# [【Github教程】史上最全github使用方法：github入门到精通](http://blog.csdn.net/hcbbt/article/details/11651229)

原文 <http://www.eoeandroid.com/thread-274556-1-1.html>

**【初识Github】**

首先让我们大家一起喊一句“Hello Github”。YEAH!就是这样。  
  
Git是一个分布式的版本控制系统，最初由Linus Torvalds编写，用作Linux内核代码的管理。在推出后，Git在其它项目中也取得了很大成功，尤其是在Ruby社区中。目前，包括Rubinius和Merb在内的很多知名项目都使用了Git。Git同样可以被诸如Capistrano和Vlad the Deployer这样的部署工具所使用。同样，eoe.cn客户端的源码也托管在github上。

GitHub可以托管各种git库，并提供一个web界面，但与其它像 SourceForge或Google Code这样的服务不同，GitHub的独特卖点在于从另外一个项目进行分支的简易性。为一个项目贡献代码非常简单：首先点击项目站点的“fork”的按钮，然后将代码检出并将修改加入到刚才分出的代码库中，最后通过内建的“pull request”机制向项目负责人申请代码合并。已经有人将GitHub称为代码玩家的MySpace。  
  
在GitHub进行分支就像在Myspace（或Facebook…）进行交友一样，在社会关系图的节点中不断的连线。  
  
GitHub 使用 git 分布式版本控制系统，而 git 最初是 Linus Torvalds 为帮助Linux开发而创造的，它针对的是 Linux 平台，因此 git 和 Windows 从来不是最好的朋友，因为它一点也不像 Windows。GitHub 发布了GitHub for Windows，为 Windows 平台开发者提供了一个易于使用的 Git 图形客户端。

**GitHub For Windows**

GitHub for Windows 是一个 Metro 风格应用程序，集成了自包含版本的 Git，bash 命令行 shell，PowerShell 的 posh-git 扩展。GitHub 为 Windows 用户提供了一个基本的图形前端去处理大部分常用版本控制任务，可以创建版本库，向本地版本库递交补丁，在本地和远程版本库之间同步。微软也通过CodePlex向开发者提供 git 版本控制系统，而 GitHub 创造了一个更具有吸引力的 Windows 版本。  
  
GitHub上已自动配置的Mac笔记本电脑，一个工具，可以转换设置Linux或Windows机器。  
  
BOXEN是GitHub的自动化工具，设置和配置的Mac笔记本电脑软件开发[3]或其他类型的工作，正在使用他们的开发人员，律师，设计师，付货人，等。我们的想法是准备系统以自动方式和作为无差错尽可能用最少的干预工作。根据GitHub上，与一个新的开发机器上，他的Mac系统成立，并准备在30分钟内提交代码。  
  
BOXEN的基础上收集了大量的几十个木偶模块，使设置的各种软件，如卡桑德拉，MongoDB中，Java软件中，Python和Ruby开发中，节点，JS，nginx的，Skype公司，甚至MINECRAFT。虽然机器上配备了一个预配置，每个用户都可以调整它的配置应有的作用。  
  
**【如何使用】**  
**1.注册账户以及创建仓库**  
要想使用github第一步当然是注册github账号了。之后就可以创建仓库了（免费用户只能建公共仓库），Create a New Repository，填好名称后Create，之后会出现一些仓库的配置信息，这也是一个git的简单教程。  
  
**2.安装客户端msysgit**  
github是服务端，要想在自己电脑上使用git我们还需要一个git客户端，我这里选用msysgit，这个只是提供了git的核心功能，而且是基于命令行的。如果想要图形界面的话只要在msysgit的基础上安装TortoiseGit即可。  
  
装完msysgit后右键鼠标会多出一些选项来，在本地仓库里右键选择Git Init Here，会多出来一个.git文件夹，这就表示本地git创建成功。右键Git Bash进入git命令行，为了把本地的仓库传到github，还需要配置ssh key。  
  
**3.配置Git**

首先在本地创建ssh key；

**[plain]** [view plaincopyprint?](http://blog.csdn.net/hcbbt/article/details/11651229)

1. ssh-keygen -t rsa -C "your\_email@youremail.com"

后面的[your\_email@youremail.com](mailto:your_email@youremail.com)改为你的邮箱，之后会要求确认路径和输入密码，我们这使用默认的一路回车就行。成功的话会在~/下生成.ssh文件夹，进去，打开id\_rsa.pub，复制里面的key。

回到github，进入Account Settings，左边选择SSH Keys，Add SSH Key,title随便填，粘贴key。为了验证是否成功，在git bash下输入：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/hcbbt/article/details/11651229)

1. $ ssh -T git@github.com

如果是第一次的会提示是否continue，输入yes就会看到：You’ve successfully authenticated, but GitHub does not provide shell access 。这就表示已成功连上github。

接下来我们要做的就是把本地仓库传到github上去，在此之前还需要设置username和email，因为github每次commit都会记录他们。

**[plain]** [view plaincopyprint?](http://blog.csdn.net/hcbbt/article/details/11651229)

1. $ git config --global user.name "your name"
2. $ git config --global user.email "your\_email@youremail.com"

进入要上传的仓库，右键git bash，添加远程地址：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/hcbbt/article/details/11651229)

1. $ git remote add origin git@github.com:yourName/yourRepo.git

后面的yourName和yourRepo表示你再github的用户名和刚才新建的仓库，加完之后进入.git，打开config，这里会多出一个remote “origin”内容，这就是刚才添加的远程地址，也可以直接修改config来配置远程地址。  
  
**4.提交、上传**

接下来在本地仓库里添加一些文件，比如README，

**[plain]** [view plaincopyprint?](http://blog.csdn.net/hcbbt/article/details/11651229)

1. $ git add README
2. $ git commit -m "first commit"

上传到github：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/hcbbt/article/details/11651229)

1. $ git push origin master

git push命令会将本地仓库推送到远程服务器。  
git pull命令则相反。  
  
修改完代码后，使用git status可以查看文件的差别，使用git add 添加要commit的文件，也可以用git add -i来智能添加文件。之后git commit提交本次修改，git push上传到github。  
  
**5.gitignore文件**

.gitignore顾名思义就是告诉git需要忽略的文件，这是一个很重要并且很实用的文件。一般我们写完代码后会执行编译、调试等操作，这期间会产生很多中间文件和可执行文件，这些都不是代码文件，是不需要git来管理的。我们在git status的时候会看到很多这样的文件，如果用git add -A来添加的话会把他们都加进去，而手动一个个添加的话也太麻烦了。这时我们就需要.gitignore了。比如一般c#的项目我的.gitignore是这样写的：

**[plain]** [view plaincopyprint?](http://blog.csdn.net/hcbbt/article/details/11651229)

1. bin
2. \*.suo
3. obj

bin和obj是编译目录，里面都不是源代码，忽略；suo文件是vs2010的配置文件，不需要。这样你在git status的时候就只会看到源代码文件了，就可以放心的git add -A了。  
  
**6.tag**  
  
我们可以创建一个tag来指向软件开发中的一个关键时期，比如版本号更新的时候可以建一个“v2.0”、“v3.1”之类的标签，这样在以后回顾的时候会比较方便。tag的使用很简单，主要操作有：查看tag、创建tag、验证tag以及共享tag，这些下面的博客中有详细讲解。  
  
  
**【Github的相关使用文章】  
Git介绍,安装,Git+Git flow使用：**[**http://my.eoe.cn/fogs/archive/799.html**](http://my.eoe.cn/fogs/archive/799.html)  
**Git 指令集：**[**http://my.eoe.cn/iceskysl/archive/463.html**](http://my.eoe.cn/iceskysl/archive/463.html)  
**在mac上安装git-flow过程：**[**http://my.eoe.cn/iceskysl/archive/118.html**](http://my.eoe.cn/iceskysl/archive/118.html)  
**git fetch 的简单用法:更新远程代码到本地仓库：**[**http://my.eoe.cn/com360/archive/3533.html**](http://my.eoe.cn/com360/archive/3533.html)  
**git 如何让单个文件回退到指定的版本：**[**http://my.eoe.cn/com360/archive/3351.html**](http://my.eoe.cn/com360/archive/3351.html)  
**如何使用Github上的开源项目：**[**http://my.eoe.cn/fengyiyezi/archive/3427.html**](http://my.eoe.cn/fengyiyezi/archive/3427.html)  
**Window（8）下安装 MSysGit 、gitflow 、GitHub：**[**http://my.eoe.cn/sunxun/archive/158.html**](http://my.eoe.cn/sunxun/archive/158.html)  
**git 打tag：**[**http://my.eoe.cn/xiayang6/archive/446.html**](http://my.eoe.cn/xiayang6/archive/446.html)  
**基于Github参与eoe的开源项目指南：**[**http://my.eoe.cn/iceskysl/archive/3195.html**](http://my.eoe.cn/iceskysl/archive/3195.html)  
**Git stash 使用方法：**[**http://my.eoe.cn/sunxun/archive/190.html**](http://my.eoe.cn/sunxun/archive/190.html)  
**Git tag的使用：**[**http://my.eoe.cn/futurexiong/archive/1943.html**](http://my.eoe.cn/futurexiong/archive/1943.html)

**大白话讲解如何给github上项目贡献代码：**[**http://my.eoe.cn/leigo/archive/3221.html**](http://my.eoe.cn/leigo/archive/3221.html)

最后再推荐几篇社区里介绍知道github使用的帖子：  
**githup的使用**  
<http://www.eoeandroid.com/thread-272837-1-1.html>  
  
**Eclipse上GIT插件EGIT使用手册**  
<http://www.eoeandroid.com/thread-273360-1-1.html>  
  
**ubuntu下git服务器的搭建**  
<http://www.eoeandroid.com/thread-273167-1-1.html>  
  
git ， vim ， ls 全局配置  
<http://www.eoeandroid.com/thread-229638-1-1.html>  
  
以上希望对大家学习起到积极的作用，一个好的程序猿势必要学会github的使用。如果本文大家觉得还不错，就告诉你身边的朋友吧，如果觉得看得过去那么就分享一下吧，如果觉得有待修改，那么请指出不足并且给打赏几个e币。最后感谢百度的无私支持，以及某个人的博客（说真的真的忘了他的地址是啥了），Hello Github。

# 【Github教程】实战命令1 – git方式

1. git init // 建立初始库
2. git add . // 添加所有文件到索引
3. git add ls // 添加ls目录
4. git config --global user.email "mooling@yeah.net"
5. git config --global user.name "mooling"
6. git commit –m “comments” // 提交变更
7. git remote add emacs-profile <https://github.com/mooling/emacs-profile.git>
8. git push -u emacs-profile master //-u 是提示用户名、密码

删除文件用 git rm

改文件名用 git mv

# 【Github教程】实战命令1 – ssh方式

1. ssh-keygen -t rsa -C [mooling@yeah.com](mailto:mooling@yeah.com)
2. cp ~/.ssh/id\_rsa.pub to github
3. ssh -T git@github.com // use to verify
4. git config --global user.name "mooling"
5. git config --global user.email "mooling@yeah.net"
6. git remote add ep  [git@github.com:mooling/emacs-profile.git](mailto:git@github.com:mooling/emacs-profile.git)
7. git push ep master