

# git 远程仓库 GitHub

## 从入门到工作：入门阶段

# 版权声明

本内容版权属杭州饥人谷教育科技有限公司（简称饥人谷）所有。

任何媒体、网站或个人未经本网协议授权不得转载、链接、转贴，  
或以其他方式复制、发布和发表。

已获得饥人谷授权的媒体、网站或个人在使用时须注明「资料来源：饥人谷」。

对于违反者，饥人谷将依法追究责任。

# 联系方式

如果你想要购买本课程  
请微信联系 **xiedaimala02** 或 **xiedaimala03**

如果你发现有人盗用本课程  
请微信联系 **xiedaimala02** 或 **xiedaimala03**

# 代码需要储存在云端

比如你需要在公司和家里写代码

比如你的笔记本被女朋友的奶茶泡坏了

比如你不小心运行了 `rm -rf /`

# GitHub 可以存储你的代码

只需要两行命令

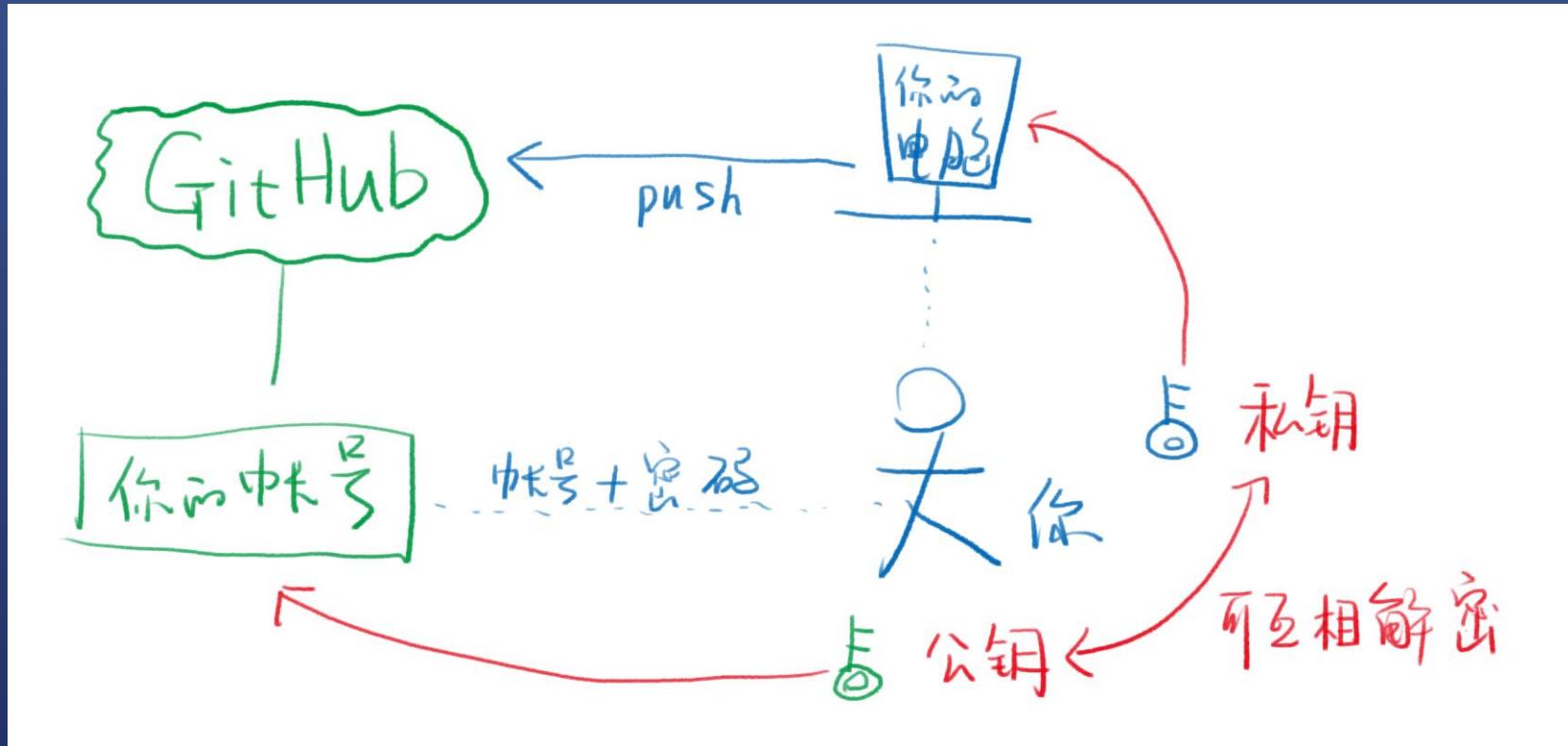
```
git remote add origin git@xxxxxxxxxx
```

```
git push -u origin master
```

# 但是我们要先解决一个问题

GitHub 怎么知道是你在操作你的代码

# SSH key 验证身份



# 一些事实

- **推理**

- ✓ 你在 GitHub 上有一个账号。
- ✓ 你的 git 仓库在你的电脑上。
- ✓ GitHub 怎么知道这个电脑和这个账号都属于你呢？

- **答案**

- ✓ 每次都输入一次密码（HTTPS协议），太不方便
- ✓ 使用 ssh key
- ✓ 电脑上放私钥，GitHub 账号里留下公钥
- ✓ 上传代码是用私钥加密，GitHub 用公钥解密
- ✓ 如果解开了，说明是配对的
- ✓ 同理，你怎么知道对方是 GitHub 呢？也需要 GitHub 提供一个公钥给你，所以第一次连接 GitHub 的时候要选择 yes 来接受对方的公钥。这个知识点不重要。

# 如何生成 ssh key

- 生成 ssh key
  - ✓ GitHub 有[帮助文档](#)
  - ✓ 运行 `ssh-keygen -t rsa -b 4096 -C 你的邮箱`
  - ✓ 然后一直回车，直到没有提示
  - ✓ `cat ~/.ssh/id_rsa.pub` 得到公钥内容，粘贴到 GitHub
  - ✓ 打开 GitHub，在设置页面填入公钥
- 如何测试配对成功了
  - ✓ `ssh -T git@github.com`
  - ✓ 如果问你 yes/no，请回答 yes 并回车

设置了公钥  
就能上传和下载了  
通过 git pull / git push / git clone

# 上传代码

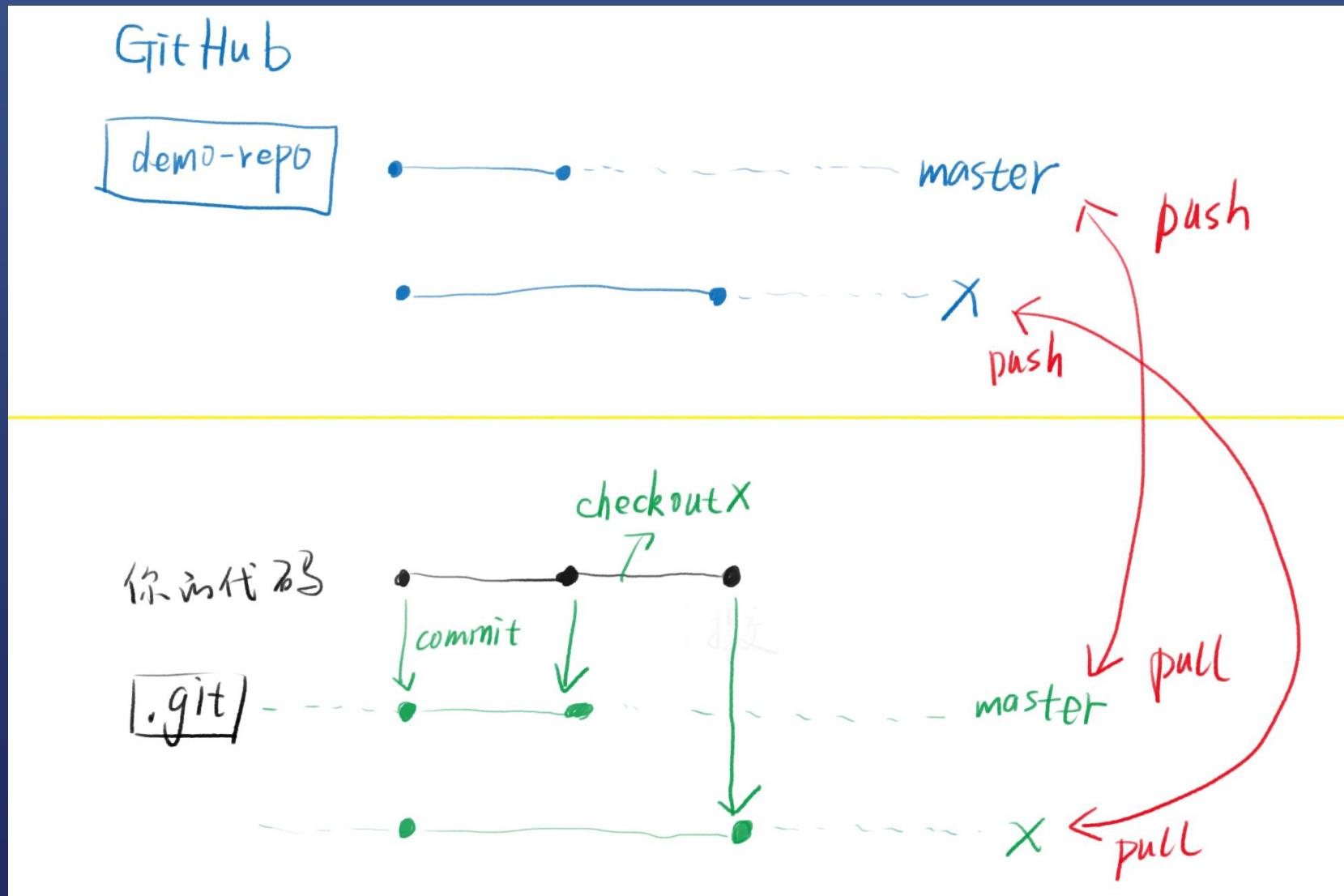
- 现在可以上传代码了
  - ✓ 新建 GitHub Repo，复制其 ssh 地址
  - ✓ 复制页面里面的代码（关掉翻译）
- `git remote add origin git@xxxxxxxxx`
  - ✓ 在本地添加远程仓库地址
  - ✓ origin 是远程仓库的默认名字，可以换，建议不要换
  - ✓ 不要使用 https:// 地址，因为每次都需要密码
- `git push -u origin master`
  - ✓ 推送本地 master 分支到远程 origin 的 master 分支
  - ✓ 如果 提示你应该 `git pull ...`，你就 `git pull` 一下
  - ✓ `git pull` 是先把远程分支合并到本地对应的分支
  - ✓ 如果远程分支没有更新过，才可以省略 `git pull`
  - ✓ `-u origin master` 的意思是设置上游分支
  - ✓ 之后就不用再设置上游分支了，直接 `git pull; git push;`

# 记住这个提示

```
Fang@DESKTOP-P0DGHMB /tmp/demo-11 (master)
$ git push -u origin master
To git@github.com:FrankFang/demo-test-1.git
  ! [rejected]      master -> master (fetch first)
error: failed to push some refs to 'git@github.com:FrankFang/demo-test-1.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

- 如果 push 的时候遇到此提示
  - ✓ 运行 git pull
  - ✓ 再重新运行之前的命令

# 原理示意图



# 如何上传其他分支

- **方法一**

- ✓ `git push origin x:x`

- **方法二**

- ✓ `git checkout x`
- ✓ `git push -u origin x`

# GitHub 用来备份 .git/ 而已

没有什么高科技原理

# 下载别人的代码

- **如何下载代码**

- ✓ git clone git@xxxxx [目标路径]
- ✓ 如果是不同机器，要写上传新的 ssh key (一机一key)
- ✓ cd 目标路径
- ✓ git add / git commit / [git pull] / git push 四连操作

- **如何下载某个分支**

- ✓ 先下载整个仓库，然后 git checkout 分支名
- ✓ 或者自己去搜一些**难记的命令**，反正我没记住

- **下载速度很慢怎么办？**

- ✓ 看我的 git clone 满速下载教程

# git clone

- **git clone git@?/xxx.git**
  - ✓ 会在当前目录下创建一个 xxx 目录
  - ✓ xxx/.git 是本地仓库
  - ✓ 一般你需要接一句 cd xxx
  - ✓ 否则我会想办法让你记住 cd xxx
- **git clone git@?/xxx.git yyy**
  - ✓ 会在本地新建 yyy 目录，记得 cd yyy
- **git clone git@?/xxx.git .**
  - ✓ 最后一个字符是点，注意有空格
  - ✓ 不会新建目录，使用当前目录容纳代码和 .git
  - ✓ 当前目录一开始最好是个空目录，不然后果自负

# 可以上传到两个远程仓库吗

- 只需要两句话

- ✓ git remote add repo2 git@xxxxxx
- ✓ git push -u repo2 master

- 如果提示 git pull...

- ✓ 说明你新建项目的时候创建了一些文件
- ✓ 你只需要运行 git pull 之后再运行刚才的命令

- 国内 GitHub 的代替品

- ✓ coding.net (腾讯战略投资)
- ✓ gitlab.com
- ✓ 码云 gitee.com (开源中国)
- ✓ 我个人只用 GitHub, 因为大牛多

# git pull 冲突了怎么办

- **解决冲突即可**
- ✓ 请回看上节课的解决冲突过程

## 解决冲突的办法

### • 发现冲突

- ✓ 你在合并分支的时候，会得到 conflict 提示
- ✓ 使用 git status -sb 查看哪个/哪些文件冲突了

### • 解决冲突

- ✓ 依次打开每个文件
- ✓ 搜索 ===== 四个等于号
- ✓ 在上下两个部分中选择要保留的代码
- ✓ 可以只选上面，也可以只选下面，甚至可以都选
- ✓ 删除不用的代码，删除 ===== >>> <<< 这些标记
- ✓ git add 对应文件
- ✓ 再次 git status -sb，解决下一个文件的冲突
- ✓ 直到没有冲突，运行 git commit (注意不需要选项)

# 总结

- 常用命令

- ✓ 大部分时候，你只需要 git clone / git pull / git push 三个命令
- ✓ 遇到报错，请仔细看报错，翻译一下，就能猜到原因

- 远程仓库

- ✓ 只是本地仓库的备份，所以变化都要先 commit 到本地仓库，然后 push 到远程
- ✓ 无法下载部分代码，只能 clone 整个仓库

# git 高级操作

只有熟练的老手知道的操作

# 使用 bash alias 简化命令

gst / ga / gc / gl / gp

# 运行以下命令即可

```
touch ~/.bashrc
echo 'alias ga="git add"'>> ~/.bashrc
echo 'alias gc="git commit -v"'>> ~/.bashrc
echo 'alias gl="git pull"'>> ~/.bashrc
echo 'alias gp="git push"'>> ~/.bashrc
echo 'alias gco="git checkout"'>> ~/.bashrc
echo 'alias gst="git status -sb"'>> ~/.bashrc
```

然后重启命令行，或者运行 source ~/.bashrc  
你就可以用这些缩写了

# 好看的 glog

在 `~/.bashrc` 的最后一行添加

```
alias glog="git log --graph --  
pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s  
%Cgreen(%cr) %C(bold blue)<%an>%Creset' --abbrev-commit  
-- | less"
```

```
* 4dbe559 - (HEAD -> master) 内容 (2 days ago) <frankfang>  
* a8746de - (origin/master) 解决 TS 类型错误 (2 weeks ago) <frankfang>  
* 6b65ec4 - 解决异步 bug (9 weeks ago) <frankfang>  
* c591bfd - update README (9 weeks ago) <frankfang>  
* 9d6f164 - 代码优化、添加 todo (3 months ago) <frankfang>  
* e778b8f - 完成异步验证 (3 months ago) <frankfang>  
* 5a33e90 - 完善样式 (3 months ago) <frankfang>  
* 56bb35f - 还没完成 (4 months ago) <frankfang>  
* 57a05aa - update README (4 months ago) <frankfang>  
* ce37fbf - update README (4 months ago) <frankfang>  
* ec23e48 - update README (4 months ago) <frankfang>  
* a771223 - fix doc.sh (4 months ago) <frankfang>  
* cf5ddfd - rename example.html (4 months ago) <frankfang>  
* edb023b - add doc.sh (4 months ago) <frankfang>  
* f2208ad - 新增 yarn doc 配置 (4 months ago) <frankfang>  
* 9e6db2a - 优化 example 页面 (4 months ago) <frankfang>  
* 5429949 - 优化 Demo 展示 (4 months ago) <frankfang>  
* 6e10a0f - 添加 Demo 组件 (4 months ago) <frankfang>  
* 4e8d16d - 完成 example 页面的布局 (4 months ago) <frankfang>  
* f62591f - 完成 Layout 各个组件 (4 months ago) <frankfang>  
* 1e63d1f - 重构 scopedClassMaker x2 (4 months ago) <frankfang>  
* 871e97a - 重构 scopedClassMaker (4 months ago) <frankfang>  
* 5668b33 - 完成 dialog 的三个便捷 API (4 months ago) <frankfang>  
:
```

git rebase -i XXXX  
美化历史命令

# git rebase -i XXXX 示例

```
| pick 6b65ec4 解决异步bug
| pick a8746de 解决 TS 类型错误
|
# Rebase c591bfd..a8746de onto c591bfd (2 command(s))
#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line as arguments)
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out
```

**r 采用，但是改写 message**

**S 采用，但是合并到上一个提交**

# 出错怎么办

- 看 log 提示

- ✓ 仔细阅读 git 给出的 log，里面说了怎么解决

- 中止

- ✓ git rebase --abort 可以取消 rebase

- 继续

- ✓ git rebase --continue 可以继续

# 通灵术

git stash / git stash pop

你不想提交代码，又不想删除代码

那么就可以找个空间把代码临时藏起来

有个印象就行  
工作的时候你自然就会用到了

# 再见

下节课学习如何玩转 GitHub