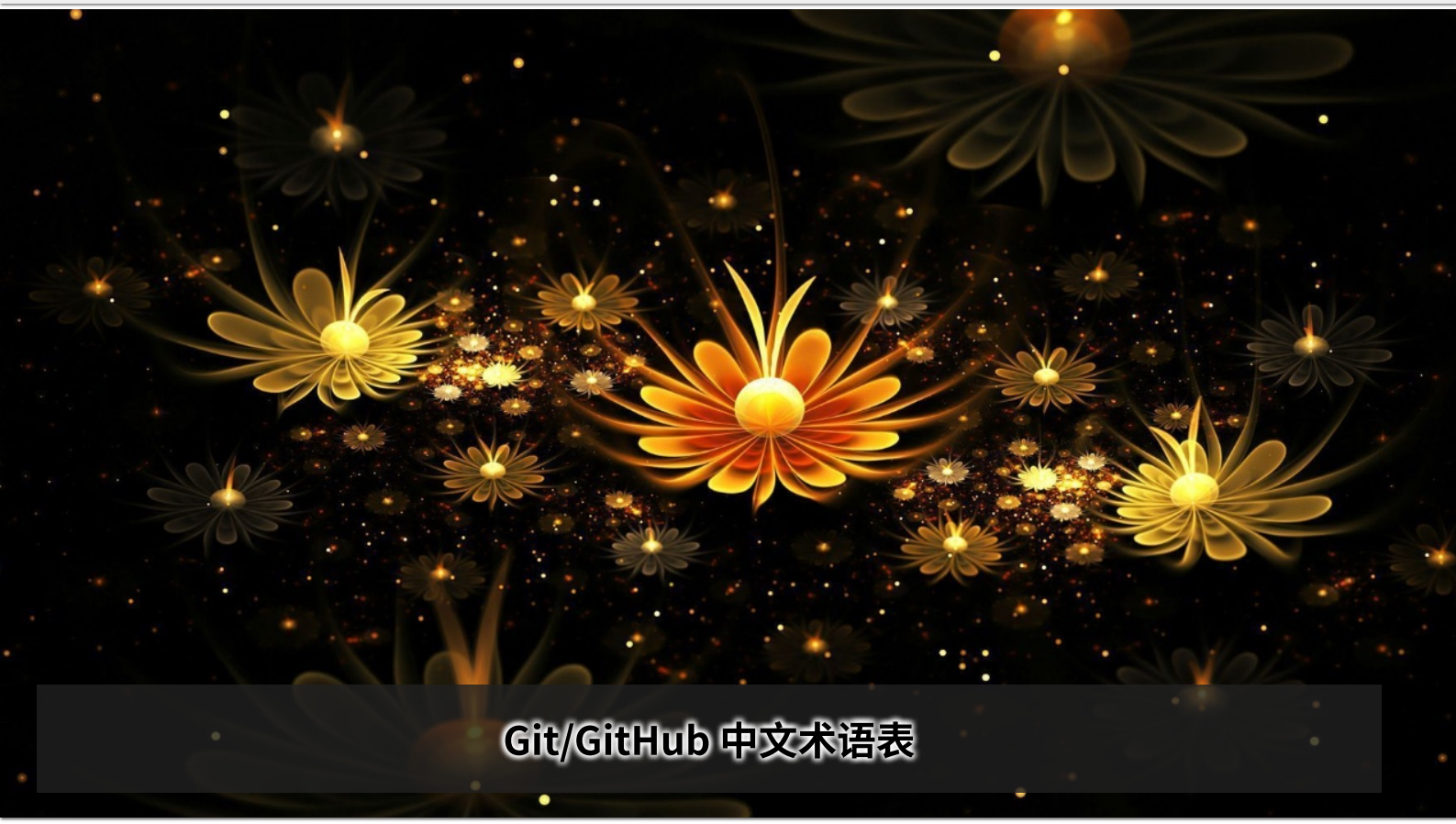

Git/GitHub 中文术语表

作者： Linux中国 硬核老王 | 2020-05-24 00:20 评论: 2 收藏: 2

Git 和 GitHub 已经成为了开发者的基础工具，尤其是参与开源软件开发时经常会使用它们。但是在 Git 和 GitHub 使用过程中遇到的很多术语并没有标准的或约定俗成的中文译名，因此，我们根据 GitHub 、 Git 等文档，并结合我们的翻译惯例，收集整理了 Git 和 GitHub 中常用术语的中文译名及其解释。

这里值得注意是术语有复刻、挂钩、议题、星标、变基、仓库等，这些术语之前要么经常中英文混杂使用，要么中文译法不确定，我们根据多年的翻译和开发经验，在 GitHub 译法的基础上进行斟酌，整理了如下的术语表供大家使用参考。此外，“复刻”这个翻译应该是我们 LCTT 首倡 的；而“议题”这个对 issue 的译法也比之前的一些其它译法更为精准；“仓库”一词还有存储库、版本库等译法，但是仓库一词似乎更加合适。

(assignee) 受 让 人

分配到某个议题的用户。

(blame) 追 溯

Git 中的“追溯”功能描述对文件每行的最新修改，一般会显示修订、作者和时间。这很有用，例如，可以跟踪何时添加了功能，或者哪个提交导致了特定漏洞。

(branch) 分 支

分支是仓库的平行版本。它包含在仓库中，但不影响主要或 master 分支，可让你自由工作而不中断“即时”版本。在执行所需的更改后，可以将分支合并回 master 分支以发布更改。

(checkout) 检 出

你可以在命令行上使用 `git checkout` 创建新分支，将当前的工作分支更改为不同的分支，甚至使用 `git checkout [branchname] [path to file]` 从不同的分支切换到不同版本的文件。“检出”操作会使用对象数据库中的树对象或 blob 更新工作树的全部或部分，以及更新索引和 HEAD（如果整个工作树指向新分支）。

(cherry-picking)
优 选

从一系列更改（通常是提交）中选择一部分更改，并在不同的代码库上将它们记录为新的更改系列。在 Git 中，这通过 `git cherry-pick` 命令来执行，在另一个分支上解压缩现有提交引入的更改，并根据当前分支的提示将其记录为新提交。

(clean)
清 洁

工作树在对应当前头部引用的版本时是清洁的。另请参阅“脏”。

(clone)
克 隆

克隆是指存在于计算机上而非网站服务器其他位置的仓库副本，或者是复制的操作。在克隆时，可在首选编辑器中编辑文件，使用 Git 跟踪更改而无需保持在线。你克隆的仓库仍与远程版本连接，以便当你在线时将本地更改推送到远程，以保持同步。

(code of conduct)
行 为 准 则

为如何参与社区制定标准的文档。

(code owner)
代码所有者

被指定为部分仓库代码所有者的个人。当有人打开对代码所有者拥有的代码进行更改的拉取请求（非草稿模式）时，会自动申请代码所有者审查。

(collaborator)
协 作 者

协作者是受仓库所有者邀请参与，对仓库拥有读取和写入权限的人。

(commit)
提 交

提交或“修订”是对一个文件（或一组文件）的个别更改。在进行提交以保存工作时，Git 会创建唯一的 ID（也称为 "SHA" 或“哈希”），用于记录提交的特定更改以及提交者和提交时间。提交通常包含一条提交消息，其中简要说明所做的更改。

(commit author)
提 交 作 者

进行提交的用户。

(commit ID)
提 交 ID

也称为 SHA。用于识别提交的 40 字符校验和的哈希。

(commit message)
提 交 消 息

随附于提交的简短描述性文字，用于沟通提交引入的更改。

(continuous integration)
持 续 集 成

也称为 CI。在个人对 GitHub 上配置的仓库提交更改后运行自动化构建和测试的过程。CI 是软件开发中一种帮助检测错误的常用最佳实践。

(contribution guidelines)
贡 献 指 南

说明人们应如何参与项目的文档。

(contributions)
贡 献

GitHub 上的特定活动。

(contributor)
贡 献 者

贡献者是指对仓库没有协作者权限但参与过项目，并且他们打开的拉取请求已合并到仓库的人员。

(default branch)
默 认 分 支

仓库中的基本分支，除非你指定不同的分支，否则会自动对它完成所有拉取请求和代码提交。此分支通常称为 `master` 。

(detached HEAD)
游离的 HEAD

如果你操作的是游离的 HEAD，Git 将会警告你，这意味着 Git 不指向某个分支，并且你的任何提交都不会出现在提交历史记录中。例如，在检出并非任何特定分支最新提交的任意提交时，你操作的是“游离的 HEAD”。

(diff)
差 异

差异是指两个提交之间的更改或保存的更改之间的区别，它将从视觉上描述文件自上次提交后添加或删除的内容。

(dirty)
脏

工作树如果包含尚未提交到当前分支的更改，将被视为“脏”。

(fast-forward)
快 进

快进是一种特殊类型的合并，在其中你有修订以及“合并”另一个分支的更改作为现有分支的子系。在这种情况下，你无法进行新的合并提交，而只是更新此修订。这在远程仓库的远程跟踪分支中经常发生。

(feature branch)
功 能 分 支

用于试验新功能或修复未正式使用的议题的分支。也称为主题分支。

(fenced code block)
围 栏 代 码 块

你可以在代码块前后使用三个反引号 ``` ，通过 GitHub Flavored Markdown 创建缩进代码块。

(fetch)
获 取

在使用 `git fetch` 时，你将从远程仓库添加更改到本地工作分支，而不提交它们。与 `git pull` 不同，提取可让你在更改提交到本地分支之前先进行审查。

(following (users))
跟 进 （ 用 户 ）

获取关于另一个用户的贡献和活动的通知。

(force push)
强 制 推 送

一种使用本地更改覆盖远程仓库的 Git 推送，不管是否冲突。

(fork)
复 刻

复刻是其他用户仓库在你的帐户上的个人副本。复刻允许你自由更改项目而不影响原始上游仓库。你也可以在上游仓库中打开拉取请求，并使复刻同步最新的更改，因为两个仓库仍然互相连接。

gitfile

一种普通的 `.git` 文件，始终位于工作树的根部，指向 Git 目录，包含整个 Git 仓库及其元数据。你可以在命令行上使用 `git rev-parse --git-dir` 查看仓库（实际仓库）的此文件。

HEAD

当前分支。

(hook) 挂钩

在多个 Git 命令正常执行时，对可选脚本进行标注以允许开发者添加功能或检查。通常，挂钩允许预先验证和潜在中止命令，并且允许在操作完成后再发事后通知。

(instance) 实例

组织包含在其配置和控制的虚拟机中的 GitHub 私有副本。

(issue) 议题

议题是提议的与仓库相关的改进、任务或问题。（对于公共仓库）任何人都可创建议题，然后由仓库协作者调解。每个议题都包含自己的讨论线程。你也可以使用标签将议题归类并分配到某人。

(key fingerprint) 密钥指纹

用于识别较长公钥的简短字节系列。

(keyword) 关键词

用在拉取请求中时关闭议题的特定文字。

(label) 标签

议题或拉取请求的标记。仓库随附一系列默认标签，但用户也可创建自定义标签。

LFS

Git Large File Storage。一种开源 Git 扩展，用于对大文件进行版本控制。

(license) 许可证

一种可随附于项目的文档，告知们能够对你的源代码执行哪些操作，不能执行哪些操作。

(line comment) 行注释

拉取请求内特定代码行上的评论。

(master) 主干

默认开发分支。只要创建 Git 仓库，就会创建一个名为 `master` 的分支，并且它会变为活动的分支。大多数情况下，这包含本地开发，但纯属惯例，而非必需。

(mention) 提及

一种通过在用户名前加上 @ 符号来发送给用户的通知。GitHub 上组织中的用户也可成为可提及的团队一部分。

(merge)
合并

合并是从一个分支（在相同的仓库中或来自一个分叉）提取更改，然后将其应用到另一个分支。这通常是作为“拉取请求”（可视为请求合并）或通过命令行完成。如果没有冲突的更改，可通过 GitHub.com web 界面使用拉取请求完成合并，或始终通过命令行完成。

(merge conflict)
合并冲突

合并的分支之间发生的差异。当人们对同一文件的同一行进行不同的更改时，或者一个人编辑某文件而另一个人删除该文件时，就会发生合并冲突。必须解决合并冲突后才可合并分支。

(merge request)
合并请求

合并请求（MR）是 GitLab 上类似于 GitHub 上的拉取请求的概念。

(milestone)
里程碑

一种跟踪仓库中议题或拉取请求组进度的方式。

(mirror)
镜像

仓库的新副本。

(non-fast-forward)
非快进

当仓库的本地副本未与上游仓库同步时，你在推送本地更改之前需要提取上游更改。

(notification)
通知

web 或电子邮件（根据你的设置）传送的更新，提供你感兴趣的活动的相关信息。

(outside collaborator)
外部协作者

已被授予访问一个或多个组织的仓库但对组织没有其他访问权限的用户，且不属于组织的成员。

(open source)
开源

开源软件是可供任何人自由使用、修改和共享（以修改和未修改的形式）的软件。今天，“开源”的概念通常扩展到软件以外，代表一种协作原则，其中工作材料在线供任何人分叉、修改、讨论和参与。

(origin)
源

默认上游仓库。大多数项目至少有一个它们跟踪的上游项目。默认情况下，源用于该目的。

(owner)
所有者

对组织有完全管理权限的组织成员。

(private contributions)
私有贡献

对私有（与公共相对）仓库的贡献。

(private repository)
私有仓库

私有仓库仅对仓库所有者和所有者指定的协作者可见。

(production branch)
生产分支

包含可使用或部署到应用程序或站点的最终更改的分支。

(profile)
个人资料

显示 GitHub 上用户活动相关信息的页面。

(protected branch)
受保护分支

受保护分支在仓库管理员选择保护的分支上禁止多项 Git 功能。必要检查未通过或必需审查未批准，不能对它们执行强制推送、删除和更改合并，或者不能从 GitHub web 界面上传文件到其中。受保护分支通常是默认分支。

(public contributions)
公共贡献

对公共（与私有相对）仓库的贡献。

(public repository)
公共仓库

公共仓库可供任何人查看，包括不是 GitHub 用户的人员。

(pull)
拉取

拉取是指提取与合并更改。例如，如果有人编辑了你操作的远程文件，你要将这些更改拉取到本地副本，以使其保持最新。另请参阅“提取”。

(pull access)
拉取权限

读取权限的同义词。

(pull request)
拉取请求

拉取请求（PR）是提议更改用户提交的仓库，然后被仓库协作者接受或拒绝。像议题一样，每个拉取请求都有自己的论坛。

(pull request review)
拉取请求审查

拉取请求中协作者批准更改或在拉取请求合并之前申请进一步更改的评论。

(push)
推送

推送是指将提交的更改发送到 GitHub.com 上的远程仓库。例如，如果你在本地更改内容，便可推送这些更改，让其他人访问。

(push a branch)
推送分支

成功将分支推送到远程仓库后，使用本地分支中的更改来更新远程分支。在你“推送分支”时，Git 将会到远程仓库中搜索分支的头部引用，并验证它是分支本地头部引用的直系原型。在验证后，Git 将拉取所有对象（从本地头部引用可获取，而远程仓库中缺失）到远程对象数据库，然后更新远程头部引用。如果远程头部不是本地头部的原型，推送将会失败。

(push access)
推送权限

写入权限的同义词。

(read access)
读取权限

对仓库的权限级别，允许用户拉取或者读取仓库中的信息。所有公共仓库向所有 GitHub 用户授予读取权限。拉取权限的同义词。

(README)
自述文件

包含仓库中文件相关信息的文本文件，通常是仓库访问者看到的第一个文件。自述文件连同仓库许可证、参与指南以及行为准则，帮助你交流要求和管理项目的参与。

(rebase)
变基

将一系列更改从一个分支重新应用到不同的基本分支，并将该分支的头部重置为结果。

(release)
发布

GitHub 封装软件并向用户提供软件的方式。

(remote)
远程

这是托管于服务器（很可能是 GitHub.com）上的仓库或分支版本。远程版本可以连接到本地克隆，以使更改保持同步。

(remote repository)
远程仓库

用于跟踪同一个项目但储存在其他位置的仓库。

(remote URL)
远程 URL

存储代码的位置：GitHub、其他用户分支甚至不同服务器 上的仓库。

(replica)
副本

为主要 GitHub Enterprise 实例提供冗余的 GitHub Enterprise 实例。

(repository)
仓库

仓库是 GitHub 最基本的元素，最容易被想象成项目的文件夹。一个仓库包含所有项目文件（包括文档），并且存储每个文件的修订历史记录。仓库可有多个协作者，也可以是公共仓库或私有仓库。

(repository maintainer)
仓库维护员

管理仓库的人员。此人可帮助对议题分类，以及使用标签和其他功能管理仓库的工作，也可负责更新自述文件和参与文件。

(resolve)
解决

手动修复自动合并失败的操作。

(revert)
还原

恢复 GitHub 上的拉取请求时，新拉取请求会自动打开，其中有一个提交用于从原始合并的拉取请求恢复合并提交。在 Git 中，你可以使用 `git revert` 恢复提交。

(review)
审查

审查允许对仓库具有访问权限的其他人评论拉取请求中提议的更改、审批更改或在拉取请求合并之前请求进一步更改。

(service hook)
服 务 挂 钩

也称为“Web 挂钩”。Web 挂钩是一种通知方式，只要仓库或组织上发生特定操作，就会发送通知到外部 web 服务器。

(squash)
压 扁

将多个提交合并为一个。也是 Git 命令。

(staging instance)
暂 存 实 例

在修改应用到实际 GitHub Enterprise 实例之前测试修改的一种方式。

(status)
状 态

拉取请求中的可视表现形式，表示你的提交符合你参与的仓库所设定的条件。

(star)
星 标

仓库的书签或赞赏表示。星标是项目受欢迎程度排名的手动方式。

(topic branch)
主 题 分 支

开发者用来识别开发概念行的常规 Git 分支。由于分支很容易并且便宜，因此往往适合拥有多个小分支，每个小分支包含定义明确的概念，或者渐进但相关的更改。也可称为“特征分支”。

(upstream)
上 游

在谈论分支或分叉时，原始仓库的主要分支通常被称为“上游”，因为它是其他更改的主要来源。你操作的分支/分叉则称为“下游”。也称为“源”。

(upstream branch)
上 游 分 支

合并到所述分支的默认分支（或所述分支变基到的分支）。它通过 `branch.<name>.remote` 和 `branch.<name>.merge` 配置。如果 A 的上游分支是源/B，有时我们会说“A 在跟踪源/B”。

(watch)
查 看

你可以关注仓库或议题，以便在议题或拉取请求有更新时接收通知。

(webhooks)
web 挂钩

Web 挂钩可让你构建或设置订阅 GitHub.com 上特定事件的 GitHub 应用程序。Web 挂钩提供一种通知方式，只要仓库或组织中发生特定操作，就会发送通知到外部 web 服务器。也称为“服务挂钩”。

(write access)
写 入 权 限

对仓库的权限级别，可让用户推送或写入更改到仓库。

来自广东广州的 Chrome 117.0|GNU/Linux 用户 2023-10-18 16:13 赞 回复

clean也可译作“清理”
哈希是普通话版的音译，意译还有另外一个词“散列码”
tag没提到，可以考虑译为“（打）标记”，与label区别开
watch可译为“关注”

来自云南曲靖的 Chrome 81.0|Windows 10 用户 2020-05-25 17:04 42 赞 回复

ID=身份或标识
...

上一篇：使用子模块和子树来管理 Git 项目
下一篇：你应该（或许）没使用过的 3 种 Python 模板语言

相关阅读

Git 更多标签

- 使用 Tig 来可视化 Git 工作流 2022-07-09
- 删除 Git 远程分支的本地引用 2022-08-11
- 我在 Linux 中使用的 5 个 Git 配置 2022-10-11
- 用 git log 命令显示在特定日期的提交记录 2022-11-06
- 使用 Git bisect 命令定位首次引入错误的提交 2022-12-02