教程（三）Shared library

前言

随着pipeline流水线技术的成熟，使用pipeline脚本的job也迅速增加。虽然我们可以做一个尽可能通用的pipeline脚本样例，让搭建者只需要修改几个赋值参数就可以在自己的项目中应用，初衷是希望所有人能理解pipeline中的过程，但也发现一些比较麻烦的问题，比如有些人不熟悉具体的脚本拿来随意删改导致各种错误，还有就是我们在pipeline脚本中增加一些新功能时又需要通知所有的pipeline维护人员去修改，过程非常纠结。
这时候就意味着我们需要用到pipline的共享库功能（Shared Libraries）了，在各种项目之间共享pipeline核心实现，以减少冗余并保证所有job在构建的时候会调用最新的共享库代码 。
这篇我们就介绍下pipeline的这个黑科技：Shared Libraries

目录结构

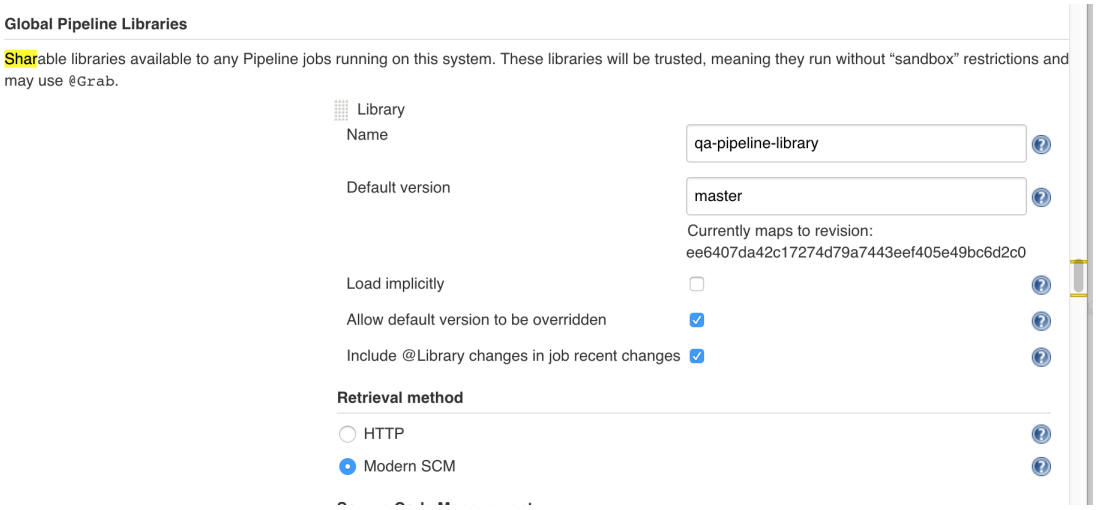
Shared Library通过库名称、代码检索方法（如SCM）、代码版本三个要素进行定义，库名称尽量简洁，因为它会在脚本中被调用，在编写Shared Library的时候，我们需要遵循固定的代码目录结构。
Shared Library代码目录结构如下：

```
(root)
+- src                     # Groovy source files
|   +- org
|       +- foo
|           +- Bar.groovy  # for org.foo.Bar class
+- vars
|   +- foo.groovy          # for global 'foo' variable
|   +- foo.txt             # help for 'foo' variable
+- resources               # resource files (external libraries only)
|   +- org
|       +- foo
|           +- bar.json    # static helper data for org.foo.Bar
```

src目录就是标准的Java源目录结构。执行Pipeline时，该目录将添加到classpath中。
vars目录托管定义可从Pipeline访问的全局脚本(一般我们可以在这里编写标准化脚本)。 我们在pipeline中调用的指令就是在这里定义的，这是我们最重要的目录。
resources目录允许libraryResource从外部库中使用步骤来加载相关联的非Groovy文件。也就是我们的pipeline脚本是可以通过一个代码来加载resource目录下的文件

定义全局库

这里只介绍全局 Shared Library的方式,通过Manage Jenkins » Configure System » Global Pipeline Libraries 的方式可以添加一个或多个共享库。
这些库将全局可用，系统中的任何Pipeline都可以利用这些库中实现的功能。并且通过配置SCM的方式，可以保证在每次构建时获取到指定Shared Library的最新代码。



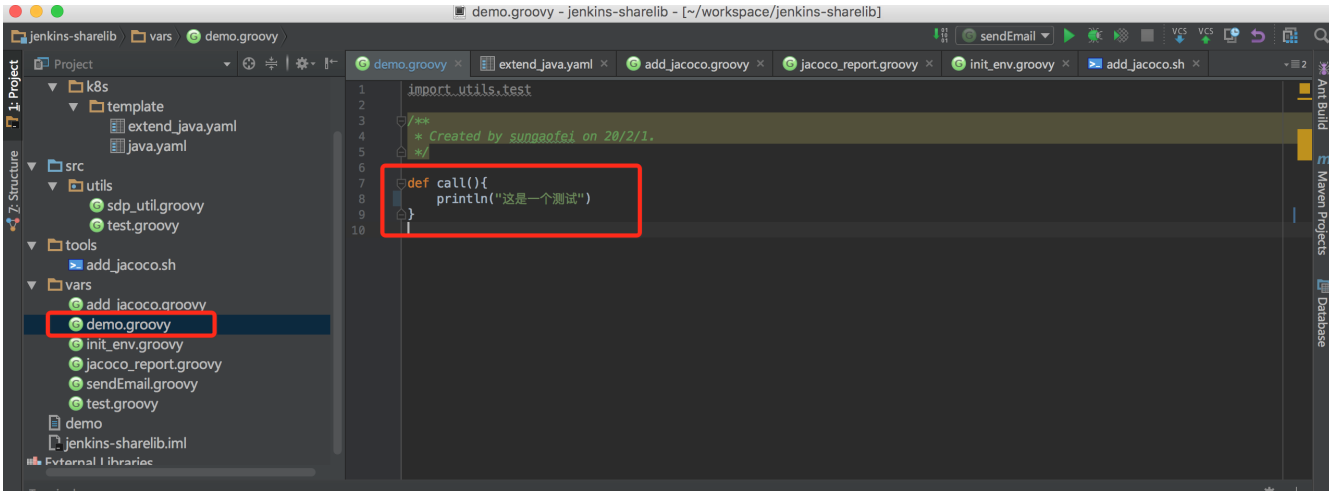
动态加载库

从2.7版本起，Pipeline: Shared Groovy Libraries plugin插件提供了一个新的参数“library”，用于在脚本中加载（non-implicit）库
如果只需要加载全局变量/函数（从vars/目录中），语法非常简单：
此后脚本中可以访问该库中的任何全局变量。

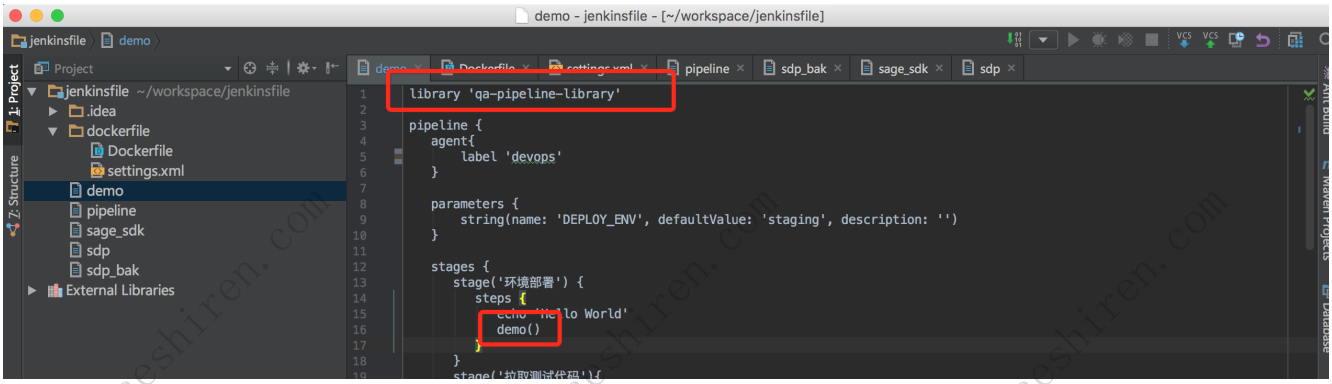
```
library 'qa-pipeline-library'
```

Shared Libraries Demo

我们还是通过一个demo开始吧。 在我们的gitlab上创建了一个repo：https://gitlab.4pd.io/qa/jenkins-sharelib
这个repo下创建了一个demo。 注意这里方法名字必须是call。 这里涉及到了groovy语言的委托机制， 所以名字必须是call。



然后在我们的pipeline中我们可以向下面这样调用。



我们只要在pipeline上面使用 library 'qa-pipeline-library' 就可以在下面的steps里直接调用demo方法了。

如果想要load一个文件， 则可以使用

```
libraryResource 'k8s/template/java.yaml'
```

如下：

上面的例子是从共享库中的resource目录下加载一个k8s的yaml文件然后给下面的agent使用，用来动态创建slave pod 来执行pipeline任务。
具体jenkins与k8s集成的内容我会放到下一篇教程讲。这里只是演示一下怎么去加载共享库中的文件.

Shared Libraries 实战

https://gitlab.4pd.io/qa/jenkins-sharelib/blob/master/vars/sendEmail.groovy

上面那个链接是我根据我们的需要开发的发送邮件用的共享库。 它会自己判断job的执行状态来发送不同的邮件内容。
并且会自动的获取allure report中的测试结果信息。 如下：

后记

Shared Libraries的方式抽象了各种项目之间共享的代码（甚至整条完整的pipeline），有效降低了业务工程师使用pipeline脚本的复杂度。同时通过外部源代码控制（SCM）的方式，可以保证最新提供的库代码功能可被所有pipeline项目即时使用，在大团队推广和协作过程中可起到非常重要的作用。