# 课程教学目标:Ps及git

1. **知识目标**
2. Ps基本操作,主要是要会切图
3. 安装git
4. 注册GitHub
5. 使用Git与GitHub管理项目
6. **能力目标**
7. 了解Ps基础操作,会用Ps切图
8. 会安装Git,了解Git基本操作
9. 会注册GitHub,了解GitHub的基本功能
10. 熟练使用Git管理GitHub项目
11. **教学过程**
12. Ps图层

使用ps显示了解啥是图层

1. Ps测量

主要是测量一些高宽,字体大小,颜色等.

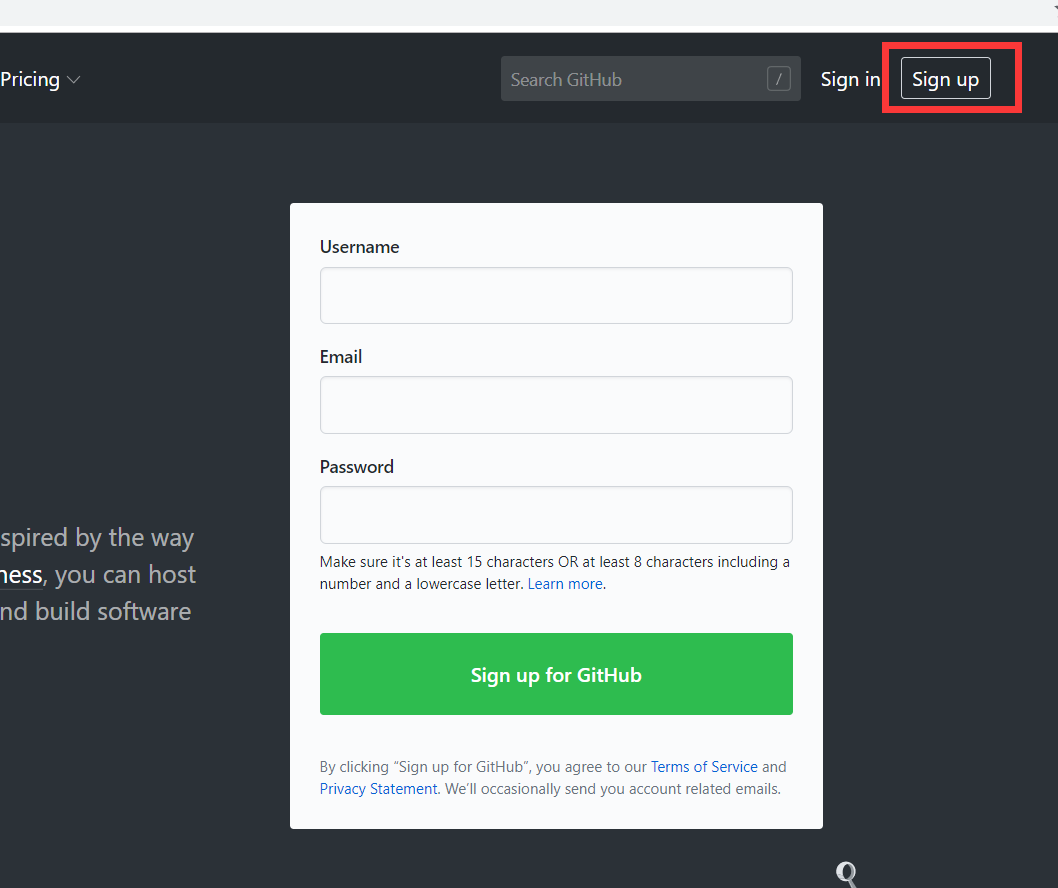
1. Ps切图

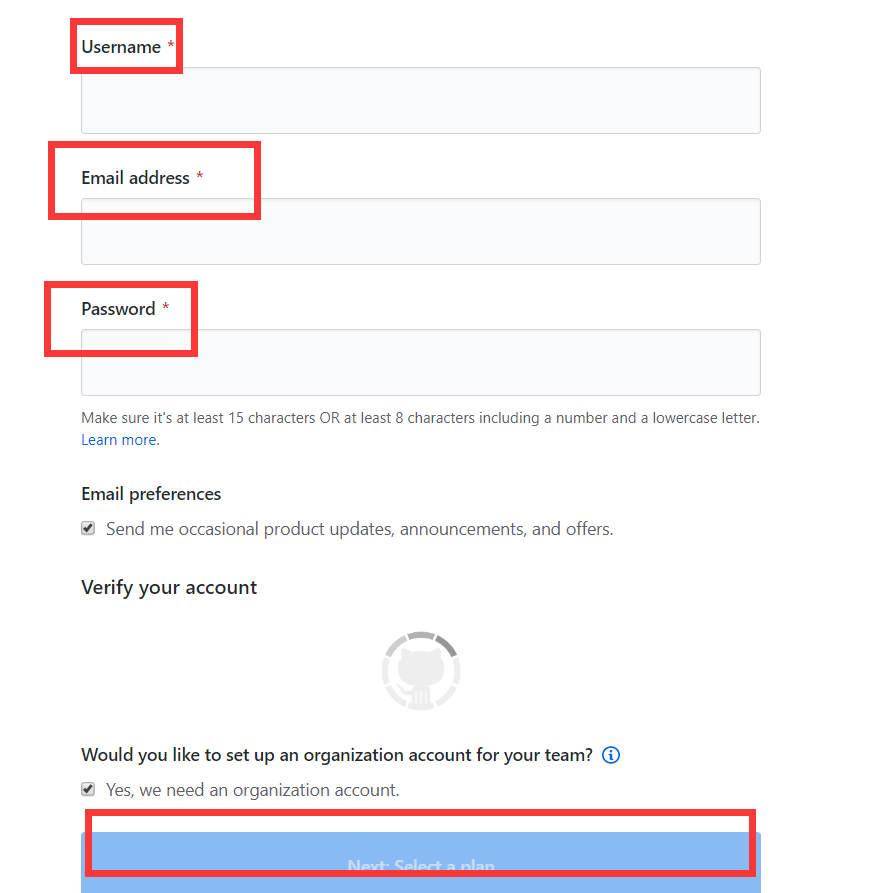
演示如何切多张图,如何切透明背景图

1. 注册GitHub,码云

注册GitHub步骤:

1. <https://github.com/>



1. 

注册码云:

1. 地址: <https://gitee.com/>

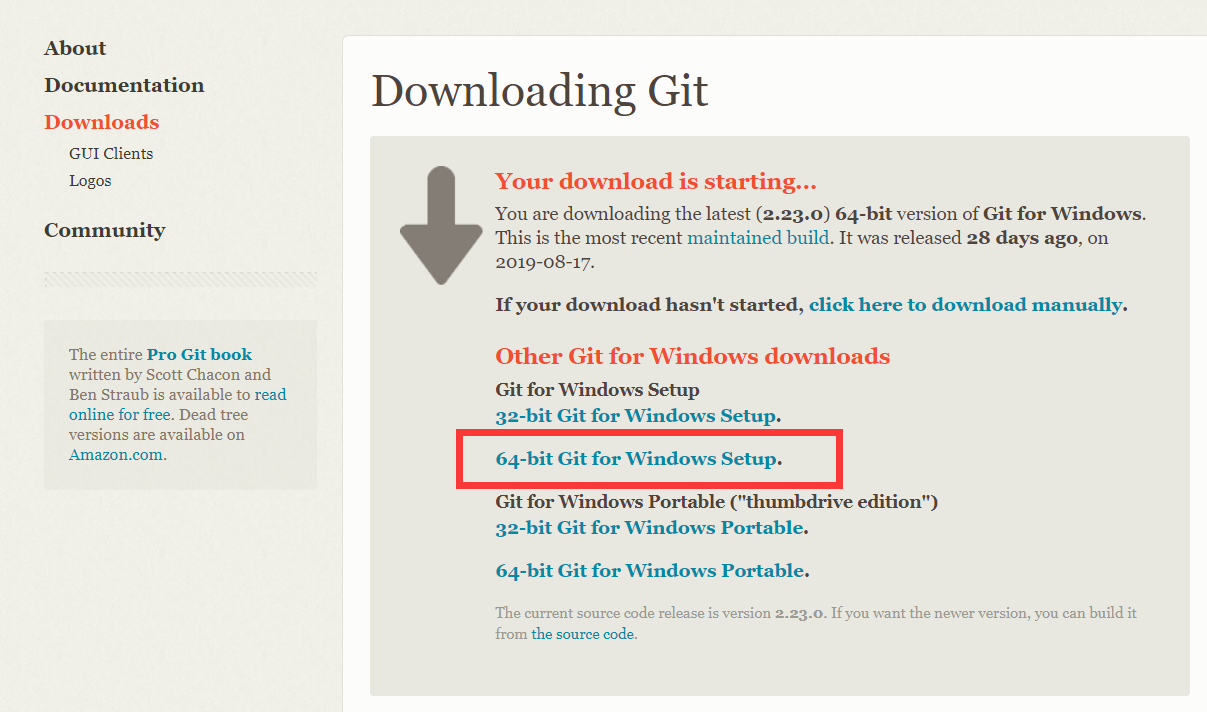






1. 安装Git

在Windows上使用Git，可以从Git官网直接下载安装程序，（网速慢的同学请移步国内镜像），然后按默认选项安装即可。



安装完成后，在开始菜单里找到“Git”->“Git Bash”，蹦出一个类似命令行窗口的东西，就说明Git安装成功！

1. 使用Git与GitHub管理项目

这个地址是推荐比较好的git学习网站

<https://www.liaoxuefeng.com/wiki/896043488029600/900003767775424>

因为Git是分布式版本控制系统，所以，每个机器都必须自报家门：你的名字和Email地址,同事一起,就知道是谁提交的代码

git config --global user.name "Your Name"

git config --global user.email [email@example.com](mailto:email@example.com)

查看自己设置的用户名和密码

git config user.name

git config user.email

查看配置所有的信息

git config –list

1 创建版本库：

1.1 创建空目录

mkdir learngit 创建空目录

cd learngit 进入到目录里

pwd 显示当先目录

1.2 git init 通过git init命令把这个目录变成Git可以管理的空仓库：

2 把文件添加到版本库

1.1 用命令git add告诉Git,实际上就是把文件修改添加到暂存区

git add readme.txt [如何一下添加所有文件，git add .]

执行上面的命令，没有任何显示，没有消息就是好消息，说明添加成功。

1.2 用命令git commit告诉Git，实际上就是把暂存区的所有内容提交到当前分支。

git commit -m "wrote a readme file"

1.3 版本回退

git log 命令显示从最近到最远的提交日志

git reset --hard HEAD^ 回退到上一个版本

git reset --hard 1094a（commit）回退到某个版本

git reflog 用来记录你的每一次命令

git status 查看文件状态，有没有修改

git checkout -- file 丢弃工作区的修改

rm test.txt 删除这个test文件

git rm test.txt 版本库中删除该文件

git remote add origin [git@github.com:wuhaijun/learngit.git](mailto:git@github.com:wuhaijun/learngit.git) 本地关联的就是我的远程库

git push -u origin master

把本地库的所有内容推送到远程库上

由于远程库是空的，我们第一次推送master分支时，加上了-u参数， Git不但会把本地的master分支内容推送的远程新的master分支，还会把本地的master分支和远程的master分支关联起来，

在以后的推送或者拉取时就可以简化命令。

ssh-keygen -t rsa -C "youremail@example.com" 创建SSH Key

1.4 从远程库克隆

git clone git@github.com:wuhaijun/learngit.git

提取远程仓库 git fetch

git merge

删除远程仓库

git remote rm [别名]

查看远程仓库

git remote -v

1.5 分支管理

1.5.1 git branch dev 创建分支

git checkout dev 切换到分支 或者 git switch dev

git checkout -b dev 创建并切换到分支

或 git switch -c dev

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

1.5.3 git branch -d dev 删除分支dev

git branch -D dev 强行删除dev

1.5.4 git branch 查看分支

1.6 解决冲突

创建分支并切换到dev1上

git checkout -b dev1

修改readme.txt最后一行，改为：

Creating a new branch is quick AND simple.

提交dev1修改

git add readme.txt

git commit -m "AND simple"

切换分支到master

git checkout master

在master分支上把readme.txt文件的最后一行改为：

Creating a new branch is quick & simple.

提交master修改

git add readme.txt

git commit -m "& simple"

将dev1合并到master

git merge dev1

修改同一个地方会有冲突

冲突如下形式：

<<<<<<< HEAD

Creating a new branch is quick & simple.

=======

Creating a new branch is quick AND simple.

>>>>>>>

将冲突解决后 并提交

git add readme.txt

git commit -m "conflict fixed"

看到分支的合并情况

git log --graph --pretty=oneline --abbrev-commit

最后在删除分支dev1

git branch -d feature1