

我们编程吧 之 git 学习手册

Version 0.4 \$at\$ 2016.01.30

[TOC]

Git 简介

Git是一个开源的软件，作者是大名鼎鼎的Linus Torvalds，写出Linux的那个哥们。

Git是一个专门用于做修改记录的程序，也被叫做版本控制软件。

安装Git

这个很简单。

- Windows用户，参考[github@windows](#)
- Mac用户，参考[github@MacOSX](#) Git不像别的软件，安装完成后不会在桌面上有个Logo什么的，直接终端操作

设置Git

查看git版本

在终端中输入`git --version`，比如我的输出为：

“

`git version 2.7.0`

下面的设置主要是告诉git是哪位大神在用git。

设定你的名字

`git config --global user.name "<yourname>"`

设定你的邮箱

`git config --global user.email "<youremail@example.com>"`

新建repo

repo

repo可以理解为一个被跟踪的文件夹，其中的任何内容做的任何更改都会被记录在册。

新建一个repo

比如我们的文件夹就叫做`hello-world`吧。执行下列指令：

- 新建文件夹`mkdir hello-world`
- 进入文件夹`cd hello-world`
- 初始化`git init`

看看初始化成功没有，输入指令`git status`，只要不显示`fatal: Not a git repository...`，那就OK了。

恭喜了，第一步成功了。

新增提交更改

新建一个文件

假设现在在刚才建立的文件`hello-world`中，如果没有就妥妥滴进来。

新建一个文件，比如`vim readme`，然后输入一些简单的内容，比如

“

`hello git.`

查看git状态

这里我们可以检查目前repo的各种状态，比如是否有过任何修改，命令为：

“

```
git status
```

此时的输出应该是类似下面这个样子:

```
➔ learn-git git:(master) ✘ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    README

nothing added to commit but untracked files present (use "git add" to track)
```

新增文件

此时我们就可以将readme文件添加到档案库里面去了。

“

```
git add README
```

小技巧如果当前文件较多，可以是用`git add .`来一股脑地把所有的文件都给新增了。

此时的输出应该是这个样子:

```
➔ learn-git git:(master) ✘ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   README
    new file:   git-status.png
git add README
➔ learn-git git:(master) ✘
```

提交文件

“

```
git commit -m "Adding a README file"
```

此时的输出应该是如下这个样子:

```
➔ learn-git git:(master) git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)
nothing to commit, working directory clean
```

修改文件查看移动

在readme文件中新增一行，此时是用命令:

“”

git diff

可以看到如下输出结果：

```
diff --git a/readme b/readme  Go
index 8d0e412..f4aced2 100644
--- a/readme shaoguangleo (shaogua
+++ b/readme
@@ -1 +1,2 @@
 hello git
+add one more line
(END)
```

注册github账号

还在本地

其实到目前为止，你的代码还滞留在你自己的机器上，别人无法看到（黑客除外）。

此时我们就需要注册一个github账号，把自己优雅的代码共享给全世界。

创建github账号

github是一个让所有人可以轻松上传自己代码，并协作的版本控制网站。移步[github](#)。

注册完后，此时还记得如何设置自己的用户名吗，参考[设置git](#)。

连接本机与远端repo

远端repo

当你在github上创建了一个repo后，所有同步到该repo的代码都会保存到github的主机服务器上。此时的这个repo就被称为remote repository，你每次把自己的更改push到这个repo，就叫做同步sync。

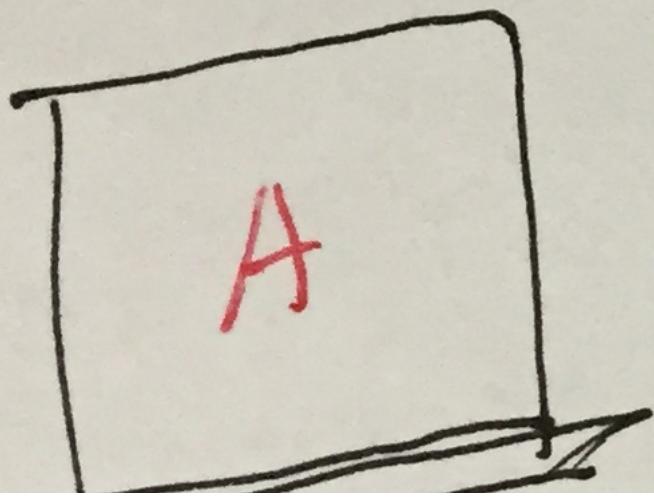
其他人就可以通过pull你的这个remote repository来获取到你最新push的版本，存储到他们自己的电脑上。这样一来大家就可以共同的push、pull来协作修改同一个文件，而不用拿着U盘四处拷贝 省去病毒的传播，想当年去打印店的兄弟们...的U盘，哪个不是病毒缠身。

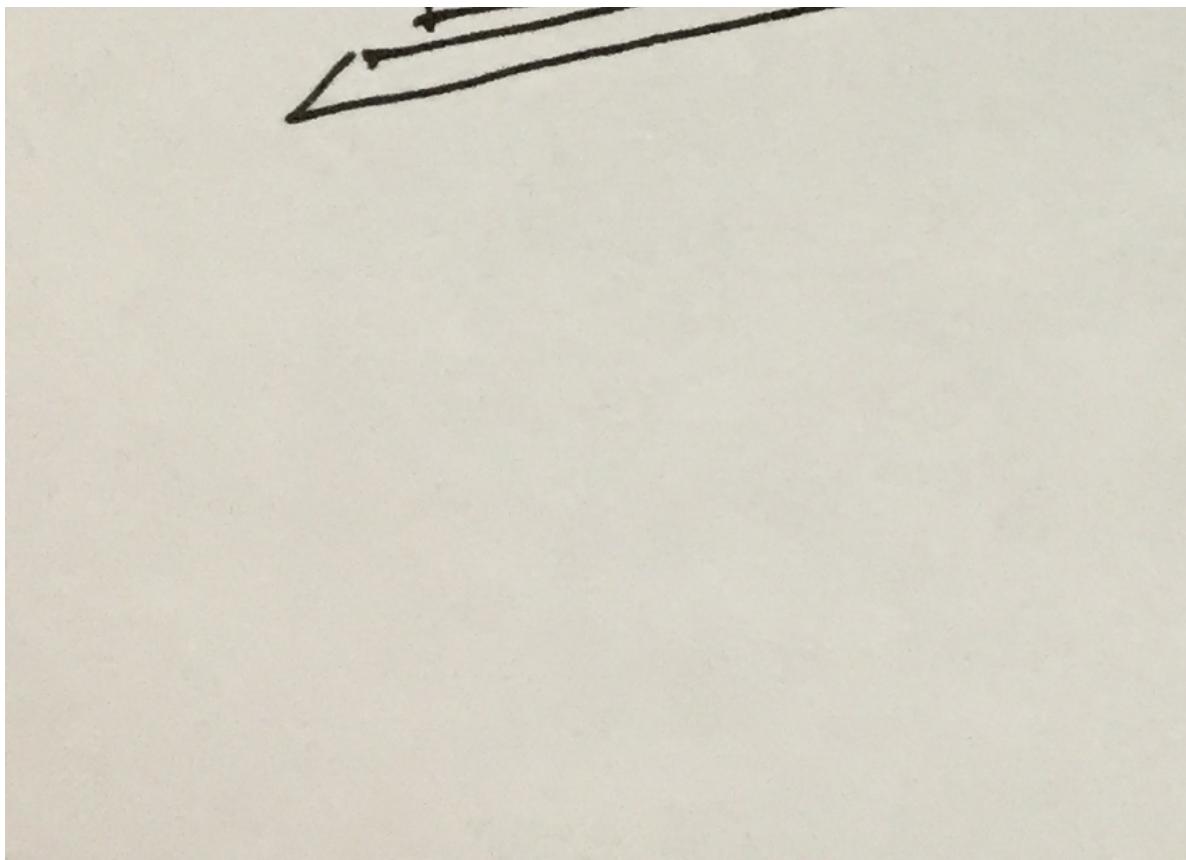
push、pull的示意图如下所示：



pus'

卡布斯





新建一个remote repository

此时我们准备将电脑上的版本和存在[github](#)的版本做同步，我们首先在github上新建一个repo。

- 登陆[github](#)，点击右上角的+号来新增一个repo
- 取一个名字比如hello-world，并给个简短的说明
- 设定为public
- 不要勾选initialize with a README，因为我们是从本机创建的git
- 暂时不需要修改.gitignore和license的设置
- 创建repo

新建的地方在这里：

The screenshot shows the GitHub homepage. At the top, there is a search bar labeled "Search GitHub" and navigation links for "Pull requests", "Issues", and "Gist". Below the search bar, there is a large blue banner with the text "Learn Git and GitHub without any code!". Underneath the banner, a message reads: "Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request." A red rectangular box highlights the "N" link in the bottom right corner of the banner area.

设置的东西在这里

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: shaoguangleo / Repository name: hello-world ✓

Great repository names are short and memorable. Need inspiration? How about [turbo-barnacle](#).

Description (optional): Learning git

Public
Anyone can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None ⓘ

Create repository

README、.gitignore和License

一般在开源的软件中，都会看到这几个文件，含义如下：

- README：通常用来解释一个程序的功能、安装及使用方法及一些贡献者等等
- .gitignore：这个主要是告诉git在更新文件的时候忽略掉那些不用跟踪的文件
- License：用来声明一个程序可以或不可以被怎样使用

连接本地的repo和remote的repo

先在我们已经在GitHub上新建了一个repository，在github的官网的右上角可以看到如下图所示的一个地址，我们这里选择https。

Will teach you how to learn git — Edit

13 commits 1 branch 0 releases 1 commit

Branch: master New pull request New file Find file

Choose a clone URL

- ✓ **HTTPS (recommended)**
Clone with Git or checkout with SVN using the repository's web address.
- SSH**
Clone with an SSH key and passphrase from your GitHub settings.

③ Learn more about clone URLs

File	Description
README.md	README
git-add.png	Adding a file
git-commit.png	Adding a commit
git-diff.png	Adding a diff
git-status.png	Add git status
learn-git-v0.1.pdf	Adding the 0.1 version
readme	A test file
wechat-letsProgramming-small.jpg	Adding wechat qrcode
wechat-letsProgramming.jpg	Adding wechat qrcode
README.md	

然后进入hello-world文件夹

```
$ git remote add origin <your-git-repo-website>
```

现在我们的电脑本地repo就知道了在github上有一个remote repo, 叫做origin, 当然这个名字是可以修改的。此时origin代表的就是长长的那个网址。

推送push修改到github上

接下来我们来推送一些本地的修改到github上的repo，因为我们总是希望本地的和github上的内容保持同步。

在推送之前，说下其实git有一个**branch**系统，可以修改一个程序不同的功能，后面会有介绍。通常系统会给我们的**branch**一个最初的名字**master**。

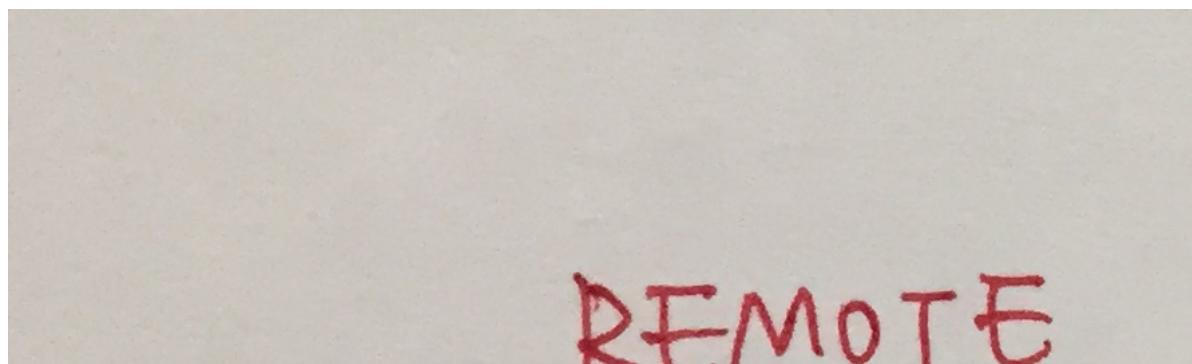
在我们需要推送的时候，需要告诉git准备推送哪一个分支的进度。

此时我们把我们的主分支**master**推送过去。

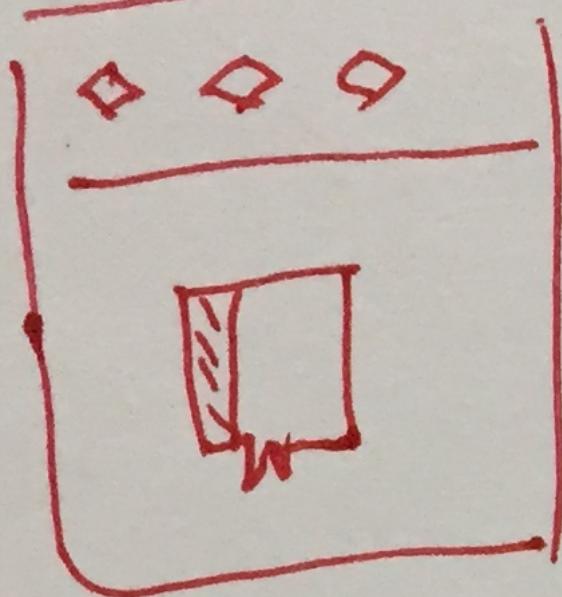
```
$ git push origin master
```

完成后，直接登录github的hello-world的repo页面，看看你创建的第一个repo吧。

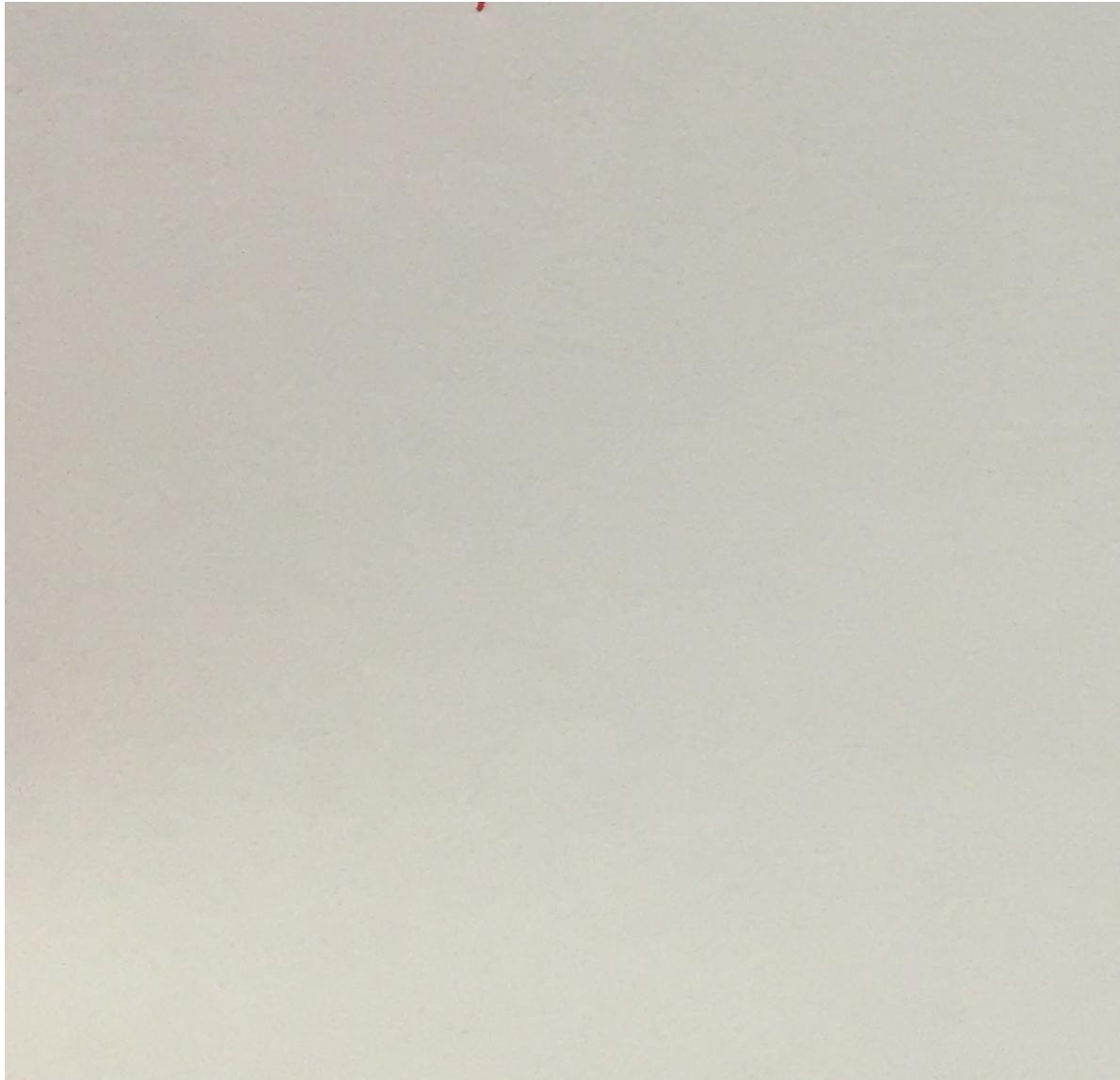
fork与clone



其他人



通过 'upstream'
来 push/pull



forks

在上一节里面我们已经push了修改到github上，但如果只能这样，git未免太小儿科了，因为git最有趣的地方其实在于与不同的人来协作开发。

当你fork一个程序的repo时，其实是复制了一份软件的repo到你自己的github账号下面。



然后我们就可以在这个基础上来做一些开发了，比如新增一些功能、修正一些错误。

当我们成功fork一个repo后，我们就可以将它从github下载clone到我们自己的电脑上，这样在没有网络的情况下，我们也可以修改软件的内容。

下载Clone我们刚刚fork的repo

现在我们就可以将刚刚fork的repo下载到本地了，方法很简单：

```
$ git clone https://github.com/yourname/learn-git
```

，但是要比较留意，千万不要在已经是一个repo的文件夹里面来clone。

比如现在我们可能还在文件夹hello-world中，那么使用 cd ..，到上一级目录，然后在执行clone的命令即可。

完成后，直接使用 `cd learn-git` 进入到刚刚下载的软件目录里面去。

此时我们就拥有了一个这个软件的副本，并自动将其链接到了你github账号下面的远程repo中。

链接到原始的repo中



到这里就有一个问题了，我们是 **fork** 了别人的repo，但是如果别人的repo更新了，我们fork的repo是无法同步更新的。

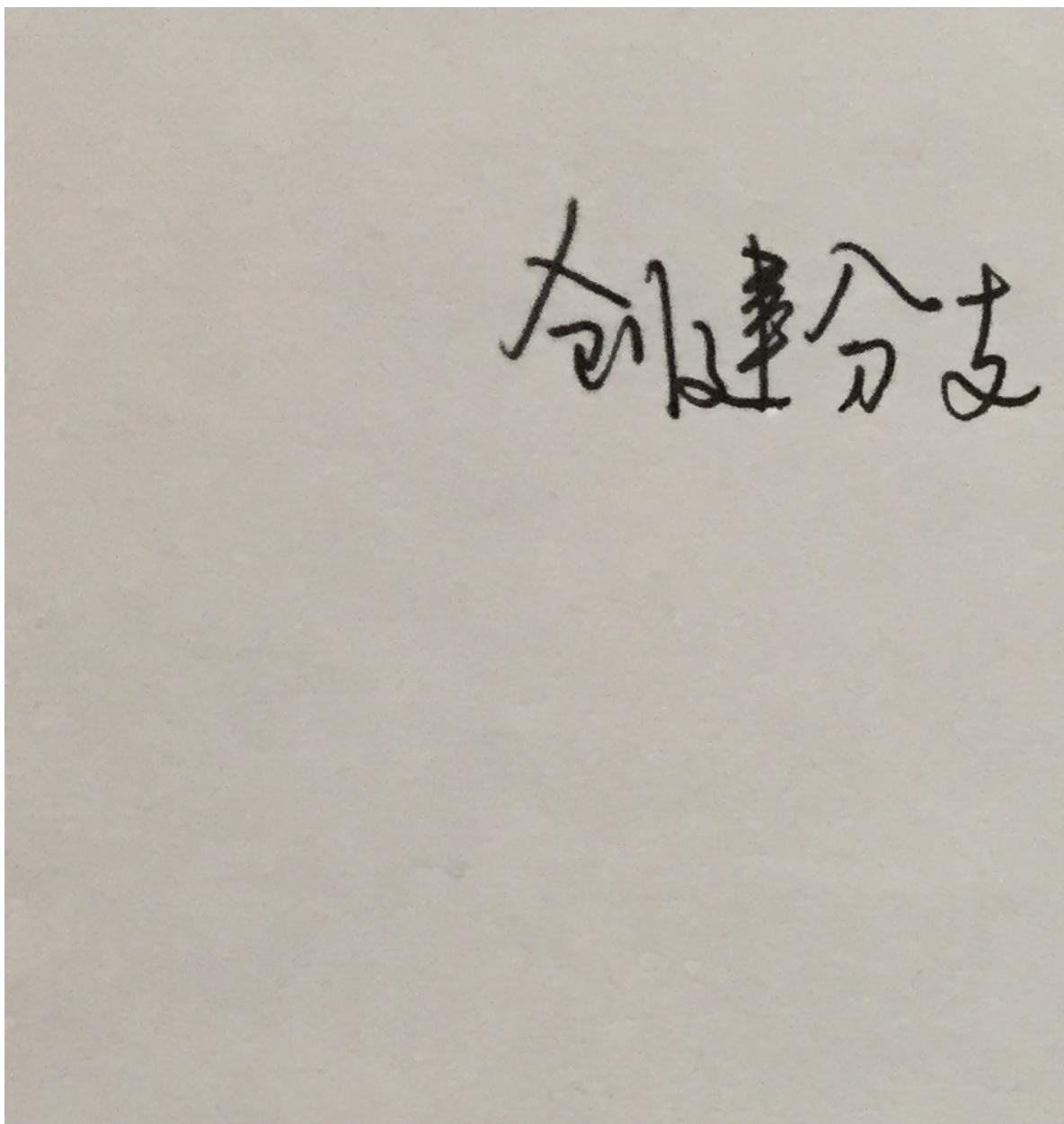
而我们还是希望更新的，那么这个时候我们就需要新增另外一个remote来链接到原始的repo中，比如原始的repo位于 `github.com/shaoguangleo/learn-git` 上，该repo的地址可以在右上角找到。

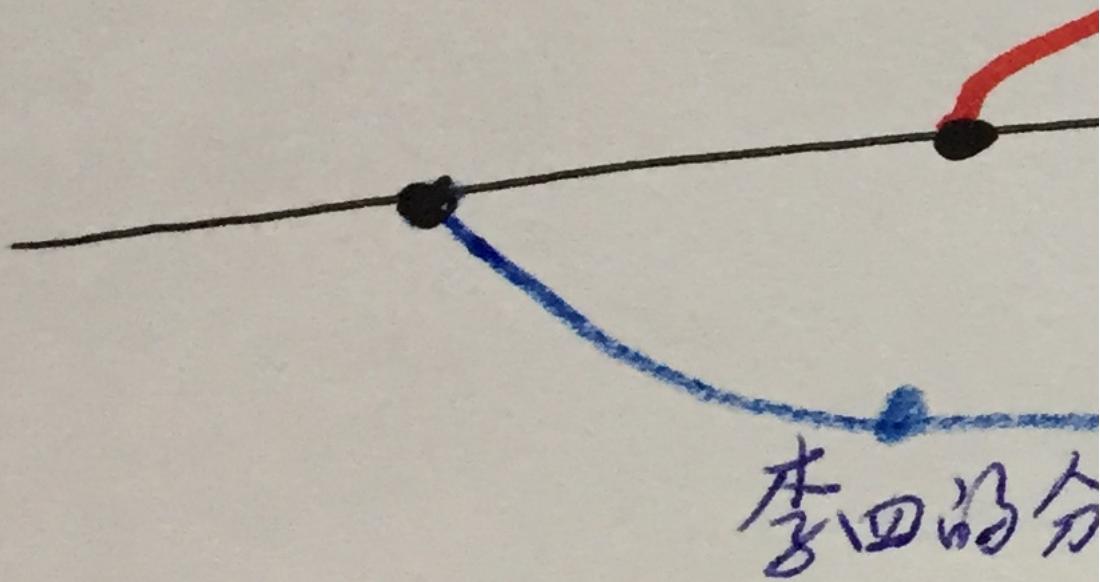
先在我们起个名字，比如大家常用的upstream，执行下面的命令：`$ git remote add upstream https://github.com/shaoguangleo/learn-git.git` 此时我们就通过upstream与原始的repo建立了链接。

新建一个特性分支

分支

git的软件repo使用分支**branches**来隔离开开发进度。

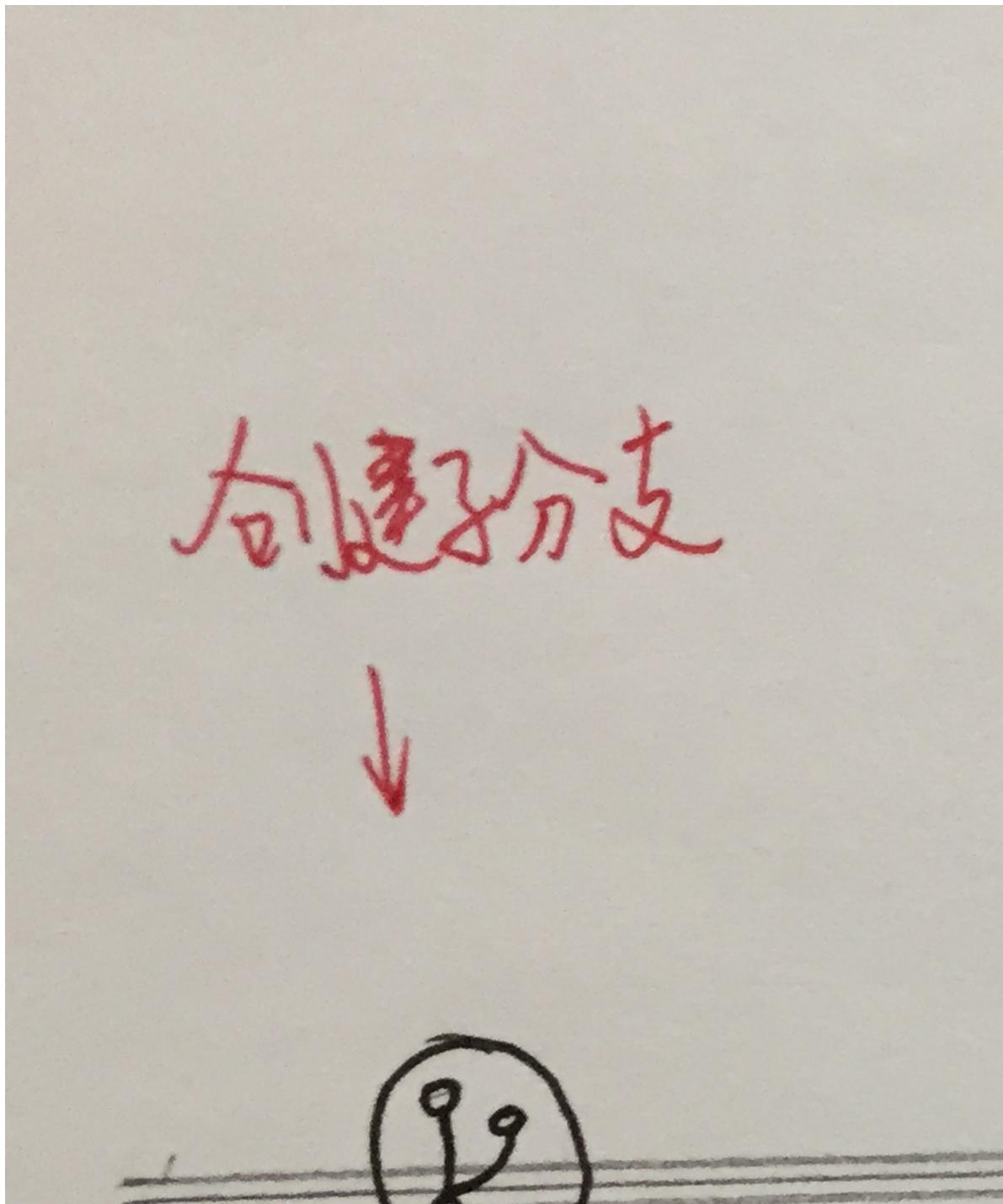


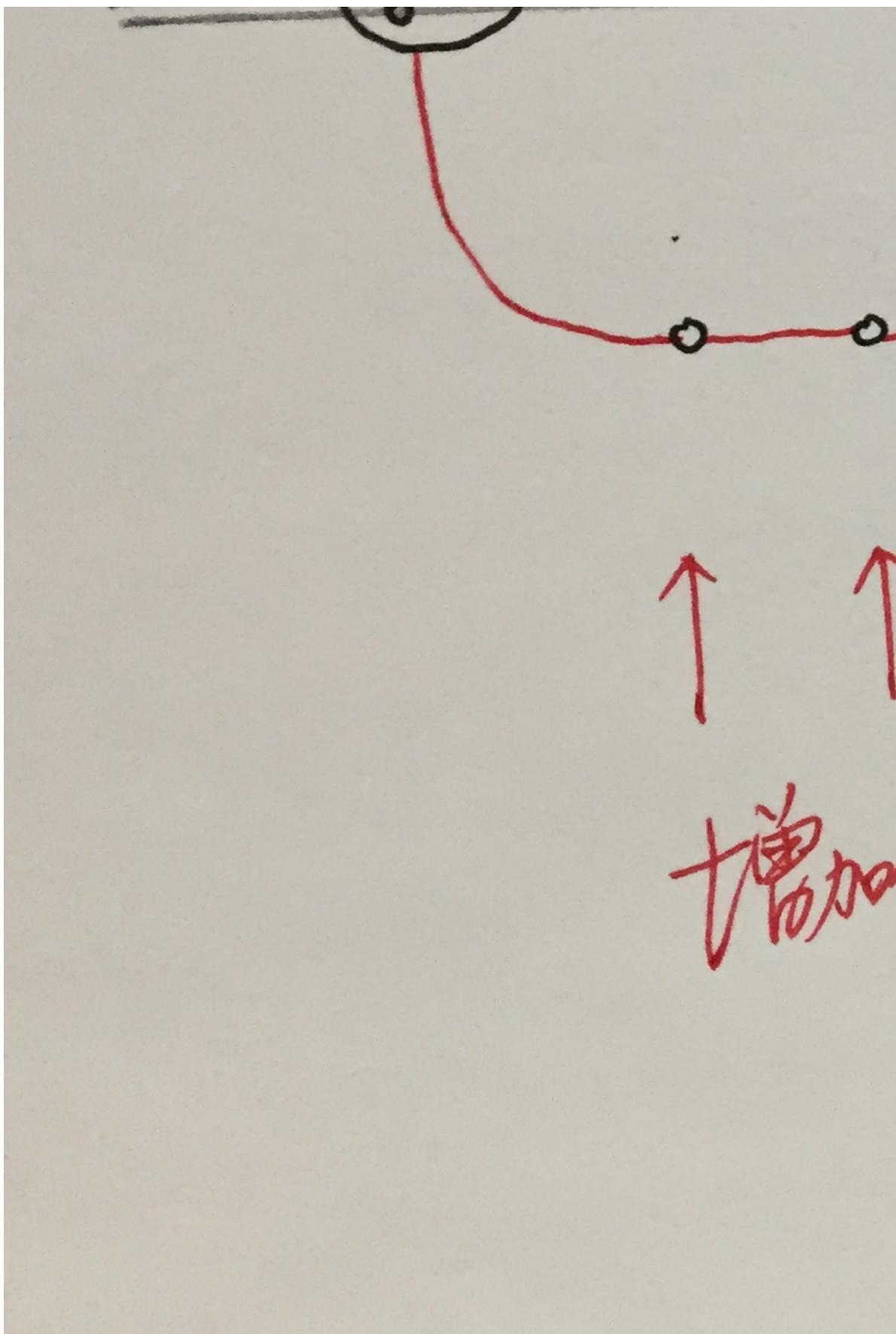


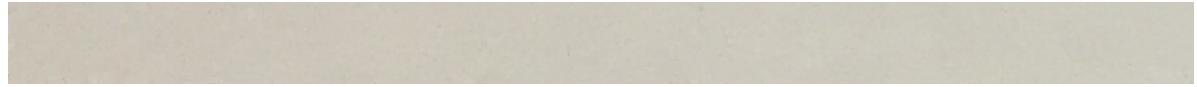
李四的分



当我们需要与其他人一起共同开发的时候，在完成自己负责的那一部分程序前，最好是创建自己的分支，这样就可以让主分支**master**保持稳定，不被未修改的修改影响。当你完成了自己的那一部分，就可以使用**merge**将分支修改合并进入**master**主分支中。







与github同步

PR

更多信息

Hi, XDJM们, 更多信息欢迎移步[我的github](#)或微信公众号letsProgramming.

