Git Best Practice

Git 规范与最佳实践

Michael He

Agenda

- 简介
- 基础命令
- 最佳实践
- 分支管理
- 深入 .git 目录
- 推荐工具

关于技术分享

- 通过分享可以更好的自我审视和学习。（费曼学习法之一）
- 希望大家能够以一种抛转引玉的心态，通过分享去获取获取新的知识。
- 让跨团队可以在另一个主题下进行沟通和学习，促进跨团队交流。我们不能在一个项目里成为战友，那么可以在一次技术分享会上成为朋友。
- 希望各位，不吝赐教，今后能够踊跃分享技术主题。
- 这就是：**字节范**

Git 简介

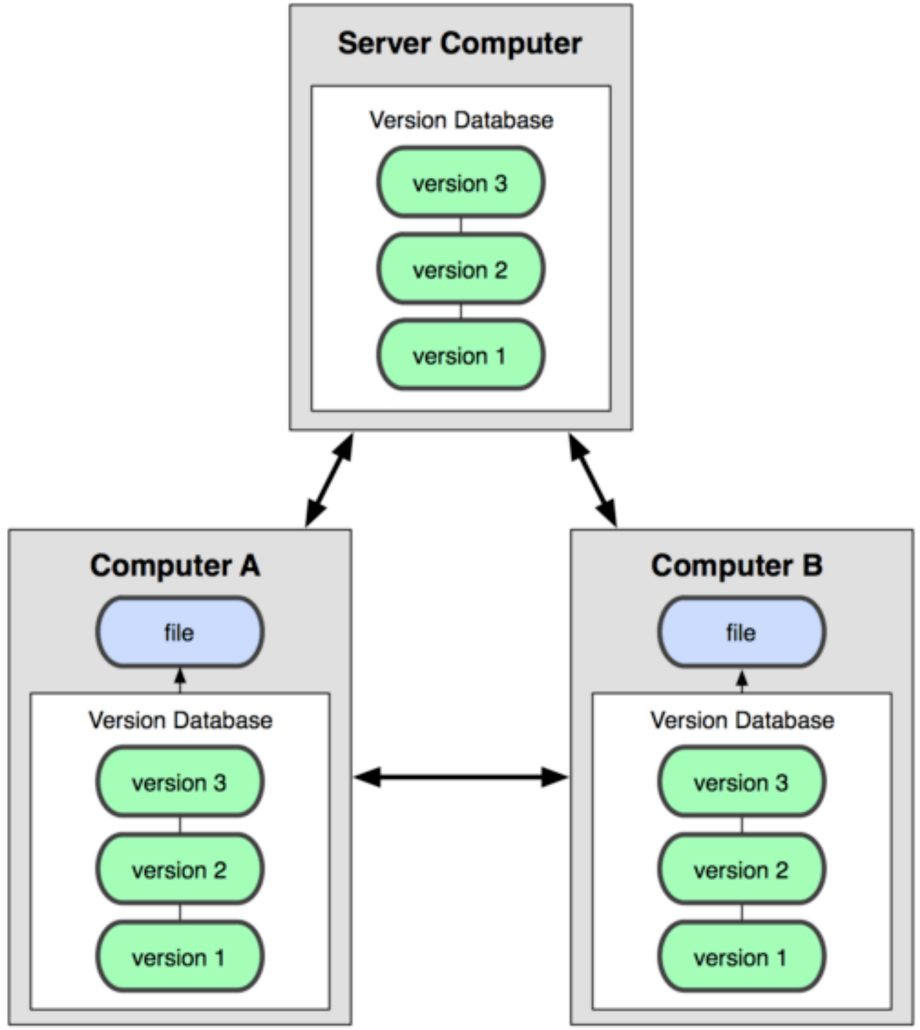
Git /git/ 是由 Linus Torvalds 也就是linux作者开发的。其实Contributer排名第一的是一个日本人叫做濱野純（From Google）。

Git 源于英语俚语，意思是 The stupid content tracker, 愚蠢的内容追踪器。

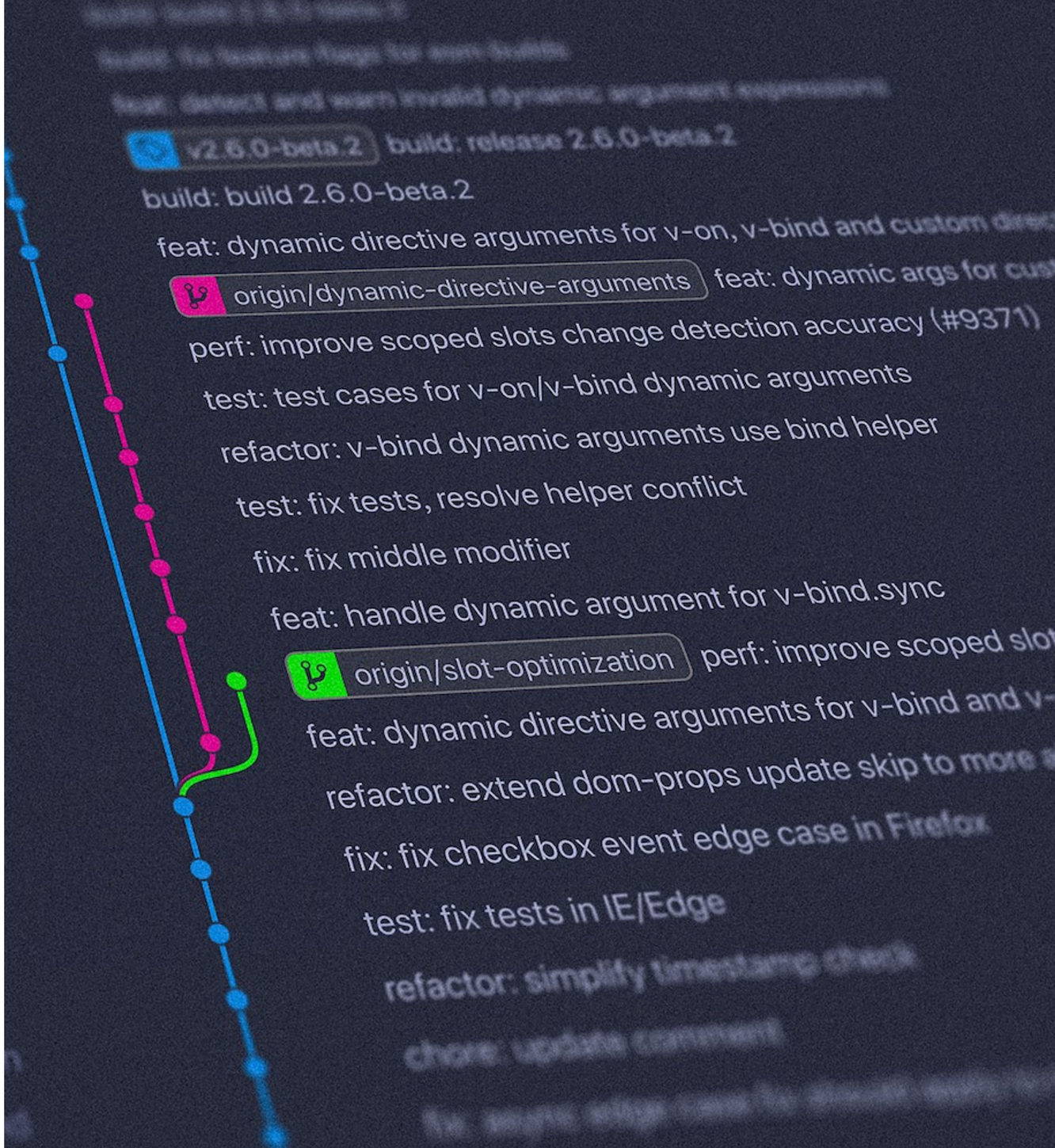
最开始被开发的时候主要是用于 Linux 内核开发的版本控制工具。

他跟以前的CVS, SVN最大的区别就是，他是分布式的管理代码库。

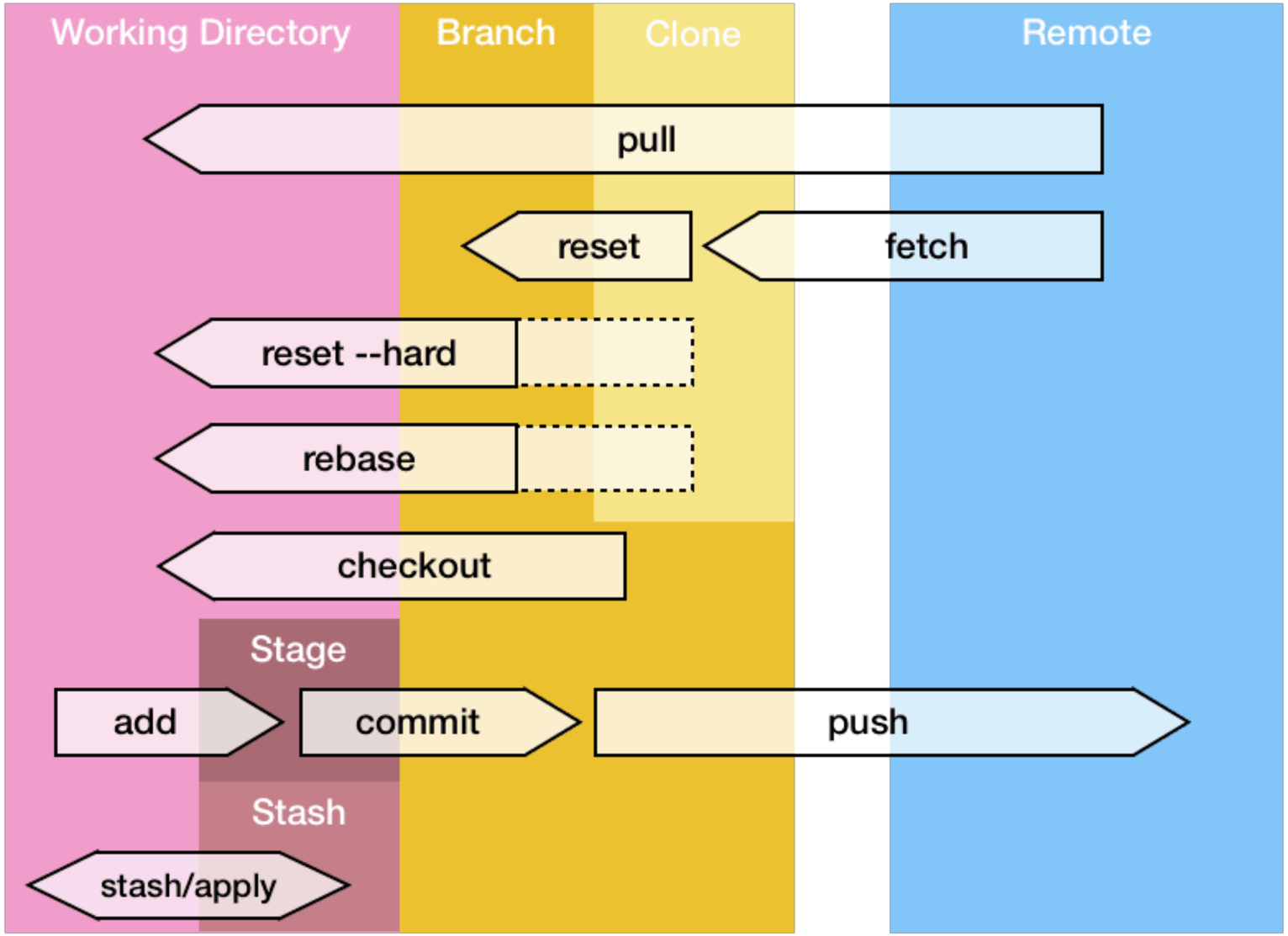
每个commit都可以在本地进行，之后同步服务器，极大增进了开发体验。



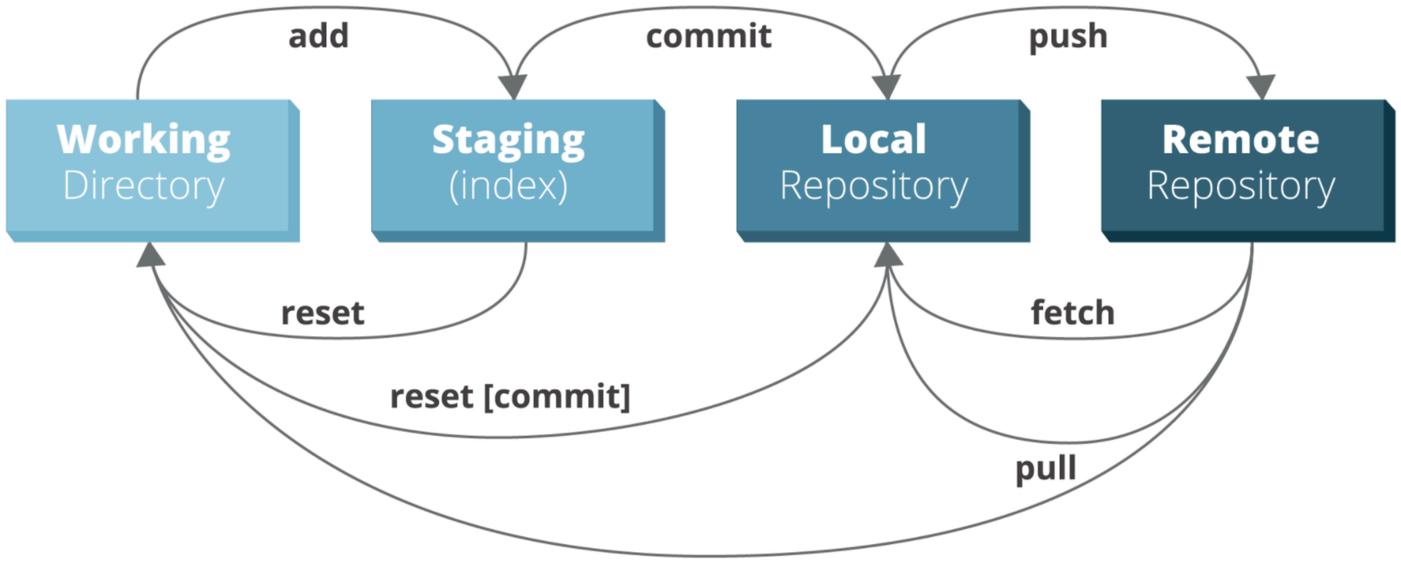
基础命令


v2.6.0-beta.2
build: release 2.6.0-beta.2
build: build 2.6.0-beta.2
feat: dynamic directive arguments for v-on, v-bind and custom
origin/dynamic-directive-arguments
feat: dynamic args for
perf: improve scoped slots change detection accuracy (#9371)
test: test cases for v-on/v-bind dynamic arguments
refactor: v-bind dynamic arguments use bind helper
test: fix tests, resolve helper conflict
fix: fix middle modifier
feat: handle dynamic argument for v-bind.sync
origin/slot-optimization
perf: improve scoped
feat: dynamic directive arguments for v-bind and
refactor: extend dom-props update skip to
fix: fix checkbox event edge case in Firefox
test: fix tests in IE/Edge
refactor: simplify
chore: update

Git基础命令

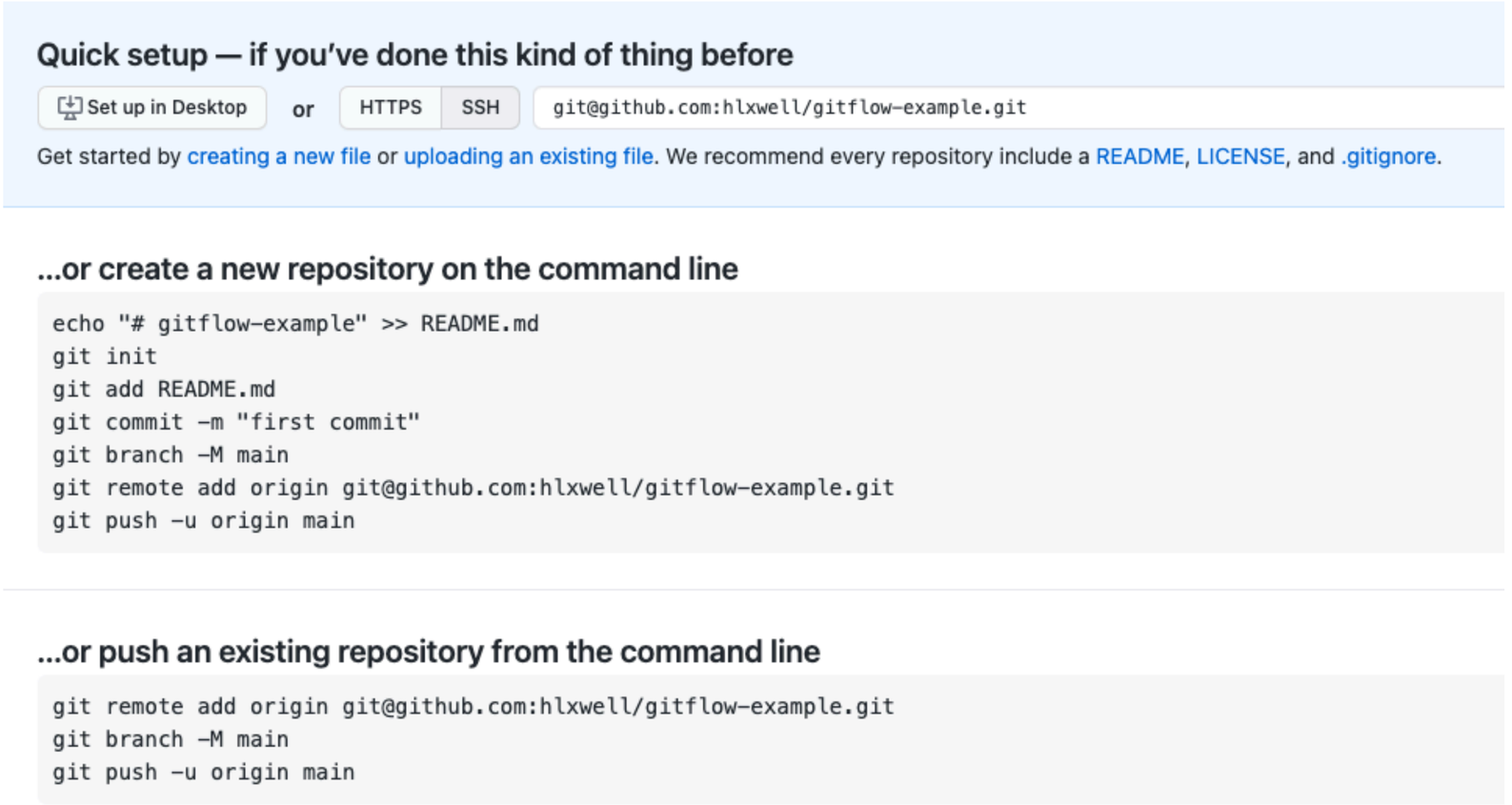

Working Directory
Branch
Clone
Remote
pull
reset
fetch
reset --hard
rebase
checkout
Stage
add
commit
push
Stash
stash/apply

Git Stages 状态流程图



Git 创建

- Git init
- Git clone [url]
  注意 https 和 ssh 的区别。
  https 使用 http basic auth
  所有会要求每次都输入密码。
- 学会查看 man
  1. man git-[subcmd]
  2. git cmd ——help
  3. git help cmd



Git Remote 管理

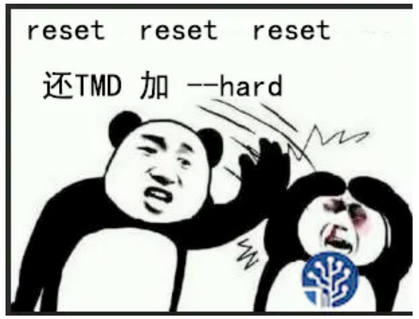
- Git fetch
  仅仅把 remote 服务器数据同步到本地的 remotes 下的分支。
  git fetch -p 可以清理远程已删除的本地分支。
- Git pull [remote] [branch]
  不加参数是把服务器上的所有分支下载到本地，加branch他会把 remote branch merge 到本地的branch。
  git fetch && git merge branch origin/branch
- Git pull ––rebase
  加了rebase以后，就等于 **git fetch && git rebase branch origin/branch**
- Git push [––force]
  把本地所有的分支 push 到 remote. 等于 **git push remote {branch1, branch2, ...}**
  ––force 的意思是，覆盖远程的分支，谨慎使用。
- Git push [remote] :branch
  删除远程分支。
- Git remote add new-remote-name [url]
  添加一个新的remote，常用于同步两个代码库的内容。

Git Local 查看

- Git status
  查看当前working directory状态。包括增删改的文件。
  git status ––sb 简易版 status 并且显示分支。
- Git branch [–all]
  查看本地所有的branch，如果加了--all 可以看到 remotes 的分支。
- Git diff
  默认是查看 working directory 和 working branch 的 difference。也可以查看任意两个分支或者commit之间的差别。生成 Patch: **git diff > xxx.patch**, **git apply xxx.patch**
- Git blame
  查看哪行代码是谁在哪个commit里面写的。
- Git show [commit]:[file]
  查看一个 commit 内容，或者 commit 下某个文件的变更。
- Git log [file|folder]
  查看commit log 或者具体文件夹或者文件的log。
  可以利用 git log branch1..branch2 去查看两个分支的差别。
- Git shortlog -sn
  查看建议压缩版的log，经常用于查看 contributer 提交数量。

Git commit 管理

- Git add [.|file]
  添加文件或者changes到 **Stage**。
- Git revert [commit-num]
  把以前的commit内容去掉，并且**生成一个新的 commit**。
- Git cherry-pick [commit-num]
  把任意一个commit给pick到当前分支。
- Git commit -m|-am “message”
  提交stage到local branch
- Git reset [–soft|–hard] [commit] [file]
  Unstage file，把local branch 的 commit 删掉，删掉的内容有soft和hard两种处理方式。
  –soft 的意思是把删掉commit放到stage，你可以重新整理再提交。
  –hard 的意思是把删掉的commit内容全部从stage也删除，就是不要了。
  Tip: **git checkout –– . 等于 git reset ––hard**可以把stage的内容全部清除。
- Git clean -fx
  删除 untracked files. (什么叫 untracked file?)
  https://www.jianshu.com/p/0b05ef199749



Git Branch 管理

- Git checkout xxx-branch
  切换分支
- Git checkout -b new-branch-name [start-from-branch]
  创建新的local branch
- Git branch -d branch-name-to-delete
  删除本地分支
- Git push remote **:branch-name-to-delete**
  删除远程分支
- Git merge [––squash]
  merge 分支，默认是保留分支所有的commit。
  加了 squash 会合并所有的 commits，目的是去掉那些没有意义的中间修改commit。
- Git merge ––no-ff
  merge 分支，并且不采用 **fast-forward** 的方式。
  为了创建一个merge commit，在历史记录里清晰表明merge过程。
- Git rebase [branch]
  Rebase 当前 branch 到其他branch。
- Git rebase –onto NEW-BASE FROM-COMMIT TO-COMMIT
  选择任意片段merge到任意commit。
- Git mergetool
  一个处理conflict的工具

Git Stash 管理

- Git stash
  Push所有work directory和stage的东西到stash。
- Git stash apply & Git stash pop
  把最近一个stash恢复到work directory和stage，pop会删除stash，apply则不会。
- Git stash show [stash_id]
  查看具体一个stash内容。
- Git stash list
  查看历史stash。
- Git stash clear
  删除所有stash。

Git Tag 管理

- Tag 主要用途就是为了发布版本
- Git tag new-tag-name
  添加一个tag
- Git tag -l "v1.*"
  通过regexp来列出 tags
- Git push origin –tags
  上传所有的tags。
- Git push origin v1.2
  上传单个tag。
- Git tag -d v1.2
  删除tag，怎么删除远程tag？(查看branch)

Git Revision

修订版本

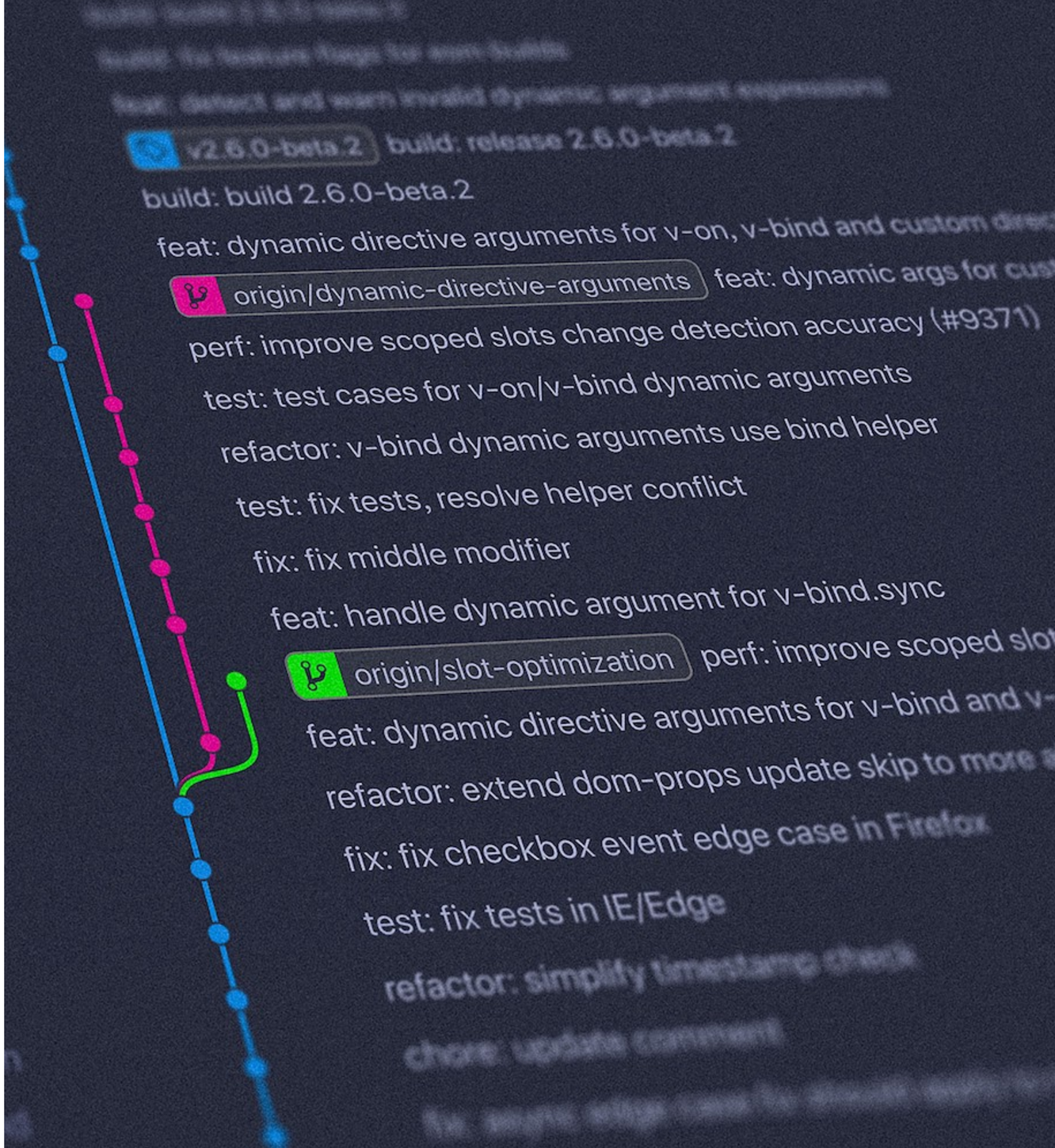
- HEAD
  当前本地分支的最新一个 commit.
- HEAD^
  HEAD的上一个commit
- HEAD^^^ == HEAD^3
  上上上一个版本
- HEAD~1
  当前本地分支的最新一个 commit.
- HEAD~3
  上上上一个版本

实战小任务

Are You Ready?

- 创建新的 git repo
- 创建 README.md
- 把 [自己名字.md] 放到项目里面来
- 在你名字的文件里面加上你的兴趣爱好
- 然后把自己的名字放到 README.md
- [老板说在README加个老板名字]
  stash 一下，去改 README，然后回来恢复stash
- 合并分支
- 查看某个分支上面发生的所有事情
- 哥们A添加一个密码到自己.md文件
- 然后我们来帮忙把这个密码给销声匿迹
- 老王辞职了，去掉"老王"分支的所有变更

最佳实践


v2.6.0-beta.2
build: release 2.6.0-beta.2
build: build 2.6.0-beta.2
feat: dynamic directive arguments for v-on, v-bind and custom
origin/dynamic-directive-arguments
feat: dynamic args for
perf: improve scoped slots change detection accuracy (#9371)
test: test cases for v-on/v-bind dynamic arguments
refactor: v-bind dynamic arguments use bind helper
test: fix tests, resolve helper conflict
fix: fix middle modifier
feat: handle dynamic argument for v-bind.sync
origin/slot-optimization
perf: improve scoped
feat: dynamic directive arguments for v-bind and
refactor: extend dom-props update skip to
fix: fix checkbox event edge case in Firefox
test: fix tests in IE/Edge
refactor: simplify
chore: update

- 一个 Commit 一个目的
- Commit Message 必须有意义
- 使用 .gitignore 避免系统生成的文件和local config提交到repo。
- 不要直接 commit 到 master, 要常开分支
- 尽量不要改写 master 的 history. (Push –force)
- 在分支上开发的时候用 rebase 而不是 merge (pull ––rebase)
- develop 到 master 需要用 merge 压缩(squash) commits
- 分支到 develop 需要用 merge without fast forward
- 使用 git-flow 来规范团队协作
- 使用 ~/.gitconifg 配置去简化操作

.gitconfig 配置

- 几大 section
  - Core 基础配置，如编辑器，显示参数
  - Alias 可以自定义很多快捷命令

    git st => git status
    git bd xxx => git branch -d xxx
    git prom => git pull ––rebase origin master
    git pushod => git push origin develop
  - User 提交时的用户信息
  - Color 各种显示颜色
- 参考规范

  https://gist.github.com/pksunkara/988716

```
[alias]
        a = add --all
        ai = add -i
        #############
        ap = apply
        as = apply --stat
        ac = apply --check
        #############
        ama = am --abort
        amr = am --resolved
        ams = am --skip
        #############
        b = branch
        ba = branch -a
        bd = branch -d
        bdd = branch -D
        br = branch -r
        bc = rev-parse --abbrev-ref HEAD
        bu = !git rev-parse --abbrev-ref --symbolic-full-name "@{u}"
        bs = !git-branch-status
        #############
        c = commit
        ca = commit -a
        cm = commit -m
        cam = commit -am
        cem = commit --allow-empty -m
        cd = commit --amend
        cad = commit -a --amend
        ced = commit --allow-empty --amend
        #############
        cl = clone
        cld = clone --depth 1
        clg = !sh -c 'git clone git://github.com/$1 $(basename $1)' -
        clgp = !sh -c 'git clone git@github.com:$1 $(basename $1)' -
        clgu = !sh -c 'git clone git@github.com:$(git config --get user.username)/$1 $1' -
```

.gitignore 配置

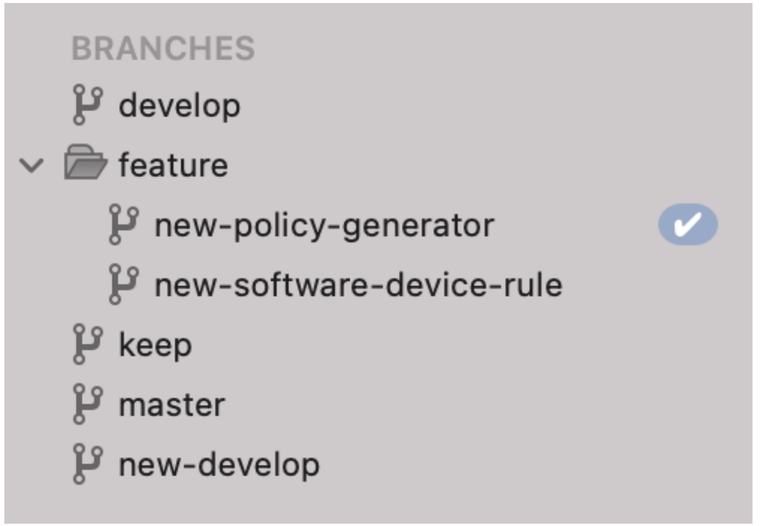
- 很多示例 project 的 .gitignore
  https://github.com/github/gitignore
- 可以ignore
  - 本地配置文件
  - 临时文件
  - 日志文件
  - 本地编辑器配置文件 .vscode
  - 第三方的库本地缓存
  - 本地 build出来的文件
- 可以用 .keep 文件去保留文件夹。

69 lines (55 sloc) | 1.32 KB

```
1   *.rbc
2   capybara-*.html
3   .rspec
4   /db/*.sqlite3
5   /db/*.sqlite3-journal
6   /db/*.sqlite3-[0-9]*
7   /public/system
8   /coverage/
9   /spec/tmp
10  *.orig
11  rerun.txt
12  pickle-email-*.html
13
14  # Ignore all logfiles and tempfiles.
15  /log/*
16  /tmp/*
17  !/log/.keep
18  !/tmp/.keep
19
20  # TODO Comment out this rule if you are OK with secrets being uploaded to the repo
21  config/initializers/secret_token.rb
22  config/master.key
23
24  # Only include if you have production secrets in this file, which is no longer a Rails default
25  # config/secrets.yml
26
27  # dotenv, dotenv-rails
28  # TODO Comment out these rules if environment variables can be committed
29  .env
30  .env.*
31
32  ## Environment normalization:
33  /.bundle
34  /vendor/bundle
35
```

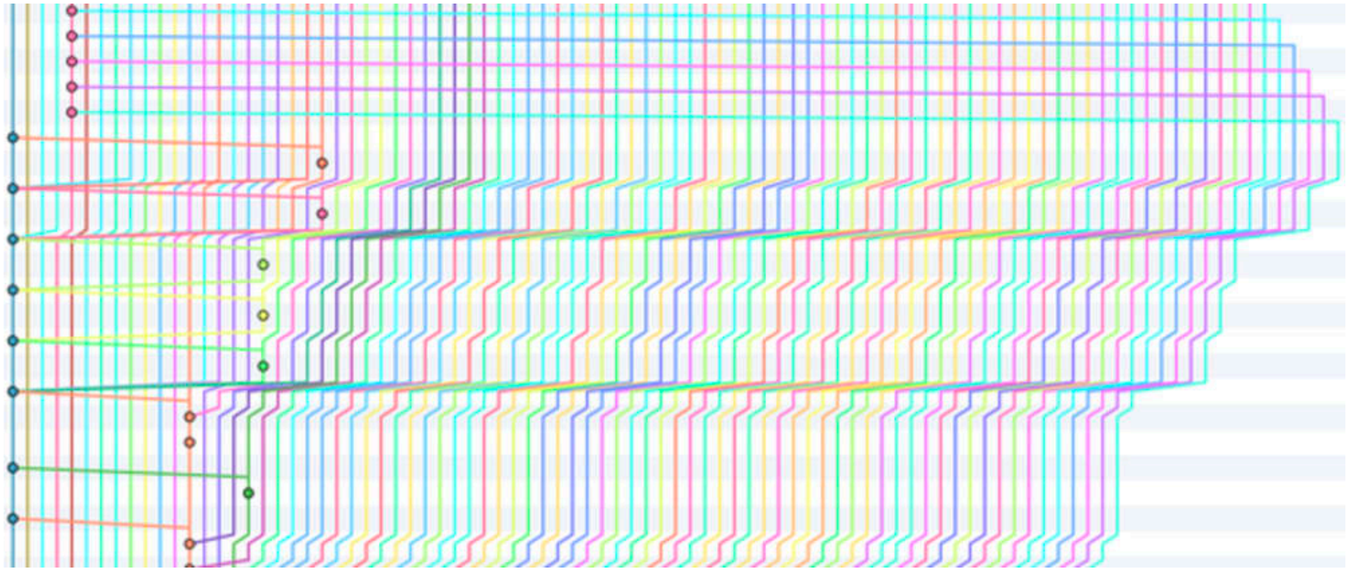
如何命名分支

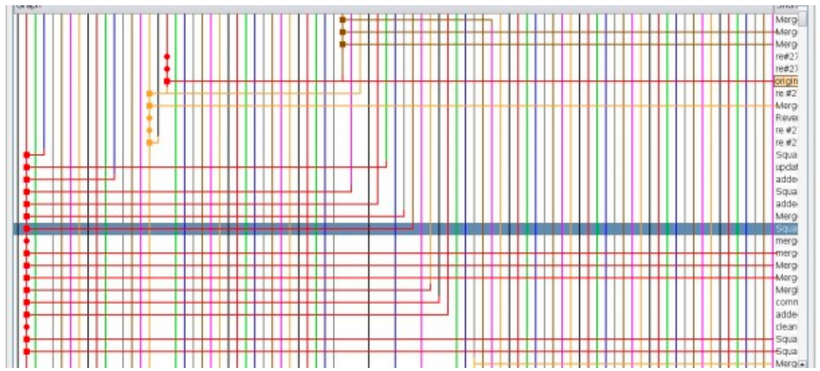
- 以一个 group 起头
  - feature/your-feature
  - hotfix/your-hotfix
  - fix/your-fix
  - chore/your-chore-work
  - wip/refactory-everything

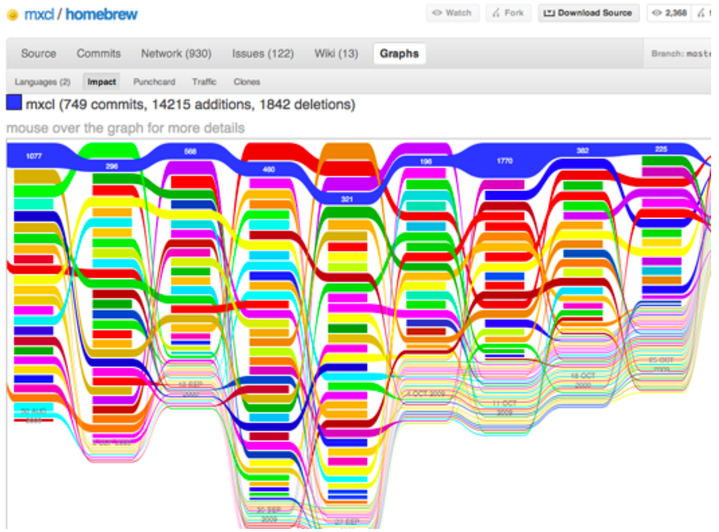


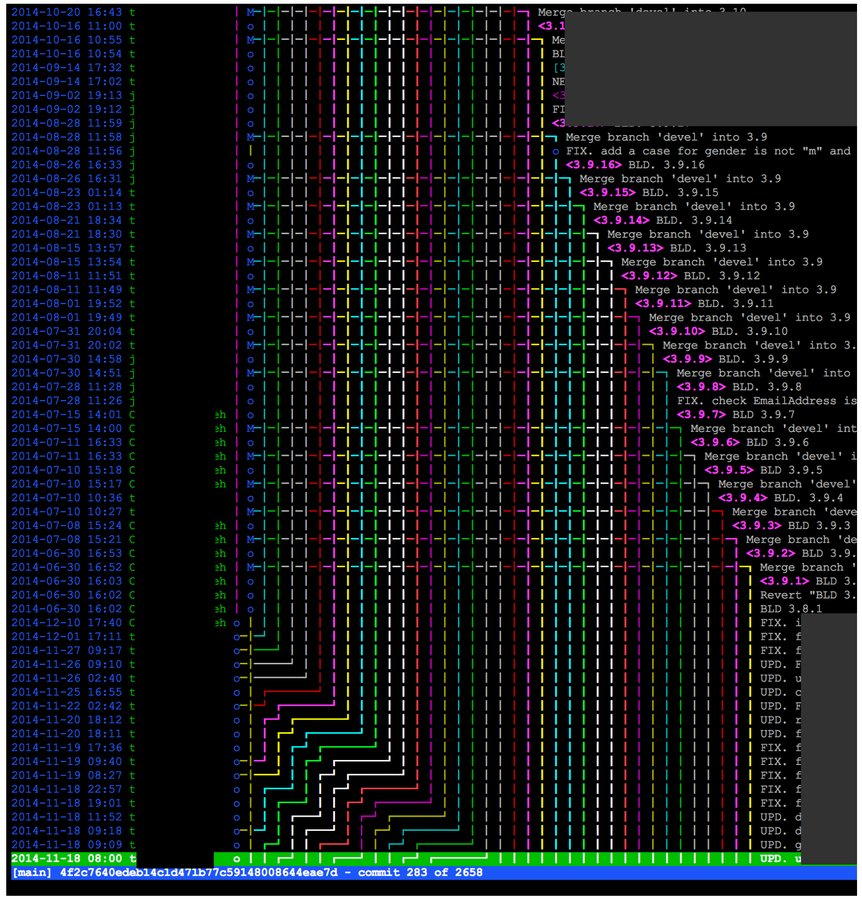
- [坑] 基于 **feature/xxx** 再建立子分支叫做 **feature/xxx/aaa** 为什么不行?
- 如果使用 issue tracker 比如gitlab issue, 那么可以使用 issue ID
  fix/1928-fix-popup-js-bug
- 使用 **– or __** 作为分隔字符

To avoid the chaos…





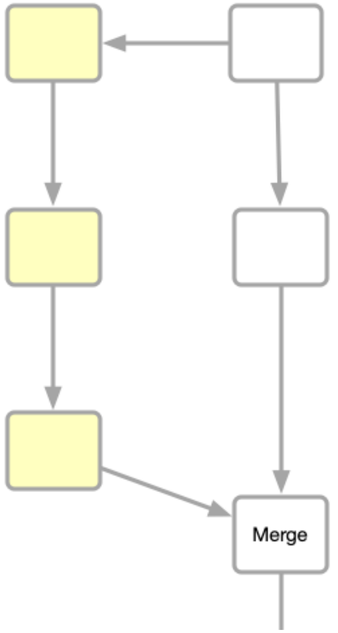




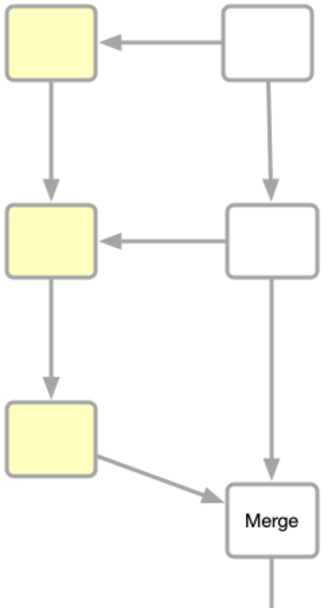
如何提交流程

- 首先你需要更新本地分支。
  git pull origin ––rebase feature/your-amazing-feature
- 如果不加 ––rebase 会出现一些merge。

  右图是期待分支，以及合并后的样子。
  原因是，为了分支管理的时候便于随时去掉某个分支的内容，最重容易管理。
- 在分支上面开发，尽量保持一条直线，除非是一个分支又有开叉，比如子功能。
- 然后 rebase 一下 develop 分支并且解决冲突。
- 解决完后，即可 push 回 origin。
- 别merge，看下面。(而且一般情况下你没权限。)



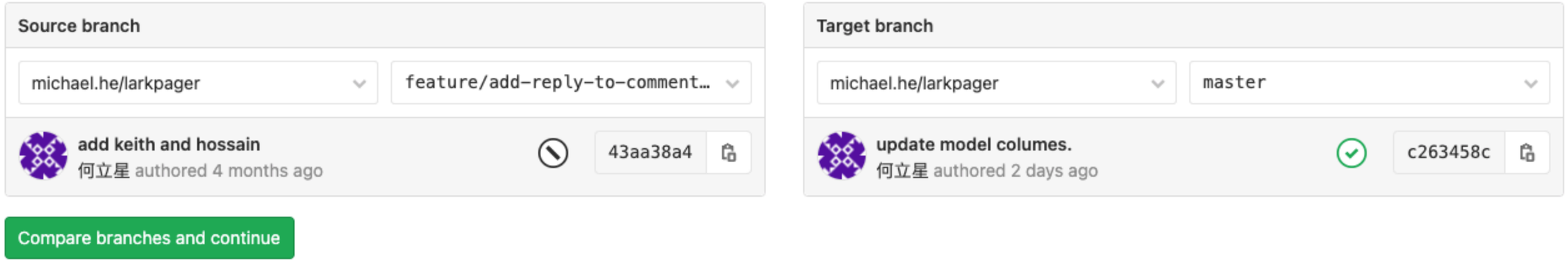
期待分支的样式



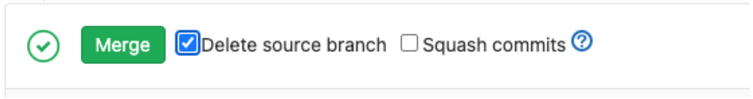
而不是分支的样式

如何合并分支

- 在 gitlab 上面点击 Create Merge Request。
- 然后选择你的分支和目标合并分支。



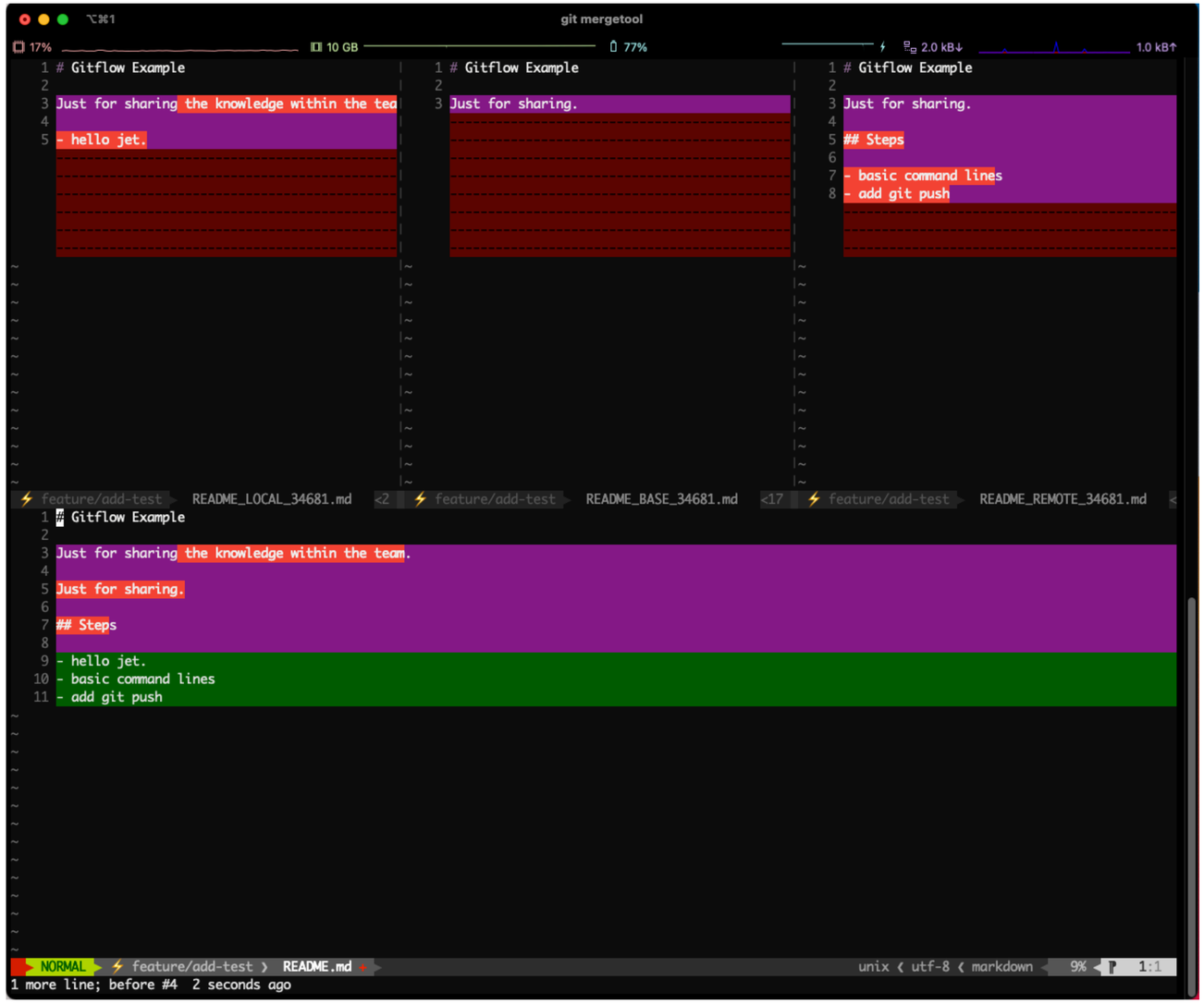
- 然后让团队成员 review 一下你的 merge request 内容。
- 最后让有权限的人来合并一下即可。
- 合并的时候可以选择删除分支，大部分情况不会用了。



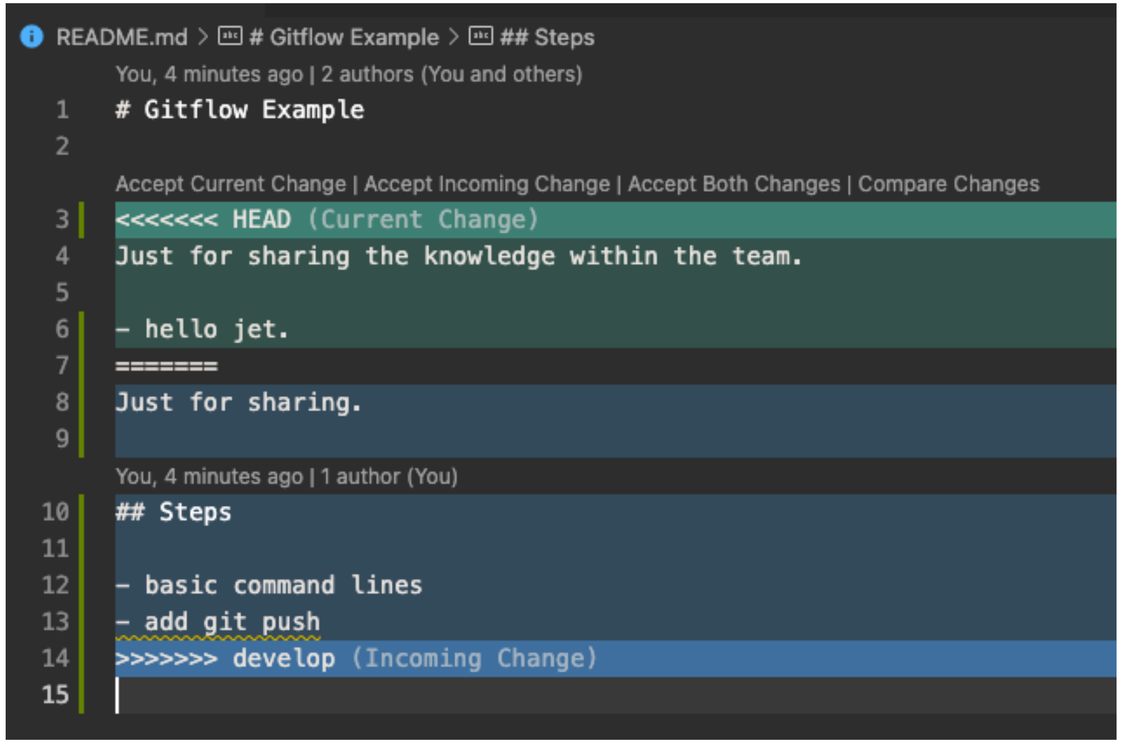
squash commits 的用途是在你的分支做了很多碎小的没意义commits时，你可以squash。(比如，加了行代码，又去掉了，又加回来了，又去掉了……)

解决冲突流程

- 搜索所有的 “<<<<<<<<<“ 就能找到所有的冲突的地方。
- 需要对本身修改的代码有所了解才能修复冲突。
- 修复完后最好要跑一下测试。
- 如果发现不明白的地方最好找一下那个分支的作者，一些商讨一下。



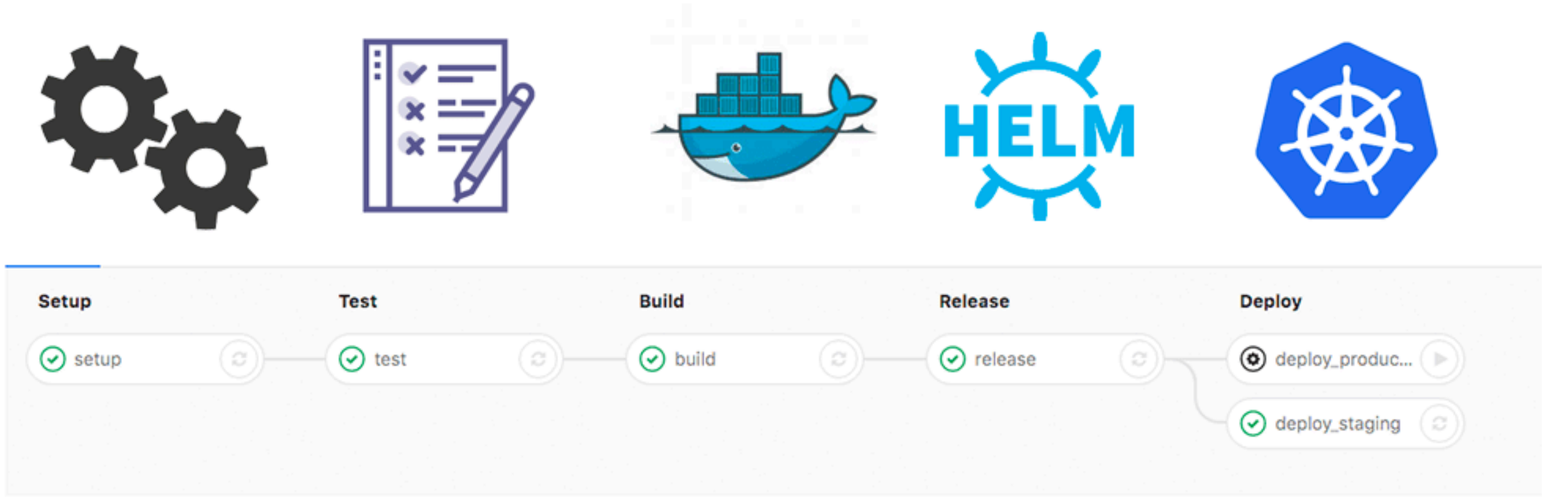
git mergetool



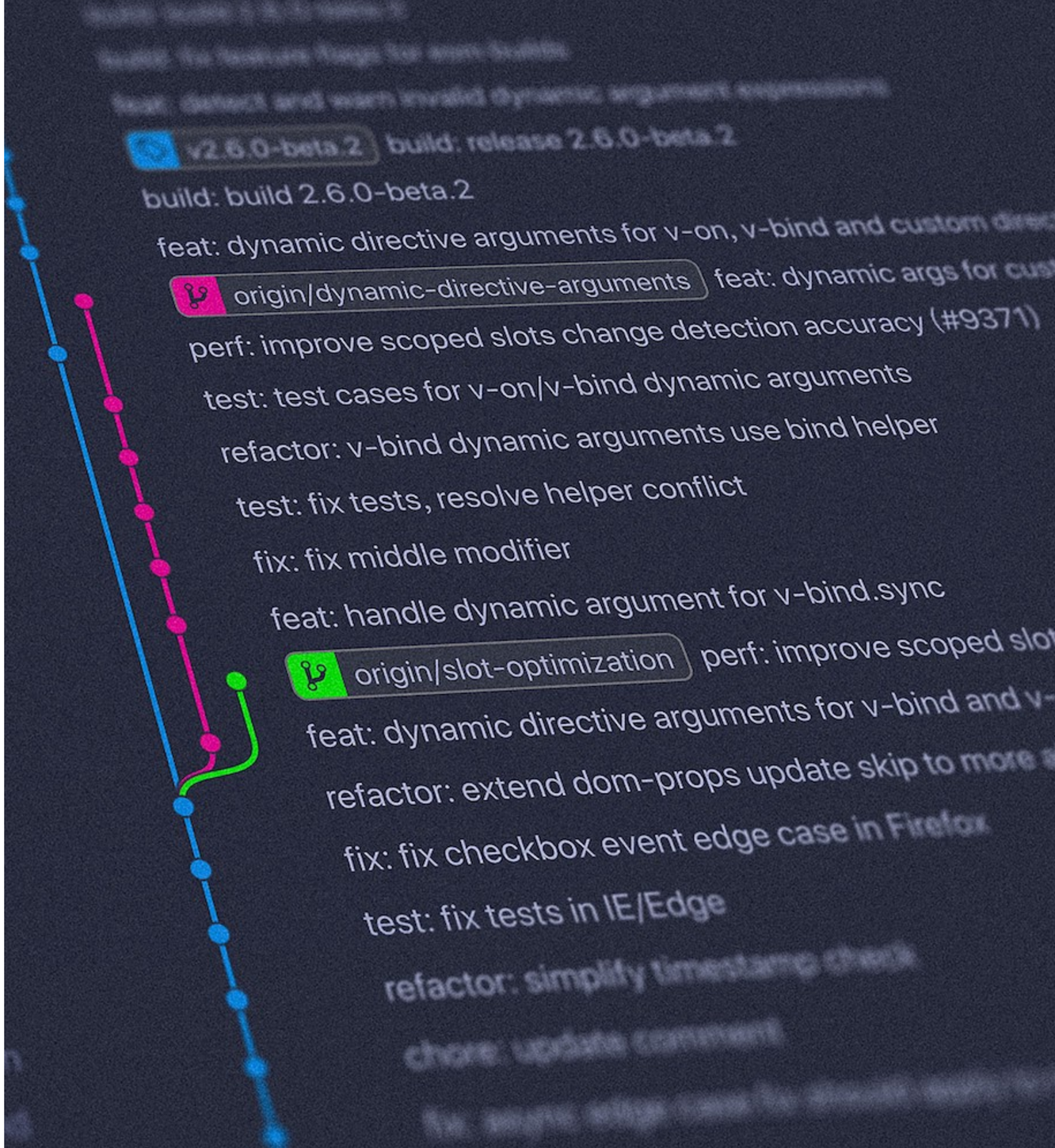
Vscode

如何发布

- 一般会通过 Gitlab CI/CD 进行测试，打包，并且发布。
- 在 gitlab CI 配置文件里面可以设置 当检测到新tag的时候，触发部署到production流程。
- 整个Pipeline的配置和设计是另外一个分享内容。

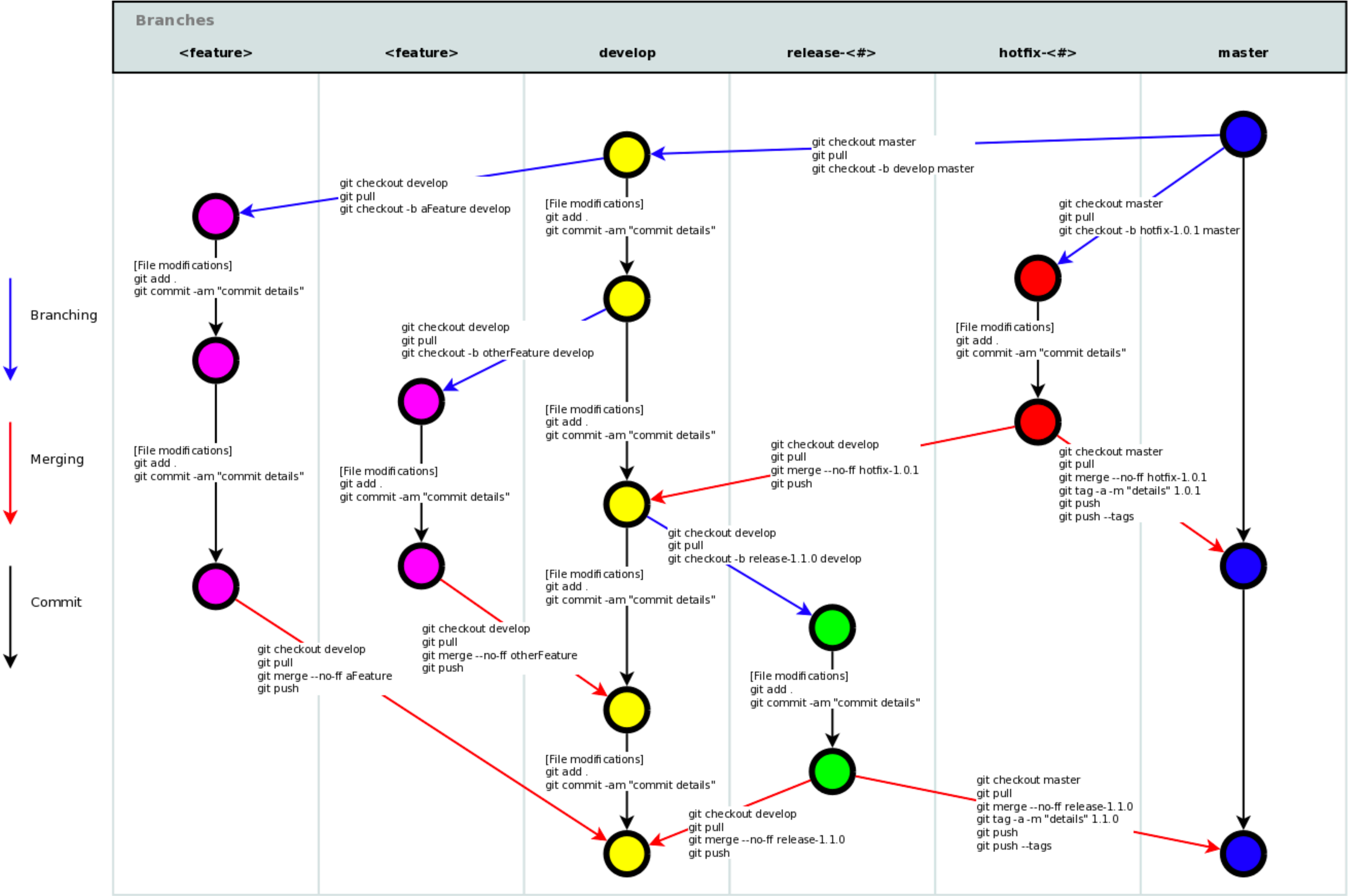


分支管理

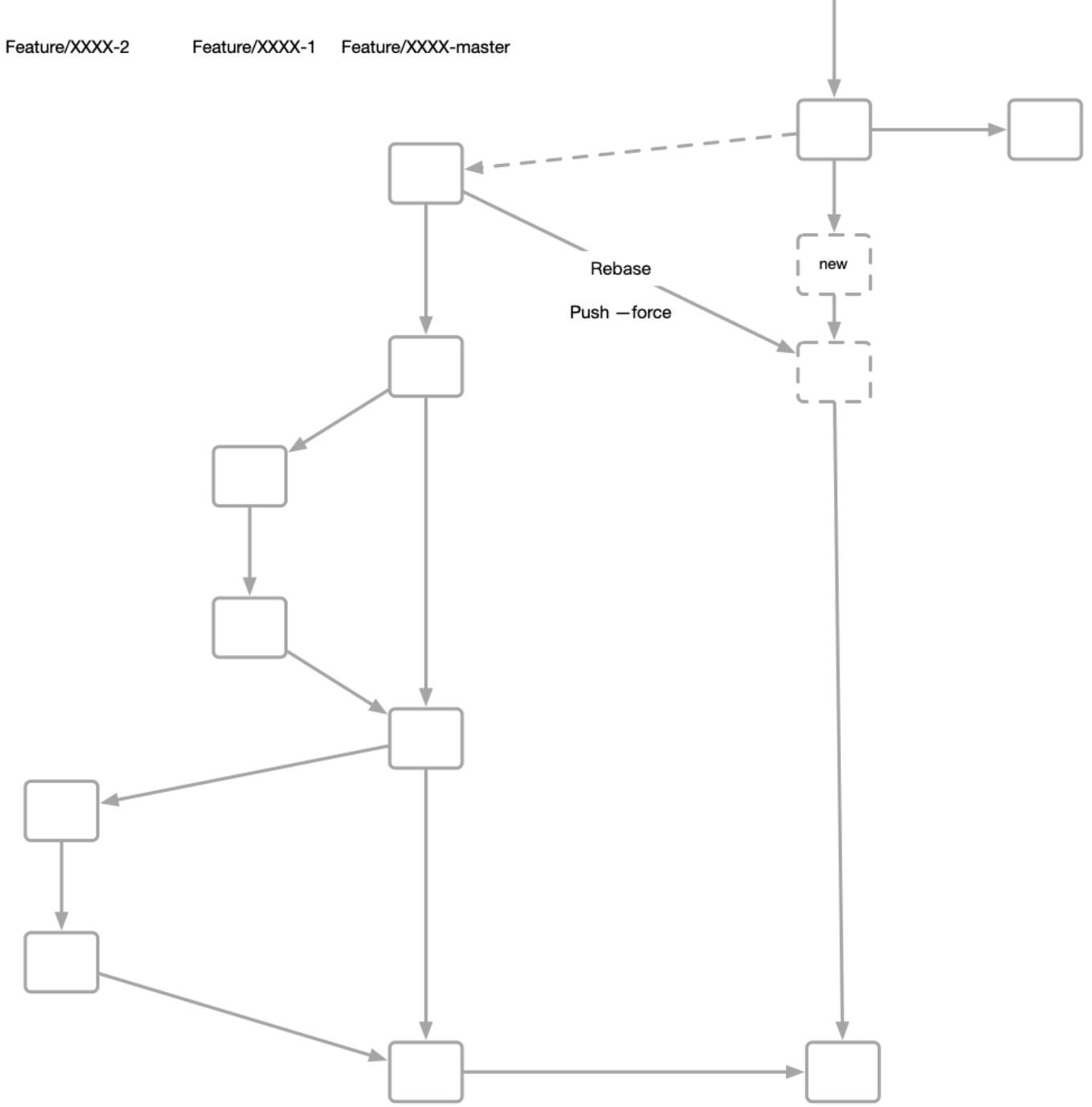

v2.6.0-beta.2 build: release 2.6.0-beta.2
build: build 2.6.0-beta.2
feat: dynamic directive arguments for v-on, v-bind and custom
origin/dynamic-directive-arguments feat: dynamic args for
perf: improve scoped slots change detection accuracy (#9371)
test: test cases for v-on/v-bind dynamic arguments
refactor: v-bind dynamic arguments use bind helper
test: fix tests, resolve helper conflict
fix: fix middle modifier
feat: handle dynamic argument for v-bind.sync
origin/slot-optimization perf: improve scoped
feat: dynamic directive arguments for v-bind and
refactor: extend dom-props update skip to
fix: fix checkbox event edge case in Firefox
test: fix tests in IE/Edge
refactor: simplify
chore: update

分支类型

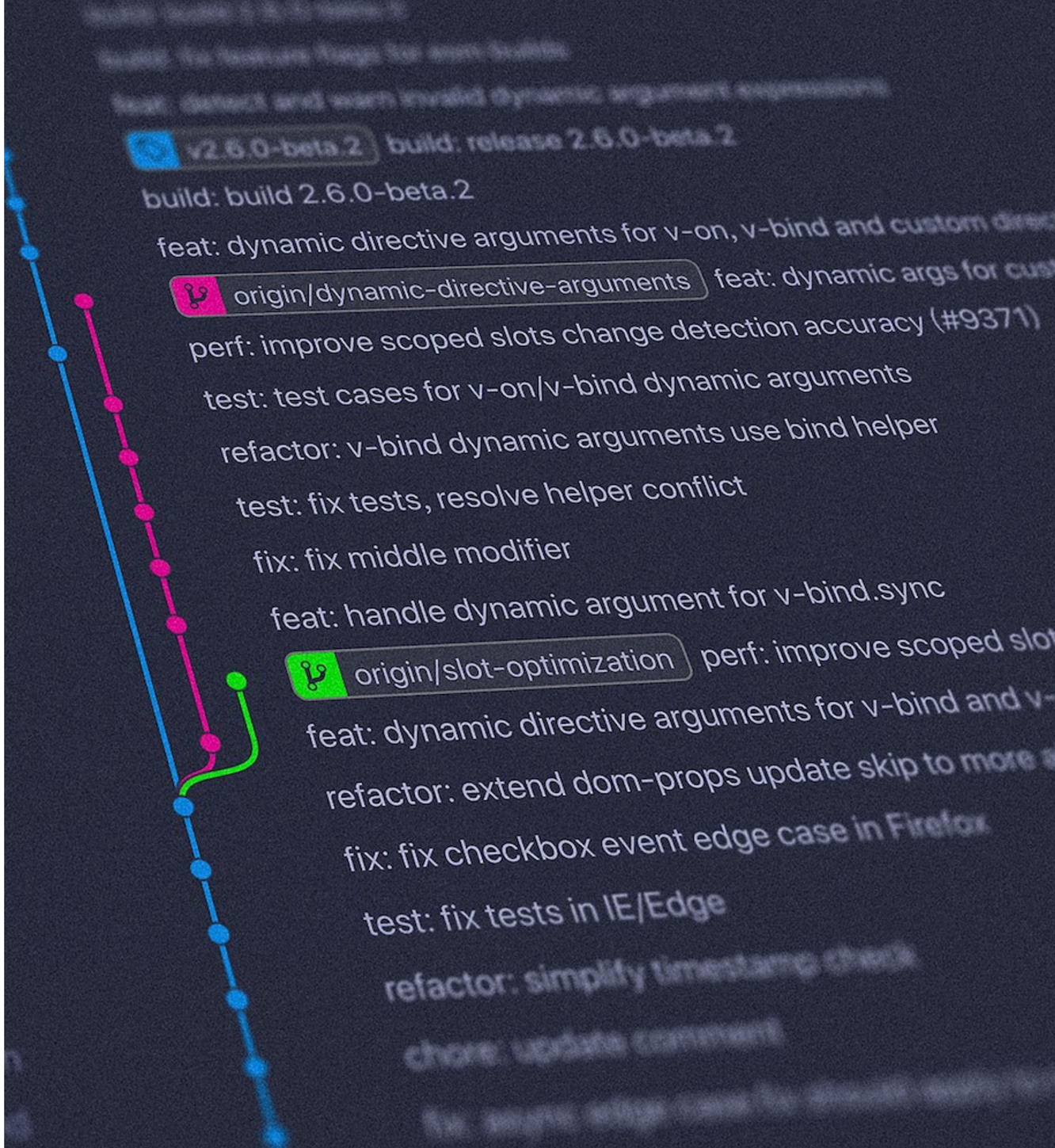
- 长期分支
  - Develop
  - Master
- 任务分支
  - Feature
  - Release
  - Hotfix


Branches
<feature>
<feature>
develop
release-<#>
hotfix-<#>
master
Branching
Merging
Commit
git checkout master
git pull
git checkout -b develop master
git checkout develop
git pull
git checkout -b aFeature develop
[File modifications]
git add .
git commit -am "commit details"
git checkout master
git pull
git checkout -b hotfix-1.0.1 master
git checkout develop
git pull
git checkout -b otherFeature develop
[File modifications]
git add .
git commit -am "commit details"
git checkout develop
git pull
git merge --no-ff hotfix-1.0.1
git push
git checkout master
git pull
git merge --no-ff hotfix-1.0.1
git tag -a -m "details" 1.0.1
git push
git push --tags
git checkout develop
git pull
git checkout -b release-1.1.0 develop
git checkout develop
git pull
git merge --no-ff otherFeature
git push
git checkout develop
git pull
git merge --no-ff aFeature
git push
git checkout develop
git pull
git merge --no-ff release-1.1.0
git push
git checkout master
git pull
git merge --no-ff release-1.1.0
git tag -a -m "details" 1.1.0
git push
git push --tags

- 当多人在同一个feature上协作的时候，可以考虑在feature分支上面再开分支。
- 在合并到 develop 分支之前，需要将所有的子分支合并后再合并到 develop 分支。

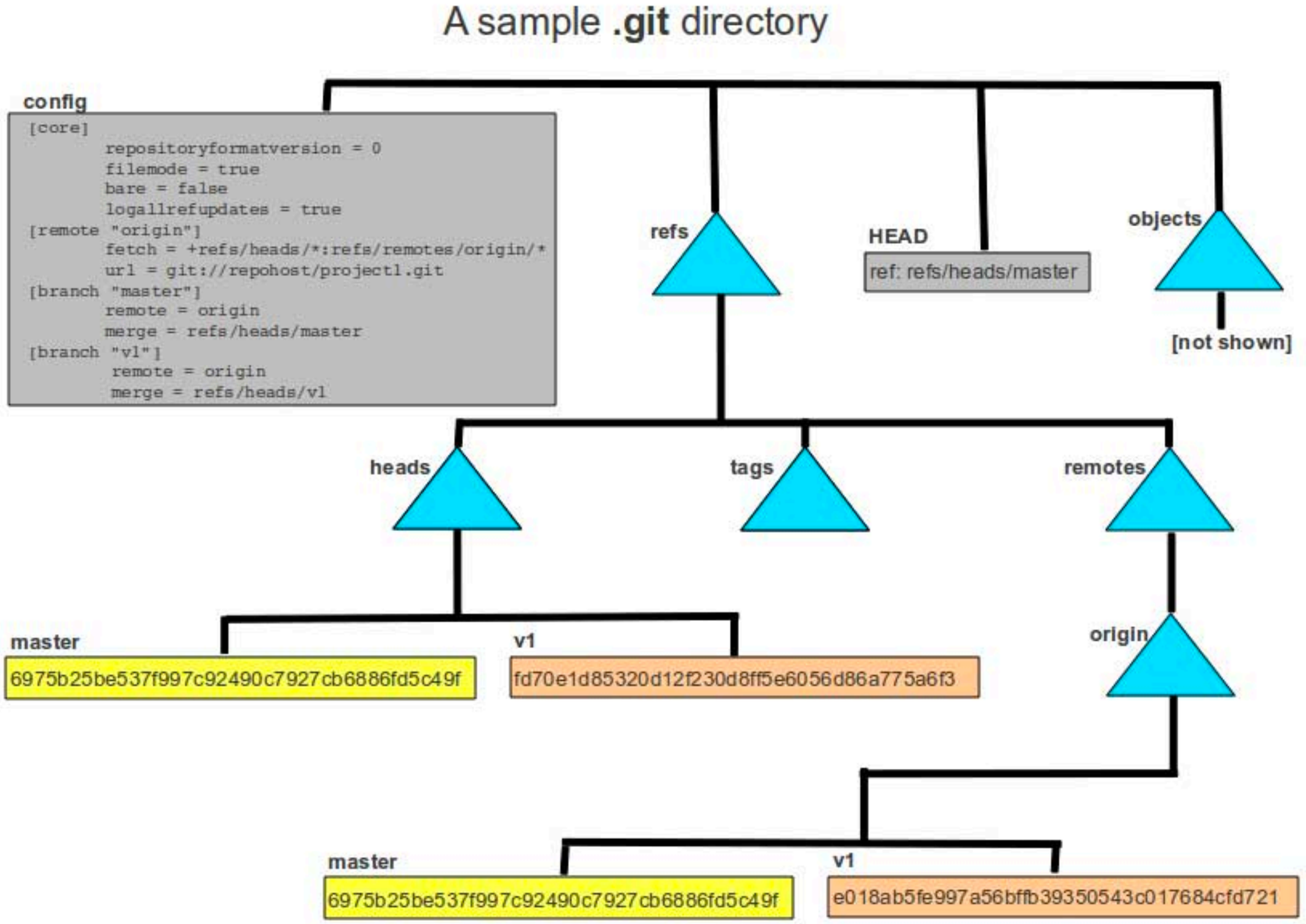


深入 .git 目录


v2.6.0-beta.2
build: release 2.6.0-beta.2
build: build 2.6.0-beta.2
feat: dynamic directive arguments for v-on, v-bind and custom
origin/dynamic-directive-arguments
feat: dynamic args for
perf: improve scoped slots change detection accuracy (#9371)
test: test cases for v-on/v-bind dynamic arguments
refactor: v-bind dynamic arguments use bind helper
test: fix tests, resolve helper conflict
fix: fix middle modifier
feat: handle dynamic argument for v-bind.sync
origin/slot-optimization
perf: improve scoped
feat: dynamic directive arguments for v-bind and
refactor: extend dom-props update skip to
fix: fix checkbox event edge case in Firefox
test: fix tests in IE/Edge
refactor: simplify
chore: update

文件结构

- .git
  - COMMIT_EDITMSG
  - FETCH_HEAD
  - HEAD
  - ORIG_HEAD
  - config
  - description
  - index
  - packed-refs
  - **hooks/** (高级功能)
    https://segmentfault.com/a/1190000022970270
  - **info/**
  - **logs/**
  - **objects/**
  - **refs/**
- 听说还能恢复出 git clean 删除的文件呢
  https://www.codenong.com/46257929/

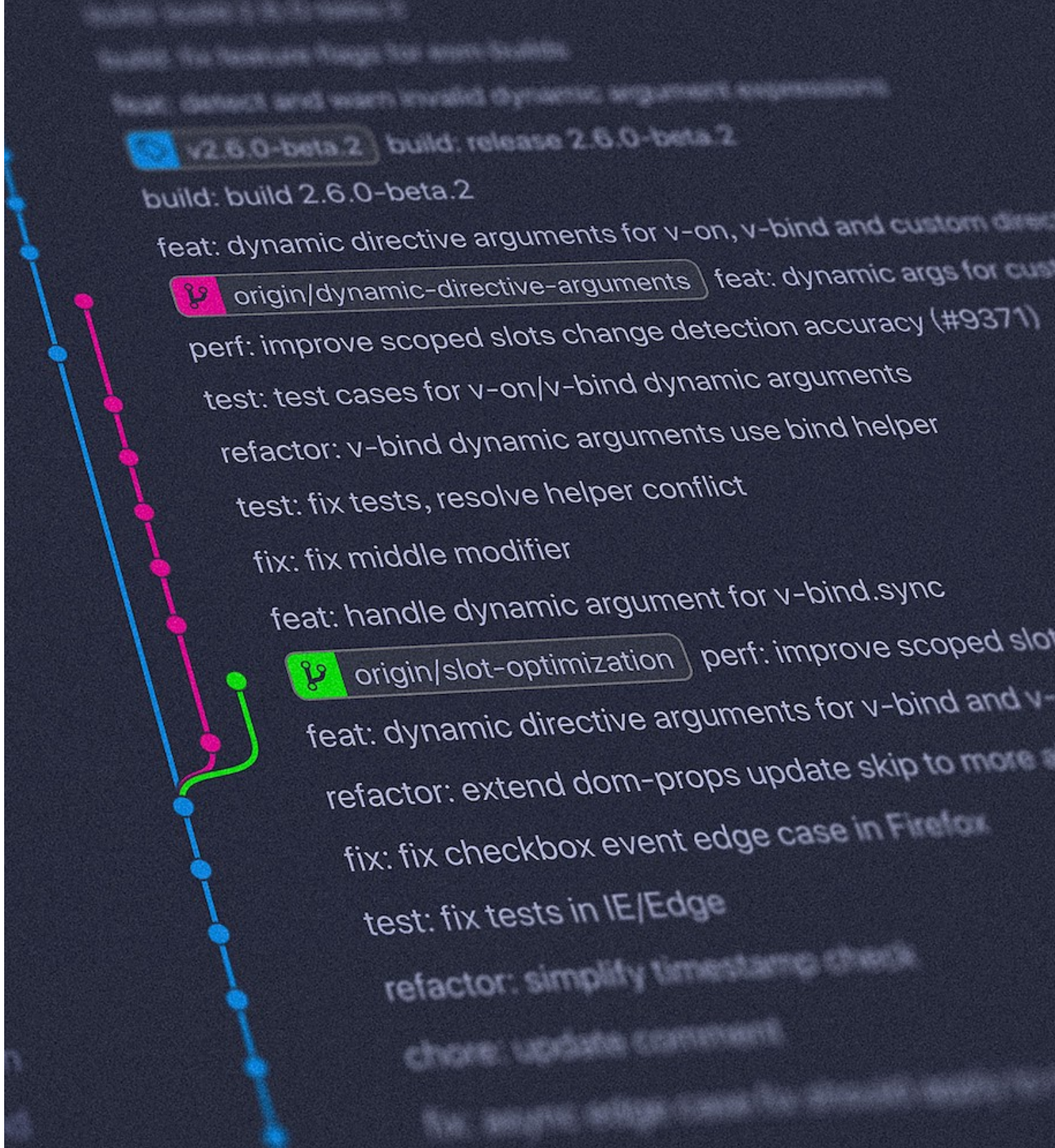


Bare git repo

- HEAD
- config
- description
- *COMMIT_EDITMSG*
- *FETCH_HEAD*
- *ORIG_HEAD*
- *index*
- *packed-refs*
- **Hooks/**
- **info/**
- **logs/**
- **objects/**
- **refs/**

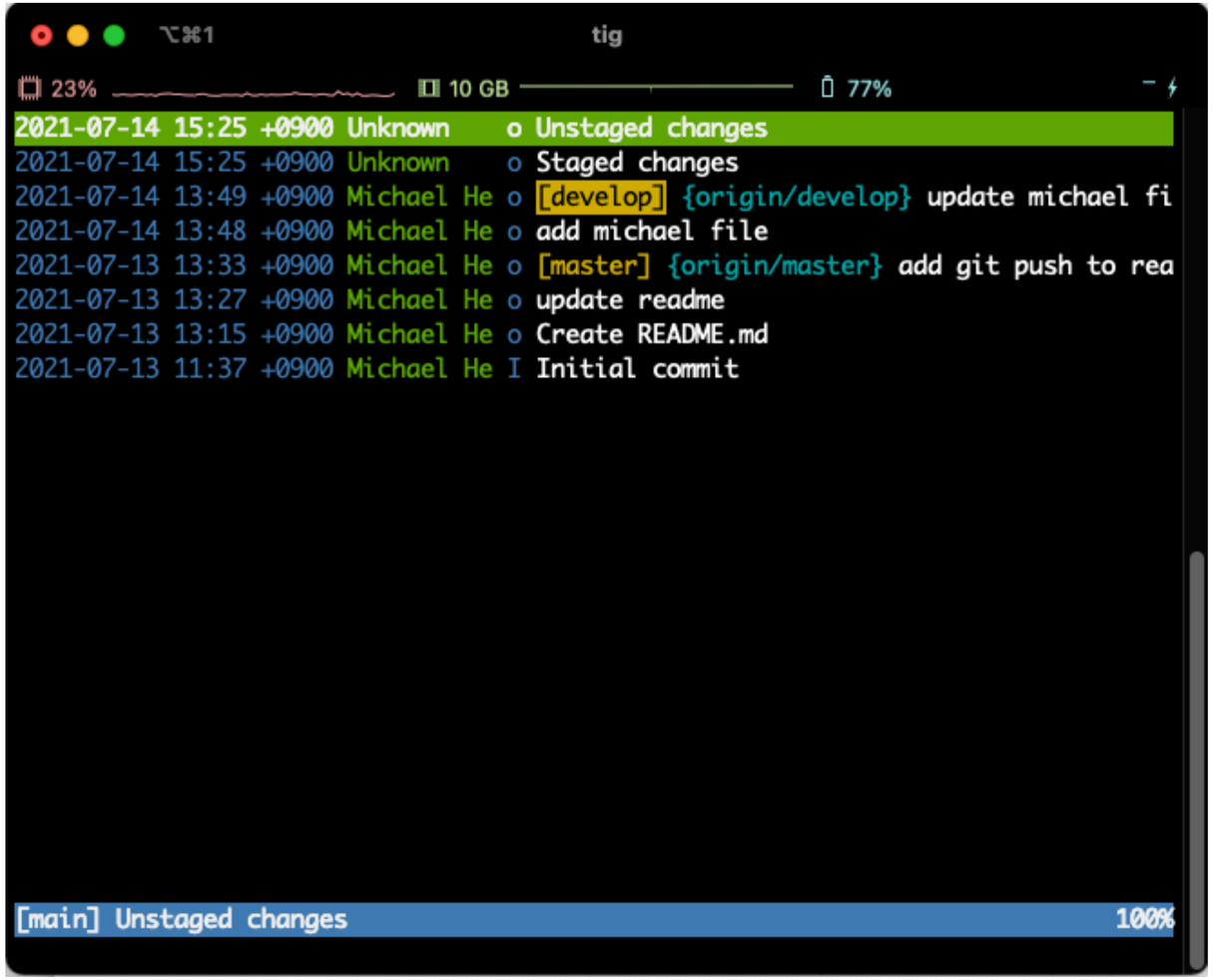
- Bare git repo
  通过 git init ––bare . 创建。
- 它的没有working directory，以及checkout出来的文件。
- 它的用途是作为分享库，而不是工作库用的。也就是说你可以通过一个 ssh 服务器去自己搭建一个git repo代替使用gitlab。
- 你可以通过下面的命令 clone
  git clone ubuntu@127.0.0.1:/path/to/bare-repo

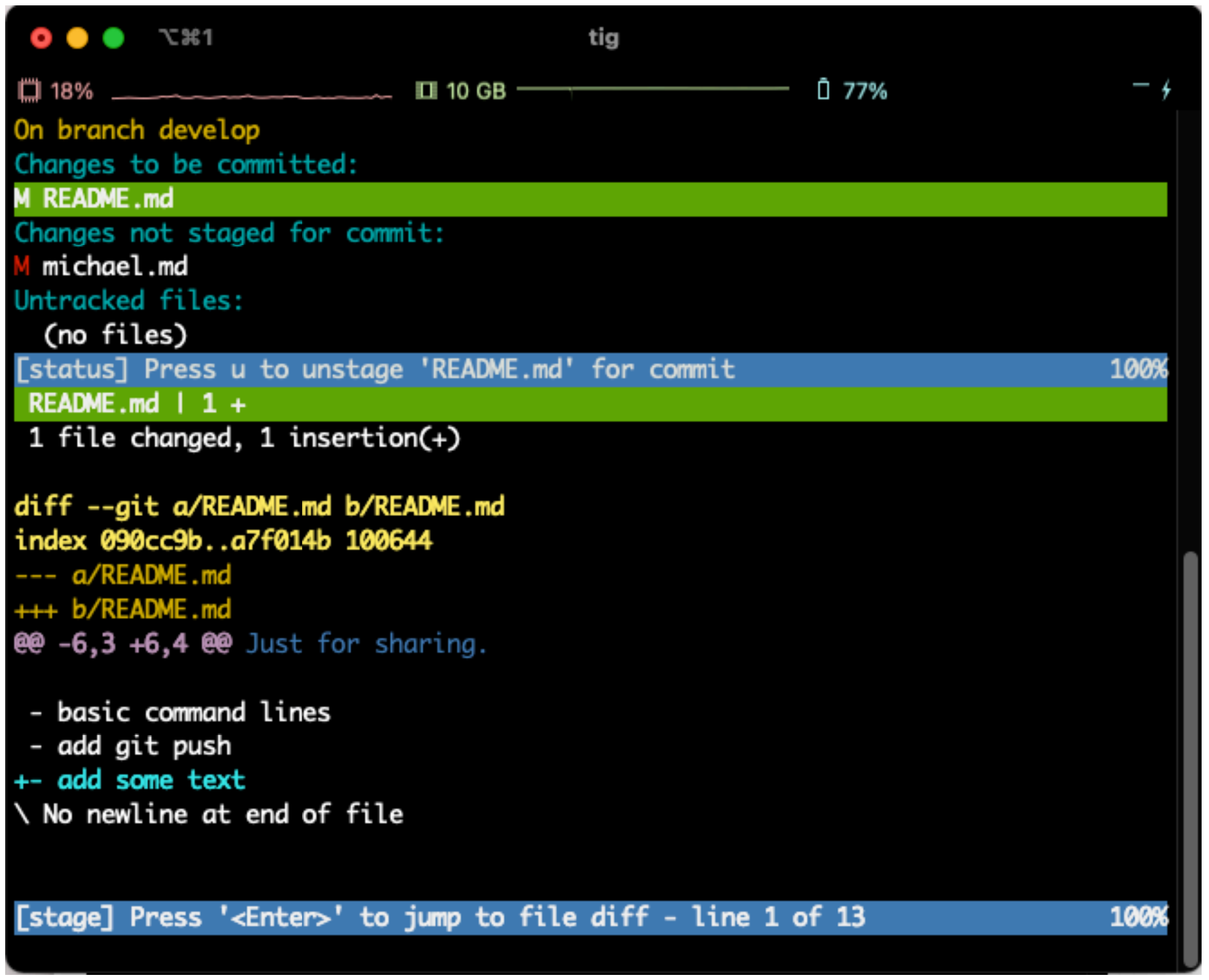
推荐工具


v2.6.0-beta.2
build: release 2.6.0-beta.2
build: build 2.6.0-beta.2
feat: dynamic directive arguments for v-on, v-bind and custom
origin/dynamic-directive-arguments
feat: dynamic args for
perf: improve scoped slots change detection accuracy (#9371)
test: test cases for v-on/v-bind dynamic arguments
refactor: v-bind dynamic arguments use bind helper
test: fix tests, resolve helper conflict
fix: fix middle modifier
feat: handle dynamic argument for v-bind.sync
origin/slot-optimization
perf: improve scoped
feat: dynamic directive arguments for v-bind and
refactor: extend dom-props update skip to
fix: fix checkbox event edge case in Firefox
test: fix tests in IE/Edge
refactor: simplify
chore: update

Tig

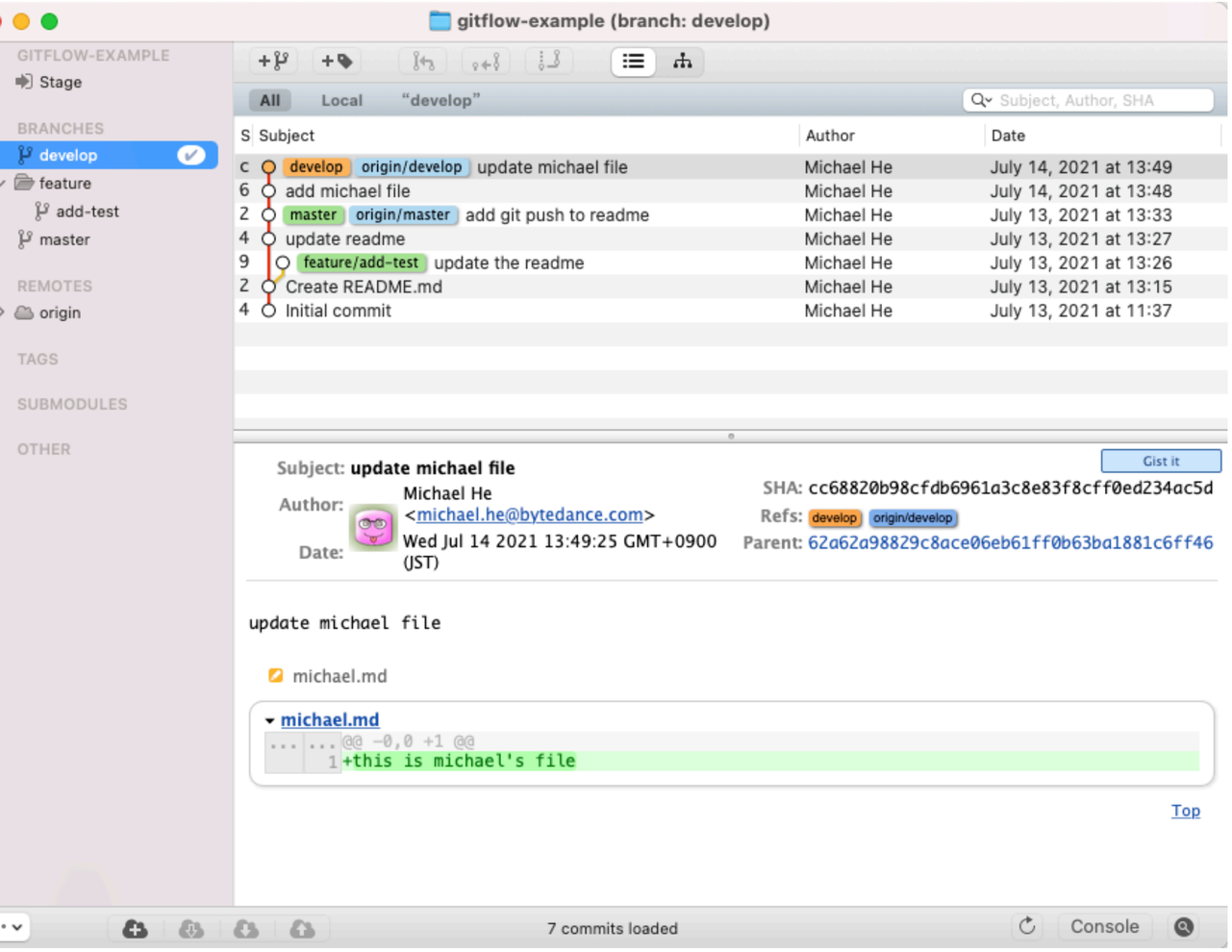
Command Line Git management tool

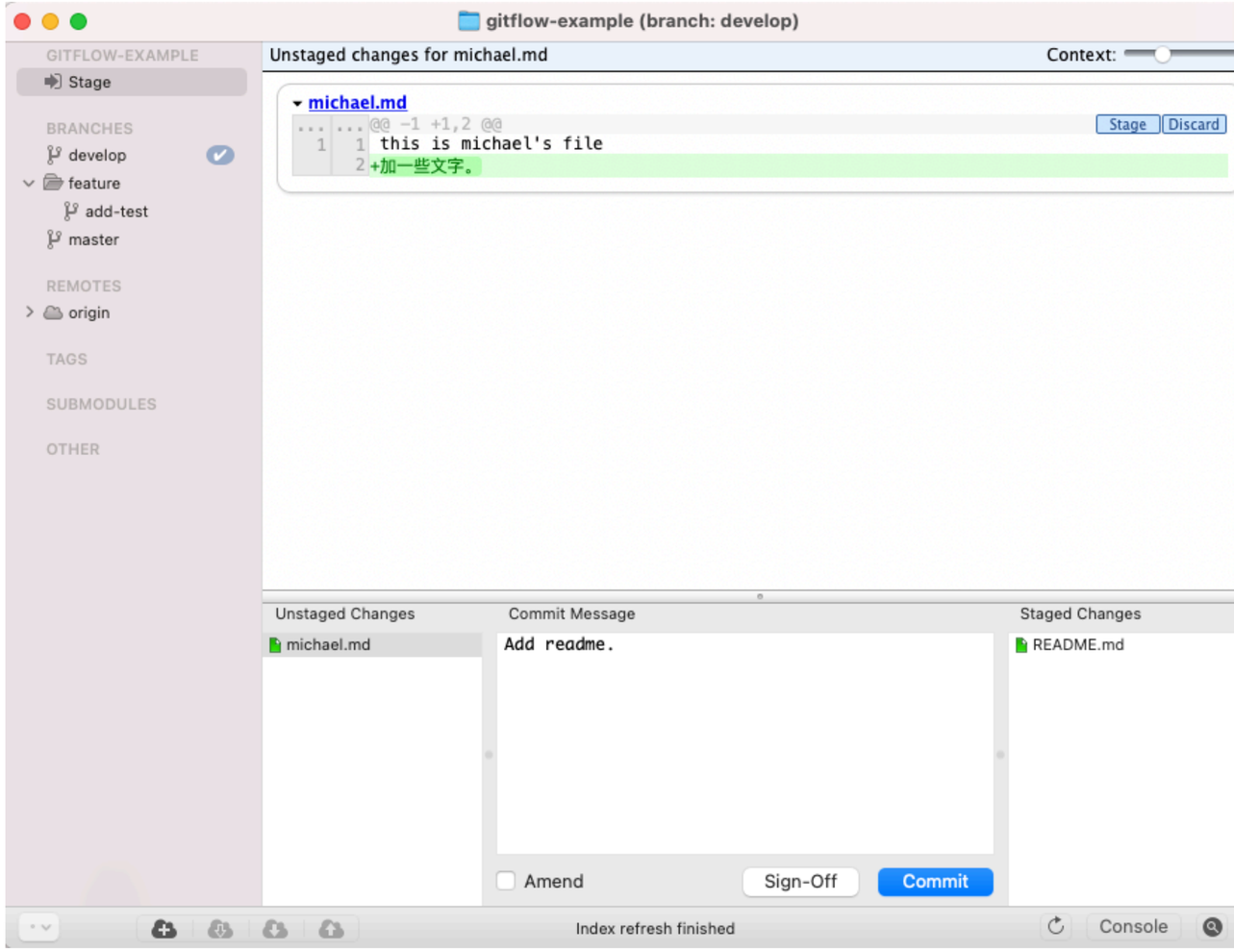




Gitx

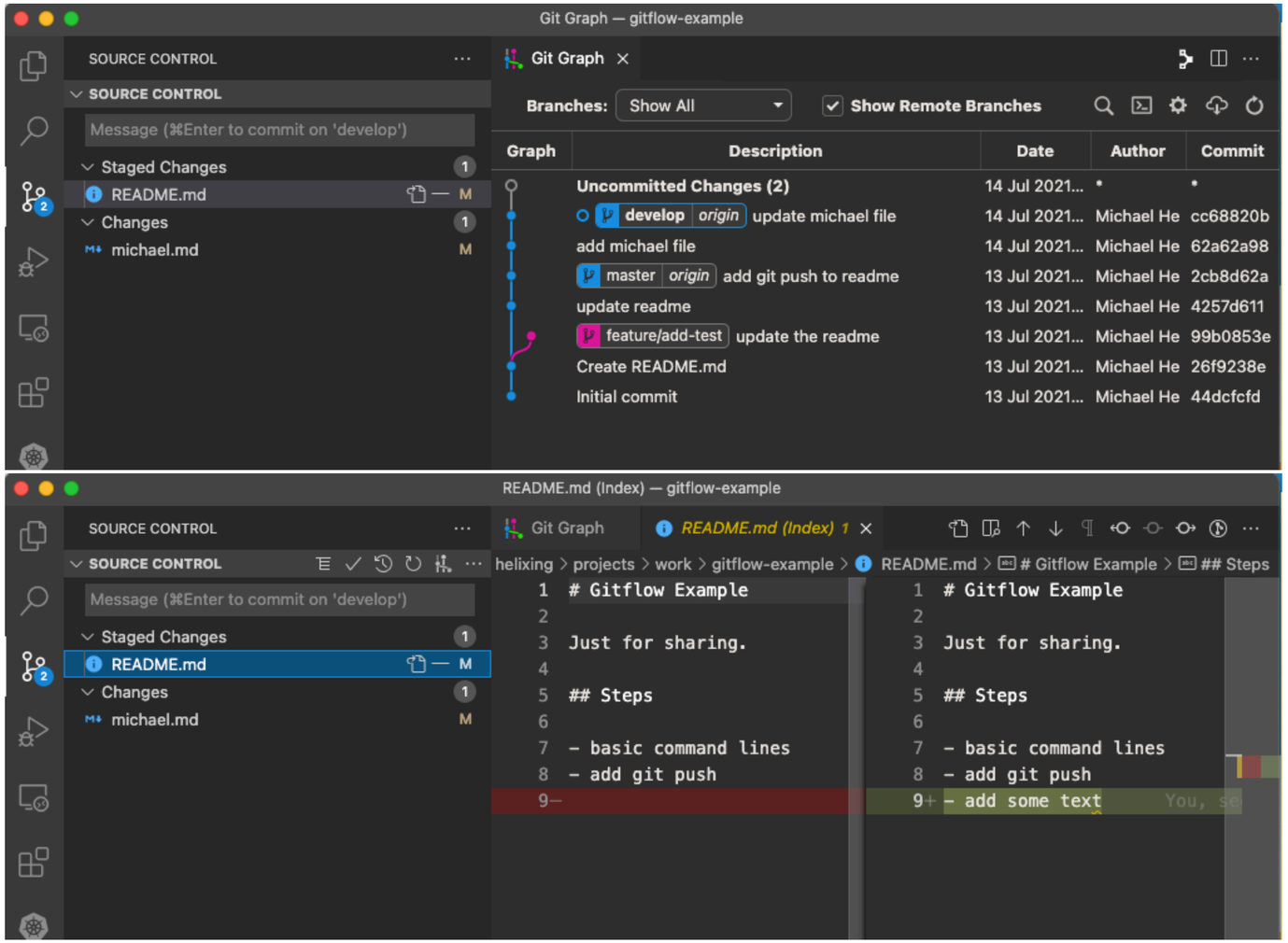
Free GUI Git management tool





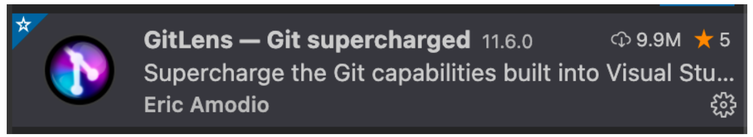
Vscode Plugin

GUI Git management tool



Vscode Plugin

GitLens - Inline Git blame



Q&A