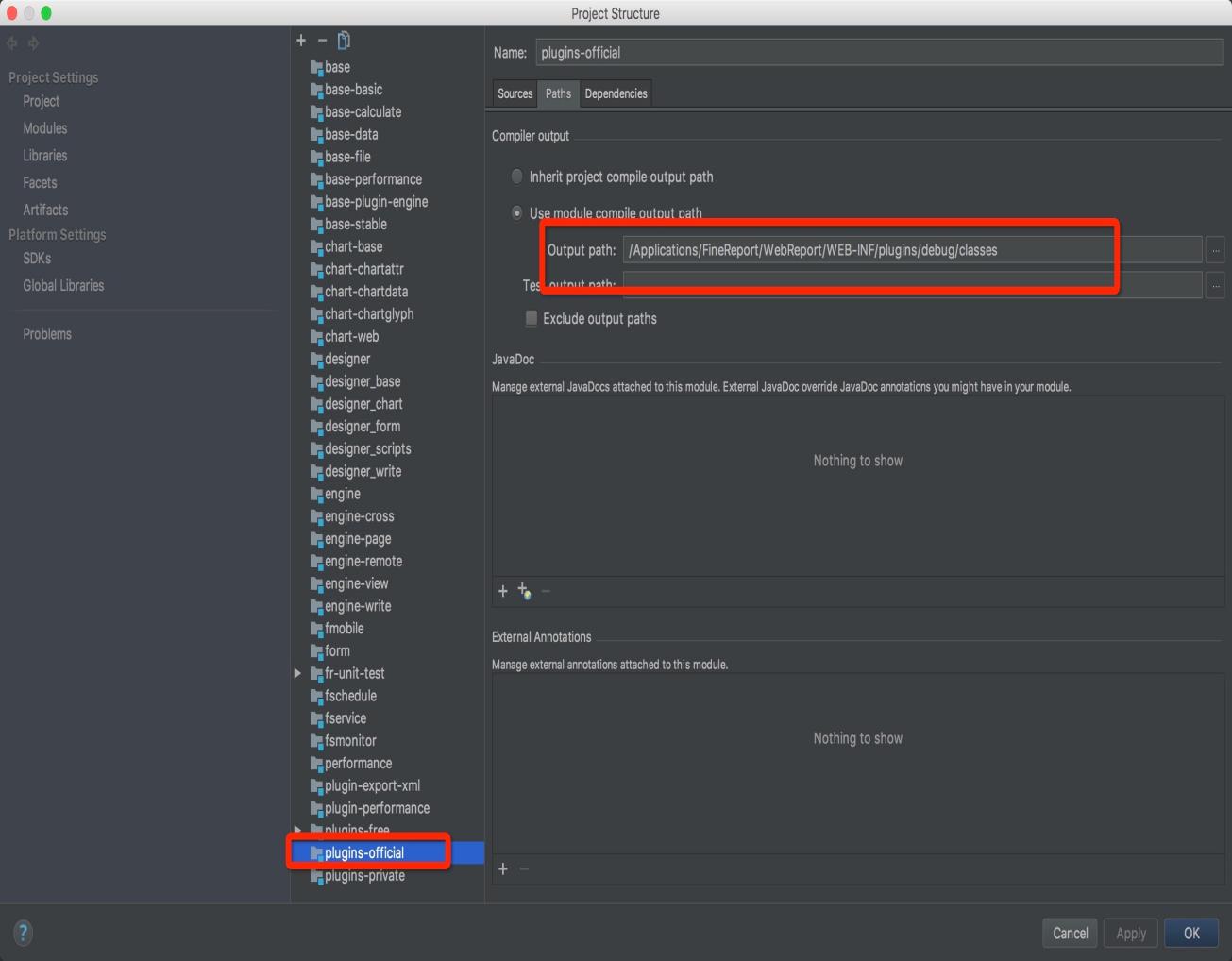


#### 3）同一个module下的多插件调试

有些时候，多个插件同属一个module，无法区分编译路径。这时可以将整个module都编译到环境plugins/debug/classes下，插件引擎会检测是否是debug模式以及当前插件的ClassPath，如果是debug，并且当前插件没有classPath，则将这个路径作为该插件的classPath.

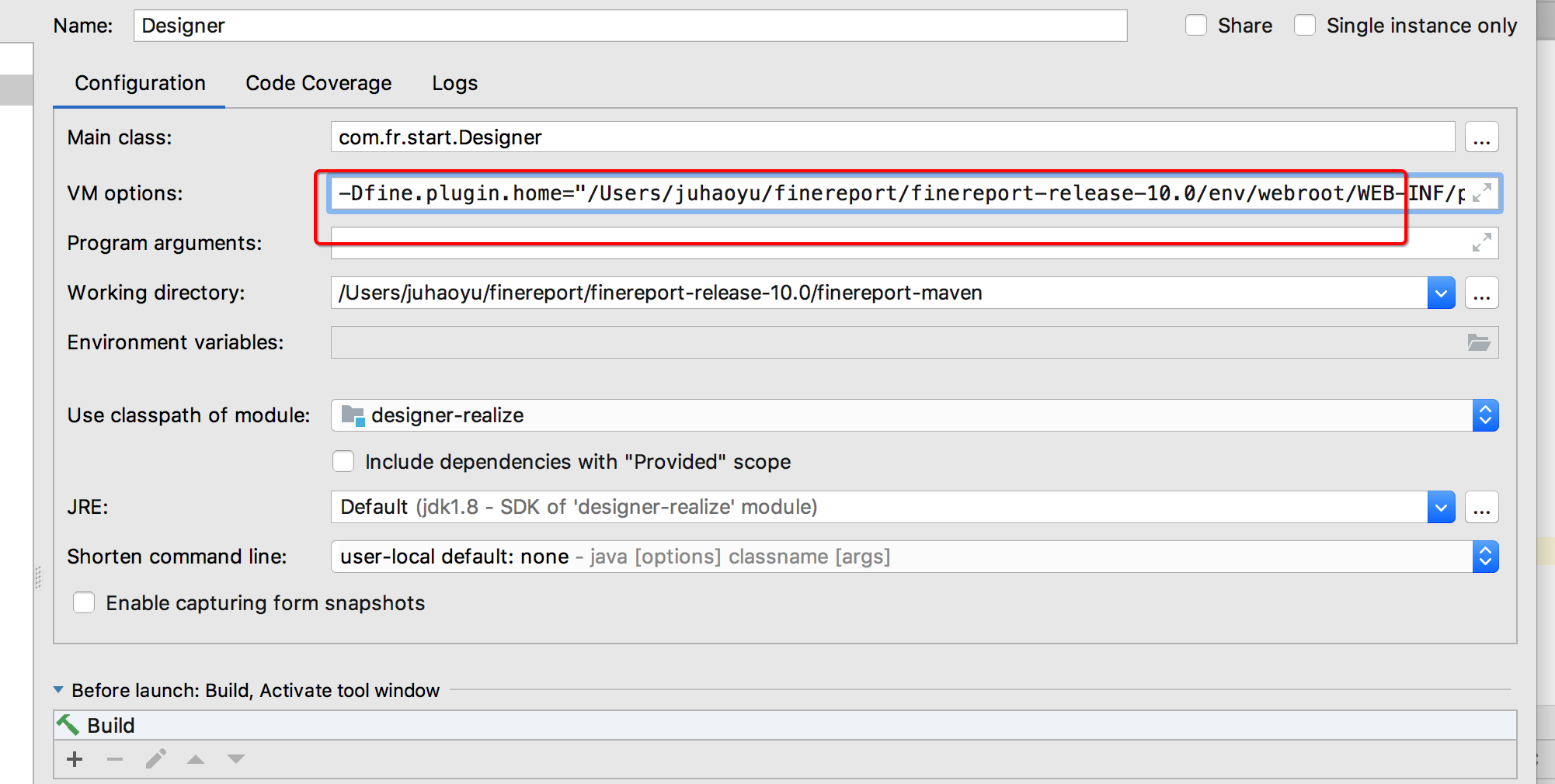
注意1：只有当插件home中只有plugin.xml时，才会将debug下的jar和classes文件夹添加为该插件的classPath.

注意2：不要将插件编译到报表的ClassPath下，否则会导致部分依赖于插件类加载器的特性出问题。



#### **4）远程环境调试**

方式一：添加启动参数 -Dfine.plugin.home="/Users/juhaoyu/finereport/finereport-release-10.0/env/webroot/WEB-INF/plugins" 指定插件根目录：



方式二：添加启动参数-Duser.dir="/Applications/FineReport10/bin" 指定一个安装目录（推荐，支持9.0）