

# Gitlab Feature Branch Workflow

罗流毅

xluoly@gmail.com

2018-7-26

# Agenda

1 Workflow

2 Git Tips

3 提交规范

# Workflow Overview

- 基本原则: **master** 为保护分支, 不直接在 **master** 上进行代码修改和提交。
- 日常开发, 从 **master** 分支上 **checkout** 一个 **feature** 分支进行开发, 检查通过后, 将 **feature** 分支合并到 **master**, 最后删除 **feature** 分支。

# Workflow Overview



# 克隆仓库

```
$ git clone git@10.8.2.38:pmo/test.git
```

## ● 基于 **master** 分支最新的代码创建工作分支

```
$ git checkout master  
$ git pull --rebase  
$ git checkout -b bugfix_xxx  
$ git checkout -b feature_xxx  
$ git checkout -b refactor_xxx
```

## ● 提交前检查

```
$ git status  
$ git diff  
$ git add file1 file2  
$ git commit  
$ git commit --fixup=commit-id
```

# 整理提交

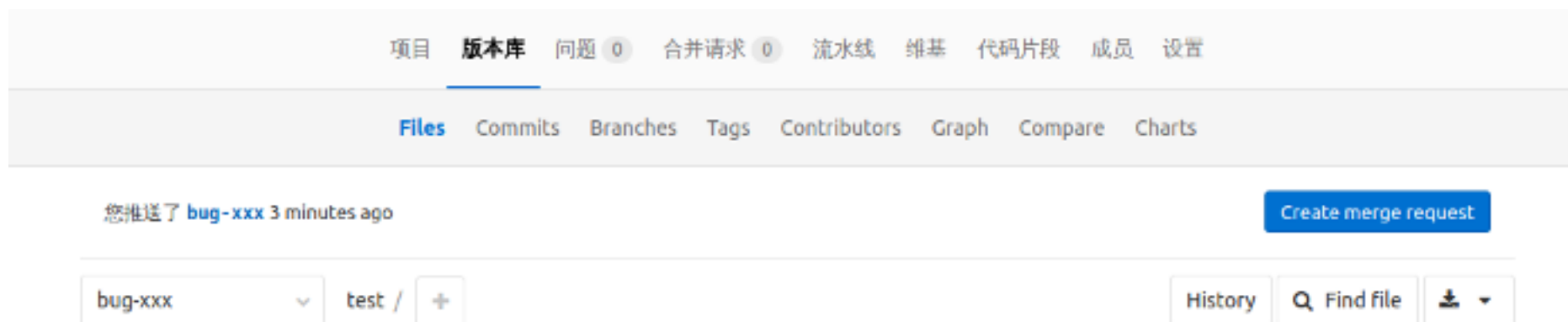
```
$ git log  
$ git show --stat commit-id  
$ git show commit-id  
$ git commit --amend  
$ git rebase -i commit-id^
```



# 推送工作分支

```
$ git fetch origin  
$ git rebase origin/master  
$ git push origin bugfix_xxx  
$ git push origin feature_xxx  
$ git push origin refactor_xxx
```

# 创建合并请求



# 创建合并请求

## 新建合并请求

从 `bug-xxx` 合并到 `master`


[改变分支](#)

标题

从标题中删除 **WIP:** 前缀 表示允许此标识为 **正在处理中** 的合并请求已准备好可以被接受。

增加 [描述模板](#) 以帮助您的贡献者们更有效地沟通！

描述 编辑 预览

**B** *I*      

撰写评论或拖放文件到此处...

[Markdown 及 quick actions 格式](#)

 [添加附件](#)

指派  [指派给自己](#)

里程碑

标记

源分支

目标分支  [修改分支](#)

☒ 接受合并请求后删除来源分支。

[提交 合并请求](#)

[取消](#)

# 代码审查

## 修改xxx，解决yyy问题

parent 33ae7f03 | bug-xxx

正在显示 1 个修改的文件 ▾ 包含 1 行增加 和 0 行删除

隐藏空白字符变更

内嵌

并排

DominickZhang.txt

查看文件 @ 41bca6bb

1	1	ac
	2	+ google.com



编写 预览

B I

撰写评论或拖放文件到此处...

Markdown 格式

添加附件

评论



# 执行合并

[项目](#) [版本库](#) [问题 0](#) [合并请求 1](#) [流水线](#) [维基](#) [代码片段](#) [成员](#) [设置](#)

未关闭

合并请求 13 在 4 minutes ago 由  liuyi.luo(罗流毅)

编辑

关闭 合并请求

## Bug xxx

编辑时间 2 minutes ago 作者 liuyi.luo(罗流毅)

Request to merge `bug-xxx` into `master`

Check out branch



 Merge

☒ Remove source branch

Modify commit message


You can merge this merge request manually using the [command line](#)

 0

 0



[讨论 0](#) [提交 1](#) [变更 1](#)

 liuyi.luo(罗流毅) @liuyi.luo 将合并请求重新标记为 进行中(WIP) 2 minutes ago



编写 预览

B I       

撰写评论或拖放文件到此处...

# Agenda

1 Workflow

2 Git Tips

3 提交规范

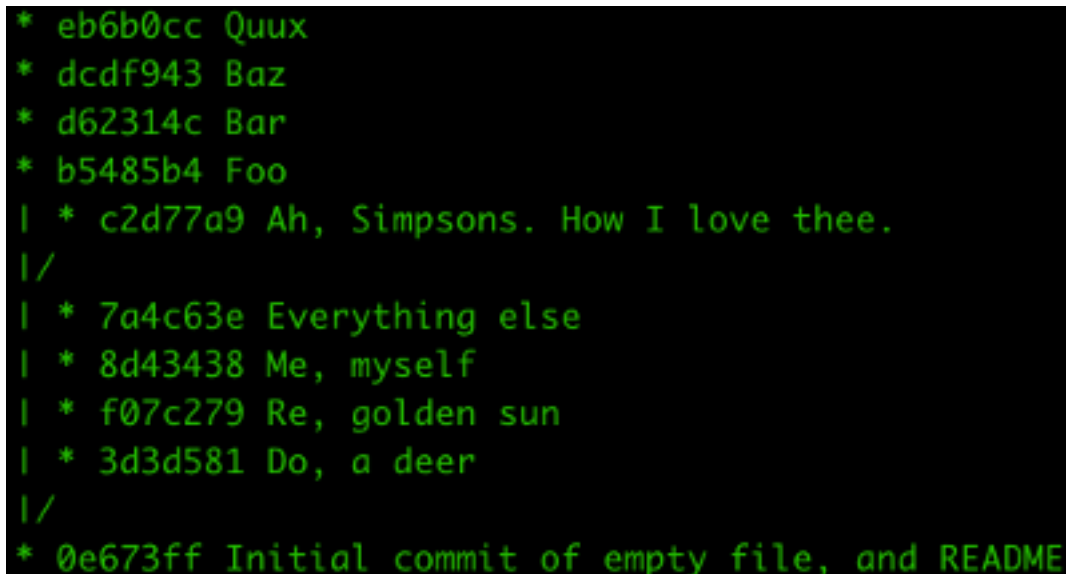
# Git 集成 Beyond Compare

- 安装 Beyond Compare4
- 配置 Git (假设 BC 安装路径为 C:\Program Files\Beyond Compare 4)

```
$ git config --global diff.tool bc4
$ git config --global difftool.prompt false
$ git config --global difftool.bc4.cmd '"C:\Program Files\Beyond Compare 4\BCompare4.exe" %A %B'
$
$ git config --global merge.tool bc4
$ git config --global mergetool.prompt false
$ git config --global mergetool.bc4.cmd '"C:\Program Files\Beyond Compare 4\BCompare4.exe" %A %B'
$ git config --global mergetool.bc4.trustexitcode true
```

# 提交日志图形化

```
$ git log --graph --decorate --oneline
$ git log --graph --decorate --oneline --all
```



```
* eb6b0cc Quux
* dcd9f43 Baz
* d62314c Bar
* b5485b4 Foo
| * c2d77a9 Ah, Simpsons. How I love thee.
|/
| * 7a4c63e Everything else
| * 8d43438 Me, myself
| * f07c279 Re, golden sun
| * 3d3d581 Do, a deer
|/
* 0e673ff Initial commit of empty file, and README
```

- 在 `~/.gitconfig` 中添加如下设置，然后可以直接使用 `git l` 命令

```
[alias]
    l = log --decorate --graph --pretty=format:'%Cred%h%Creset
-%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset'
--abbrev-commit --date=relative
```



- 修改之前提交的日志
- 已经 **merge** 到远程仓库 **master** 分支的提交不允许修改
- 修改最近一次提交使用如下命令

```
$ git commit --amend # 修改最近一次提交
```

- 将本次提交作为之前提交的补充
- 已经 **merge** 到远程仓库 **master** 分支的提交不允许修改

```
$ git commit --fixup=commit-id # commit-id为补充到的目的SHA  
$ git rebase -i commit-id^ --autosquash
```

# 合并提交

- 将多个提交合并为单个提交
- 已经 **merge** 到远程仓库 **master** 分支的提交不允许修改
- 执行如下命令后，将需要合并的提交集中放置在一起，将 **pick** 改为 **squash** 或 **s**

```
$ git rebase -i commit-id^ # commit-id为需要合并的最早一次提交的SHA
```

# Agenda

- 1 Workflow
- 2 Git Tips
- 3 提交规范

# 提交的格式

- **Header:** commit 的标题，不能太长，避免自动换行，必需的，不可省略
- **Body:** 是对本次 commit 的详细描述，可以分成多行，可以省略
- **Footer:** 附加说明，可以省略

```
<type>(<scope>) : <subject>
// 空一行
<body>
// 空一行
<footer>
```

- **type:** 用于说明 **commit** 的类别，必需，不可省略
- **scope:** 用于说明 **commit** 影响的范围，比如数据层、控制层、视图层等等，视项目不同而不同，可省略
- **subject:** **commit** 目的的简短描述，不超过 **50** 个字符，必需，不可省略

# Header -- type

- feat: 新功能 (feature)
- fix: 修补 bug
- docs: 文档 (documentation)
- style: 格式 (不影响代码运行的变动)
- refactor: 重构 (即不是新增功能, 也不是修改 bug 的代码变动)
- test: 增加测试
- chore: 构建过程或辅助工具的变动
- revert: 撤销以前的 commit

# Header -- subject

- 以动词开头，使用第一人称现在时，比如 `change`，而不是 `changed` 或 `changes`
- 第一个字母小写
- 结尾不加句号 (.)



- **Body** 部分是对本次 **commit** 的详细描述，可以分成多行。
- 使用第一人称现在时，比如使用 **change** 而不是 **changed** 或 **changes**。
- 应该说明代码变动的动机，以及与以前行为的对比。

- 如果当前 **commit** 针对某个 **issue**，那么可以在 **Footer** 部分关闭这个 **issue**。

Closes #234

# 一个范例

`docs (document)`：标题50个字符以内，描述主要变更内容

更详细的说明文本，建议72个字符以内。需要描述的信息包括：

- \* 变更的原因，可能是用来修复一个bug，增加一个feature，提升性能、可靠性、稳定性等等
- \* 他如何解决这个问题？具体描述解决问题的步骤
- \* 是否存在副作用、风险？

如果需要的化可以添加一个链接到issue地址或者其它文档，或者关闭某个issue。

# The End

