Git

凯盛软件

相关资料

凯盛软件

http://git.oschina.net/progit/

http://www.bootcss.com/p/git-guide/

https://www.atlassian.com/git/

http://www.liaoxuefeng.com/wiki/001373951630592960 6dd18361248578c67b8067c8c017b000

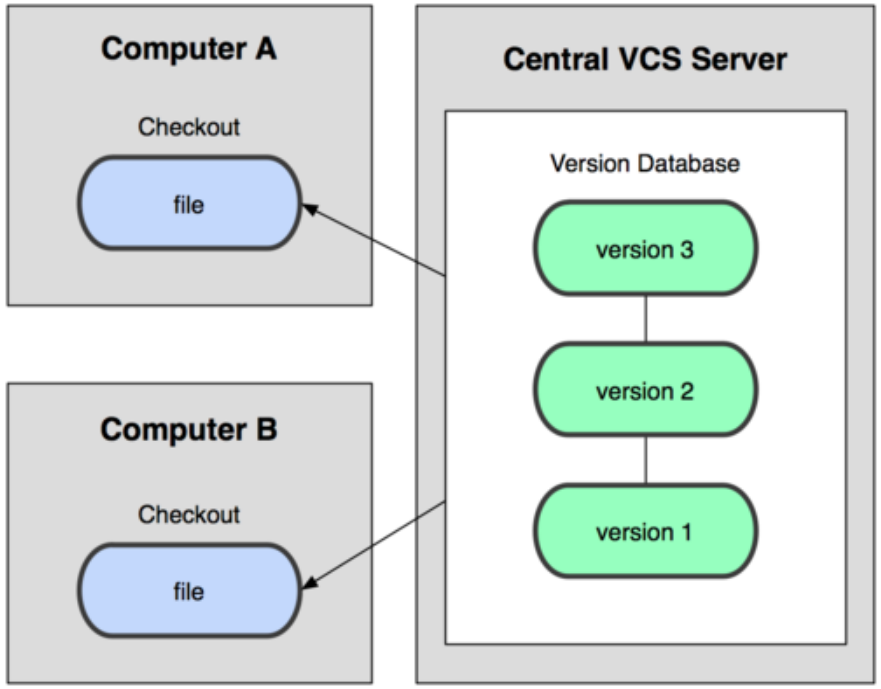
http://www.ruanyifeng.com/blog/2015/12/git-cheat-sheet.html

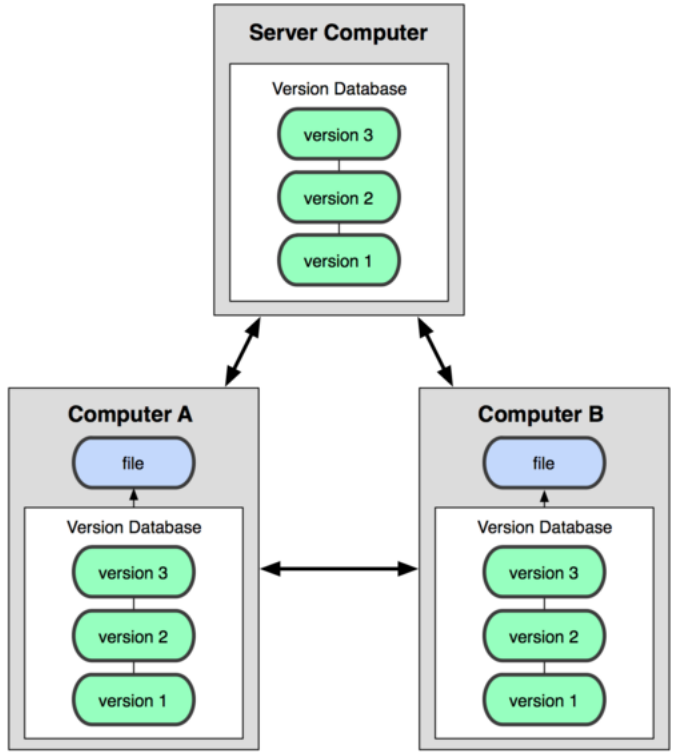
http://www.ruanyifeng.com/blog/2015/12/git-workflow.html

Git

凯盛软件

git是一个分布式版本控制软件，最初由林纳斯·托瓦兹（Linus Torvalds）创作，于2005年以GPL发布。最初目的是为更好地管理Linux内核开发而设计。






git
github
SOCIAL CODING

文件的状态

- 已修改(modifed)

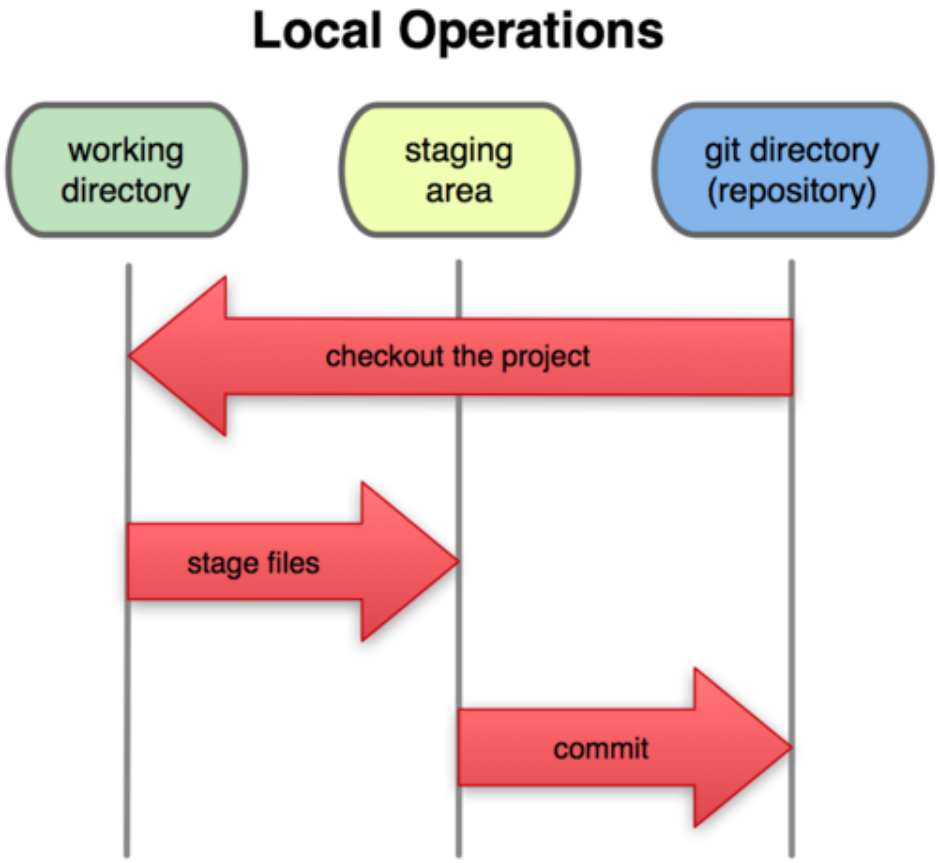
  表示修改了某个文件，但还没有提交保存

- 已暂存(staged)

  表示把已修改的文件放在下次提交时要保存的清单中

- 已提交(commited)

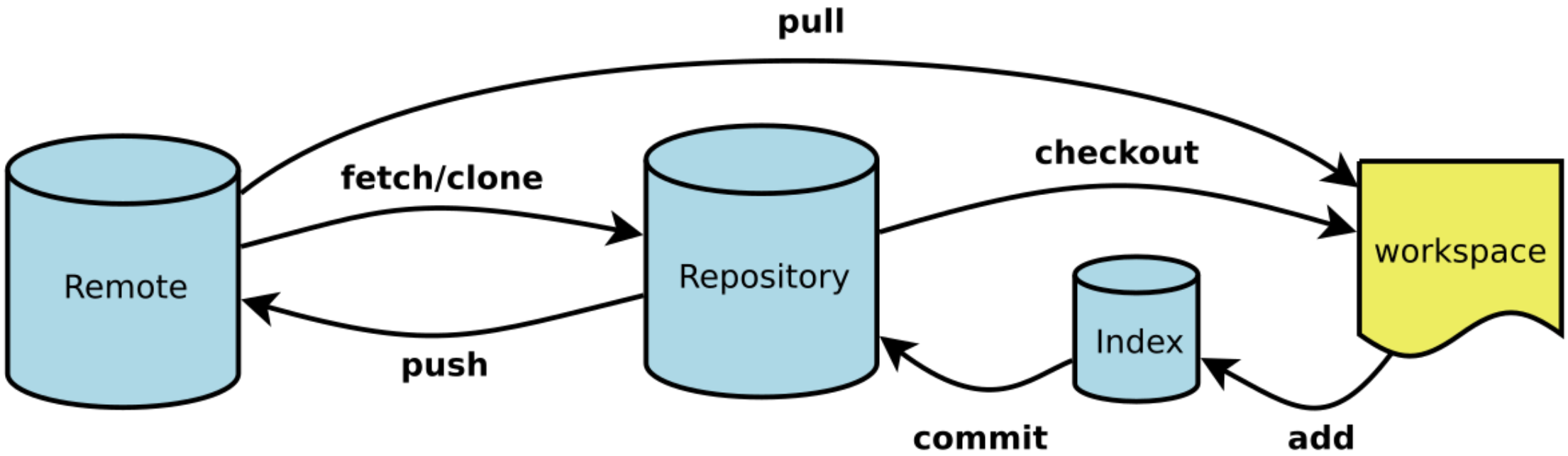
  表示该文件已经被安全地保存在本地数据库中了



Git流程

1. 在工作目录中修改某些文件。
2. 对修改后的文件进行快照，然后保存到暂存区域。
3. 提交更新，将保存在暂存区域的文件快照永久转储到 Git 目录中。

Git流程


pull
fetch/clone
checkout
Remote
Repository
workspace
Index
push
commit
add

基本配置

凯盛软件

全局配置

git config --global user.name fankay

git config --global user.email fankai@kaishengit.com

某个项目中配置

git config user.name fankay

git config user.email fankai@kaishengit.com

初始化

git init

该命令会在当前目录中生成 .git 的文件夹，该文件夹是git的数据库

工作目录文件状态

- 已跟踪
- 未跟踪

已跟踪的文件是指本来就被纳入版本控制管理的文件，在上次快照中有它们的记录，工作一段时间后，它们的状态可能是未更新，已修改或者已放入暂存区。而所有其他文件都属于未跟踪文件。它们既没有上次更新时的快照，也不在当前的暂存区域。初次克隆某个仓库时，工作目录中的所有文件都属于已跟踪文件，且状态为未修改。

检测文件状态

git status

将工作区文件放入暂存区

凯盛软件

#添加文件到暂存区

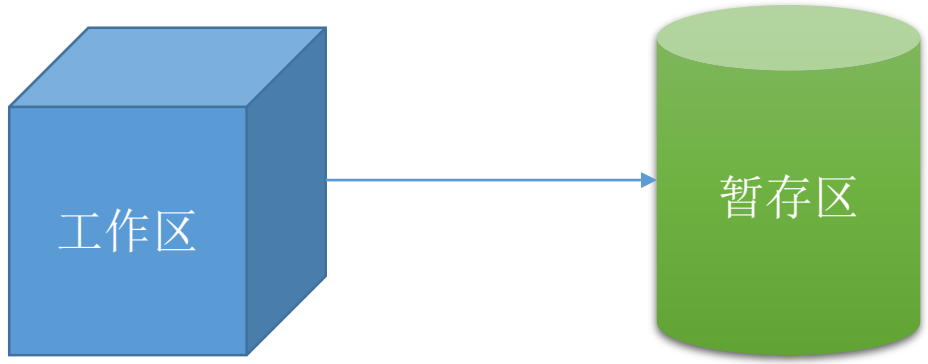
git add 文件

#添加指定的目录到暂存区，包括子目录

git add 文件夹

#添加当前目录的所有文件到暂存区

git add .



```
➜  demo2 git:(master) touch hello.txt
➜  demo2 git:(master) ✗ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        hello.txt

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

新文件，未跟踪状态

```
➜  demo2 git:(master) ✗ git add hello.txt
➜  demo2 git:(master) ✗ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   hello.txt
```

放入暂存区，已跟踪状态

修改已跟踪的文件

凯盛软件

```
➜  demo2 git:(master) ✗ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   hello.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   hello.txt
```

文件已修改，但是新修改未提交到暂存区

查看文件更新了哪些部分

凯盛软件

- 查看尚未暂存的文件更新了哪些部分

git diff

1. 将文件放入暂存区
2. 修改文件
3. 执行git diff 命令

当前命令比较的是工作目录中当前文件和暂存区域快照之间的差异，也就是修改之后还没有暂存起来的变化内容

```
diff --git a/hello.txt b/hello.txt
index 15b8f2a..1463283 100644
--- a/hello.txt
+++ b/hello.txt
@@ -1 +1,2 @@
 hello1
+hellooooooo
```

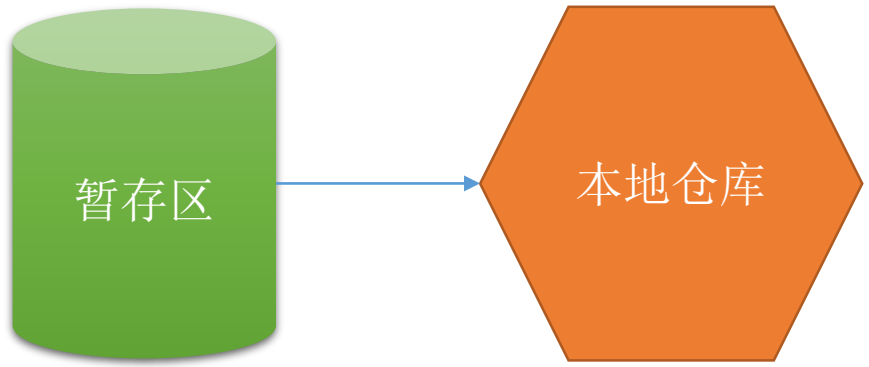
提交文件到本地仓库

凯盛软件

git commit -m “提交日志”

提交时记录的是放在暂存区域的快照，任何还未暂存的仍然保持已修改状态，可以在下次提交时纳入版本管理。每一次运行提交操作，都是对你项目作一次快照，以后可以回到这个状态，或者进行比较。

```
➜  demo2 git:(master) ✗ git commit -m "修复了某些Bug"
[master ad26d03] 修复了某些Bug
 1 file changed, 1 insertion(+)
```



删除文件

凯盛软件

#删除工作区文件，并且将这次删除放入暂存区

git rm 文件

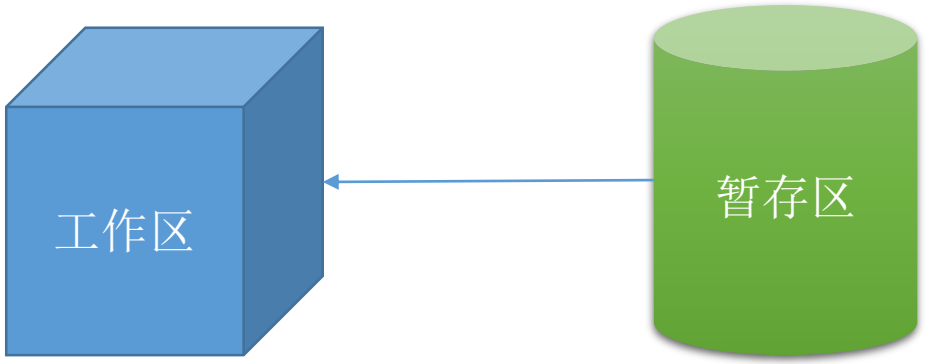
```
➜  demo2 git:(master) rm sec.txt
➜  demo2 git:(master) ✗ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    sec.txt

no changes added to commit (use "git add" and/or "git commit -a")
➜  demo2 git:(master) ✗ git rm sec.txt
rm 'sec.txt'
➜  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        deleted:    sec.txt

➜  demo2 git:(master) ✗ git commit -m "删除了sec.txt"
[master 7654f31] 删除了sec.txt
 1 file changed, 1 deletion(-)
 delete mode 100644 sec.txt
```

删除文件

#停止追踪指定文件，但该文件会保留在工作区

git rm --cached 文件

```
→  demo2 git:(master) ✗ git add hello.txt
→  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   hello.txt

→  demo2 git:(master) ✗ git rm --cached hello.txt
rm 'hello.txt'
→  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        deleted:    hello.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        hello.txt

→  demo2 git:(master) ✗
```



文件重命名

凯盛软件

git mv 旧文件名 新文件名

不像其他的 VCS 系统，Git 并不跟踪文件移动操作。如果在 Git 中重命名了某个文件，仓库中存储的元数据并不会体现出这是一次改名操作。

```
➜  demo2 git:(master) git mv hello.txt hi.txt
➜  demo2 git:(master) ✗ ls
hi.txt
➜  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        renamed:    hello.txt -> hi.txt

➜  demo2 git:(master) ✗ git commit -m "重命名文件"
[master e02ab3d] 重命名文件
 1 file changed, 0 insertions(+), 0 deletions(-)
 rename hello.txt => hi.txt (100%)
➜  demo2 git:(master)
```

查看文件提交历史

凯盛软件

git log

```
commit e02ab3d8edf1dcae5015f24463bdc76277d6d450
Author: fankay <fankai@kaishengit.com>
Date:   Mon Feb 29 21:36:59 2016 +0800

    重命名文件

commit fc9c3f4599d26d6b95212321be6f2f2eaa3a696b
Author: fankay <fankai@kaishengit.com>
Date:   Mon Feb 29 21:34:49 2016 +0800

    又一次提交了
```

#将提交信息在一行显示

git log --pretty=oneline

```
e02ab3d8edf1dcae5015f24463bdc76277d6d450 重命名文件
fc9c3f4599d26d6b95212321be6f2f2eaa3a696b 又一次提交了
7654f31e19dff058047c34312447d65ec4c0cb0d 删除了sec.txt
```

撤销操作

凯盛软件

- 修改最后一次提交

#使用一次新的commit，替代上一次提交

如果代码没有任何新变化，则用来改写上一次commit的提交信息

git commit --amend -m “提交信息”

```
➜  demo2 git:(master) ✗ git add hi.txt
➜  demo2 git:(master) ✗ git commit -m "添加新的数据"
[master e087ecf] 添加新的数据
 1 file changed, 1 insertion(+)
➜  demo2 git:(master) touch new.txt
➜  demo2 git:(master) ✗ git add new.txt
➜  demo2 git:(master) ✗ git commit --amend -m "修改后的添加新数据"
[master c0cd912] 修改后的添加新数据
 Date: Mon Feb 29 21:49:21 2016 +0800
 2 files changed, 1 insertion(+)
 create mode 100644 new.txt
```

- 取消已经暂存的文件

git reset HEAD 文件名

```
➜  demo2 git:(master) ✗ git add .
➜  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   hi.txt
        modified:   new.txt

➜  demo2 git:(master) ✗ git reset HEAD new.txt
Unstaged changes after reset:
M       new.txt
➜  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   hi.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   new.txt
```

git rm命名操作后文件处于未跟踪状态

git reset命令是撤销了刚提交到暂存区的数据（处于已修改状态）

- 取消对文件的修改

git checkout -- 文件名

```
➜  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   hi.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   new.txt

➜  demo2 git:(master) ✗ git checkout -- new.txt
➜  demo2 git:(master) ✗ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   hi.txt
```

恢复代码到提交的某个版本

凯盛软件

1. 使用log命令获取版本号

```
4e63def653cc5abe3036e9aad3caca7d285f18b7 修改了bug
c0cd91218c0931e7bbc03daa81d3ff92483c0ba8 修改后的添加新数据
38b9eece419d1e60ea215f8a6985a1ae6ff6d7d9 重命名文件
```

2. 回退到指定版本号

   git reset --hard 版本号

```
➜  demo2 git:(master) git log --pretty=oneline
➜  demo2 git:(master) git reset --hard c0cd912
HEAD is now at c0cd912 修改后的添加新数据
```

重新回到最新版本

1. 查看最新版本的版本号

```
c0cd912 HEAD@{0}: reset: moving to c0cd912
4e63def HEAD@{1}: reset: moving to 4e63def
c0cd912 HEAD@{2}: reset: moving to c0cd912
4e63def HEAD@{3}: commit: 修改了bug
c0cd912 HEAD@{4}: commit (amend): 修改后的添加新数据
e087ecf HEAD@{5}: commit: 添加新的数据
```

2. 回退

```
➜  demo2 git:(master) git reflog
➜  demo2 git:(master) git reset --hard 4e63def
HEAD is now at 4e63def 修改了bug
```

远程仓库

凯盛软件

- 国外
  - Github（https://www.github.com）
  - Bitbucket （https://www.bitbucket.org）
- 国内
  - 码云（http://git.oschina.net/）
  - Coding（https://coding.net/）

- 私服
  - Gogs（https://gogs.io/）
  - GitLab（https://about.gitlab.com/）

GitHub

https://www.github.com

1. 注册账号
2. 生成SSH公钥

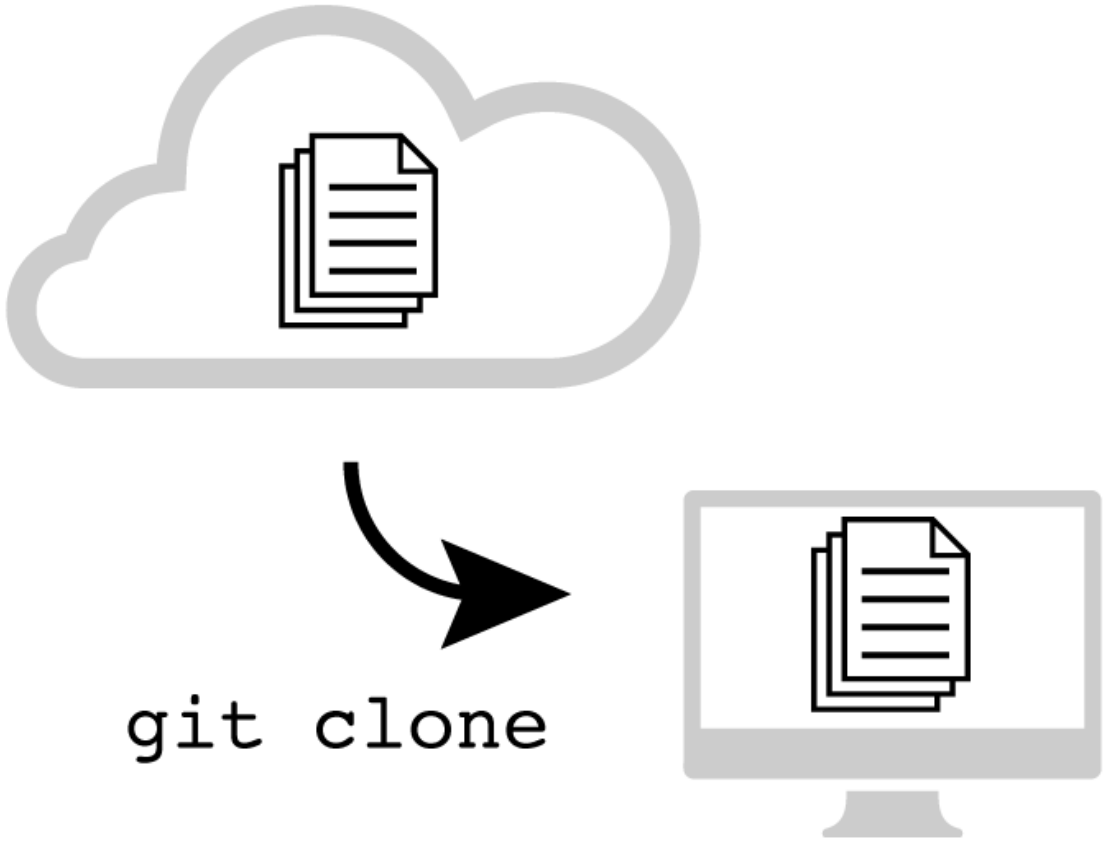
   ssh-keygen -t rsa -C “电子邮件地址”

3. 添加SSH keys(https://github.com/settings/ssh)

从远程仓库中获取数据

凯盛软件

git clone url



添加远程仓库

git remote add 缩略名 地址

```
➜  demo2 git:(master) git remote add origin git@github.com:fankay/gitdemo.git
➜  demo2 git:(master) git remote
origin
```

将本地仓库推送到远程服务器

凯盛软件

git push 缩略名 分支名称

```
➜  demo2 git:(master) git push -u origin master
Counting objects: 24, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (13/13), done.
Writing objects: 100% (24/24), 2.03 KiB | 0 bytes/s, done.
Total 24 (delta 1), reused 0 (delta 0)
To git@github.com:fankay/gitdemo.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
➜  demo2 git:(master)
```

远程仓库的重命名和删除

凯盛软件

git remote rename 旧缩略名 新缩略名

```
➜  demo2 git:(master) git remote rename origin repo
➜  demo2 git:(master) git remote
repo
```

git remote rm 缩略名

```
➜  demo2 git:(master) git remote rm repo
➜  demo2 git:(master) git remote
```

获取远程修改

凯盛软件

git fetch 缩略名

```
➜  demo2 git:(master) git fetch origin
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:fankay/gitdemo
   b68145c..567bda5  master     -> origin/master
```

fetch操作后，文件内容并不会和远程仓库进行同步，还需要进行merge操作

git merge

```
➜  demo2 git:(master) git merge
Updating b68145c..567bda5
Fast-forward
 hi.txt | 1 +
 1 file changed, 1 insertion(+)
```

git pull 缩略名 分支名称

pull = fetch + merge

```
➜  demo2 git:(master) git pull origin master
From github.com:fankay/gitdemo
 * branch            master     -> FETCH_HEAD
Updating 4e63def..b68145c
Fast-forward
 hi.txt | 1 +
 1 file changed, 1 insertion(+)
```

解决push冲突

凯盛软件

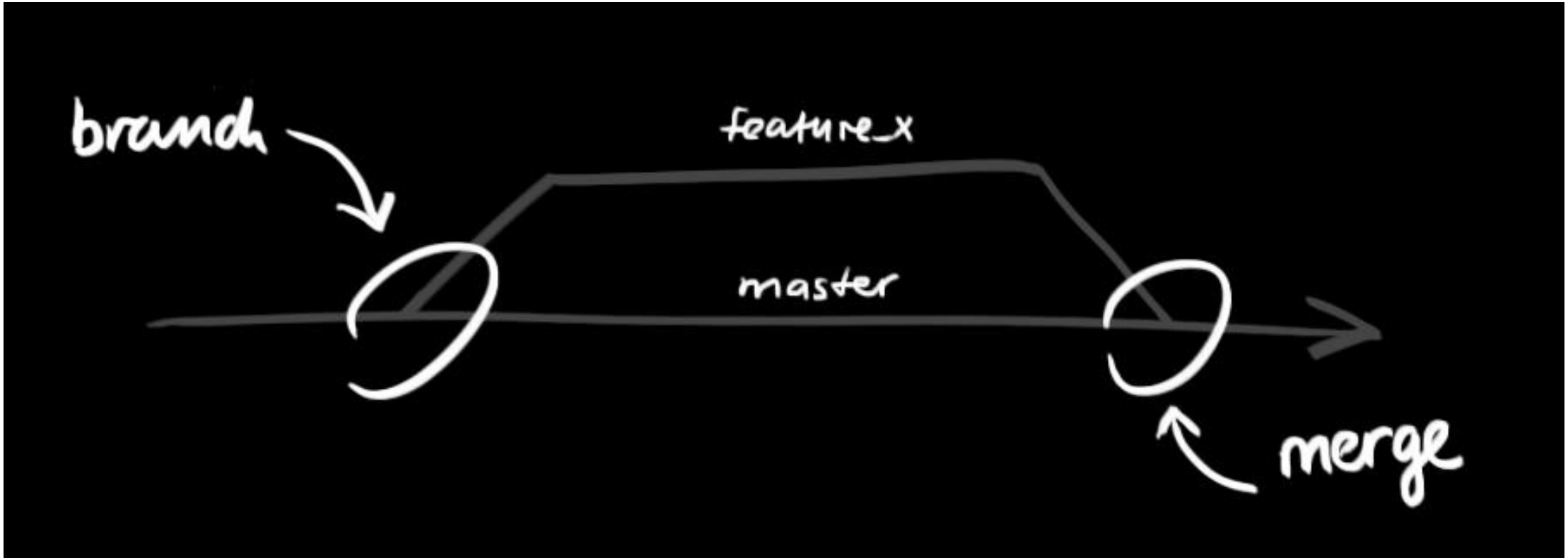
```
➜  demo2 git:(master) ✗ git add hi.txt
➜  demo2 git:(master) ✗ git commit -m "修改了hi.txt"
[master 332e69a] 修改了hi.txt
 1 file changed, 1 insertion(+)
➜  demo2 git:(master) git push -u origin master
To git@github.com:fankay/gitdemo.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'git@github.com:fankay/gitdemo.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.
➜  demo2 git:(master) git pull origin master
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:fankay/gitdemo
 * branch            master     -> FETCH_HEAD
   567bda5..d939660  master     -> origin/master
Auto-merging hi.txt
CONFLICT (content): Merge conflict in hi.txt
Automatic merge failed; fix conflicts and then commit the result.
```

git会使用标记处冲突的位置，需要手动处理冲突代码。

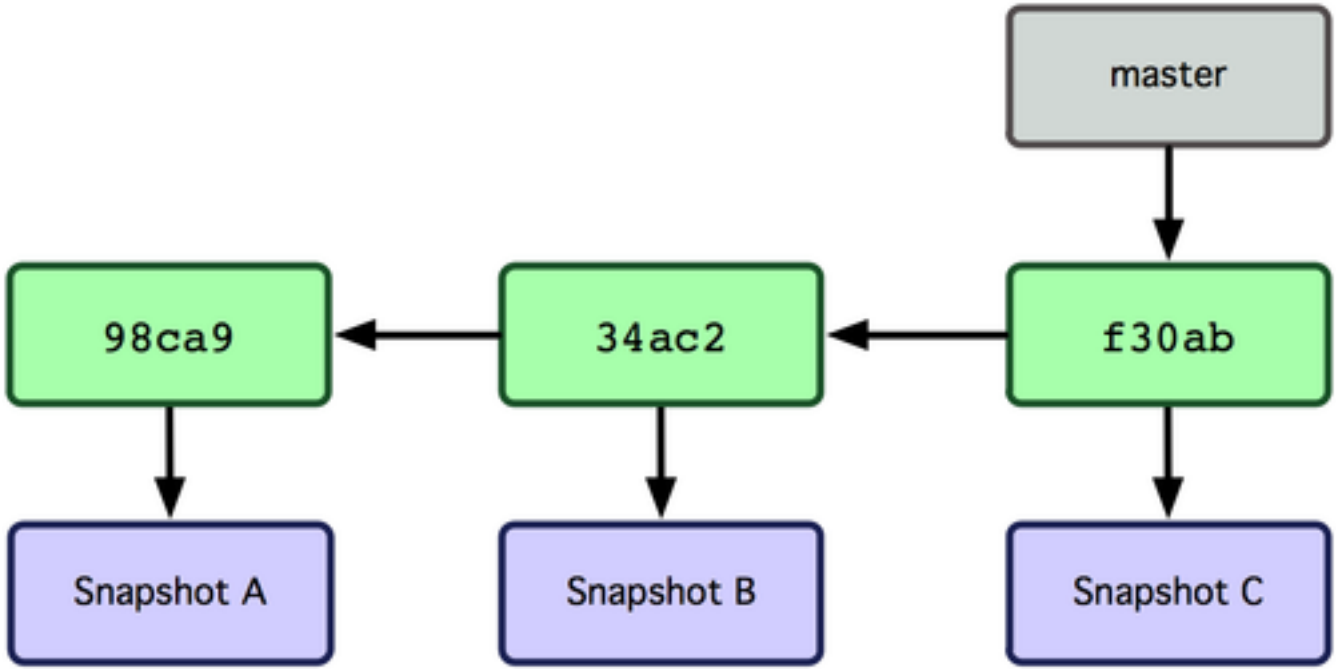
处理后在重新提交代码即可

```
<<<<<<< HEAD
jack and rose
=======
tom and jerry
>>>>>>> d93966041af245cc4f824630f5e3ac44f53d6441
```

分支管理



master分支



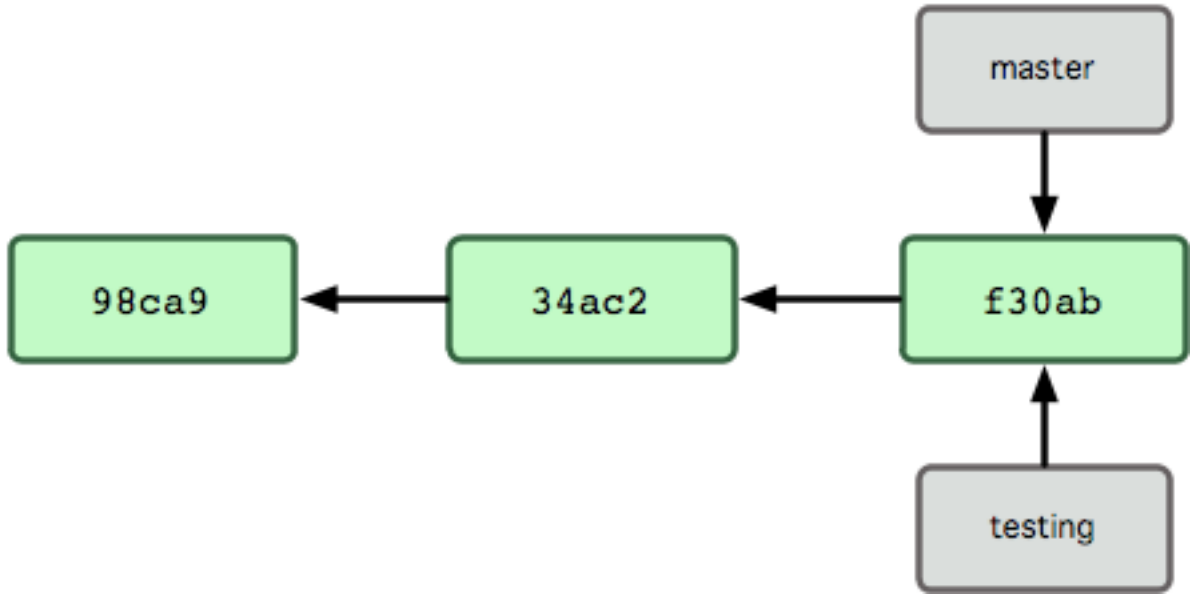
开启新分支

#创建新的分支
git branch 分支名称

#查看当前仓库中的分支
git branch

#查看远程分支
git branch -r

#查看所有分支
git branch -a



切换分支

凯盛软件

#切换分支

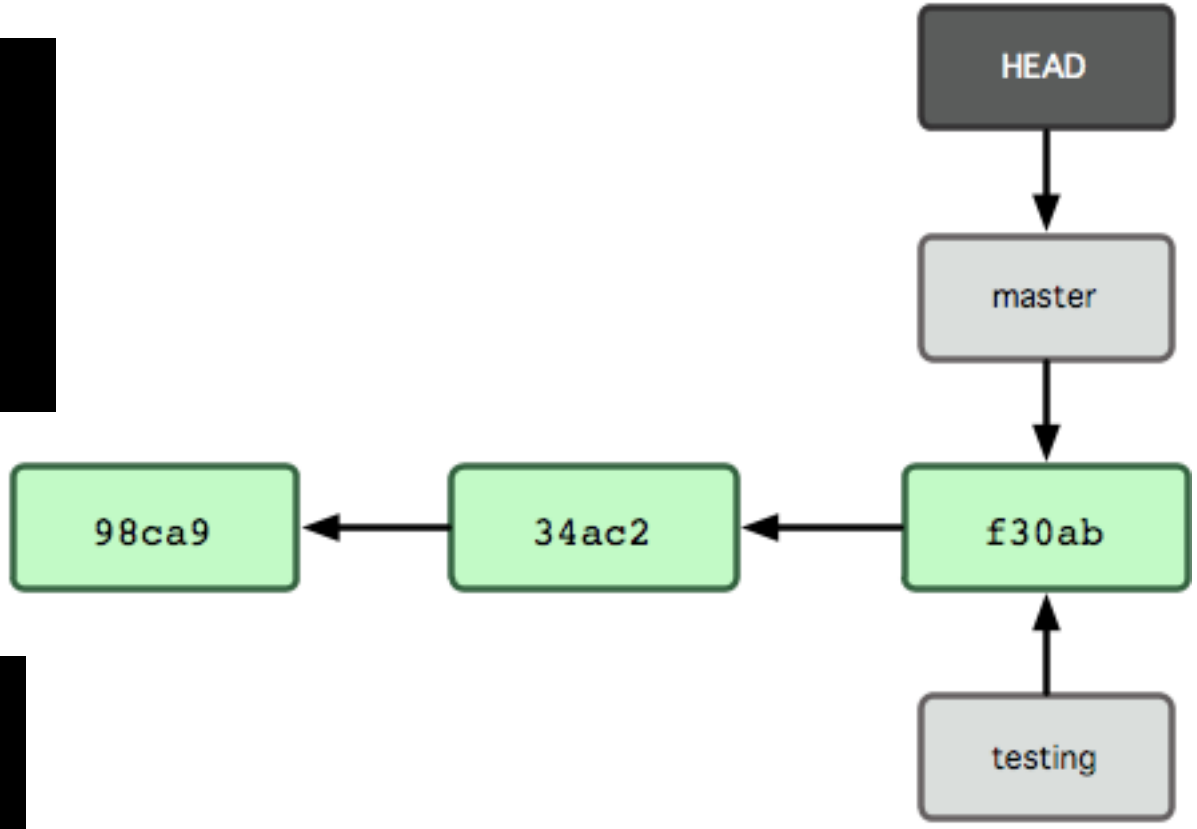
git checkout 分支名称

```
➜  demo2 git:(master) git branch testing
➜  demo2 git:(master) git branch
* master
  testing
➜  demo2 git:(master) git checkout testing
Switched to branch 'testing'
```

#创建并切换分支

git checkout -b 分支名称

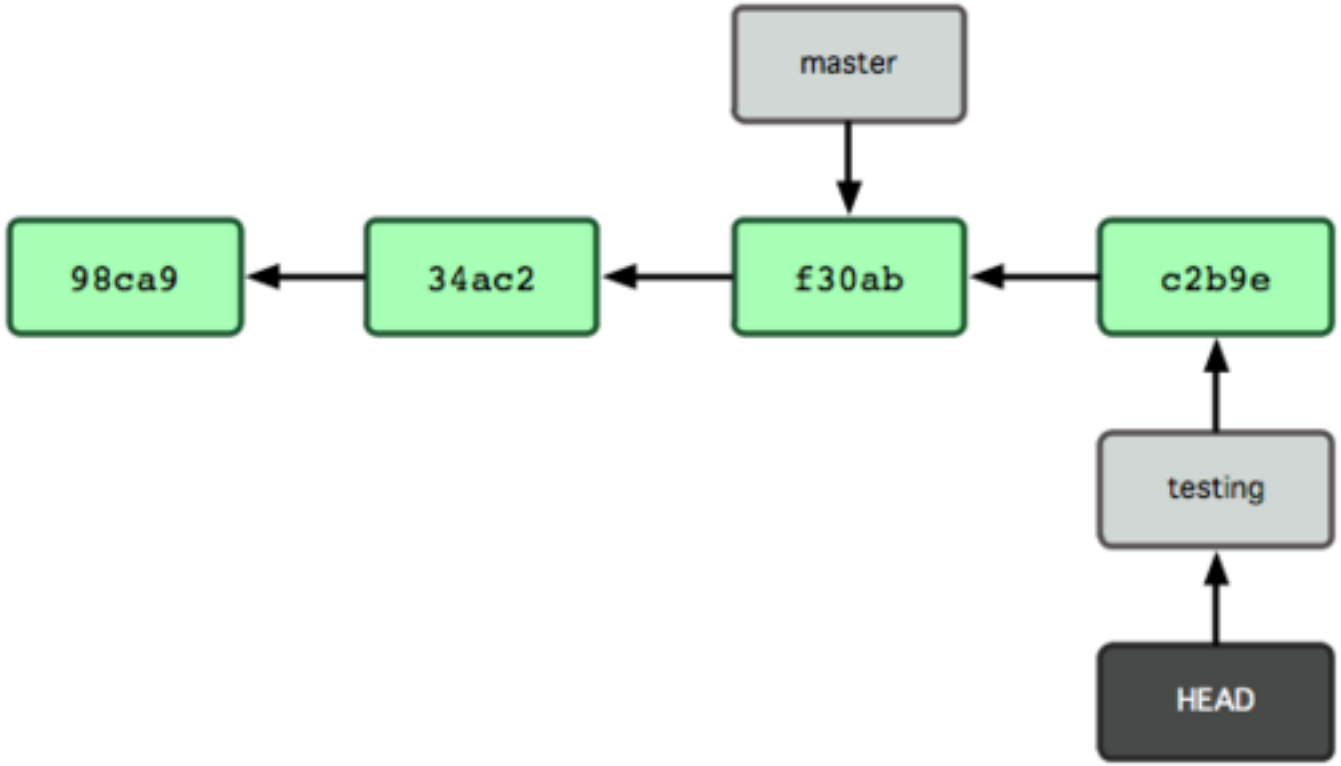
```
➜  demo2 git:(testing) git checkout -b dev
Switched to a new branch 'dev'
➜  demo2 git:(dev)
```



在分支上修改文件

凯盛软件

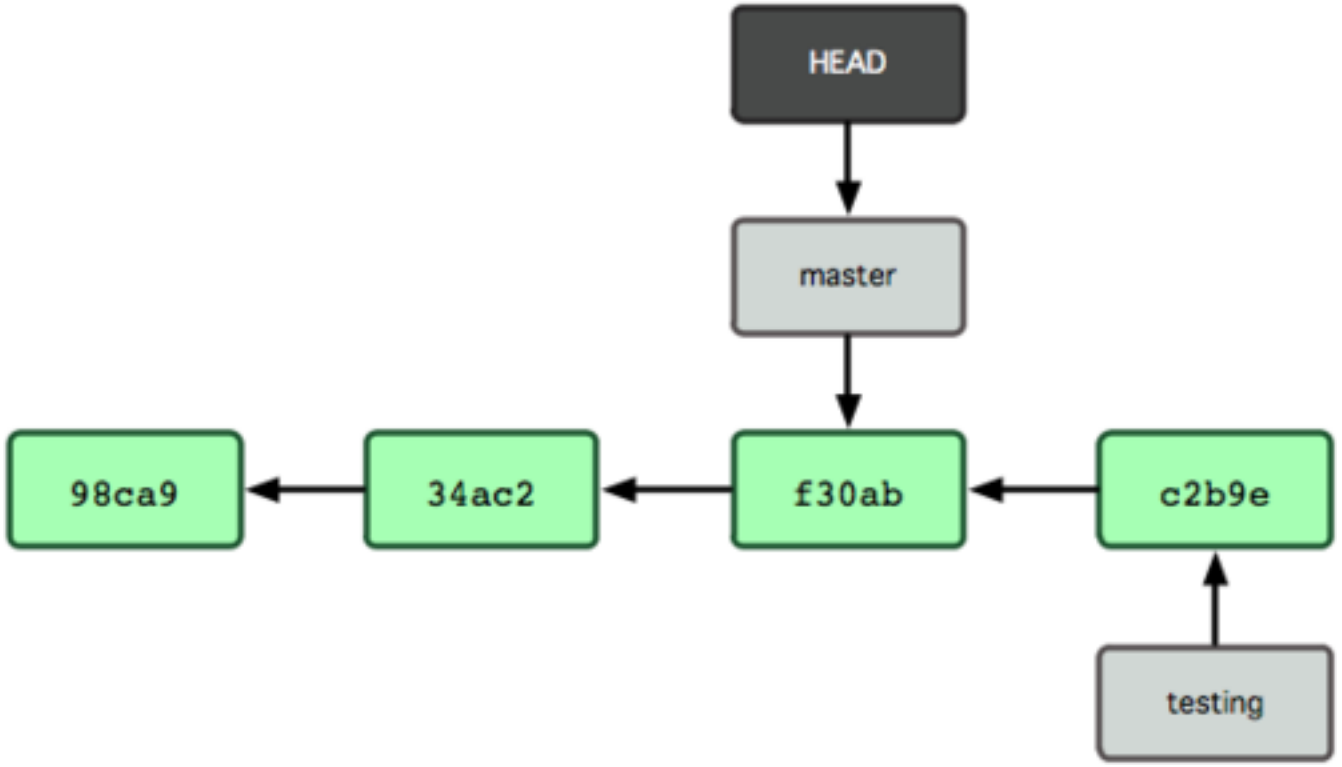
```
➜  demo2 git:(master) git checkout testing
Switched to branch 'testing'
➜  demo2 git:(testing) git add .
➜  demo2 git:(testing) ✗ git commit -m "修改了new.txt"
[testing ca15991] 修改了new.txt
 1 file changed, 1 insertion(+)
```



合并分支

1. 切换回master分支

```
➜  demo2 git:(testing) git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```



2. 将testing分支合并到master分支上

git merge 分支名称

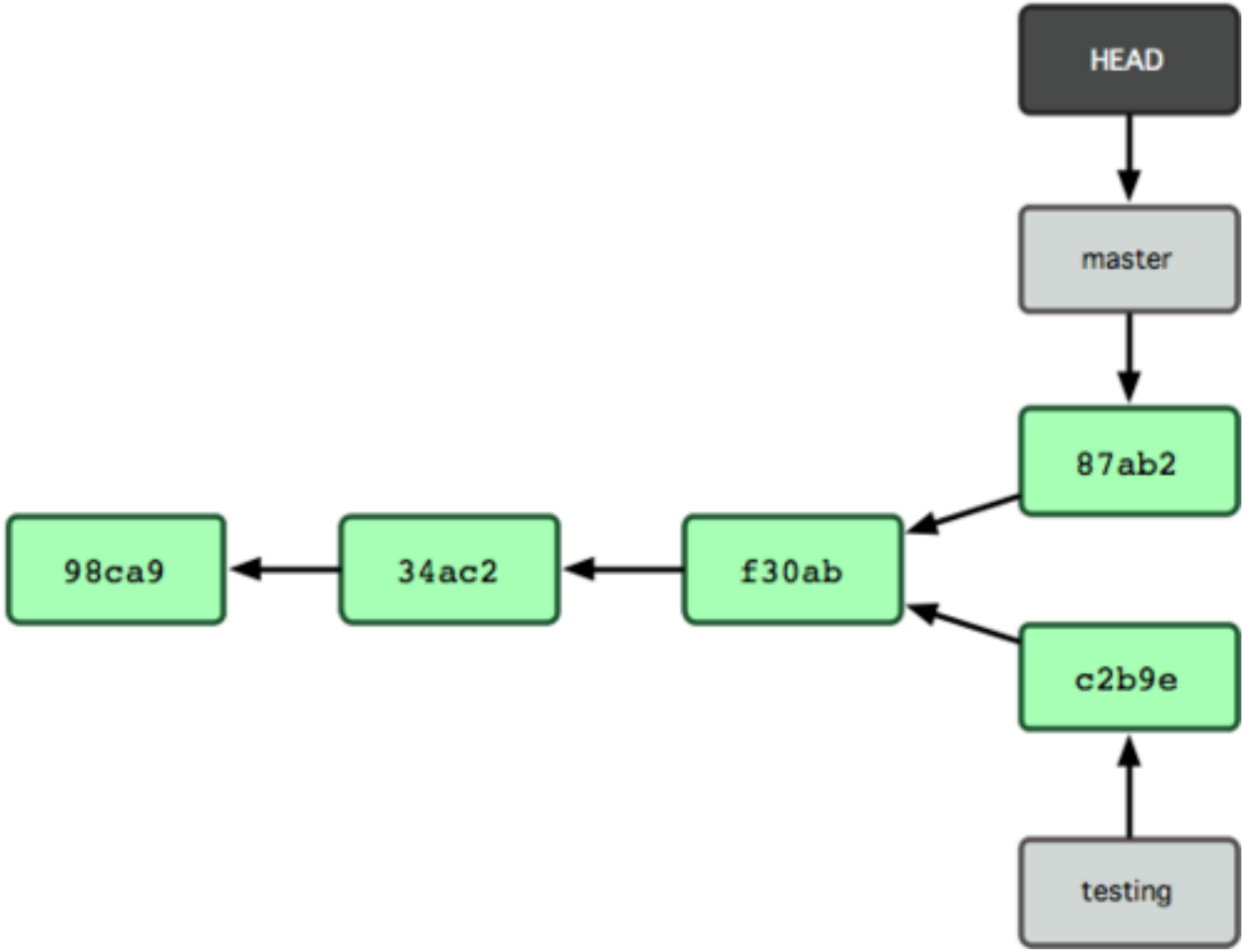
```
➜  demo2 git:(master) git merge testing
Updating 4c1f8d8..ca15991
Fast-forward
 new.txt | 1 +
 1 file changed, 1 insertion(+)
```

3.删除分支

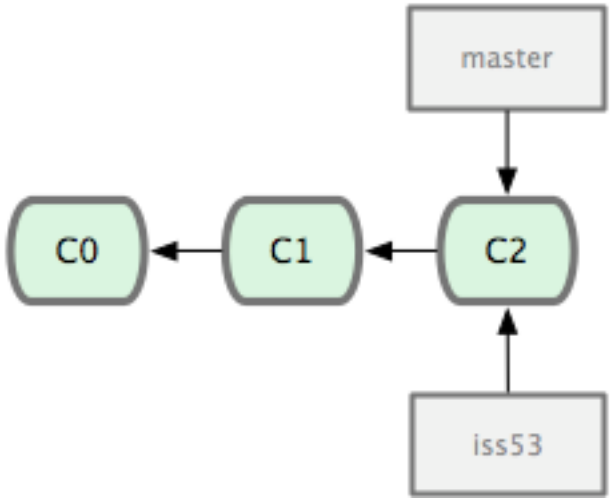
git branch -d 分支名称

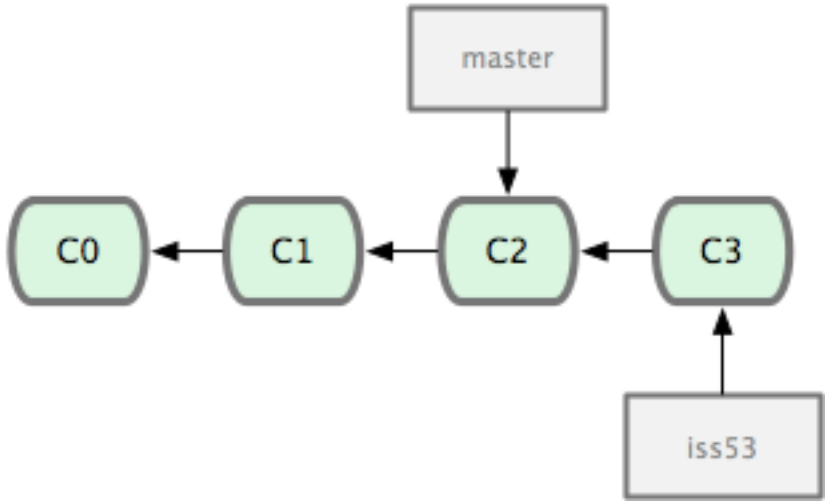
```
➜  demo2 git:(master) git branch -d testing
Deleted branch testing (was ca15991).
➜  demo2 git:(master) git branch
  dev
* master
```

常见的分支合并场景



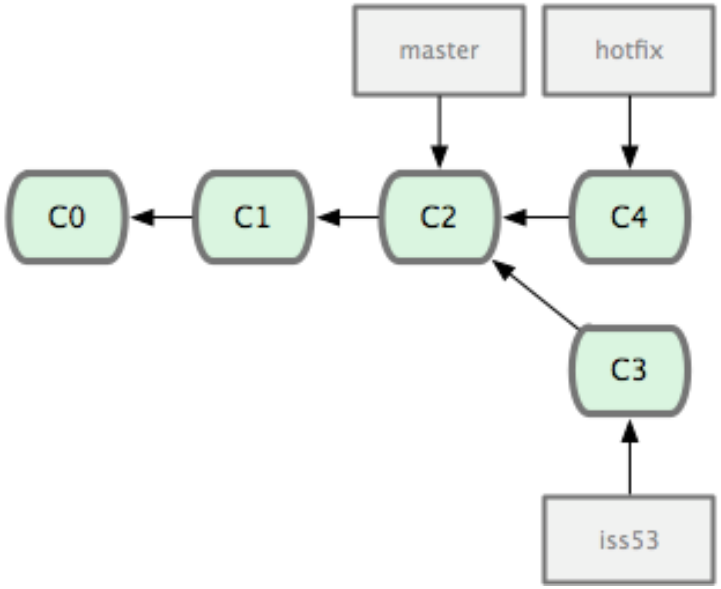
`` ➜ demo2 git:(master) git checkout -b testing Switched to a new branch 'testing' ➜ demo2 git:(testing) touch a.txt ➜ demo2 git:(testing) ✗ git add a.txt ➜ demo2 git:(testing) ✗ git commit -m "添加a.txt" [testing 9fcbe3e] 添加a.txt 1 file changed, 0 insertions(+), 0 deletions(-) create mode 100644 a.txt ```	创建testing分支，在该分支中创建新文件，并提交到该分支中
`` ➜ demo2 git:(testing) git checkout master Switched to branch 'master' Your branch is ahead of 'origin/master' by 1 commit. (use "git push" to publish your local commits) ➜ demo2 git:(master) git add new.txt ➜ demo2 git:(master) ✗ git commit -m "修改了new.txt,添加新数据" [master 5e65003] 修改了new.txt,添加新数据 1 file changed, 1 insertion(+) ```	切换回master分支，在该分支中修改文件，并提交到该分支中
`` ➜ demo2 git:(master) git merge testing Merge made by the 'recursive' strategy. a.txt | 0 1 file changed, 0 insertions(+), 0 deletions(-) create mode 100644 a.txt ```	合并分支



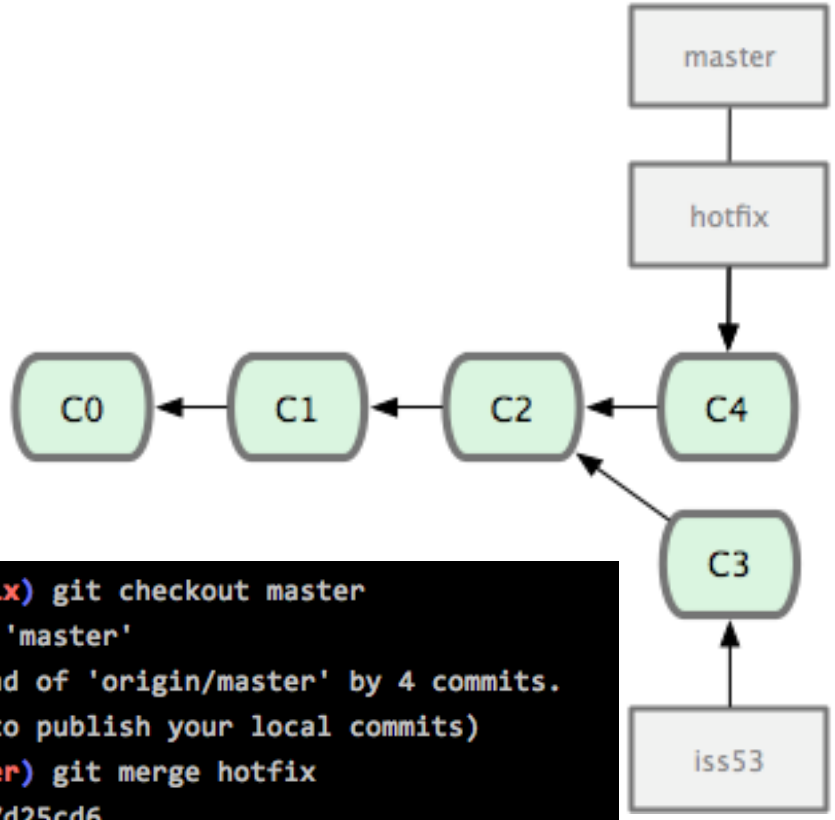



```
➜  demo2 git:(master) git checkout -b iss53
Switched to a new branch 'iss53'
```

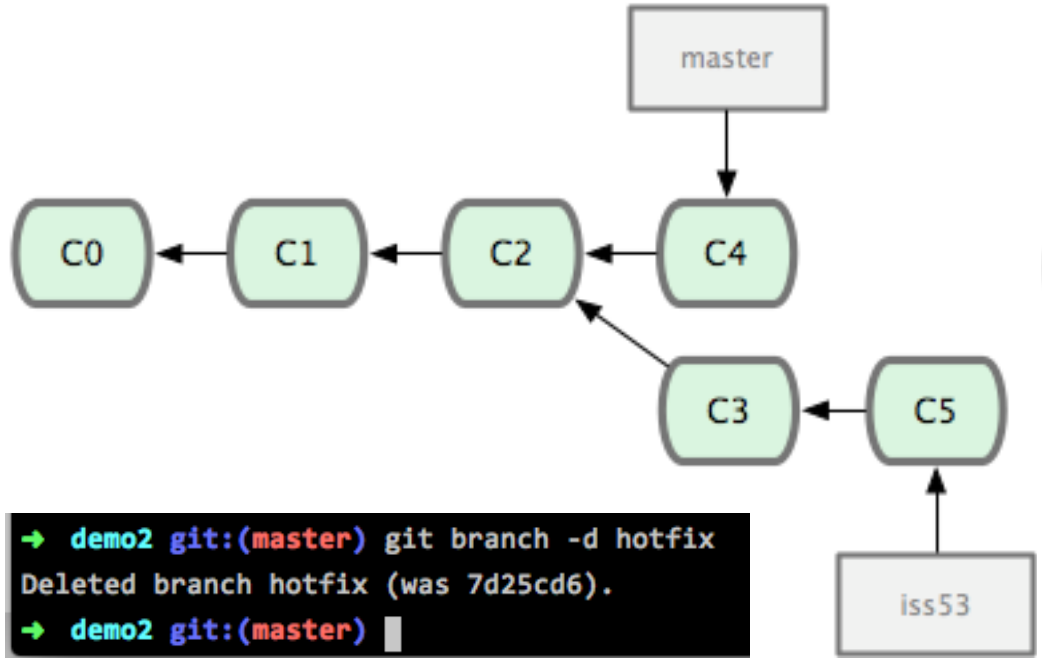
```
➜  demo2 git:(iss53) touch b.txt
➜  demo2 git:(iss53) ✗ git add b.txt
➜  demo2 git:(iss53) ✗ git commit -m "添加b.txt文件"
[iss53 db470c4] 添加b.txt文件
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 b.txt
```



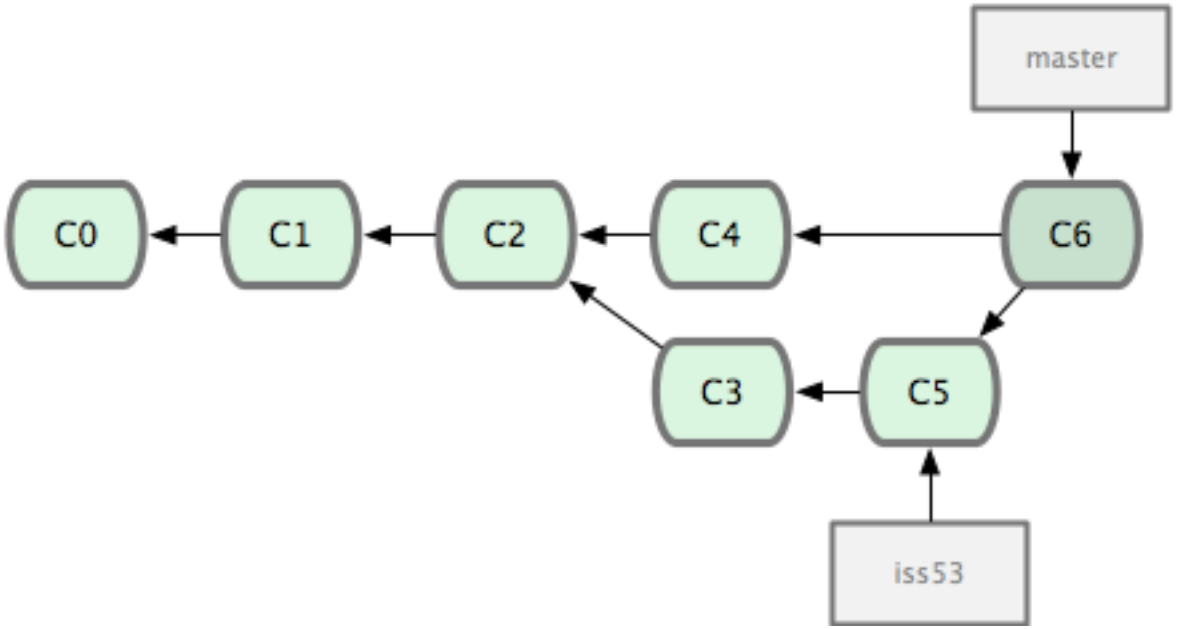

```
➜  demo2 git:(iss53) git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 4 commits.
  (use "git push" to publish your local commits)
➜  demo2 git:(master) git checkout -b hotfix
Switched to a new branch 'hotfix'
➜  demo2 git:(hotfix) git add hi.txt
➜  demo2 git:(hotfix) ✗ git commit -m "修复了hi.txt的bug"
[hotfix 7d25cd6] 修复了hi.txt的bug
 1 file changed, 1 deletion(-)
➜  demo2 git:(hotfix)
```




```
➜  demo2 git:(hotfix) git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 4 commits.
  (use "git push" to publish your local commits)
➜  demo2 git:(master) git merge hotfix
Updating 261e0fb..7d25cd6
Fast-forward
 hi.txt | 1 -
 1 file changed, 1 deletion(-)
```




```
➜  demo2 git:(master) git branch -d hotfix
Deleted branch hotfix (was 7d25cd6).
➜  demo2 git:(master)
```




```
➜  demo2 git:(master) git merge iss53 -m "合并iss53分支"
Merge made by the 'recursive' strategy.
 b.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 b.txt
```

其他命令

凯盛软件

#查看哪些分支已被并入当前分支（也就是说哪些分支是当前分支的直接上游）

git branch --merged

#查看尚未合并的分支

git branch --no-merged

远程分支

#将分支推送到远程仓库

git push 缩略名 分支名称

```
➜  demo2 git:(master) git push origin iss53
Total 0 (delta 0), reused 0 (delta 0)
To git@github.com:fankay/gitdemo.git
 * [new branch]      iss53 -> iss53
```

#删除远程分支

git push 缩略名 --delete 分支名称

```
➜  demo2 git:(master) git push origin --delete iss53
To git@github.com:fankay/gitdemo.git
 - [deleted]         iss53
```

#获取远程分支

git fetch origin

```
➜  demo2 git:(master) git fetch origin
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:fankay/gitdemo
 * [new branch]      hotfix     -> origin/hotfix
```

#将新的远程分支合并到当前分支

git merge 缩略名/分支名

```
➜  demo2 git:(master) git merge origin/hotfix
Updating 31ff1cd..2290af7
Fast-forward
 a.txt | 1 +
 1 file changed, 1 insertion(+)
```

#获取远程分支并创建本地分支

git checkout -b 分支名称 缩略名/分支名称

```
➜  demo2 git:(master) git fetch origin
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:fankay/gitdemo
 * [new branch]      iss02      -> origin/iss02
➜  demo2 git:(master) git checkout -b iss02 origin/iss02
Branch iss02 set up to track remote branch iss02 from origin.
Switched to a new branch 'iss02'
➜  demo2 git:(iss02) git merge origin/iss02
Already up-to-date.
```

Stashing

凯盛软件

场景：

经常有这样的事情发生，当你正在进行项目中某一部分的工作，里面的东西处于一个比较杂乱的状态，而你想转到其他分支上进行一些工作。但是，当前你不想提交进行了一半的工作。

暂存：

git stash

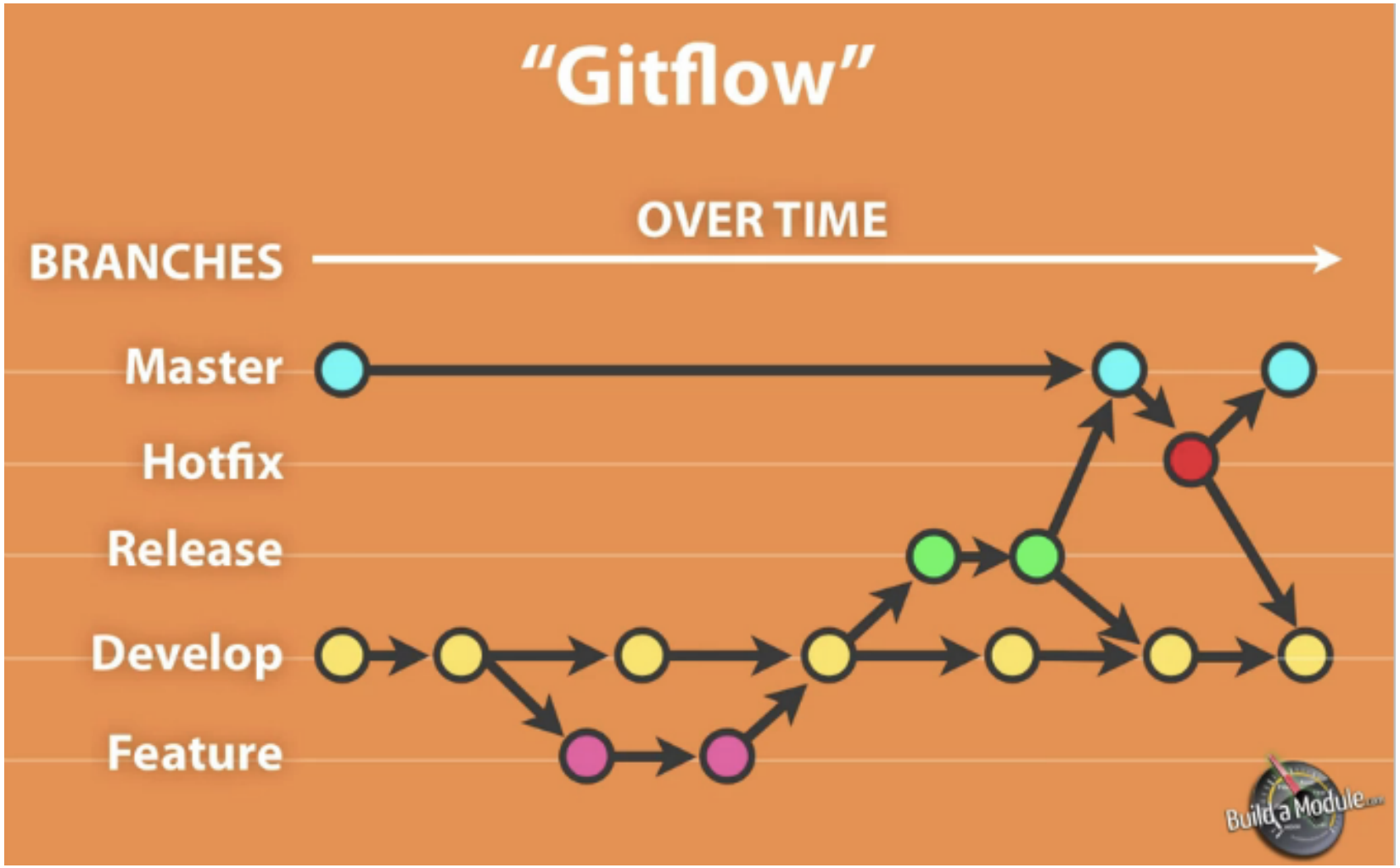
查看暂存列表

git stash list

从暂存列表中取出

git stash pop

Git工作流


"Gitflow"
OVER TIME
BRANCHES
Master
Hotfix
Release
Develop
Feature
Build a Module