# 工具使用环境

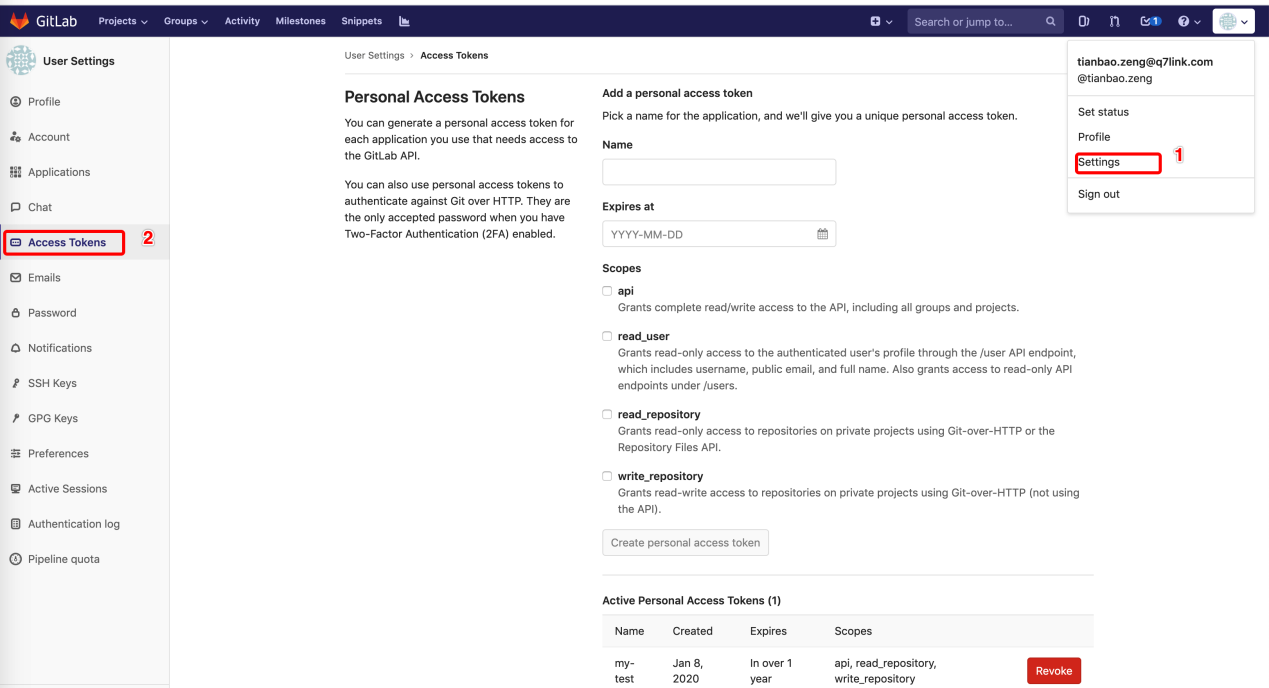
1.需要安装python3及一些python模块（python-gitlab(3.0.0以下版本)、pyyaml）

命令：pip3 install pyyaml(模块名称)

pip3 install python-gitlab==2.10.0

# 工具初始化

1. 在本地新建目录，然后按照git中的目录结构将各工程检出
2. 将工具包解压到检出工程的根目录下
3. 在git上创建token，然后放入branch/utils.py中的TOKEN变量





1. 检查branch/path.yaml中各工程路径是否正确（非常重要，路径不对是找不到本地工程进行相关处理的）

# 工具说明

**所有工具操作的工程均是branch/path.yaml中配置的工程，若有工程未配置，则不会操作未配置的工程**

## 1.changeVersion.py

### **功能**：

根据build工程的config.yaml文件将各工程版本号进行修改及开发脚本/接口数据修复文件清空

注：为保证有同学新增了依赖关系而工具未修改到的情况，最好是全局检查一下版本总数。

### 命令

python3 changeVersion.py 分支名称 [true]

true 代表是否清空开发脚本及接口数据修复文件

### 例：

python3 changeVersion.py hotfix

根据hotfix分支的build工程config.yaml文件修改hotfix分支各工程依赖的其他工程版本以及自身版本

python3 changeVersion.py hotfix true

根据hotfix分支的build工程config.yaml文件修改hotfix分支各工程依赖的其他工程版本以及自身版本，最后清空开发脚本及接口数据修复文件

## checkanddeleted.py

### 功能

获取拥有指定分支的工程，检查要删除的分支是否已合并至指定分支，若已合并则删除，如有工程未合并则所有工程均不进行删除。

### 命令

python3 checkanddeleted.py 要删除的分支名称 合并的目标分支名称 [工程名称..,不传则删除所有工程]

### 例：

python3 checkanddeleted.py hotfix master  
 检查所有工程hotfix分支是否已合并至master分支，若已合并则删除hotfix分支  
  
 python3 checkanddeleted.py hotfix none  
 不检查分支是否合并，直接删除所有工程的hotfix分支  
  
 python3 checkanddeleted.py hotfix master finance basebi  
 检查finance、basebi工程hotfix分支是否已合并至master分支，若已合并则删除finance、basebi的hotfix分支

## 3.createBranch.py

### 功能

根据来源分支创建目标分支，创建出来的分支，如果分支是hotfix/release/emergency/stage-emergency/hotfix-inte/dev，则会自动创建分支保护，所有工程只有管理员有全权限（mr、push）

### 命令

python3 createBranch.py 来源分支 目标分支 [工程名称...]  
以来源分支为模板创建目标分支，只创建来源分支存在的工程，如果工程不存在来源分支，则不创建目标分支

### 例：

python3 createBranch.py master hotfix  
 由master分支创建hotfix分支，并将hotfix进行分支保护，如果有工程存在hotfix分支则报错，并且所有工程均不创建目标分支  
  
 python3 createBranch.py master feature-xxx build finance  
 将build和finance工程，由master分支创建feature-xxx分支

## 4.protectBranch.py

### 功能

对指定分支进行分支保护，目前仅预制了三套权限

**hotfix权限**：build、init-data为管理员全权限（mr、push）；其余工程为管理员mr权限，所有人禁止push

**release权限**：所有工程管理员全权限（mr、push）

**dev权限**：所有工程所有人全权限（mr、push）

**none权限**：所有工程所有人禁止mr和push

**d**：删除分支保护

### 命令

python3 protectBranch.py 分支 权限 [工程名称...]

### 例：

python3 protectBranch.py hotfix release

将hotfix分支设置为release权限，一般在改版本号时修改为此权限

python3 protectBranch.py hotfix hotfix project budget  
 将project和budget工程的hotfix分支设置为hotfix权限，

python3 protectBranch.py hotfix d

将hotfix分支的分支保护删除

## checkcommit.py

### 功能

检查指定分支的提交记录是否合并(cherry pick 或merge)到目标分支，并输出检查报告（输出地址：log/noMerge/）

### 命令

python3 checkcommit.py 检查分支 目标分支 [偏移量，单位：天，默认10]

### 例：

python3 checkcommit.py hotfix dev

检查10天内hotfix的提交记录是否均合并到dev分支

python3 checkcommit.py hotfix dev 3

检查3天内hotfix的提交记录是否均合并到dev分支

## checkout.py

### 功能

检出指定分支代码

### 命令

python3 checkout.py 分支 [是否关闭IDEA的git管理]

是否关闭IDEA的git管理：默认false。false：不处理IDEA的git管理；true：将没有该分支的工程关闭IDEA git管理

注：由于idea的特性，在关闭前请将输入符焦点聚焦到有此分支的工程文件上。否则关闭git管理之后会有问题

### 例：

python3 checkout.py hotfix

检出hotfix分支到本地

python3 checkout.py hotfix true

检出hotfix分支到本地，并关闭没有hotfix分支工程的idea git管理

## closeGit.py

### 功能

关闭没有指定分支的工程的IDEA git管理

### 命令

python3 closeGit.py 分支

注：由于idea的特性，在关闭前请将输入符焦点聚焦到有此分支的工程文件上。否则关闭git管理之后会有问题

### 例：

python3 closeGit.py hotfix

关闭没有hotfix分支工程的IDEA git管理

## tag.py

### 功能

为指定分支最新提交打tag

### 命令

python3 tag.py 分支 上线日期

### 规则

1.build工程tag: 上线日期-分支

2.platform下的工程: framework的版本号-分支

3.其他工程: 工程自身版本号-分支

4.工程版本取自此分支的build工程中的config.yaml文件。如果build工程没有此分支则报错

5.如果tag存在则不打，platform下的工程如果最新提交上面有tag并且tag上记录的分支有此分支，则不打

### 例：

python3 tag.py master 20210402

为master分支工程打tag