### Git使用

如果本文有不太明白的地方请参考上面的文章。

Github: git项目托管网站，请先免费申请一个github帐号：[www.github.com](http://www.github.com/)

Git:分布式版本控制工具，<http://d.download.csdn.net/down/3169511/z_y_liu89>

Github的申请与git的安装我这里就不说了。这个不难。

#### ----//git/github环境配置

##### 一 . github上创建立一个项目

用户登录后系统，在github首页，点击页面右下角“New Repository”

填写项目信息：

**project name**: hibernate-demo

**description** : my first project

点击“**Create Repository**” ； 现在完成了一个项目在github上的创建。

说明：我们创建的是一个github仓库，一个仓库里只能存放（或叫对应）一个项目。

当你创建完成一个仓库的之后，github已经给你一个提示：当你看完了我的文章再来看这个提示就非常清楚了。

[复制代码](javascript:void(0);)

**Global setup:**  
 Set up git  
 git config --global user.name "Your Name"  
 git config --global user.email defnngj@gmail.com  
   
**Next steps:**  
 mkdir hibernaet-demo2  
 cd hibernaet-demo2  
 git init  
 touch README  
 git add README  
 git commit -m 'first commit'  
 git remote add origin git@github.com:defnngj/hibernaet-demo2.git  
 git push -u origin master

**Existing Git Repo?**  
 cd existing\_git\_repo  
 git remote add origin git@github.com:defnngj/hibernaet-demo2.git  
 git push -u origin master

**Importing a Subversion Repo?**  
 Click here   
When you're done:  
 Continue

[复制代码](javascript:void(0);)

##### 二 . 创建密钥

我们如何让本地git项目与远程的github建立联系呢？之里就用的密钥。通俗点叫口令吧！（天王盖地老，宝塔镇河妖。。）

$ cd ~/. ssh 检查本机的ssh密钥

如果提示：No such file or directory 说明你是第一次使用git。

如果不是第一次使用，请执行下面的操作,清理原有ssh密钥。

$ mkdir key\_backup  
 $ cp id\_rsa\* key\_backup  
 $ rm id\_rsa\*

生成新的密钥：

Ssh-keygen –t rsa –C “defnngj@gmai.com”

**注意:** 此处的邮箱地址，你可以输入自己的邮箱地址。在回车中会提示你输入一个密码，这个密码会在你提交项目时使用，如果为空的话提交项目时则不用输入。这个设置是防止别人往你的项目里提交内容。

打开本地C:\Documents and Settings\Administrator\.ssh\id\_rsa.pub文件。此文件里面内容为刚才生成人密钥。

登陆github系统。点击右上角的 [Account Settings](https://github.com/account)--->SSH Public keys ---> add another public keys

把你本地生成的密钥复制到里面（key文本框中）， 点击 add key 就ok了

在git中运行下面命令：

$ git –T git@github.com

如果提示：Hi defnngj You've successfully authenticated, but GitHub does not provide shell access. 说明你连接成功了。

##### 三 . 设置用户信息

这一步不是很重要，貌似不设置也行，但github官方步骤中有，所以这里也提一下。

在git中设置用户名，邮箱

$ git config --global user.name "defnngj"//给自己起个用户名  
$ git config --global user.email "defnngj@gmail.com"//填写自己的邮箱

在github中找到 Account Settings--->Account Admin ,找到一下信息：

Your API token is e97279836f0d415a3954c1193dba522f ---keep it secret! Changing your password will

generate a new token

$ git config --global github.user defnngj //github 上的用户名  
$ git config --global github.token e97279836f0d415a3954c1193dba522f

#### ----//小玩一下git

上面都是准备工作，一次完成，以后就不用设置了。下面内容才是亮点。

先来说说git下常用的几个基本操作，和linux系统的操作是一样一样的：

[复制代码](javascript:void(0);)

$ ls 查看当前目录的内容  
  
$ cd /d 切换到d盘  
  
$ cd java/ 打开当前目录下的java目录  
  
$ cd j(table键) 如果当你想打开java目录且当前目录下只有一个j开头的目录，输入J 然后按键盘上的table键，会自动帮你补齐。  
  
$ cd .. 返回上一级目录

[复制代码](javascript:void(0);)

##### 假如你现在新创建了一个项目，想把它提交到github上面？

假设你创建好了一个项目，并切换到项目的根目录下面：

**$ git status** //查看当前项目下所有文的状态，如果第一次，你会发现都红颜色的，因为它还没有交给git/github管理。

**$ git add .** //（.）点表示当前目录下的所有内容，交给git管理，也就是提交到了git的本地仓库。

**Ps**:git的强大之处就是有一个本地仓库的概念，在没有网络的情况下可以先将更新的内容提交到本地仓库。

**$ git commit –m”new natter ”** //对你更新或修改了哪些内容做一个描述。

**$ git remote add origin git@github.com:defnngj/hibernate-demo.git**

//如果你是第一次提交项目，这一句非常重要，这是你本地的当前的项目与远程的哪个仓库建立连接。

**Ps**: **origin**可以改为别人的名字，但是在你下一次push（提交）时，也要用你修改之后的名字。

**$ git remote -v** //查看你当前项目远程连接的是哪个仓库地址。

**$ git push -u origin master** //将本地的项目提交到远程仓库中。

------------------------------------------------------------

##### 假如，你回到了家，想把公司提交的项目克隆到本地？

如果你是第一次想把github上面的项目克隆到本地或者要克隆别人的项目到地。

**$ git clone git@github.com:defnngj/hibernate-demo.git** //在git下面切换到想存放此项目的文件目录下，运行这条命令就可以将项目克隆下来。

假如本地已经存在了这个项目，而仓库中又有一新的更新，如何把更的合并到本地的项目中？

**$ git fetch origin** //取得远程更新，这里可以看做是准备要取了

**$ git merge origin/master** //把更新的内容合并到本地分支/master

-------------------------------------------

##### 项目中删除了一些文件，如何提交？

假如远程仓库中已经存了aaa这个文件，我fetch了下来，并删除了aaa这个文件，想再push上到远程仓库中，并使远程仓库中的项目被新的修改覆盖（也是是远程仓库中的aaa也被删除）

**$ git status** //可以看到我们删除的哪些文件

**$ git add .** //删除之后的文件提交git管理。

**$ git rm src/com/hzh/hibernate/dao/aaa.java** //移除我们删除的那个文件，不然git不允许我们往远程仓库提交。

Ps: 如果你想删除的是某个目录（java包），这里想移除整个目录的内容。

**$ git rm src/com/hzh/hibernate/bbb/ -r** // **-r** 会把bbb/目录下的所有内容一次性移动。

------------------------------------------------------------------------

##### 远程创建了一个新仓库，本地创建了一个新项目，如何使新的项目与仓库对应起来？

其实，这个也很简单，只是我当时对那些命令不太理解，所以比较模糊，不知如何对应。

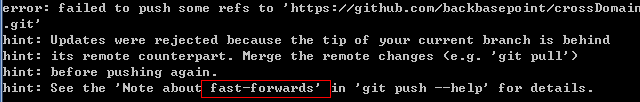
**$ git remote add origin git@github.com:defnngj/hibernate-demo.git**

//还是这个命令，在你push项目之前加上这一句就OK了。

**git@github.com:defnngj/hibernate-demo.git** 就是你常见的新仓库的地址啊。git切换到新项目下，在push之前，加上这一句，我们创建的新仓库就与新项目建立了连接。

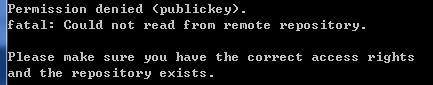
### 常见问题解决

#### 问题一（Non-fast-forward）



问题（Non-fast-forward）的出现原因在于：git仓库中已经有一部分代码，所以它不允许你直接把你的代码覆盖上去，特别对于在Github上新建版本仓库时，因为它为了增加其它人对用户提交的代码的可读性时，默认是要求新建一个README.md文件，和系统会生成.gitignore 文件。所以在push化码到远程仓库时，因为我们本地仓库中没有类似于README.md或.gitignore 等文件，这样就是提示（Non-fast-forward). 有2个方式可以解决这个问题：  
   
强推，即利用强覆盖方式用你本地的代码替代git仓库内的内容，这种方式不建议使用。  
git push -f  
   
2. 先把git的东西fetch到你本地然后merge后再push  
$ git fetch  
$ git merge  
这2句命令等价于  
$ git pull   
可是，这时候又出现了如下的问题：  
上面出现的 [branch "master"]是需要明确(.git/config)如下的内容  
[branch "master"]  
    remote = origin  
    merge = refs/heads/master  
这等于告诉git2件事:  
1，当你处于master branch, 默认的remote就是origin。  
2，当你在master branch上使用git pull时，没有指定remote和branch，那么git就会采用默认的remote（也就是origin）来merge在master branch上所有的改变  
如果不想或者不会编辑config文件的话，可以在bush上输入如下命令行：  
$ git config branch.master.remote origin   
$ git config branch.master.merge refs/heads/master   
之后再重新git pull下。最后git push你的代码，到此步顺利完成时，则可以在Github上看到你新建的仓库以及你提交到仓库中文件了,OK。

#### 问题二（用ssh访问不了）



有两种可能要不是公司不准ssh通讯就是服务器端没有设置SSH Keys

##### Step 1: Check for SSH keys

Have an existing keypair you'd like to use? You can skip to***Step 4***.

First, we need to check for existing ssh keys on your computer. Open up Git Bash and run:

cd ~/.ssh# Checks to see if there is a directory named ".ssh" in your user directory

If it says "No such file or directory" skip to **step 3**. Otherwise continue to **step 2**.

##### Step 2: Backup and remove existing SSH keys

Since there is already an SSH directory you'll want to back the old one up and remove it:

ls# Lists all the subdirectories in the current directory

# config id\_rsa id\_rsa.pub known\_hosts

mkdir key\_backup# Makes a subdirectory called "key\_backup" in the current directory

cp id\_rsa\* key\_backup# Copies the id\_rsa keypair into key\_backup

rm id\_rsa\*# Deletes the id\_rsa keypair

##### Step 3: Generate a new SSH key

To generate a new SSH key, enter the code below. We want the default settings so when asked to enter a file in which to save the key, just press enter.

ssh-keygen -t rsa -C "your\_email@example.com"# Creates a new ssh key using the provided email

# Generating public/private rsa key pair.

# Enter file in which to save the key (/c/Users/you/.ssh/id\_rsa): [Press enter]

Now you need to enter a passphrase.

[Why do passphrases matter?](https://help.github.com/articles/generating-ssh-keys)

# Enter passphrase (empty for no passphrase): [Type a passphrase]

# Enter same passphrase again: [Type passphrase again]

Which should give you something like this:

# Your identification has been saved in /c/Users/you/.ssh/id\_rsa.

# Your public key has been saved in /c/Users/you/.ssh/id\_rsa.pub.

# The key fingerprint is:

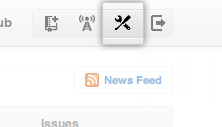
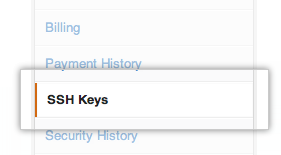
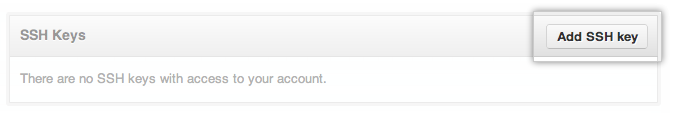
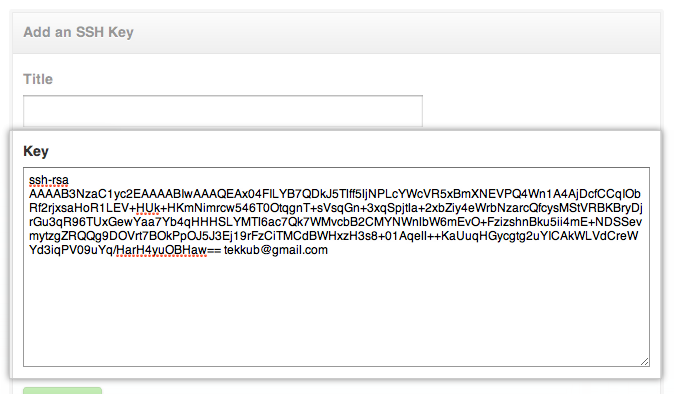
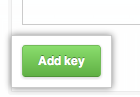
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your\_email@example.com

##### Step 4: Add your SSH key to GitHub

Run the following code to copy the key to your clipboard.

clip < ~/.ssh/id\_rsa.pub# Copies the contents of the id\_rsa.pub file to your clipboard

**Be warned:** it is important to copy the key exactly without adding newlines or whitespace. Thankfully the clip command makes it easy to perform this setup perfectly.

1. Go to your [Account Settings](https://github.com/settings)
2. Click ["SSH Keys"](https://github.com/settings/ssh) in the left sidebar
3. Click "Add SSH key"
4. Paste your key into the "Key" field
5. Click "Add key"
6. Confirm the action by entering your GitHub password

##### Step 5: Test everything out

To make sure everything is working you'll now SSH to GitHub. When you do this, you will be asked to authenticate this action using your password, which for this purpose is the passphrase you created earlier. Don't change the git@github.com part. That's supposed to be there.

ssh -T git@github.com# Attempts to ssh to github

You may see this warning:

# The authenticity of host 'github.com (207.97.227.239)' can't be established.

# RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.

# Are you sure you want to continue connecting (yes/no)?

Don't worry, this is supposed to happen. Verify that the fingerprint matches the one here and type "yes".

# Hi username! You've successfully authenticated, but GitHub does not

# provide shell access.

If that username is correct, you've successfully set up your SSH key. Don't worry about the shell access thing, you don't want that anyway.

If you see "access denied" please consider using [HTTPS](https://help.github.com/articles/set-up-git) instead of SSH. If you need SSH start at[these instructions](https://help.github.com/articles/error-permission-denied-publickey) for diagnosing the issue.

<http://omiga.org/blog/archives/1896> git 介绍