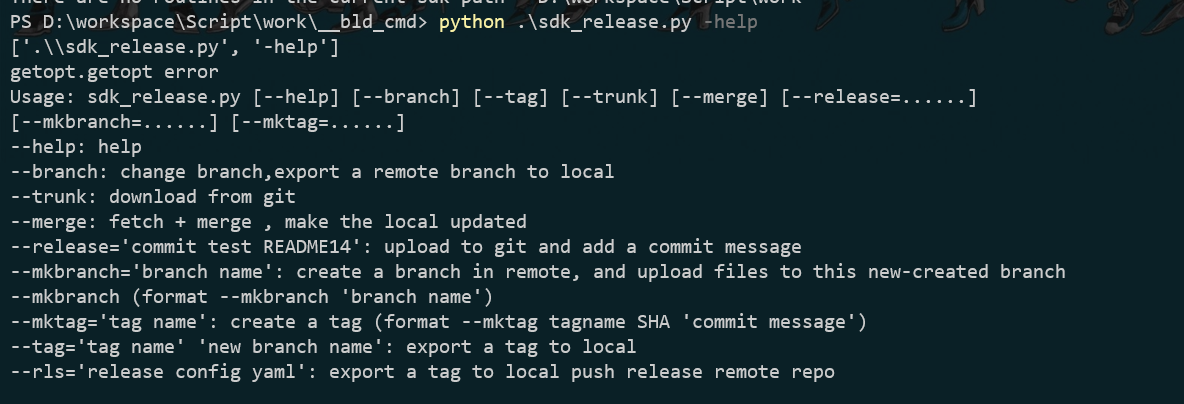
# Release Comd Notes

## sdk release



### 参数解析：

| 参数名称 | 传入的值 | 描述 |
| --- | --- | --- |
| --help | None | 获取脚本的使用方法。 |
| --branch | None | 返回所有branch，如要切换对应分支，可再输入对应的分支序号 |
| --rls | rls-file | 指定的file文件名称，需要确保该文件在\_bld\_script 文件下，传入的时候去掉后缀。 |
| --trunk | ssh\_clone\_addr | clone指定的仓库，基本上不用 |
| --merge | None | 相当于pull， |
| --release | rls\_yml\_file | 指定的rls\_yml配置文件，根据配置文件，将代码推到目标仓库中 |
| --mkbranch | branch\_name | 创建branch |
| --tag | 'tag name' 'branch name' | 对指定的branch 创建tag，branch 不能是已存在的，之前创建后删除了的也是不行的。tag 不能是未存在的 |
| --mktag | tag\_name | 创建tag |

### 使用流程（使用流程跟标号无关，以下流程互不影响）：

#### 1.切换分支：

该命令会列出当前所有分支，再选择某个分支切换

python .\sdk\_release.py --branch

此时会打印所有分支，并返回分支对应的index，再输入需要分支的index即可切换成功



#### 2.clone 仓库：

python .\sdk\_release.py --trunk git@192.168.1.11:bin.yu/release.git

该命令是一条保护命令，当 当前文件夹存在已拉取的代码仓库（文件夹）时，会提示git file exist，no need to clone。

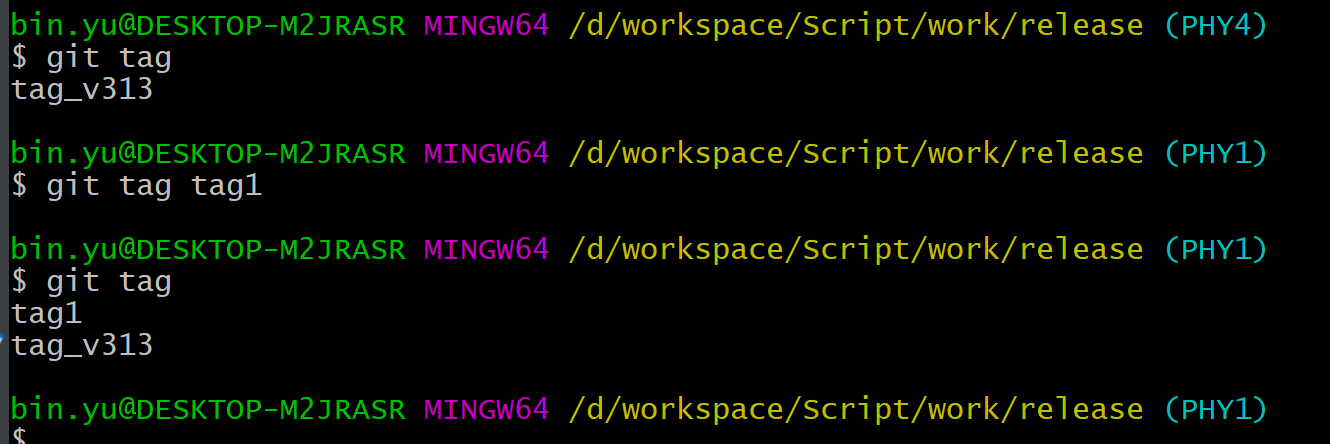
若需要用该命令拉取代码时，需要将当前sdk文件夹删除，但是基于改脚本是存放在sdk路径下的，所以该命令基本上不用。

#### 3.创建标签：

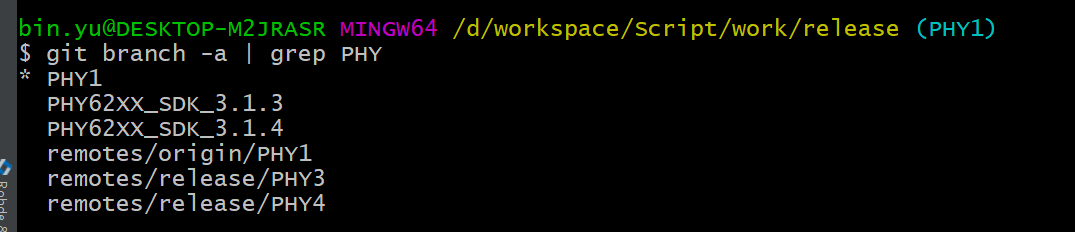
python .\sdk\_release.py --tag 'tag\_name' branch\_name

创建标签之前需要检查以下两项：

1. 需要创建的标签是否存在，如果不存在需要手动创建

* 
* 创建标签的命令为：git tag tag\_name

1. 需要贴标签的分支是否已存在，或者存在过，

* 
* 检查分支存在或者存在过的命令为： git branch -a | grep branch\_name
* 当以上两步检查通过时，即可为指定的branch 创建标签了：

python .\sdk\_release.py --tag 'tag1' PHY2

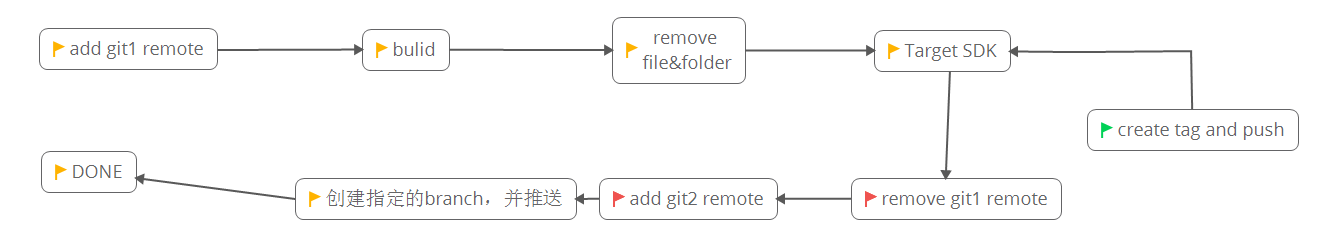
 当返回 tag checkout successfully 即成功。

#### 4.推送release 版本

python .\sdk\_release.py --rls rls\_yml\_file

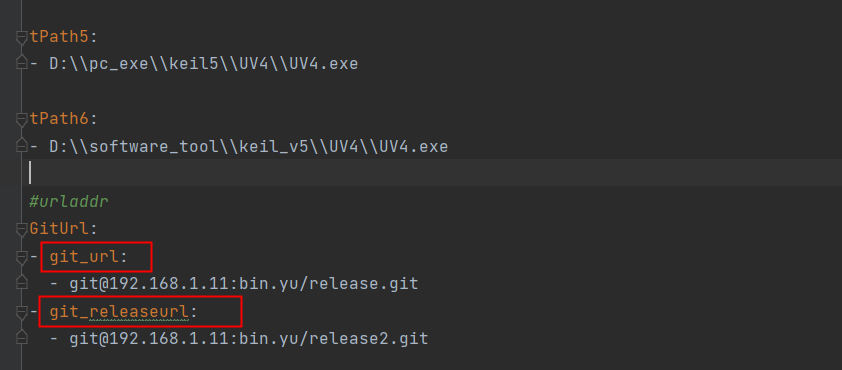
该命令是将仓库1的代码拉到本地，然后通过指定的bld版本文件指定需要的工程进行编译，删除整理。再将整理好的SDK文件打好标签推送到仓库1。

这时仓库1的工作是完成了，再将仓库1的remote 移除掉，同时添加仓库2的，在本地代码不动的基础上，创建指定的branch，在将代码push上去。此时所有的工作已完成。以下是简单的流程图：



rls文件参数详解：参考 rls\_example.yml

两个仓库的地址需要提前配置好，配置地址在\_bld\_script/sdk\_bld\_tool.yml文件中

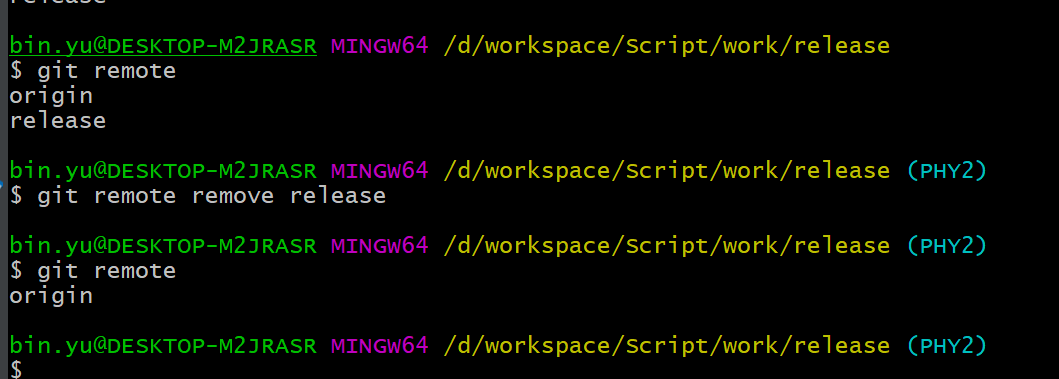


git\_url:为数据源仓库

git\_releaseurl：目标仓库

在执行改命令之前，需要做以下检测：

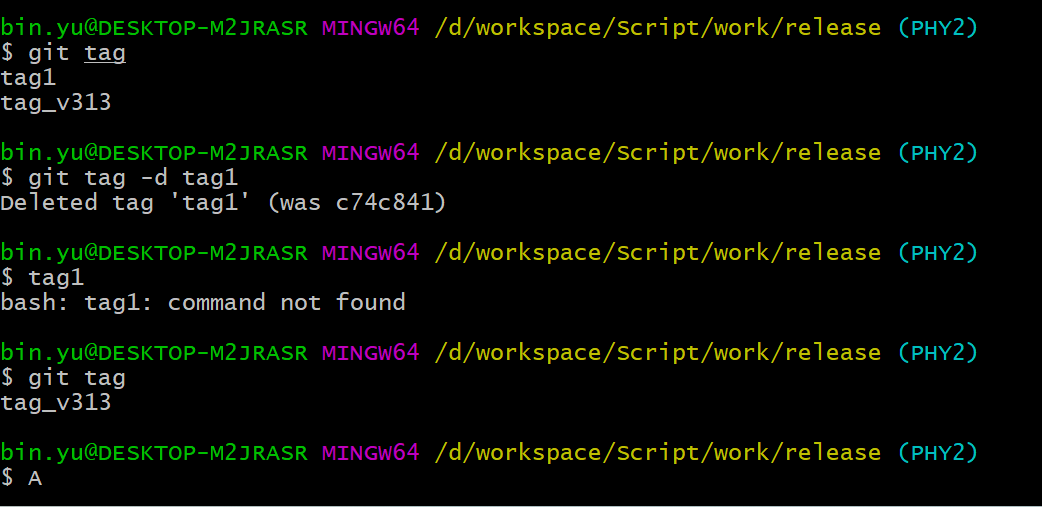
1.工作路径下是否已经加入release 仓库，如已添加需删除。



检查命令为:git remote

删除命令为：git remote remove release

2.确定需要创建的tag是否已存在，如已存在需删除。



本地检查命令为：git tag

本地删除命令为：git tag -d tag\_name

**远程tag请在页面上面删除。一定要删除，不然会失败！！！**

3.指定的branch是否已存在，如已存在需要手动删除

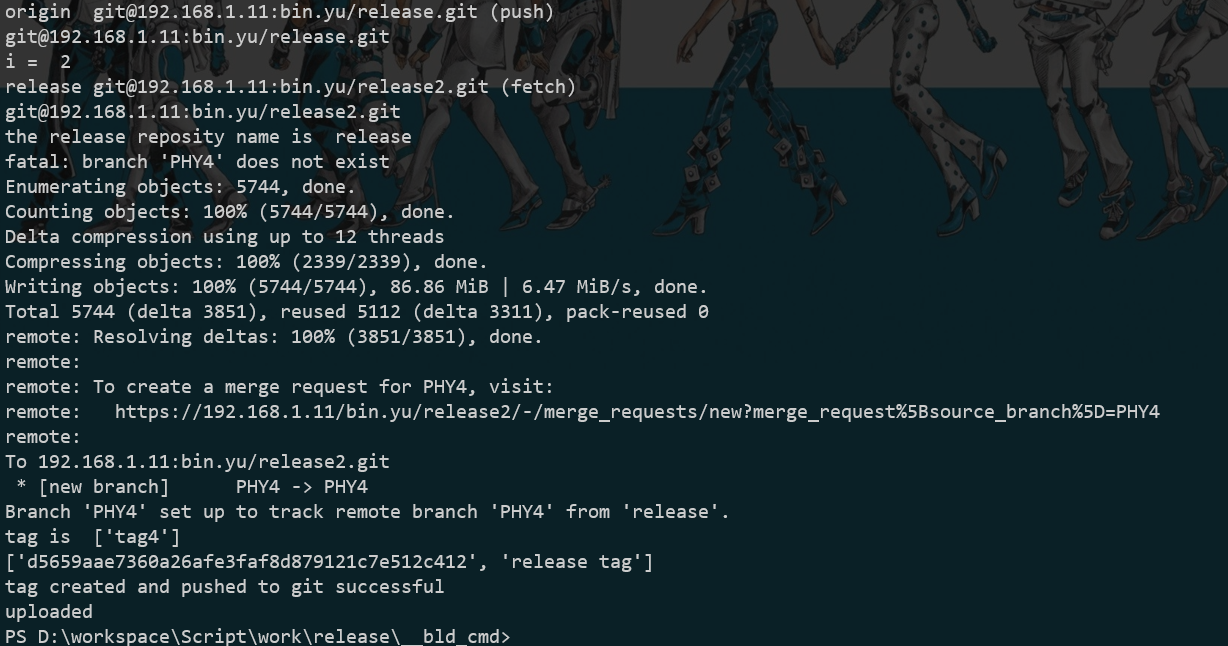
检查命令为：git branch |grep branch\_name

删除命令为：git branch -D branch\_name

远程的需要在页面上删掉，切记，如若没删除，两边代码冲突，会推送失败。

当以上检查通过时即可执行该命令，例如：

python .\sdk\_release.py --rls rls\_v313



当返回tag created and pushed to git successful 时，即release成功。

在此过程中，会有build SDK 工程的步骤，该步骤中如果有报错会停止release，所以在执行之前请多多检查工程代码是否正常。

### debug info:

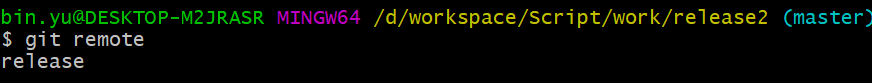
#### 1.debug: git remote failed



当在执行脚本过程中提示该debug消息时，表示当前执行的地址中，之前已经添加过一遍release remote了，需要手动删除，以下提供删除命令，按照步骤来删除

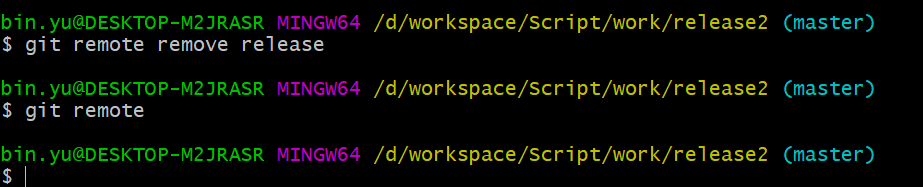
1.查看是否存在： git remote

存在的话，返回列表中会存在 release名称：

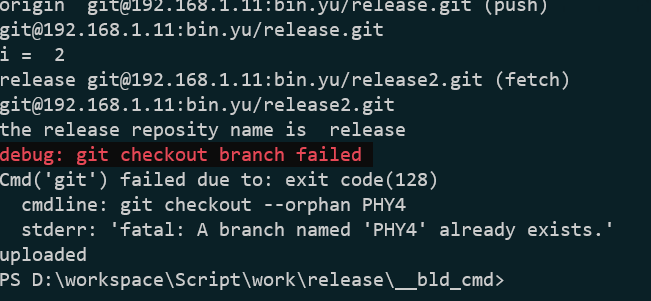


2.删除：git remote remove release

删除成功后，再输入git remote验证是否存在即可

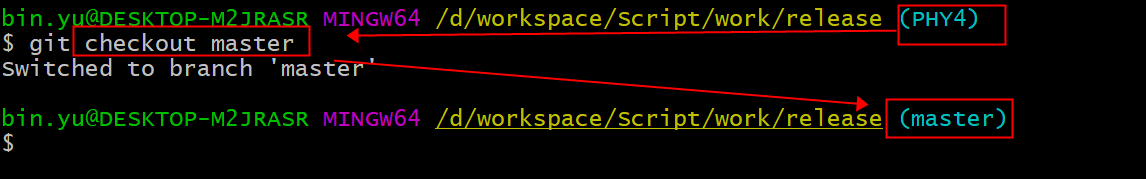


#### 2.debug: git checkout branch failed



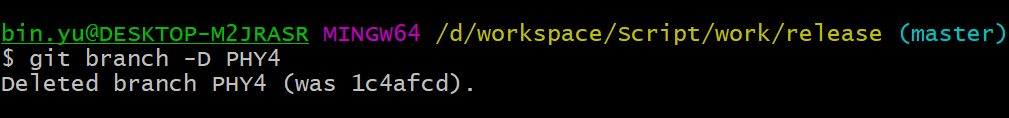
该提示是本地已存在PHY4的分支，需要手动删除。

1.注意删除的时候需要确定当前所在分支，如刚好处在需要删除的分支时，需要切换出去才能删除，以下图为例：



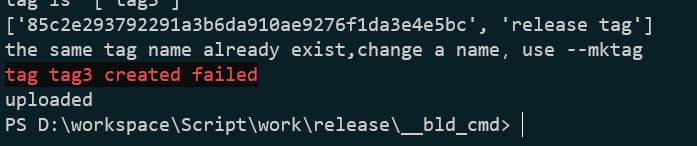
我需要删除PHY4的分支，刚好我正处于该分支上，这时需要切换到别的分支上，切换命令为：git checkout branchname

2.删除分支，git branch -D branchname

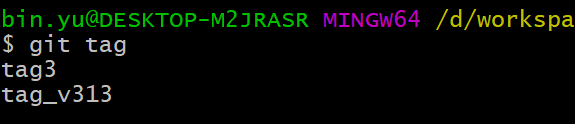


此时，本地分支即已删除成功

#### 3.debug:tag created failed

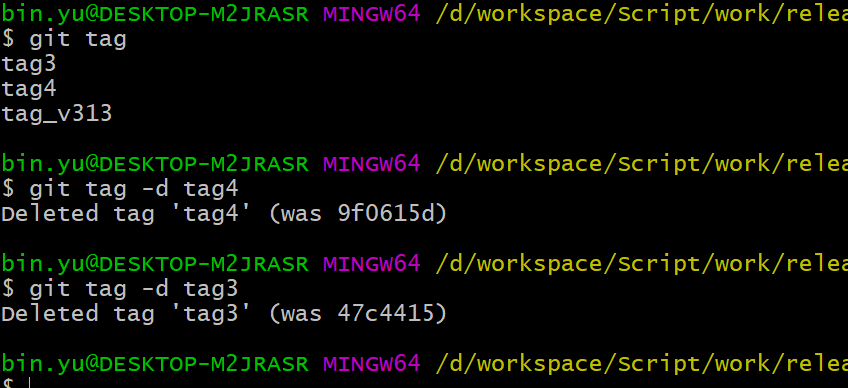


该报错是因为已经存在tag，需要手动删除掉，删除步骤如下：



1.在工作路径下查看tag list：git tag

2，删除tag：git tag -d tagname



3.本地删除后，再删除远程tag，删除远程tag最简单的方法。打开git仓库页面，找到tag 页面，选择需要删除的tag，点击删除即可

