Git介绍

Git介绍

Git是一种**代码管理工具**，支持**代码存放**、**版本控制**和**团队协作**的功能。

举个比较简单的例子：同学A和同学B合作完成一个项目，同学A先写好了一个`demo.py`文件:

```
print("Hello World!")
```

而同学B在这个基础上又进行更改：

```
print("Hello C!")   # 替换了原来的语句
```

为了不产生文件替换的冲突，同学B不得不将文件命名为`demo_1.py`

如果两位同学不断交替更改，可能会产生以下多个冗余的文件：

```
|-demo.py
|-demo_1.py
|-demo_2.py
|...
```

这时如果有了**Git**，就可以只用一个文件`demo_git.py`对这种更改代码的过程进行管理。

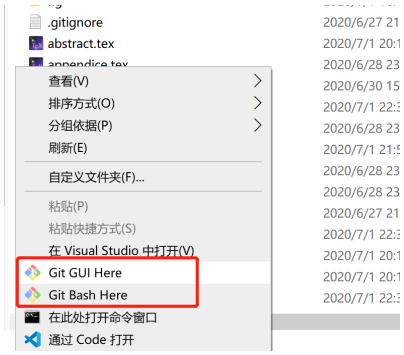
1 使用Git命令管理（命令行）

1.1 Git的下载与安装

在Git官网选择与操作系统适配的版本下载并安装：https://git-scm.com/download

以Win10 64位为例（**红框2选1**），安装过程略（一路下一步即可）

安装完毕后，在每个文件夹右键应该有**git bash**出现（**git bash是一个Unix环境**，可以在里面使用例如`cd`、`ls`、`mkdir`等Unix指令）：

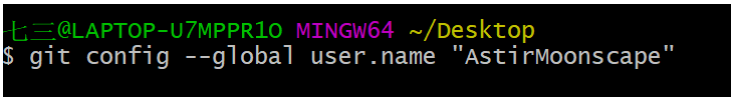


接下来进行**初步的配置**，因为Git是分布式版本控制系统，所以需要填写用户名和邮箱作为一个标识。

- 配置用户名（GitHub上注册的用户名）

在git bash输入：

```
git config --global user.name "你的用户名"
```

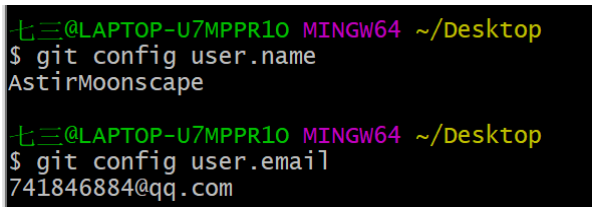


- 配置用户邮箱（GitHub上注册的邮箱）

```
git config --global user.email "你的邮箱"
```

- 查看是否配置成功

```
git config user.name
git config user.email
```



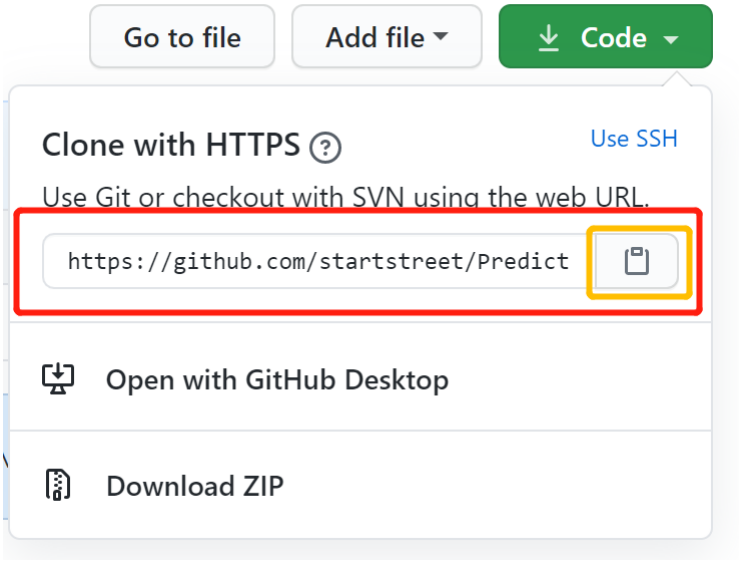
1.2 Git命令简介及使用

这里只介绍了最常用的五种指令。

- **初始化**

```
git clone "仓库地址"
```

首先在GitHub仓库页面找到仓库地址，点击复制：



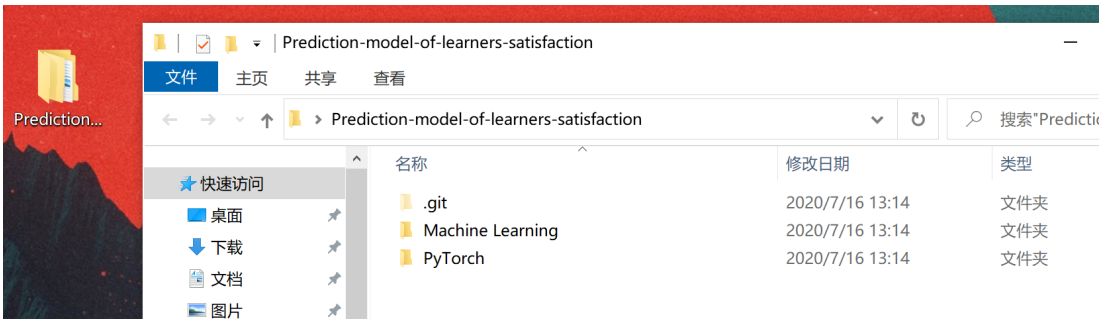
然后在git bash输入：

```
git clone "https://github.com/startstreet/Prediction-model-of-learners-satisfaction.git"
```

效果如下：

```
七三@LAPTOP-U7MPPR1O MINGW64 ~/Desktop
$ git clone "https://github.com/startstreet/Prediction-model-of-learners-satisfa
ction.git"
Cloning into 'Prediction-model-of-learners-satisfaction'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 10 (delta 2), reused 9 (delta 1), pack-reused 0
Unpacking objects: 100% (10/10), done.
```

这时在本地就有仓库对应的文件（文件路径取决于你在哪打开git bash，也就是当前bash下的定位）：



- **添加至暂存区**

```
git add "要提交的文件"
```

在本地编辑好代码，准备提交到暂存区，可以使用add指令

常用——提交当前被修改和新增的文件：

```
git add .
```

首先进入到工作目录下：

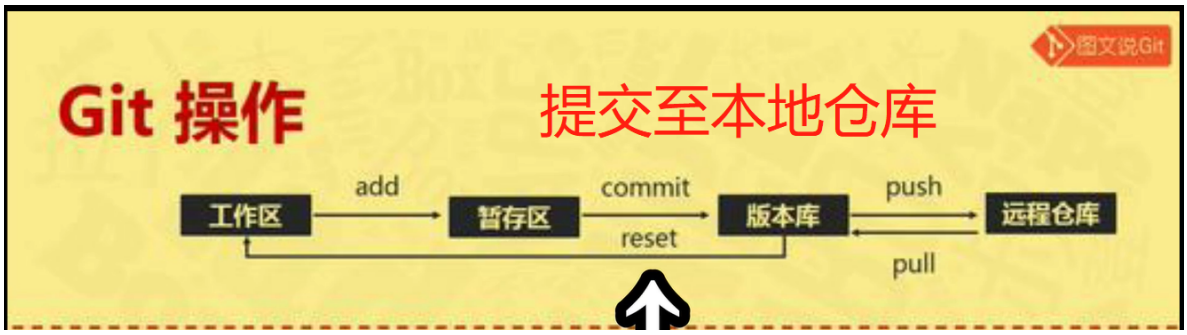
```
七三@LAPTOP-U7MPPR1O MINGW64 ~/Desktop
$ cd Prediction-model-of-learners-satisfaction/

七三@LAPTOP-U7MPPR1O MINGW64 ~/Desktop/Prediction-model-of-learners-satisfaction (m
aster)
$
```

输入指令（**注意add和.之前有空格**）：

```
七三@LAPTOP-U7MPPR1O MINGW64 ~/Desktop/Prediction-model-of-learners-satisfaction (m
aster)
$ git add .
```

- **添加到本地仓库**

```
git commit -m "本次提交描述的语句，更新了xxx"
```



在前面**git add的基础**上，将代码提交至本地仓库：

```
git commit -m "update README"
```

```
七三@LAPTOP-U7MPPR1O MINGW64 ~/Desktop/Prediction-model-of-learners-satisfaction (m
aster)
$ git commit -m "update README"
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

【注】这里由于没有修改，只是把原来的再提交一遍，所以显示"**up to date"——已经是最新**

- **推送至远程仓库**

```
git push
```

在前面**git commit -m**的基础上，使用push指令将本地仓库的代码推送至**远程仓库**，也就是github上的仓库，效果如下：

```
七三@LAPTOP-U7MPPR1O MINGW64 ~/Desktop/Prediction-model-of-learners-satisfaction (master)
$ git push
Everything up-to-date
```

- **从远程仓库拉取更新**

```
git pull
```

假如源程仓库有其他人提交的更新，只需要在工作目录使用pull指令即可**拉取/同步更新到本地仓库**，会自动合并更新到原来的本地文件中。

效果如下：

```
七三@LAPTOP-U7MPPR1O MINGW64 ~/Desktop/Prediction-model-of-learners-satisfaction (master)
$ git pull
Already up to date.
```

2 使用GitHub Desktop管理（图形化）

以上第一点主要介绍了在bash命令行下操作git的过程，当然也可以用GitHub自带的图形化软件操作git。

2.1 下载与安装

安装网址：https://desktop.github.com/

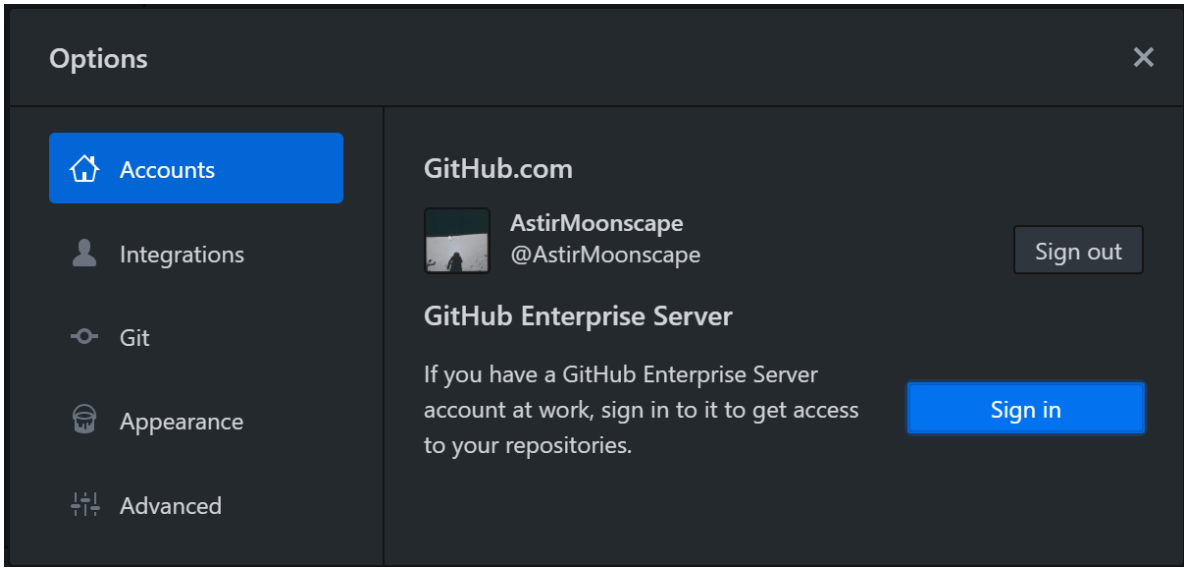
安装过程略

2.2 使用说明

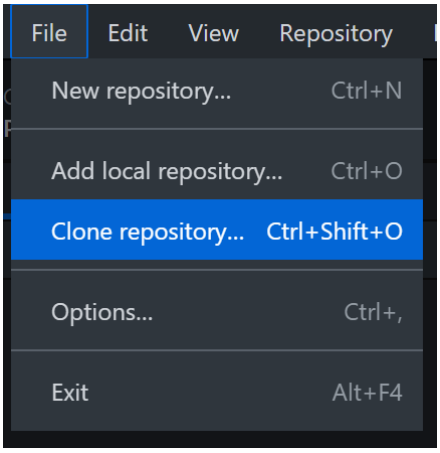
桌面版的操作也是类似于上面git操作的流程，暂存区-本地仓库-远程仓库

- clone仓库

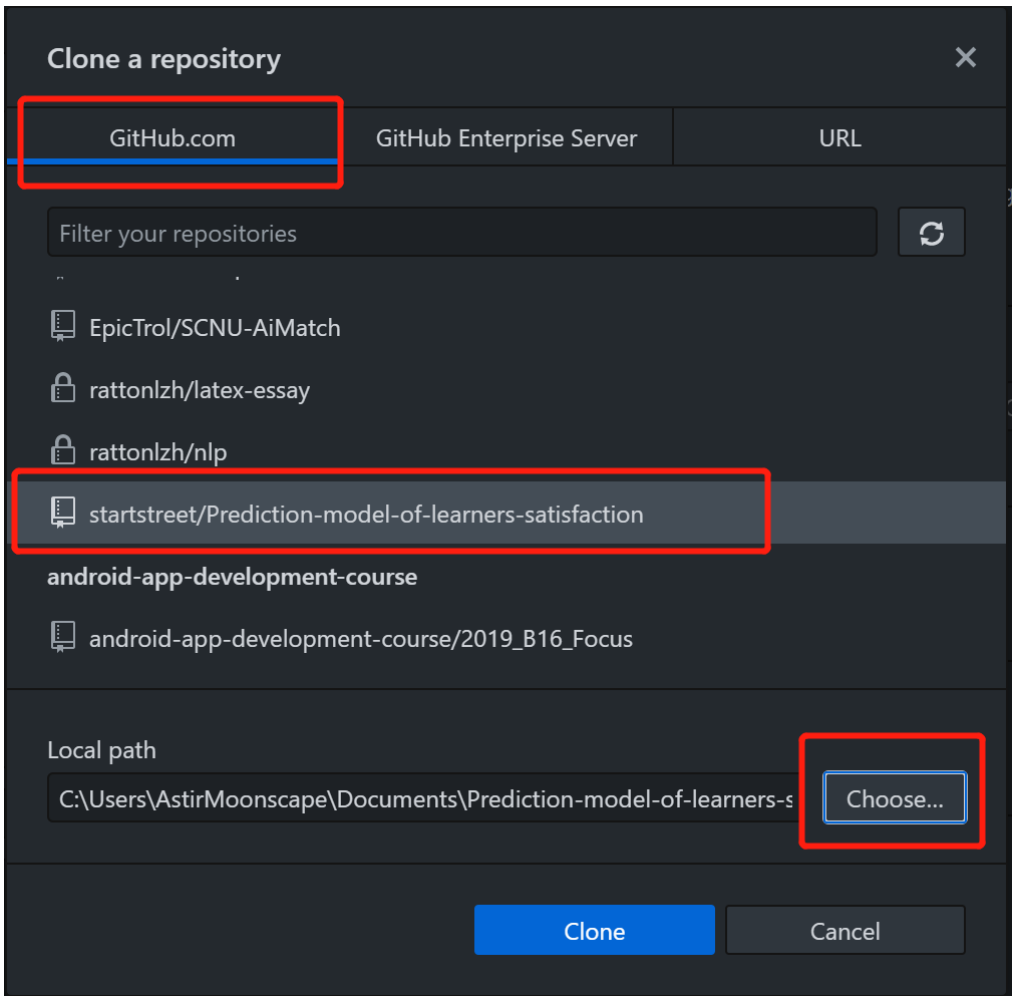
首先在File-Options登陆github账号：



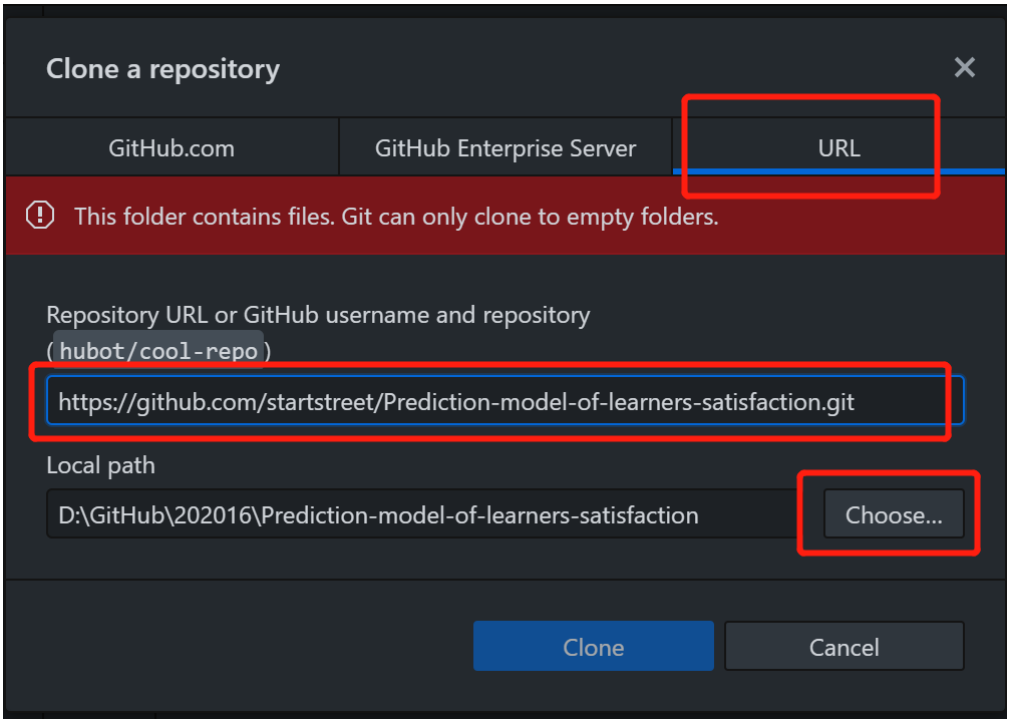
然后选择clone：



可以直接clone你已经加入的仓库，记得下面的**choose**选择要存放的路径：

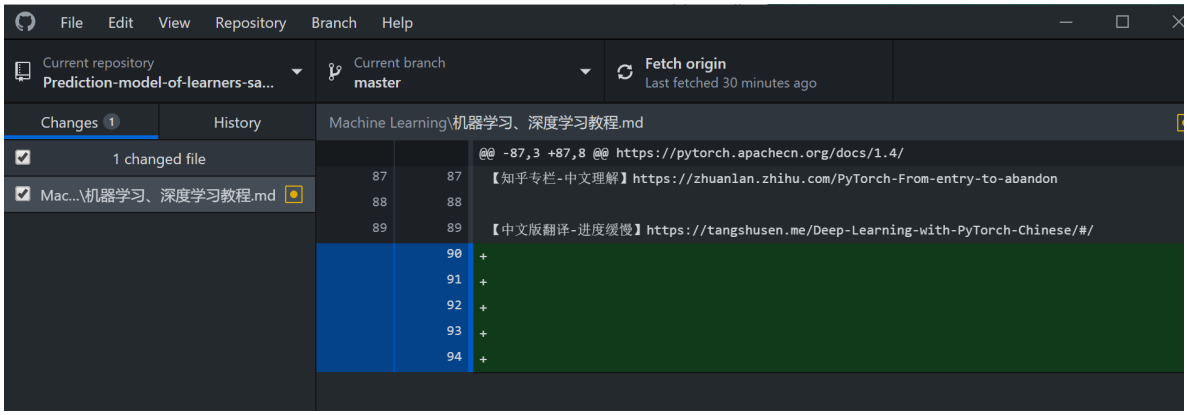


当然也可以复制远程仓库的https地址进行创建：

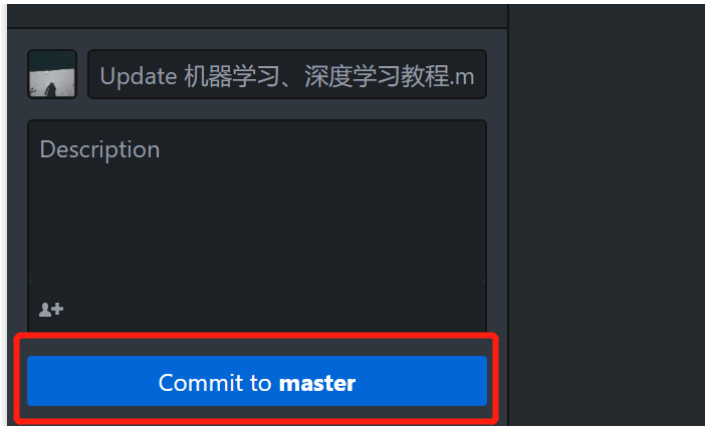


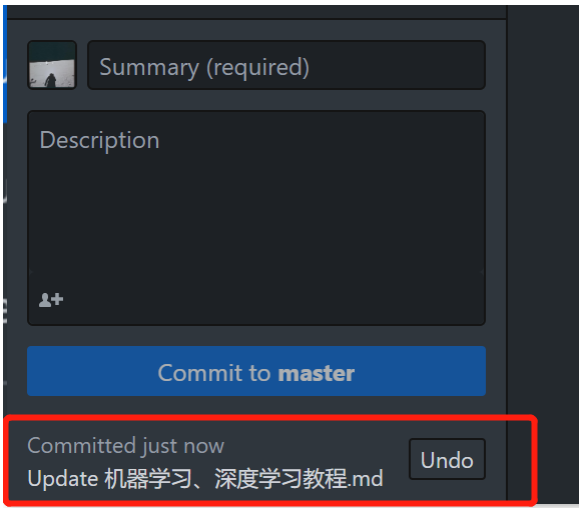
- push提交

当本地代码修改时，客户端会自动识别修改的内容：

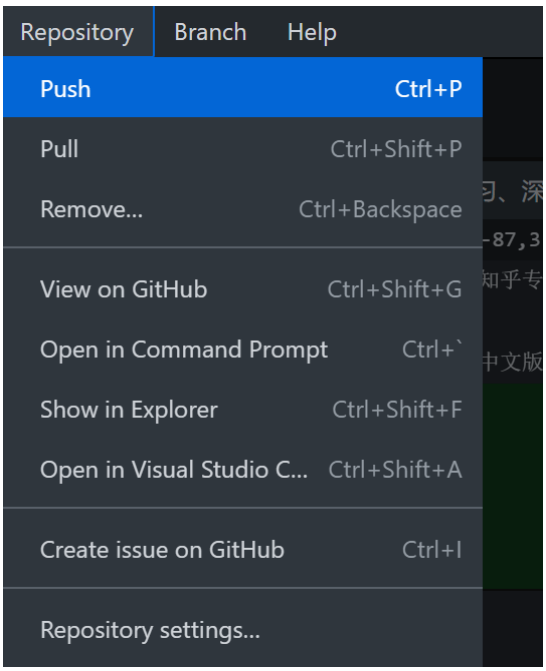


点击左下角的**commit to master**（文字可以添加更新的描述）直接提交到本地仓库：

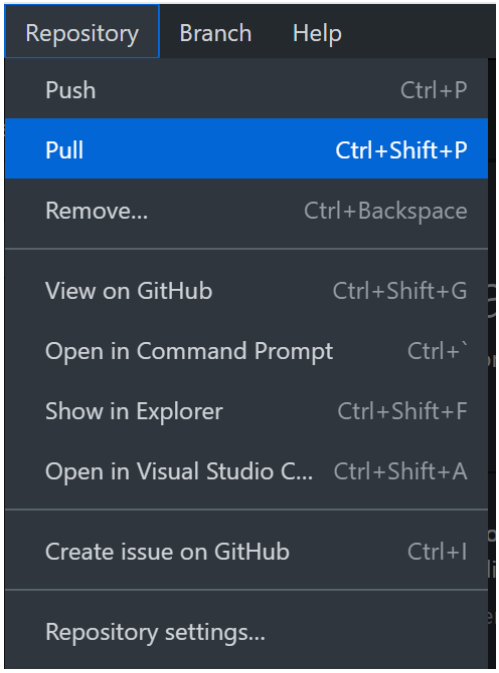




选择**Repository-Push**推送到远程仓库：



- pull拉取



3 关于GitHub访问慢的问题

（修改DNS）https://blog.csdn.net/weixin_44091178/article/details/104557823 亲试，效果不错

（修改IP地址）https://blog.csdn.net/yh0503/article/details/90233216 效果一般

还有就是科学上网~