# 1 Sign up and download applications

GITHUB: jellyzxc jelly.zhang@me.com

Github for Windows

Git Bash

**Recommend:**

Notepad++: UTF-8 without BOM

**DONOT** use Windows note as it adds 0xefbbbf at the beginning of each file

# 2 Configure Git

ssh-keygen -t rsa -C “jelly.zhang@me.com”

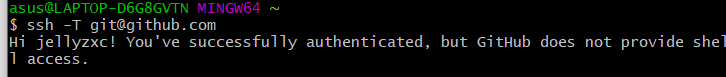
cat c:/Users/asus/.ssh/id\_rsa.pub

# id\_rsa is private key ，id\_rsa.pub is public key

$ cat c:/Users/asus/.ssh/id\_rsa.pub

ssh-rsa  jelly.zhang@me.com

ssh -T git@github.com #test



git config --global user.name “jellyzxc”

git config --global  user.email “jelly.zhang@me.com”

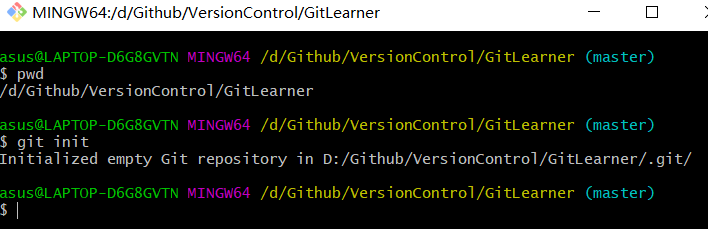
# **3 Create local repository**

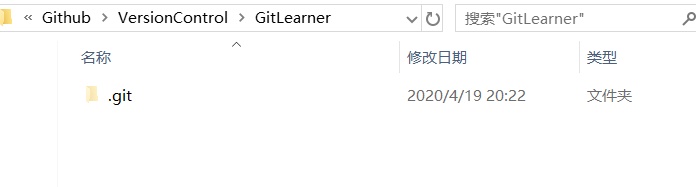
## 3.1 Initialized empty Git repository

#Initialized empty Git repository in /d/Github/VersionControl/GitLearner /.git/

cd /d/Github/VersionControl/GitLearner

git init

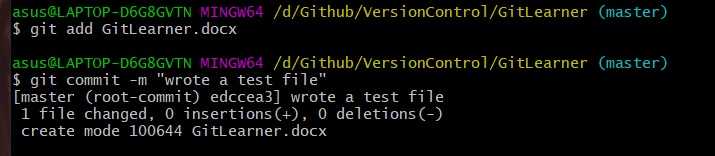




## 3.2 Commit a file to repository

git **add** GitLearner.docx

git **commit** -m "wrote a test file"



## 3.3 Control the version

git **diff** readme.txt

git **status**

git **log**

git **reset** --hard HEAD^ readme.txt

# HEAD: current version HEAD^:the last version HEAD~100 the 100th version

git **reflog** #old log

## 3.4 revoke update/commit



git **checkout** -- readme.txt

#go back to the nearest status when git commit, or git add

git **reset** HEAD readme.txt

# Changes in the stage area are returned to the workspace

## 3.5 Remove a file

rm 'test.txt' #delete local

git **rm** test.txt # delete stage

git commit -m "remove test.txt" #delete all

git checkout -- test.txt #**get it back**

<https://www.liaoxuefeng.com/wiki/896043488029600/896827951938304>

# 4 Connect remote repository

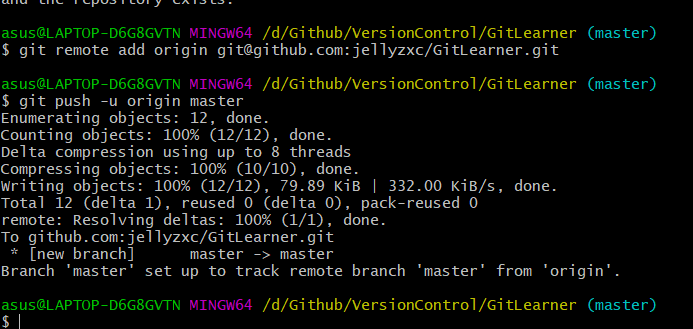
## 4.1 Local to remote

#at the local GitLearner repository

git **remote add origin** git@github.com:jellyzxc/GitLearner.git

git **push -u** origin master #the first time to push all file in master

#**origin** : the default name of remote repository



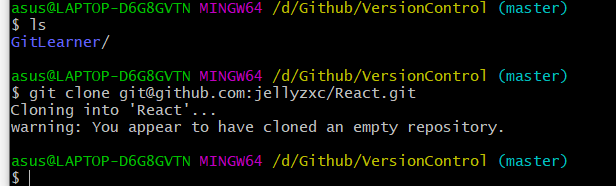
git **add** GitLearner.docx

git **commit** -m "update 20200419"

git **push** origin master

## 4.2 remote to local

git **clone** git@github.com:jellyzxc/React.git



git **pull** git@github.com: jellyzxc/VersionControl.git

VersionControl

git remote add origin git@github.com:jellyzxc/VersionControl.git

git clone https://github.com/jellyzxc/VersionControl.git

# 5 Branch management

GitHub encourage us to create a new branch and merge it to master ,then delete the branch when we finish the task. It is easy but much safer.

## 5.1 Crate a new branch and switch to this branch

### 5.1.1 create branch locally

git **checkout -b** dev or git **switch -c** dev

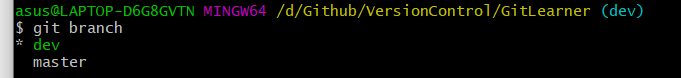
Equal to

git **branch** dev #create a branch called dev

git **checkout** dev #switch to dev

git **branch** #check current branch





### 5.1.2 remote branch

The remote branch is automatically created when you push it to the remote server. So, when you feel ready for it, you can just do:

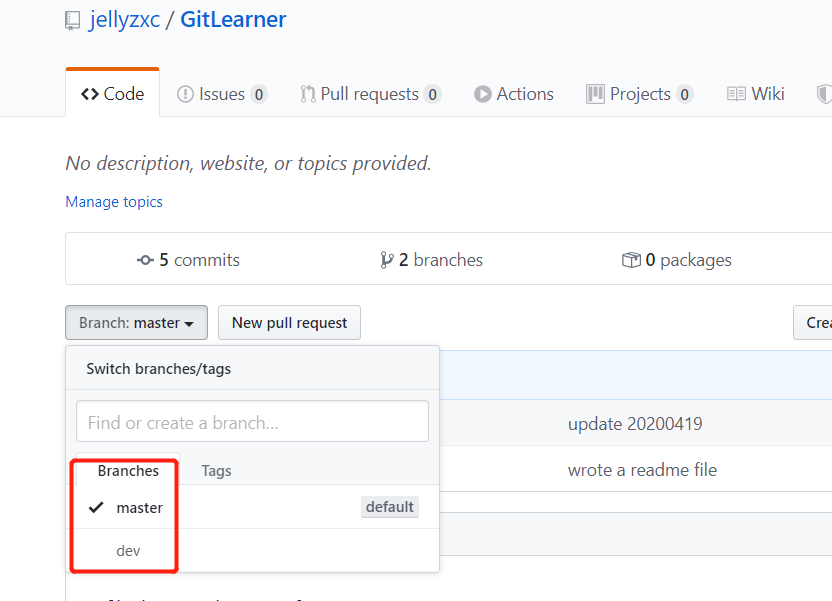
git push <remote-name> <branch-name>  => git **push** **origin** **dev**

git **add** GitLearner.docx

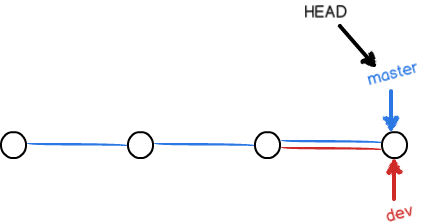
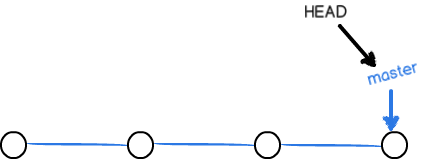
git **commit** -m "update 20200504"

git **push** origin **dev**





## 5.2Merge to master and delete branch

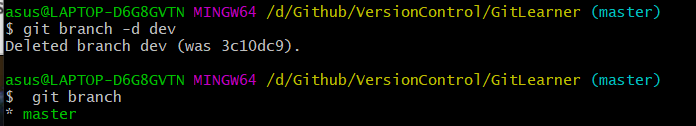
### 5.2.1 Local

git **checkout** master #go back to master git **switch** master

git **merge** dev #merge :change

git branch **-d** dev #delete dev

git branch **-D** brach1 #delete even not merge



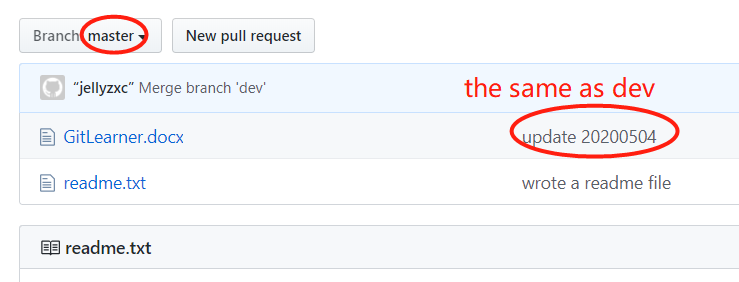
### 5.2.2 remote

git checkout master

git **pull origin master**

git merge dev

git **push origin master**



## 5.3 deal with conflict

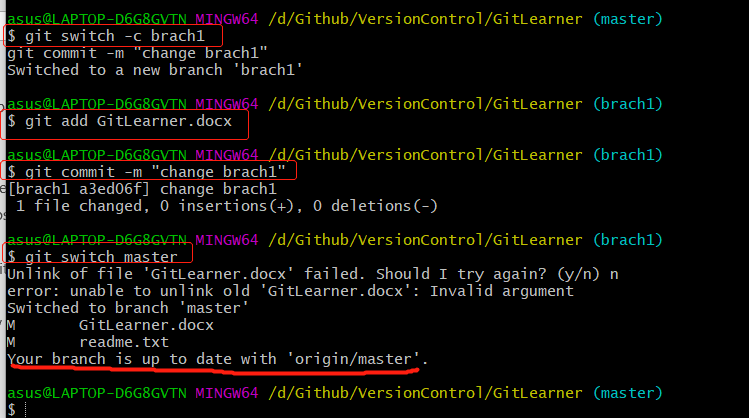
**Update a file and commit in brach1**

git **switch -c** brach1

git **add** GitLearner.docx

git **commit** -m "change brach1"

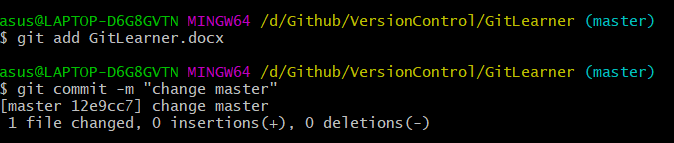
git **switch** **master**



**Update the same file and commit in master**

git **add** GitLearner.docx

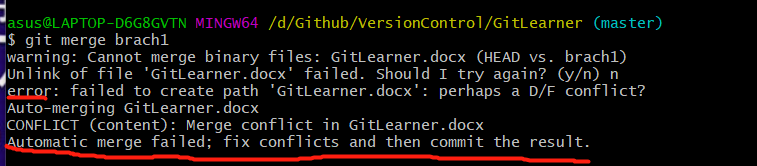
git **commit** -m "change master"

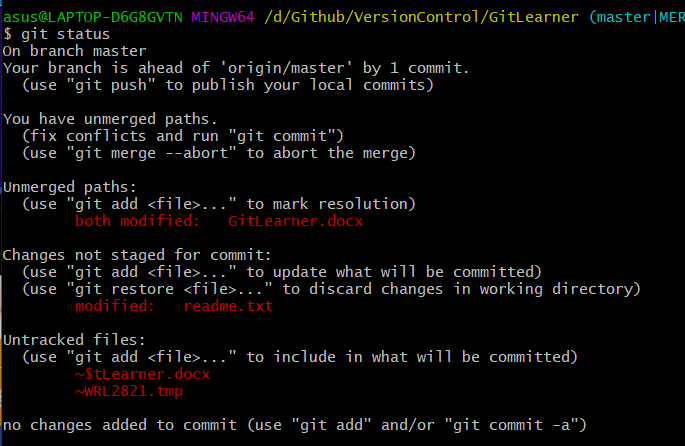




**Merge and conflicts occurred**

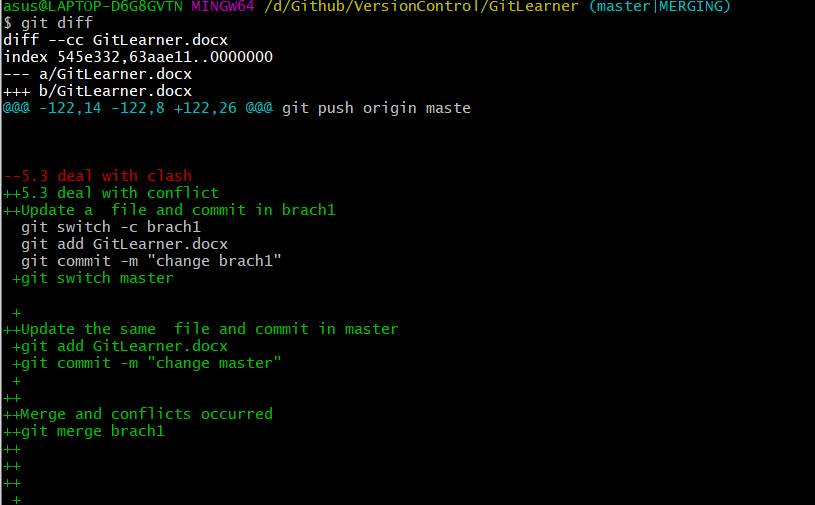
git merge brach1





The most direct way to resolve a merge conflict is to edit the conflicted file

git **diff**



## 5.4 Stash current work

git stash

takes your uncommitted changes (both staged and upstaged), saves them away for later use

the stash is local to your Git repository; stashes are not transferred to the server when you push.