Git

参考文档 ：<http://www.runoob.com/git/git-basic-operations.html>

<http://www.yiibai.com/git/git_push.html>

http://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000/0013744142037508cf42e51debf49668810645e02887691000

1：切换目录 cd Path 切换到所在目录 右击 Bit Bash

2：git Clone git clone path path 是/ 这个来分离的

3：提交添加所有的文件到缓存git add .

4：提交 git commit –m “remark”

5: push git push

6: 删除命令 物理文件也被删除 git rm -f \*

7: 提交的缓存里面删除 ，不删除物理文件 git rm -f --cached \*

8: 提交缓存删除 git rm --cached \*.docx 匹配的是正在表达式

9: 查看缓存详细信息 查看已缓存的改动： git diff –cached

10：查看提交之后信息 git status

11: 配置用户名邮箱 git config user.name “Colin” git config user.email [940912218@que.com](mailto:940912218@que.com)

12:查看全局配置信息 git config –l

13:添加远程仓库 remote origin already exists

删除远程git仓库

git remote rm origin

git remote add origin git@github.com:zhanghuming/learngit.git

$ git push -u origin master

14: failed to push some refs

因是github中的README.md文件不在本地代码目录中

git pull --rebase origin master

git push -u origin master 第一次推送master分支的所有内容；

git push origin master 推送最新修改；

15: git rebase –abort

16: Updates were rejected because a pushed branch tip is behind its remote

解决方案：<http://blog.csdn.net/shiren1118/article/details/7761203>

 git push -u origin master -f

17:回到指定版本 git reset –hard

18:查看git 日志 git log

19：查看记录你的每一次命令 git reflog

20:查看具体的差异化 git diff HEAD read.txt 工作区域与commit版本

21:查看具体的差异化git diff –cached read.txt 已经暂存起来的文件(staged)和上次提交时的快照之间(HEAD)的差异

22：git checkout –read.txt 种是readme.txt自修改后还没有被放到暂存区，现在，撤销修改就回到和版本库一模一样的状态.

种是readme.txt已经添加到暂存区后，又作了修改，现在，撤销修改就回到添加到暂存区后的状态.

23:git checkout . 这条命令最危险！会取消所有本地的  修改（相对于暂存区）。相当于用暂存区的所有文件直接覆盖本地文件，不给用户任何       确认的机会！

24:git rm test.txt 从版本库中删除该文件

25: ssh-keygen -t rsa -C "Hell@qq.com" 生成ssh 的公钥与私钥 若果想下次不输入key，在提示生成的时候，都用空格，不然下次再push的时候还是会提示输入key

26: $ cd ~/.ssh  检查本机是否有ssh key设置

27: ssh -T [git@github.com](mailto:git@github.com) 测试ssh keys是否设置成功。

28:ls 列出目录

29:git checkout –b dev 创建dev分支，然后切换到dev分

30:git branch 查看当前分支

31: git checkout master 就可以切换回master分支

32: git merge dev 把dev分支的工作成果合并到master分支上

33:git branch 查看分支

34：git branch dev 创建dev分支

35：git checkout name 切换分支

36：git branch–d name 删除分支

37: git push -u origin dev git提交新建分支到远程服务器

38: git checkout a69e3ae 拉去指定版本 sha

39:git pull origin master 更新到主分支的最新版本

40: 指定到指定文件解决完冲突之后，在

$ git add readme.txt

$ git commit -m "conflict fixed"

执行如下代码，提交

41:git branch –D name 强制删除分支

42: git merge --no-ff -m "merge with no-ff" dev

分支历史上就可以看出分支信息

43:主干mster 发布，branch 开发，开发完成合并会主干

git checkout –b dev001

git add .

git commit –m “update dev001”

git push origin dev001

git branch master

git branch -- merge devoo1

git push origin master

git branch –d dev001

完成整个的流程，但是github上面会存在dev001 的记录，如果想删除掉手动删除即可

44: git stash

遇到dev分支没有提交，bug分支亟待修改，修改完成合并分支，继续开发，再次回到dev环境 继续开发，开发完成再次合并提交情况。

git stash apply 但是恢复后，stash内容并不删除，你需要用git stash drop来删除

git stash pop，恢复的同时把stash内容也删了

45:查看远程git信息 git remote –v

46:多人协作开发，

git checkout –b dev origin/dev

git add .

git commit –m “update dev”

如果冲突了

git pull origin dev 拉去最新分支

修改文件

git add .

git commit –m “update dev”

git push origin dev

然后合并分支就可以了哈

47: git 建立tag git tag v1.0

48:git 打包 git push origin v1.0 上传到仓储建立v1.0 release 发布版本

49：删除远程服务器上面的tag git push origin :refs/tags/v0.9

50:git 忽略文件

Vim .gitignore

I 编辑

： 进入命令行

Wq+enter 退出保存即可

git add -f App.class 强制提交文件

git check-ignore -v App.class 来检查文件哪里出了问题哈