

使用Git和Gitee (GitHub) 进行协同开发

获得最新master分支

- 本地无仓库

```
git clone https://gitee.com/用户名/仓库名.git
```

- 本地有仓库

```
git checkout master
```

```
git pull origin master
```

注：如果当前工作目录在另一分支X上，那么在pull后会把master上的最新修改自动合并到X上。所以我们要先检出master分支。

查看当前分支历史的工具：git log

```
git log --graph --pretty=oneline
```

```
$ git log --graph --pretty=oneline
* a2d20a26befb4f4a833b145be42bc1fa7c03ab26 (HEAD -> master, origin/master) test.
* 6d1834878cbbed446859f4c8a2a3e1ff530041d7 test.
* fcbc4d9b05c0c84b10f67b52c5bd774459621f82 Merged.
| \
| * 39c1c7336ae5ae0384a51827641a6c15a61c26fb DIRECTLY MODIFIED ON C
* | 4f9cc32be15becab583f5d36e04ddbb64a1211a5 Edited on a local wind
| /
* c686b36bc00e5a955c5dc6a4bc847ca4b4ff251c updated notes.txt
* 5617186ab21d465e8e3dcfb54a54b93631b05dc0 update notes.txt.
* 9b0ea27d93d426cc7019b91cf4d0901e1d504bf8 merged manually.
| \
| * 5abf0cfa952e1ce2dbb4b8707eaab61d05c5d81c update notes.txt.
* | 8a3e845e23193469e370aff87f01ba15624922b0 updated locally.
| /
* c81d76e5617c39944d8b5a4aac208134f6a9c8cd Added a new text file.
```

创建工作分支（特性分支）

- 保持master分支始终正确可用；修改工作在工作分支上进行。
- 创建分支
 - 基于当前分支创建新分支。如果你意图基于master分支创建新分支，那么要确保当前分支是master。检查当前分支名称：

`git branch` 或 `git status`

- 创建

`git branch 新分支名称` （此时工作目录仍为master分支）

`git checkout 新分支名称` （此时工作目录变成新分支）

- 删除分支

`git branch -D 分支名称`

创建调试分支

如果用户报告了软件使用中的一个问题，需要对在用户版本基础上进行调试和修改，那么可以以用户版本为基点创建分支。

- `git checkout 用户版本对应的提交哈希值`（只需要输入哈希值的前几位，不需要输入完整的哈希值，只要唯一即可）
 - 注意：当前工作目录将处于DETACHED HEAD（分离头）状态！
- `git branch bugfix`（以当前检出的提交作为基点创建bugfix分支）
- `git checkout bugfix`（从此不是DETACHED HEAD状态了）
- 在bugfix分支上进行调试和修补。

在工作分支上实现新功能

- 根据需要创建或修改文件（代码、文档等）
- 测试
- 然后写入仓库，并且推送到Gitee：

```
git add .
```

```
git commit -m "我的修改日志。"
```

```
git push origin 当前分支名称
```

（于是Gitee上也有了该分支）

单人项目：直接合并

- 如果是单人项目，无人审核代码，则可以直接合并到master分支：

```
git checkout master
```

```
git merge 要合并的分支名称
```

（将其合并到当前分支即master分支上）

- 无冲突
- 有冲突（可能在多台电脑上进行过提交）

- 手工解决冲突，或借用图形化工具：

```
git mergetool
```

- ```
git add .
```

- ```
git commit
```

```
git push origin master
```

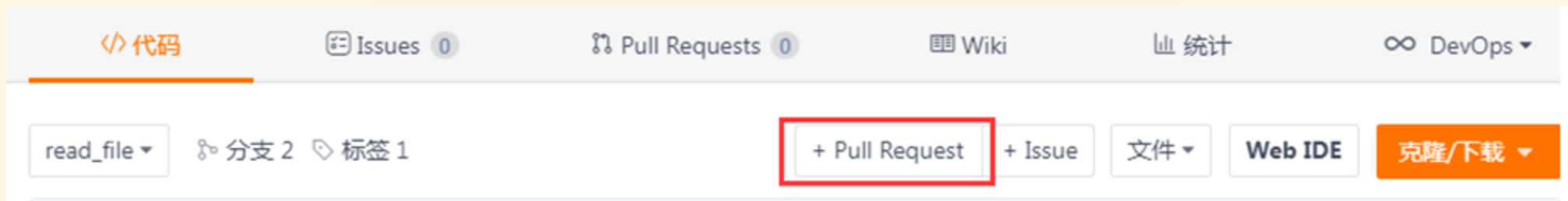
（将master分支的最新修改同步到Gitee）

团队项目：通过Gitee上的Pull Request (PR)合并分支

虽然团队成员都可以通过上述直接合并的方式合并分支，但对于团队项目，这样做比较容易引起冲突。建议指定一位成员（仓库维护者）负责合并操作，其他成员通过PR发起合并请求。

- 设置：管理 → 仓库设置 → 代码审查设置
- 创建Pull Request
- 审核Pull Request

创建Pull Request



- 在创建页面，选择正确的分支。
- 在说明框里可以@代码审核者。

审核Pull Request

- 审核者会收到Pull Request
- 可以fetch后在本地测试
- 可以直接在Gitee的文件修改记录中插入评论
- 如果源分支和目标分支（ master ）没有冲突，可在Gitee上自动合并

如果PR不可自动合并

- Pull Request提交者本地修改后再push到他的工作分支

- Pull Request会自动更新

- 仓库维护者在本地按以下步骤手动合并

```
git checkout master
```

```
git pull origin PR分支
```

（注：每个PR有其特定的分支名称）

- 如有冲突，按上面提到的冲突解决方法解决

```
git push origin master
```

简易团队 workflow 总结

1. 更新本地仓库 (clone 或 pull)
2. 从master分支创建工作分支
3. 在工作分支上完成开发工作
4. 提交Pull Request
5. 团队成员审核、测试Pull Request
6. 仓库维护者合并Pull Request到master分支
7. 所有团队成员更新本地仓库的master分支

合并基于旧master的工作分支

在master分支更新后，基于旧master的工作分支如何合并到新master？

- 更新本地的master分支
- 新master合并到工作分支（注意方向）
 - 手动解决冲突
- commit、push工作分支
- 提交Pull Request

标签（里程碑版本的名称）

- 创建标签：`git tag -a v1.0 -m "这是第一个正式版本。"`
 - 你可以像使用提交ID一样使用标签，比如：`git checkout v1.0`
（注意：DETACHED HEAD状态！）
- 列出所有标签：`git tag`
- 显示标签信息：`git show v1.0`
- 删除标签：`git tag -d v1.0`
- 推送标签到Gitee：`git push --tags`