

目录

Git教程

Git简介

Git的诞生

集中式vs分布式

安装Git

创建版本库

时光机穿梭

版本回退

工作区和暂存区

管理修改

撤销修改

删除文件

远程仓库

添加远程库

从远程库克隆

分支管理

创建与合并分支

解决冲突

分支管理策略

Bug分支

Feature分支

多人协作

标签管理

创建标签

操作标签

使用GitHub

自定义Git

忽略特殊文件

配置别名

搭建Git服务器

期末总结

关于作者



廖雪峰

北京 朝阳区

加关注

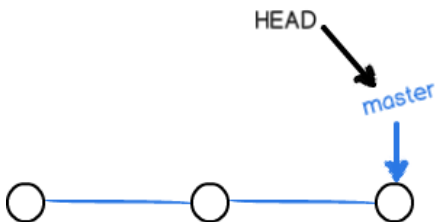
创建与合并分支

阅读: 151495

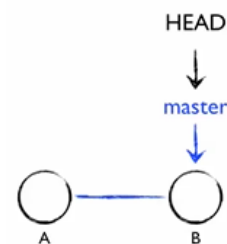


在版本回退里，你已经知道，每次提交，Git都把它们串成一条时间线，这条时间线就是一个分支。截止到目前，只有一条时间线，在Git里，这个分支叫主分支，即 `master` 分支。`HEAD` 严格来说不是指向提交，而是指向 `master`，`master` 才是指向提交的，所以，`HEAD` 指向的就是当前分支。

一开始的时候，`master` 分支是一条线，Git用 `master` 指向最新的提交，再用 `HEAD` 指向 `master`，就能确定当前分支，以及当前分支的提交点：



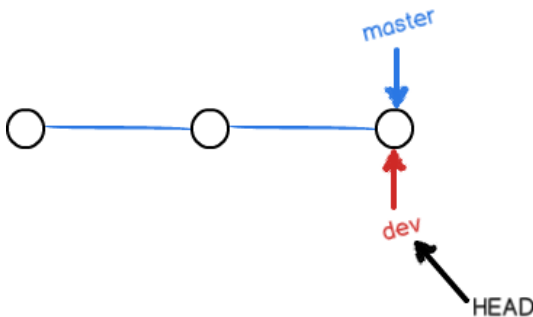
每次提交，`master` 分支都会向前移动一步，这样，随着你不断提交，`master` 分支的线也越来越长：



00:00

-00:19

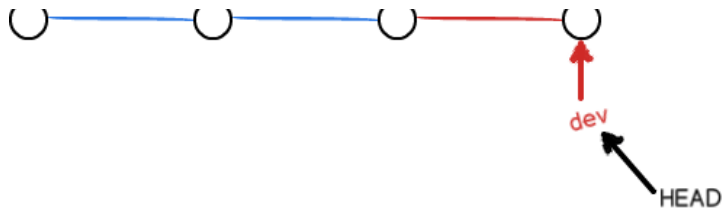
当我们创建新的分支，例如 `dev` 时，Git新建了一个指针叫 `dev`，指向 `master` 相同的提交，再把 `HEAD` 指向 `dev`，就表示当前分支在 `dev` 上：



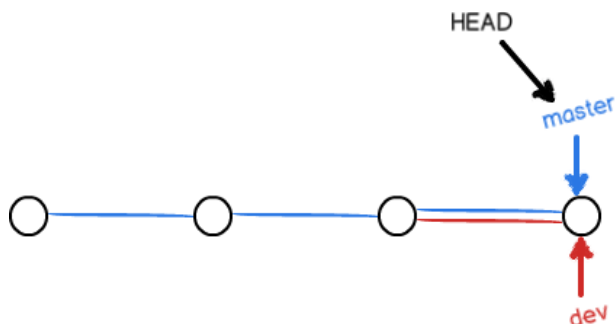
你看，Git创建一个分支很快，因为除了增加一个 `dev` 指针，改改 `HEAD` 的指向，工作区的文件都没有任何变化！

不过，从现在开始，对工作区的修改和提交就是针对 `dev` 分支了，比如新提交一次后，`dev` 指针往前移动一步，而 `master` 指针不变：



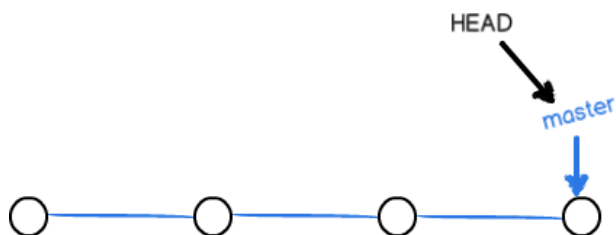


假如我们在 `dev` 上的工作完成了，就可以把 `dev` 合并到 `master` 上。Git怎么合并呢？最简单的方法，就是直接把 `master` 指向 `dev` 的当前提交，就完成了合并：

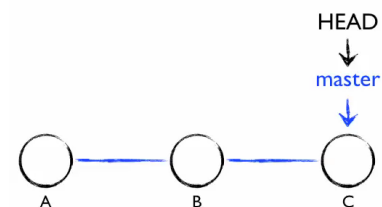


所以Git合并分支也很快！就改改指针，工作区内容也不变！

合并完分支后，甚至可以删除 `dev` 分支。删除 `dev` 分支就是把 `dev` 指针给删掉，删掉后，我们就剩下了一条 `master` 分支：



真是太神奇了，你看得出来有些提交是通过分支完成的吗？



00:00

-00:33

下面开始实战。

首先，我们创建 `dev` 分支，然后切换到 `dev` 分支：

```
$ git checkout -b dev
Switched to a new branch 'dev'
```

`git checkout` 命令加上 `-b` 参数表示创建并切换，相当于以下两条命令：

```
$ git branch dev
$ git checkout dev
Switched to branch 'dev'
```

然/口，用 `git branch` 即可查看当前分支。

```
$ git branch
* dev
  master
```

`git branch` 命令会列出所有分支，当前分支前面会标一个 `*` 号。

然后，我们就可以在 `dev` 分支上正常提交，比如对 `readme.txt` 做个修改，加上一行：

```
Creating a new branch is quick.
```

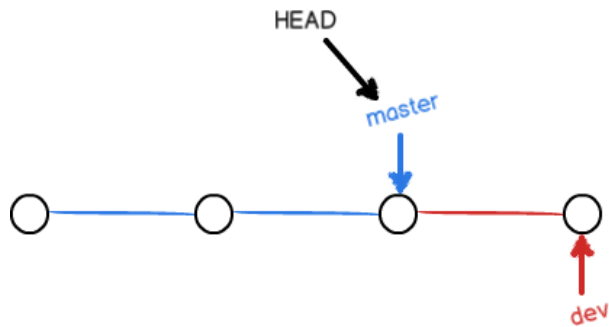
然后提交：

```
$ git add readme.txt
$ git commit -m "branch test"
[dev fec145a] branch test
1 file changed, 1 insertion(+)
```

现在，`dev` 分支的工作完成，我们就可以切换回 `master` 分支：

```
$ git checkout master
Switched to branch 'master'
```

切换回 `master` 分支后，再查看一个 `readme.txt` 文件，刚才添加的内容不见了！因为那个提交是在 `dev` 分支上，而 `master` 分支此刻的提交点并没有变：



现在，我们把 `dev` 分支的工作成果合并到 `master` 分支上：

```
$ git merge dev
Updating d17efd8..fec145a
Fast-forward
 readme.txt | 1 +
 1 file changed, 1 insertion(+)
```

`git merge` 命令用于合并指定分支到当前分支。合并后，再查看 `readme.txt` 的内容，就可以看到，和 `dev` 分支的最新提交是完全一样的。

注意到上面的 `Fast-forward` 信息，Git 告诉我们，这次合并是“快进模式”，也就是直接把 `master` 指向 `dev` 的当前提交，所以合并速度非常快。

当然，也不是每次合并都能 `Fast-forward`，我们后面会将其他方式的合并。


合并完成后，就可以放心地删除 `dev` 分支了：

```
$ git branch -d dev
Deleted branch dev (was fec145a).
```

删除后，查看 `branch`，就只剩下 `master` 分支了：

```
$ git branch
* master
```

因为创建、合并和删除分支非常快，所以 Git 鼓励你使用分支完成某个任务，合并后再删掉分支，这和直



learngit — bash — 80x24

macbookpro ~/learngit \$

00:00 -01:05

Git鼓励大量使用分支：

创建分支: `git branch <name>`

切换分支: `git checkout <name>`

创建+切换分支: `git checkout -b <name>`

合并某分支到当前分支: `git merge <name>`

删除分支: `git branch -d <name>`



分享 果殼裏 J... 等2人分享过



如果您认为本网站的教程质量不错，读后觉得收获很大，预期工资涨幅能超过30%，不妨小额赞助我一下，让我有动力继续写出高质量的教程：

<http://www.liaoxuefeng.com/wiki/0013739516305929606dd1836...7b000/001375840038939c291467cc7c747b1810aab2fb8863508000>

解决冲突

发表评论



廖雪峰的官方网站©2015
Powered by [iTranswarp.js](#)
由阿里云托管
[广告合作](#)



[Feedback](#)
[License](#)

友情链接: [中华诗词](#) - [阿里云](#) - [SICP](#) - [4closure](#)