

**Maven中java与Groovy的混合开发**

**Groovy 在后端也能尝到脚本语言的快感**

Groovy  
Apache的Groovy是Java平台上设计的面向对象编程语言。这门动态语言拥有类似Python、Ruby和Smalltalk中的一些特性，可以作为Java平台的脚本语言使用，Groovy代码动态地编译成运行于Java虚拟机（JVM）上的Java字节码，并与其他Java代码和库进行互操作。由于其运行在JVM上的特性，Groovy可以使用其他Java语言编写的库。Groovy的语法与Java非常相似，大多数Java代码也符合Groovy的语法规则，尽管可能语义不同。 --[Groovy wiki](https://link.zhihu.com/?target=https%3A//zh.wikipedia.org/wiki/Groovy)

关于maven添加Groovy开发的文档已经有一些了 我想做一些详细的介绍

**目录**

* Maven 配置groovy 插件
* 在 idea 中开发的配置&答疑

**Maven 配置groovy 插件**

**版本要求**

* JDK 1.6及以上
* Maven 3.0.1及以上

**配置pom.xml**

先贴Groovy官方插件使用介绍  
[Groovy- GMavenPlus/groovy-maven-plugins](https://link.zhihu.com/?target=https%3A//github.com/groovy/GMavenPlus/wiki/Examples%23groovy-maven-plugins)

要编译同时包含Java和Groovy的项目（并且两者之间具有依赖关系），除了正常编译之外，还需要生成存根(stubs)。

在pom.xml中需要添加如下配置 PS:以下配置只包含需要添加进pom.xml的内容

pom.xml

<project>

<build>

<plugins>

*<!-- 增加 gmavenplus 插件 允许集成Groovy到Maven-->*

<plugin>

<groupId>org.codehaus.gmavenplus</groupId>

<artifactId>gmavenplus-plugin</artifactId>

<version>1.7.1</version>

<executions>

<execution>

<goals>

<goal>addSources</goal>

<goal>addTestSources</goal>

<goal>generateStubs</goal>

<goal>compile</goal>

<goal>generateTestStubs</goal>

<goal>compileTests</goal>

<goal>removeStubs</goal>

<goal>removeTestStubs</goal>

</goals>

</execution>

</executions>

</plugin>

<plugin>

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

<artifactId>maven-compiler-plugin</artifactId>

</plugin>

</plugins>

</build>

<dependencies>

<dependency>

<groupId>org.codehaus.groovy</groupId>

*<!-- groovy-all 包含所有groovy GDK 中的包,*

*groovy 只包含基础 Groovy 编译需要的包-->*

<artifactId>groovy-all</artifactId>

*<!-- any version of Groovy \>= 1.8.2 should work here -->*

<version>2.5.7</version>

*<!--指定类型为 pom -->*

<type>pom</type>

</dependency>

</dependencies>

</project>

这些配置将会把src/main/java和src/main/groovy作为源码目录.

如果没有src/main/groovy请创建,并在此处编辑.groovy文件

**增加其他目录作为源码目录**

如果要使用其他目录作为Groovy源码目录,  
比如src/additionalGroovy,则需要做以下配置

pom.xml

<plugin>

<groupId>org.codehaus.gmavenplus</groupId>

<artifactId>gmavenplus-plugin</artifactId>

<version>1.7.1</version>

<executions>

<execution>

<goals>

<goal>addSources</goal>

<goal>addTestSources</goal>

<goal>generateStubs</goal>

<goal>compile</goal>

<goal>generateTestStubs</goal>

<goal>compileTests</goal>

<goal>removeStubs</goal>

<goal>removeTestStubs</goal>

</goals>

</execution>

</executions>

<configuration>

<sources>

*<!-- 在此节点下配置源码目录,可配置多个 -->*

<source>

<directory>${project.basedir}/src/main/groovy</directory>

<includes>

<include>\*\*/\*.groovy</include>

</includes>

</source>

<source>

<directory>${project.basedir}/src/additionalGroovy</directory>

<includes>

<include>\*\*/\*.groovy</include>

</includes>

</source>

</sources>

</configuration>

</plugin>

</plugins>

</build>

以上配置会把src/main/groovy和 src/additionalGroovy目录作为Groovy源码目录 **maven 的生命周期中就能正确的编译了,然后,愉快的使用Groovy吧**

**Groovy 为什么能和java混合开发**

1.所有.groovy文件会经过groovyc编译成class文件,对于JVM来说与Java编译后没有区别.

2.在Maven中,引入gmavenplus-plugin插件后,插件会将.groovy,.java分别编译并添加到Maven源目录,然后通过Maven打包

**gmavenplus-plugin插件参数简介**

完整介绍参阅  
[gmavenplus-plugin config](https://link.zhihu.com/?target=http%3A//groovy.github.io/GMavenPlus/plugin-info.html)

**<goals>介绍**

**为什么要加这么多<goal>标签?(来自官网)**  
Because we need to mix Groovy and Java lifecycles together. The only other way to do this is a with a custom lifecycle. This seemed inappropriate since you have to introduce a new type (which both GMaven and GMavenPlus did at one time as groovy-jar). But what if you are mixing Groovy, Java, and another language (maybe Scala)? We'd have to define types for all the combinations. It also seemed inappropriate that consumers of your artifact should be aware of the fact it was written in another language.  
个人翻译:  
因为我们需要将Groovy和Java生命周期混合在一起。另外还有一种方法是使用自定义生命周期,但是不太合适，因为你必须引入一种新类型（GMaven和GMavenPlus同时作为groovy-jar）才能使用。但是如果你混合使用Groovy，Java和另一种语言（也许是Scala）,我们必须为所有组合定义类型。不能让引用你代码的地方自己判断源码的语言.所以需要增加这些标签处理为JVM可以理解的class,统一处理

1. <goal> 列表

知乎文章好像没法识别Markdown的表格格式,只能先传图了,如果是我不会用,请告诉我怎么做表格

goal 各项解释

**idea 配置&答疑**

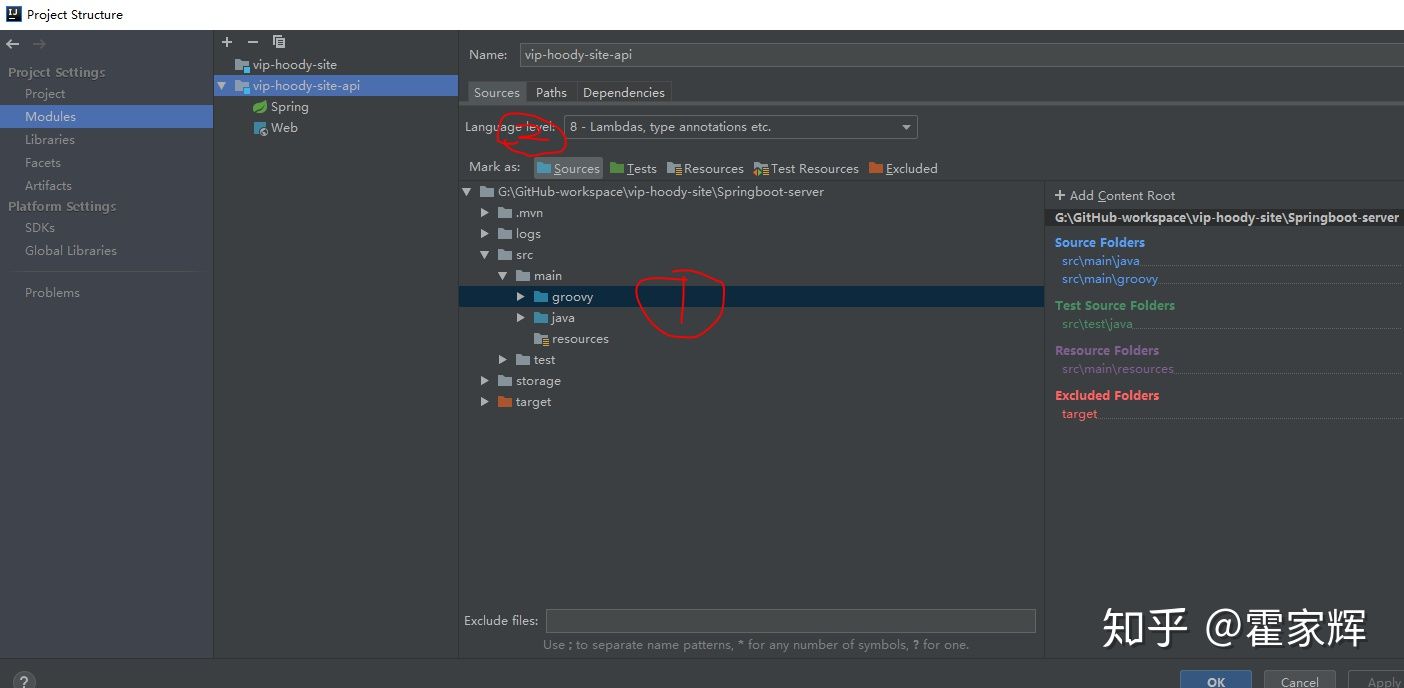
**在使用Idea集成开发环境进行代码开发的时候,会出现找不到Groovy类的情况**

是因为Idea会按照Mavenpom.xml配置的源文件目录进行识别(默认目录${project.basedir}/src/main/java)  
就算在GMaven-plugin配置了指定源目录也不行,因为Idea不懂这个(可能后续会添加支持)

有2个方法可以解决这个问题, **主要目的是让Idea知道我们Groovy的目录是源码目录**

**方法1.配置Idea Project Structure**

1. 打开 Idea 的菜单 并进入 **File-> Project Structure -> module** 2.如下图①选择需要增加为源码的目录 3.如图②,标记为源码目录



Idea 配置 源码目录

**方法2.配置Maven pom.xml 修改源码目录<sourceDirectory>**

更多Maven 配置参见 [Apache Maven教程 -管理目录](https://link.zhihu.com/?target=https%3A//www.codetab.org/tutorial/apache-maven/plugins/manage-maven-directories/)

在以下节点添加配置 pom.xml

<project>

<build>

*<!-- 将${project.basedir}/src/main整个作为源码目录,你再添加什么源码都在里面了 -->*

<sourceDirectory>${project.basedir}/src/main</sourceDirectory>

</build>

</project>

结束,

原文发布于我的个人博客

[Hoody's Blog - 个人博客 - Maven中java与Groovy的混合开发​hoody.tech](https://link.zhihu.com/?target=https%3A//hoody.tech/blog/detail/29)