繁體中文,

Translations started for

Беларуская,
Deutsch,
فارسی,
Ελληνικά,
Italiano,
Bahasa Melayu,
Polski,
Português (Brasil),
Türkçe,
Ўзбекча.

The source of this book is hosted on GitHub.
Patches, suggestions and comments are welcome.

Chapters ▼

1st Edition

.6 Git 基础 - 打标签

打标签

同大多数 VCS 一样，Git 也可以对某一时间点上的版本打上标签。人们在发布某个软件版本（比如 v1.0 等等）的时候，经常这么做。本节我们一起来学习如何列出所有可用的标签，如何新建标签，以及各种不同类型标签之间的差别。

列显已有的标签

列出现有标签的命令非常简单，直接运行 git tag 即可：

```
$ git tag
v0.1
v1.3
```

显示的标签按字母顺序排列，所以标签的先后并不表示重要程度的轻重。

我们可以用特定的搜索模式列出符合条件的标签。在 Git 自身项目仓库中，有着超过 240 个标签，如果你只对 1.4.2 系列的版本感兴趣，可以运行下面的命令：

```
$ git tag -l 'v1.4.2.*'
v1.4.2.1
v1.4.2.2
v1.4.2.3
v1.4.2.4
```

新建标签

Git 使用的标签有两种类型：轻量级的（lightweight）和含附注的（annotated）。轻量级标签就像是个不会变化的分支，实际上它就是个指向特定提交对象的引用。而含附注标签，实际上是存储在仓库中的一个独立对象，它有自身的校验和信息，包含着标签的名字，电子邮件地址和日期，以及标签说明，标签本身也允许使用 GNU Privacy Guard (GPG) 来签署或验证。一般我们都建议使用含附注型的标签，以便保留相关信息；当然，如果只是临时性加注标签，或者不需要旁注额外信息，用轻量级标签也没问题。

含附注的标签

创建一个含附注类型的标签非常简单，用 -a （译注：取 annotated 的首字母）指定标签名字即可：

```
$ git tag -a v1.4 -m 'my version 1.4'
$ git tag
v0.1
v1.3
v1.4
```

而 -m 选项则指定了对应的标签说明，Git 会将此说明一同保存在标签对象中。如果没有给出该选项，Git 会启动文本编辑软件供你输入标签说明。

可以使用 git show 命令查看相应标签的版本信息，并连同显示打标签时的提交对象。

```
$ git show v1.4
tag v1.4
Tagger: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Feb 9 14:45:11 2009 -0800

my version 1.4

commit 15027957951b64cf874c3557a0f3547bd83b3ff6
Merge: 4a447f7... a6b4c97...
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sun Feb 8 19:02:46 2009 -0800

    Merge branch 'experiment'
```

我们可以看到在提交对象信息上面，列出了此标签的提交者和提交时间，以及相应的标签说明。

签署标签

如果你有自己的私钥，还可以用 GPG 来签署标签，只需要把之前的 -a 改为 -s （译注： 取 signed 的首字母）即可：

```
$ git tag -s v1.5 -m 'my signed 1.5 tag'
You need a passphrase to unlock the secret key for
user: "Scott Chacon <schacon@gee-mail.com>"
1024-bit DSA key, ID F721C45A, created 2009-02-09
```

现在再运行 git show 会看到对应的 GPG 签名也附在其内：

```
$ git show v1.5
tag v1.5
Tagger: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Feb 9 15:22:20 2009 -0800

my signed 1.5 tag
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.8 (Darwin)

iEYEABECAAYFAkmQurIACgkQON3DxfchxFr5cACeIMN+ZxLKggJQf0QYiQBwgySN
Ki0An2JeAVUCAiJ7Ox6ZEtK+NvZAj82/
=WryJ
-----END PGP SIGNATURE-----
commit 15027957951b64cf874c3557a0f3547bd83b3ff6
Merge: 4a447f7... a6b4c97...
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sun Feb 8 19:02:46 2009 -0800

    Merge branch 'experiment'
```

稍后我们再学习如何验证已经签署的标签。

轻量级标签

轻量级标签实际上就是一个保存着对应提交对象的校验和信息的文件。要创建这样的标签，一个 -a，-s 或 -m 选项都不用，直接给出标签名字即可：

```
$ git tag v1.4-lw
$ git tag
v0.1
v1.3
v1.4
v1.4-lw
v1.5
```

现在运行 git show 查看此标签信息，就只有相应的提交对象摘要：

```
$ git show v1.4-lw
commit 15027957951b64cf874c3557a0f3547bd83b3ff6
Merge: 4a447f7... a6b4c97...
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sun Feb 8 19:02:46 2009 -0800

    Merge branch 'experiment'
```

验证标签

可以使用 git tag -v [tag-name] （译注：取 verify 的首字母）的方式验证已经签署的标签。此命令

会调用 GPG 来验证签名，所以你需要有签署者的公钥，存放在 keyring 中，才能验证：

```
$ git tag -v v1.4.2.1
object 883653babd8ee7ea23e6a5c392bb739348b1eb61
type commit
tag v1.4.2.1
tagger Junio C Hamano <junkio@cox.net> 1158138501 -0700

GIT 1.4.2.1

Minor fixes since 1.4.2, including git-mv and git-http with alternates.
gpg: Signature made Wed Sep 13 02:08:25 2006 PDT using DSA key ID F3119B9A
gpg: Good signature from "Junio C Hamano <junkio@cox.net>"
gpg:                 aka "[jpeg image of size 1513]"
Primary key fingerprint: 3565 2A26 2040 E066 C9A7  4A7D C0C6 D9A4 F311 9B9A
```

若是没有签署者的公钥，会报告类似下面这样的错误：

```
gpg: Signature made Wed Sep 13 02:08:25 2006 PDT using DSA key ID F3119B9A
gpg: Can't check signature: public key not found
error: could not verify the tag 'v1.4.2.1'
```

后期加注标签

你甚至可以在后期对早先的某次提交加注标签。比如在下面展示的提交历史中：

```
$ git log --pretty=oneline
15027957951b64cf874c3557a0f3547bd83b3ff6 Merge branch 'experiment'
a6b4c97498bd301d84096da251c98a07c7723e65 beginning write support
0d52aaab4479697da7686c15f77a3d64d9165190 one more thing
6d52a271eda8725415634dd79daabbc4d9b6008e Merge branch 'experiment'
0b7434d86859cc7b8c3d5e1dddfed66ff742fcbc added a commit function
4682c3261057305bdd616e23b64b0857d832627b added a todo file
166ae0c4d3f420721acbb115cc33848dfcc2121a started write support
9fceb02d0ae598e95dc970b74767f19372d61af8 updated rakefile
964f16d36dfccde844893cac5b347e7b3d44abbc commit the todo
8a5cbc430f1a9c3d00faaeffd07798508422908a updated readme
```

我们忘了在提交 “updated rakefile” 后为此项目打上版本号 v1.2，没关系，现在也能做。只要在打标签的时候跟上对应提交对象的校验和（或前几位字符）即可：

```
$ git tag -a v1.2 9fceb02
```

可以看到我们已经补上了标签：

```
$ git tag
v0.1
v1.2
v1.3
v1.4
v1.4-lw
```

```
v1.5

$ git show v1.2
tag v1.2
Tagger: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Feb 9 15:32:16 2009 -0800

version 1.2
commit 9fceb02d0ae598e95dc970b74767f19372d61af8
Author: Magnus Chacon <mchacon@gee-mail.com>
Date:   Sun Apr 27 20:43:35 2008 -0700

    updated rakefile
...
```

分享标签

默认情况下，git push 并不会把标签传送到远端服务器上，只有通过显式命令才能分享标签到远端仓库。其命令格式如同推送分支，运行 git push origin [tagname] 即可：

```
$ git push origin v1.5
Counting objects: 50, done.
Compressing objects: 100% (38/38), done.
Writing objects: 100% (44/44), 4.56 KiB, done.
Total 44 (delta 18), reused 8 (delta 1)
To git@github.com:schacon/simplegit.git
* [new tag]         v1.5 -> v1.5
```

如果要一次推送所有本地新增的标签上去，可以使用 --tags 选项：

```
$ git push origin --tags
Counting objects: 50, done.
Compressing objects: 100% (38/38), done.
Writing objects: 100% (44/44), 4.56 KiB, done.
Total 44 (delta 18), reused 8 (delta 1)
To git@github.com:schacon/simplegit.git
 * [new tag]         v0.1 -> v0.1
 * [new tag]         v1.2 -> v1.2
 * [new tag]         v1.4 -> v1.4
 * [new tag]         v1.4-lw -> v1.4-lw
 * [new tag]         v1.5 -> v1.5
```

现在，其他人克隆共享仓库或拉取数据同步后，也会看到这些标签。

prev | next
About this site