在Github上同步笔记

当你写了一篇笔记,兴冲冲地发出去之后才发现笔记有错需要更改,而你又拉不下面子的时候,你该怎么做?或者,你想要把笔记保存在云端,以防止意外事件发生(比如水倒在键盘上)时,你又会怎么做?这两个让人很头疼的问题只需要一个工具就可以解决,那就是**GitHub**.

> **什么是GitHub?**
>
> **GitHub**是世界上最大的**代码托管平台**.自然,它也可以当做普通文件的云盘使用.

这篇笔记将会指导你从零开始在**Windows**平台上使用GitHub(非常原始地)保存自己的文件.

首先,我们介绍无GitHub客户端时的操作.

第一阶段:初始设置

1. **安装Git**

   > **Git是什么?它和GitHub有什么关系?**
   > 简单地说,**Git**是管你电脑上文件修改历史的*工具*,而**GitHub**是存这些文件及其历史让大家一起用的*网站*.因此,我们要使用GitHub,最重要的是学会Git.

   Git官网下载地址:https://git-scm.com/downloads.
   注意:
   (1) 安装时记得勾选"**Add Git to PATH**"(环境变量配置),能省很多麻烦;
   (2) 同时,安装时勾选"**Launch Git Bash**"会打开一个类似Powershell的界面,可以打开备用,但如果关闭了也没有关系.
2. **注册GitHub**
   既然GitHub是个类似网盘的网站,我们首先得注册一个账号.官网 https://github.com/,在右上角找到Sign Up即可注册.注意:由于网络问题,GitHub官网可能会出现连不上的问题,请自行解决.
3. **SSH认证**(关键安全/省事步骤)

   > **SSH又是什么?**

> **SSH**(Secure Shell)是一种用于在不安全的网络上进行安全通信的网络协议.而为了我们能够顺利地把文件上传到Github,我们需要给Github提供SSH密钥.

GitHub注册完毕后,如果你还没有打开Git Bash页面,那么首先在Windows的搜索页面搜索"Git Bash",然后打开即可.我们接下来开始创建并认证SSH.

- 3.1 **检查SSH**
  此时你的电脑应该还没有创建SSH.但以防万一,在GIt Bash内输入:

```
cd ~./ssh
```

  这是一条跳转到地址以`/ssh`结尾的文件或文件夹的代码.随后,应该会弹出`no such file or directory`,表明电脑上确实没有SSH.
- 3.2 **创建SSH**
  接下来,在GIt Bash内输入:

```
ssh-keygen -t ed25519 -C "[注册Git时使用的邮箱]"
```

  然后按三次`enter`以启用默认设置.这条代码会使用ed25519(基于椭圆曲线的加密方案)生成一串SSH,随后你应该能在`C/Users/[你的用户名]/.ssh`处找到两个分别名为`id_ed25519`和`id_ed25519.pub`的文件.或者,也可以将`ed25519`替换为`rsa`,采取RSA(基于大数分解)的方案生成SSH.
  `id_ed25519`是私钥,不能公开.而`id_ed25519.pub`是公钥,用记事本打开,复制里面的密钥.或者,在Git Bash内使用

```
cat ~/.ssh/id_ed25519.pub | clip
```

  以直接将公钥复制到剪贴板.
- 3.3 **添加SSH**
  进入Github首页.随后以`头像->Settings->SSH and GPG keys->New SSH Key`顺序操作,在Title处随便填个名字,在Key处粘贴密钥后点击`Add SSH Key`即可.
- 3.4 **确认SSH与配置用户名与邮箱**
  首先在Git Bash内输入:

```
ssh -T git@github.com
```

然后输入 `yes`.如果出现 `Hi! [你GitHub的用户名]`,那么就说名我们的SSH绑定成功了.
随后我们需要输入

```
git -config --global user.name"[你想使用的用户名]"
git -config --global user.email"[你GitHub注册时使用的邮箱]"
```

来配置user.name和user.email.这将会成为你上传文件到GitHub后的身份标识.

4. **创建Repository**

   这是最简单的一步."Repository",即仓库的创建只需来到GitHub首页,然后点击"+",选择"New repository"即可.至于仓库名(Repository name),仓库的描述(description),仓库是否公开(Public/Private),是否需要readme文件,按需填写/选择即可.

第二阶段：本地仓库初始化

虽然我们到现在终于配置完了GitHub的云端仓库,但是不要忘记我们的目的:将笔记上传到云端.为此,我们需要在本地设立一个特别的存储路径,以便将这个路径内的文件上传到GitHub.

1. **创建本地笔记目录**
   依然,在GIt Bash内输入:

   ```
   mkdir -p ~/Documents/Github/Notes && cd ~/Documents/Github/Notes
   ```

   这会在 `C:\Users\[你的用户名]\Documents\Github` 处创建一个 `Notes` 文件夹,并且用 `cd` 命令跳转到该文件夹.
2. **初始化Git仓库**
   随即,输入:

   ```
   git init
   ```

   这会将 `Notes` 文件夹初始化为Git仓库,具体表现为:当你开启显示隐藏的文件后,可在 `Notes` 处发现一个名为 `.git` 的文件夹.

3. **关联远程仓库**
   然后我们要做的就是把本地文件夹和云端仓库关联起来即可.输入:

   ```
   git remote add origin git@github.com:[你GitHub的用户名]/Notes.git
   ```

   这样,我们就创建了一个可以同步到云端的仓库了!
4. **创建基础结构**(可选)
   如果你需要在 `Notes` 内创建子文件夹以区分文件,手动右键添加,或在 `cd` 后使用:

   ```
   mkdir [文件夹名字]
   ```

   即可.
5. **添加忽略规则(防止同步临时文件)**
   考虑到GItHub每个仓库最多只有1GB的容量,并且文件多了不方便管理,我们需要让Git忽略同步临时文件.具体来说,依然在 `Notes` 处使用Git Bash输入:

   ```
   echo "*.tmp" >> .gitignore
   ```

   即可.这会忽略所有扩展名为 `.tmp` 的文件被同步到云端.

第三阶段：首次上传笔记

下面,我们假设Downloads地址处有一个名为"算法笔记.pdf"的文件(开启查看文件拓展名).

1. **复制笔记到对应文件夹**
   手动复制,或者在Git Bash内输入:

   ```
   cp ~/Downloads/算法笔记.pdf ~/Documents/Github/Notes
   ```

   即可将"算法笔记.pdf"复制到GitHub本地仓库.
2. **添加所有文件到暂存区**
   首先确保Git Bash此时地址位于 `~/Github/Notes` .具体操作为在Git Bash输入

   ```
   cd ~/Documents/Github/Notes
   ```

或者手动在文件夹页面右键选择`Open Git Bash here`.
随后,输入

```
git add .
```

注意:`.`不能省略,这表明我们将上传**所有文件**.

3. **创建首次提交**
   在Git Bash内输入:

```
git commit -m "初始提交：添加全部笔记"
```

即可创建首次提交.其中"-m"是你这次提交的具体描述.这是你提交的文件的重要标识.

4. **推送到云端**
   随后,输入:

```
git push -u origin main
```

我们就能把"算法笔记.pdf"上传到云端了.其中,`-u origin main`是为了设置上流跟踪,这可以使得我们之后的提交只用输入`git push`即可.

第四阶段：日常维护

前面所有的配置完毕后,我们接下来只需享受GitHub给我们带来的便利即可.
下面是我们对笔记进行日常维护的步骤.

1. **创建cd的别名**
   首先我们为`cd ~/Documents/Github/Notes`这个命令添加一个alias以方便操作.
   在Git Bash内,输入:

```
nano ~/.bashrc
```

这将打开nano编辑器.然后,输入:

```
alias cd-main='cd ~/Documents/Github/Notes'
```

这将创造一个 `cd-main` 作为 `cd ~/Documents/Github/Notes` 的快捷调用方式.接下来按 `ctrl+x` 退出并保存即可.

2. **进入笔记目录**
   打开Git Bash,输入:

   ```
   cd-main
   ```

   或者手动跳转.
3. **添加所有更新**
   不要忘记这一步.这是由于Git的特性决定的,我们得先创建一个文件的暂存区.

   ```
   git add .
   ```

4. **提交并推送**
   随后,只需要输入:

   ```
   git commit -m "更新笔记" && git push
   ```

   即可.这会自动提交并push到云端.

综上,我们便在没有GitHub Desktop完成了从创建GitHub仓库到进行维护的所有步骤.