

THE EXPERT'S VOICE®

SECOND EDITION

# Pro Git

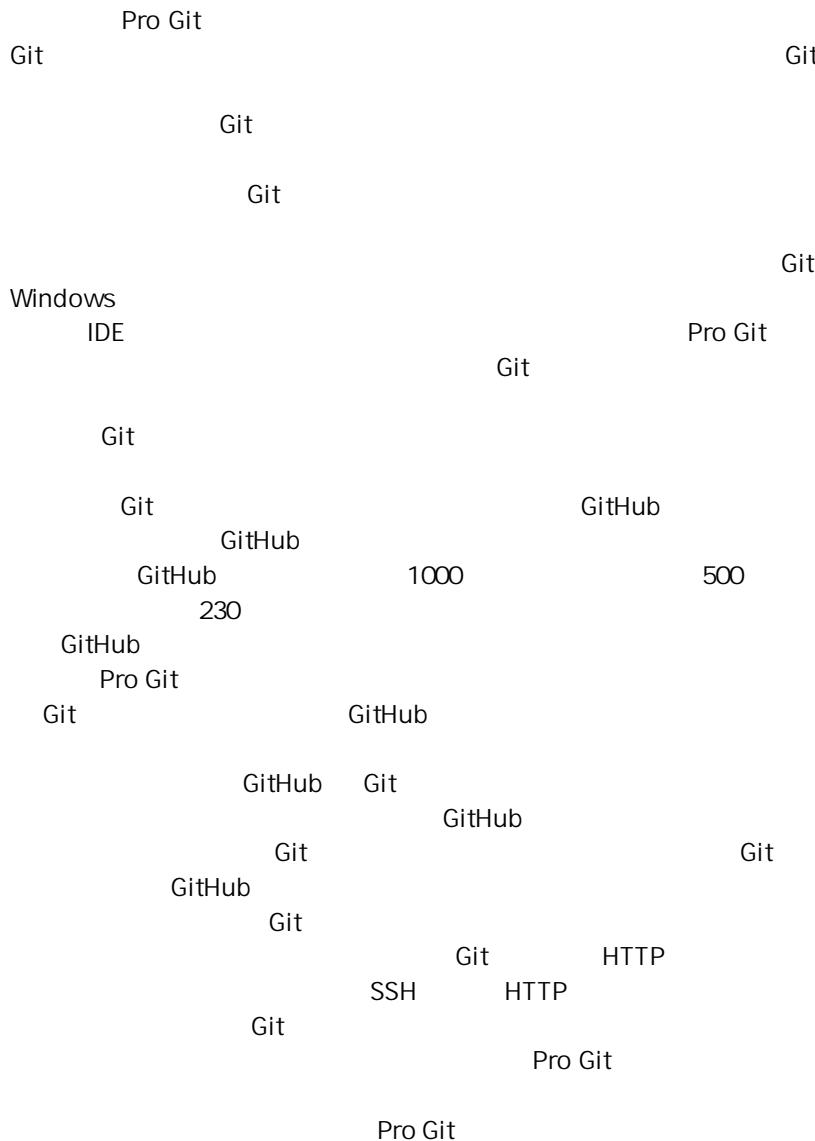
*EVERYTHING YOU NEED TO  
KNOW ABOUT GIT*

Scott Chacon and Ben Straub

Apress®

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/3.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

# Scott Chacon 序





# Ben Straub 序

Git

Git

Git Scott

Git



## 献辞

*Becky*

*Josephine*

*— Ben*

*Jessica*

*— Scott*



# 贡献者

ProGit

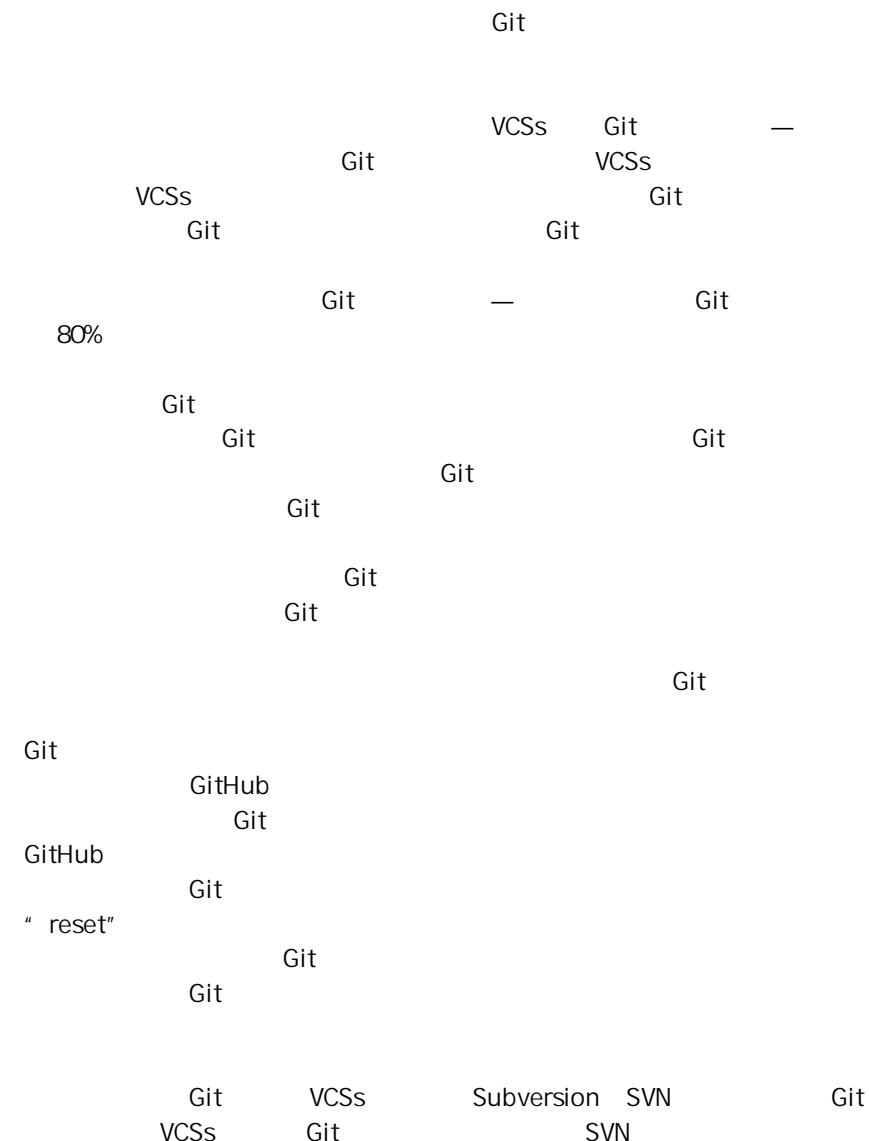
2 Aaron Schumacher  
4 Aggelos Orfanakos  
4 Alec Clews  
1 Alex Moundalexis  
2 Alexander Harkness  
1 Alexander Kahn  
1 Andrew McCarthy  
1 AntonioK  
1 Benjamin Bergman  
1 Brennon Bortz  
2 Brian P O'Rourke  
1 Bryan Goines  
1 Cameron Wright  
1 Chris Down  
1 Christian Kluge  
1 Christoph Korn  
2 Ciro Santilli  
2 Cor  
1 Dan Croak  
1 Dan Johnson  
1 Daniel Kay  
2 Daniel Rosen  
1 DanielWeber  
1 Dave Dash  
10 Davide Fiorentino lo Regio  
2 Dilip M  
1 Dimitar Bonev  
1 Emmanuel Trillaud  
1 Eric-Paul Lecluse  
1 Eugene Serkin  
1 Fernando Dobladez  
2 Gordon McCreight  
1 Helmut K. C. Tessarek  
31 Igor Murzov

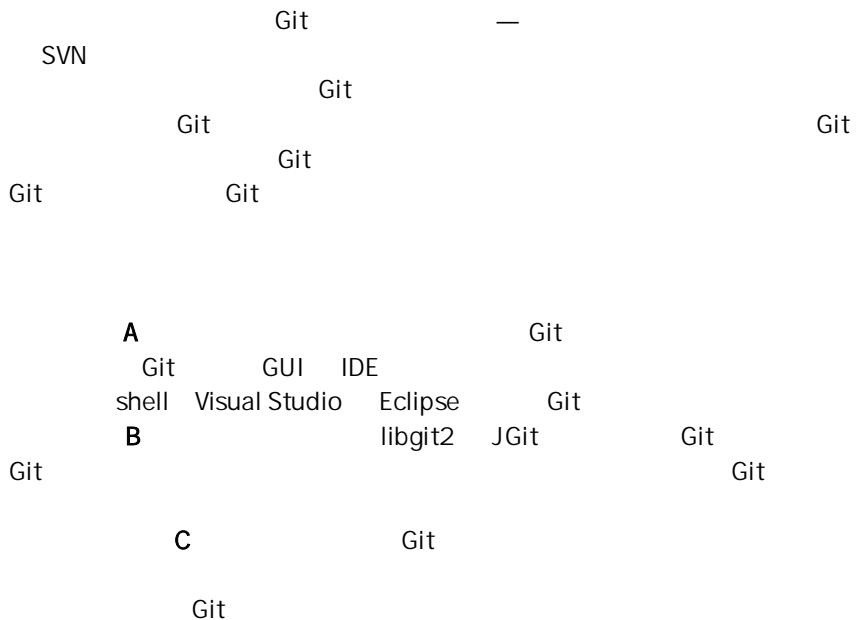
1 Ilya Kuznetsov  
1 Jason St. John  
1 Jay Taggart  
1 Jean Jordaan  
51 Jean-Noël Avila  
1 Jean-Noël Rouvignac  
1 Jed Hartman  
1 Jeffrey Forman  
1 John DeStefano  
1 Junior  
1 Kieran Spear  
1 Larry Shatzer, Jr  
1 Linquize  
1 Markus  
7 Matt Deacalion Stevens  
1 Matthew McCullough  
1 Matthieu Moy  
1 Max F. Albrecht  
1 Michael Schneider  
8 Mike D. Smith  
1 Mike Limansky  
1 Olivier Trichet  
1 Ondrej Novy  
6 Ori Avtalion  
1 Paul Baumgart  
1 Peter Vojtek  
1 Philipp Kempgen  
2 Philippe Lhoste  
1 PowerKiKi  
1 Radek Simko  
1 Rasmus Abrahamsen  
1 Reinhart Holler  
1 Ross Light  
1 Ryuichi Okumura  
1 Sebastian Wiesinger  
1 Severyn Kozak  
1 Shane  
2 Shannen  
8 Sitaram Chamarty  
5 Soon Van  
4 Sven Axelsson  
2 Tim Court  
1 Tuomas Suutari  
1 Vlad Gorodetsky  
3 W. Trevor King  
1 Wyatt Carss  
1 Włodzimierz Gajda  
1 Xue Fuqiao  
1 Yue Lin Ho  
2 adelcambre

1 anaran  
1 bdukes  
1 burningTyger  
1 cor  
1 iosias  
7 nicesw123  
1 onovy  
2 pcasaretto  
1 sampablokuper



## 引言





# Table of Contents

<b>Scott Chacon 序</b>	iii
<b>Ben Straub 序</b>	v
献辞	vii
贡献者	ix
引言	xiii
 <b>CHAPTER 1: 起步</b>	 27
	27
	27
	28
	29
Git	30
Git	31
31	31
32	32
Git	33
Git	33
33	34
35	35
Git	35
Linux	35

## Table of Contents

Mac	36
Windows	37
	37
Git	38
	38
	39
	39
	40
	40
<b>CHAPTER 2: Git 基础</b>	<b>41</b>
Git	41
	41
	42
	43
	43
	44
	45
	46
	47
	48
	51
	52
	53
	54
	55
	60
	61
	62
	63
	64

	64
	65
	66
	67
	67
	68
	69
	69
	69
	70
	70
	71
	72
	73
Git	73
	74
<b>CHAPTER 3: Git 分支</b>	<b>75</b>
	75
	77
	79
	82
	83
	87
	89
	91
	93
	93
	94
	96
	102

	104
	105
	105
	106
	106
	108
	111
	114
VS.	115
	115
<b>CHAPTER 4: 服务器上的 Git</b>	<b>117</b>
	117
	117
HTTP	119
SSH	121
Git	122
Git	122
	123
	124
SSH	124
	126
Git	128
Smart HTTP	130
GitWeb	131
GitLab	133
	134
	134
	137
	137
	138

	138
<b>CHAPTER 5: 分布式 Git</b>	<b>139</b>
	139
	139
	140
	141
	142
	142
	143
	145
	152
	157
	161
	164
	164
	165
	165
	169
	169
	171
	177
	178
	178
	179
	179
<b>CHAPTER 6: GitHub</b>	<b>181</b>
	181
SSH	182
	184

## Table of Contents

	185
	186
	187
Fork	187
GitHub	187
	195
Markdown	199
	204
	204
	206
	207
	213
	214
	215
	216
	216
README	217
CONTRIBUTING	217
	218
	219
	219
	220
	221
GitHub	222
	223
GitHub API	227
	228
	229
Pull Request	230
Octokit	232

	232
<b>CHAPTER 7: Git 工具</b>	<b>233</b>
	233
	233
SHA-1	233
	235
	235
	237
	238
	241
	241
	244
	245
	245
	247
	249
	249
	251
GPG	251
	251
	252
	253
	255
	255
Git Grep	255
Git	257
	258
	258
	259
	261

## Table of Contents

	261
	263
filter-branch	264
	265
	265
	267
	273
	278
	280
	283
	285
	285
	286
	296
	300
Rerere	303
Git	309
	310
	311
	313
	313
	315
	317
	327
	329
	331
	334
	343
	344
	346

		348
<b>CHAPTER 8: 自定义 Git</b>		<b>349</b>
Git		349
		350
Git		353
		354
		357
		359
Git		360
		361
		363
		366
		368
Git		368
		368
		369
		371
		372
		372
		377
		380
<b>CHAPTER 9: Git 与其他系统</b>		<b>381</b>
Git		381
Git Subversion		381
Git Mercurial		392
Git Perforce		400
Git TFS		415
Git		424
Subversion		424

## Table of Contents

Mercurial	426
Perforce	428
TFS	430
	432
	439
<b>CHAPTER 10: Git 内部原理</b>	<b>441</b>
	441
Git	442
	444
	448
	451
Git	453
HEAD	454
	455
	456
	457
	461
	462
	463
	463
	463
	466
	469
	469
	469
	470
	472
	476
	476
	477

	477
	477
	478
	478
	479
	480
	481
其它环境中的 Git	483
将 Git 嵌入你的应用	497
Git 命令	507
Index	523



# 起步 1

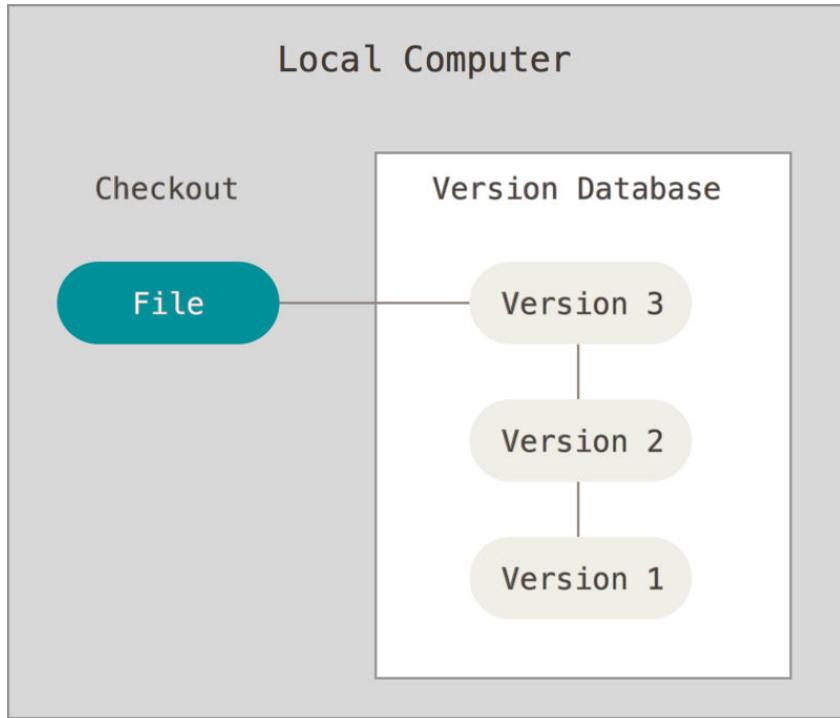
Git  
Git  
Git  
Git  
Git

## 关于版本控制

VCS

**FIGURE 1-1**

本地版本控制

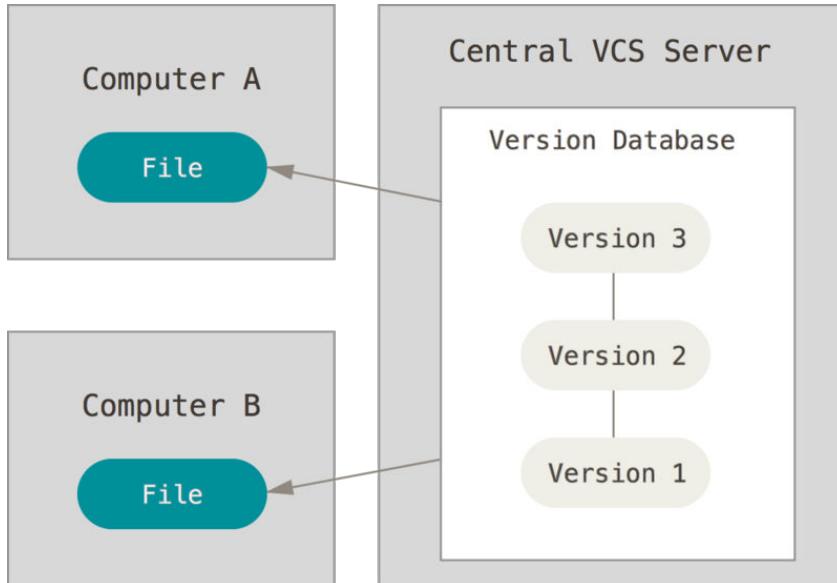


RCS  
Mac OS X  
rcs

CVCS Centralized Version Control Systems  
CVS Subversion Perforce

**FIGURE 1-2**

集中化的版本控制.



VCS

CVCS

—

Distributed Version Control System

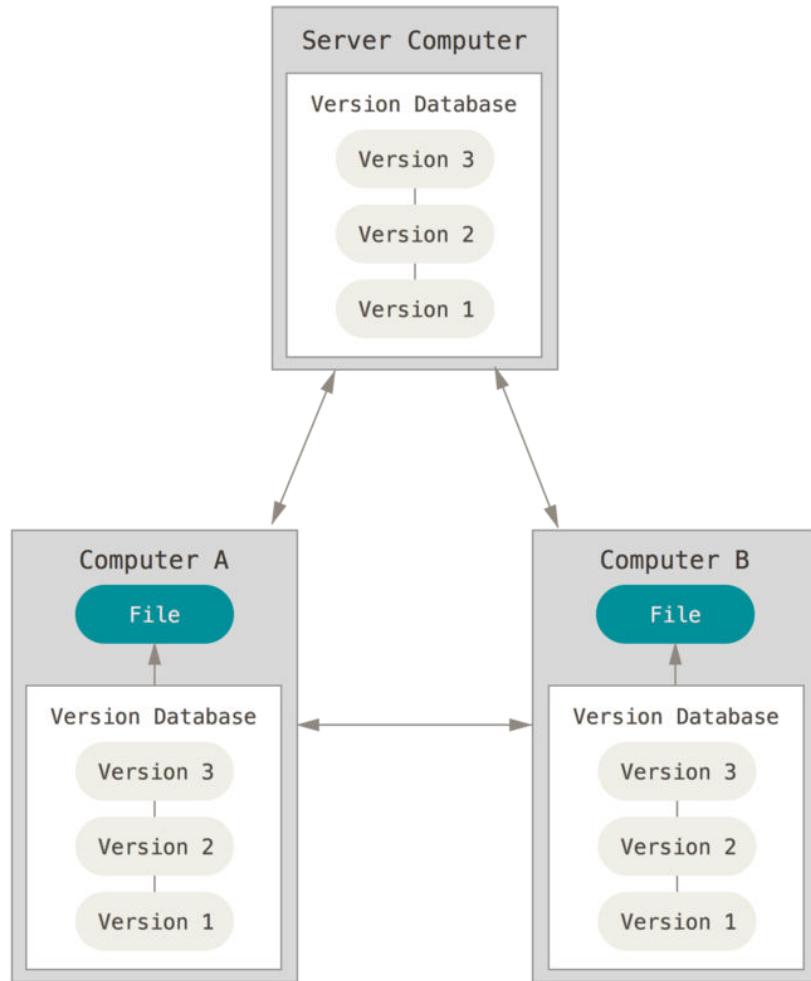
DVCS

Git Mercurial Bazaar

Darcs

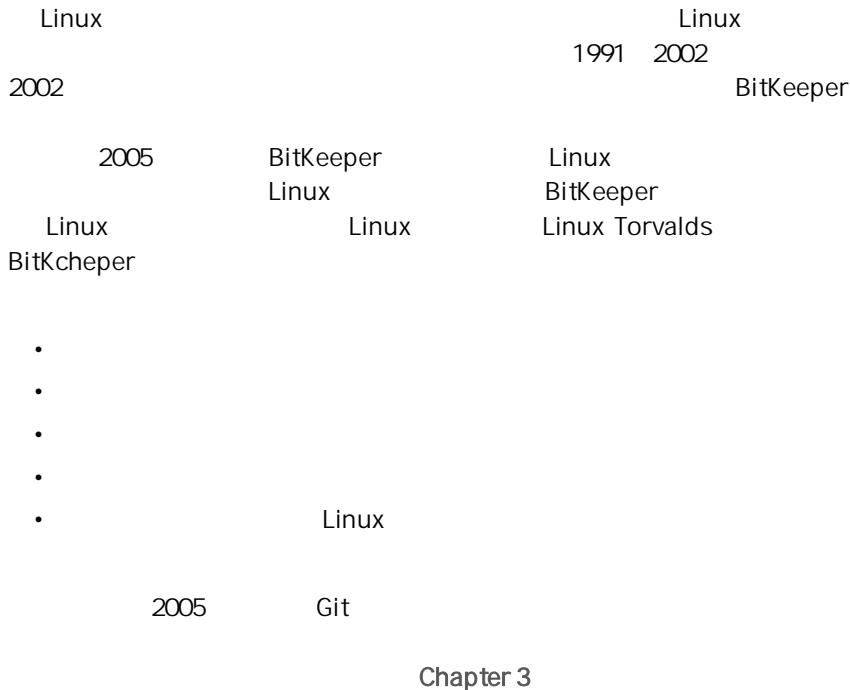
**FIGURE 1-3**

分布式版本控制.



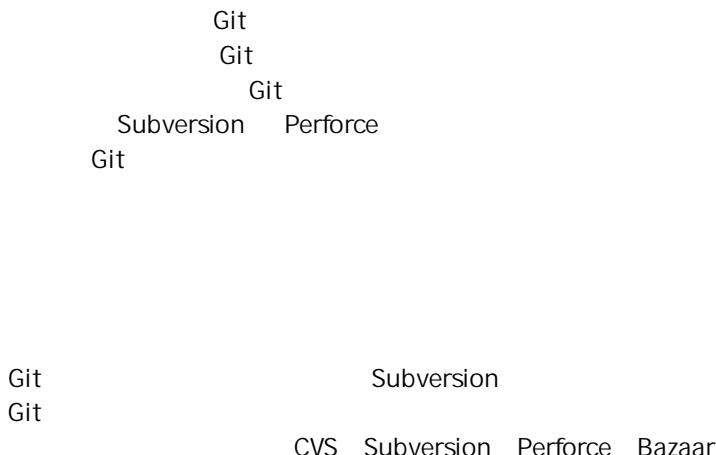
## Git 简史

Git



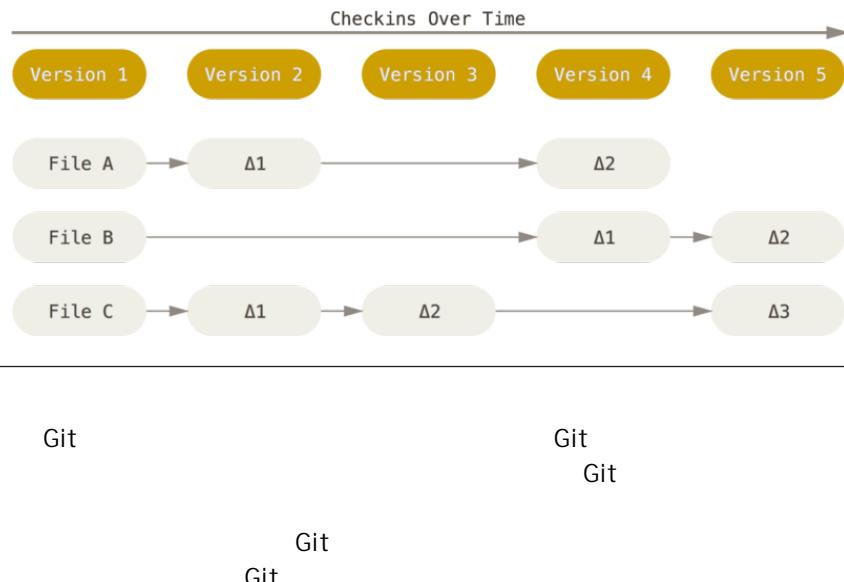
### Chapter 3

## Git 基础



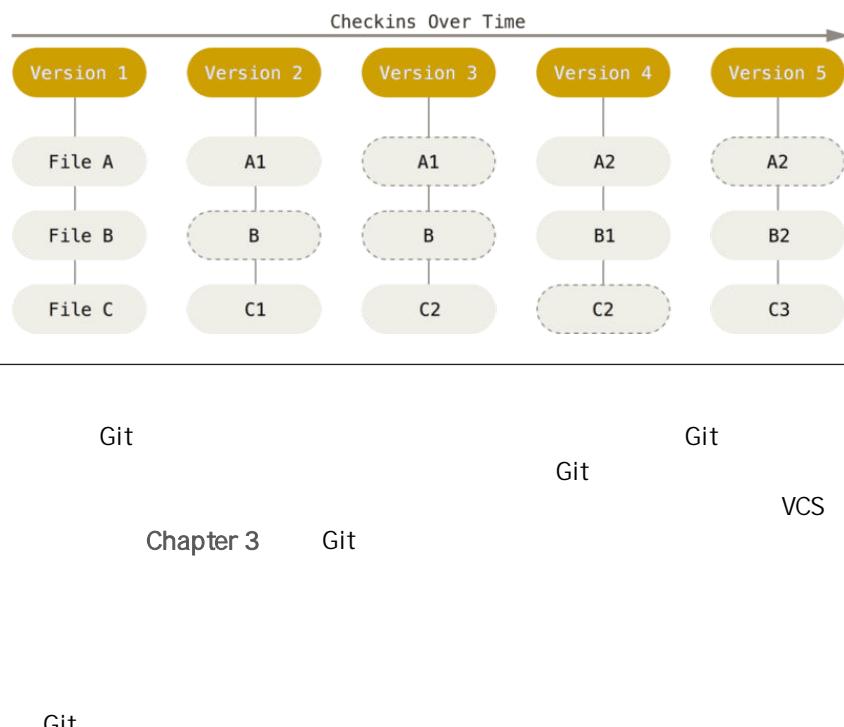
**FIGURE 1-4**

存储每个文件与初始版本的差异。



**FIGURE 1-5**

存储项目随时间改变的快照。



Git

Git

Git

—  
Git

VPN

VPN

Subversion      Perforce  
CVS

## Git

Git

Git  
Git  
Git  
Git  
40                0-9      a-f  
                   SHA-1      hash  
                   SHA-1

Git

24b9da6552252987aa493b52f8696cd6d3b00373

Git  
Git

## Git

Git

Git

Git

VCS

Git

Git

Git

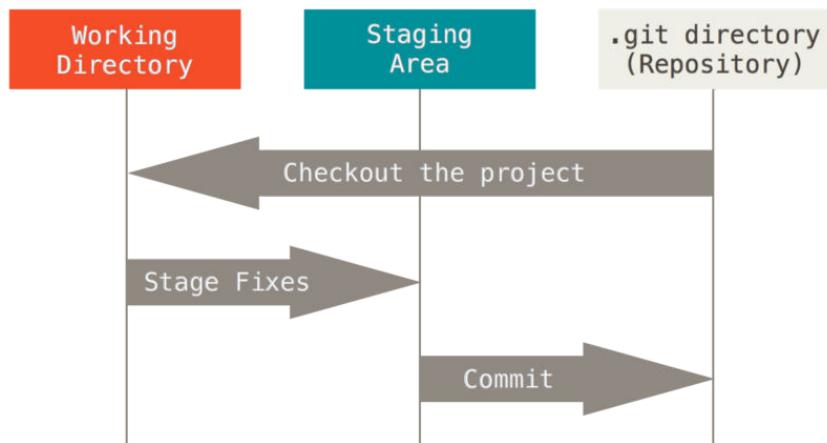
u                n

Git  
ted                  modified                  staged

Git  
commit

**FIGURE 1-6**

工作目录、暂存区域  
以及 Git 仓库。



Git                  Git                  Git

"                  "

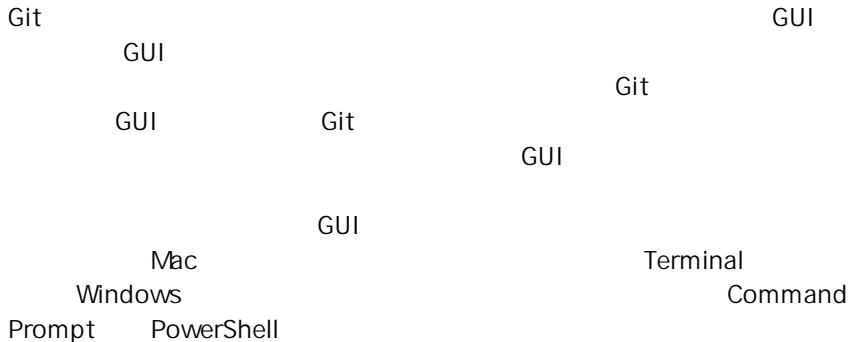
Git

- 1.
- 2.
- 3.

Git

Git

## 命令行



## 安装 Git

Git

---

本书写作时使用的 Git 版本为 2.0.0。我们使用的大部分命令仍然可以在很古老的 Git 版本上使用，但也有少部分命令不好用或者在旧版本中的行为有差异。因为 Git 在保持向后兼容方面表现很好，本书使用的这些命令在 2.0 之后的版本应该有效。

---

### Linux

Linux                                  Git  
Fedora                                 yum

```
$ sudo yum install git
```

Debian

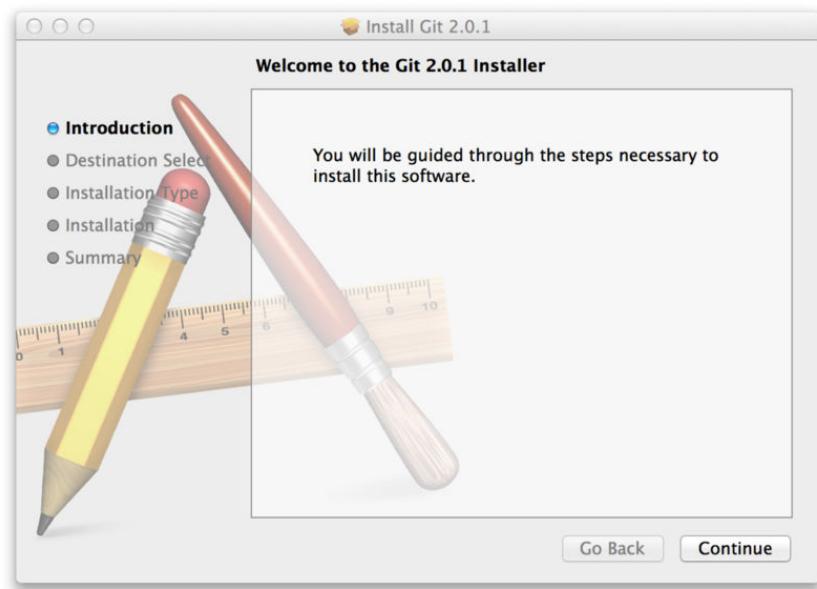
apt-get

```
$ sudo apt-get install git
```



**FIGURE 1-7**

Git OS X 安装程序.



GitHub for Mac                      Git  
GitHub for Mac  
<http://mac.github.com>

## Windows



```

$ sudo yum install curl-devel expat-devel gettext-devel \
  openssl-devel zlib-devel
$ sudo apt-get install libcurl4-gnutls-dev libexpat1-dev gettext \
  libz-dev libssl-dev
  
```

doc, html, info

```

$ sudo yum install asciidoc xmlto docbook2x
$ sudo apt-get install asciidoc xmlto docbook2x
  
```



```
$ tar -zxf git-2.0.0.tar.gz
$ cd git-2.0.0
$ make configure
$ ./configure --prefix=/usr
$ make all doc info
$ sudo make install install-doc install-html install-info
```

Git      Git

```
$ git clone git://git.kernel.org/pub/scm/git/git.git
```

## 初次运行 Git 前的配置

Git      git config      Git

1. /etc/gitconfig :  
                --system      git config
  
2. ~/.gitconfig    ~/.config/git/config  
                --global      Git
  
3.                Git      config      .git/config

.git/config

<i>/etc/gitconfig</i>	<i>Windows</i>	<i>\$HOME</i>
<i>\\$USER</i>	<i>.gitconfig</i>	<i>Git</i>
<i>MSys</i>		<i>Git</i>

Git      Git

```
$ git config --global user.name "John Doe"  
$ git config --global user.email johndoe@example.com
```

--global  
Git

--global  
GUI

Git  
Vim                            Emacs

```
$ git config --global core.editor emacs
```

---

Vim 和 Emacs 是像 Linux 与 Mac 等基于 Unix 的系统上开发者经常使用的流行的文本编辑器。如果你对这些编辑器都不是很了解或者你使用的是 Windows 系统，那么可能需要搜索如何在 Git 中配置你最常用的编辑器。如果你不设置编辑器并且不知道 Vim 或 Emacs 是什么，当它们运行起来后你可能会被弄糊涂、不知所措。

---

git config --list                            Git

```
$ git config --list  
user.name=John Doe  
user.email=johndoe@example.com  
color.status=auto  
color.branch=auto  
color.interactive=auto  
color.diff=auto  
...
```

Git

/etc/gitconfig ~/.gitconfig

Git

git config <key>

Git

```
$ git config user.name  
John Doe
```

## 获取帮助

Git

Git

```
$ git help <verb>  
$ git <verb> --help  
$ man git-<verb>
```

config

```
$ git help config
```

Freenode IRC

irc.free

node.net #git #github

Git

## 总结

Git

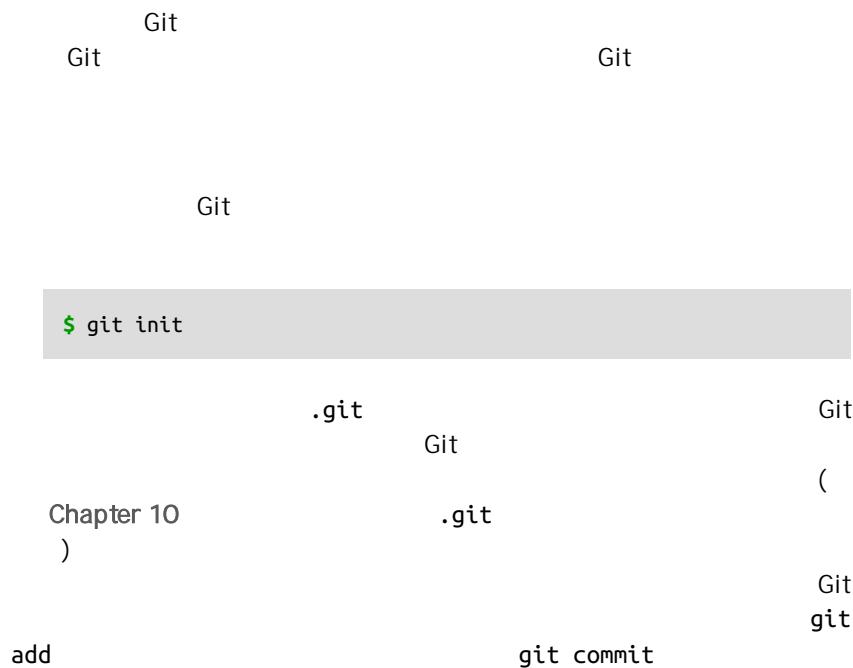
Git

Git

# Git 基础 2

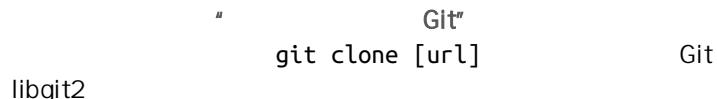
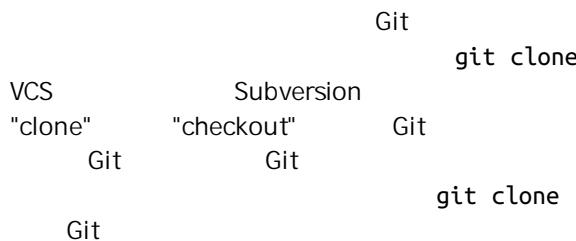


## 获取 Git 仓库

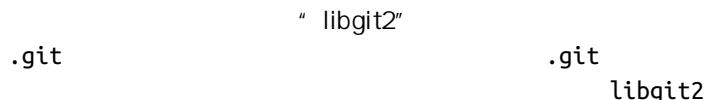


```
$ git add *.c
$ git add LICENSE
$ git commit -m 'initial project version'
```

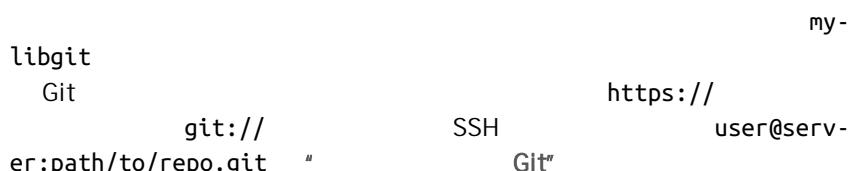
## Git



```
$ git clone https://github.com/libgit2/libgit2
```



```
$ git clone https://github.com/libgit2/libgit2 mylibgit
```

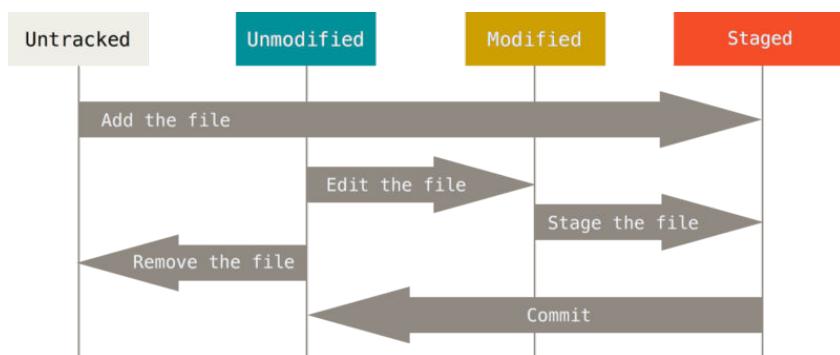


## 记录每次更新到仓库

Git

Git

Git



**FIGURE 2-1**  
文件的状态变化周期

git status

```
$ git status
On branch master
nothing to commit, working directory clean
```

Git

" master" ,

Chapter 3

```
 README  
git status
```

```
$ echo 'My Project' > README  
$ git status  
On branch master  
Untracked files:  
  (use "git add <file>..." to include in what will be committed)  
  
 README  
  
nothing added to commit but untracked files present (use "git add" to track)
```

```
 README          Untracked files  
Git             Git  
"              "  
"  
 README
```

```
git add          README
```

```
$ git add README
```

```
git status       README
```

```
$ git status  
On branch master  
Changes to be committed:  
  (use "git reset HEAD <file>..." to unstage)  
  
 new file:   README
```

Changes to be committed

git init git add (files)  
git add

TING.md

git status

CONTRIBU-

```
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file: README

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified: CONTRIBUTING.md
```

CONTRIBUTING.md

Changes not staged for commit

git add

add "CONTRIBUTING.md"

git status

git

```
$ git add CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file: README
    modified: CONTRIBUTING.md
```

```
CONTRIBUTING.md
    git status
```

```
$ vim CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   README
    modified:   CONTRIBUTING.md

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   CONTRIBUTING.md
```

```
CONTRIBUTING.md
    Git          git add
    CONTRIBUTING.md      git
add           git commit
                  git add
git add
```

```
$ git add CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   README
    modified:   CONTRIBUTING.md
```

```
git status
status -s      git status --short
                  git status -s
```

```
$ git status -s
M README
```

```
MM Rakefile
A lib/git.rb
M lib/simplegit.rb
?? LICENSE.txt
```

```
?? A
M M
M
README
,lib/simplegit.rb
Rakefile
```

Git

.gitignore

```
$ cat .gitignore
*.[oa]
*~
```

```
Git      .o   .a          Git
        Emacs
        log  tmp    pid
        .gitignore
```

.gitignore

```
•      #          Git
•      glob
•      /
•      /
•          !

```

```

glob      shell      *
[abc]      b          c      ?
a          [0-9]      0      9
*)          a/**/z    a/z,
a/b/z    a/b/c/z
.gitignore

# no .a files
*.a

# but do track lib.a, even though you're ignoring .a files above
!lib.a

# only ignore the TODO file in the current directory, not subdir/TODO
/TODO

# ignore all files in the build/ directory
build/

# ignore doc/notes.txt, but not doc/server/arch.txt
doc/*.txt

# ignore all .pdf files in the doc/ directory
doc/**/*.pdf

```

---

GitHub 有一个十分详细的针对数十种项目及语言的 `.gitignore` 文件列表，你可以在 <https://github.com/github/gitignore> 找到它。

---

```

git status
git diff           git diff

```

```

git status
git diff

```

```

README
status

```

```

CONTRIBUTING.md

```

```

$ git status
On branch master
Changes to be committed:

```

```
(use "git reset HEAD <file>..." to unstage)

modified: README

Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git checkout -- <file>..." to discard changes in working directory)

modified: CONTRIBUTING.md
```

### git diff

```
$ git diff
diff --git a/CONTRIBUTING.md b/CONTRIBUTING.md
index 8ebb991..643e24f 100644
--- a/CONTRIBUTING.md
+++ b/CONTRIBUTING.md
@@ -65,7 +65,8 @@ branch directly, things can get messy.
Please include a nice description of your changes when you submit your PR;
if we have to read the whole diff to figure out why you're contributing
in the first place, you're less likely to get feedback and have your change
-merged in.
+merged in. Also, split your changes into comprehensive chunks if your patch is
+longer than a dozen lines.
```

If you are starting to work on a particular area, feel free to submit a PR that highlights your work in progress (and note in the PR title that it's

cached                  Git 1.6.1                  git diff --  
    git diff --staged

```
$ git diff --staged
diff --git a/README b/README
new file mode 100644
index 0000000..03902a1
--- /dev/null
+++ b/README
@@ -0,0 +1 @@
+My Project
```

<pre>git dif diff CONTRIBUTING.md</pre>	<pre>git git status</pre>
---	---------------------------

```
$ git add CONTRIBUTING.md
$ echo '# test line' >> CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   CONTRIBUTING.md

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   CONTRIBUTING.md
```

**git diff**

```
$ git diff
diff --git a/CONTRIBUTING.md b/CONTRIBUTING.md
index 643e24f..87f08c8 100644
--- a/CONTRIBUTING.md
+++ b/CONTRIBUTING.md
@@ -119,3 +119,4 @@ at the
## Starter Projects

See our [projects list](https://github.com/libgit2/libgit2/blob/development/PROJE
+# test line
```

<pre>git diff --cached cached</pre>	<pre>--staged  -- --</pre>
-------------------------------------	----------------------------

```
$ git diff --cached
diff --git a/CONTRIBUTING.md b/CONTRIBUTING.md
index 8ebb991..643e24f 100644
--- a/CONTRIBUTING.md
+++ b/CONTRIBUTING.md
@@ -65,7 +65,8 @@ branch directly, things can get messy.
Please include a nice description of your changes when you submit your PR;
```

```
if we have to read the whole diff to figure out why you're contributing  
in the first place, you're less likely to get feedback and have your change  
-merged in.  
+merged in. Also, split your changes into comprehensive chunks if your patch is  
+longer than a dozen lines.
```

```
If you are starting to work on a particular area, feel free to submit a PR  
that highlights your work in progress (and note in the PR title that it's
```

---

### GIT DIFF 的插件版本

在本书中，我们使用 `git diff` 来分析文件差异。但是，如果你喜欢通过图形化的方式或其它格式输出方式的话，可以使用 `git difftool` 命令来用 Araxis，`emerge` 或 `vimdiff` 等软件输出 diff 分析结果。使用 `git difftool --tool-help` 命令来看你的系统支持哪些 Git Diff 插件。

---

```
git add
```

```
git status  
git commit
```

```
$ git commit
```

\$EDITOR Chapter 1	( shell vim emacs git config --global core.editor
-----------------------	---

Vim

```
# Please enter the commit message for your changes. Lines starting  
# with '#' will be ignored, and an empty message aborts the commit.  
# On branch master  
# Changes to be committed:  
#   new file: README  
#   modified: CONTRIBUTING.md  
#  
~  
~  
~  
.git/COMMIT_EDITMSG" 9L, 283C
```

```
git status
```

```
(          -v
dif
Git
```

```
commit      -m
```

```
$ git commit -m "Story 182: Fix benchmarks for speed"
[master 463dc4f] Story 182: Fix benchmarks for speed
 2 files changed, 2 insertions(+)
 create mode 100644 README
```

master	SHA-1
463dc4f	

```
Git
git commit      -a      Git
                  git add
```

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   CONTRIBUTING.md

no changes added to commit (use "git add" and/or "git commit -a")
$ git commit -a -m 'added new benchmarks'
[master 83e38c7] added new benchmarks
  1 file changed, 5 insertions(+), 0 deletions(-)
```

```
git add      " CONTRIBUTING.md"
```

Git

```
git rm
```

```
git status
```

" Changes not staged for commit"

```
$ rm PROJECTS.md
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    PROJECTS.md

no changes added to commit (use "git add" and/or "git commit -a")
```

```
git rm
```

```
$ git rm PROJECTS.md
rm 'PROJECTS.md'
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        deleted:    PROJECTS.md
```

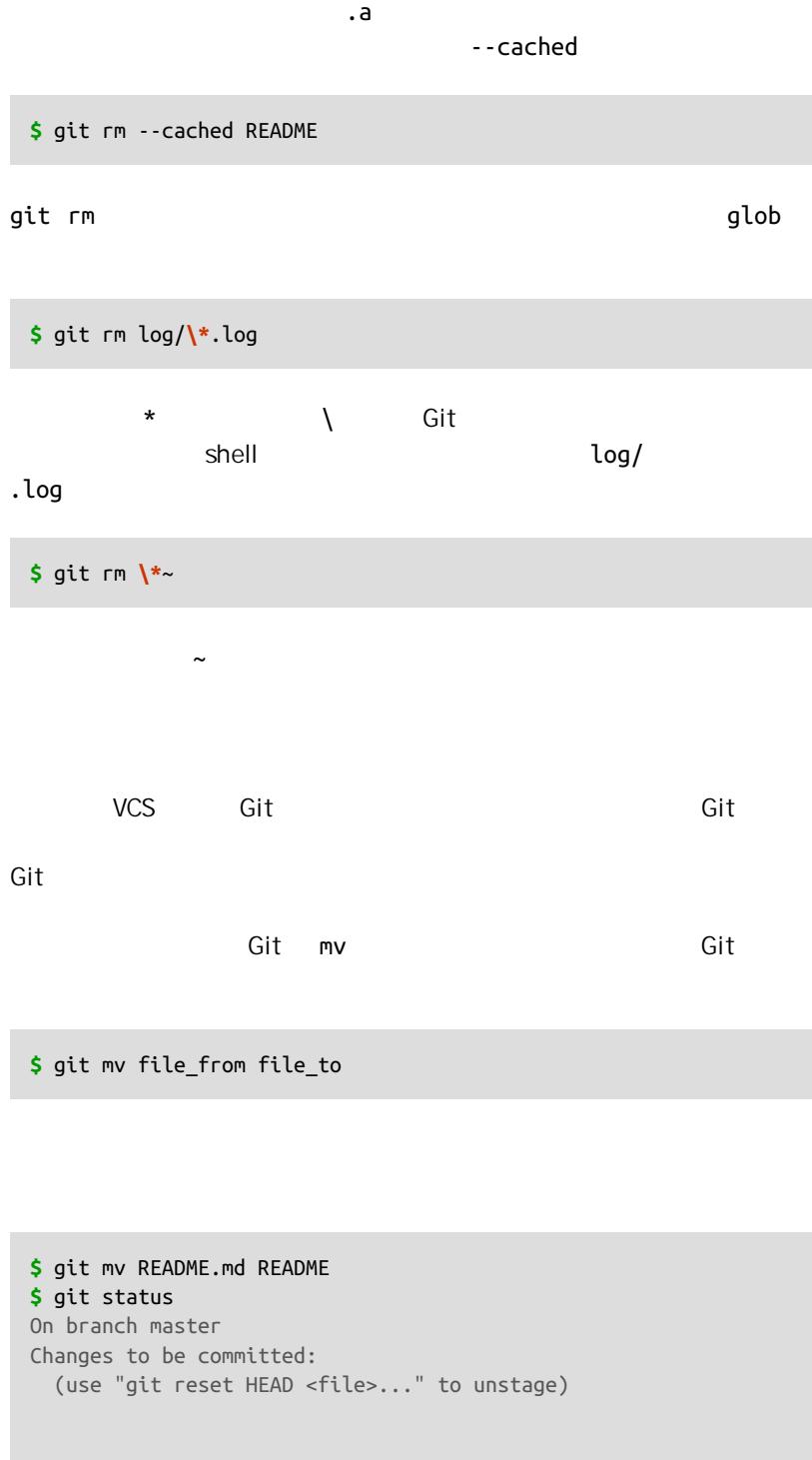
-f                force

Git

Git

Git

.gitignore



```
renamed: README.md -> README
```

```
git mv
```

```
$ mv README.md README  
$ git rm README.md  
$ git add README
```

Git

mv

```
git mv
```

## 查看提交历史

```
git log  
simplegit
```

```
git clone https://github.com/schacon/simplegit-progit
```

```
git log
```

```
$ git log  
commit ca82a6dff817ec66f44342007202690a93763949  
Author: Scott Chacon <schacon@gee-mail.com>  
Date: Mon Mar 17 21:52:11 2008 -0700
```

changed the version number

```
commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7  
Author: Scott Chacon <schacon@gee-mail.com>  
Date: Sat Mar 15 16:40:33 2008 -0700
```

removed unnecessary test

```
commit a11bef06a3f659402fe7563abf99ad00de2209e6  
Author: Scott Chacon <schacon@gee-mail.com>  
Date: Sat Mar 15 10:31:28 2008 -0700
```

first commit

git log

SHA-1

git log

-p

-2

```
$ git log -p -2
commit ca82a6dff817ec66f44342007202690a93763949
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

        changed the version number

diff --git a/Rakefile b/Rakefile
index a874b73..8f94139 100644
--- a/Rakefile
+++ b/Rakefile
@@ -5,7 +5,7 @@ require 'rake/gempackagetask'
spec = Gem::Specification.new do |s|
    s.platform = Gem::Platform::RUBY
    s.name      = "simplegit"
-   s.version   = "0.1.0"
+   s.version   = "0.1.1"
    s.author    = "Scott Chacon"
    s.email     = "schacon@gee-mail.com"
    s.summary   = "A simple gem for using Git in Ruby code."

```

```
commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

        removed unnecessary test

diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index a0a60ae..47c6340 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -18,8 +18,3 @@ class SimpleGit
end

end
-
```

```
-if $0 == __FILE__
- git = SimpleGit.new
- puts git.show
-end
\ No newline at end of file
```

```
commit
commit
git log
--stat
```

```
$ git log --stat
commit ca82a6dff817ec66f44342007202690a93763949
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

    changed the version number

Rakefile | 2 ++
1 file changed, 1 insertion(+), 1 deletion(-)

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test

lib/simplegit.rb | 5 -----
1 file changed, 5 deletions(-)

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sat Mar 15 10:31:28 2008 -0700

    first commit

README          |  6 ++++++
Rakefile        | 23 ++++++++++++++++++++++
lib/simplegit.rb | 25 ++++++++++++++++++++++
3 files changed, 54 insertions(+)
```

```
--stat
```

```
--pretty  

oneline  

short full fuller  

$ git log --pretty=oneline
ca82a6dff817ec66f44342007202690a93763949 changed the version number
085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7 removed unnecessary test
a11bef06a3f659402fe7563abf99ad00de2209e6 first commit  

format  

—  

Git
```

```
$ git log --pretty=format:"%h - %an, %ar : %s"
ca82a6d - Scott Chacon, 6 years ago : changed the version number
085bb3b - Scott Chacon, 6 years ago : removed unnecessary test
a11bef0 - Scott Chacon, 6 years ago : first commit
```

Table 2-1

**TABLE 2-1.** *git log --pretty=format* 常用的选项

---

%H	commit
%h	
%T	tree
%t	
%P	parent
%p	
%an	author
%ae	
%ad	--date=
%ar	
%cn	(committer)

---

---

```
%ce
```

```
%cd
```

```
%cr
```

```
%s
```

---

## Chapter 5

```
oneline    format      log     --graph  
          ASCII
```

```
$ git log --pretty=format:"%h %s" --graph  
* 2d3acf9 ignore errors from SIGCHLD on trap  
* 5e3ee11 Merge branch 'master' of git://github.com/dustin/grit  
|\  
| * 420eac9 Added a method for getting the current branch.  
* | 30e367c timeout code and tests  
* | 5a09431 add timeout protection to grit  
* | e1193f8 support for heads with slashes in them  
|/  
* d6016bc require time for xschema  
* 11d191e Merge branch 'defunkt' into local
```

git log

Table 2-2  
log

**TABLE 2-2.** *git log* 的常用选项

---

---

```
-p
```

```
--stat
```

```
--shortstat      --stat
```

```
--name-only
```

---



---

--name-status					
--abbrev-commit	SHA-1				40
--relative-date				" 2 weeks ago"	
--graph	ASCII				
--pretty	line short full fuller	format			one

---

git log  
-2  
-<n> n

Git  
--since --until

```
$ git log --since=2.weeks
```

"2008-01-15"  
"2 years 1 day 3 minutes ago"  
--author  
--grep  
--all-match  
-S

```
$ git log -Sfunction_name
```

git log (path)  
git log  
--

Table 2-3

**TABLE 2-3.** 限制`git log`输出的选项

-n	n
--since, --after	
--until, --before	
--author	
--committer	
--grep	
-S	

Git 2008 10 Ju  
nio Hamano

```
$ git log --pretty="%h - %s" --author=gitster --since="2008-10-01" \  
  --before="2008-11-01" --no-merges -- t/  
5610e3b - Fix testcase failure when extended attributes are in use  
acd3b9e - Enhance hold_lock_file_for_{update,append}() API  
f563754 - demonstrate breakage of detached checkout with symbolic link HEAD  
d1a43f2 - reset --hard/read-tree --reset -u: remove unmerged new paths  
51a94af - Fix "checkout --track -b newbranch" on detached HEAD  
b0ad11e - pull: allow "git pull origin $something:$current_branch" into an unborn branch
```

40000 6

## 撤销操作

Git

--amend

```
$ git commit --amend
```

```
$ git commit -m 'initial commit'  
$ git add forgotten_file  
$ git commit --amend
```

git add \*  
status

```
$ git add *  
$ git status  
On branch master  
Changes to be committed:  
(use "git reset HEAD <file>..." to unstage)  
  
    renamed: README.md -> README  
    modified: CONTRIBUTING.md
```

“ Changes to be committed”  
HEAD <file>...  
TING.md

```
$ git reset HEAD CONTRIBUTING.md  
Unstaged changes after reset:  
M      CONTRIBUTING.md  
$ git status  
On branch master  
Changes to be committed:  
(use "git reset HEAD <file>..." to unstage)
```

```
renamed: README.md -> README  
  
Changes not staged for commit:  
(use "git add <file>..." to update what will be committed)  
(use "git checkout -- <file>..." to discard changes in working directory)  
  
modified: CONTRIBUTING.md
```

## CONTRIBUTING.md

---

虽然在调用时加上 `--hard` 选项可以令 `git reset` 成为一个危险的命令（译注：可能导致工作目录中所有当前进度丢失！），但本例中工作目录内的文件并不会被修改。不加选项地调用 `git reset` 并不危险 — 它只会修改暂存区域。

---

```
git reset  
" " reset
```

## CONTRIBUTING.md

```
-  
git status
```

```
Changes not staged for commit:  
(use "git add <file>..." to update what will be committed)  
(use "git checkout -- <file>..." to discard changes in working directory)  
  
modified: CONTRIBUTING.md
```

```
$ git checkout -- CONTRIBUTING.md  
$ git status  
On branch master  
Changes to be committed:  
(use "git reset HEAD <file>..." to unstage)
```

```
renamed: README.md -> README
```

---

你需要知道 `git checkout -- [file]` 是一个危险的命令，这很重要。你对那个文件做的任何修改都会消失 – 你只是拷贝了另一个文件来覆盖它。除非你确实清楚不想要那个文件了，否则不要使用这个命令。

---

## Chapter 3

### Git

`--amend`

"

"

## 远程仓库的使用

### Git

`git remote`

origin - Git

```
$ git clone https://github.com/schacon/ticgit
Cloning into 'ticgit'...
remote: Reusing existing pack: 1857, done.
remote: Total 1857 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (1857/1857), 374.35 KiB | 268.00 KiB/s, done.
Resolving deltas: 100% (772/772), done.
Checking connectivity... done.
$ cd ticgit
```

```
$ git remote  
origin
```

-v

Git

URL

```
$ git remote -v  
origin https://github.com/schacon/ticgit (fetch)  
origin https://github.com/schacon/ticgit (push)
```

```
$ cd grit  
$ git remote -v  
bakkdoor https://github.com/bakkdoor/grit (fetch)  
bakkdoor https://github.com/bakkdoor/grit (push)  
cho45 https://github.com/cho45/grit (fetch)  
cho45 https://github.com/cho45/grit (push)  
defunkt https://github.com/defunkt/grit (fetch)  
defunkt https://github.com/defunkt/grit (push)  
koke git://github.com/koke/grit.git (fetch)  
koke git://github.com/koke/grit.git (push)  
origin git@github.com:mojombo/grit.git (fetch)  
origin git@github.com:mojombo/grit.git (push)
```

"

Git"

```
<url>          Git           git remote add <shortname>
```

```
$ git remote  
origin  
$ git remote add pb https://github.com/paulboone/ticgit  
$ git remote -v  
origin https://github.com/schacon/ticgit (fetch)
```

```
origin  https://github.com/schacon/ticgit (push)
pb      https://github.com/paulboone/ticgit (fetch)
pb      https://github.com/paulboone/ticgit (push)
```

Paul                    pb                    URL  
                        git fetch pb

```
$ git fetch pb
remote: Counting objects: 43, done.
remote: Compressing objects: 100% (36/36), done.
remote: Total 43 (delta 10), reused 31 (delta 5)
Unpacking objects: 100% (43/43), done.
From https://github.com/paulboone/ticgit
 * [new branch] master      -> pb/master
 * [new branch] ticgit     -> pb/ticgit
```

Paul      master                    pb/master      -

### Chapter 3

```
$ git fetch [remote-name]
```

clone                    git fetch origin  
   " origin"                    git fetch  
                        -

Chapter 3  
                        git pull

git clone                    master  
   master                    git pull

```
git  
push [remote-name] [branch-name] master  
origin
```

```
$ git push origin master
```

## Chapter 3

```
git remote show  
[remote-name]  
origin
```

```
$ git remote show origin  
* remote origin  
  Fetch URL: https://github.com/schacon/ticgit  
  Push URL: https://github.com/schacon/ticgit  
  HEAD branch: master  
  Remote branches:  
    master                  tracked  
    dev-branch              tracked  
  Local branch configured for 'git pull':  
    master merges with remote master  
  Local ref configured for 'git push':  
    master pushes to master (up to date)
```

URL		Git
master		git pull
master		master
git remote show		

```
$ git remote show origin
* remote origin
  URL: https://github.com/my-org/complex-project
  Fetch URL: https://github.com/my-org/complex-project
  Push URL: https://github.com/my-org/complex-project
  HEAD branch: master
  Remote branches:
    master                  tracked
    dev-branch              tracked
    markdown-strip          tracked
    issue-43                new (next fetch will store in remotes/origin)
    issue-45                new (next fetch will store in remotes/origin)
    refs/remotes/origin/issue-11  stale (use 'git remote prune' to remove)
  Local branches configured for 'git pull':
    dev-branch merges with remote dev-branch
    master     merges with remote master
  Local refs configured for 'git push':
    dev-branch      pushes to dev-branch
    markdown-strip  pushes to markdown-strip
    master         pushes to master
(up to date)
(up to date)
(up to date)
```

**git push****git pull**

git remote rename  
pb paul git remote re-  
name

```
$ git remote rename pb paul
$ git remote
origin
paul
```

pb/  
master paul/master

**git remote rm**

```
$ git remote rm paul  
$ git remote  
origin
```

## 打标签

VCS      Git

v1.0

Git

git tag

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

Git  
500      1.8.5

```
$ git tag -l 'v1.8.5*'  
v1.8.5  
v1.8.5-rc0  
v1.8.5-rc1  
v1.8.5-rc2  
v1.8.5-rc3  
v1.8.5.1  
v1.8.5.2  
v1.8.5.3  
v1.8.5.4  
v1.8.5.5
```

Git  
notated

lightweight      an

## Git

GNU Privacy Guard GPG

Git tag  
-a

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

-m  
Git  
git show

```
$ git show v1.4  
tag v1.4  
Tagger: Ben Straub <ben@straub.cc>  
Date:   Sat May 3 20:19:12 2014 -0700  
  
my version 1.4  
  
commit ca82a6dff817ec66f44342007202690a93763949  
Author: Scott Chacon <schacon@gee-mail.com>  
Date:   Mon Mar 17 21:52:11 2008 -0700  
  
changed the version number
```

-a -s -m

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

### git show

```
$ git show v1.4-lw
commit ca82a6dff817ec66f44342007202690a93763949
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

changed the version number
```

```
$ git log --pretty=oneline
15027957951b64cf874c3557a0f3547bd83b3ff6 Merge branch 'experiment'
a6b4c97498bd301d84096da251c98a07c7723e65 beginning write support
0d52aaab4479697da7686c15f77a3d64d9165190 one more thing
6d52a271eda8725415634dd79daabbc4d9b6008e Merge branch 'experiment'
0b7434d86859cc7b8c3d5e1dddfe66ff742fcbe added a commit function
4682c3261057305bdd616e23b64b0857d832627b added a todo file
166ae0c4d3f420721acbb115cc33848dfcc2121a started write support
9fce02d0ae598e95dc970b74767f19372d61af8 updated rakefile
964f16d36dfccde844893cac5b347e7b3d44abbc commit the todo
8a5cbc430f1a9c3d00faaeffd07798508422908a updated readme
```

```
v1.2                                     " updated rake
file"
```

:

```
$ git tag -a v1.2 9fce02
```

```
$ 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
...
```

**git push**

- **git push origin [tagname]**

```
$ git push origin v1.5
Counting objects: 14, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (14/14), 2.05 KiB | 0 bytes/s, done.
Total 14 (delta 3), reused 0 (delta 0)
To git@github.com:schacon/simplegit.git
 * [new tag]           v1.5 -> v1.5
```

**push** **--tags** **git**

```
$ git push origin --tags
Counting objects: 1, done.
Writing objects: 100% (1/1), 160 bytes | 0 bytes/s, done.
Total 1 (delta 0), reused 0 (delta 0)
To git@github.com:schacon/simplegit.git
```

```
* [new tag]          v1.4 -> v1.4
* [new tag]          v1.4-lw -> v1.4-lw
```

Git

git

checkout -b [branchname] [tagname]

```
$ git checkout -b version2 v2.0.0
Switched to a new branch 'version2'
```

version2	
version2	v2.0.0

## Git 别名

Git

Git

Git

Git

git config

```
$ git config --global alias.co checkout
$ git config --global alias.br branch
$ git config --global alias.ci commit
$ git config --global alias.st status
```

git commit

git ci

Git

Git

```
$ git config --global alias.unstage 'reset HEAD --'
```

```
$ git unstage fileA  
$ git reset HEAD -- fileA
```

last

```
$ git config --global alias.last 'log -1 HEAD'
```

```
$ git last  
commit 66938dae3329c7aebe598c2246a8e6af90d04646  
Author: Josh Goebel <dreamer3@example.com>  
Date:   Tue Aug 26 19:48:51 2008 +0800  
  
        test for current head  
  
        Signed-off-by: Scott Chacon <schacon@example.com>
```

Git  
Git  
!  
git visual      gitk

```
$ git config --global alias.visual '!gitk'
```

## 总结

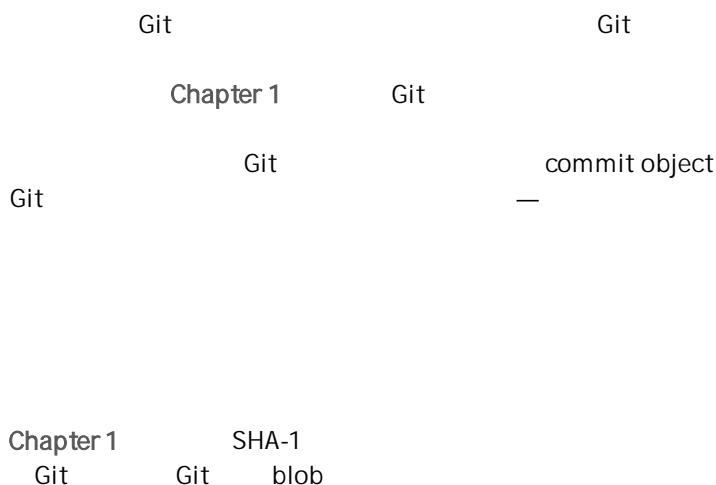
Git

Git

# Git 分支 3



## 分支简介



```
$ git add README test.rb LICENSE
$ git commit -m 'The initial commit of my project'
```

git commit

Git

Git

Git

Git

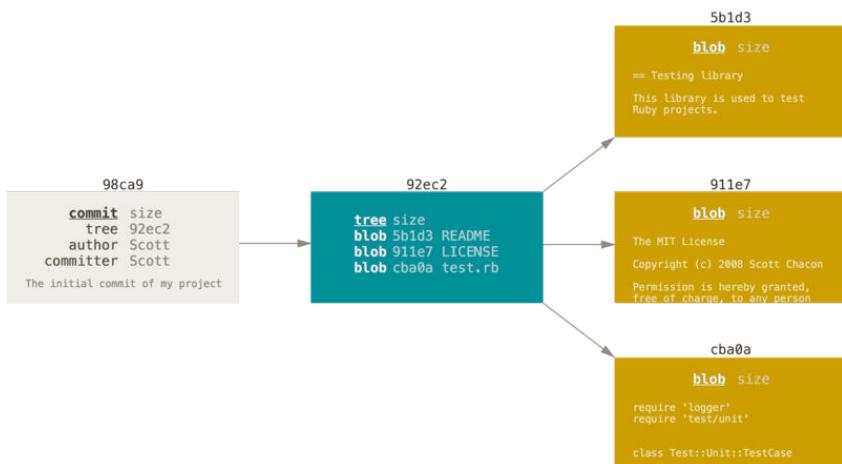
Git

blob

blob

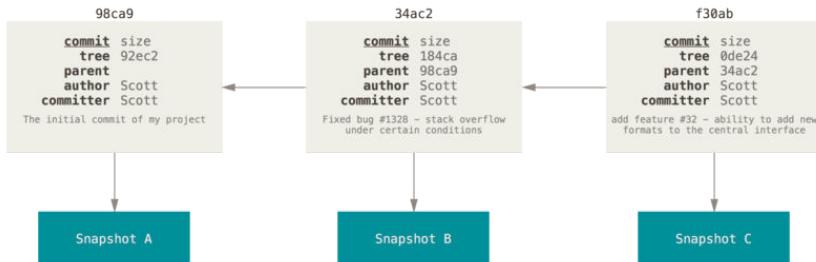
**FIGURE 3-1**

首次提交对象及其树  
结构



**FIGURE 3-2**

提交对象及其父对象



Git

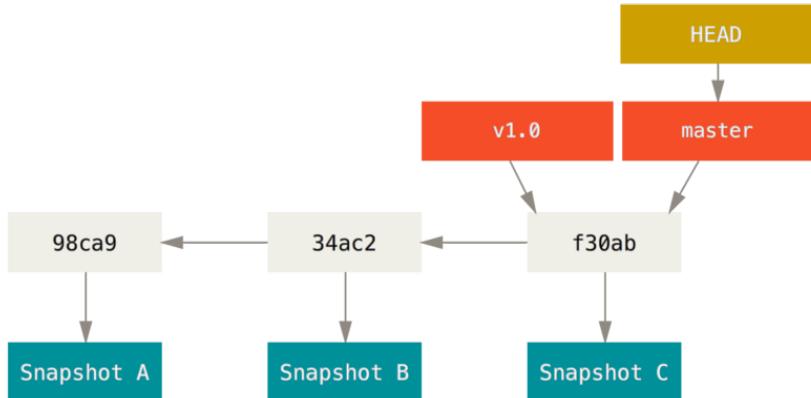
master  
master

Git 的 “master” 分支并不是一个特殊分支。它就跟其它分支完全没有区别。之所以几乎每一个仓库都有 master 分支，是因为 git init 命令默认创建它，并且大多数人都懒得去改动它。

Git

**FIGURE 3-3**

分支及其提交历史



Git

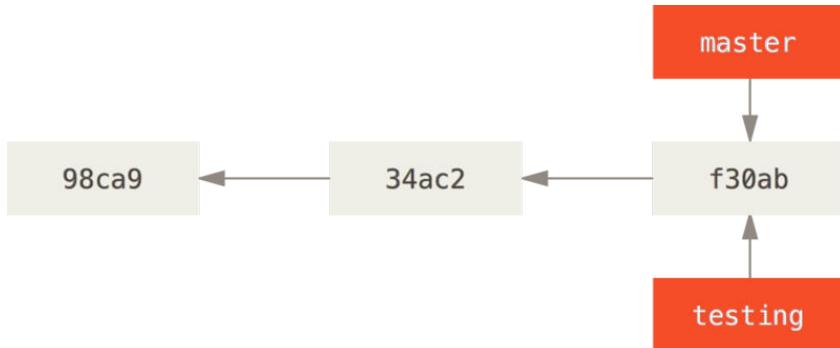
testing

git branch

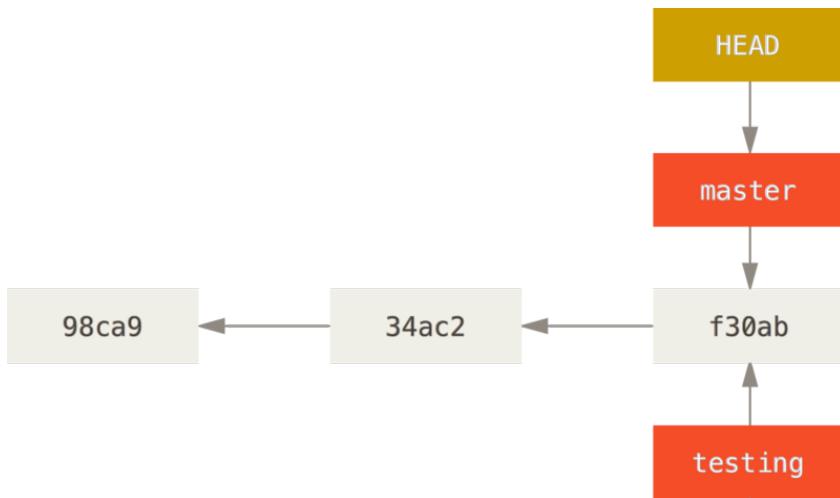
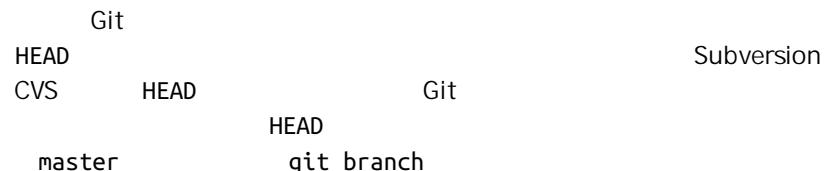
```
$ git branch testing
```

**FIGURE 3-4**

两个指向相同提交历史的分支

**FIGURE 3-5**

HEAD 指向当前所在的分支



```
git log  
--decorate
```

```
$ git log --oneline --decorate  
f30ab (HEAD, master, testing) add feature #32 - ability to add new  
34ac2 fixed bug #1328 - stack overflow under certain conditions  
98ca9 initial commit of my project
```

f30ab                    " master"  
                          " testing"

```
git checkout  
testing
```

```
$ git checkout testing
```

HEAD                    testing

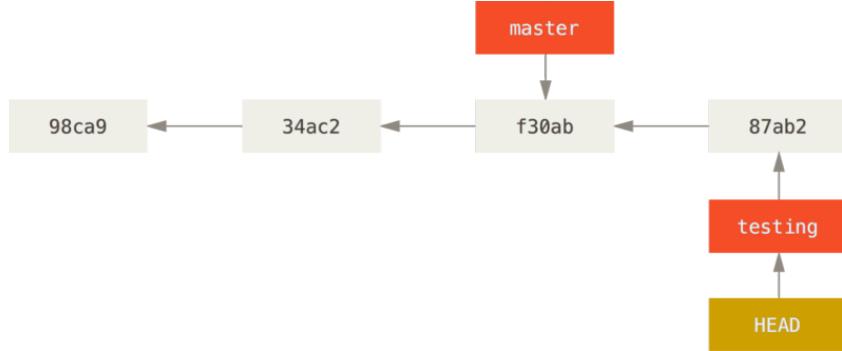


**FIGURE 3-6**  
HEAD 指向当前所在的分支

```
$ vim test.rb
$ git commit -a -m 'made a change'
```

**FIGURE 3-7**

HEAD 分支隨着提交  
操作自動向前移動

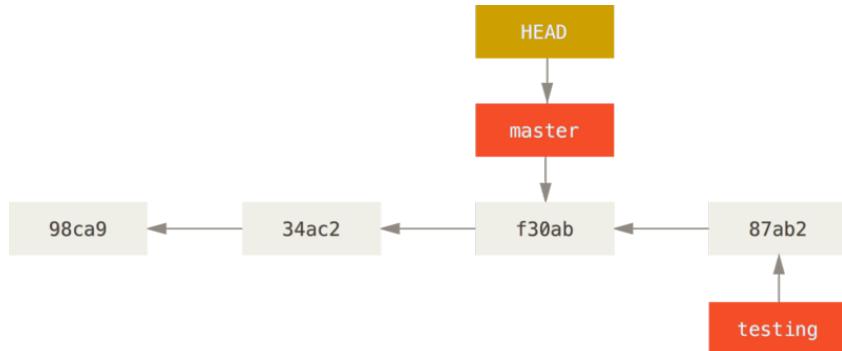


testing  
git checkout  
master

```
$ git checkout master
```

**FIGURE 3-8**

檢出時 HEAD 隨之移  
動



HEAD  
master  
master

```
testing
```

---

## 分支切换会改变你工作目录中的文件

在切换分支时，一定要注意你工作目录里的文件会被改变。如果是切换到一个较旧的分支，你的工作目录会恢复到该分支最后一次提交时的样子。如果 Git 不能干净利落地完成这个任务，它将禁止切换分支。

---

```
$ vim test.rb  
$ git commit -a -m 'made other changes'
```

Figure 3-9

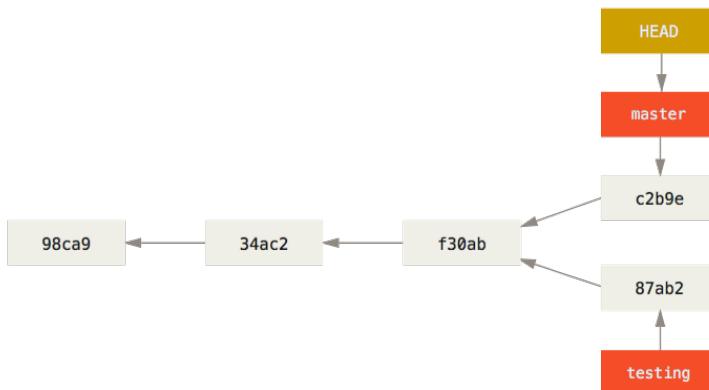
mas

ter

branch checkout commit

FIGURE 3-9

项目分叉历史



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

```
git log --
```

```
$ git log --oneline --decorate --graph --all
* c2b9e (HEAD, master) made other changes
| * 87ab2 (testing) made a change
|/
* f30ab add feature #32 - ability to add new formats to the
* 34ac2 fixed bug #1328 - stack overflow under certain conditions
* 98ca9 initial commit of my project
```

Git

40 SHA-1

41

40

1

Git

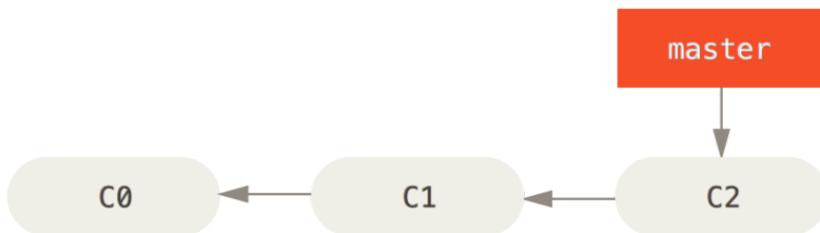
Git

## 分支的新建与合并

- 1.
- 2.
- 3.

1. production branch
- 2.
- 3.
- 4.

FIGURE 3-10



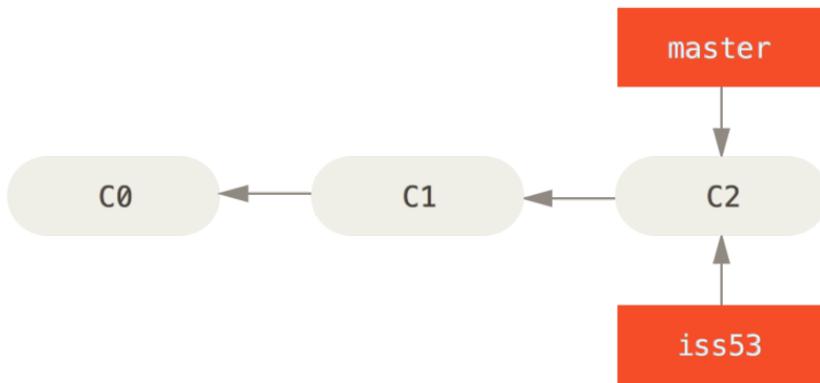
一个简单提交历史

```
#53  
-b  
git checkout
```

```
$ git checkout -b iss53  
Switched to a new branch "iss53"
```

```
$ git branch iss53  
$ git checkout iss53
```

FIGURE 3-11



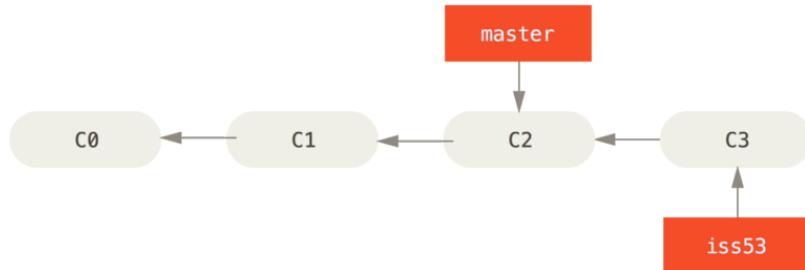
创建一个新分支指针

#53  
iss53iss53  
HEAD

```
$ vim index.html
$ git commit -a -m 'added a new footer [issue 53]'
```

**FIGURE 3-12**

iss53 分支随着工作的进展向前推进



Git  
iss53  
53#  
master

Git  
stashing  
commit amending

master

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

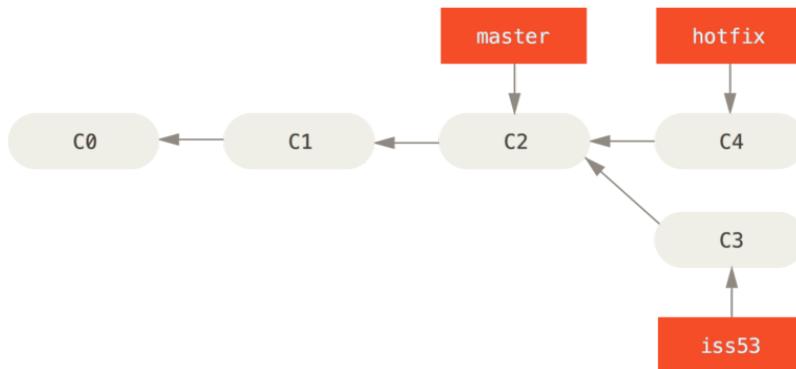
#53  
Git  
Git

hotfix branch

```
$ git checkout -b hotfix
Switched to a new branch 'hotfix'
$ vim index.html
$ git commit -a -m 'fixed the broken email address'
[hotfix 1fb7853] fixed the broken email address
1 file changed, 2 insertions(+)
```

FIGURE 3-13

基于 master 分支的  
紧急问题分支 hotfix  
branch



master

git merge

```
$ git checkout master
$ git merge hotfix
Updating f42c576..3a0874c
Fast-forward
 index.html | 2 ++
 1 file changed, 2 insertions(+)
```

master

Git

" fast-forward "

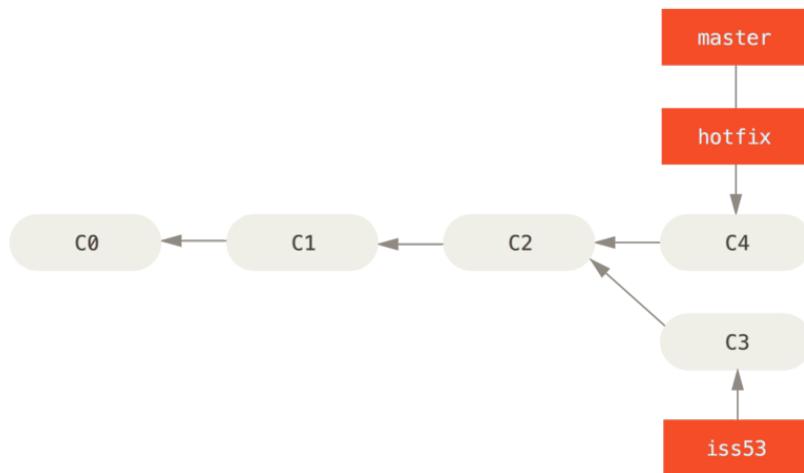
hotfix

Git

" fast-forward "

**FIGURE 3-14**

*master* 被快进到  
*hotfix*



hotfix  
master —  
-d git branch

```
$ git branch -d hotfix
Deleted branch hotfix (3a0874c).
```

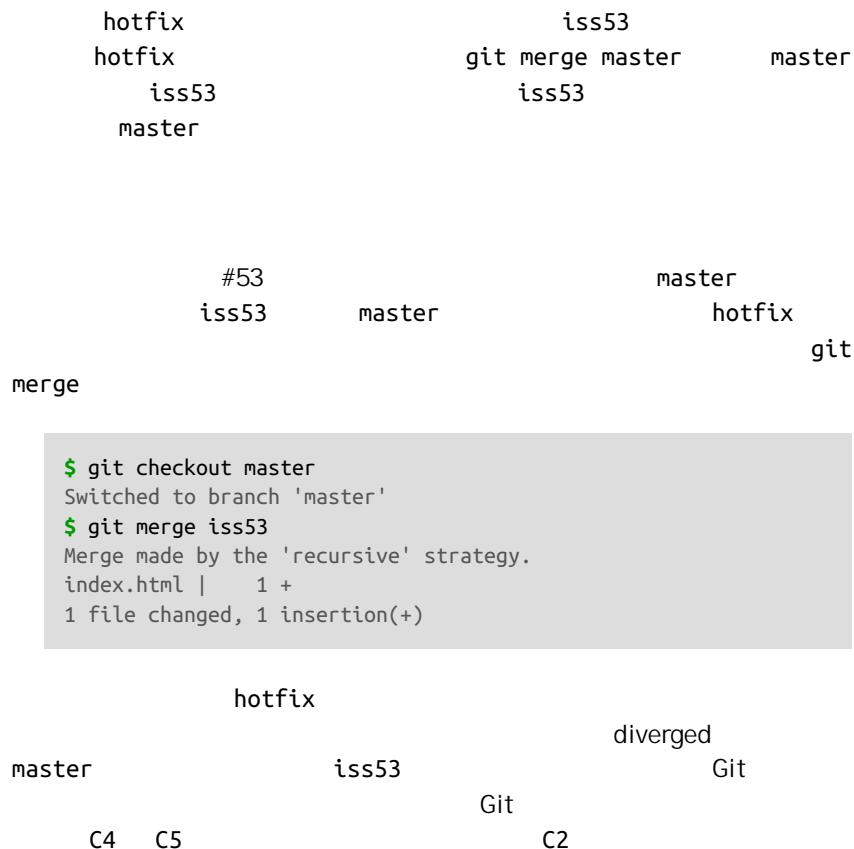
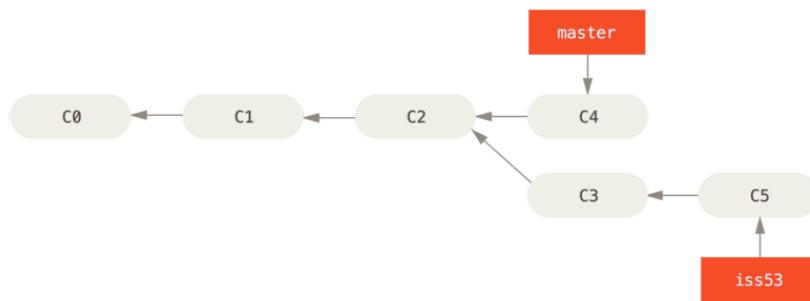
#53

iss53

```
$ git checkout iss53
Switched to branch "iss53"
$ vim index.html
$ git commit -a -m 'finished the new footer [issue 53]'
[iss53 ad82d7a] finished the new footer [issue 53]
1 file changed, 1 insertion(+)
```

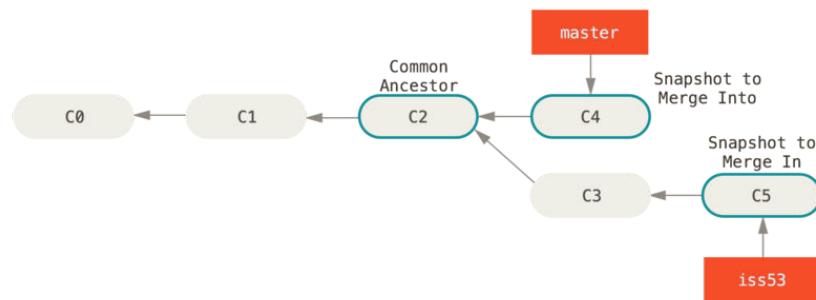
**FIGURE 3-15**

继续在 iss53 分支上的工作



**FIGURE 3-16**

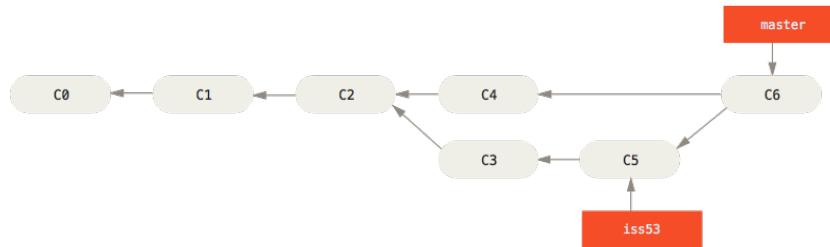
一次典型合并中所用到的三个快照



Git

**FIGURE 3-17**

一个合并提交



Git

CVS

Subversion 1.5

Git

iss53

```
$ git branch -d iss53
```

Git

#53 hotfix

```
$ git merge iss53
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

Git

Git

git status  
merged

un

```
$ git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")

Unmerged paths:
  (use "git add <file>..." to mark resolution)

    both modified:    index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

Git

```
<<<<< HEAD:index.html
<div id="footer">contact : email.support@github.com</div>
=====
<div id="footer">
  please contact us at support@github.com
</div>
>>>>> iss53:index.html
```

HEAD

master

merge

```
===== iss53 =====
=====

<div id="footer">
please contact us at email.support@github.com
</div>

<<<<<< ,
===== , >>>>>
git add
Git
git mergetool
```

```
$ git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff kdifff3 tkdiff xxdiff meld tortoisemerge gvimdiff diffuse diffmerge ecmerge
Merging:
index.html

Normal merge conflict for 'index.html':
{local}: modified file
{remote}: modified file
Hit return to start merge resolution tool (opendiff):
```

```
Git      opendiff
Mac      "
one of the following tools "
```

---

如果你需要更加高级的工具来解决复杂的合并冲突，我们会在“高级合并”介绍更多关于分支合并的内容。

---

```
Git
Git      git status
```

```
$ git status
On branch master
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:

  modified:   index.html
```

```
git commit
```

```
Merge branch 'iss53'

Conflicts:
  index.html
#
# It looks like you may be committing a merge.
# If this is not correct, please remove the file
#       .git/MERGE_HEAD
# and try again.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# All conflicts fixed but you are still merging.
#
# Changes to be committed:
#       modified:   index.html
#
```

## 分支管理

```
git branch
```

```
$ git branch
  iss53
* master
  testing
```

```
master      *
HEAD
git branch -v
```

```
$ git branch -v
  iss53  93b412c fix javascript issue
* master  7a98805 Merge branch 'iss53'
  testing 782fd34 add scott to the author list in the readmes
```

--merged --no-merged

git branch --merged

```
$ git branch --merged
  iss53
* master
```

iss53
\* git branch -d

git branch --no-merged

```
$ git branch --no-merged
  testing
```

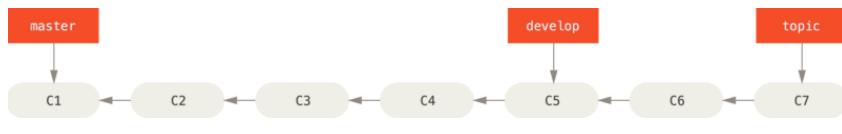
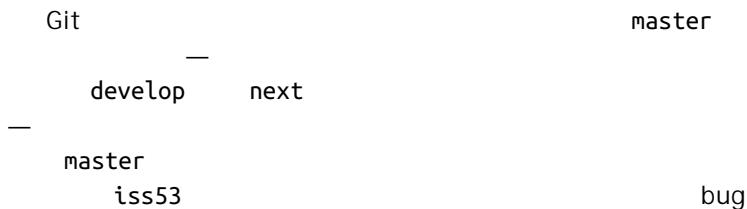
git
branch -d

```
$ git branch -d testing
error: The branch 'testing' is not fully merged.
If you are sure you want to delete it, run 'git branch -D testing'.
```

-D

## 分支开发工作流

Git



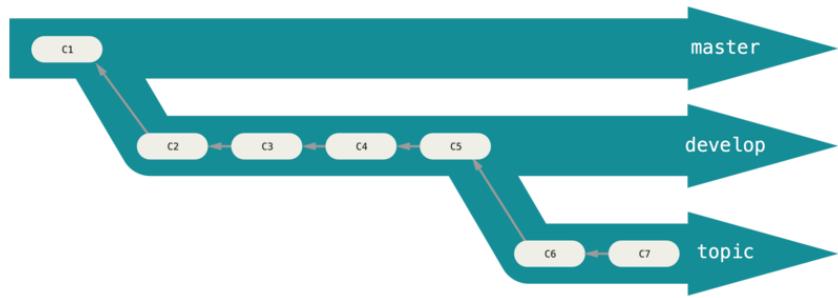
**FIGURE 3-18**

渐进稳定分支的线性图

work silos

**FIGURE 3-19**

渐进稳定分支的流水线 (“silo”) 视图



posed                      pu: proposed updates  
                              next              master                      pro-

VCS

Git

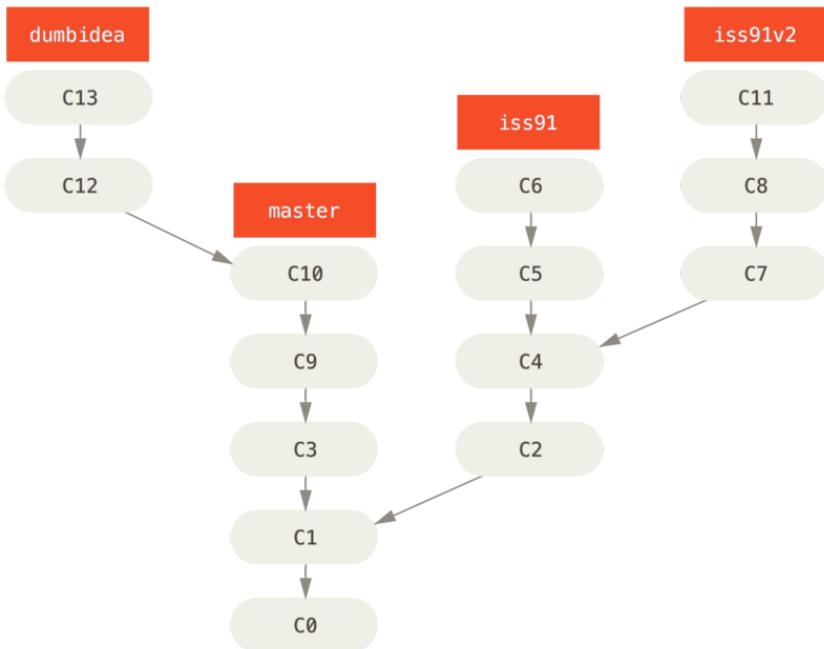
iss53    hotfix  
iss53    hotfix

context-switch —

iss91                      master                      C1  
                              iss91                      C4  
                              master                      iss91v2  
                              C10                          dumbidea

**FIGURE 3-20**

拥有多个特性分支的  
提交历史



iss91v2

dumbidea

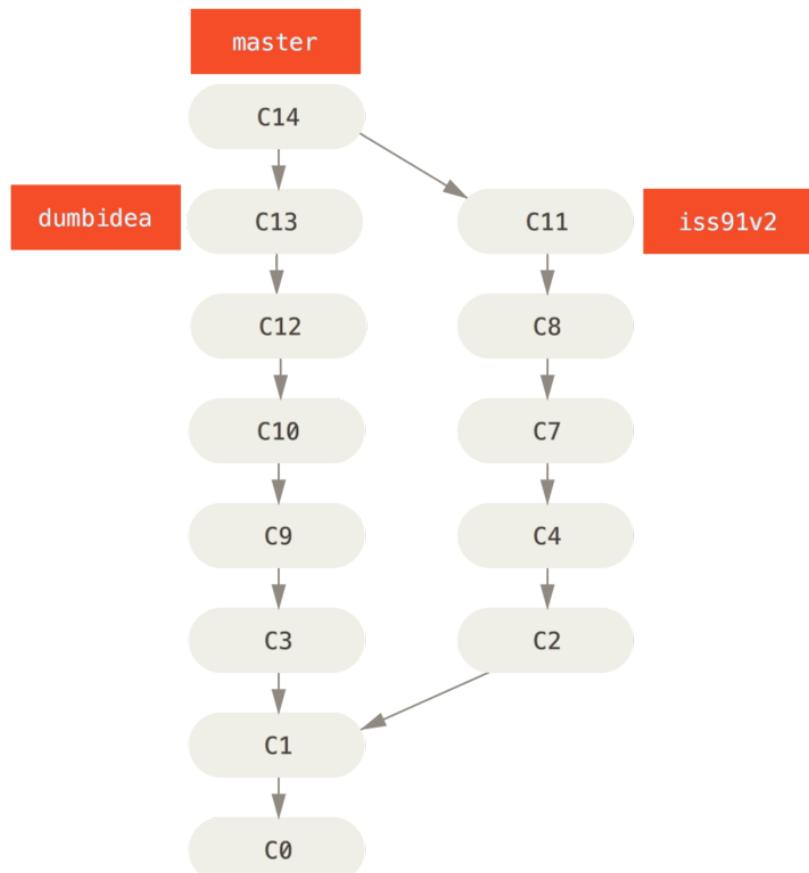
iss91

C5

C6

FIGURE 3-21

合并了 dumbidea 和 iss91v2 分支之后的提交历史



## Chapter 5

branching scheme

Git

—

## 远程分支

git ls-remote (remote)

```
git remote show (remote)
```

```
(remote)/(branch)
      origin      master          origin/
master           iss53        iss53
                  iss53          ori-
gin/iss53

git.ourcompany.com   Git
clone              origin
      master
Git                origin  master  origin/master
                           master
```

---

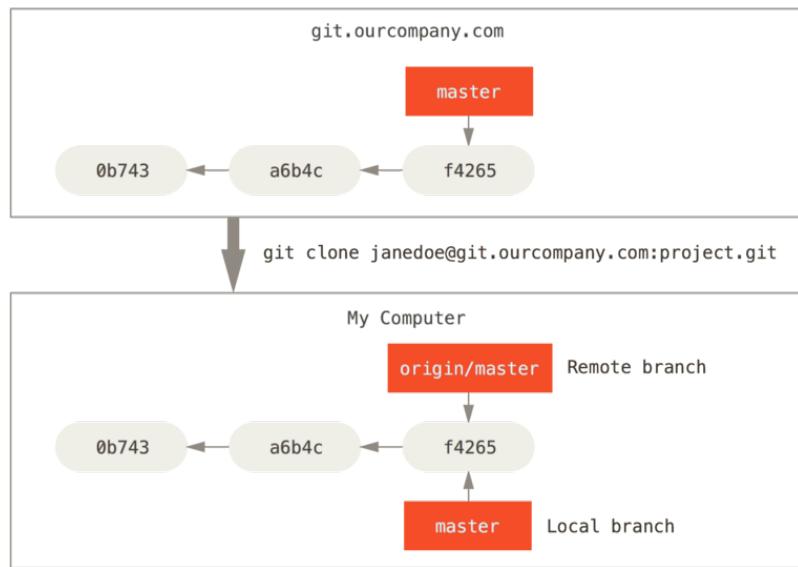
### “ORIGIN” 并无特殊含义

远程仓库名字 “origin” 与分支名字 “master” 一样，在 Git 中并没有任何特别的含义一样。同时 “master” 是当你运行 `git init` 时默认的起始分支名字，原因仅仅是它的广泛使用，“origin” 是当你运行 `git clone` 时默认的远程仓库名字。如果你运行 `git clone -o booyah`，那么你默认的远程分支名字将会是 `booyah/master`。

---

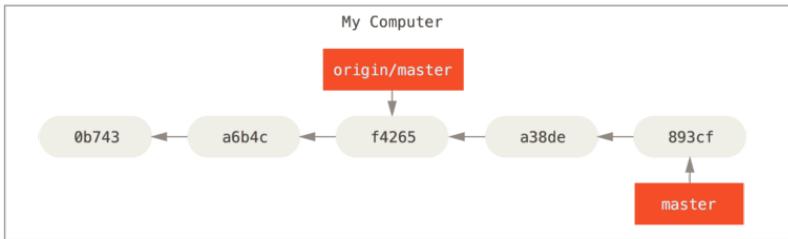
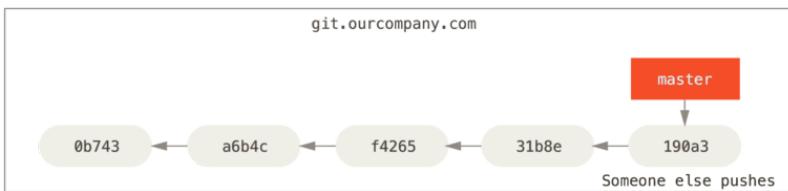
**FIGURE 3-22**

克隆之后的服务器与本地仓库



**FIGURE 3-23**

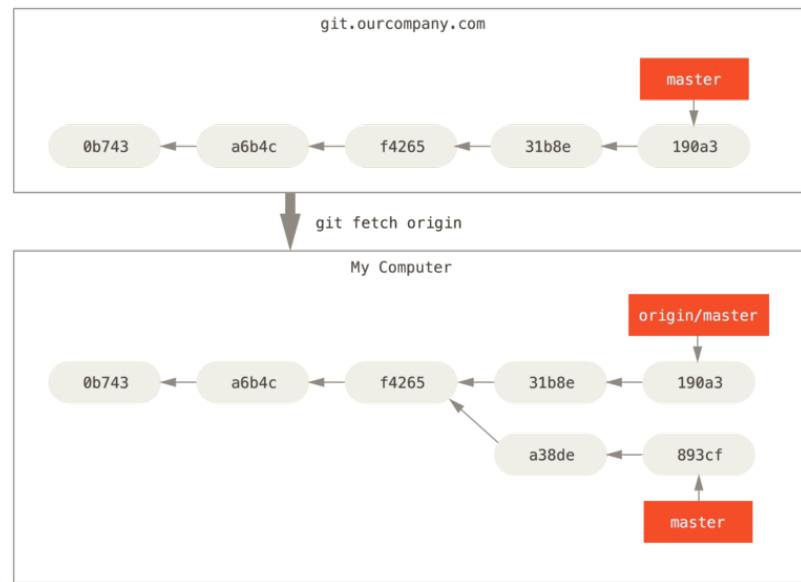
本地与远程的工作可以分支



git fetch origin  
" origin"  
git.ourcompany.com  
origin/master

**FIGURE 3-24**

*git fetch* 更新你的  
远程仓库引用



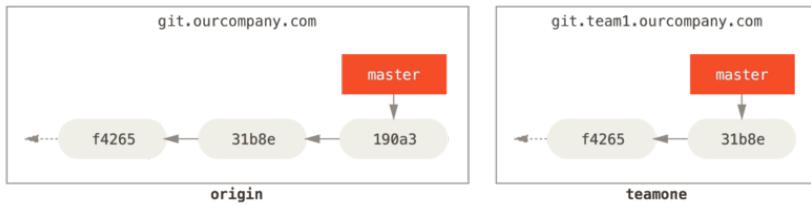
Git  
git.team1.ourcompany.com

sprint  
teamone

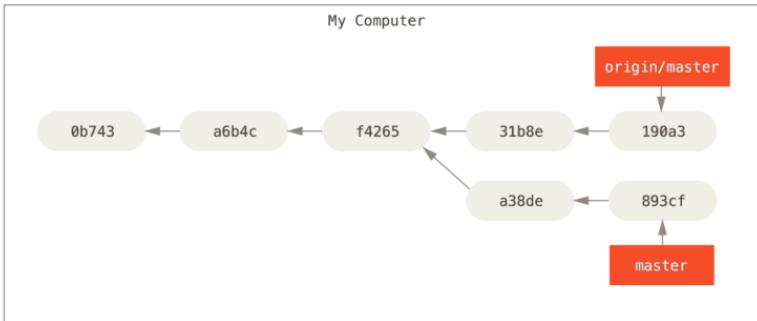
git remote add  
Chapter 2  
URL

**FIGURE 3-25**

添加另一个远程仓库



git remote add teamone git://git.team1.ourcompany.com



git fetch teamone

teamone

Git

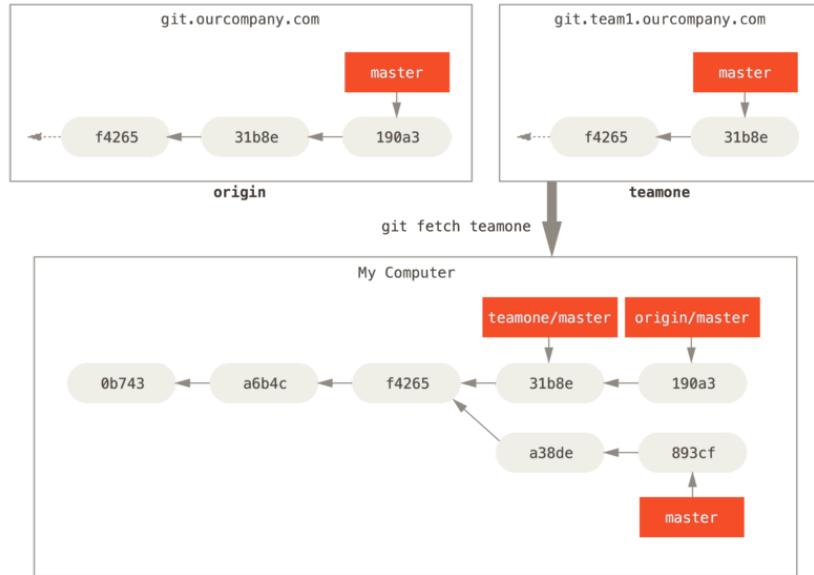
teamone master

origin

teamone/master

**FIGURE 3-26**

远程跟踪分支  
*teamone/master*



*serverfix*  
git push (remote) (branch):

```
$ git push origin serverfix
Counting objects: 24, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (15/15), done.
Writing objects: 100% (24/24), 1.91 KiB | 0 bytes/s, done.
Total 24 (delta 2), reused 0 (delta 0)
To https://github.com/schacon/simplegit
 * [new branch]      serverfix -> serverfix
```

Git	serverfix	refs/
heads/serverfix:refs/heads/serverfix		"
serverfix	serverfix	"
Chapter 10 refs/heads/		

```
git push origin serverfix:serverfix
      "           -           "
      serverfix          serverfix
serverfix:awesomebranch      git push origin
                           serverfix
awesomebranch
```

---

### 如何避免每次输入密码

如果你正在使用 HTTPS URL 来推送，Git 服务器会询问用户名与密码。默认情况下它会在终端中提示服务器是否允许你进行推送。

如果不想在每一次推送时都输入用户名与密码，你可以设置一个“credential cache”。最简单的方式就是将其保存在内存中几分钟，可以简单地运行 `git config --global credential.helper cache` 来设置它。

想要了解更多关于不同验证缓存的可用选项，查看“[凭证存储](#)”。

---

```
origin/serverfix          serverfix
```

```
$ git fetch origin
remote: Counting objects: 7, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0)
Unpacking objects: 100% (3/3), done.
From https://github.com/schacon/simplegit
 * [new branch]      serverfix    -> origin/serverfix
```

```
serverfix      -          origin/serverfix
               git merge origin/serverfix
                           serverfix
```

```
$ git checkout -b serverfix origin/serverfix
Branch serverfix set up to track remote branch serverfix from origin.
Switched to a new branch 'serverfix'
```

```
origin/serverfix
```

```

          "      "
          "      "
git pull  Git

origin/master
master
          master
git checkout -b [branch] [remotename]/[branch]
          Git      --track

```

```
$ git checkout --track origin/serverfix
Branch serverfix set up to track remote branch serverfix from origin.
Switched to a new branch 'serverfix'
```

```
$ git checkout -b sf origin/serverfix
Branch sf set up to track remote branch serverfix from origin.
Switched to a new branch 'sf'
```

```

sf          origin/serverfix
           -u      --set-upstream-to
git branch
```

```
$ git branch -u origin/serverfix
Branch serverfix set up to track remote branch serverfix from origin.
```

---

### 上游快捷方式

当设置好跟踪分支后，可以通过 @{upstream} 或 @{u} 快捷方式来引用它。所以在 master 分支时并且它正在跟踪 origin/master 时，如果愿意的话可以使用 git merge @{u} 来取代 git merge origin/master。

---

```
git branch -vv
```

```
$ git branch -vv
  iss53      7e424c3 [origin/iss53: ahead 2] forgot the brackets
  master     1ae2a45 [origin/master] deploying index fix
* serverfix f8674d9 [teamone/server-fix-good: ahead 3, behind 1] this should do it
  testing    5ea463a trying something new
```

2	iss53	origin/iss53	" ahead"
	origin/master	master	
1	teamone	server-fix-good	serverfix
	testing		2

```
$ git fetch --all; git branch
-vv
```

```
git fetch
git pull          git fetch          git
merge           clone   checkout       git pull
```

```
git pull          fetch
merge
```

```
          master
--delete      git push
serverfix
```

```
$ git push origin --delete serverfix
To https://github.com/schacon/simplegit
 - [deleted]      serverfix
```

## 变基

Git                    merge                    rebase

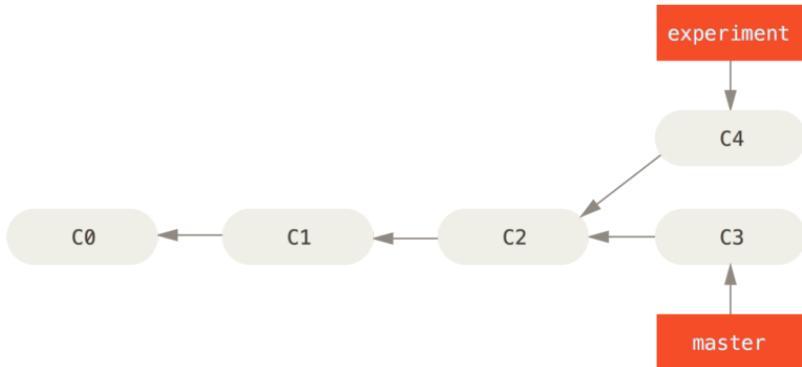
"                    "                    "

"                    "                    "

"                    "                    "

**FIGURE 3-27**

分叉的提交历史



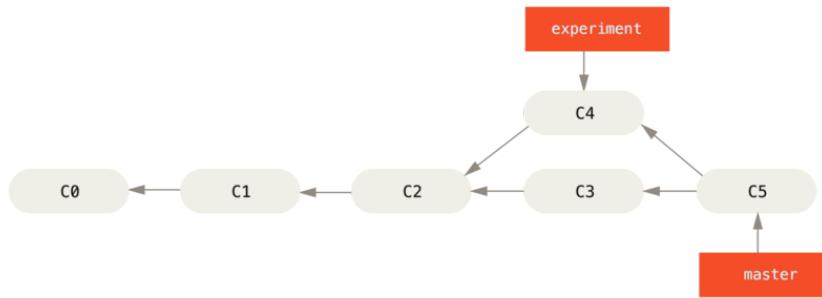
merge

C3      C4

C2

**FIGURE 3-28**

通过合并操作来整合  
分叉了的历史



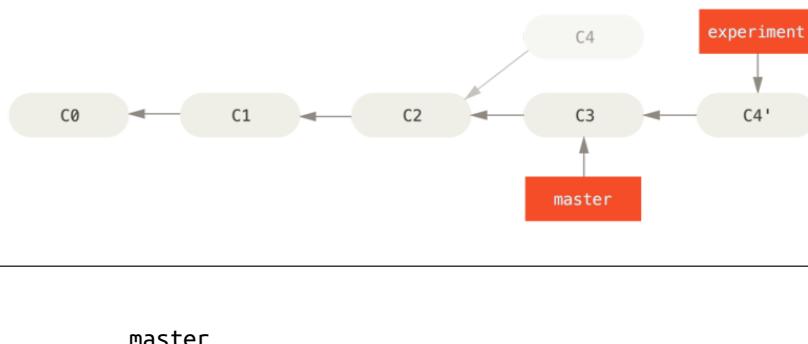
C4  
Git  
base  
"  
C3  
re-  
"

```
$ git checkout experiment
$ git rebase master
First, rewinding head to replay your work on top of it...
Applying: added staged command
```

experiment  
master  
C2  
C3,  
commit id

**FIGURE 3-29**

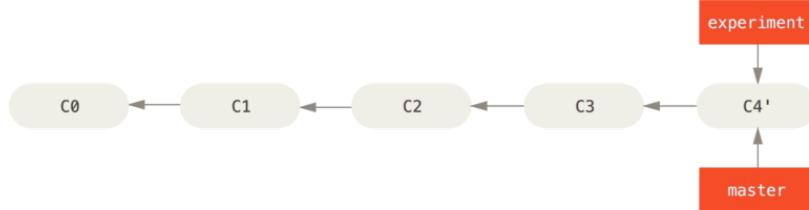
将 C4 中的修改变基到  
C3 上



```
$ git checkout master  
$ git merge experiment
```

FIGURE 3-30

master 分支的快进合并



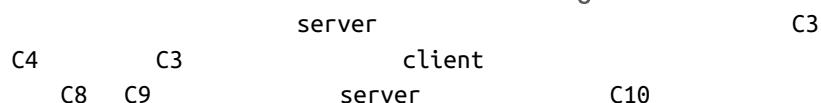
C4'

merge

C5

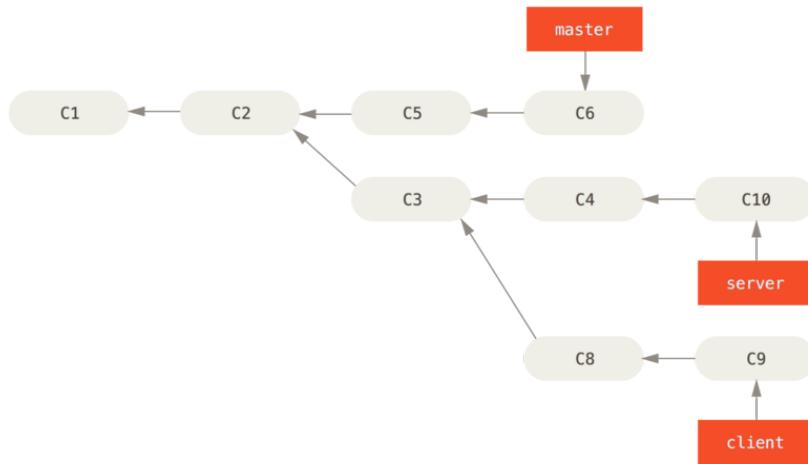
—  
gin/master

Figure 3-31



**FIGURE 3-31**

从一个特性分支里再  
分出一个特性分支的  
提交历史



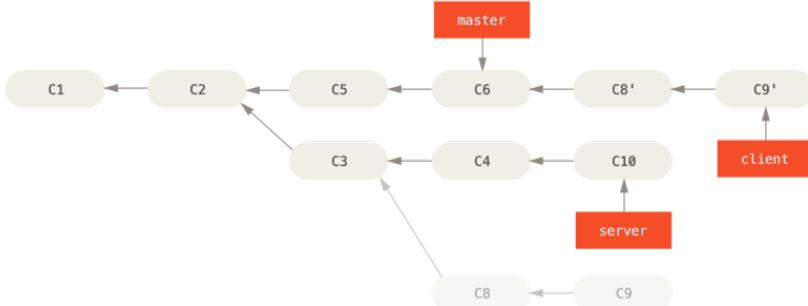
client  
server  
git rebase --onto  
C8 C9 master  
client  
server

```
$ git rebase --onto master server client
```

" client client  
server master

**FIGURE 3-32**

截取特性分支上的另  
一个特性分支，然后  
变基到其他分支



master

Figure 3-33



FIGURE 3-33

快进合并 master 分  
支，使之包含来自  
client 分支的修改

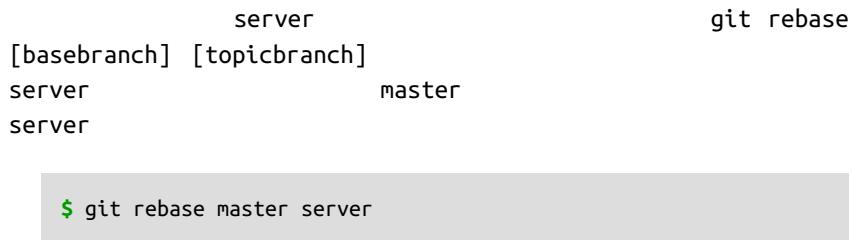
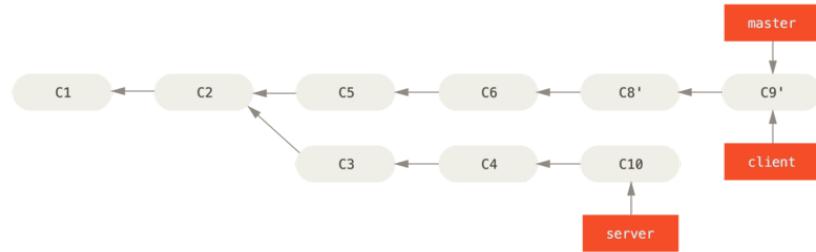


Figure 3-34

server

" " "

master

FIGURE 3-34

将 server 中的修改变  
基到 master 上



master

```
$ git checkout master
$ git merge server
```

client server

Figure 3-35

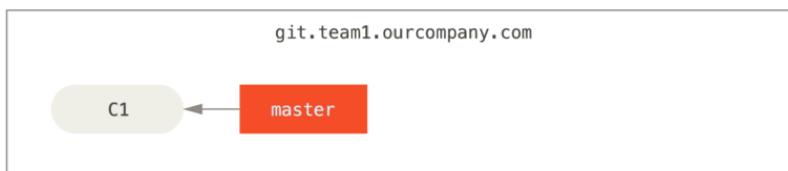
```
$ git branch -d client  
$ git branch -d server
```



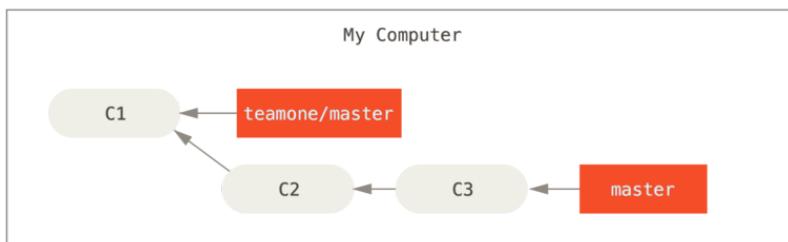
**FIGURE 3-35**

最终的提交历史

git rebase

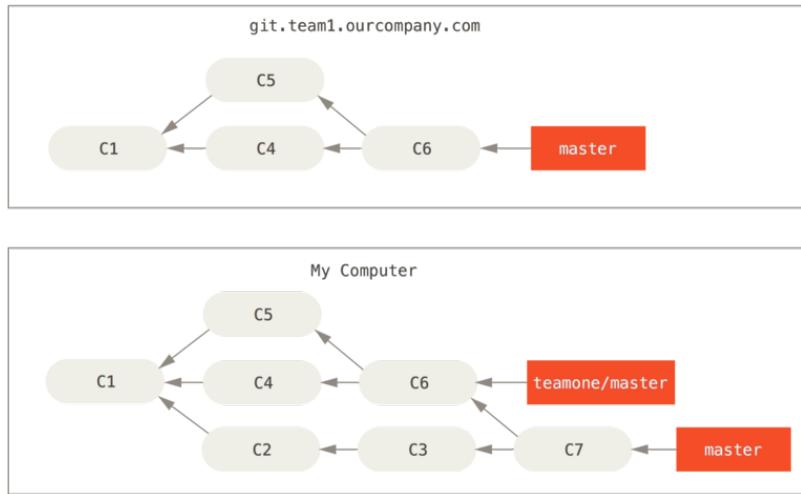


**FIGURE 3-36**



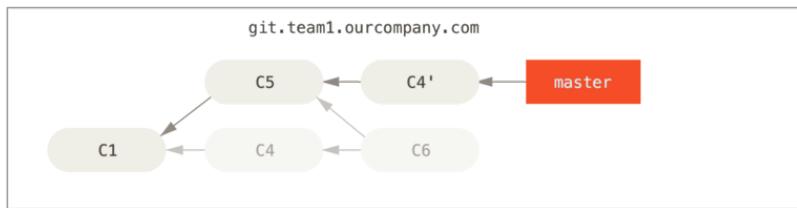
**FIGURE 3-37**

抓取别人的提交，合并到自己的开发分支

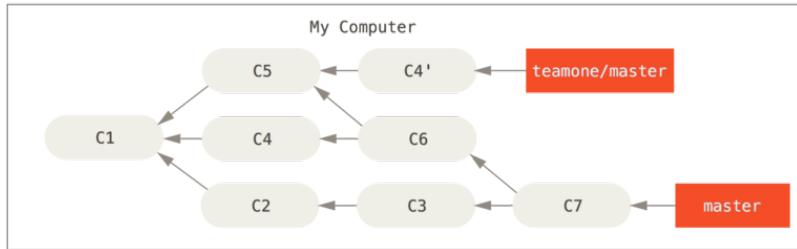


`git push  
--force`

**FIGURE 3-38**

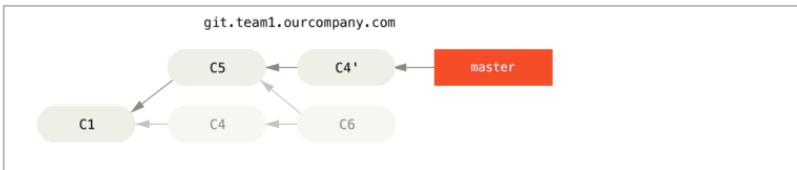


有人推送了经过变基的提交，并丢弃了你的本地开发所基于的一些提交

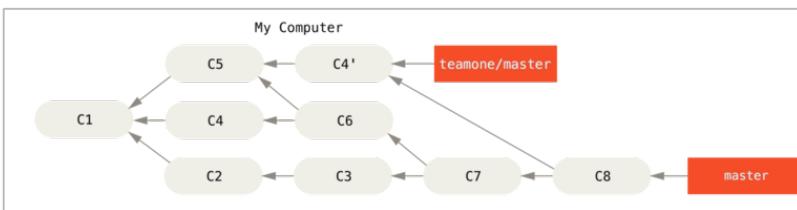


git pull

**FIGURE 3-39**



你将相同的内容又合并了一次，生成了一个新的提交



git log

C4 C6

Git

Git                                    SHA-1  
       —        “patch-id”

Git

Figure 3-38

git rebase teamone/master, Git

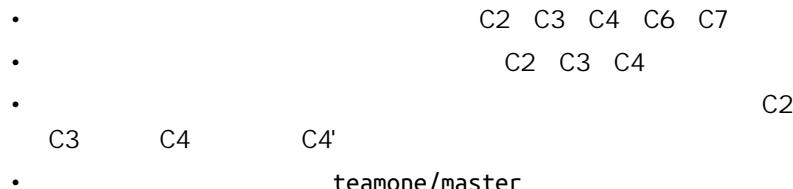
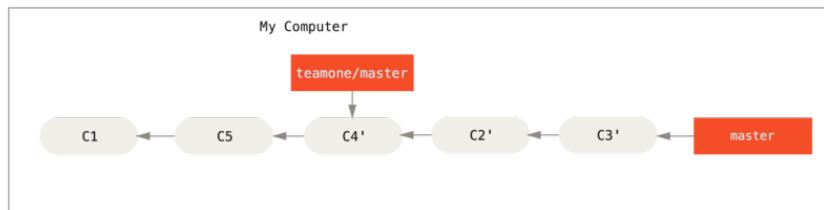
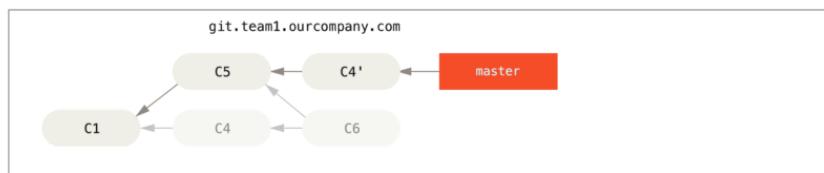


Figure 3-39

Figure 3-40

**FIGURE 3-40**

在一个被变基然后强制推送的分支上再次执行变基



C4'  
C4  
C4

```
git pull --rebase  
git pull  
git config --global pull.rebase true  
pull.rebase
```

```
git pull --rebase
```

vs.

```
rebase filter-branch
```

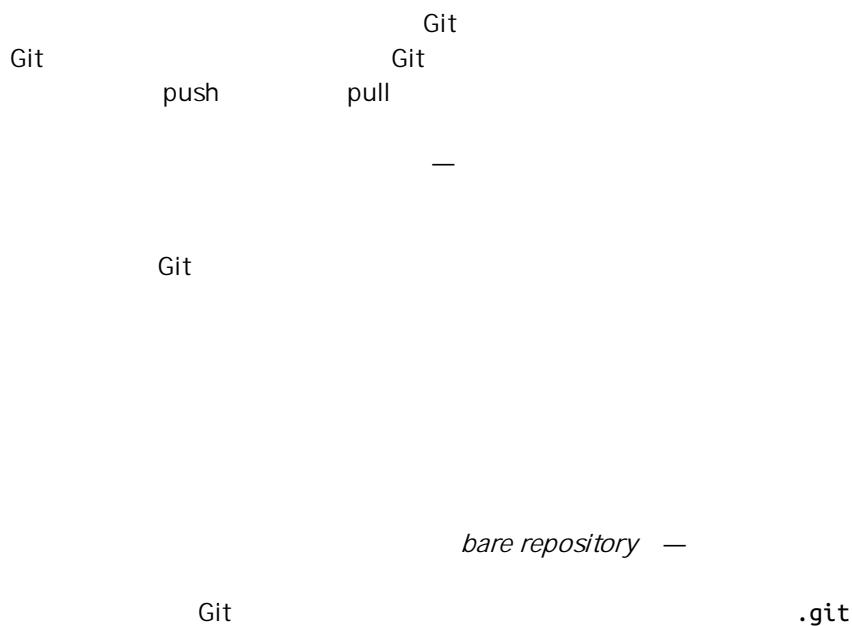
Git

## 总结

Git

## Git

# 4 服务器上的 Git



## 协议

Git    Local      HTTP  
SSH    Secure Shell    Git

*Local protocol*

NFS



```
$ git clone /opt/git/project.git
```

```
$ git clone file:///opt/git/project.git
```

URL                    file://                Git  
                        Git                      hard link  
file://                Git  
                        file://  
extraneous references                         object  
                        -  
                        Chapter 10 for  
maintenance tasks

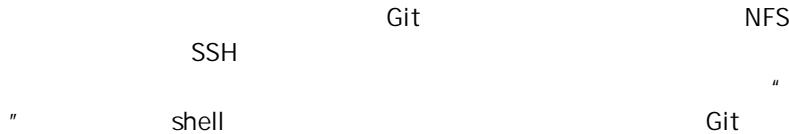
Git

```
$ git remote add local_proj /opt/git/project.git
```

优点

/                        "                        git pull /  
Git"                    home/john/project

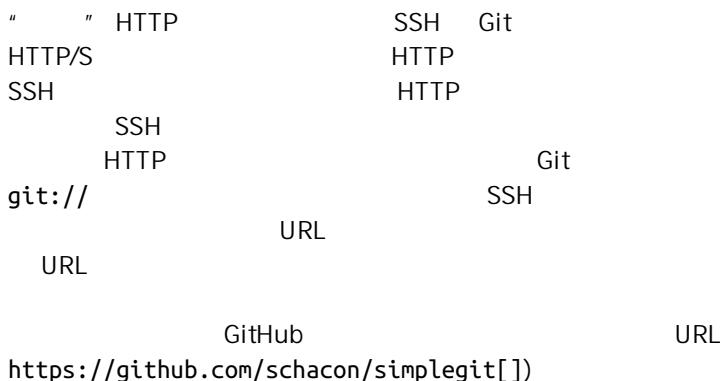
## 缺点



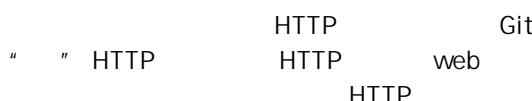
## HTTP



## 智能 ( SMART ) HTTP 协议



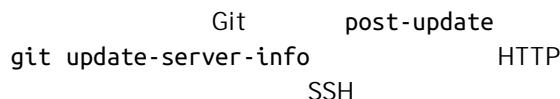
## 哑 ( DUMB ) HTTP 协议





```

$ cd /var/www/htdocs/
$ git clone --bare /path/to/git_project gitproject.git
$ cd gitproject.git
$ mv hooks/post-update.sample hooks/post-update
$ chmod a+x hooks/post-update
  
```

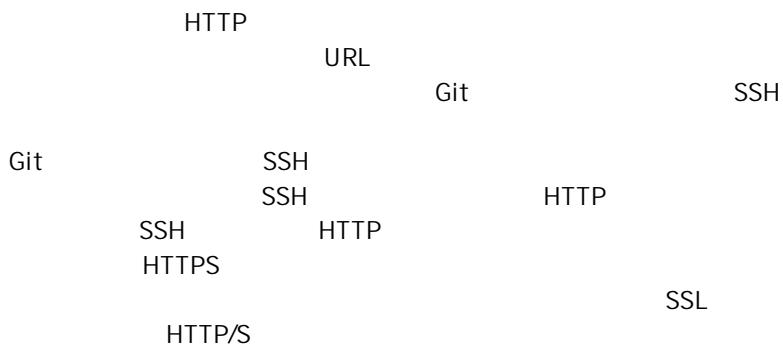


```

$ git clone https://example.com/gitproject.git
  
```



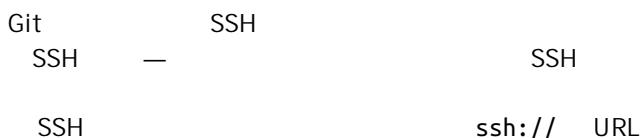
## 优点



## 缺点



## SSH



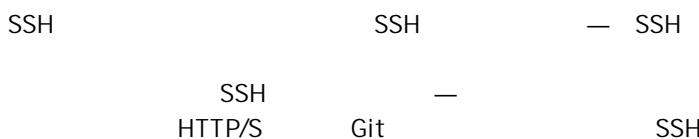
```
$ git clone ssh://user@server/project.git
```

scp

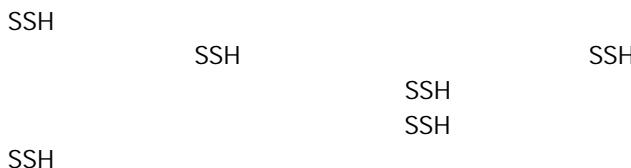
```
$ git clone user@server:project.git
```

Git

## 优势



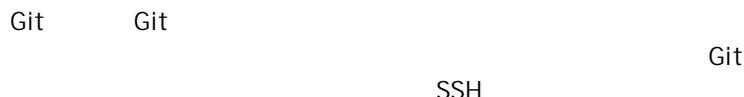
## 缺点



## Git



### 优点



### 缺点



## 在服务器上搭建 Git

Git

---

这里我们将要演示在 Linux 服务器上进行一次基本且简化的安装所需的命令与步骤，当然在 Mac 或 Windows 服务器上同样可以运行这些服务。事实上，在你的计算机基础架构中建立一个生产环境服务器，将不可避免的使用到不同的安全措施与操作系统工具。但是，希望你能从本节中获得一些必要的知识。

---

```

Git --- —
          --- bare
.git
  
```

```
$ git clone --bare my_project my_project.git  
Cloning into bare repository 'my_project.git'...  
done.
```

my\_project.git                    Git

```
$ cp -Rf my_project/.git my_project.git
```

Git

git.example.com  
SSH                    Git                    /opt/git  
                      /opt/git/

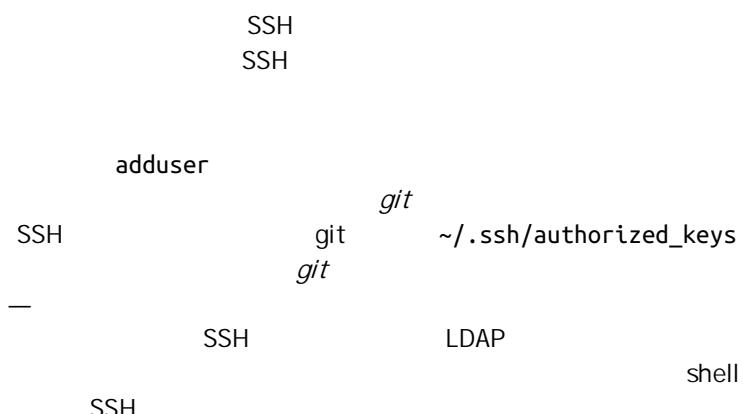
```
$ scp -r my_project.git user@git.example.com:/opt/git
```

SSH    /opt/git

```
$ git clone user@git.example.com:/opt/git/my_project.git
```

my\_project.git                            SSH                            /opt/git/  
                      git init    --shared  
Git

```
$ ssh user@git.example.com  
$ cd /opt/git/my_project.git  
$ git init --bare --shared
```



## 生成 SSH 公钥



SSH

~/.ssh

```
$ cd ~/.ssh
$ ls
authorized_keys2  id_dsa      known_hosts
config           id_dsa.pub
```

id\_dsa id\_rsa

.pub

.pub

.ssh

ssh-keygen

Linux/Mac

ssh-keygen

SSH

Windows

MSysGit

```
$ ssh-keygen
```

Generating public/private rsa key pair.

Enter file in which to save the key (/home/schacon/.ssh/id\_rsa):

Created directory '/home/schacon/.ssh'.

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /home/schacon/.ssh/id\_rsa.

Your public key has been saved in /home/schacon/.ssh/id\_rsa.pub.

The key fingerprint is:

```
d0:82:24:8e:d7:f1:bb:9b:33:53:96:93:49:da:9b:e3 schacon@mylaptop.local
```

ssh-keygen

.ssh/id\_rsa

Git

SSH

.pub

```
$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAkloUpkDHrfHY17SbrmTiPnLTGK9Tjom/BWDSU
GPl+nafzLHDTYW7hdI4yZ5ew18JH4JW9jbhUFrviQzM7xLELEVf4h9lFX5QVkbPppSwg0cda3
Pbv7k0dJ/MTyBlWXFCR+HAo3FXRitBqxiX1nKhXpHAZsMcilQ8V6RjsNAQwdsdMFvSlVK/7XA
t3FaoJoAsncM1Q9x5+3V0Ww68/eIFmb1zuUFljQJKprrX88XypNDvjYNby6vw/Pb0rwert/En
mZ+AW40ZPnPjTPI89ZPmVMLuayrD2cE86Z/il8b+gw3r3+1nKatmIkjn2so1d01QraTlMqVSsbx
NrRFi9wrf+M7Q== schacon@mylaptop.local
```

SSH    GitHub  
<https://help.github.com/articles/generating-ssh-keys>

## 配置服务器

SSH    author -  
ized\_keys  
Linux    git  
.ssh  
Ubuntu

```
$ sudo adduser git
$ su git
$ cd
$ mkdir .ssh && chmod 700 .ssh
$ touch .ssh/authorized_keys && chmod 600 .ssh/authorized_keys
```

git    authorized\_keys  
SSH

```
$ cat /tmp/id_rsa.john.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCB007n/ww+ouN4gSLKssMxXnB0vf9Lgt4L
ojG6rs6hPB09j9R/T17/x4lhJA0F3FR1rP6kYBRsWj2aThGw6HXLm9/5zytK6Ztg3RPKK+4k
Yjh6541NYsnEAZuXz0jTTyAUfrtu3Z5E003C4ox0j6H0rfIF1kKI9MAQLMdPwGW1GYEigS9Ez
Sdfd8AccIicTDWbqlAcU4UpkaX8KyGlLwsNuuGztobF8m72ALC/nLF6JLtPofwFBlgc+myiv
07TCUSbdLQlgMVOfq1I2uPWQ0kOWQAHuke0mfjy2jctxSDBQ220ymjaNsHT4kgtzg2AYygPq
dAv8JggJICUvax2T9va5 gsg-keypair
```

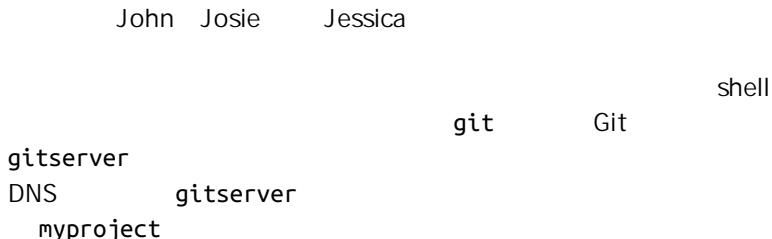
git    .ssh    authorized\_keys

```
$ cat /tmp/id_rsa.john.pub >> ~/.ssh/authorized_keys
$ cat /tmp/id_rsa.josie.pub >> ~/.ssh/authorized_keys
$ cat /tmp/id_rsa.jessica.pub >> ~/.ssh/authorized_keys
```

- -bare    git  
init

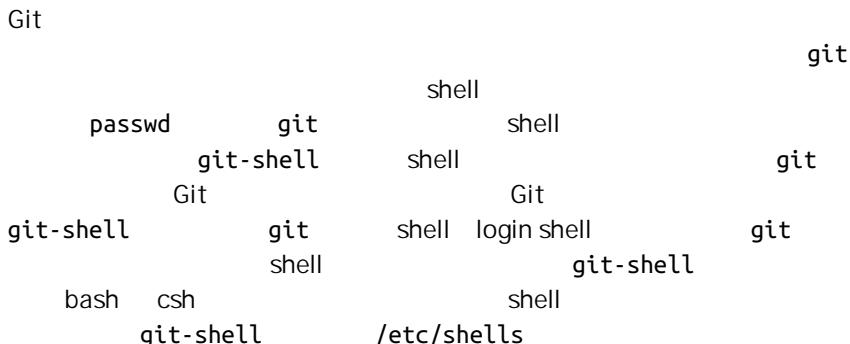
```
$ cd /opt/git
$ mkdir project.git
```

```
$ cd project.git  
$ git init --bare  
Initialized empty Git repository in /opt/git/project.git/
```



```
# on John's computer  
$ cd myproject  
$ git init  
$ git add .  
$ git commit -m 'initial commit'  
$ git remote add origin git@gitserver:/opt/git/project.git  
$ git push origin master
```

```
$ git clone git@gitserver:/opt/git/project.git  
$ cd project  
$ vim README  
$ git commit -am 'fix for the README file'  
$ git push origin master
```



```
$ cat /etc/shells  # see if `git-shell` is already in there. If not...
$ which git-shell  # make sure git-shell is installed on your system.
$ sudo vim /etc/shells  # and add the path to git-shell from last command
```

chsh <username>

shell

```
$ sudo chsh git  # and enter the path to git-shell, usually: /usr/bin/git-shell
```

git	SSH	Git
	shell	

```
$ ssh git@gitserver
fatal: Interactive git shell is not enabled.
hint: ~/git-shell-commands should exist and have read and execute access.
Connection to gitserver closed.
```

Git		
shell		
	git-shell	
		git
		SSH
	shell	
		git help
shell		

## Git 守护进程

" Git"	
Git	
	SSH
	Git

```
git daemon --reuseaddr --base-path=/opt/git/ /opt/git/
```

```
--reuseaddr          --base-
path
Git
9418
```

Ubuntu

Upstart

```
/etc/event.d/local-git-daemon
```

```
start on startup
stop on shutdown
exec /usr/bin/git daemon \
    --user=git --group=git \
    --reuseaddr \
    --base-path=/opt/git/ \
    /opt/git/
respawn
```

```
-           git-ro
git-shell           git
```

Git

```
initctl start local-git-daemon
```

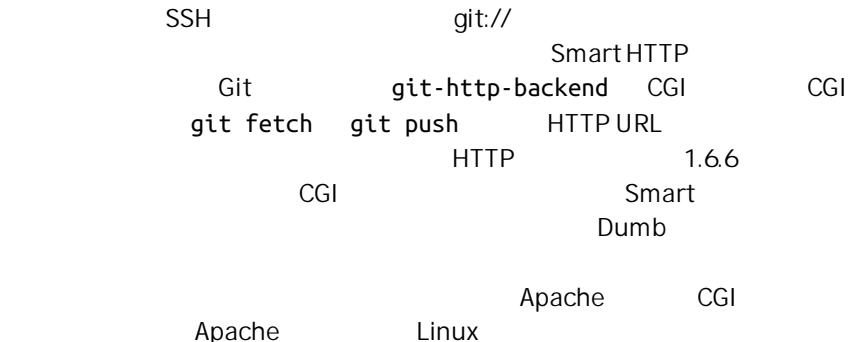
```
sysvinit      xinetd
```

```
-           Git
git-daemon-export-ok
```

```
$ cd /path/to/project.git
$ touch git-daemon-export-ok
```

Git

## Smart HTTP



```
$ sudo apt-get install apache2 apache2-utils  
$ a2enmod cgi alias env
```

```
mod_cgi mod_alias mod_env Apache  
mod_cgi mod_alias mod_env Apache  
Apache  
backend Web /git  
git-https-
```

```
SetEnv GIT_PROJECT_ROOT /opt/git  
SetEnv GIT_HTTP_EXPORT_ALL  
ScriptAlias /git/ /usr/lib/git-core/git-http-backend/
```

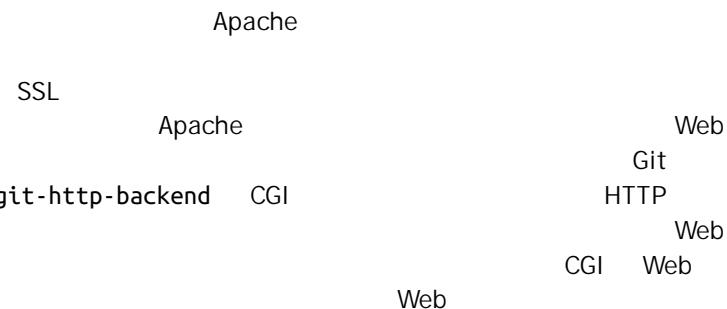
```
GIT_HTTP_EXPORT_ALL Git  
git-daemon-export-ok Git  
Apache  
Apache
```

```
<Directory "/usr/lib/git-core*">  
    Options ExecCGI Indexes  
    Order allow,deny  
    Allow from all  
    Require all granted  
</Directory>
```

```
<LocationMatch "^/git/.*/git-receive-pack$">
    AuthType Basic
    AuthName "Git Access"
    AuthUserFile /opt/git/.htpasswd
    Require valid-user
</LocationMatch>
```

.htaccess  
" schacon"

```
$ htdigest -c /opt/git/.htpasswd "Git Access" schacon
```



---

欲了解更多的有关配置 Apache 授权访问的信息，请通过以下链接浏览 Apache 文档：<http://httpd.apache.org/docs/current/howto/auth.html>

---

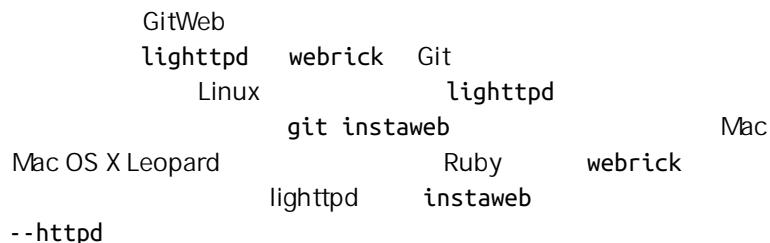
## GitWeb

Git      GitWeb      CGI

**FIGURE 4-1**

GitWeb 的网页用户界面

The screenshot shows a web-based interface for a Git repository. At the top, there's a header with links for 'projects / .git / summary', 'summary', 'shortlog', 'log', 'commit', 'commitdiff', 'tree', and a search bar. Below the header, there's a summary section with details about the repository: 'description: Unnamed repository; edit this file 'description' to name the repository.', 'owner: Ben Straub', and 'last change: Wed, 11 Jun 2014 12:20:23 -0700 (21:20 +0200)'. The main content area is titled 'shortlog' and lists several commit entries from June 2014. Each entry includes the author, date, commit message, and a small preview of the changes. Below the shortlog is a 'tags' section listing various versions of the software, each with a link to its corresponding tag commit.



```
$ git instaweb --httpd=webrick
[2009-02-21 10:02:21] INFO  WEBrick 1.3.1
[2009-02-21 10:02:21] INFO  ruby 1.8.6 (2008-03-03) [universal-darwin9.0]
```

1234                   HTTP

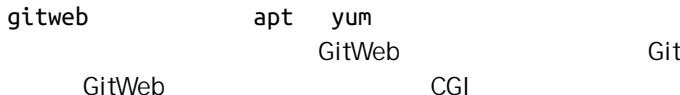
--stop

```
$ git instaweb --httpd=webrick --stop
```

Web

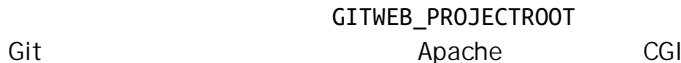
CGI

Linux



```

$ git clone git://git.kernel.org/pub/scm/git/git.git
$ cd git
$ make GITWEB_PROJECTROOT="/opt/git" prefix=/usr gitweb
      SUBDIR gitweb
      SUBDIR ../
make[2]: `GIT-VERSION-FILE' is up to date.
      GEN gitweb.cgi
      GEN static/gitweb.js
$ sudo cp -Rf gitweb /var/www/
  
```

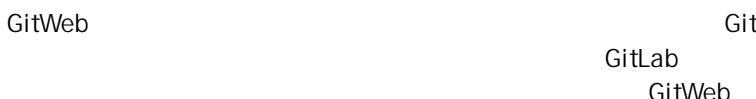


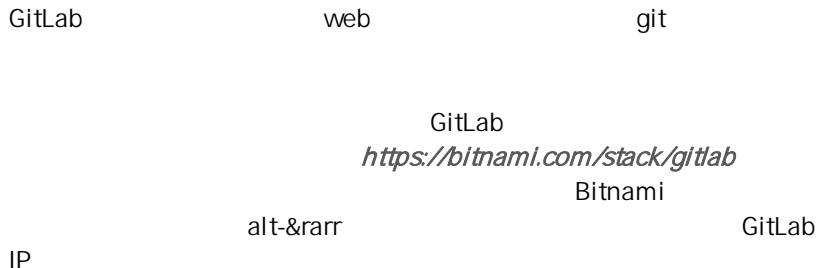
```

<VirtualHost *:80>
    ServerName gitserver
    DocumentRoot /var/www/gitweb
    <Directory /var/www/gitweb>
        Options ExecCGI +FollowSymLinks +SymLinksIfOwnerMatch
        AllowOverride All
        order allow,deny
        Allow from all
        AddHandler cgi-script cgi
        DirectoryIndex gitweb.cgi
    </Directory>
</VirtualHost>
  
```

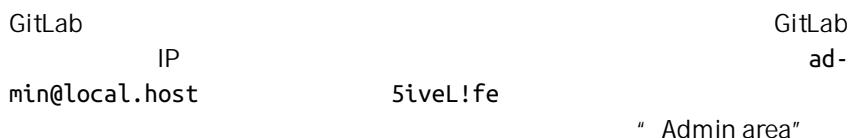
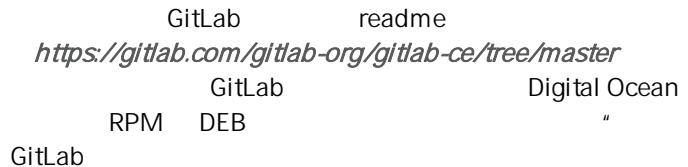
GitWeb      CGI      Perl  
<http://gitserver/>

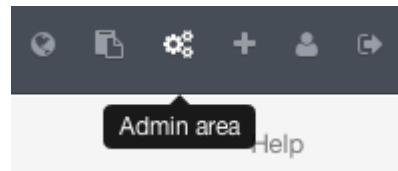
## GitLab



**FIGURE 4-2**

*Bitnami GitLab 虚拟机登录界面。*



**FIGURE 4-3**

GitLab 主栏的“Admin area”图标。

## 使用者

GitLab

project                          jane  
                                    url         *http://server/jane/project*

Users (3)	
Administrator (Admin) It's you!	<a href="#">Edit</a>
Rachel Myers (Admin)	<a href="#">Edit</a> <a href="#">Block</a> <a href="#">Destroy</a>
Russell Belfer	<a href="#">Edit</a> <a href="#">Block</a> <a href="#">Destroy</a>

**FIGURE 4-4**

GitLab 用户管理界面。

GitLab                                  " Blocking "

" Destroying "

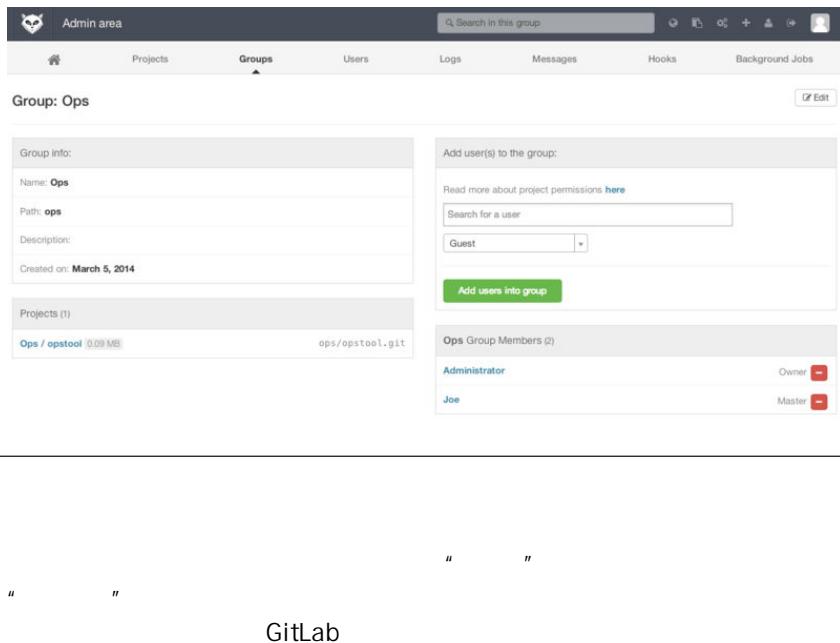
## 组

GitLab

training materials url  
*http://server/training/materials*

**FIGURE 4-5**

GitLab 组管理界面。



项目

GitLab

git

钩子

GitLab

GitLab

git

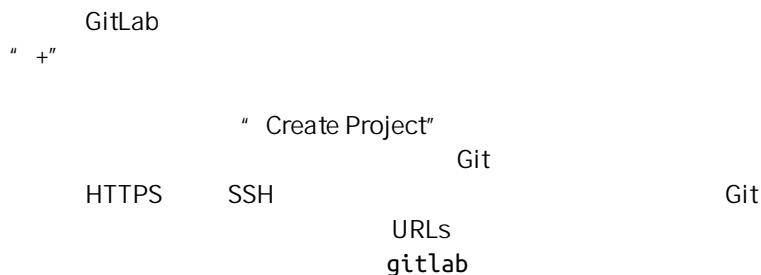
web

git " fetch"

JSON

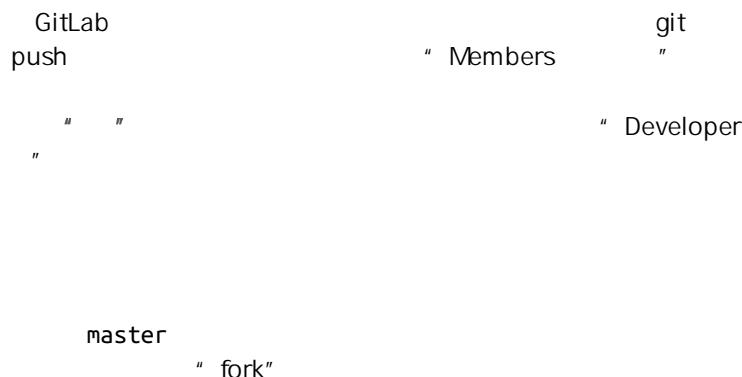
HTTP

CI

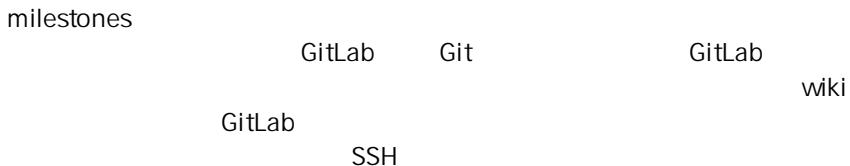


```
$ git remote add gitlab https://server/namespace/project.git
```

web



GitLab



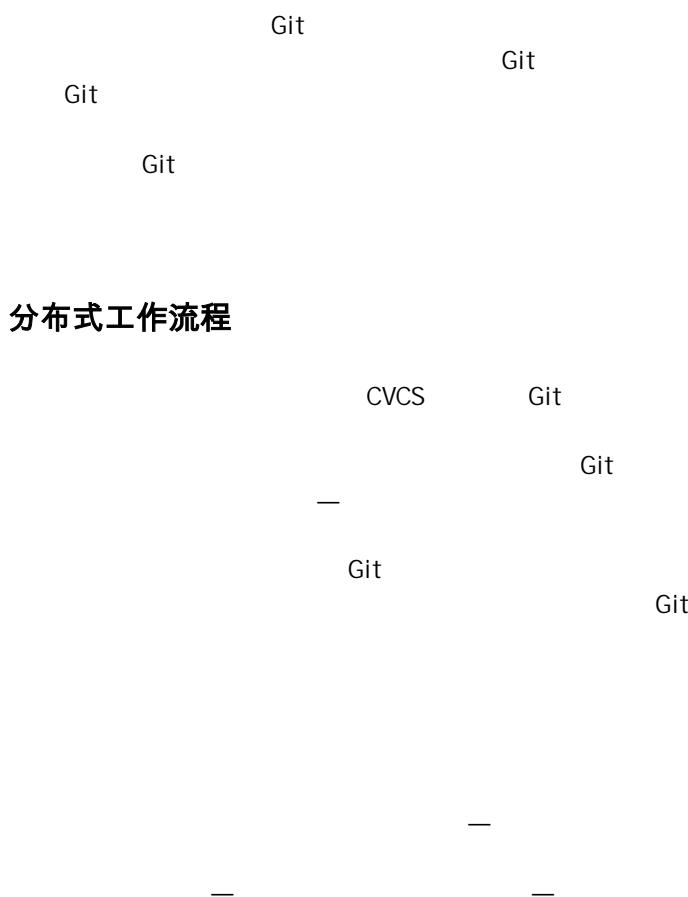
## 第三方托管的选择



## 总结

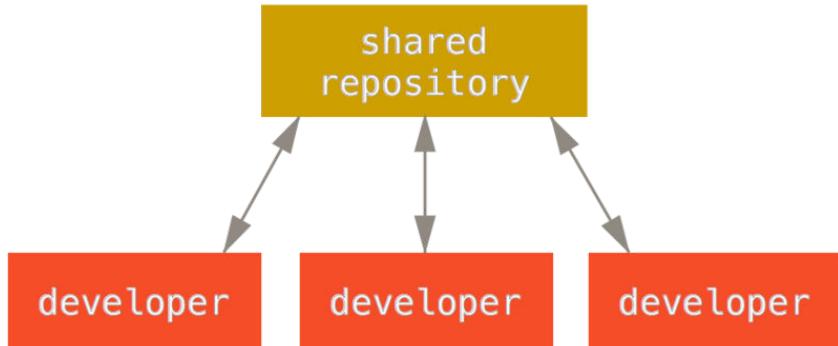
Git

# 分布式 Git 5



**FIGURE 5-1**

集中式工作流。



Subversion  
Git

CVCS

John      Jessica  
          Jessica  
                          Git  
                          non-fast-forward

Git

Git

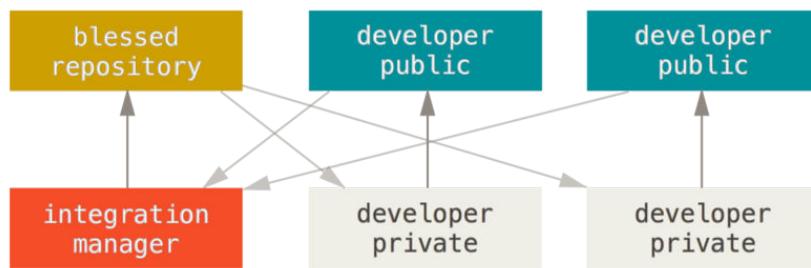
"      "

Figure 5-2

1.

2.

3.  
4.  
5.  
6.



**FIGURE 5-2**  
集成管理者工作流。

GitHub      GitLab      hub-based

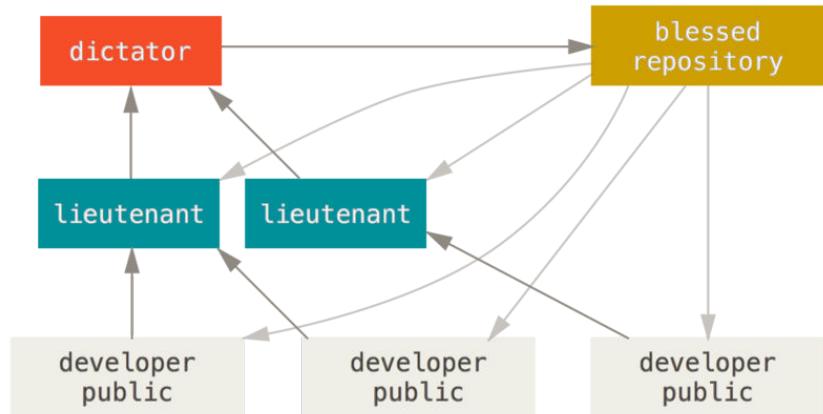
—  
lieutenant      Linux  
dictator

( Figure 5-3):

1.      master
2.      master
3.      master      master
4.      master

FIGURE 5-3

司令官与副官工作流。



Git

## 向一个项目贡献

Git

Git Documentation/SubmittingPatches

git help dif

git diff --check

```
bash
lib/simplegit.rb:5: trailing whitespace.
+   @git_dir = File.expand_path(git_dir)
lib/simplegit.rb:7: trailing whitespace.
+ [REDACTED]
lib/simplegit.rb:20: trailing whitespace.
+end
(END)
```

**FIGURE 5-4**

*git diff --check*  
的输出

git  
add --patch

Git 50 25

Git

" Add tests  
for." " I added tests for" " Adding tests for,"  
Tim Pope

修改的摘要 ( 50 个字符或更少 )

如果必要的话，加入更详细的解释文字。在大概 72 个字符的时候换行。在某些情形下，第一行被当作一封电子邮件的标题，剩下的文本作为正文。分隔摘要与正文的空行是必须的（除非你完全省略正文）；如果你将两者混在一起，那么类似变基等工具无法正常工作。

空行接着更进一步的段落。

- 句号也是可以的。
- 项目符号可以使用典型的连字符或星号前面一个空格，之间用空行隔开，但是可以依据不同的惯例有所不同。

Git  
git log --no-merges

-m git commit

"

Subversion

John

...

```
# John's Machine
$ git clone john@githost:simplegit.git
Initialized empty Git repository in /home/john/simplegit/.git/
...
$ cd simplegit/
$ vim lib/simplegit.rb
$ git commit -am 'removed invalid default value'
[master 738ee87] removed invalid default value
 1 files changed, 1 insertions(+), 1 deletions(-)
```

Jessica

-

```
# Jessica's Machine
$ git clone jessica@githost:simplegit.git
Initialized empty Git repository in /home/jessica/simplegit/.git/
...
$ cd simplegit/
$ vim TODO
$ git commit -am 'add reset task'
[master fbff5bc] add reset task
 1 files changed, 1 insertions(+), 0 deletions(-)
```

Jessica

```
# Jessica's Machine
$ git push origin master
...
To jessica@githost:simplegit.git
  1edee6b..fbff5bc  master -> master
```

John

```
# John's Machine
$ git push origin master
To john@githost:simplegit.git
 ! [rejected]          master -> master (non-fast forward)
error: failed to push some refs to 'john@githost:simplegit.git'
```

John  
Subversion

Subversion

Git

Jessica

John

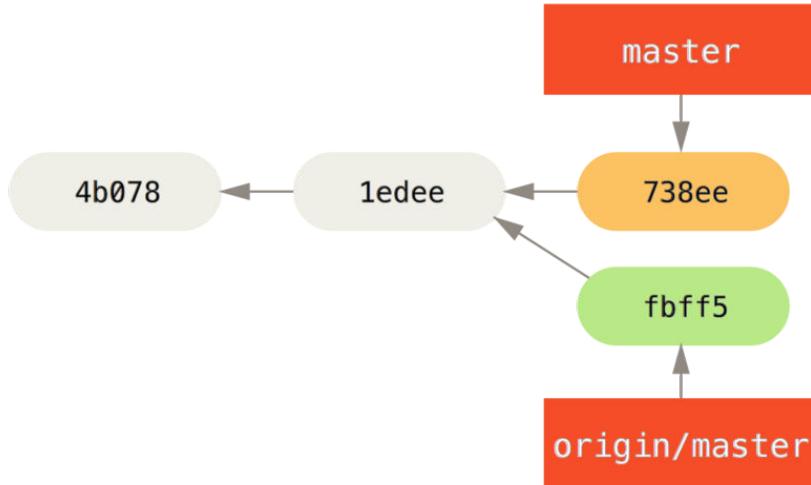
Jessica

```
$ git fetch origin
...
From john@githost:simplegit
 + 049d078...fbff5bc  master      -> origin/master
```

John

**FIGURE 5-5**

John 的分叉历史



John

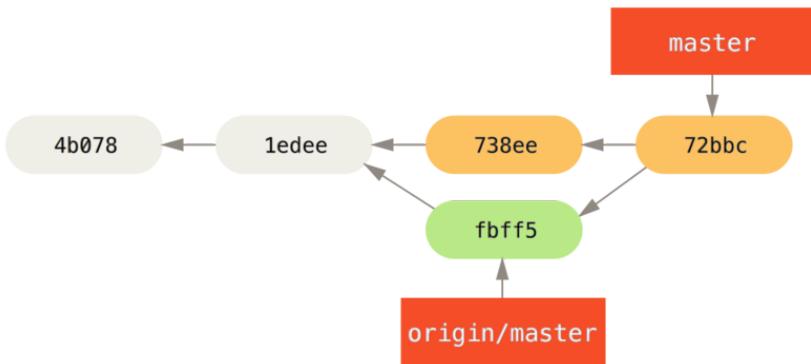
Jessica

```
$ git merge origin/master
Merge made by recursive.
TODO | 1 +
1 files changed, 1 insertions(+), 0 deletions(-)
```

- John

**FIGURE 5-6**

合并了 origin/master 之后 John 的仓库



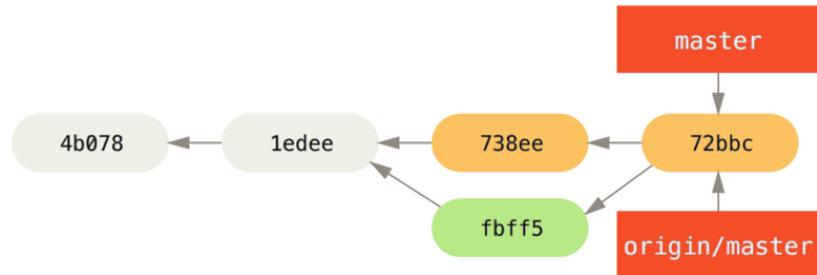
John

```
$ git push origin master
...
To john@githost:simplegit.git
fbff5bc..72bbc59  master -> master
```

John

**FIGURE 5-7**

推送到了 origin 服务  
后 John 的历史



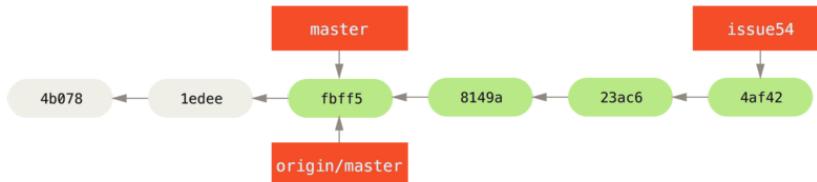
Jessica

issue54

John

**FIGURE 5-8**

Jessica 的特性分支



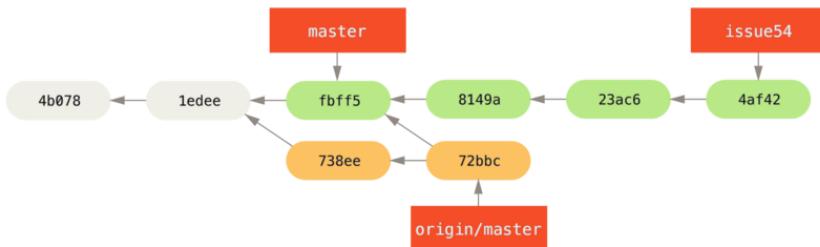
Jessica

John

```
# Jessica's Machine
$ git fetch origin
...
```

```
From jessica@githost:simplegit
fbff5bc..72bbc59  master      -> origin/master
```

John                    Jessica



**FIGURE 5-9**

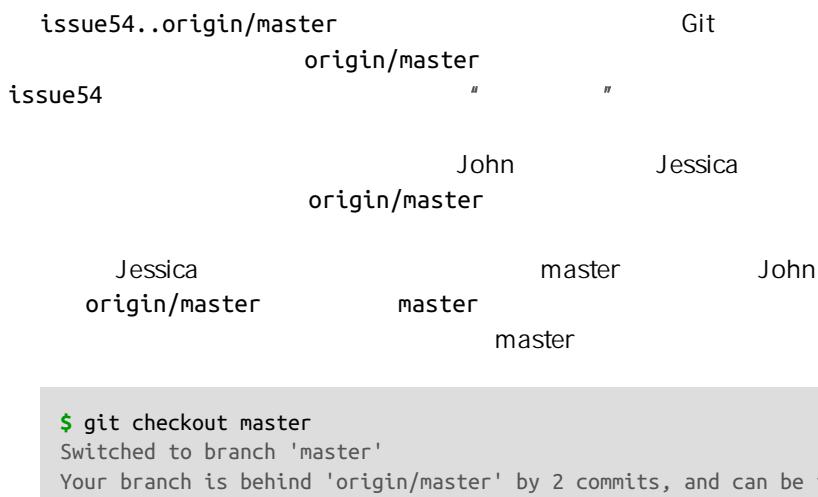
抓取 John 的改动后  
Jessica 的历史

Jessica

git log

```
$ git log --no-merges issue54..origin/master
commit 738ee872852dfa9d6634e0dea7a324040193016
Author: John Smith <jsmith@example.com>
Date:   Fri May 29 16:01:27 2009 -0700

removed invalid default value
```



```
origin/master          issue54 -  
                     issue54  
  

$ git merge issue54
Updating fbff5bc..4af4298
Fast forward
 README           |      1 +
 lib/simplegit.rb |      6 +++++-
 2 files changed, 6 insertions(+), 1 deletions(-)
```

Jessica

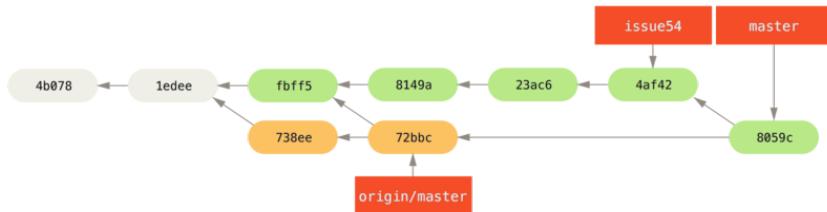
John      origin/master

```
$ git merge origin/master
Auto-merging lib/simplegit.rb
Merge made by recursive.
 lib/simplegit.rb |    2 ++
 1 files changed, 1 insertions(+), 1 deletions(-)
```

Jessica

**FIGURE 5-10**

合并了John的改动后Jessica的历史

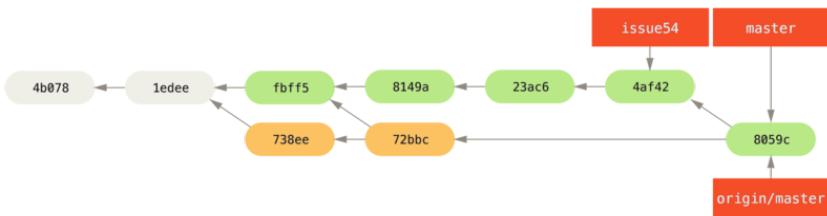


origin/master      Jessica    master  
                      John

```
$ git push origin master
...
To jessica@githost:simplegit.git
 72bbc59..8059c15  master -> master
```

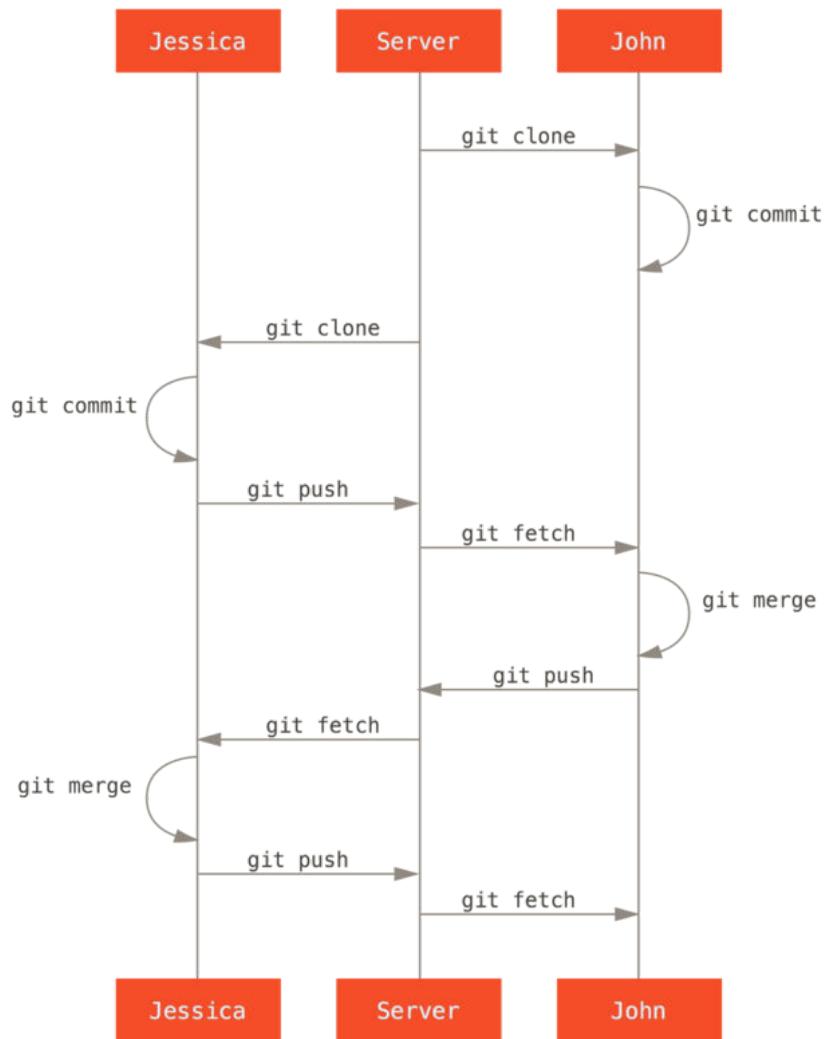
**FIGURE 5-11**

推送所有的改动回服务器后 Jessica 的历史



**FIGURE 5-12**

一个简单的多人 Git  
工作流程的通常事件  
顺序



John      Jessica

Jessica      Josie

master

Jessica

featureA

```
# Jessica's Machine
$ git checkout -b featureA
Switched to a new branch 'featureA'
$ vim lib/simplegit.rb
$ git commit -am 'add limit to log function'
[featureA 3300904] add limit to log function
1 files changed, 1 insertions(+), 1 deletions(-)
```

```
          John      featureA
Jessica    master      -
          John      -
```

```
$ git push -u origin featureA
...
To jessica@githost:simplegit.git
 * [new branch]      featureA -> featureA
```

```
Jessica  John      featureA
          John      Jessica  Josie
featureB           master
```

```
# Jessica's Machine
$ git fetch origin
$ git checkout -b featureB origin/master
Switched to a new branch 'featureB'
```

Jessica featureB

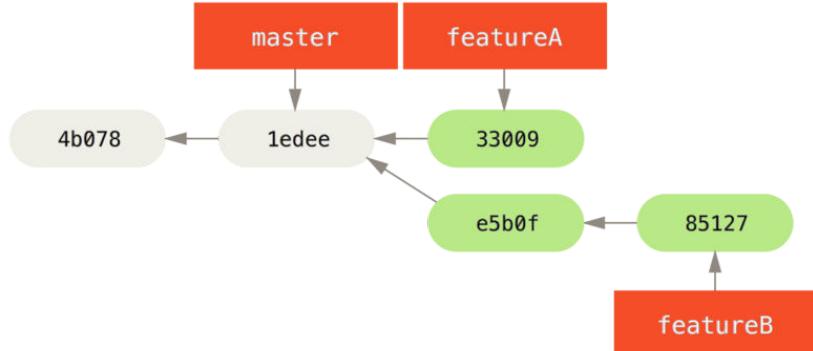
```
$ vim lib/simplegit.rb
$ git commit -am 'made the ls-tree function recursive'
[featureB e5b0fdc] made the ls-tree function recursive
1 files changed, 1 insertions(+), 1 deletions(-)
$ vim lib/simplegit.rb
$ git commit -am 'add ls-files'
```

```
[featureB 8512791] add ls-files
 1 files changed, 5 insertions(+), 0 deletions(-)
```

Jessica

**FIGURE 5-13**

Jessica 的初始提交历史



Josie  
featureBee      Jessica  
git fetch      Josie

```
$ git fetch origin
...
From jessica@githost:simplegit
 * [new branch]      featureBee -> origin/featureBee
```

Jessica      git merge

```
$ git merge origin/featureBee
Auto-merging lib/simplegit.rb
Merge made by recursive.
 lib/simplegit.rb |    4 +++
 1 files changed, 4 insertions(+), 0 deletions(-)
```

-

featureB

featureBee      git push

:

```
$ git push -u origin featureB:featureBee
...
To jessica@githost:simplegit.git
  fba9af8..cd685d1  featureB -> featureBee
```



```
$ git fetch origin
...
From jessica@githost:simplegit
  3300904..aad881d  featureA -> origin/featureA
```

git log

```
$ git log featureA..origin/featureA
commit aad881d154acdaeb2b6b18ea0e827ed8a6d671e6
Author: John Smith <jsmith@example.com>
Date:   Fri May 29 19:57:33 2009 -0700
```

changed log output to 30 from 25

John                    featureA

```
$ git checkout featureA
Switched to branch 'featureA'
$ git merge origin/featureA
Updating 3300904..aad881d
Fast forward
 lib/simplegit.rb | 10 ++++++----
 1 files changed, 9 insertions(+), 1 deletions(-)
```

Jessica

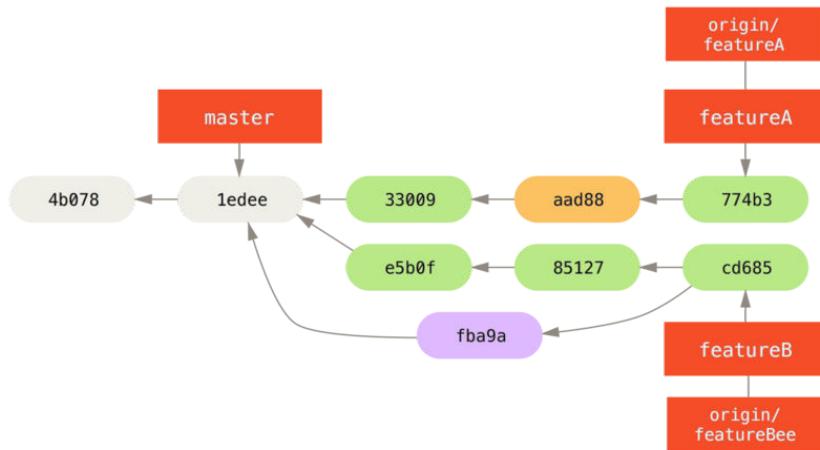
```
$ git commit -am 'small tweak'
[featureA 774b3ed] small tweak
 1 files changed, 1 insertions(+), 1 deletions(-)
```

```
$ git push
...
To jessica@githost:simplegit.git
  3300904..774b3ed  featureA -> featureA
```

Jessica

**FIGURE 5-14**

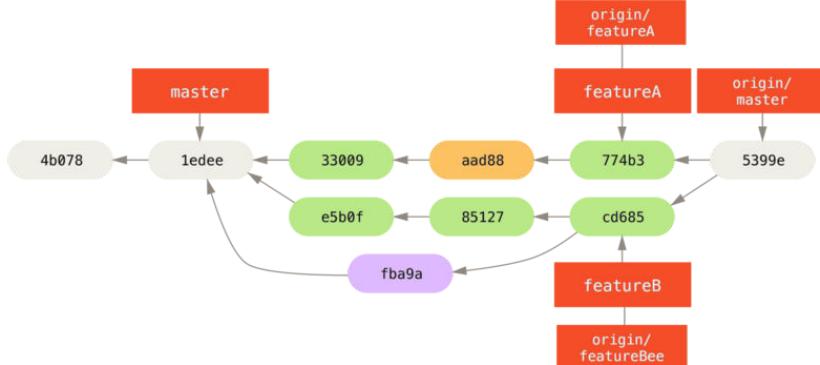
在一个特性分支提交后 Jessica 的历史



Jessica Josie John  
Bee                    featureA    featureB

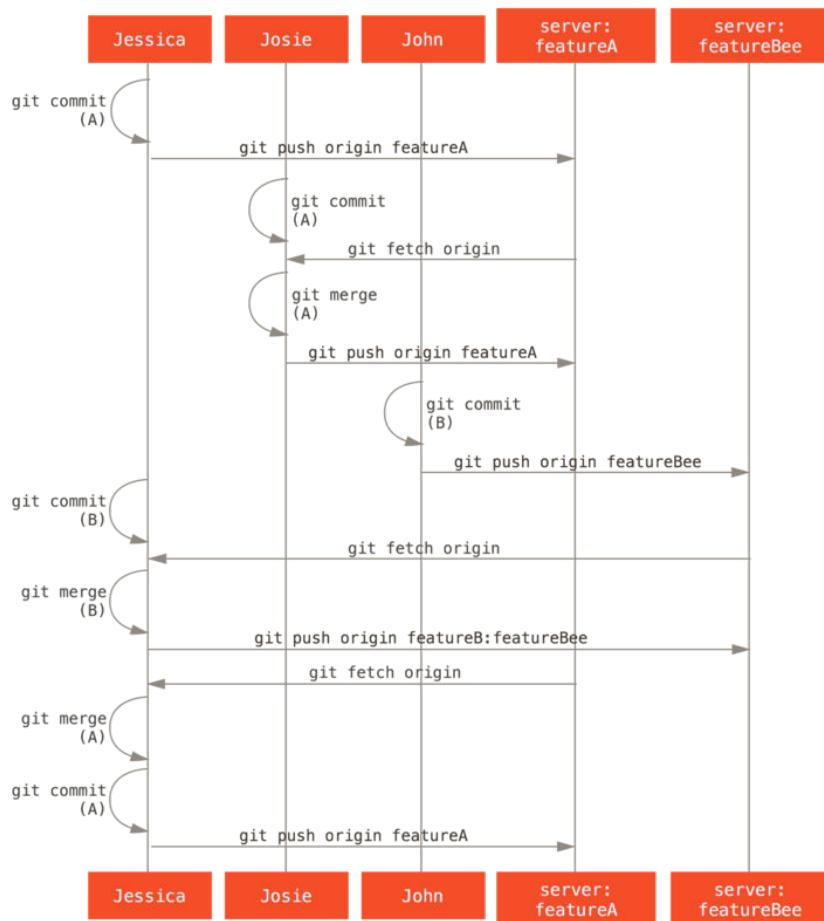
**FIGURE 5-15**

合并了 Jessica 的两个特性分支后她的历史



# Git

# Git



**FIGURE 5-16**

这种管理团队工作流程的基本顺序

Git  
GitHub Bit

Bucket Google Code repo.or.cz

```
$ git clone (url)
$ cd project
$ git checkout -b featureA
# (work)
$ git commit
# (work)
$ git commit
```

---

你可能会想要使用 `rebase -i` 来将工作压缩成一个单独的提交，或者重排提交中的工作使补丁更容易被维护者审核 - 查看“重写历史”了解关于交互式变基的更多信息。

---

" Fork"

URL

myfork

```
$ git remote add myfork (url)
```

master

```
$ git push -u myfork featureA
```

pull request  
Pull Request  
request-pull  
request-pull  
URL  
John

Chapter 6

- GitHub  
- git  
Git  
Jessica

```
$ git request-pull origin/master myfork
The following changes since commit 1edee6b1d61823a2de3b09c160d7080b8d1b3a40:
  John Smith (1):
    added a new function

are available in the git repository at:

  git://githost/simplegit.git featureA

  Jessica Smith (2):
    add limit to log function
    change log output to 30 from 25

  lib/simplegit.rb |   10 ++++++++-+
  1 files changed, 9 insertions(+), 1 deletions(-)
```

origin/master

master

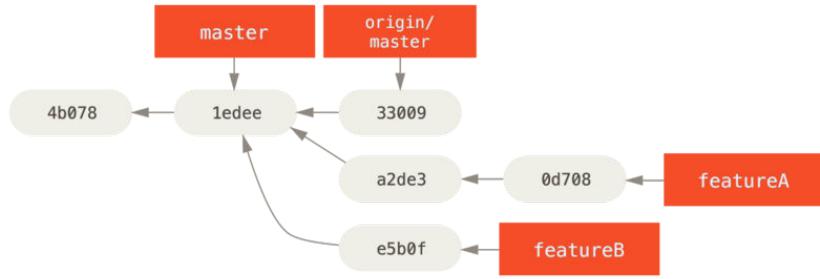
rebase

master

```
$ git checkout -b featureB origin/master
# (work)
$ git commit
$ git push myfork featureB
# (email maintainer)
$ git fetch origin
```

**FIGURE 5-17**

*featureB* 的初始提交历史



ori-  
gin/master

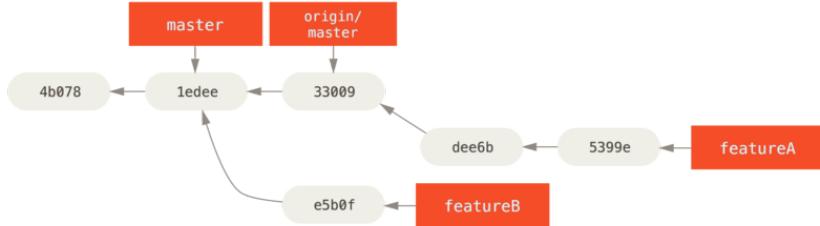
```

$ git checkout featureA
$ git rebase origin/master
$ git push -f myfork featureA
  
```

Figure 5-18

**FIGURE 5-18**

*featureA* 工作之后的提交历史



-f  
featureA  
featur-

eAv2

master origin/master

featureB

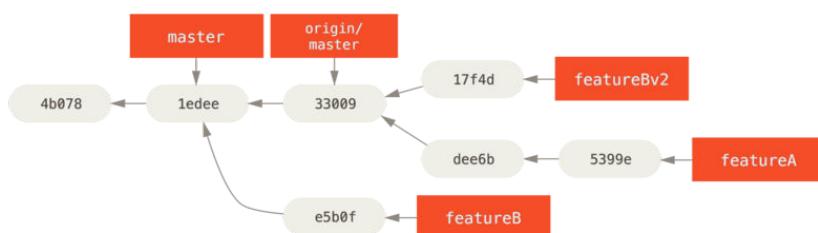
```
$ git checkout -b featureBv2 origin/master
$ git merge --no-commit --squash featureB
# (change implementation)
$ git commit
$ git push myfork featureBv2
```

--squash

--no-

commit

featureBv2



**FIGURE 5-19**

*featureBv2 工作之后的提交历史*

```
$ git checkout -b topicA
# (work)
$ git commit
```

```
# (work)
$ git commit
```

git format-patch  
mbox -

format-patch

```
$ git format-patch -M origin/master
0001-add-limit-to-log-function.patch
0002-changed-log-output-to-30-from-25.patch
```

format-patch                                    -M                                    Git

```
$ cat 0001-add-limit-to-log-function.patch
From 330090432754092d704da8e76ca5c05c198e71a8 Mon Sep 17 00:00:00 2001
From: Jessica Smith <jessica@example.com>
Date: Sun, 6 Apr 2008 10:17:23 -0700
Subject: [PATCH 1/2] add limit to log function

Limit log functionality to the first 20

---
lib/simplegit.rb |    2 ++
1 files changed, 1 insertions(+), 1 deletions(-)

diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index 76f47bc..f9815f1 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -14,7 +14,7 @@ class SimpleGit
end

def log(treeish = 'master')
-   command("git log #{treeish}")
+   command("git log -n 20 #{treeish}")
end

def ls_tree(treeish = 'master')
-- 
2.1.0
```

```
--- diff --git
```

```
Git           IMAP  
Gmail  
Documentation/  
SubmittingPatches  
~/gitconfig      imap
```

```
git config
```

```
[imap]  
folder = "[Gmail]/Drafts"  
host = imaps://imap.gmail.com  
user = user@gmail.com  
pass = p4ssw0rd  
port = 993  
sslverify = false
```

```
IMAP          SSL          host  
imap://       imaps://  
              IMAP        Drafts  
                           git imap-send
```

```
$ cat *.patch |git imap-send  
Resolving imap.gmail.com... ok  
Connecting to [74.125.142.109]:993... ok  
Logging in...  
sending 2 messages  
100% (2/2) done
```

```
Drafts
```

```
SMTM  
git config  
~/gitconfig      sendmail  
  
[sendemail]  
smtpencryption = tls  
smtpserver = smtp.gmail.com
```

```
smtpuser = user@gmail.com
smtpserverport = 587
```

git send-email

```
$ git send-email *.patch
0001-added-limit-to-log-function.patch
0002-changed-log-output-to-30-from-25.patch
Who should the emails appear to be from? [Jessica Smith <jessica@example.com>]
Emails will be sent from: Jessica Smith <jessica@example.com>
Who should the emails be sent to? jessica@example.com
Message-ID to be used as In-Reply-To for the first email? y
```

Git

```
(mbox) Adding cc: Jessica Smith <jessica@example.com> from
      \line 'From: Jessica Smith <jessica@example.com>'
OK. Log says:
Sendmail: /usr/sbin/sendmail -i jessica@example.com
From: Jessica Smith <jessica@example.com>
To: jessica@example.com
Subject: [PATCH 1/2] added limit to log function
Date: Sat, 30 May 2009 13:29:15 -0700
Message-Id: <1243715356-61726-1-git-send-email-jessica@example.com>
X-Mailer: git-send-email 1.6.2.rc1.20.g8c5b.dirty
In-Reply-To: <y>
References: <y>
```

Result: OK

Git

Git

## 维护项目

format-patch

ru-

by\_client

```
sc/ruby_client      sc  
                   master
```

```
$ git branch sc/ruby_client master
```

checkout -b

```
$ git checkout -b sc/ruby_client master
```

git apply

git am

使用APPLY命令应用补丁

```
git diff   Unix diff  
          git apply  
/tmp/patch-ruby-client.patch
```

```
$ git apply /tmp/patch-ruby-client.patch
```

```
patch -p1  
patch  
git diff
```

```

patch          git apply      "
apply all or abort all   "
patch
patch          git
apply          —
—             git apply
git apply --check

```

```
$ git apply --check 0001-seeing-if-this-helps-the-gem.patch
error: patch failed: ticgit.gemspec:1
error: ticgit.gemspec: patch does not apply
```

### 使用 AM 命令应用补丁

Git	format-patch
format-patch	diff
git apply	
format-patch	git am
git am	mbox
	git am
	mbox

```
From 330090432754092d704da8e76ca5c05c198e71a8 Mon Sep 17 00:00:00 2001
From: Jessica Smith <jessica@example.com>
Date: Sun, 6 Apr 2008 10:17:23 -0700
Subject: [PATCH 1/2] add limit to log function

Limit log functionality to the first 20
```

format-patch	git send-email
mbox	mbox
git am	
	mbox

	mbox	git am
Ticket	format-patch	Request
		git am

```
$ git am 0001-limit-log-function.patch
Applying: add limit to log function
```

From	Date
Subject	mbox

```
$ git log --pretty=fuller -1
commit 6c5e70b984a60b3cecd395edd5b48a7575bf58e0
Author: Jessica Smith <jessica@example.com>
AuthorDate: Sun Apr 6 10:17:23 2008 -0700
Commit: Scott Chacon <schacon@gmail.com>
CommitDate: Thu Apr 9 09:19:06 2009 -0700
```

add limit to log function

Limit log functionality to the first 20

Commit	Author
--------	--------

git am

```
$ git am 0001-seeing-if-this-helps-the-gem.patch
Applying: seeing if this helps the gem
error: patch failed: ticgit.gemspec:1
error: ticgit.gemspec: patch does not apply
Patch failed at 0001.
When you have resolved this problem run "git am --resolved".
If you would prefer to skip this patch, instead run "git am --skip".
To restore the original branch and stop patching run "git am --abort".
```

—

```
git am --resolved
```

```
$ (fix the file)
$ git add ticgit.gemspec
$ git am --resolved
Applying: seeing if this helps the gem
```

Git  
-3        Git

—  
-3

```
$ git am -3 0001-seeing-if-this-helps-the-gem.patch
Applying: seeing if this helps the gem
error: patch failed: ticgit.gemspec:1
error: ticgit.gemspec: patch does not apply
Using index info to reconstruct a base tree...
Falling back to patching base and 3-way merge...
No changes -- Patch already applied.
```

-3

mbox  
am

```
$ git am -3 -i mbox
Commit Body is:
-----
seeing if this helps the gem
-----
Apply? [y]es/[n]o/[e]dit/[v]iew patch/[a]ccept all
```

URL

Jessica  
client

ruby-

```
$ git remote add jessica git://github.com/jessica/myproject.git
$ git fetch jessica
$ git checkout -b rubyclient jessica/ruby-client
```

—

Git

-3

URL        git pull  
URL

```
$ git pull https://github.com/onetimeguy/project
From https://github.com/onetimeguy/project
 * branch            HEAD      -> FETCH_HEAD
Merge made by recursive.
```

master
--not
master..contrib

**contrib**

```
$ git log contrib --not master
commit 5b6235bd297351589efc4d73316f0a68d484f118
Author: Scott Chacon <schacon@gmail.com>
Date:   Fri Oct 24 09:53:59 2008 -0700

    seeing if this helps the gem

commit 7482e0d16d04bea79d0dba8988cc78df655f16a0
Author: Scott Chacon <schacon@gmail.com>
Date:   Mon Oct 22 19:38:36 2008 -0700

    updated the gemspec to hopefully work better
```

git log  
dif  
dif

```
$ git diff master
```

dif  
master  
Git master  
 master

master  
 dif  
 master

—  
master  
Git master

dif

```
$ git merge-base contrib master
36c7dba2c95e6bbb78dfa822519ecfec6e1ca649
$ git diff 36c7db
```

Git  
diff ...

```
$ git diff master...contrib
```

master

## 合并工作流

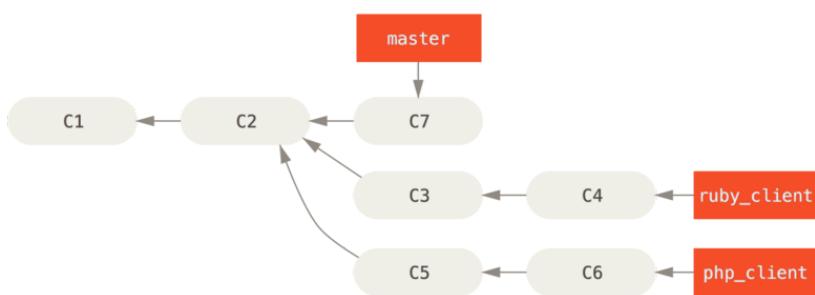


Figure 5-20

ruby\_client    php\_client

php\_client

Figure 5-21

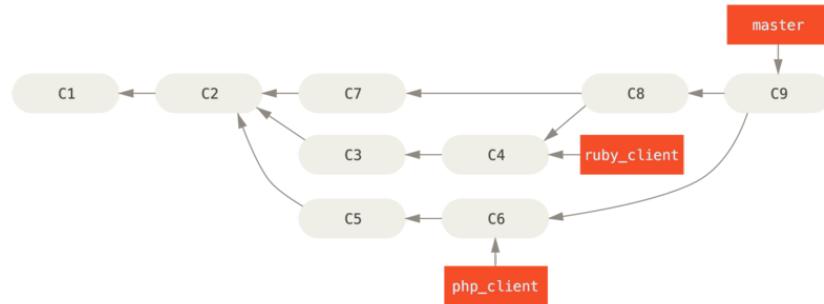


**FIGURE 5-20**

包含若干特性分支的提交历史。

**FIGURE 5-21**

合并特性分支之后。



develop

Figure 5-23  
develop

Figure 5-22

Figure 5-24

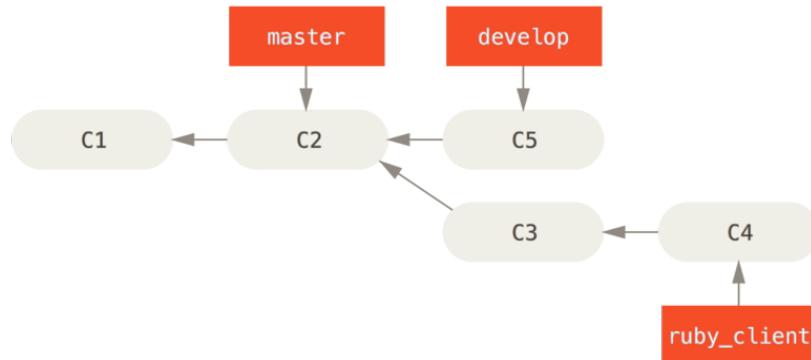
master develop master

develop

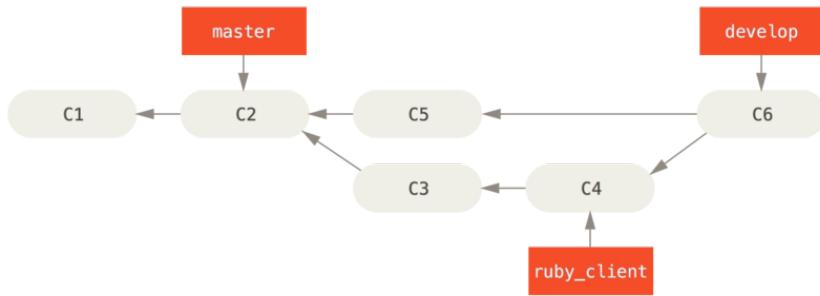
master

**FIGURE 5-22**

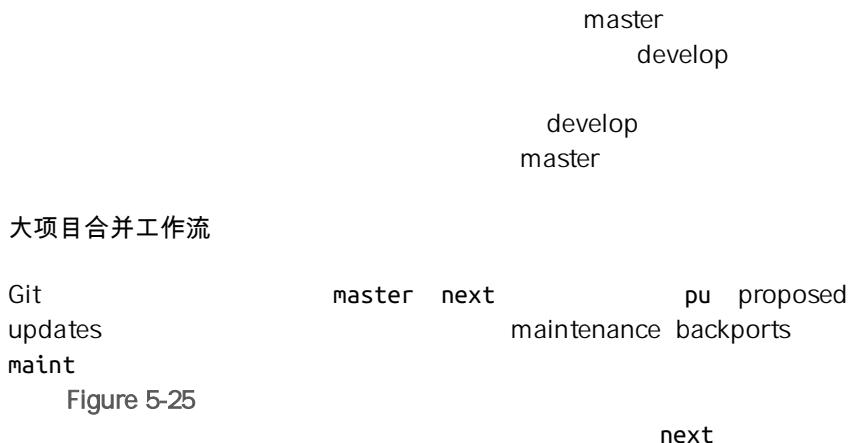
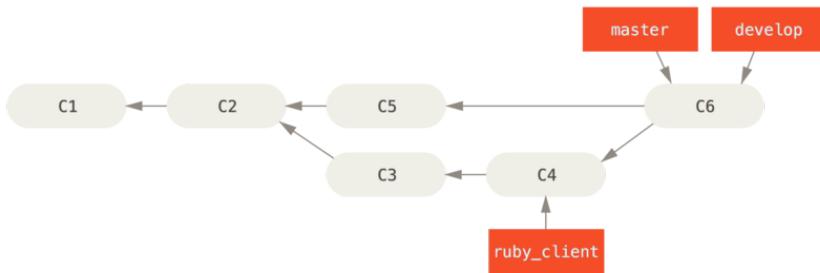
合并特性分支前。



**FIGURE 5-23**  
合并特性分支后。

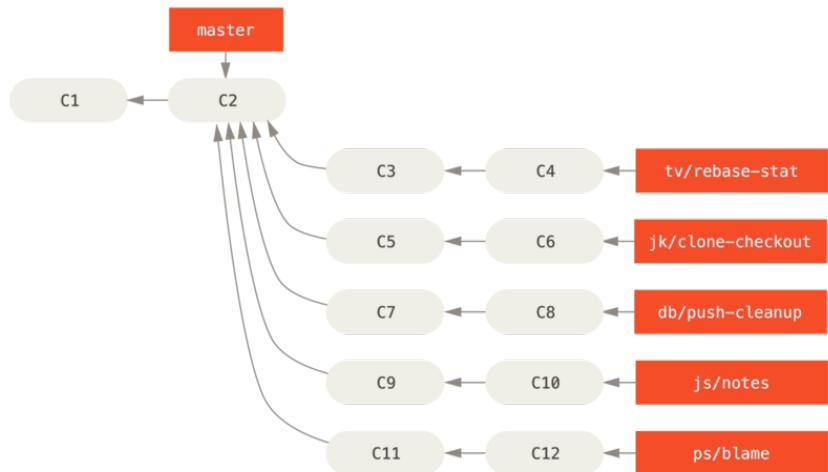


**FIGURE 5-24**  
一次发布之后。

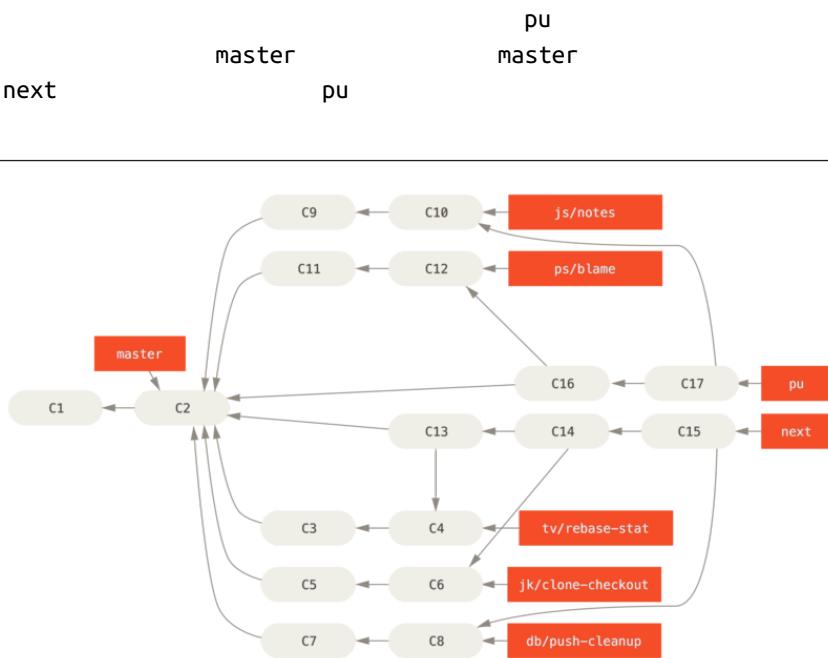


**FIGURE 5-25**

管理复杂的一系列接收贡献的平行特性分支。

**FIGURE 5-26**

将贡献的特性分支并入长期整合分支。



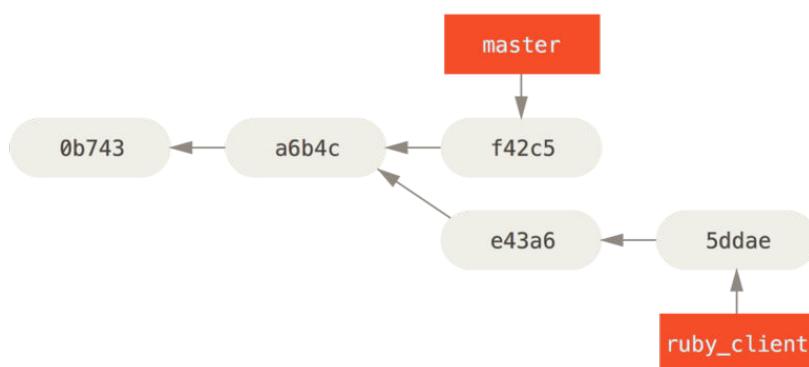
master  
maint  
Git

## 变基与拣选工作流



**FIGURE 5-27**

拣选之前的示例历史。



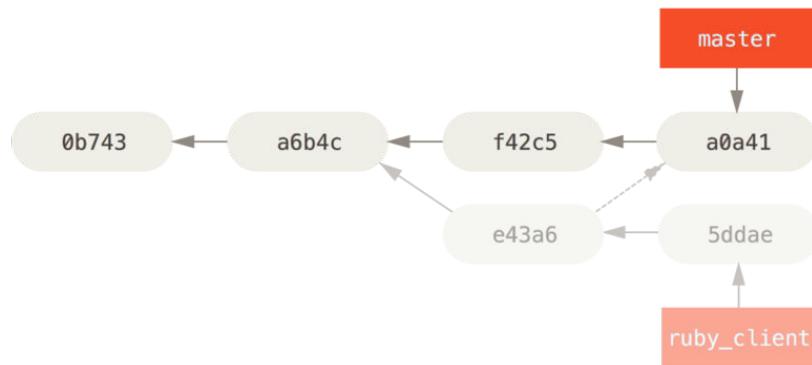
e43a6      master

```
$ git cherry-pick e43a6fd3e94888d76779ad79fb568ed180e5fcdf
Finished one cherry-pick.
[master]: created a0a41a9: "More friendly message when locking the index fails."
 3 files changed, 17 insertions(+), 3 deletions(-)
```

e43a6  
SHA-1

**FIGURE 5-28**

拣选特性分支中的一个提交后的历史。

**RERERE**

Git  
" rerere"  
Rerere " —  
Git  
reuse recorded resolution  
rerere Git

`rerere.enabled`

```
$ git config --global rerere.enabled true
```

rerere  
Git  
" Rerere"  
git rerere  
rerere.enabled  
true

## Chapter 2

```
$ 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@gmail.com>"  
1024-bit DSA key, ID F721C45A, created 2009-02-09
```

Git

PGP

blob

```
gpg --list-keys
```

key

```
$ gpg --list-keys  
/Users/schacon/.gnupg/pubring.gpg  
-----  
pub 1024D/F721C45A 2009-02-09 [expires: 2010-02-09]  
uid Scott Chacon <schacon@gmail.com>  
sub 2048g/45D02282 2009-02-09 [expires: 2010-02-09]
```

	key	git hash-object	
key	Git	git hash-object	Git
	blob	blob	SHA-1

```
$ gpg -a --export F721C45A | git hash-object -w --stdin  
659ef797d181633c87ec71ac3f9ba29fe5775b92
```

Git	key	hash-
object	SHA-1	

```
$ git tag -a maintainer-pgp-pub 659ef797d181633c87ec71ac3f9ba29fe5775b92
```

git push --tags	maintainer-pgp-pub	
		blob
GPG	PGP key	

```
$ git show maintainer-pgp-pub | gpg --import
```

key

git show &lt;tag&gt;

Git " v123"  
 git describe Git  
 SHA-1

```
$ git describe master  

v1.6.2-rc1-20-g8c5b85c
```

Git Git git  
 --version  
 git describe -a -s  
 git describe  
 checkout show  
 SHA-1 Linux SHA-1  
 8 10 git de-  
 scribe

Git  
 git archive

```
$ git archive master --prefix='project/' | gzip > `git describe master`.tar.gz  

$ ls *.tar.gz  

v1.6.2-rc1-20-g8c5b85c.tar.gz
```

```
archive          --format=zip
```

```
$ git archive master --prefix='project/' --format=zip > `git describe master`.zip
```

tar zip

```
shortlog      git
changelog
v1.0.1
```

```
$ git shortlog --no-merges master --not v1.0.1
Chris Wanstrath (8):
    Add support for annotated tags to Grit::Tag
    Add packed-refs annotated tag support.
    Add Grit::Commit#to_patch
    Update version and History.txt
    Remove stray `puts`
    Make ls_tree ignore nils

Tom Preston-Werner (4):
    fix dates in history
    dynamic version method
    Version bump to 1.0.2
    Regenerated gemspec for version 1.0.2
```

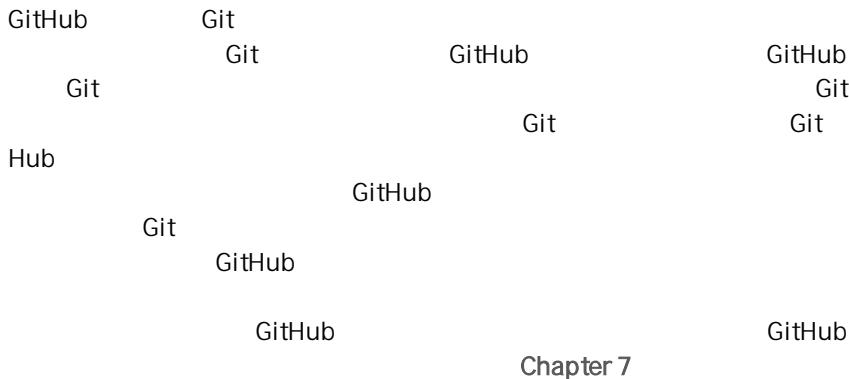
v1.0.1

## 总结

```
Git
Git
Git      GitHub
```



# GitHub 6



---

## 接口的改变

需要注意一点，同很多活跃的网站一样，书中截取的界面会随时间而改变。希望我们试图表达的核心思想一直是不变的，但是，如果你想要这些截图的更新版本，本书的在线版本或许有更新的截图。

---

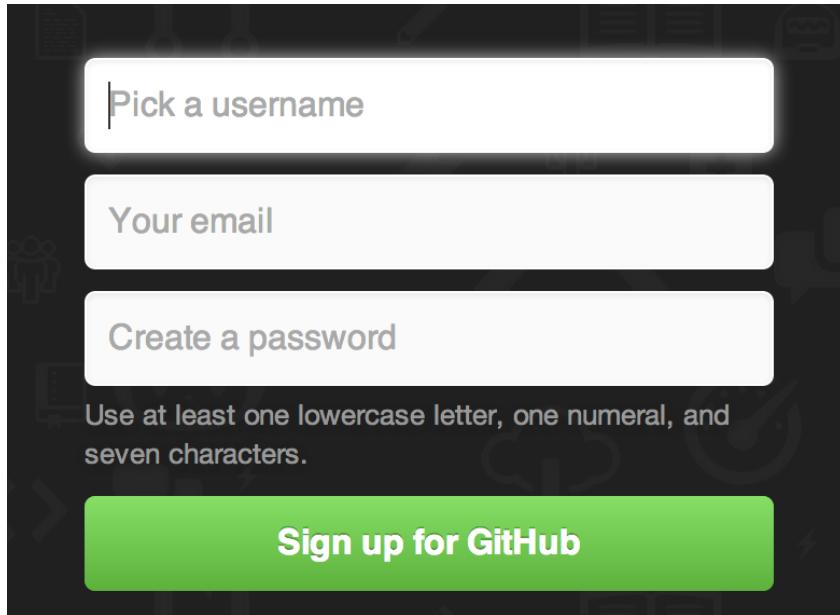
## 账户的创建和配置

*github.com*  
“Sign up for GitHub”

*https://*

**FIGURE 6-1**

GitHub 注册表单。



GitHub

---

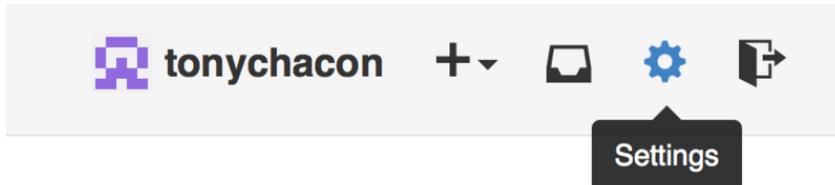
GitHub 为免费账户提供了完整功能，限制是你的项目都将被完全公开（每个人都具有读权限）。GitHub 的付费计划可以让你拥有一定数目的私有项目，不过本书将不涉及这部分内容。

---

Octocat  
GitHub

SSH

https://  
Git  
fork  
SSH  
SSH



**FIGURE 6-2**

"Account settings"链接。

“SSH keys”

A screenshot of the "SSH Keys" section of the GitHub account settings. On the left is a sidebar with links: Profile, Account settings (which is highlighted), Emails, Notification center, Billing, SSH keys (which is also highlighted), Security, Applications, Repositories, and Organizations. The main area has a heading "SSH Keys" with a sub-section "Add an SSH Key". It includes fields for "Title" (with a text input box) and "Key" (with a large text area). A green "Add key" button is at the bottom. Above the main area, there's a note: "Need help? Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#)".

**FIGURE 6-3**

"SSH keys"链接。

“Add an SSH key”

```
~/.ssh/id_rsa.pub  
“Add key”
```

确保给你的 SSH 密钥起一个能够记得住的名字。你可以为每一个密钥起名字（例如，“我的笔记本电脑”或者“工作账户”等），以便以后需要吊销密钥时能够方便地区分。

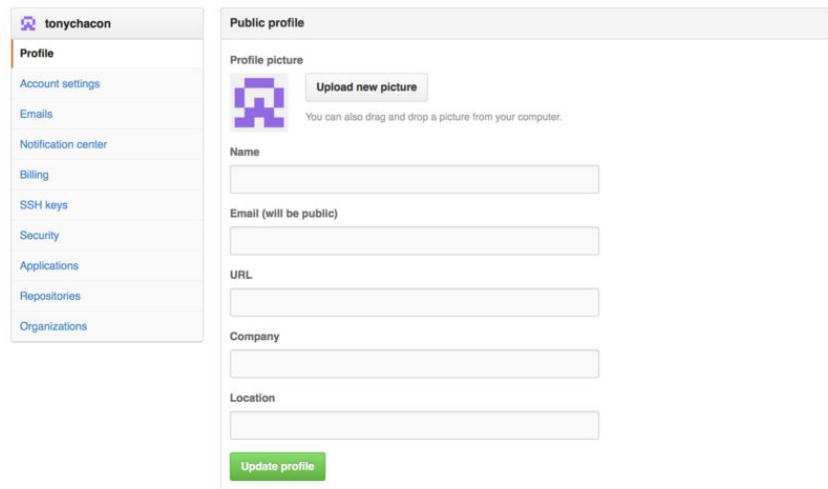
“Profile”  
new picture”

“SSH Keys”

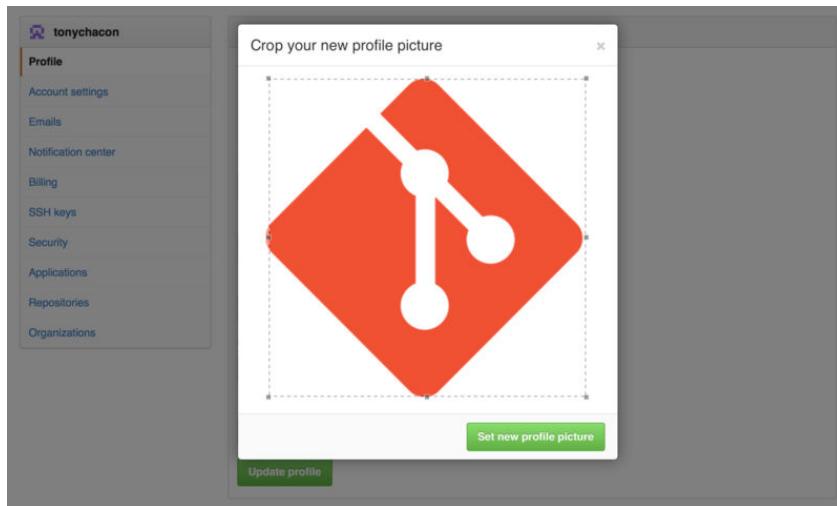
“Upload”

**FIGURE 6-4**

“Profile”链接。

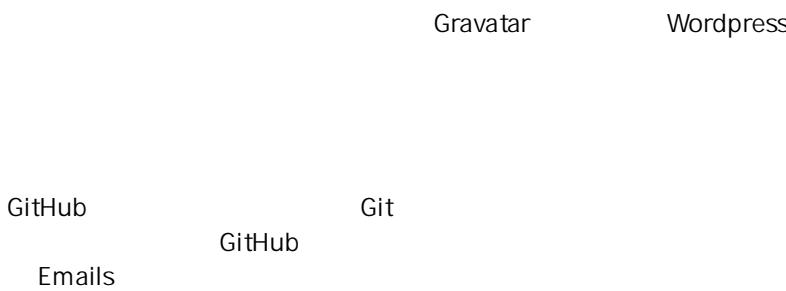


Git



**FIGURE 6-5**

裁剪头像



Gravatar

Wordpress

GitHub  
GitHub  
Emails

Git

Email	
Your <b>primary GitHub email address</b> will be used for account-related notifications (e.g. account changes and billing receipts) as well as any web-based GitHub operations (e.g. edits and merges).	
<input checked="" type="checkbox"/> tonychacon@example.com	<span>Primary</span> <span>Public</span>
<input type="checkbox"/> tchacon@example.com	<span>Unverified</span>
<input type="button" value="Send verification email"/>	
<b>Add email address</b> <input type="text"/> <input type="button" value="Add"/>	
<input type="checkbox"/> <b>Keep my email address private</b> <small>We will use <b>tonychacon@users.noreply.github.com</b> when performing Git operations and sending email on your behalf.</small>	

**FIGURE 6-6**

添加邮件地址

Figure 6-6

GitHub

“2FA”

GitHub

Account settings  
Authentication

Security

Two-factor

**FIGURE 6-7**Security 标签页中的  
2FA

The screenshot shows the GitHub Security page for the user 'tonychacon'. On the left, a sidebar menu includes 'Profile', 'Account settings', 'Emails', 'Notification center', 'Billing', 'SSH keys', 'Security' (which is selected), 'Applications', 'Repositories', and 'Organizations'. The main content area has a header 'Two-factor authentication'. It displays the status as 'Off' with a red 'X' icon. A button labeled 'Set up two-factor authentication' is present. Below this, a note explains that two-factor authentication provides another layer of security to the account, with links to 'Learn more about two-factor auth' and 'GitHub Help'. At the bottom, a section titled 'Sessions' lists a single session: 'Paris 85.168.227.34' (Your current session), 'Safari on OS X 10.9.4', located in 'Paris, Ile-de-France, France', and signed in on 'September 30, 2014'.

“ Set up two-factor authentication ”

app

GitHub

SMS

2FA

GitHub

## 对项目做出贡献

Fork

GitHub

---

在以前，“fork”是一个贬义词，指的是某个人使开源项目向不同的方向发展，或者创建一个竞争项目，使得原项目的贡献者分裂。在 GitHub，“fork”指的是你自己的空间中创建的项目副本，这个副本允许你以一种更开放的方式对其进行修改。

---

Pull Request

“Fork”



**FIGURE 6-8**

“Fork”按钮

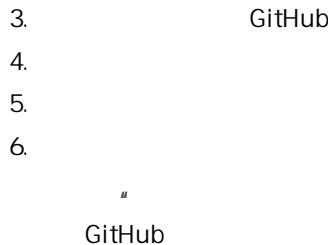
GitHub

GitHub  
Chapter 3

“Fork”

1. master

2.



### 创建合并请求

Tony Arduino <https://github.com/schacon/blink>

**FIGURE 6-9**

他想要做出贡献的项目

The screenshot shows a GitHub repository page for 'schacon/blink'. The repository has 0 stars, 0 forks, and 0 contributors. The master branch contains a single file, 'blink.ino', which is 25 lines (20 sloc) and 0.71 kb. The code is as follows:

```
1 //  
2 // BLINK  
3 // Turns on an LED on for one second, then off for one second, repeatedly.  
4 //  
5 // This example code is in the public domain.  
6 //  
7 // PIn 13 has an LED connected on most Arduino boards.  
8 // give it a name:  
9 int led = 13;  
10 // the setup routine runs once when you press reset:  
11 void setup() {  
12     // initialize the digital pin as an output.  
13     pinMode(led, OUTPUT);  
14 }  
15 // the loop routine runs over and over again forever:  
16 void loop() {  
17     digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)  
18     delay(1000); // wait for a second  
19     digitalWrite(led, LOW); // turn the LED off by making the voltage LOW  
20     delay(1000); // wait for a second  
21 }
```

3 1

“ Fork”  
“ tonychacon ”  
<https://github.com/tonychacon/blink>  
GitHub

```

$ git clone https://github.com/tonychacon/blink ❶
Cloning into 'blink'...

$ cd blink
$ git checkout -b slow-blink ❷
Switched to a new branch 'slow-blink'

$ sed -i '' 's/1000/3000/' blink.ino ❸

$ git diff --word-diff ❹
diff --git a/blink.ino b/blink.ino
index 15b9911..a6cc5a5 100644
--- a/blink.ino
+++ b/blink.ino
@@ -18,7 +18,7 @@ void setup() {
// the loop routine runs over and over again forever:
void loop() {
    digitalWrite(led, HIGH);    // turn the LED on (HIGH is the voltage level)
[-delay(1000);-]{+delay(3000);+}          // wait for a second
    digitalWrite(led, LOW);     // turn the LED off by making the voltage LOW
[-delay(1000);-]{+delay(3000);+}          // wait for a second
}

$ git commit -a -m 'three seconds is better' ❺
[slow-blink 5ca509d] three seconds is better
 1 file changed, 2 insertions(+), 2 deletions(-)

$ git push origin slow-blink ❻
Username for 'https://github.com': tonychacon
Password for 'https://tonychacon@github.com':
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 340 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To https://github.com/tonychacon/blink
 * [new branch]      slow-blink -> slow-blink

```

❶

❷

❸

❹

❺



“Branches”  
<https://github.com/<用户名>/<项目名>/branches>

**FIGURE 6-10**

合并请求按钮

The screenshot shows the GitHub repository page for `tonychacon / blink`. The repository was forked from `schacon/blink`. It has 2 commits, 2 branches, 0 releases, and 1 contributor. A recent push to the `slow-blink` branch is shown. The `master` branch is selected. The repository contains files `README.md` and `blink.ino`. The `README.md` file describes an example file to blink the LED on an Arduino board. On the right side, there are links for Code, Pull Requests, Wiki, Pulse, Graphs, Settings, and cloning options via HTTPS, SSH, or Subversion.

" " ahead

**FIGURE 6-11**

合并请求创建页面

The screenshot shows a GitHub pull request interface. At the top, it says "tonychacon / blink" and "forked from schacon/blink". There are buttons for "Unwatch", "Star", and "Fork". The main area has tabs "Write" and "Preview". The "Preview" tab is selected, showing a title "Three seconds is better" and a body of text: "Studies have shown that 3 seconds is a far better LED delay than 1 second." Below the preview is a note: "Attach images by dragging & dropping or selecting them." On the right, there's a sidebar with a green checkmark icon and the text "Able to merge. These branches can be automatically merged." A large green button at the bottom right says "Create pull request". Below the preview, there's a summary bar with "1 commit", "1 file changed", "0 commit comments", and "1 contributor". The commit details show "Commits on Oct 01, 2014" by "tonychacon" with the commit message "three seconds is better" and hash "db44c53". The "Unified" tab is selected in the diff viewer, which shows a single file "blink.ino" with changes. The diff highlights two additions and two deletions. The code snippet is as follows:

```
4 4 blink.ino
  @@ -18,7 +18,7 @@ void setup() {
18 18 // the loop routine runs over and over again forever:
19 19 void loop() {
20 20     digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
21 21     - delay(1000);          // wait for a second
22 22     + delay(3000);          // wait for a second
23 23     digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
24 24     - delay(1000);          // wait for a second
25 25     + delay(3000);          // wait for a second
26 }
```

“Create pull request”

---

虽然合并请求通常是在贡献者准备好在公开项目中提交改动的时候提交，但是也常被用在仍处于开发阶段的内部项目中。因为合并请求在提交后 **依然可以加入新的改动**，它也经常被用来建立团队合作的环境，而不只是在最终阶段使用。

---

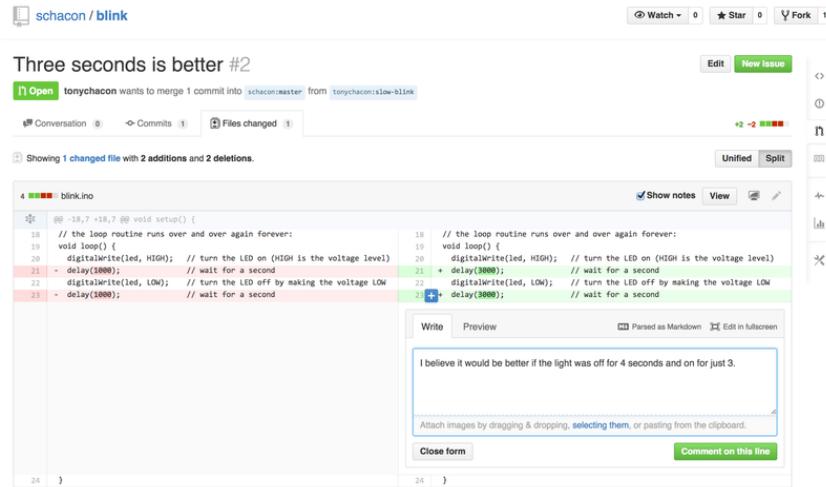
## 利用合并请求

Chapter 5

GitHub

**FIGURE 6-12**

对合并请求内的特定  
一行发表评论

**FIGURE 6-13**

通过电子邮件发送的  
评论提醒



Figure 6-14

**FIGURE 6-14**

合并请求讨论页面

Three seconds is better #2

**Open** tonychacon wants to merge 1 commit into `schacon:master` from `tonychacon:slow-blink`

Conversation 1 Commits 1 Files changed 1

tonychacon commented 6 minutes ago

Studies have shown that 3 seconds is a far better LED delay than 1 second.  
<http://studies.example.com/optimal-led-delays.html>

three seconds is better

db44c53

schacon commented on the diff just now

View full changes

blink.ino

```
22 22  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
23 - delay(1000); // wait for a second
23 + delay(3000); // wait for a second
```

schacon added a note just now

I believe it would be better if the light was off for 4 seconds and on for just 3.

Add a line note

schacon commented just now

If you make that change, I'll be happy to merge this.

GitHub

GitHub

**FIGURE 6-15**

最终的合并请求

**Three seconds is better #2**

**Open** tonychacon wants to merge 3 commits into `schacon:master` from `tonychacon:slow-blink`

Conversation 3 Commits 3 Files changed 1

**tonychacon** commented 11 minutes ago

Studies have shown that 3 seconds is a far better LED delay than 1 second.

<http://studies.example.com/optimal-led-delays.html>

three seconds is better db44c53

**schacon** commented on an outdated diff 5 minutes ago Show outdated diff

**schacon** commented 5 minutes ago

If you make that change, I'll be happy to merge this.

**tonychacon** added some commits 2 minutes ago

- longer off time 0c1f66f
- remove trailing whitespace ef4725c

**tonychacon** commented 10 seconds ago

I changed it to 4 seconds and also removed some trailing whitespace that I found. Anything else you would like me to do?

This pull request can be automatically merged. You can also merge branches on the command line. Merge pull request

“ Files Changed”

`git diff master...<分支名>`

GitHub

GitHub

fast-

non-fast-forward  
forward GitHub

master GitHub  
GitHub

---

## 不必总是 **FORK**

---

有件很重要的事情：你可以在同一个版本库中不同的分支提交合并请求。如果你正在和某人实现某个功能，而且你对项目有写权限，你可以推送分支到版本库，并在 master 分支提交一个合并请求并在此进行代码审查和讨论的操作。不需要进行“Fork”。

---

GitHub

将合并请求制作成补丁

GitHub

Figure 6-15

“ Merge ”

与上游保持同步

GitHub



This pull request contains merge conflicts that must be resolved.  
Only those with write access to this repository can merge pull requests.



FIGURE 6-16

不能进行干净合并

Figure 6-16

master

GitHub

" tonychacon"

```
$ git remote add upstream https://github.com/schacon/blink ❶

$ git fetch upstream ❷
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
Unpacking objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0)
From https://github.com/schacon/blink
 * [new branch]      master      -> upstream/master

$ git merge upstream/master ❸
Auto-merging blink.ino
CONFLICT (content): Merge conflict in blink.ino
Automatic merge failed; fix conflicts and then commit the result.

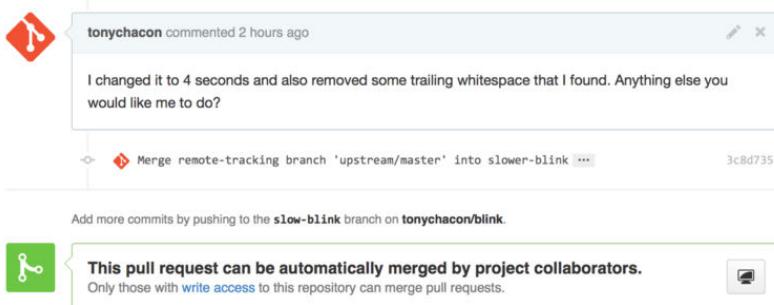
$ vim blink.ino ❹
$ git add blink.ino
$ git commit
[slow-blink 3c8d735] Merge remote-tracking branch 'upstream/master' \
  into slower-blink

$ git push origin slow-blink ❺
Counting objects: 6, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 682 bytes | 0 bytes/s, done.
Total 6 (delta 2), reused 0 (delta 0)
To https://github.com/tonychacon/blink
  ef4725c..3c8d735  slower-blink -> slow-blink
```

❶

" upstream"

- ②
- ③
- ④
- ⑤



**FIGURE 6-17**

合并请求现在可以干净地合并了

Git

GitHub

参考



用户名#<编号>  
版本库名#<编号>

Figure 6-18

FIGURE 6-18

在合并请求中的交叉引用

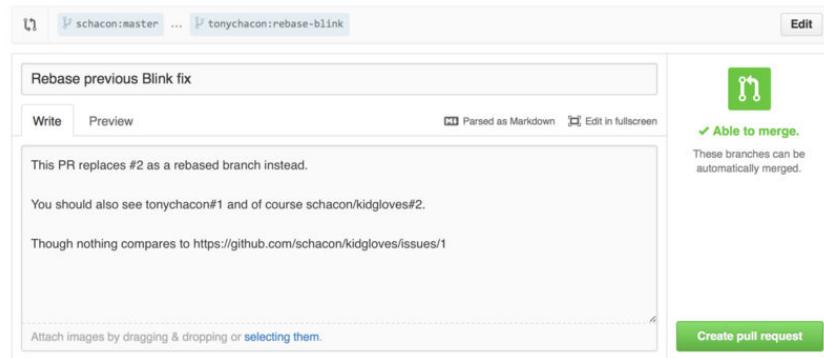
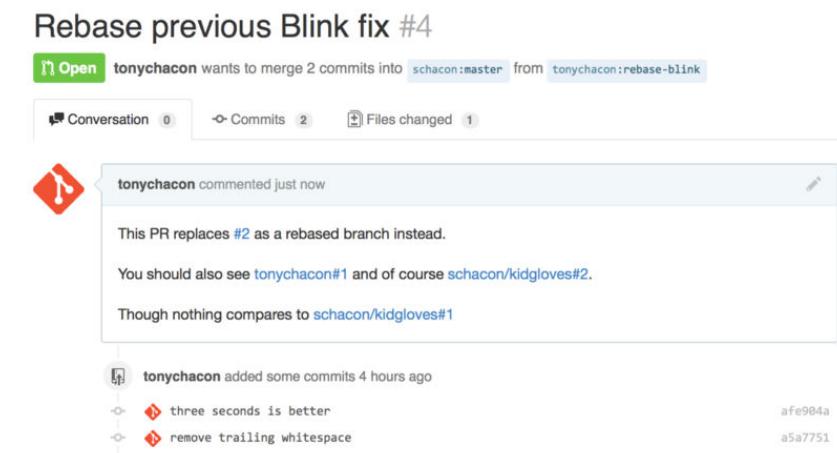


Figure 6-19

FIGURE 6-19

在合并请求中渲染后的交叉引用



GitHub  
Tony  
GitHub

Figure 6-20

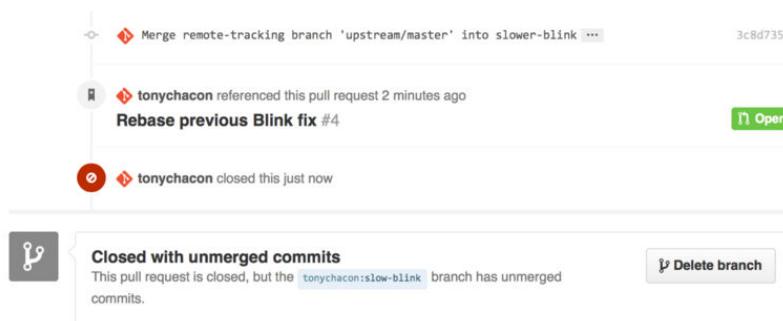


FIGURE 6-20

在合并请求中渲染后的交叉引用

SHA-1  
40      SHA GitHub  
" Fork"

## Markdown

GitHub  
" GitHub      Markdown"      Markdown

Figure 6-21  
down

Mark

**FIGURE 6-21**

一个 Markdown 的例子和渲染效果

The screenshot shows two side-by-side views of GitHub's Markdown editor. On the left, the 'Write' tab is active, displaying a code block with a warning about a 'big' problem with the blink code. The right side shows the rendered output, which includes a red diamond icon, the word 'git', and a quote from tonychacon.

**A Markdown Example**

Write Preview Parse as Markdown Edit in fullscreen

There is a “big” problem with the blink code. Not with the idea, but with the `_code_`.

```
## What is the problem?
```

As you can see [here](https://github.com/schacon/blink/blob/master/blink.ino#L10), the LED uses the number 13 which has the following issues:

- \* It is unlucky
- \* It is two decimal places

The if we replace `int led = 13;` with `int led = 7;`, it will be far more lucky.

As Kanye West said:

>We're living the future so  
>the present is our past.

git logo(<https://logos.example.com/git-logo.png>)

Attach images by dragging & dropping or [selecting them](#).

Submit new issue

**tonychacon commented just now**

There is a **big** problem with the blink code. Not with the idea, but with the code.

**What is the problem?**

As you can see [here](#), the LED uses the number 13 which has the following issues:

- It is unlucky
- It is two decimal places

The if we replace `int led = 13;` with `int led = 7;`, it will be far more lucky.

As Kanye West said:

We're living the future so  
the present is our past.

**git**

### GITHUB 风格的 MARKDOWN

GitHub

Markdown

Markdown

GitHub

Markdown

- [X] 编写代码
- [ ] 编写所有测试程序
- [ ] 为代码编写文档

Figure 6-22

**FIGURE 6-22**

Markdown 评论中渲染后的任务列表

The screenshot shows a GitHub pull request comment from tonychacon. The comment text is: "This PR replaces #2 as a rebased branch instead." Below the text is a rendered task list with three items: "Write the code" (checked), "Write all the tests" (unchecked), and "Document the code" (unchecked).

tonychacon commented 4 hours ago

This PR replaces #2 as a rebased branch instead.

- Write the code
- Write all the tests
- Document the code

Markdown

Figure 6-23

**FIGURE 6-23**

在合并请求列表中的任务列表总结

```
```java
for(int i=0 ; i < 5 ; i++)
{
    System.out.println("i is : " + i);
}
```
" java" GitHub
```

Figure 6-24

**FIGURE 6-24**

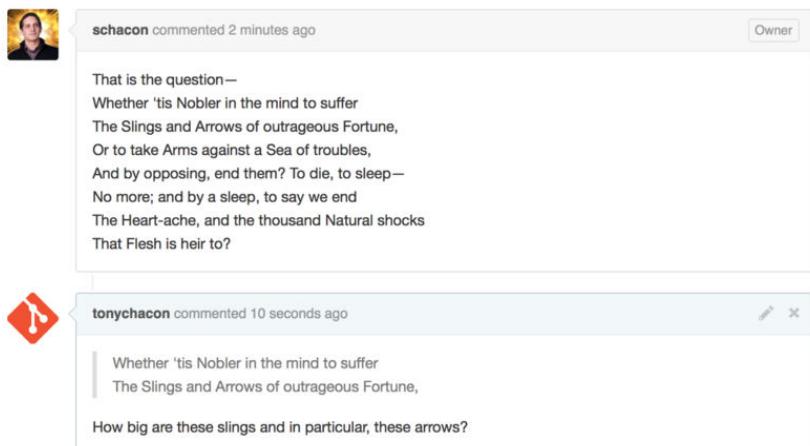
渲染后的摘录代码示例

>  
|  
|  
|  
> Whether 'tis Nobler in the mind to suffer  
> The Slings and Arrows of outrageous Fortune,  
  
How big are these slings and in particular, these arrows?

Figure 6-25

**FIGURE 6-25**

渲染后的引用示例



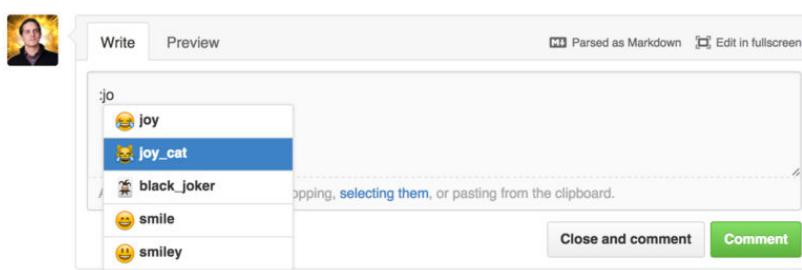
Emoji

GitHub

GitHub

:

**FIGURE 6-26**



表情符号自动完成器

:<表情名称>:

```
I :eyes: that :bug: and I :cold_sweat:.

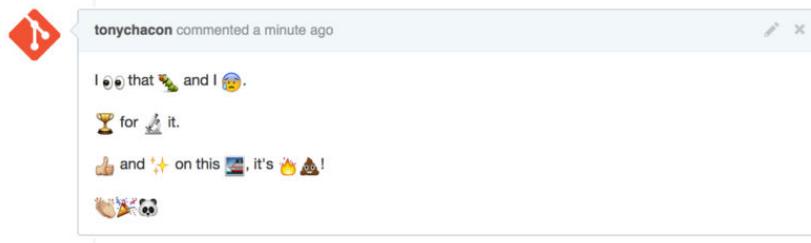
:trophy: for :microscope: it.

:+1: and :sparkles: on this :ship:, it's :fire::poop:!

:clap::tada::panda_face:
```

Figure 6-27

**FIGURE 6-27**



使用了大量表情符号的评论

事实上现在已经有大量的在线服务可以使用表情符号，这里有个列表可以让你快速的找到能表达你的情绪的表情符号：

<http://www.emoji-cheat-sheet.com>

**FIGURE 6-28**

通过拖拽的方式自动  
插入图片

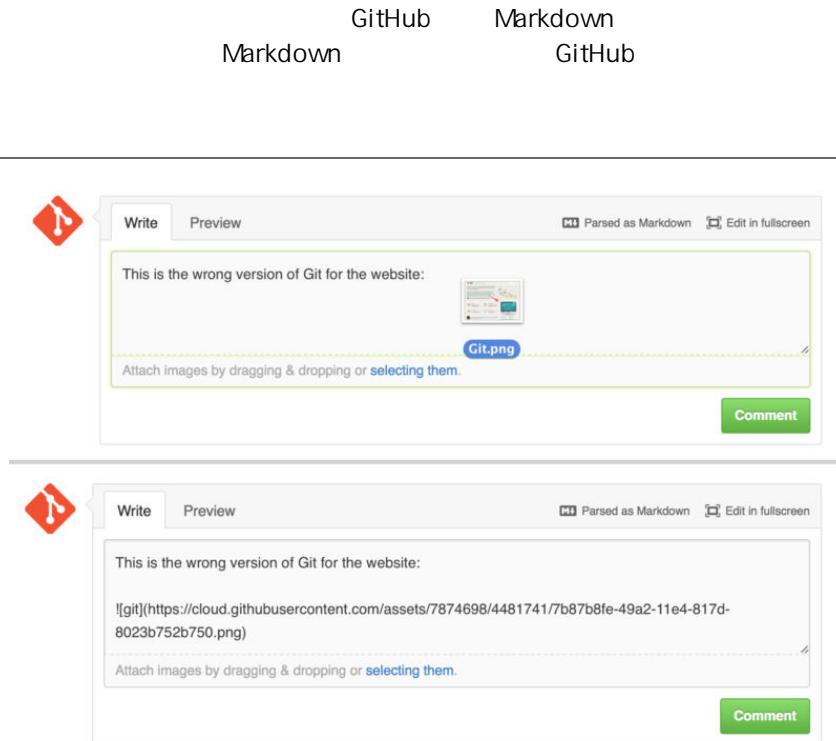


Figure 6-18 “Parsed as GitHub”  
“Parsed as Mark”

## 维护项目

repository” “New  
+

Figure 6-30

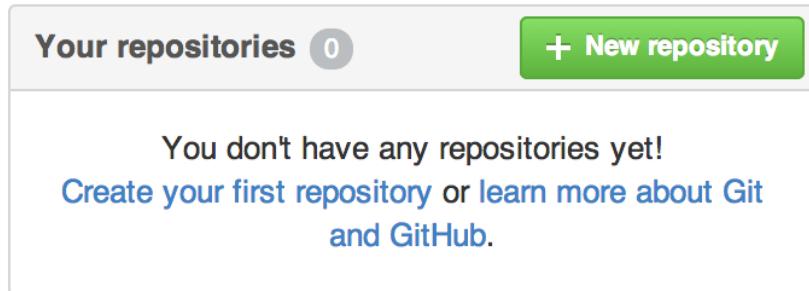


FIGURE 6-29

这是“Your repositories”区域。

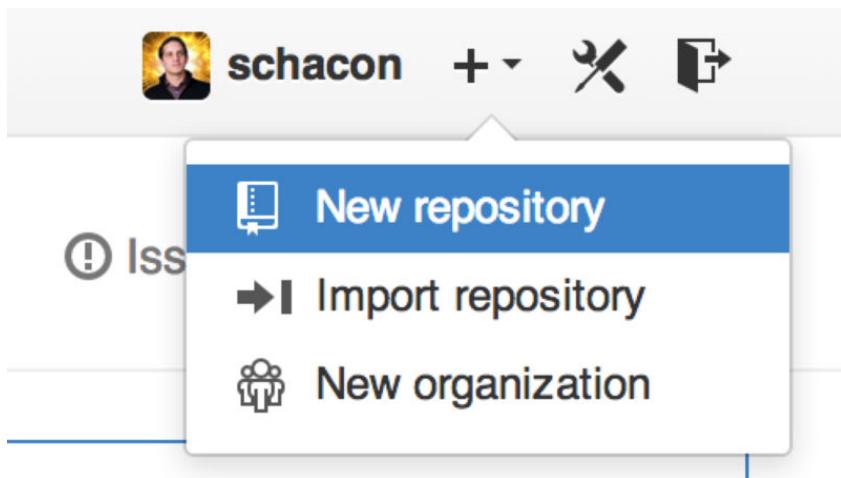


FIGURE 6-30

这是“New repository”下拉列表。

" new repository" :

**FIGURE 6-31**

这是“new repository”表单。

PUBLIC Owner / Repository name  
 ben / iOSApp ✓  
 Great repository names are short and memorable. Need inspiration? How about [drunken-dubstep](#).  
 Description (optional)  
 iOS project for our mobile group  
 **Public**  
 Anyone can see this repository. You choose who can commit.  
 **Private**  
 You choose who can see and commit to this repository.  
 Initialize this repository with a README  
 This will allow you to `git clone` the repository immediately. Skip this step if you have already run `git init` locally.  
 Add .gitignore: **None** | Add a license: **None** ⓘ  
 Create repository

“ Create Repository”      Duang!!! -      GitHub  
 <user>/<project\_name>  
 GitHub  
 Git  
 Chapter 2  
 GitHub      URL  
 GitHub      HTTP      SSH      HTTP  
 https://github.com/<user>/<project\_name>      SSH      git@git-  
 hub.com:<user>/<project\_name>      Git      URL

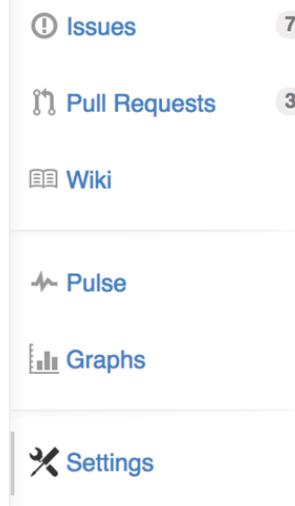
---

通常对于公开项目可以优先分享基于 HTTP 的 URL，因为用户克隆项目不需要有一个 GitHub 帐号。如果你分享 SSH URL，用户必须有一个帐号并且上传 SSH 密钥才能访问你的项目。HTTP URL 与你贴到浏览器里查看项目用的地址是一样的。

---

“ Collaborators”      Ben    Jef    Louise      GitHub  
 ”      “  
 ”      Git  
 “ Settings”

**FIGURE 6-32**



版本库设置链接。

“ Collaborators”  
“ Add collaborator.”  
“ X”

**FIGURE 6-33**

版本库合作者。

A screenshot of the GitHub repository settings page under the "Collaborators" tab. It shows three collaborators: Ben Straub, Jeff King, and Louise Corrigan, each with a small profile picture and their GitHub handle. Below the list is a search bar labeled "Type a username" and a button labeled "Add collaborator".

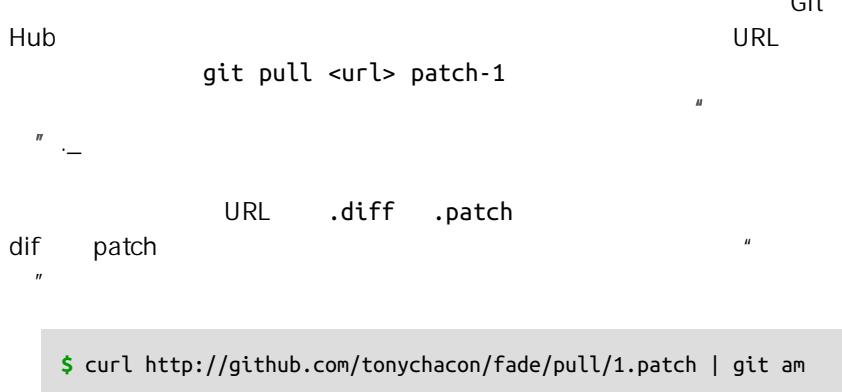
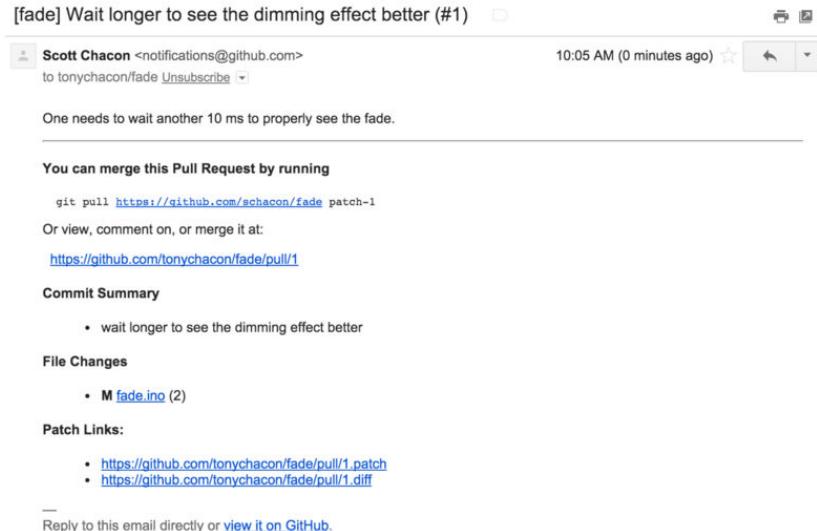
fork  
“ tonychacon”  
Arduino .  
“ fade”

## 邮件通知

Figure 6-34

**FIGURE 6-34**

新的合并请求的邮件  
通知。

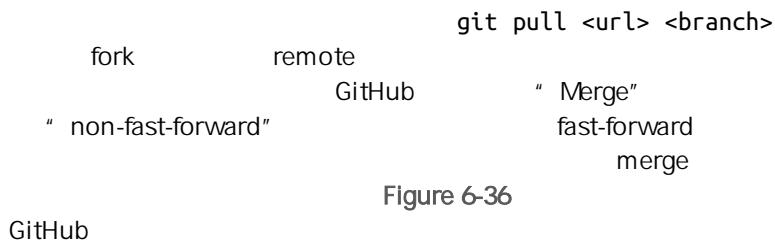


## 在合并请求上进行合作



**FIGURE 6-35**

*Responses to emails are included in the thread.*



GitHub

**FIGURE 6-36**

合并按钮和手工合并  
一个合并请求的指令。

This pull request can be automatically merged.  
You can also merge branches on the command line.

**Merging via command line**  
If you do not want to use the merge button or an automatic merge cannot be performed, you can perform a manual merge on the command line.

HTTP Git Patch https://github.com/schacon/fade.git

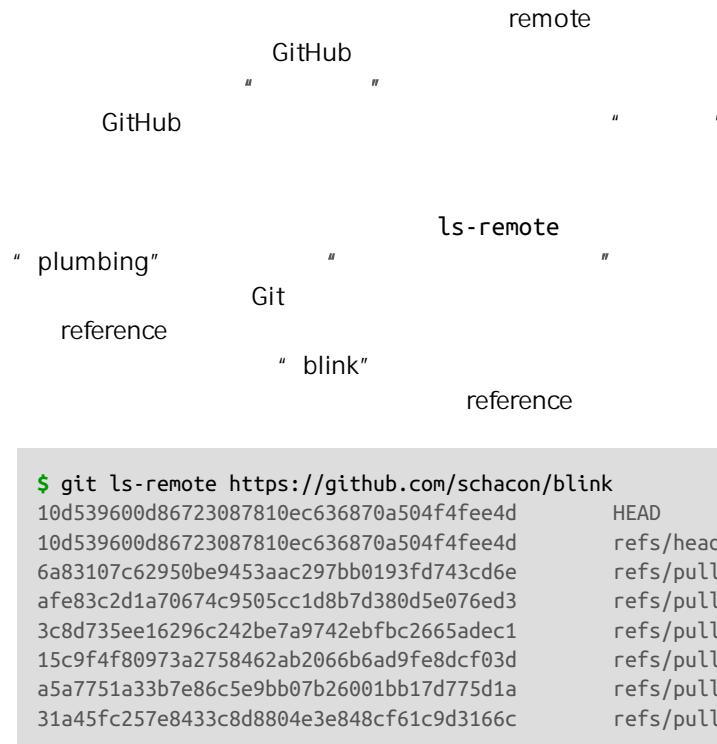
Step 1: From your project repository, check out a new branch and test the changes.

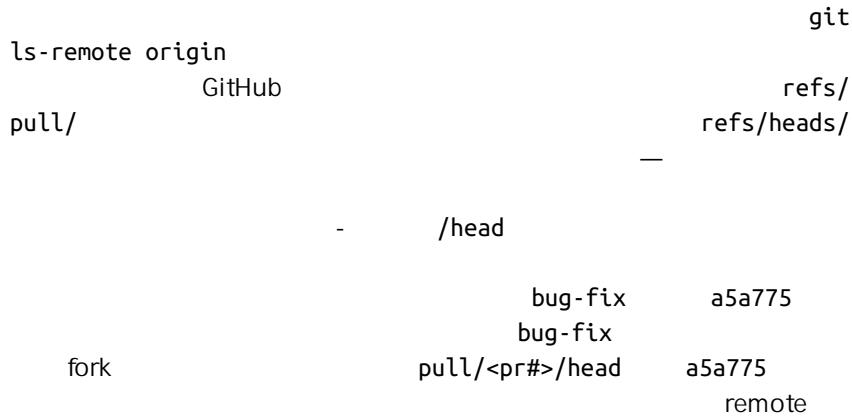
```
git checkout -b schacon-patch-1 master
git pull https://github.com/schacon/fade.git patch-1
```

Step 2: Merge the changes and update on GitHub.

```
git checkout master
git merge --no-ff schacon-patch-1
git push origin master
```

## 合并请求引用





```
$ git fetch origin refs/pull/958/head
From https://github.com/libgit2/libgit2
 * branch           refs/pull/958/head  -> FETCH_HEAD
```

```
Git      "      origin      remote      refs/
pull/958/head      "  Git
                    .git/FETCH_HEAD
  git merge FETCH_HEAD
```

```
re
mote
origin      remote      .git/config

[remote "origin"]
  url = https://github.com/libgit2/libgit2
  fetch = +refs/heads/*:refs/remotes/origin/*

  fetch =           "  refspec."      remote
  .git
  "  remote      refs/heads      Git
  remotes/origin   "      refs/
                                         refspec

[remote "origin"]
  url = https://github.com/libgit2/libgit2.git
  fetch = +refs/heads/*:refs/remotes/origin/*
  fetch = +refs/pull/*/*:refs/remotes/origin/pr/*
```

```
Git    "          refs/pull/123/head  
refs/remotes/origin/pr/123      "  
     git fetch
```

```
$ git fetch  
# ...  
* [new ref]      refs/pull/1/head -> origin/pr/1  
* [new ref]      refs/pull/2/head -> origin/pr/2  
* [new ref]      refs/pull/4/head -> origin/pr/4  
# ...
```

```
$ git checkout pr/2  
Checking out files: 100% (3769/3769), done.  
Branch pr/2 set up to track remote branch pr/2 from origin.  
Switched to a new branch 'pr/2'
```

|        | refspec           | remote | head |
|--------|-------------------|--------|------|
| GitHub | refs/pull/#/merge |        |      |
|        | " merge"          |        |      |

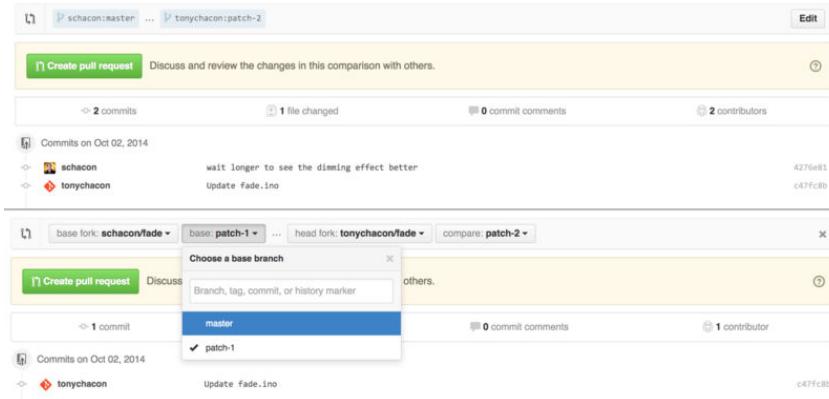
### 合并请求之上的合并请求

master

" Edit"  
fork

**FIGURE 6-37**

手工修改合并请求的目标.



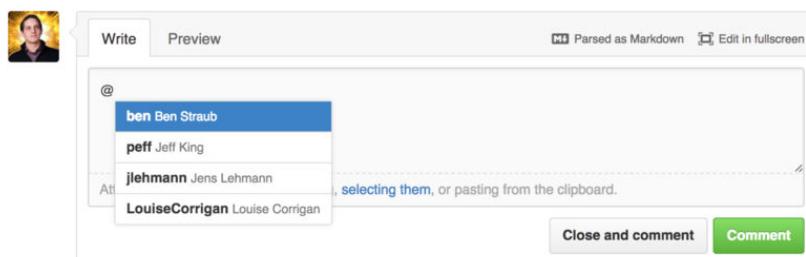
fork

GitHub

@

**FIGURE 6-38**

输入 @ 来提醒某人.



GitHub

“Unsubscribe”

FIGURE 6-39

取消订阅一个问题或  
合并请求。

## Notifications

►× **Unsubscribe**

You're receiving notifications  
because you commented.

GitHub “notifications”

GitHub

“Notification center”

**FIGURE 6-40**

通知中心选项.

The screenshot shows the GitHub notification center settings. On the left is a sidebar with options: Profile, Account settings, Emails, Notification center (which is selected and highlighted in orange), Billing, SSH keys, Security, Applications, Repositories, and Organizations. The main content area is titled "How you receive notifications". It has two sections: "Participating" and "Watching". Under "Participating", it says "When you participate in a conversation or someone brings you in with an @mention." with checkboxes for "Email" and "Web", both of which are checked. Under "Watching", it says "Updates to any repositories or threads you're watching." with checkboxes for "Email" and "Web", both of which are checked. Below this is a section titled "Notification email" with a "Primary email address" input field containing "tchacon@example.com" and a "Save" button. At the bottom is a section titled "Custom routing" with the note "You can send notifications to different **verified** email addresses depending on the organization that owns the repository."

(Email)

(Web)

GitHub

GitHub

Figure 6-41

**FIGURE 6-41**

通知中心.

The screenshot shows the GitHub notifications interface. At the top, there is a search bar, navigation links for Explore, Gist, Blog, Help, and a user profile for tonychacon. A notification badge indicates "You have unread notifications" and a "Mark all as read" button. Below the header, there are tabs for "Notifications" (selected) and "Watching". The main area is divided into sections based on repository names: "Unread" (4 notifications), "Participating" (3 notifications), "All notifications" (1 notification), "schacon/blink" (2 notifications), "mycorp/project1" (1 notification), and "git/git-scm.com" (1 notification). Each notification item includes the repository name, a green circular icon, the notification type (e.g., "SF Corporate Housing Search", "Front Page"), the timestamp ("an hour ago", "3 hours ago", "5 days ago"), and a "mark as read" checkbox.

GitHub

GitHub

Figure 6-13    Figure 6-34

GitHub

Figure 6-34        Tony

To: tonychacon/fade <fade@noreply.github.com>  
Message-ID: <tonychacon/fade/pull/1@github.com>  
Subject: [fade] Wait longer to see the dimming effect better (#1)  
X-GitHub-Recipient: tonychacon  
List-ID: tonychacon/fade <fade.tonychacon.github.com>  
List-Archive: https://github.com/tonychacon/fade  
List-Post: <mailto:reply+i-4XXX@reply.github.com>  
List-Unsubscribe: <mailto:unsub+i-XXX@reply.github.com>, ...  
X-GitHub-Recipient-Address: tchacon@example.com

|           |                        |  |
|-----------|------------------------|--|
| <id>      | Message-ID             | <i>&lt;user&gt;/&lt;project&gt;/&lt;type&gt;/</i><br>(issue) |
| <type>    | " issues"      " pull" |  |
| List-Post | List-Unsubscribe       |  |

" Unsubscribe"

GitHub

## README

README GitHub  
README README.md README.asciidoc GitHub  
README

- 
- 
- 
- 
- 

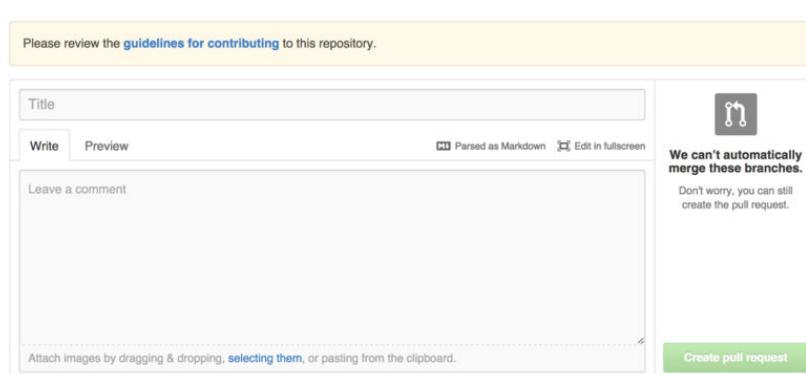
GitHub

## CONTRIBUTING

GitHub CONTRIBUTING

GitHub

Figure 6-42



**FIGURE 6-42**

开启合并请求时有  
CONTRIBUTING 文件  
存在。

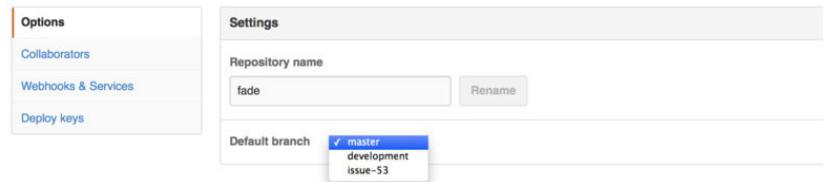
### 改变默认分支

“ master”

“ options”

**FIGURE 6-43**

改变项目的默认分支.



### 移交项目

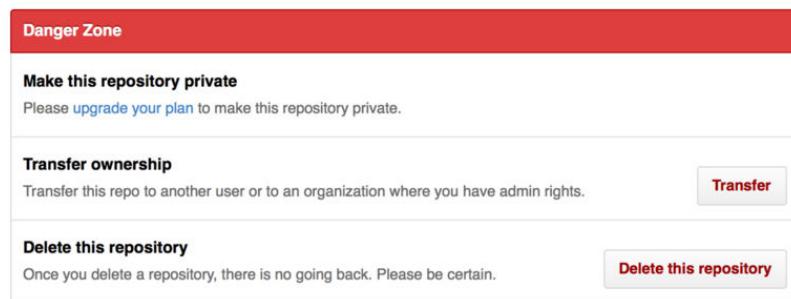
GitHub

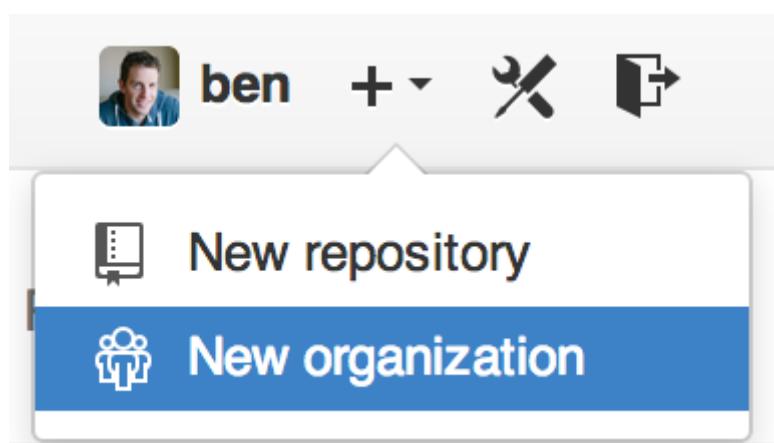
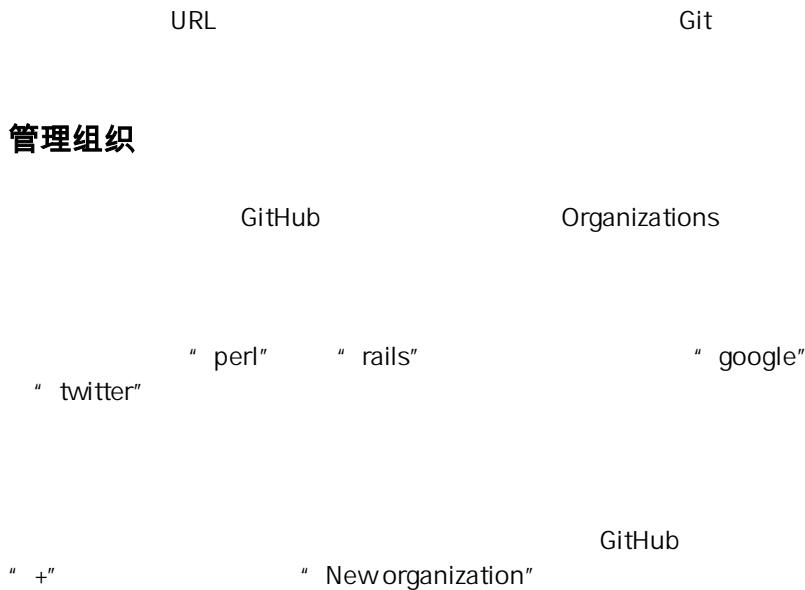
options

“ Transfer ownership”

**FIGURE 6-44**

把项目移交给另一个  
GitHub 用户或组织。





**FIGURE 6-45**  
*"New organization"*  
 菜单项



landing page



FIGURE 6-46

组织页面

The screenshot shows the GitHub organization page for 'chaconcorp'. At the top, there's a purple logo and the organization name 'chaconcorp'. Below the header, there's a search bar with 'Find a repository...' and a '+ New repository' button. The main content area displays three repositories:

- deployscripts**: Scripts for deployment, updated 16 hours ago.
- backend**: Backend Code, updated 16 hours ago.
- frontend**: Frontend Code, updated 16 hours ago.

To the right, there are two sidebar sections: 'People' and 'Teams'.

**People** section:

- dragonchacon (Dragon Chacon)
- schacon (Scott Chacon)
- tonychacon (Tony Chacon)

**Teams** section:

- Owners**: 1 member - 3 repositories
- Frontend Developers**: 2 members - 2 repositories
- Ops**: 3 members - 1 repository

At the bottom right of the sidebar, there's a 'Create new team' button.

Figure 6-46

tings"

The screenshot shows the GitHub interface for the 'Frontend Developers' team under the organization 'chaconcorp'. The team has 2 members and 2 repositories. It grants Admin access. Members listed are Tony Chacon and Scott Chacon, both with 'Remove' buttons. A 'Members' tab is selected, and a 'Repositories' tab is also present. A search bar for inviting users is at the top right. The URL is @acmecorp/frontend.

FIGURE 6-47

团队页面

Figure 6-47

" Set

@mentions @acmecorp/

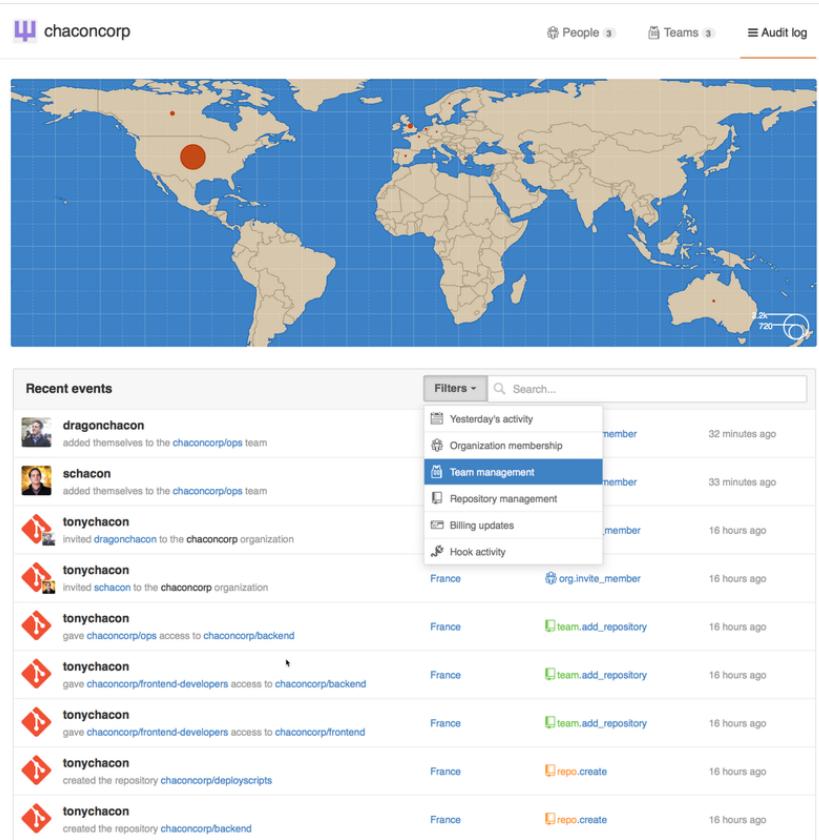
frontend

ux, css      refactoring  
legal      colorblind

Audit Log

**FIGURE 6-48**

审计日志



## 脚本 GitHub

GitHub

 GitHub  
 GitHub  
 GitHub

API

 Hack  
 GitHub

GitHub

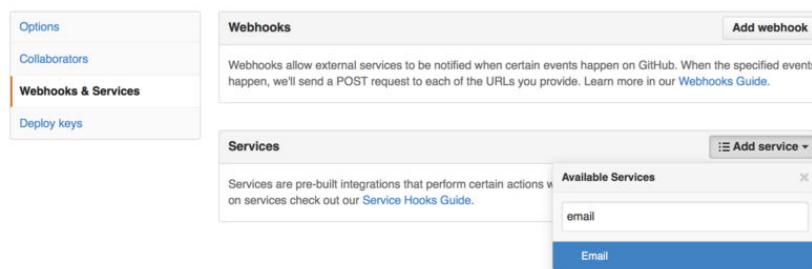
GitHub

服务

hooks and Services"

Figure 6-49

" Web

**FIGURE 6-49**

服务与钩子配置区域

" Add Service"

" email "

BUG

Figure 6-50

**FIGURE 6-50**

电子邮件服务配置

The screenshot shows the 'Services / Add Email' configuration page. On the left, a sidebar lists 'Options', 'Collaborators', 'Webhooks & Services' (which is selected and highlighted in orange), and 'Deploy keys'. The main area is titled 'Install Notes' with three items: 1. address whitespace separated email addresses (at most two), 2. secret fills out the Approved header to automatically approve the message in a read-only or moderated mailing list, and 3. send\_from\_author uses the commit author email address in the From address of the email. Below this is a 'Address' field containing 'tchacon@example.com', a 'Secret' field (empty), and a checked 'Send from author' checkbox. A 'Active' checkbox is checked with the note 'We will run this service when an event is triggered.' At the bottom is a green 'Add service' button.

“Add service”

GitHub  
Jenkins  
Jenkins

钩子

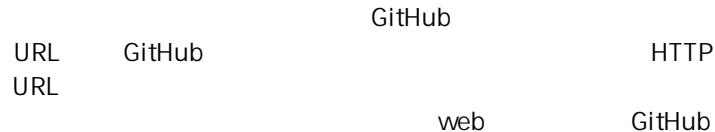


Figure 6-49 “Add webhook”  
Figure 6-51

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation.

**Payload URL \***

**Content type**

**Secret**

**Which events would you like to trigger this webhook?**

Just the push event.

Send me **everything**.

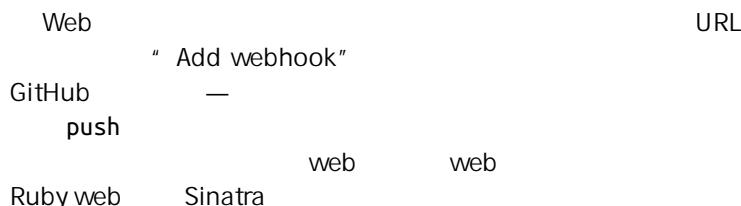
Let me select individual events.

Active  
We will deliver event details when this hook is triggered.

**Add webhook**

**FIGURE 6-51**

Web 钩子配置



```

require 'sinatra'
require 'json'
require 'mail'

post '/payload' do
  push = JSON.parse(request.body.read) # parse the JSON

  # gather the data we're looking for
  pusher = push["pusher"]["name"]
  branch = push["ref"]

  # get a list of all the files touched
  files = push["commits"].map do |commit|
    commit['added'] + commit['modified'] + commit['removed']
  end

```

```
end
files = files.flatten.uniq

# check for our criteria
if pusher == 'schacon' &&
  branch == 'ref/heads/special-branch' &&
  files.include?('special-file.txt')

  Mail.deliver do
    from      'tchacon@example.com'
    to        'tchacon@example.com'
    subject   'Scott Changed the File'
    body      "ALARM"
  end
end
end
```

GitHub                  JSON

GitHub                  webhook

**Recent Deliveries**

|                                      |                                      |                     |     |
|--------------------------------------|--------------------------------------|---------------------|-----|
| <span style="color: red;">⚠</span>   | 4aeae280-4e38-11e4-9bac-c130e992644b | 2014-10-07 17:40:41 | ... |
| <span style="color: green;">✓</span> | aff20880-4e37-11e4-9089-35319435e08b | 2014-10-07 17:36:21 | ... |
| <span style="color: green;">✓</span> | 90f37680-4e37-11e4-9508-227d13b2ccfc | 2014-10-07 17:35:29 | ... |

Request    Response 200    Completed in 0.61 seconds. Redeliver

**Headers**

```
Request URL: https://hooks.example.com/payload
Request method: POST
content-type: application/json
Expect:
User-Agent: GitHub-Hookshot/64a1910
X-GitHub-Delivery: 90f37680-4e37-11e4-9508-227d13b2ccfc
X-GitHub-Event: push
```

**Payload**

```
{
  "ref": "refs/heads/remove-whitespace",
  "before": "99d4fe5bffff827f8a9e7cde00cbb0ab06a35e48",
  "after": "9370a6c3349331bac7e4c3c78c10bc8460c1e3e8",
  "created": false,
  "deleted": false,
  "forced": false,
  "base_ref": null,
  "compare": "https://github.com/tonychacon/fade/compare/99d4fe5bffff..9370a6c33493",
  "commits": [
    {
      "id": "9370a6c3349331bac7e4c3c78c10bc8460c1e3e8",
      "distinct": true,
      "message": "remove whitespace",
      "timestamp": "2014-10-07T17:35:22+02:00",
      "url": "https://github.com/tonychacon/fade/commit/9370a6c3349331bac7e4c3c78c10bc8460c1e3e8"
    }
  ]
}
```

**FIGURE 6-52**

Web 钩子调试信息

web  
<https://developer.github.com/webhooks/> GitHub

## GitHub API

GitHub API  
 API

GitHub

API  
Pull Request

GET

" schacon"

```
$ curl https://api.github.com/users/schacon
{
  "login": "schacon",
  "id": 70,
  "avatar_url": "https://avatars.githubusercontent.com/u/70",
# ...
  "name": "Scott Chacon",
  "company": "GitHub",
  "following": 19,
  "created_at": "2008-01-27T17:19:28Z",
  "updated_at": "2014-06-10T02:37:23Z"
}
```

---

GitHub API  
Markdown .gitignore

```
$ curl https://api.github.com/gitignore/templates/Java
{
  "name": "Java",
  "source": "*.class

# Mobile Tools for Java (J2ME)
.mtj.tmp/

# Package Files #
*.jar
*.war
*.ear

# virtual machine crash logs, see http://www.java.com/en/download/help/error_hotspot
hs_err_pid*
"
}
```

**FIGURE 6-53**

从设置页的  
“Applications” 标签  
生成访问令牌。

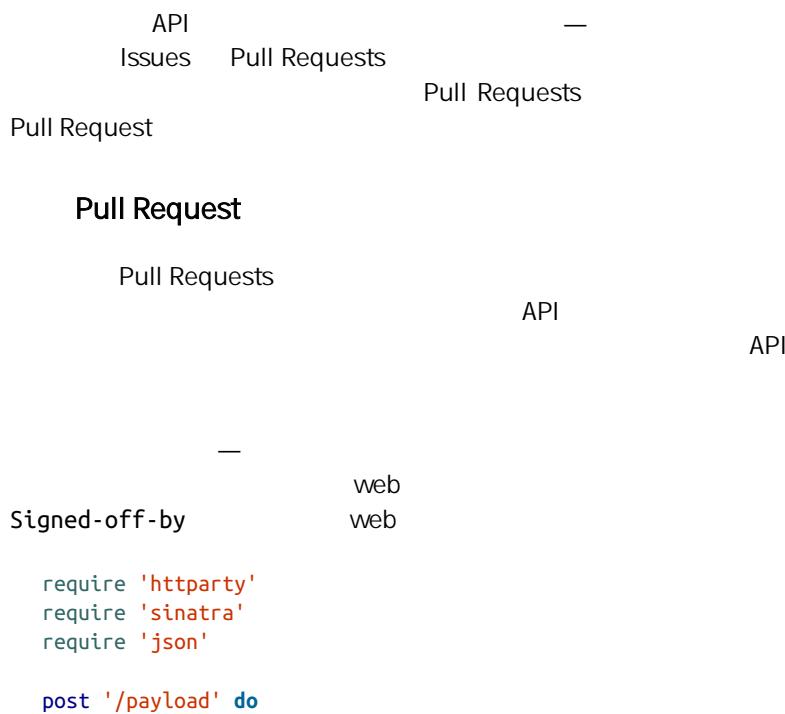
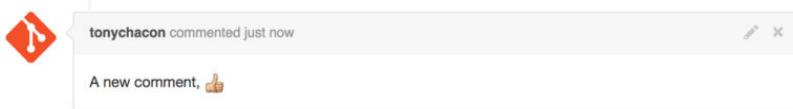


```
"id": 58322100,  
"html_url": "https://github.com/schacon/blink/issues/6#issuecomment-58322100",  
...  
"user": {  
    "login": "tonychacon",  
    "id": 7874698,  
    "avatar_url": "https://avatars.githubusercontent.com/u/7874698?v=2",  
    "type": "User",  
},  
"created_at": "2014-10-08T07:48:19Z",  
"updated_at": "2014-10-08T07:48:19Z",  
"body": "A new comment, :+1:"  
}
```

Figure 6-54

**FIGURE 6-54**

从 GitHub API 发布的一条评论



```

push = JSON.parse(request.body.read) # parse the JSON
repo_name = push['repository']['full_name']

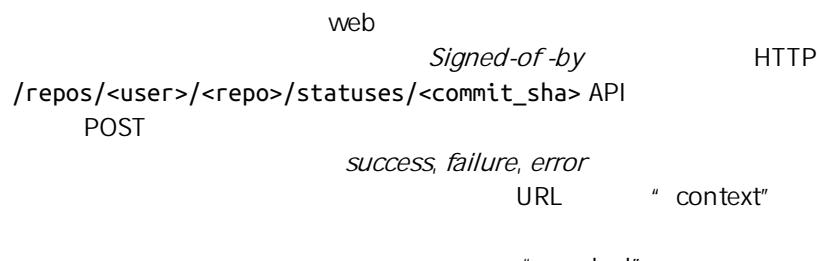
# look through each commit message
push["commits"].each do |commit|

  # look for a Signed-off-by string
  if /Signed-off-by/.match commit['message']
    state = 'success'
    description = 'Successfully signed off!'
  else
    state = 'failure'
    description = 'No signoff found.'
  end

  # post status to GitHub
  sha = commit["id"]
  status_url = "https://api.github.com/repos/#{repo_name}/statuses/#{sha}"

  status = {
    "state"      => state,
    "description" => description,
    "target_url"  => "http://example.com/how-to-signoff",
    "context"     => "validate/signoff"
  }
  HTTParty.post(status_url,
    :body => status.to_json,
    :headers => {
      'Content-Type'  => 'application/json',
      'User-Agent'    => 'tonychacon/signoff',
      'Authorization' => "token #{ENV['TOKEN']}" })
end
end

```



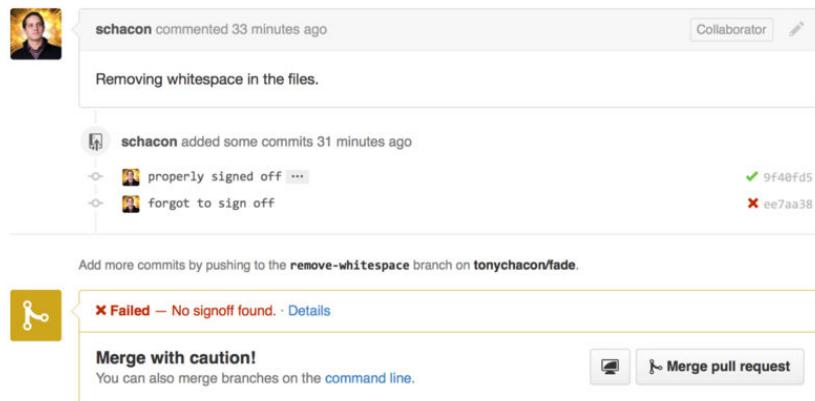
GitHub

Pull Request

Figure 6-55

**FIGURE 6-55**

通过 API 的提交状态



“ Signed-of -by”

Pull Request  
API

Octokit

|      |             |      |      |  |
|------|-------------|------|------|--|
| curl | HTTP        |      |      |  |
| API  |             |      |      |  |
| Go   | Objective-C | Ruby | .NET | <a href="http://github.com/octokit">http://<br/>github.com/octokit</a> |

|        |  |
|--------|--|
| GitHub | <a href="https://developer.github.com">https://<br/>developer.github.com</a> |
| API    |  |

总结

GitHub

Git

# Git 工具 7

Git

Git

## 选择修订版本

Git

Git      SHA-1

**SHA-1**

```
Git                    SHA-1  
      SHA-1            4            —  
                      SHA-1  
                      git log  
                      :  
                      
```

```
$ git log  
commit 734713bc047d87bf7eac9674765ae793478c50d3  
Author: Scott Chacon <schacon@gmail.com>  
Date:   Fri Jan 2 18:32:33 2009 -0800  
  
fixed refs handling, added gc auto, updated tests
```

```

commit d921970aadf03b3cf0e71becdaab3147ba71cdef
Merge: 1c002dd... 35cfb2b...
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Dec 11 15:08:43 2008 -0800

    Merge commit 'phedders/rdocs'

commit 1c002dd4b536e7479fe34593e72e6c6c1819e53b
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Dec 11 14:58:32 2008 -0800

    added some blame and merge stuff

```

1c002dd.... git show

```

$ git show 1c002dd4b536e7479fe34593e72e6c6c1819e53b
$ git show 1c002dd4b536e7479f
$ git show 1c002d

```

|                 |       |         |
|-----------------|-------|---------|
| Git             | SHA-1 | git log |
| --abbrev-commit |       |         |
|                 |       | SHA-1   |

```

$ git log --abbrev-commit --pretty=oneline
ca82a6d changed the version number
085bb3b removed unnecessary test code
a11bef0 first commit

```

|       |     |       |
|-------|-----|-------|
| 8 10  | Git | SHA-1 |
| Linux |     | 45    |
| 360   | 11  |       |

---

## 关于 **SHA-1** 的简短说明

许多人觉得他们的仓库里有可能出现两个 SHA-1 值相同的对象。然后呢？

如果你真的向仓库里提交了一个跟之前的某个对象具有相同 SHA-1 值的对象，Git 发现仓库里已经存在了拥有相同 HASH 值的对象，就会认为这个新的提交是已经被写入仓库的。如果之后你想检出那个对象时，你将得到先前那个对象的数据。

但是这种情况发生的概率十分渺小。SHA-1 摘要长度是 20 字节，也就是 160 位。 $2^{80}$  个随机哈希对象才有 50% 的概率出现一次冲突（计算冲突机率的公式是  $p = (n(n-1)/2) * (1/2^{160})$ ）。 $2^{80}$  是  $1.2 \times 10^{24}$  也就是一亿亿亿。那是地球上沙粒总数的 1200 倍。

举例说一下怎样才能产生一次 SHA-1 冲突。如果地球上 65 亿个人类都在编程，每人每秒都在产生等价于整个 Linux 内核历史（360 万个 Git 对象）的代码，并将之提交到一个巨大的 Git 仓库里面，这样持续两年的时间才会产生足够的对象，使其拥有 50% 的概率产生一次 SHA-1 对象冲突。这要比你编程团队的成员同一天晚上在互不相干的意外中被狼袭击并杀死的机率还要小。

---

|         |        |
|---------|--------|
| Git     | SHA-1  |
| ca82a6d | topic1 |
| ⋮       |        |

```
$ git show ca82a6dff817ec66f44342007202690a93763949  
$ git show topic1
```

|            |           |           |
|------------|-----------|-----------|
| SHA-1      | rev-parse | Git       |
| Chapter 10 |           | rev-parse |
|            |           | Git       |
|            |           | rev-parse |

```
$ git rev-parse topic1  
ca82a6dff817ec66f44342007202690a93763949
```

|            |          |
|------------|----------|
| Git        | (reflog) |
| HEAD       |          |
| git reflog |          |

```
$ git reflog
734713b HEAD@{0}: commit: fixed refs handling, added gc auto, updated
d921970 HEAD@{1}: merge phedders/rdocs: Merge made by recursive.
1c002dd HEAD@{2}: commit: added some blame and merge stuff
1c36188 HEAD@{3}: rebase -i (squash): updating HEAD
95df984 HEAD@{4}: commit: # This is a combination of two commits.
1c36188 HEAD@{5}: rebase -i (squash): updating HEAD
7e05da5 HEAD@{6}: rebase -i (pick): updating HEAD
```

HEAD  
HEAD  
HEAD  
@{n}      reflog

```
$ git show HEAD@{5}
```

master

```
$ git show master@{yesterday}
```

git log -g                  git log

```
$ git log -g master
commit 734713bc047d87bf7eac9674765ae793478c50d3
Reflog: master@{0} (Scott Chacon <schacon@gmail.com>)
Reflog message: commit: fixed refs handling, added gc auto, updated
Author: Scott Chacon <schacon@gmail.com>
Date:   Fri Jan 2 18:32:33 2009 -0800

        fixed refs handling, added gc auto, updated tests

commit d921970aadf03b3cf0e71becdaab3147ba71cdef
Reflog: master@{1} (Scott Chacon <schacon@gmail.com>)
Reflog message: merge phedders/rdocs: Merge made by recursive.
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Dec 11 15:08:43 2008 -0800

        Merge commit 'phedders/rdocs'
```

```
git show HEAD@{2.months.ago}
```

```
—
```

```
^
```

Git

```
$ git log --pretty=format:'%h %s' --graph
* 734713b fixed refs handling, added gc auto, updated tests
*   d921970 Merge commit 'phedders/rdocs'
|\ 
| * 35cfb2b Some rdoc changes
* | 1c002dd added some blame and merge stuff
|/
* 1c36188 ignore *.gem
* 9b29157 add open3_detach to gemspec file list
```

HEAD<sup>^</sup>

" HEAD

"

```
$ git show HEAD^
commit d921970aadf03b3cf0e71becdaab3147ba71cdef
Merge: 1c002dd... 35cfb2b...
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Dec 11 15:08:43 2008 -0800

Merge commit 'phedders/rdocs'
```

```
^
—
"
```

d921970^2 " d921970
(merge)

```
$ git show d921970^
commit 1c002dd4b536e7479fe34593e72e6c6c1819e53b
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Dec 11 14:58:32 2008 -0800

added some blame and merge stuff
```

```
$ git show d921970^2
commit 35cfb2b795a55793d7cc56a6cc2060b4bb732548
Author: Paul Hedderly <paul+git@mjr.org>
Date:   Wed Dec 10 22:22:03 2008 +0000
```

Some rdoc changes

HEAD~  
HEAD~2  
~  
HEAD^  
"  
"  
"  
— Git

HEAD~3

```
$ git show HEAD~3
commit 1c3618887afb5fbcbea25b7c013f4e2114448b8d
Author: Tom Preston-Werner <tom@mojombo.com>
Date:   Fri Nov 7 13:47:59 2008 -0500
```

ignore \*.gem

HEAD^^^

```
$ git show HEAD^^^
commit 1c3618887afb5fbcbea25b7c013f4e2114448b8d
Author: Tom Preston-Werner <tom@mojombo.com>
Date:   Fri Nov 7 13:47:59 2008 -0500
```

ignore \*.gem

—  
HEAD~3^2

"

"

Figure 7-1

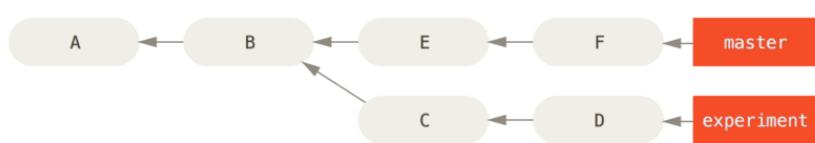


FIGURE 7-1

Example history for range selection.

experiment  
master..experiment Git master  
" experiment master " "

```
$ git log master..experiment  
D  
C
```

master experiment  
experiment..master master  
experiment

```
$ git log experiment..master  
F  
E
```

experiment

```
$ git log origin/master..HEAD
```

origin  
origin/master git log  
git push origin/master..HEAD  
origin/master..HEAD Git HEAD git log origin/

master.. — Git HEAD

多点

Git ^ --not  
3

```
$ git log refA..refB
$ git log ^refA refB
$ git log refB --not refA
```

refA refB  
refC

```
$ git log refA refB ^refC
$ git log refA refB --not refC
```

三点

master experiment

```
$ git log master...experiment
F
E
D
C
```

log log 4  
--left-right

```
$ git log --left-right master...experiment
< F
< E
> D
> C
```

Git

## 交互式暂存

Git

```
git add -i  
--interactive Git
```

```
$ git add -i
      staged      unstaged path
 1: unchanged      +0/-1 TODO
 2: unchanged      +1/-1 index.html
 3: unchanged      +5/-1 lib/simplegit.rb

*** Commands ***
 1: status       2: update       3: revert       4: add untracked
 5: patch        6: diff         7: quit         8: help
What now>
```

```
-          git sta-
tus
```

What now> 2 u

```
What now> 2
      staged      unstaged path
1:   unchanged      +0/-1 TODO
2:   unchanged      +1/-1 index.html
3:   unchanged      +5/-1 lib/simplegit.rb
Update>>
```

TODO index.html

```
Update>> 1,2
      staged      unstaged path
* 1:   unchanged      +0/-1 TODO
* 2:   unchanged      +1/-1 index.html
3:   unchanged      +5/-1 lib/simplegit.rb
Update>>
```

\*                          Update>>  
                            Git

```
Update>>
updated 2 paths

*** Commands ***
1: status      2: update      3: revert      4: add untracked
5: patch       6: diff        7: quit        8: help
What now> 1
      staged      unstaged path
1:      +0/-1      nothing TODO
2:      +1/-1      nothing index.html
3:   unchanged      +5/-1 lib/simplegit.rb
```

TODO index.html                          simplegit.rb  
  TODO                          3     r

```
*** Commands ***
1: status      2: update      3: revert      4: add untracked
5: patch       6: diff        7: quit        8: help
What now> 3
      staged      unstaged path
1:      +0/-1      nothing TODO
2:      +1/-1      nothing index.html
3:   unchanged      +5/-1 lib/simplegit.rb
```

```
Revert>> 1
      staged      unstaged path
* 1:      +0/-1      nothing TODO
  2:      +1/-1      nothing index.html
  3:  unchanged      +5/-1 lib/simplegit.rb
Revert>> [enter]
reverted one path
```

Git

TODO

```
*** Commands ***
1: status      2: update      3: revert      4: add untracked
5: patch      6: diff       7: quit       8: help
What now> 1
      staged      unstaged path
1:  unchanged      +0/-1 TODO
2:      +1/-1      nothing index.html
3:  unchanged      +5/-1 lib/simplegit.rb
```

6 d

**git diff --cached**

```
*** Commands ***
1: status      2: update      3: revert      4: add untracked
5: patch      6: diff       7: quit       8: help
What now> 6
      staged      unstaged path
1:      +1/-1      nothing index.html
Review diff>> 1
diff --git a/index.html b/index.html
index 4d07108..4335f49 100644
--- a/index.html
+++ b/index.html
@@ -16,7 +16,7 @@ Date Finder

<p id="out">...</p>

-<div id="footer">contact : support@github.com</div>
+<div id="footer">contact : email.support@github.com</div>

<script type="text/javascript">
```

Git

simplegit.rb  
Git

5 p

Git

```
diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index dd5ecc4..57399e0 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -22,7 +22,7 @@ class SimpleGit
end

def log(treeish = 'master')
-  command("git log -n 25 #{treeish}")
+  command("git log -n 30 #{treeish}")
end

def blame(path)
Stage this hunk [y,n,a,d/,j,J,g,e,?]?
```

?

```
Stage this hunk [y,n,a,d/,j,J,g,e,?]?
y - stage this hunk
n - do not stage this hunk
a - stage this and all the remaining hunks in the file
d - do not stage this hunk nor any of the remaining hunks in the file
g - select a hunk to go to
/ - search for a hunk matching the given regex
j - leave this hunk undecided, see next undecided hunk
J - leave this hunk undecided, see next hunk
k - leave this hunk undecided, see previous undecided hunk
K - leave this hunk undecided, see previous hunk
s - split the current hunk into smaller hunks
e - manually edit the current hunk
? - print help
```

y n

```
What now> 1
      staged      unstaged path
1:   unchanged          +0/-1 TODO
2:           +1/-1      nothing index.html
3:           +1/-1      +4/-0 lib/simplegit.rb
```

simplegit.rb

git commit

```
git add -p  git add --patch
          reset --patch
checkout --patch                      stash save --patch
```

## 储藏与清理

git stash

git status

```
$ git status
Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

modified:   index.html

Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git checkout -- <file>..." to discard changes in working directory)

modified:   lib/simplegit.rb
```

**git stash git stash save**

```
$ git stash
Saved working directory and index state \
    "WIP on master: 049d078 added the index file"
HEAD is now at 049d078 added the index file
(To restore them type "git stash apply")
```

```
$ git status
# On branch master
nothing to commit, working directory clean
```

**git stash list**

```
$ git stash list
stash@{0}: WIP on master: 049d078 added the index file
stash@{1}: WIP on master: c264051 Revert "added file_size"
stash@{2}: WIP on master: 21d80a5 added number to log
```

stash  
**git stash apply**  
**git stash apply stash@{2}**

Git

```
$ git stash apply
# On branch master
# Changed but not updated:
#   (use "git add <file>..." to update what will be committed)
#
#       modified:   index.html
#       modified:   lib/simplegit.rb
#
```

Git

## Git

--index git stash apply

```
$ git stash apply --index
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   index.html
#
# Changed but not updated:
#   (use "git add <file>..." to update what will be committed)
#
#       modified:   lib/simplegit.rb
#
```

- git  
stash drop

```
$ git stash list
stash@{0}: WIP on master: 049d078 added the index file
stash@{1}: WIP on master: c264051 Revert "added file_size"
stash@{2}: WIP on master: 21d80a5 added number to log
$ git stash drop stash@{0}
Dropped stash@{0} (364e91f3f268f0900bc3ee613f9f733e82aaed43)
```

git stash pop

--keep-index Git stash save  
git add

```
$ git status -s
M  index.html
```

```
M lib/simplegit.rb

$ git stash --keep-index
Saved working directory and index state WIP on master: 1b65b17 added the index file
HEAD is now at 1b65b17 added the index file
```

```
$ git status -s
M index.html
```

--  
include-untracked -u Git

```
$ git status -s
M index.html
M lib/simplegit.rb
?? new-file.txt
```

```
$ git stash -u
Saved working directory and index state WIP on master: 1b65b17 added the index file
HEAD is now at 1b65b17 added the index file
```

```
$ git status -s
$
```

--patch Git

```
$ git stash --patch
diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index 66d332e..8bb5674 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -16,6 +16,10 @@ class SimpleGit
      return `#{git_cmd} 2>&1`.chomp
    end
  end
+
+  def show(treeish = 'master')
+    command("git show #{treeish}")
+  end
end
test
Stash this hunk [y,n,q,a,d,/ ,e,?]?
```

```
Saved working directory and index state WIP on master: 1b65b17 added the index file
```

### git stash branch

```
$ git stash branch testchanges
Switched to a new branch "testchanges"
# On branch testchanges
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   index.html
#
# Changed but not updated:
#   (use "git add <file>..." to update what will be committed)
#
#       modified:   lib/simplegit.rb
#
Dropped refs/stash@{0} (f0dfc4d5dc332d1cee34a634182e168c4efc3359)
```

```
git
clean
```

### git stash --all

```
git clean
clean -f -d
f          "          "
```

-n  
" " "

```
$ git clean -d -n
Would remove test.o
Would remove tmp/
```

git clean  
.gitignore

.o clean -x

```
$ git status -s
M lib/simplegit.rb
?? build.TMP
?? tmp/
```

```
$ git clean -n -d
Would remove build.TMP
Would remove tmp/
```

```
$ git clean -n -d -x
Would remove build.TMP
Would remove test.o
Would remove tmp/
```

git clean -n -f -i  
" interactive"  
clean

```
$ git clean -x -i
Would remove the following items:
  build.TMP  test.o
*** Commands ***
  1: clean           2: filter by pattern   3: select by numbers  4: ask
What now>
```

## 签署工作

Git

Git

GPG

GPG

GPG

```
$ gpg --list-keys  
/Users/schacon/.gnupg/pubring.gpg  
-----  
pub 2048R/0A46826A 2014-06-04  
uid           Scott Chacon (Git signing key) <schacon@gmail.com>  
sub 2048R/874529A9 2014-06-04
```

gpg --gen-key

gpg --gen-key

Git user.signingkey

```
git config --global user.signingkey 0A46826A
```

Git

GPG

-s -a

```
$ git tag -s v1.5 -m 'my signed 1.5 tag'
```

You need a passphrase to unlock the secret key for  
user: "Ben Straub <ben@straub.cc>"  
2048-bit RSA key, ID 800430EB, created 2014-05-04

git show

GPG

```
$ git show v1.5
tag v1.5
Tagger: Ben Straub <ben@straub.cc>
Date:   Sat May 3 20:29:41 2014 -0700

my signed 1.5 tag
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1

iQEcBAABAgAGBQJTZbQlAAoJEF0+sviABDDrZbQH/09PfE51KPVPlanr6q1v4/Ut
LQxfojUWiLQdg2ESJItkcuweYg+kC3HCyFejeDIBw9dpXt00rY26p05qrpnG+85b
hM1/PswpPLuBSr+oCIDj5GMC2r2iEKsfv2fJbNW8iWAXVLoWZRF8B0MfqX/YTMbm
ecorc4iXzQu7tupRihslbNkfvcimnSDeSzCpWAhl7h8Wj6hhqePmLm9lAYqnKp
8S5B/1SSQuEAjRZgI4IexpZoeKGVDptPHxLLS38fozsyi0QyDyzEgJxcJQVMXxVi
RUysgqjcpT8+iQM1PblGfHR4XAh0qN5Fx06PSaFZhqvWFezJ28/CLyX5q+oIVk=
=EFTF
-----END PGP SIGNATURE-----

commit ca82a6dff817ec66f44342007202690a93763949
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

changed the version number
```

git tag -v [tag-name]

GPG

```
$ 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
```

```
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'
```

Git v1.7.9

-S

```
git commit
```

```
$ git commit -a -S -m 'signed commit'
```

```
You need a passphrase to unlock the secret key for  
user: "Scott Chacon (Git signing key) <schacon@gmail.com>"  
2048-bit RSA key, ID 0A46826A, created 2014-06-04
```

```
[master 5c3386c] signed commit  
 4 files changed, 4 insertions(+), 24 deletions(-)  
 rewrite Rakefile (100%)  
 create mode 100644 lib/git.rb
```

```
git log --show-signature
```

```
$ git log --show-signature -1  
commit 5c3386cf54bba0a33a32da706aa52bc0155503c2  
gpg: Signature made Wed Jun 4 19:49:17 2014 PDT using RSA key ID 0A46826A  
gpg: Good signature from "Scott Chacon (Git signing key) <schacon@gmail.com>"  
Author: Scott Chacon <schacon@gmail.com>  
Date:   Wed Jun 4 19:49:17 2014 -0700  
  
        signed commit
```

git log

%G?

```
$ git log --pretty=format:%h %G? %aN %s"
```

```
5c3386c G Scott Chacon signed commit  
ca82a6d N Scott Chacon changed the version number
```

```
085bb3b N Scott Chacon removed unnecessary test code
a11bef0 N Scott Chacon first commit
```

Git 1.8.3                    “git merge”     “git pull”  
**verify-signatures**    GPG

```
$ git merge --verify-signatures non-verify
fatal: Commit ab06180 does not have a GPG signature.
```

```
$ git merge --verify-signatures signed-branch
Commit 13ad65e has a good GPG signature by Scott Chacon (Git signing key) <schacon...
Updating 5c3386c..13ad65e
Fast-forward
 README | 2 ++
 1 file changed, 2 insertions(+)
```

**git merge**                    -S

```
$ git merge --verify-signatures -S signed-branch
Commit 13ad65e has a good GPG signature by Scott Chacon (Git signing key) <schacon...
You need a passphrase to unlock the secret key for
user: "Scott Chacon (Git signing key) <schacon@gmail.com>"
2048-bit RSA key, ID 0A46826A, created 2014-06-04

Merge made by the 'recursive' strategy.
 README | 2 ++
 1 file changed, 2 insertions(+)
```

GPG

## 搜索

Git

### Git Grep

Git            grep

Git

Git            -n

Git

```
$ git grep -n gmtime_r
compat/gmtime.c:3:#undef gmtime_r
compat/gmtime.c:8:    return git_gmtime_r(timep, &result);
compat/gmtime.c:11:struct tm *git_gmtime_r(const time_t *timep, struct tm *result)
compat/gmtime.c:16:    ret = gmtime_r(timep, result);
compat/mingw.c:606:struct tm *gmtime_r(const time_t *timep, struct tm *result)
compat/mingw.h:162:struct tm *gmtime_r(const time_t *timep, struct tm *result);
date.c:429:        if (gmtime_r(&now, &now_tm))
date.c:492:        if (gmtime_r(&time, tm)) {
git-compat-util.h:721:struct tm *git_gmtime_r(const time_t *, struct tm *);
git-compat-util.h:723:#define gmtime_r git_gmtime_r
```

grep

--count

Git

```
$ git grep --count gmtime_r
compat/gmtime.c:4
compat/mingw.c:1
compat/mingw.h:1
date.c:2
git-compat-util.h:2
```

-p

```
$ git grep -p gmtime_r *.c
date.c=static int match_multi_number(unsigned long num, char c, const char *date,
date.c:           if (gmtime_r(&now, &now_tm))
date.c=static int match_digit(const char *date, struct tm *tm, int *offset, int *t
date.c:           if (gmtime_r(&time, tm)) {
```

|             |                   |                    |
|-------------|-------------------|--------------------|
| match_digit | date.c<br>--and   | match_multi_number |
|             | gmtime_r          |                    |
|             | "LINK" "BUF_MAX"  |                    |
|             | --break --heading |                    |

1.80 Git

```
$ git grep --break --heading \
-n -e '#define' --and \(
-e LINK -e BUF_MAX \) v1.8.0
v1.8.0:builtin/index-pack.c
62:#define FLAG_LINK (1u<<20)

v1.8.0:cache.h
73:#define S_IFGITLINK 0160000
74:#define S_ISGITLINK(m) (((m) & S_IFMT) == S_IFGITLINK)

v1.8.0:environment.c
54:#define OBJECT_CREATION_MODE OBJECT_CREATIONUSES_HARDLINKS

v1.8.0:strbuf.c
326:#define STRBUF_MAXLINK (2*PATH_MAX)

v1.8.0:symlinks.c
53:#define FL_SYMLINK (1 << 2)

v1.8.0:zlib.c
30:/* #define ZLIB_BUF_MAX ((uInt)-1) */
31:#define ZLIB_BUF_MAX ((uInt) 1024 * 1024 * 1024) /* 1GB */
```

grep ack git grep

Git

Git

## Git

```
git log
```

```
dif
```

```
ZLIB_BUF_MAX  
-S
```

```
$ git log -SZLIB_BUF_MAX --oneline  
e01503b zlib: allow feeding more than 4GB in one go  
ef49a7a zlib: zlib can only process 4GB at a time
```

```
dif          ef49a7a  
e01503b  
-G
```

## 行日志搜索

```
git log      -L  
            zlib.c      git_deflate_bound  
git log -L :git_deflate_bound:zlib.c  Git
```

```
$ git log -L :git_deflate_bound:zlib.c  
commit ef49a7a0126d64359c974b4b3b71d7ad42ee3bc  
Author: Junio C Hamano <gitster@pobox.com>  
Date:   Fri Jun 10 11:52:15 2011 -0700  
  
        zlib: zlib can only process 4GB at a time  
  
diff --git a/zlib.c b/zlib.c  
--- a/zlib.c  
+++ b/zlib.c  
@@ -85,5 +130,5 @@  
-unsigned long git_deflate_bound(z_streamp strm, unsigned long size)  
+unsigned long git_deflate_bound(git_zstream *strm, unsigned long size)  
{  
-    return deflateBound(strm, size);  
+    return deflateBound(&strm->z, size);  
}
```

```
commit 225a6f1068f71723a910e8565db4e252b3ca21fa
Author: Junio C Hamano <gitster@pobox.com>
Date:   Fri Jun 10 11:18:17 2011 -0700

    zlib: wrap deflateBound() too

diff --git a/zlib.c b/zlib.c
--- a/zlib.c
+++ b/zlib.c
@@ -81,0 +85,5 @@
+unsigned long git_deflate_bound(z_streamp strm, unsigned long size)
+{
+    return deflateBound(strm, size);
+}
+
```

Git

```
git log -L '/  
unsigned long git_deflate_bound/','/^}/:zlib.c
```

## 重写历史

Git

stash

Git

```
$ git commit --amend
```

```
git add    git rm  
git commit --amend
```

SHA-1

-

Git

HEAD

```
git rebase -i
```

```
git rebase  
-i      HEAD~2^  HEAD~3      ~3
```

```
$ git rebase -i HEAD~3
```

- HEAD~3..HEAD

```
pick f7f3f6d changed my name a bit  
pick 310154e updated README formatting and added blame  
pick a5f4a0d added cat-file
```

```
# Rebase 710f0f8..a5f4a0d onto 710f0f8  
#  
# Commands:  
#   p, pick = use commit  
#   r, reword = use commit, but edit the commit message  
#   e, edit = use commit, but stop for amending
```

```
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out
```

log  
*log*

```
$ git log --pretty=format:"%h %s" HEAD~3..HEAD
a5f4a0d added cat-file
310154e updated README formatting and added blame
f7f3f6d changed my name a bit
```

HEAD~3

' pick'        ' edit'

```
edit f7f3f6d changed my name a bit
pick 310154e updated README formatting and added blame
pick a5f4a0d added cat-file
```

Git

```
$ git rebase -i HEAD~3
Stopped at f7f3f6d... changed my name a bit
You can amend the commit now, with

    git commit --amend
```

Once you're satisfied with your changes, run

```
git rebase --continue
```

```
$ git commit --amend
```

```
$ git rebase --continue
```

| pick | edit | edit |
|------|------|------|
| Git  |      |      |

" added cat-file"

```
pick f7f3f6d changed my name a bit
pick 310154e updated README formatting and added blame
pick a5f4a0d added cat-file
```

```
pick 310154e updated README formatting and added blame
pick f7f3f6d changed my name a bit
```

| 310154e | Git               |
|---------|-------------------|
|         | f7f3f6d           |
|         | " added cat-file" |

```

#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out

```

“ squash”        “ pick”        “ edit”        Git

```

pick f7f3f6d changed my name a bit
squash 310154e updated README formatting and added blame
squash a5f4a0d added cat-file

```

Git

```

# This is a combination of 3 commits.
# The first commit's message is:
changed my name a bit

# This is the 2nd commit message:

updated README formatting and added blame

# This is the 3rd commit message:

added cat-file

```

```
" updated README formatting"  
" added blame"           " updated README formatting and  
added blame"            rebase -i  
" edit"
```

```
pick f7f3f6d changed my name a bit  
edit 310154e updated README formatting and added blame  
pick a5f4a0d added cat-file
```

```
Git  
f7f3f6d  
310154e          git reset HEAD^  
  
git  
rebase --continue
```

```
$ git reset HEAD^  
$ git add README  
$ git commit -m 'updated README formatting'  
$ git add lib/simplegit.rb  
$ git commit -m 'added blame'  
$ git rebase --continue
```

```
Git          a5f4a0d
```

```
$ git log -4 --pretty=format:"%h %s"  
1c002dd added cat-file  
9b29157 added blame  
35cfb2b updated README formatting  
f3cc40e changed my name a bit
```

SHA-1

## filter-branch

filter-branch

从每一个提交移除一个文件

git add .

filter-branch  
--tree-filter filter-branch  
passwords.txt

```
$ git filter-branch --tree-filter 'rm -f passwords.txt' HEAD
Rewrite 6b9b3cf04e7c5686a9cb838c3f36a8cb6a0fc2bd (21/21)
Ref 'refs/heads/master' was rewritten
```

--tree-filter  
passwords.txt

HEAD  
Git  
git filter-branch --tree-filter 'rm -f \*~'

master  
filter-branch  
--all

使一个子目录做为新的根目录

trunk tags  
filter-branch  
trunk

```
$ git filter-branch --subdirectory-filter trunk HEAD
Rewrite 856f0bf61e41a27326cd8f09fe708d679f596f (12/12)
Ref 'refs/heads/master' was rewritten
```

trunk Git

全局修改邮箱地址

git config

filter-branch

--commit-filter

```
$ git filter-branch --commit-filter '
  if [ "$GIT_AUTHOR_EMAIL" = "schacon@localhost" ];
  then
    GIT_AUTHOR_NAME="Scott Chacon";
    GIT_AUTHOR_EMAIL="schacon@example.com";
    git commit-tree "$@";
  else
    git commit-tree "$@";
  fi' HEAD
```

SHA-1

SHA-1

## 重置揭密

reset checkout

Git

reset checkout Git

" "

"

"

Git

---



---

HEAD  
Index  
Working Directory

---

**HEAD**

HEAD

HEAD

HEAD

HEAD

SHA-1

```
$ git cat-file -p HEAD
tree cfda3bf379e4f8dba8717dee55aab78aef7f4daf
author Scott Chacon 1301511835 -0700
committer Scott Chacon 1301511835 -0700

initial commit

$ git ls-tree -r HEAD
100644 blob a906cb2a4a904a152... README
100644 blob 8f94139338f9404f2... Rakefile
040000 tree 99f1a6d12cb4b6f19... lib
```

**cat-file ls-tree****索引**

Git     "

"                **git commit**    Git

Git

**git commit**

```
$ git ls-files -s
100644 a906cb2a4a904a152e80877d4088654daad0c859 0 README
100644 8f94139338f9404f26296befa88755fc2598c289 0 Rakefile
100644 47c6340d6459e05787f644c2447d2595f5d3a54b 0 lib/simplegit.rb
```

```
ls -files
```

工作目录

```
.git
```

```
$ tree
.
├── README
└── Rakefile
└── lib
    └── simplegit.rb

1 directory, 3 files
```

Git

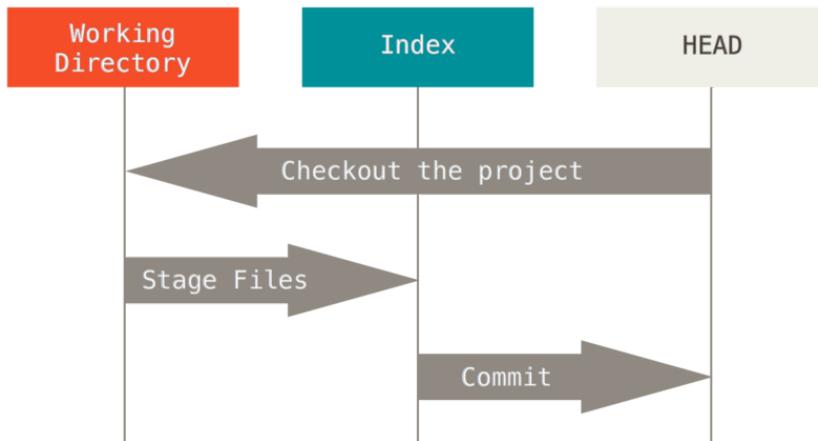
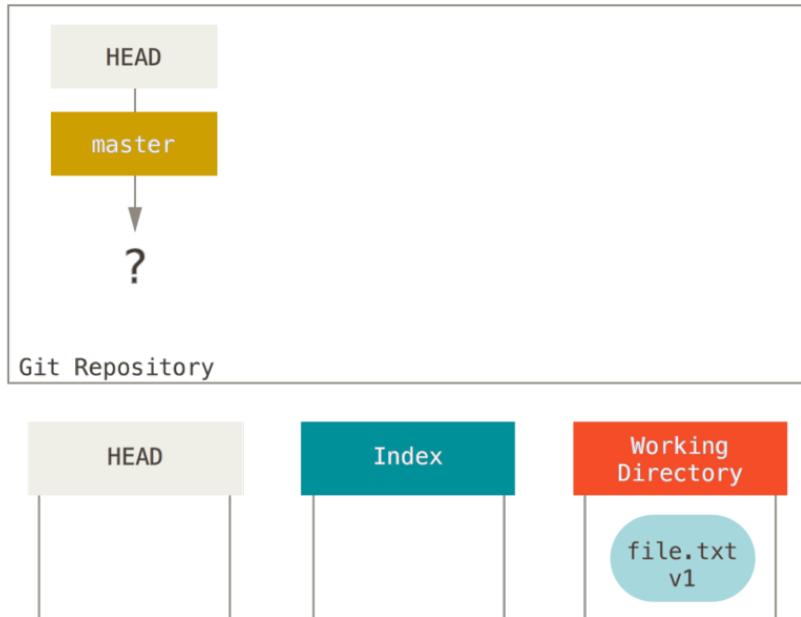


FIGURE 7-2

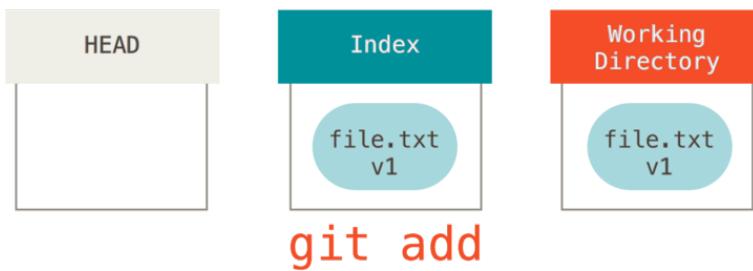
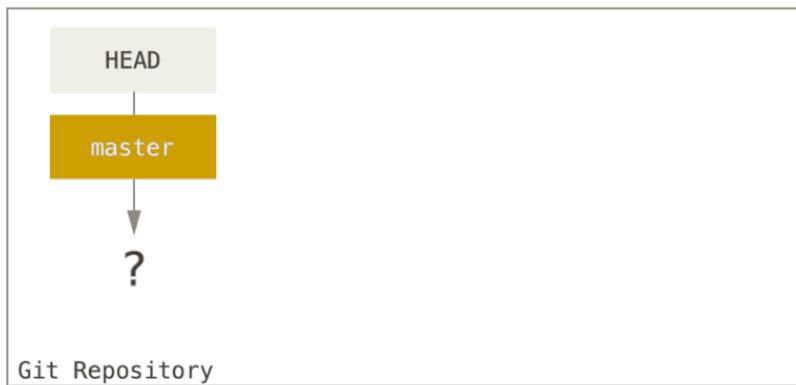
it              Git              v1              HEAD              git in-  
                                master

**FIGURE 7-3**



git add

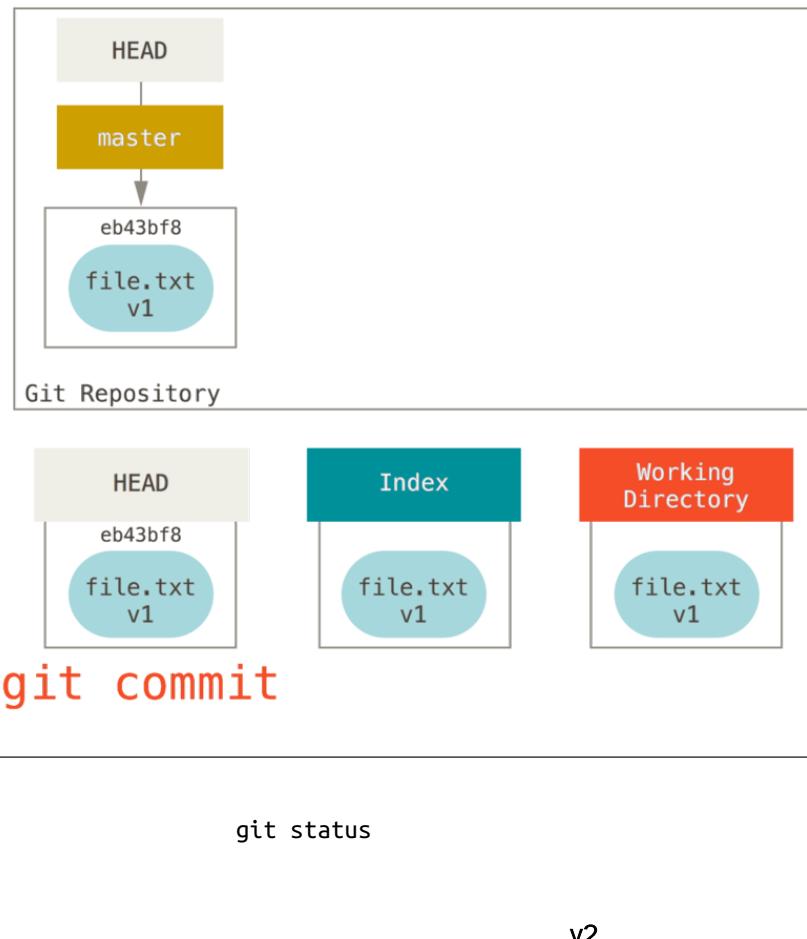
**FIGURE 7-4**



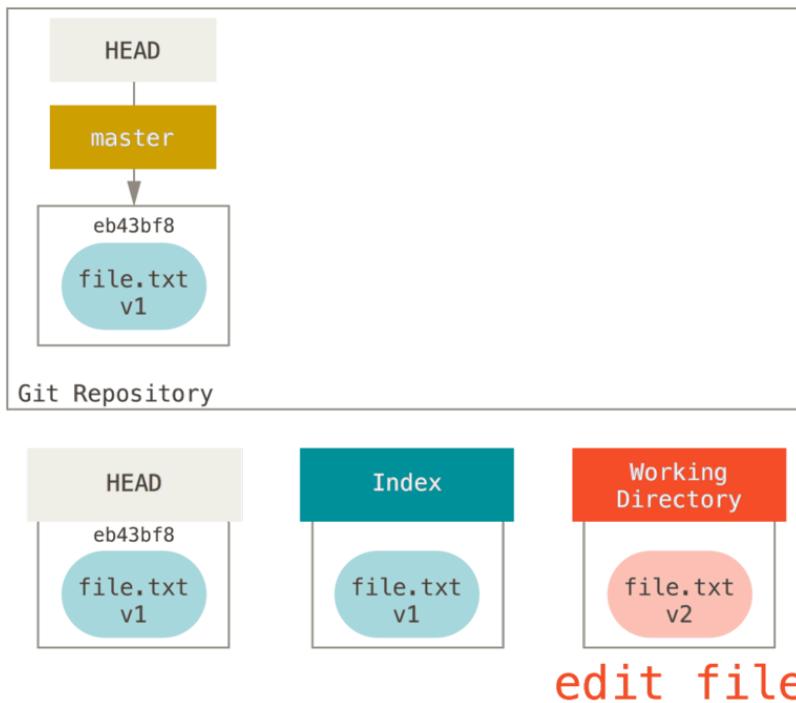
`git commit`

master

FIGURE 7-5



**FIGURE 7-6**

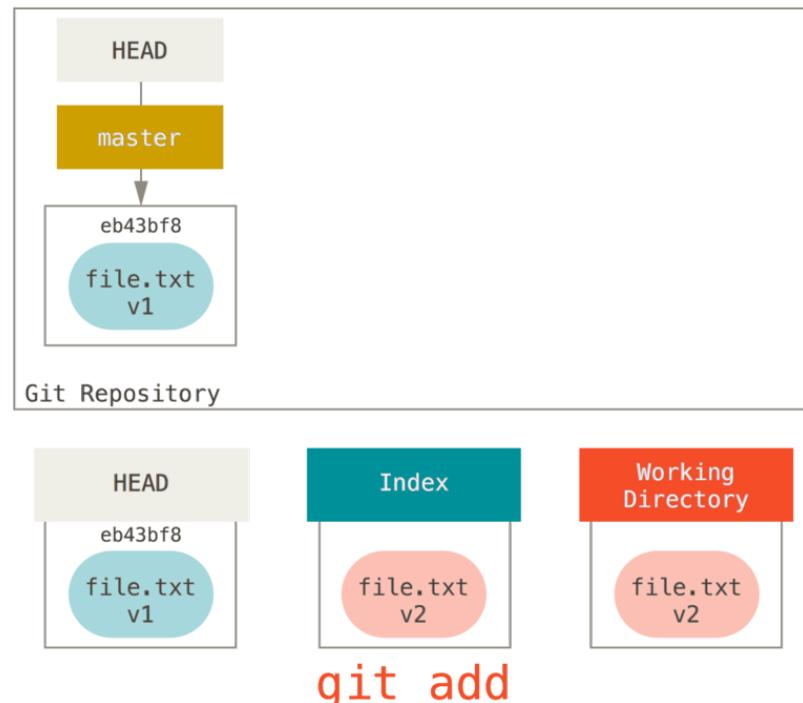


`git status`  
staged for commit,"

`git add`

" Changes not

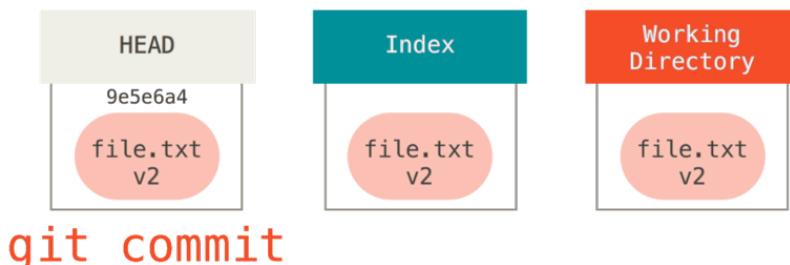
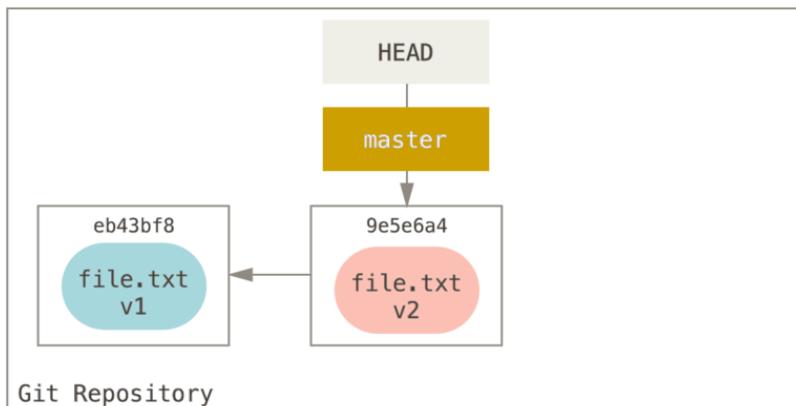
FIGURE 7-7



HEAD  
“ Changes to be committed”

git status  
—  
git commit

**FIGURE 7-8**

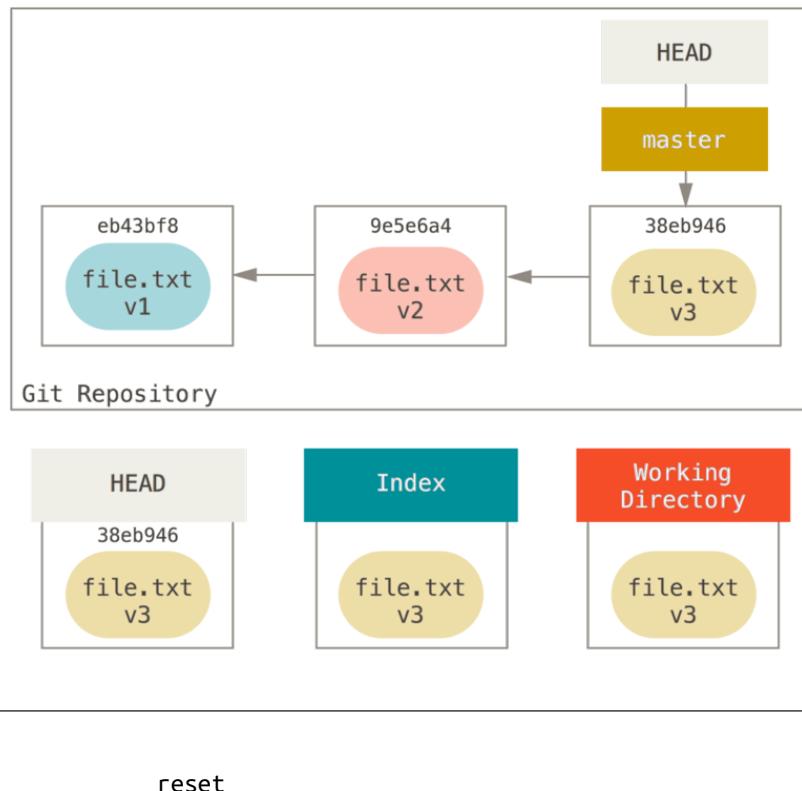


git status

HEAD

reset

file.txt

**FIGURE 7-9****第1步：移动 HEAD**

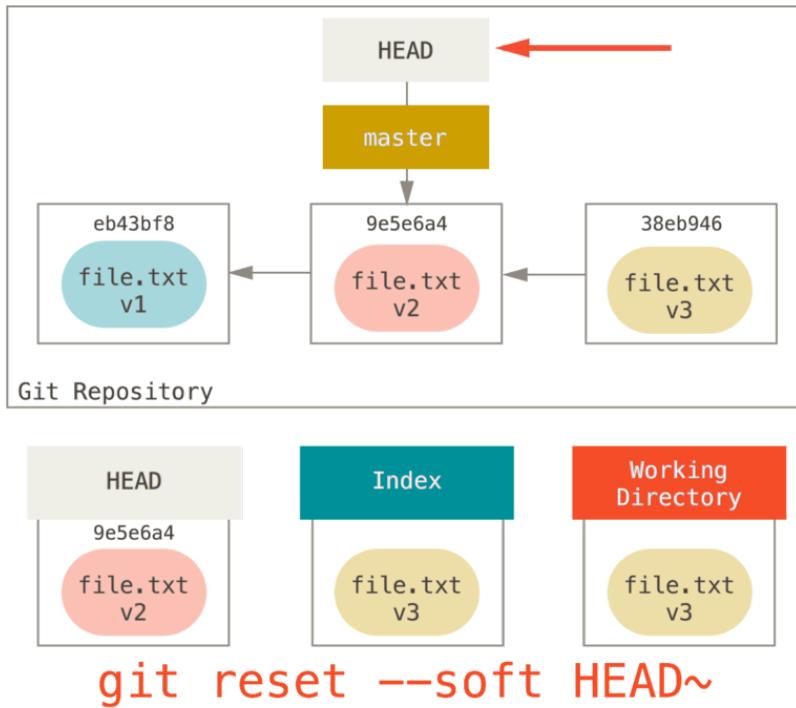
```

reset           HEAD           HEAD
  checkout      reset       HEAD
HEAD   master    master     master
set 9e5e64a      master    9e5e64a

```

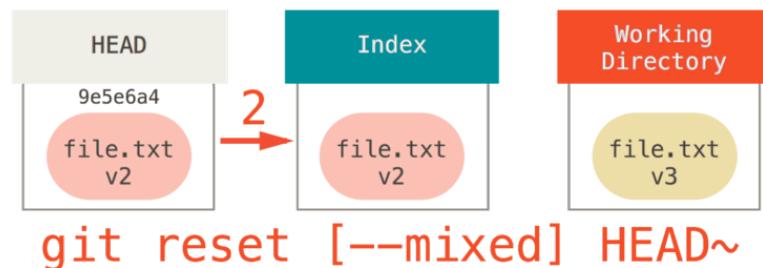
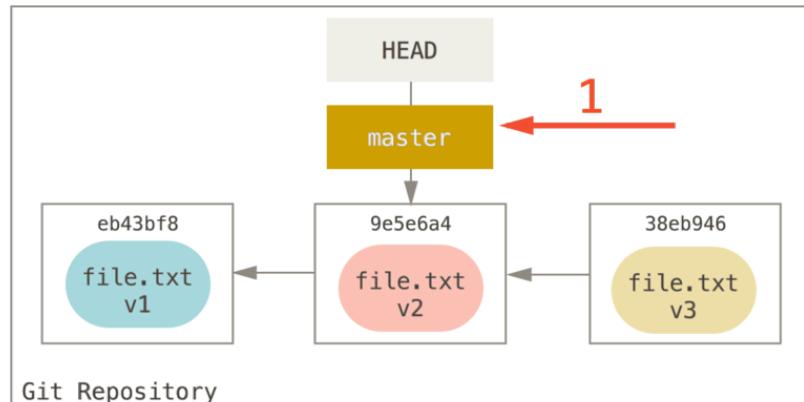
git re-

**FIGURE 7-10**

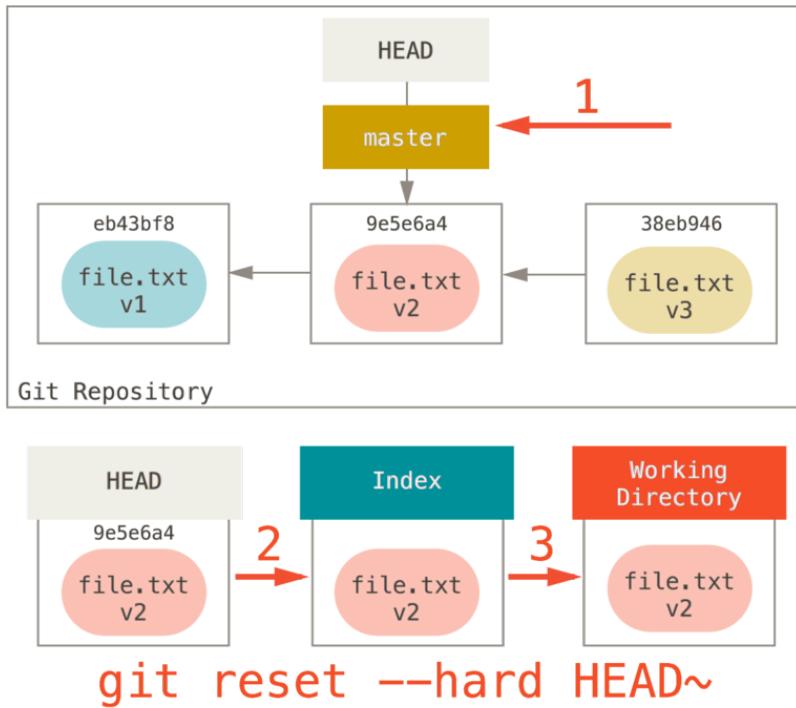


```
reset  
reset --soft  
commit  
  HEAD  
  HEAD  
commit --amend  
git commit  
git commit  
git  
  reset  
  HEAD~  
git commit  
git  
  HEAD~  
git status  
HEAD  
reset HEAD
```

第 2 步：更新索引 ( --MIXED )

**FIGURE 7-11**`--mixed``reset``git reset HEAD~1``commit``提交``git add git``第 3 步：更新工作目录 ( --HARD )``reset``hard``--`

**FIGURE 7-12**



```
git commit  
    --hard      reset  
    --hard  
        Git  
    reflog  
        Git  
        v3  
        Git  
        Git
```

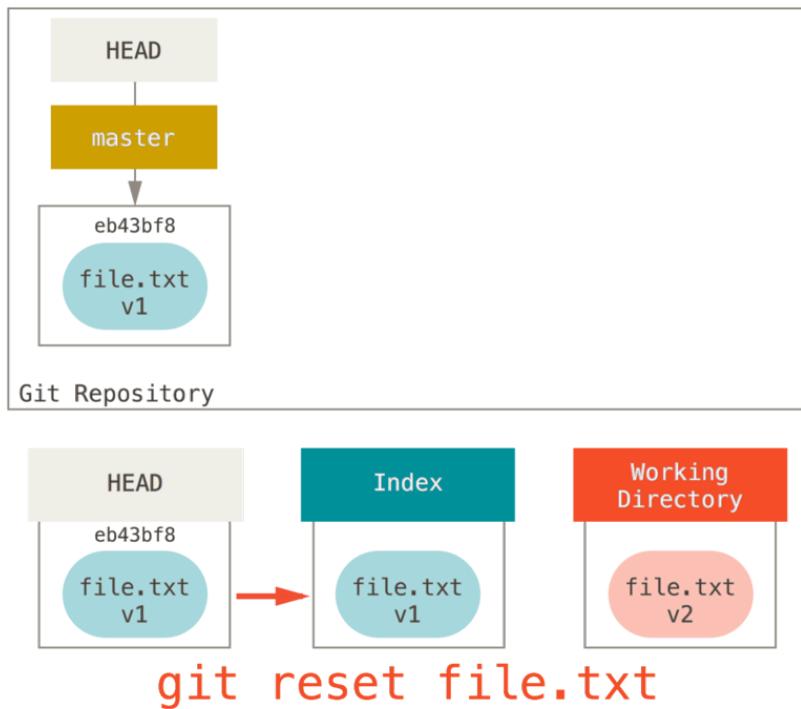
`git add`  
`Git`  
`reset`  
`v3`  
`Git`

回顾

`reset`

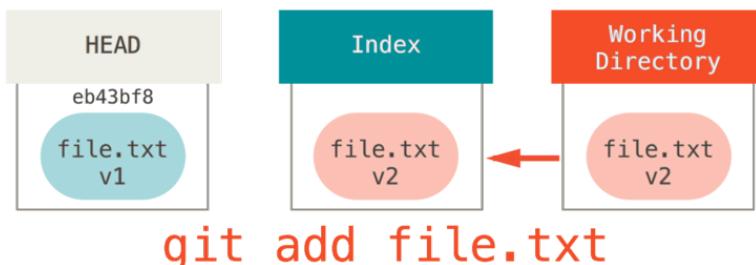
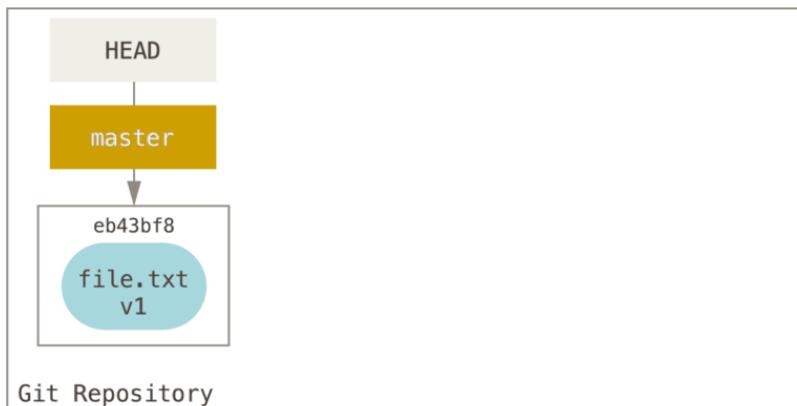
1. HEAD `--soft`
2. HEAD `--hard`
- 3.

```
reset          reset      1  
               HEAD  
  
2 3  
git reset file.txt      git reset --  
mixed HEAD file.txt      SHA-1  
--soft   --hard  
  
1.    HEAD  
2.    HEAD  
      file.txt  HEAD
```

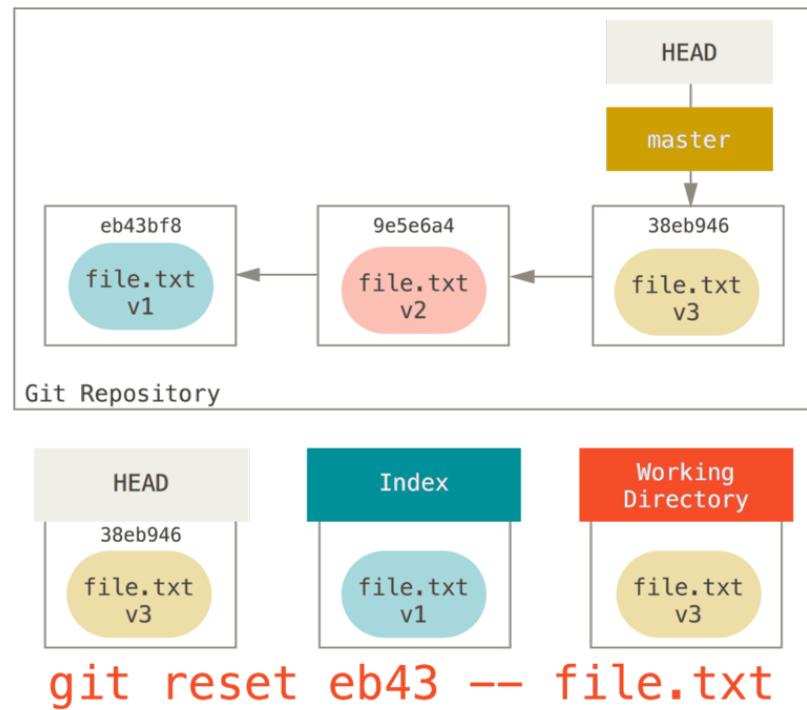
**FIGURE 7-13**

`git add`

**FIGURE 7-14**



git status  
" " " "  
Git HEAD  
git reset eb43bf file.txt

**FIGURE 7-15**

```

git add           v3
v1      "        git commit      "
          "         "           "
          "         git add       reset
          "         "         --patch

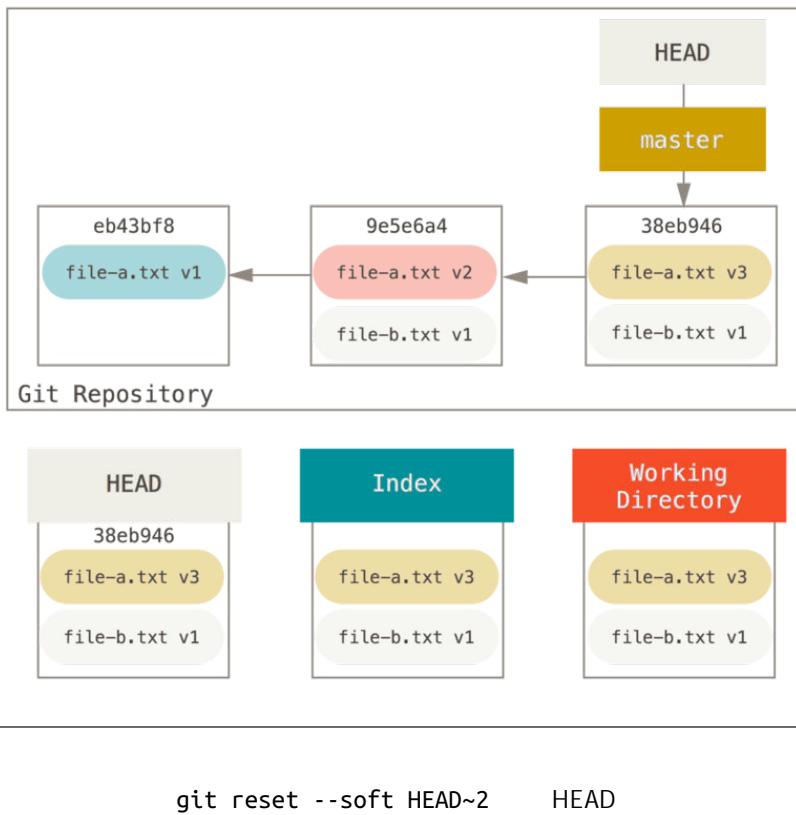
```

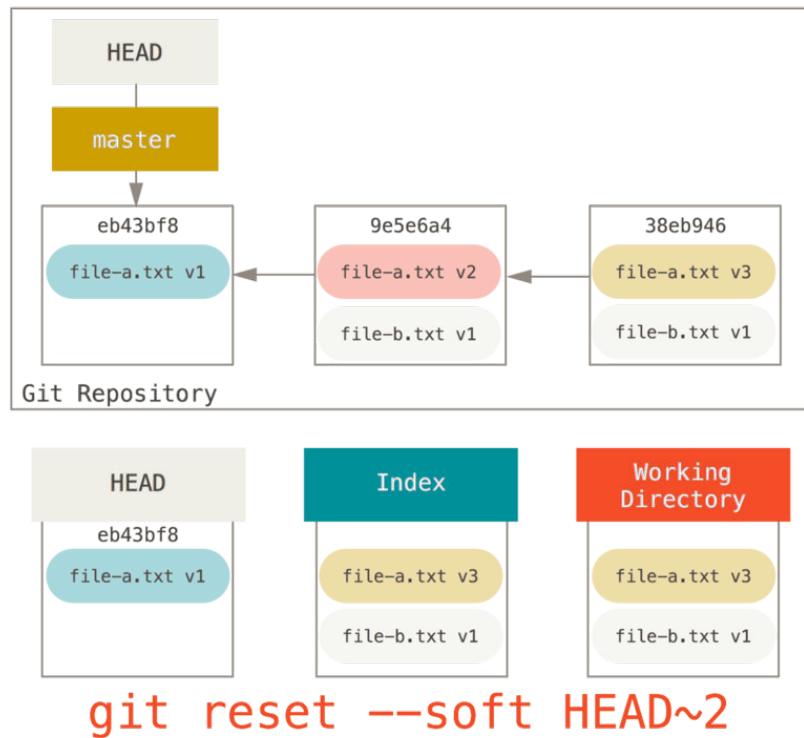
```

          " oops. "   " WIP"    " forgot this
file"      reset      "           "
          "           "           "
          "           re-
set

```

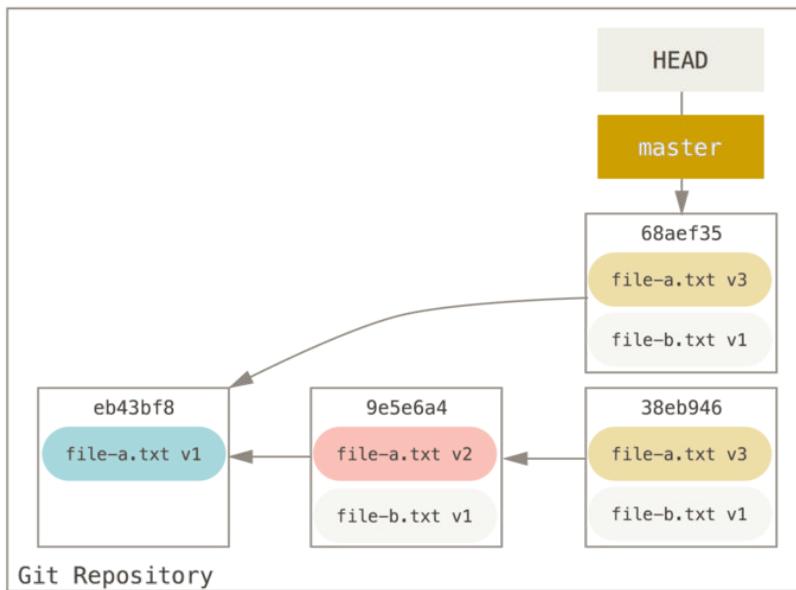
**FIGURE 7-16**



**FIGURE 7-17**

`git commit`

**FIGURE 7-18**



|                          |           |                          |           |
|--------------------------|-----------|--------------------------|-----------|
| file-a.txt<br>file-b.txt | <b>v2</b> | file-a.txt<br>file-b.txt | <b>v1</b> |
|--------------------------|-----------|--------------------------|-----------|

|                                    |              |              |
|------------------------------------|--------------|--------------|
| <b>checkout</b><br><b>checkout</b> | <b>reset</b> | <b>reset</b> |
|------------------------------------|--------------|--------------|

## 不带路径

`git checkout [branch]`      `git reset --hard [branch]  
[branch]`

`reset --hard checkout`

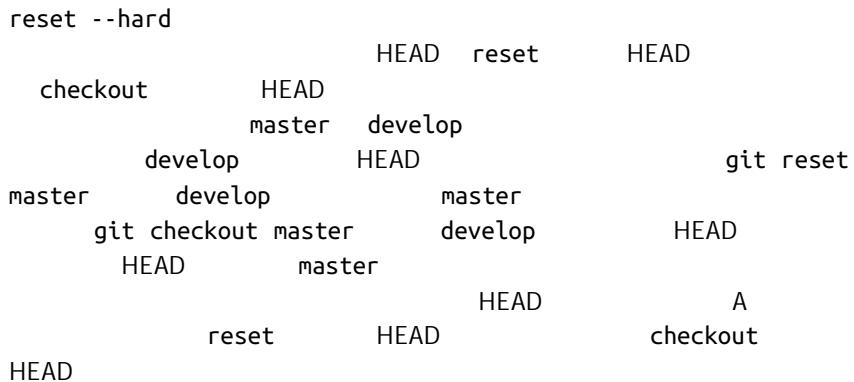
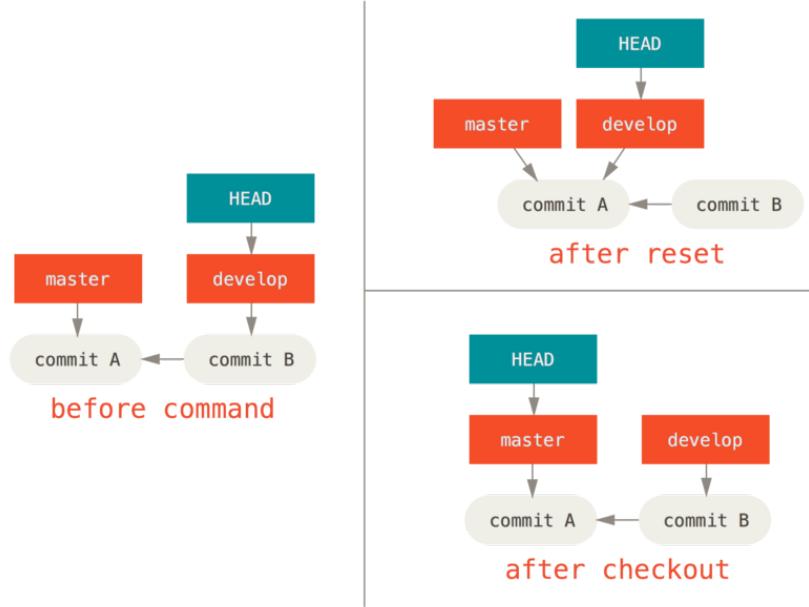


FIGURE 7-19



## 带路径

```
checkout          reset
HEAD      git reset [branch] file
reset --hard [branch] file      git
                           -  
reset
                           HEAD
git reset   git add   checkout   --patch
```

```
reset          checkout
               " HEAD"      " REF"
HEAD          " HEAD"
WD Safe?      -           NO
```

|                          | HEAD | Index | Workdir | WD Safe? |
|--------------------------|------|-------|---------|----------|
| <b>Commit Level</b>      |      |       |         |          |
| reset --soft [commit]    | REF  | NO    | NO      | YES      |
| reset [commit]           | REF  | YES   | NO      | YES      |
| reset --hard [commit]    | REF  | YES   | YES     | NO       |
| checkout [commit]        | HEAD | YES   | YES     | YES      |
| <b>File Level</b>        |      |       |         |          |
| reset (commit) [file]    | NO   | YES   | NO      | YES      |
| checkout (commit) [file] | NO   | YES   | YES     | NO       |

## 高级合并

Git                          Git

                                Git

Git

*“ “*

Git

*hello world Ruby**#! /usr/bin/env ruby*

```
def hello
  puts 'hello world'
end
```

*hello()*

whitespace

Unix

DOS

*“ hello world” “ hello mundo”*

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

$ unix2dos hello.rb
unix2dos: converting file hello.rb to DOS format ...
$ git commit -am 'converted hello.rb to DOS'
[whitespace 3270f76] converted hello.rb to DOS
 1 file changed, 7 insertions(+), 7 deletions(-)

$ vim hello.rb
$ git diff -b
diff --git a/hello.rb b/hello.rb
index ac51efd..e85207e 100755
--- a/hello.rb
+++ b/hello.rb
```

```
@@ -1,7 +1,7 @@
#! /usr/bin/env ruby

def hello
- puts 'hello world'
+ puts 'hello mundo'^M
end

hello()

$ git commit -am 'hello mundo change'
[whitespace 6d338d2] hello mundo change
1 file changed, 1 insertion(+), 1 deletion(-)
```

### master

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

$ vim hello.rb
$ git diff
diff --git a/hello.rb b/hello.rb
index ac51efd..36c06c8 100755
--- a/hello.rb
+++ b/hello.rb
@@ -1,5 +1,6 @@
 #! /usr/bin/env ruby

+# prints out a greeting
def hello
  puts 'hello world'
end

$ git commit -am 'document the function'
[master bec6336] document the function
1 file changed, 1 insertion(+)
```

### whitespace

```
$ git merge whitespace
Auto-merging hello.rb
CONFLICT (content): Merge conflict in hello.rb
Automatic merge failed; fix conflicts and then commit the result.
```

## 中断一次合并

```
git merge --abort
```

```
$ git status -sb
## master
UU hello.rb

$ git merge --abort

$ git status -sb
## master
```

```
git merge --abort
```

```
git reset --hard HEAD
```

## 忽略空白

Git

-Xignore-all-space    -Xignore-space-change

```
$ git merge -Xignore-space-change whitespace
Auto-merging hello.rb
Merge made by the 'recursive' strategy.
hello.rb | 2 ++
1 file changed, 1 insertion(+), 1 deletion(-)
```

## 手动文件再合并

Git

Git

Git

`dos2unix`

Git

" stages"  
  stage 2  
    " theirs"  
`git show`

Stage 1  
stage 3  
MERGE\_HEAD

```
$ git show :1:hello.rb > hello.common.rb  
$ git show :2:hello.rb > hello.ours.rb  
$ git show :3:hello.rb > hello.theirs.rb
```

`ls-files -u`

Git blob

SHA-1

```
$ git ls-files -u  
100755 ac51efdc3df4f4fd328d1a02ad05331d8e2c9111 1      hello.rb  
100755 36c06c8752c78d2aff89571132f3bf7841a7b5c3 2      hello.rb  
100755 e85207e04dfdd5eb0a1e9febbc67fd837c44a1cd 3      hello.rb
```

:1:hello.rb        blob        SHA-1

`git merge-file`

```
$ dos2unix hello.theirs.rb
dos2unix: converting file hello.theirs.rb to Unix format ...

$ git merge-file -p \
    hello.ours.rb hello.common.rb hello.theirs.rb > hello.rb

$ git diff -b
diff --cc hello.rb
index 36c06c8,e85207e..0000000
--- a/hello.rb
+++ b/hello.rb
@@@ -1,8 -1,7 +1,8 @@@
  #! /usr/bin/env ruby

  +# prints out a greeting
  def hello
-  puts 'hello world'
+  puts 'hello mundo'
  end

  hello()
```

ignore-space-change

ignore-space-change

DOS

git diff

git diff --ours

```
$ git diff --ours
* Unmerged path hello.rb
diff --git a/hello.rb b/hello.rb
index 36c06c8..44d0a25 100755
--- a/hello.rb
+++ b/hello.rb
@@ -2,7 +2,7 @@
  # prints out a greeting
  def hello
-  puts 'hello world'
+  puts 'hello mundo'
  end
```

```
hello()
```

```
git  
diff --theirs -b  
Git  
hello.theirs.rb
```

```
$ git diff --theirs -b  
* Unmerged path hello.rb  
diff --git a/hello.rb b/hello.rb  
index e85207e..44d0a25 100755  
--- a/hello.rb  
+++ b/hello.rb  
@@ -1,5 +1,6 @@  
#! /usr/bin/env ruby  
  
+# prints out a greeting  
def hello  
  puts 'hello mundo'  
end
```

```
git diff --base
```

```
$ git diff --base -b  
* Unmerged path hello.rb  
diff --git a/hello.rb b/hello.rb  
index ac51efd..44d0a25 100755  
--- a/hello.rb  
+++ b/hello.rb  
@@ -1,7 +1,8 @@  
#! /usr/bin/env ruby  
  
+# prints out a greeting  
def hello  
- puts 'hello world'  
+ puts 'hello mundo'  
end  
  
hello()
```

```
git clean
```

```
$ git clean -f
Removing hello.common.rb
Removing hello.ours.rb
Removing hello.theirs.rb
```

检出冲突

```
$ git log --graph --oneline --decorate --all
* f1270f7 (HEAD, master) update README
* 9af9d3b add a README
* 694971d update phrase to hola world
| * e3eb223 (mundo) add more tests
| * 7cff591 add testing script
| * c3ffff1 changed text to hello mundo
|/
* b7dcc89 initial hello world code
```

```
graph TD; master1[b7dcc89] --> master2[mundo]; mundo[mundo] --> master2
```

```
$ git merge mundo
Auto-merging hello.rb
CONFLICT (content): Merge conflict in hello.rb
Automatic merge failed; fix conflicts and then commit the result.
```

```
#!/usr/bin/env ruby

def hello
<<<<<< HEAD
```

```
    puts 'hola world'  
=====  
    puts 'hello mundo'  
>>>>> mundo  
end  
  
hello()
```

--conflict git checkout

```
--conflict diff3 merge  
diff3 Git  
" ours" " theirs" " base"
```

```
$ git checkout --conflict=diff3 hello.rb
```

```
#!/usr/bin/env ruby  
  
def hello  
<<<<< ours  
    puts 'hola world'  
||||||| base  
    puts 'hello world'  
=====  
    puts 'hello mundo'  
>>>>> theirs  
end  
  
hello()
```

```
merge.conflictstyle  
diff3
```

```
$ git config --global merge.conflictstyle diff3
```

git checkout --ours --theirs

## 合并日志

git log

```
$ git log --oneline --left-right HEAD...MERGE_HEAD
< f1270f7 update README
< 9af9d3b add a README
< 694971d update phrase to hola world
> e3eb223 add more tests
> 7cff591 add testing script
> c3ffff1 changed text to hello mundo
```

6

--merge git log

```
$ git log --oneline --left-right --merge
< 694971d update phrase to hola world
> c3ffff1 changed text to hello mundo
```

-p

## 组合式差异格式

Git

git diff

git diff

```
$ git diff
diff --cc hello.rb
index 0399cd5,59727f0..0000000
--- a/hello.rb
+++ b/hello.rb
@@@ -1,7 -1,7 +1,11 @@@
#! /usr/bin/env ruby

def hello
+<<<<< HEAD
+  puts 'hola mundo'
+=====
+  puts 'hello mundo'
+>>>>> mundo
end

hello()
```

" " "  
" ours"  
" theirs"  
  
<<<<<< >>>>>

### git diff

```
$ vim hello.rb
$ git diff
diff --cc hello.rb
index 0399cd5,59727f0..0000000
--- a/hello.rb
+++ b/hello.rb
@@@ -1,7 -1,7 +1,7 @@@
#! /usr/bin/env ruby

def hello
-  puts 'hola mundo'
-  puts 'hello mundo'
++  puts 'hola mundo'
end

hello()
```

“ hola world”  
mundo” “ hello mundo”

git log  
git show Git  
git log -p  
--cc

```
$ git log --cc -p -1
commit 14f41939956d80b9e17bb8721354c33f8d5b5a79
Merge: f1270f7 e3eb223
Author: Scott Chacon <schacon@gmail.com>
Date:   Fri Sep 19 18:14:49 2014 +0200

Merge branch 'mundo'

Conflicts:
  hello.rb

diff --cc hello.rb
index 0399cd5,59727f0..e1d0799
--- a/hello.rb
+++ b/hello.rb
@@@ -1,7 -1,7 +1,7 @@@
  #! /usr/bin/env ruby

  def hello
-  puts 'hola world'
-  puts 'hello mundo'
++  puts 'hola mundo'
  end

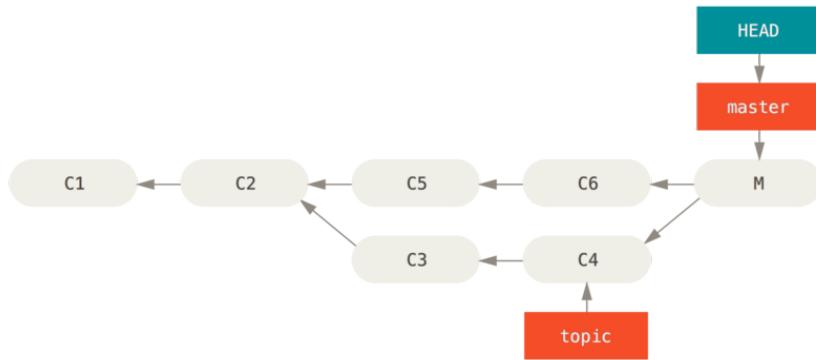
  hello()
```

Git

master

**FIGURE 7-20**

意外的合并提交

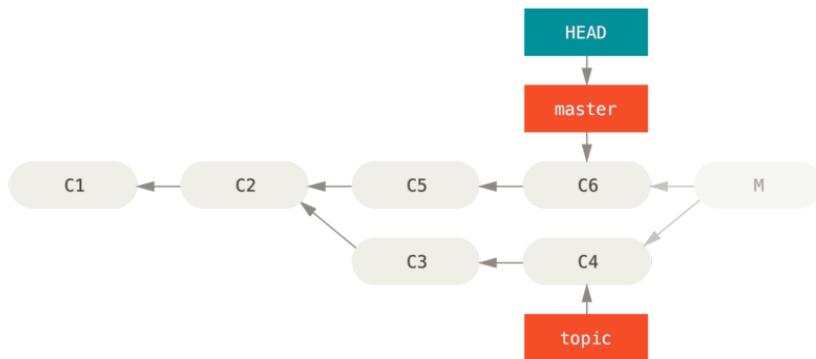


修复引用

`git merge`      `git reset --hard HEAD~`

**FIGURE 7-21**

在 `git reset --hard HEAD~` 之后的历史



1. HEAD  
   C6

      " " "

      reset  
      reset --hard

                        master

2.  
3.

HEAD

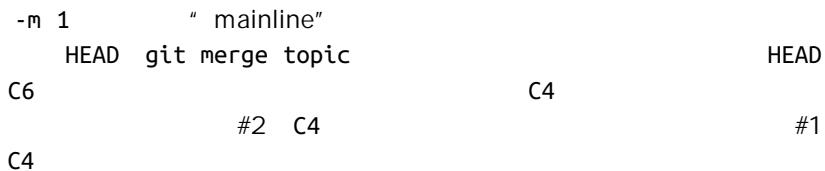
"

"

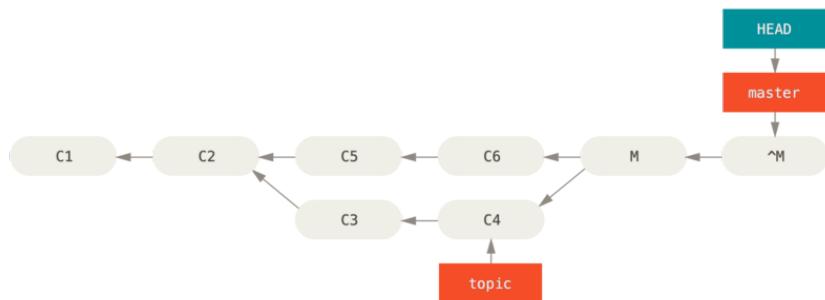
reset

还原提交

```
$ git revert -m 1 HEAD
[master b1d8379] Revert "Merge branch 'topic'"
```

**FIGURE 7-22**

在 `git revert -m 1`  
后的历史

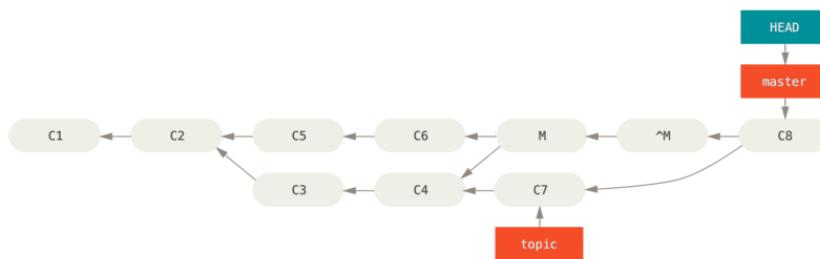


^M C6  
" "  
topic master Git

```
$ git merge topic  
Already up-to-date.
```

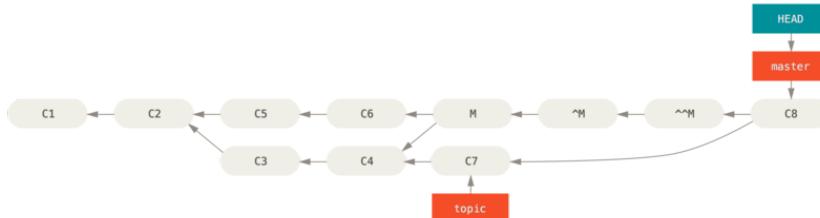
topic  
topic

master  
Git



**FIGURE 7-23**  
含有坏掉合并的历史

```
$ git revert ^M  
[master 09f0126] Revert "Revert "Merge branch 'topic'""  
$ git merge topic
```



**FIGURE 7-24**  
在重新合并一个还原  
合并后的历史

M      ^M      ^^M  
C7      topic      C3      C4      C8

“ recursive”

我们的或他们的偏好

```
        “ recursive”
-X ignore-all-space ignore-space-change
Git
Git
Git
Git
merge -Xours -Xtheirs
```

“ hello world”

```
$ git merge mundo
Auto-merging hello.rb
CONFLICT (content): Merge conflict in hello.rb
Resolved 'hello.rb' using previous resolution.
Automatic merge failed; fix conflicts and then commit the result.
```

-Xours -Xtheirs

```
$ git merge -Xours mundo
Auto-merging hello.rb
Merge made by the 'recursive' strategy.
hello.rb | 2 ++
test.sh | 2 ++
2 files changed, 3 insertions(+), 1 deletion(-)
create mode 100644 test.sh
```

“ hello mundo” “ hola world”
“ hola world”

```
git merge-file
git merge-file --ours
```

Git

" ours"

" ours" recursive

```
$ git merge -s ours mundo
Merge made by the 'ours' strategy.
$ git diff HEAD HEAD~
$
```

Git

release

```
bugfx          master      master
lease          release     bugfx      re-
               merge -s ours    master
                           release
bugfx
```

子树合并

Git

Rack

Rack

```
$ git remote add rack_remote https://github.com/rack/rack
$ git fetch rack_remote
warning: no common commits
remote: Counting objects: 3184, done.
remote: Compressing objects: 100% (1465/1465), done.
remote: Total 3184 (delta 1952), reused 2770 (delta 1675)
Receiving objects: 100% (3184/3184), 677.42 KiB | 4 KiB/s, done.
Resolving deltas: 100% (1952/1952), done.
From https://github.com/rack/rack
 * [new branch]      build      -> rack_remote/build
 * [new branch]      master     -> rack_remote/master
 * [new branch]      rack-0.4   -> rack_remote/rack-0.4
```

```
* [new branch]      rack-0.9  -> rack_remote/rack-0.9
$ git checkout -b rack_branch rack_remote/master
Branch rack_branch set up to track remote branch refs/remotes/rack_remote/master.
Switched to a new branch "rack_branch"
```

rack\_branch                      Rack  
master

```
$ ls
AUTHORS            KNOWN-ISSUES    Rakefile        contrib        lib
COPYING            README            bin              example        test
$ git checkout master
Switched to branch "master"
$ ls
README
```

Rack                      master
Git                  git read-tree      Chapter 10
read-tree
rack\_back            master              rack
                            master

```
$ git read-tree --prefix=rack/ -u rack_branch
```

Rack                      —

Rack

```
$ git checkout rack_branch
$ git pull
```

master                      --squash
-Xsubtree

```
$ git checkout master
$ git merge --squash -s recursive -Xsubtree=rack rack_branch
Squash commit -- not updating HEAD
Automatic merge went well; stopped before committing as requested
```

Rack

```
— master          rack
rack_branch
```

```
" "
```

```
          rack      rack_branch
          —          diff
git diff-tree
```

```
$ git diff-tree -p rack_branch
```

```
rack           master
```

```
$ git diff-tree -p rack_remote/master
```

## Rerere

```
git rerere             " reuse recorded
resolution"          Git
Git
```

```
rerere            rerere
```

## Git

rerere

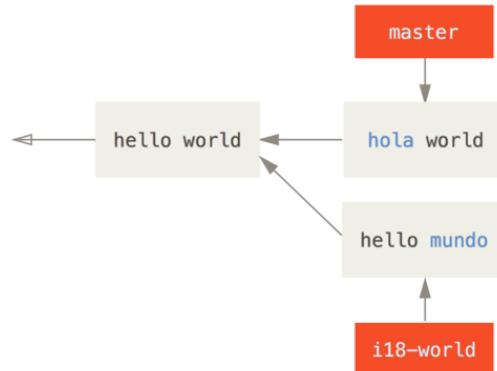
```
$ git config --global rerere.enabled true
```

```
.git/rr-cache
```

```
#!/usr/bin/env ruby

def hello
    puts 'hello world'
end
```

```
" hello"      " hola"
" world"      " mundo"
```

**FIGURE 7-25**

```
$ git merge i18n-world
Auto-merging hello.rb
CONFLICT (content): Merge conflict in hello.rb
Recorded preimage for 'hello.rb'
Automatic merge failed; fix conflicts and then commit the result.
```

```
Recorded preimage for FILE
          rerere
git status
```

```
$ git status
# On branch master
# Unmerged paths:
#   (use "git reset HEAD <file>..." to unstage)
#   (use "git add <file>..." to mark resolution)
#
#       both modified:    hello.rb
#
```

```
git rerere      git rerere status
```

```
$ git rerere status
hello.rb
```

```
git rerere diff
```

```
$ git rerere diff
--- a/hello.rb
+++ b/hello.rb
@@ -1,11 +1,11 @@
 #! /usr/bin/env ruby

 def hello
-<<<<<
- puts 'hello mundo'
-=====
+<<<<< HEAD
 puts 'hola world'
```

```
->>>>>
=====
+ puts 'hello mundo'
+>>>>> i18n-world
end
```

rerere                            ls-files -u

```
$ git ls-files -u
100644 39804c942a9c1f2c03dc7c5ebcd7f3e3a6b97519 1       hello.rb
100644 a440db6e8d1fd76ad438a49025a9ad9ce746f581 2       hello.rb
100644 54336ba847c3758ab604876419607e9443848474 3       hello.rb
```

rere diff                        puts 'hola mundo'                    re-
                                      rerere

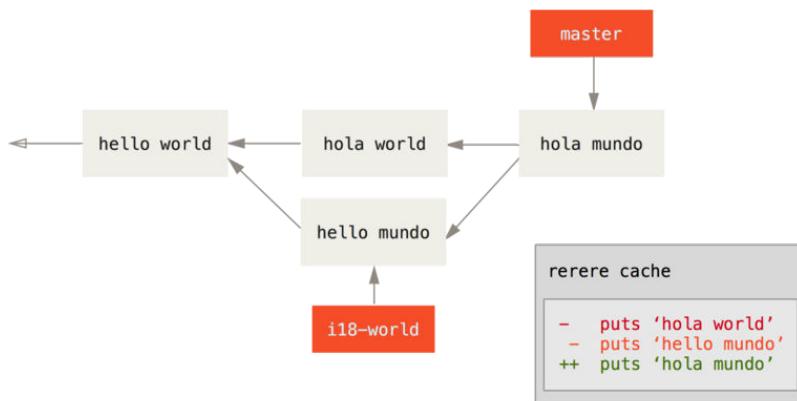
```
$ git rerere diff
--- a/hello.rb
+++ b/hello.rb
@@ -1,11 +1,7 @@
#! /usr/bin/env ruby

def hello
-<<<<<
- puts 'hello mundo'
-=====
- puts 'hola world'
->>>>>
+ puts 'hola mundo'
end
```

Git                                hello.rb
“ hello mundo”                    “ hola world”
“ hola mundo”

```
$ git add hello.rb
$ git commit
Recorded resolution for 'hello.rb'.
[master 68e16e5] Merge branch 'i18n'
```

“ Recorded resolution for FILE”

**FIGURE 7-26**

master  
"  
reset

```
$ git reset --hard HEAD^
HEAD is now at ad63f15 i18n the hello
```

```
$ git checkout i18n-world
Switched to branch 'i18n-world'

$ git rebase master
First, rewinding head to replay your work on top of it...
Applying: i18n one word
Using index info to reconstruct a base tree...
Falling back to patching base and 3-way merge...
Auto-merging hello.rb
CONFLICT (content): Merge conflict in hello.rb
Resolved 'hello.rb' using previous resolution.
Failed to merge in the changes.
Patch failed at 0001 i18n one word
```

Re-  
solved FILE using previous resolution

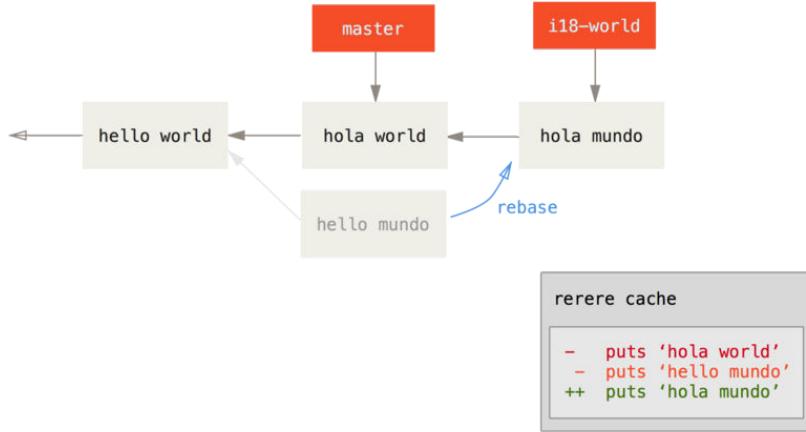
```
$ cat hello.rb
#!/usr/bin/env ruby

def hello
  puts 'hola mundo'
end

git diff
```

```
$ git diff
diff --cc hello.rb
index a440db6,54336ba..0000000
--- a/hello.rb
+++ b/hello.rb
@@@ -1,7 -1,7 +1,7 @@@
  #! /usr/bin/env ruby

  def hello
-   puts 'hola world'
-   puts 'hello mundo'
++   puts 'hola mundo'
  end
```

**FIGURE 7-27**

checkout

```
$ git checkout --conflict=merge hello.rb
$ cat hello.rb
#!/usr/bin/env ruby

def hello
<<<<< ours
  puts 'hola mundo'
=====
  puts 'hello mundo'
>>>>> theirs
end
```

      "      "

rerere

```
$ git rerere
Resolved 'hello.rb' using previous resolution.
$ cat hello.rb
#!/usr/bin/env ruby

def hello
  puts 'hola mundo'
end
```

      "      "

rerere

```
$ git add hello.rb
$ git rebase --continue
Applying: i18n one word
```

master

rerere

## 使用 Git 调试

Git

Git

bug

bug

git  
**blame**  
 -L                          12 22

```
$ git blame -L 12,22 simplegit.rb
^4832fe2 (Scott Chacon 2008-03-15 10:31:28 -0700 12) def show(tree = 'master')
^4832fe2 (Scott Chacon 2008-03-15 10:31:28 -0700 13)   command("git show #{tree}")
^4832fe2 (Scott Chacon 2008-03-15 10:31:28 -0700 14) end
^4832fe2 (Scott Chacon 2008-03-15 10:31:28 -0700 15)
9f6560e4 (Scott Chacon 2008-03-17 21:52:20 -0700 16) def log(tree = 'master')
79eaf55d (Scott Chacon 2008-04-06 10:15:08 -0700 17)   command("git log #{tree}")
9f6560e4 (Scott Chacon 2008-03-17 21:52:20 -0700 18) end
9f6560e4 (Scott Chacon 2008-03-17 21:52:20 -0700 19)
42cf2861 (Magnus Chacon 2008-04-13 10:45:01 -0700 20) def blame(path)
42cf2861 (Magnus Chacon 2008-04-13 10:45:01 -0700 21)   command("git blame #{path}")
42cf2861 (Magnus Chacon 2008-04-13 10:45:01 -0700 22) end
```

SHA-1

— —

^4832fe2

Git ^ SHA-1

Git

|        |                    |
|--------|--------------------|
| Git    | git blame          |
| -C Git | GITServerHandler.m |
|        | GITPackUpload.m    |
| load.m | GITPackUp-         |
| -C     | blame              |

```
$ git blame -C -L 141,153 GITPackUpload.m
f344f58d GITServerHandler.m (Scott 2009-01-04 141)
f344f58d GITServerHandler.m (Scott 2009-01-04 142) - (void) gatherObjectShasFromC
f344f58d GITServerHandler.m (Scott 2009-01-04 143) {
70befddd GITServerHandler.m (Scott 2009-03-22 144)           //NSLog(@"GATHER COMM
ad11ac80 GITPackUpload.m    (Scott 2009-03-24 145)
```

```

ad11ac80 GITPackUpload.m      (Scott 2009-03-24 146)      NSString *parentSha;
ad11ac80 GITPackUpload.m      (Scott 2009-03-24 147)      GITCommit *commit = [g
ad11ac80 GITPackUpload.m      (Scott 2009-03-24 148)      //NSLog(@"%@", commit);
ad11ac80 GITPackUpload.m      (Scott 2009-03-24 149)      if(commit) {
ad11ac80 GITPackUpload.m      (Scott 2009-03-24 150)      [refDict setOb
56ef2caf GITServerHandler.m  (Scott 2009-01-05 151)
56ef2caf GITServerHandler.m  (Scott 2009-01-05 152)
56ef2caf GITServerHandler.m  (Scott 2009-01-05 153)

```

Git

git bisect                      bisect

bug  
bug

```

                                  git bisect start
git bisect bad
                                  bisect
                                  git bi-
sect good [good_commit]

```

```

$ git bisect start
$ git bisect bad
$ git bisect good v1.0
Bisecting: 6 revisions left to test after this
[ecb6e1bc347ccecc5f9350d878ce677feb13d3b2] error handling on repo

```

Git                              (v1.0)                      12  
Git

git bisect good                Git

```
$ git bisect good
Bisecting: 3 revisions left to test after this
[b047b02ea83310a70fd603dc8cd7a6cd13d15c04] secure this thing
```

git bisect bad      Git

```
$ git bisect bad
Bisecting: 1 revisions left to test after this
[f71ce38690acf49c1f3c9bea38e09d82a5ce6014] drop exceptions table
```

Git  
SHA-1  
bug

```
$ git bisect good
b047b02ea83310a70fd603dc8cd7a6cd13d15c04 is first bad commit
commit b047b02ea83310a70fd603dc8cd7a6cd13d15c04
Author: PJ Hyett <pjhyett@example.com>
Date:   Tue Jan 27 14:48:32 2009 -0800

        secure this thing

:040000 040000 40ee3e7821b895e52c1695092db9bdc4c61d1730
f24d3c6ebfc639b1a3814550e62d60b8e68a8e4 M config
```

git bisect reset      HEAD

```
$ git bisect reset
```

0            git bisect                          bug  
    0  
    bisect start

```
$ git bisect start HEAD v1.0  
$ git bisect run test-error.sh
```

Git                            test-error.sh  
make        make tests

## 子模块

Atom  
Atom  
CPAN      Ruby gem

Git                            Git  
Git

Git  
git submodule add                            URL  
    " DbConnector"

```
$ git submodule add https://github.com/chaconinc/DbConnector  
Cloning into 'DbConnector'...  
remote: Counting objects: 11, done.  
remote: Compressing objects: 100% (10/10), done.  
remote: Total 11 (delta 0), reused 11 (delta 0)  
Unpacking objects: 100% (11/11), done.  
Checking connectivity... done.
```

“ DbConnector”

`git status`

```
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   .gitmodules
    new file:   DbConnector
```

.gitmodules URL

```
$ cat .gitmodules
[submodule "DbConnector"]
  path = DbConnector
  url = https://github.com/chaconinc/DbConnector
```

.gitignore

---

由于 `.gitmodules` 文件中的 URL 是人们首先尝试克隆/拉取的地方，因此请尽可能确保你使用的 URL 大家都能访问。例如，若你要使用的推送 URL 与他人的拉取 URL 不同，那么请使用他人能访问到的 URL。你也可以根据自己的需要，通过在本地执行 `git config submodule.DbConnector.url <私有 URL>` 来覆盖这个选项的值。如果可行的话，一个相对路径会很有帮助。

---

git status git  
diff

```
$ git diff --cached DbConnector
diff --git a/DbConnector b/DbConnector
new file mode 160000
index 000000..c3f01dc
--- /dev/null
+++ b/DbConnector
```

```
@@ -0,0 +1 @@
+Subproject commit c3f01dc8862123d317dd46284b05b6892c7b29bc
```

DbConnector

Git

```
git diff --submodule
```

```
$ git diff --cached --submodule
diff --git a/.gitmodules b/.gitmodules
new file mode 100644
index 0000000..71fc376
--- /dev/null
+++ b/.gitmodules
@@ -0,0 +1,3 @@
+[submodule "DbConnector"]
+    path = DbConnector
+    url = https://github.com/chaconinc/DbConnector
Submodule DbConnector 0000000...c3f01dc (new submodule)
```

```
$ git commit -am 'added DbConnector module'
[master fb9093c] added DbConnector module
2 files changed, 4 insertions(+)
create mode 100644 .gitmodules
create mode 160000 DbConnector
```

DbConnector

160000

Git

```
$ git clone https://github.com/chaconinc/MainProject
Cloning into 'MainProject'...
remote: Counting objects: 14, done.
remote: Compressing objects: 100% (13/13), done.
```

```

remote: Total 14 (delta 1), reused 13 (delta 0)
Unpacking objects: 100% (14/14), done.
Checking connectivity... done.
$ cd MainProject
$ ls -la
total 16
drwxr-xr-x  9 schacon  staff  306 Sep 17 15:21 .
drwxr-xr-x  7 schacon  staff  238 Sep 17 15:21 ..
drwxr-xr-x 13 schacon  staff  442 Sep 17 15:21 .git
-rw-r--r--  1 schacon  staff   92 Sep 17 15:21 .gitmodules
drwxr-xr-x  2 schacon  staff   68 Sep 17 15:21 DbConnector
-rw-r--r--  1 schacon  staff  756 Sep 17 15:21 Makefile
drwxr-xr-x  3 schacon  staff  102 Sep 17 15:21 includes
drwxr-xr-x  4 schacon  staff  136 Sep 17 15:21 scripts
drwxr-xr-x  4 schacon  staff  136 Sep 17 15:21 src
$ cd DbConnector/
$ ls
$
```

DbConnector    git  
submodule init    git submodule update

```

$ git submodule init
Submodule 'DbConnector' (https://github.com/chaconinc/DbConnector) registered for
$ git submodule update
Cloning into 'DbConnector'...
remote: Counting objects: 11, done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 0), reused 11 (delta 0)
Unpacking objects: 100% (11/11), done.
Checking connectivity... done.
Submodule path 'DbConnector': checked out 'c3f01dc8862123d317dd46284b05b6892c7b29b'
```

DbConnector    git clone                                  --recursive

```

$ git clone --recursive https://github.com/chaconinc/MainProject
Cloning into 'MainProject'...
remote: Counting objects: 14, done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 14 (delta 1), reused 13 (delta 0)
Unpacking objects: 100% (14/14), done.
Checking connectivity... done.
Submodule 'DbConnector' (https://github.com/chaconinc/DbConnector) registered for
```

```
Cloning into 'DbConnector'...
remote: Counting objects: 11, done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 0), reused 11 (delta 0)
Unpacking objects: 100% (11/11), done.
Checking connectivity... done.
Submodule path 'DbConnector': checked out 'c3f01dc8862123d317dd46284b05b6892c7b29bc'
```

## 拉取上游修改

```
git fetch  
git merge
```

```
$ git fetch  
From https://github.com/chaconinc/DbConnector  
  c3f01dc..d0354fc  master      -> origin/master  
$ git merge origin/master  
Updating c3f01dc..d0354fc  
Fast-forward  
  scripts/connect.sh | 1 +  
  src/db.c          | 1 +  
  2 files changed, 2 insertions(+)
```

```
git diff --submodule  
  
git diff --submodule diff.submodule  
" log"
```

```
$ git config --global diff.submodule log  
$ git diff  
Submodule DbConnector c3f01dc..d0354fc:  
> more efficient db routine  
> better connection routine
```

```
git submodule update --remote Git
```

```
$ git submodule update --remote DbConnector
remote: Counting objects: 4, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 4 (delta 2), reused 4 (delta 2)
Unpacking objects: 100% (4/4), done.
From https://github.com/chaconinc/DbConnector
  3f19983..d0354fc master      -> origin/master
Submodule path 'DbConnector': checked out 'd0354fc054692d3906c85c3af05ddce39a1c064'
```

```
          master
          DbConnector
" stable"           .gitmodules
                     .git/config
.gitmodules
```

```
$ git config -f .gitmodules submodule.DbConnector.branch stable
```

```
$ git submodule update --remote
remote: Counting objects: 4, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 4 (delta 2), reused 4 (delta 2)
Unpacking objects: 100% (4/4), done.
From https://github.com/chaconinc/DbConnector
  27cf5d3..c87d55d stable -> origin/stable
Submodule path 'DbConnector': checked out 'c87d55d4c6d4b05ee34fbc8cb6f7bf4585ae668'
```

```
-f .gitmodules
```

```
git status Git
```

```
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
```

```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

modified:   .gitmodules
modified:   DbConnector (new commits)
```

```
no changes added to commit (use "git add" and/or "git commit -a")
```

### status.submodulesummary Git

```
$ git config status.submodulesummary 1

$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   .gitmodules
    modified:   DbConnector (new commits)

Submodules changed but not updated:

* DbConnector c3f01dc...c87d55d (4):
  > catch non-null terminated lines
```

git diff .gitmodules

```
$ git diff
diff --git a/.gitmodules b/.gitmodules
index 6fc0b3d..fd1cc29 100644
--- a/.gitmodules
+++ b/.gitmodules
@@ -1,3 +1,4 @@
 [submodule "DbConnector"]
     path = DbConnector
     url = https://github.com/chaconinc/DbConnector
+
     branch = stable
Submodule DbConnector c3f01dc..c87d55d:
  > catch non-null terminated lines
  > more robust error handling
  > more efficient db routine
  > better connection routine
```

git log -p

```
$ git log -p --submodule
commit 0a24fcf121a8a3c118e0105ae4ae4c00281cf7ae
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Sep 17 16:37:02 2014 +0200

        updating DbConnector for bug fixes

diff --git a/.gitmodules b/.gitmodules
index 6fc0b3d..fd1cc29 100644
--- a/.gitmodules
+++ b/.gitmodules
@@ -1,3 +1,4 @@
[submodule "DbConnector"]
    path = DbConnector
    url = https://github.com/chaconinc/DbConnector
+   branch = stable
Submodule DbConnector c3f01dc..c87d55d:
    > catch non-null terminated lines
    > more robust error handling
    > more efficient db routine
    > better connection routine
```

`git submodule update --remote`      Git

## 在子模块上工作

Maven    Rubygems

```
git submodule update
Git
"
HEAD"
" master"
                               Git
git submodule update --remote
```

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

```
" merge" update  
--merge
```

```
$ git submodule update --remote --merge  
remote: Counting objects: 4, done.  
remote: Compressing objects: 100% (2/2), done.  
remote: Total 4 (delta 2), reused 4 (delta 2)  
Unpacking objects: 100% (4/4), done.  
From https://github.com/chaconinc/DbConnector  
  c87d55d..92c7337 stable    -> origin/stable  
Updating c87d55d..92c7337  
Fast-forward  
 src/main.c | 1 +  
 1 file changed, 1 insertion(+)  
Submodule path 'DbConnector': merged in '92c7337b30ef9e0893e758dac2459d07362ab5ea'
```

```
DbConnector  
stable
```

```
$ cd DbConnector/  
$ vim src/db.c  
$ git commit -am 'unicode support'  
[stable f906e16] unicode support  
 1 file changed, 1 insertion(+)
```

```
$ git submodule update --remote --rebase  
First, rewinding head to replay your work on top of it...  
Applying: unicode support  
Submodule path 'DbConnector': rebased into '5d60ef9bbebf5a0c1c1050f242ceeb54ad58da94'
```

```
--rebase  --merge  Git  
HEAD
```

```
$ git submodule update --remote
```

Submodule path 'DbConnector': checked out '5d60ef9bbebf5a0c1c1050f242ceeb54ad58da9'

origin/stable

## Git

```
$ git submodule update --remote
```

remote: Counting objects: 4, done.

remote: Compressing objects: 100% (3/3), done.

remote: Total 4 (delta 0), reused 4 (delta 0)

Unpacking objects: 100% (4/4), done.

From https://github.com/chaconinc/DbConnector

  5d60ef9..c75e92a stable -> origin/stable

error: Your local changes to the following files would be overwritten by checkout:

  scripts/setup.sh

Please, commit your changes or stash them before you can switch branches.

Aborting

Unable to checkout 'c75e92a2b3855c9e5b66f915308390d9db204aca' in submodule path 'D

Git

```
$ git submodule update --remote --merge
```

Auto-merging scripts/setup.sh

CONFLICT (content): Merge conflict in scripts/setup.sh

Recorded preimage for 'scripts/setup.sh'

Automatic merge failed; fix conflicts and then commit the result.

Unable to merge 'c75e92a2b3855c9e5b66f915308390d9db204aca' in submodule path 'DbC

## 发布子模块改动

```
$ git diff
```

Submodule DbConnector c87d55d..82d2ad3:

  > Merge from origin/stable

```
> updated setup script  
> unicode support  
> remove unnecessary method  
> add new option for conn pooling
```

## Git

```
git push " check" " on-demand"  
--recurse-submodules  
" check" push
```

```
$ git push --recurse-submodules=check  
The following submodule paths contain changes that can  
not be found on any remote:
```

```
DbConnector
```

Please try

```
git push --recurse-submodules=on-demand
```

or cd to the path and use

```
git push
```

to push them to a remote.

" on-demand"

```
$ git push --recurse-submodules=on-demand  
Pushing submodule 'DbConnector'  
Counting objects: 9, done.  
Delta compression using up to 8 threads.  
Compressing objects: 100% (8/8), done.  
Writing objects: 100% (9/9), 917 bytes | 0 bytes/s, done.  
Total 9 (delta 3), reused 0 (delta 0)  
To https://github.com/chaconinc/DbConnector  
 c75e92a..82d2ad3 stable -> stable  
Counting objects: 2, done.  
Delta compression using up to 8 threads.  
Compressing objects: 100% (2/2), done.
```

```
Writing objects: 100% (2/2), 266 bytes | 0 bytes/s, done.
Total 2 (delta 1), reused 0 (delta 0)
To https://github.com/chaconinc/MainProject
  3d6d338..9a377d1  master -> master
```

Git      DbConnector

### 合并子模块改动

Git

Git

```
$ git pull
remote: Counting objects: 2, done.
remote: Compressing objects: 100% (1/1), done.
remote: Total 2 (delta 1), reused 2 (delta 1)
Unpacking objects: 100% (2/2), done.
From https://github.com/chaconinc/MainProject
  9a377d1..eb974f8  master      -> origin/master
Fetching submodule DbConnector
warning: Failed to merge submodule DbConnector (merge following commits not found)
Auto-merging DbConnector
CONFLICT (submodule): Merge conflict in DbConnector
Automatic merge failed; fix conflicts and then commit the result.
```

Git

“ merge following commits not found”

Git

SHA-1

**git diff**

SHA-1

```
$ git diff
diff --cc DbConnector
index eb41d76,c771610..0000000
```

```
--- a/DbConnector  
+++ b/DbConnector
```

eb41d76

c771610

eb41d76

SHA-1

SHA-1

git diff

SHA

```
$ cd DbConnector
```

```
$ git rev-parse HEAD  
eb41d764bccf88be77aced643c13a7fa86714135
```

```
$ git branch try-merge c771610  
(DbConnector) $ git merge try-merge  
Auto-merging src/main.c  
CONFLICT (content): Merge conflict in src/main.c  
Recorded preimage for 'src/main.c'  
Automatic merge failed; fix conflicts and then commit the result.
```

```
$ vim src/main.c ①  
$ git add src/main.c  
$ git commit -am 'merged our changes'  
Recorded resolution for 'src/main.c'.  
[master 9fd905e] merged our changes
```

```
$ cd .. ②  
$ git diff ③  
diff --cc DbConnector  
index eb41d76,c771610..0000000  
--- a/DbConnector  
+++ b/DbConnector  
@@@ -1,1 -1,1 +1,1 @@@  
- Subproject commit eb41d764bccf88be77aced643c13a7fa86714135  
- Subproject commit c77161012afbbe1f58b5053316ead08f4b7e6d1d  
++Subproject commit 9fd905e5d7f45a0d4cbc43d1ee550f16a30e825a  
$ git add DbConnector ④
```

```
$ git commit -m "Merge Tom's Changes" ⑤
[master 10d2c60] Merge Tom's Changes
```

①

②

③ SHA-1

④

⑤

Git

Git

" merge following commits not found"

```
$ git merge origin/master
warning: Failed to merge submodule DbConnector (not fast-forward)
Found a possible merge resolution for the submodule:
  9fd905e5d7f45a0d4cbc43d1ee550f16a30e825a: > merged our changes
If this is correct simply add it to the index for example
by using:
```

```
git update-index --cacheinfo 160000 9fd905e5d7f45a0d4cbc43d1ee550f16a30e825a "Db
```

which will accept this suggestion.

Auto-merging DbConnector

CONFLICT (submodule): Merge conflict in DbConnector

Automatic merge failed; fix conflicts and then commit the result.

**git add**

```
$ cd DbConnector/
$ git merge 9fd905e
```

```
Updating eb41d76..9fd905e
Fast-forward

$ cd ..
$ git add DbConnector
$ git commit -am 'Fast forwarded to a common submodule child'
```

## 子模块遍历

foreach

```
$ git submodule foreach 'git stash'
Entering 'CryptoLibrary'
No local changes to save
Entering 'DbConnector'
Saved working directory and index state WIP on stable: 82d2ad3 Merge from origin/stable
HEAD is now at 82d2ad3 Merge from origin/stable
```

```
$ git submodule foreach 'git checkout -b featureA'
Entering 'CryptoLibrary'
Switched to a new branch 'featureA'
Entering 'DbConnector'
Switched to a new branch 'featureA'
```

```
$ git diff; git submodule foreach 'git diff'
Submodule DbConnector contains modified content
diff --git a/src/main.c b/src/main.c
```

```

index 210f1ae..1f0acdc 100644
--- a/src/main.c
+++ b/src/main.c
@@ -245,6 +245,8 @@ static int handle_alias(int *argcp, const char ***argv)

        commit_page_choice();

+    url = url_decode(url_orig);
+
/* build alias_argv */
alias_argv = xmalloc(sizeof(*alias_argv) * (argc + 1));
alias_argv[0] = alias_string + 1;
Entering 'DbConnector'
diff --git a/src/db.c b/src/db.c
index 1aaefb6..5297645 100644
--- a/src/db.c
+++ b/src/db.c
@@ -93,6 +93,11 @@ char *url_decode_mem(const char *url, int len)
        return url_decode_internal(&url, len, NULL, &out, 0);
}

+char *url_decode(const char *url)
+{
+    return url_decode_mem(url, strlen(url));
+}
+
char *url_decode_parameter_name(const char **query)
{
    struct strbuf out = STRBUF_INIT;

```

## 有用的别名

|     |         |     |
|-----|---------|-----|
| Git | " Git " | Git |
|-----|---------|-----|

```

$ git config alias.sdiff '!"git diff && git submodule foreach "git diff"'
$ git config alias.spush 'push --recurse-submodules=on-demand'
$ git config alias.supdate 'submodule update --remote --merge'

```

|       |             |     |
|-------|-------------|-----|
| spush | git supdate | git |
|-------|-------------|-----|

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

$ git submodule add https://github.com/chaconinc/CryptoLibrary
Cloning into 'CryptoLibrary'...
...

$ git commit -am 'adding crypto library'
[add-crypto 4445836] adding crypto library
 2 files changed, 4 insertions(+)
 create mode 160000 CryptoLibrary

$ git checkout master
warning: unable to rmdir CryptoLibrary: Directory not empty
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.

$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    CryptoLibrary/

nothing added to commit but untracked files present (use "git add" to track)
```

```
submodule update
--init
```

```
$ git clean -fdx
Removing CryptoLibrary/

$ git checkout add-crypto
Switched to branch 'add-crypto'

$ ls CryptoLibrary/
```

```
$ git submodule update --init
Submodule path 'CryptoLibrary': checked out 'b8dda6aa182ea4464f3f3264b11e026854517'
```

```
$ ls CryptoLibrary/
Makefile      includes      scripts      src
```

Git  
submodule  
add Git

```
$ rm -Rf CryptoLibrary/
$ git submodule add https://github.com/chaconinc/CryptoLibrary
'CryptoLibrary' already exists in the index
```

CryptoLibrary

```
$ git rm -r CryptoLibrary
$ git submodule add https://github.com/chaconinc/CryptoLibrary
Cloning into 'CryptoLibrary'...
remote: Counting objects: 11, done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 0), reused 11 (delta 0)
Unpacking objects: 100% (11/11), done.
Checking connectivity... done.
```

```
$ git checkout master
error: The following untracked working tree files would be overwritten by checkout
  CryptoLibrary/Makefile
  CryptoLibrary/includes/crypto.h
  ...
Please move or remove them before you can switch branches.
Aborting
```

check -f

```
$ git checkout -f master
warning: unable to rmdir CryptoLibrary: Directory not empty
Switched to branch 'master'
```

```
CryptoLibrary
git submodule update
git checkout .
foreach
    .git
        Git
            Git
```

## 打包

```
Git           HTTP   SSH
Git           "       "
format-patch      40
git bundle      bundle      git push
```

```
$ git log
commit 9a466c572fe88b195efd356c3f2bbeccdb504102
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Mar 10 07:34:10 2010 -0800

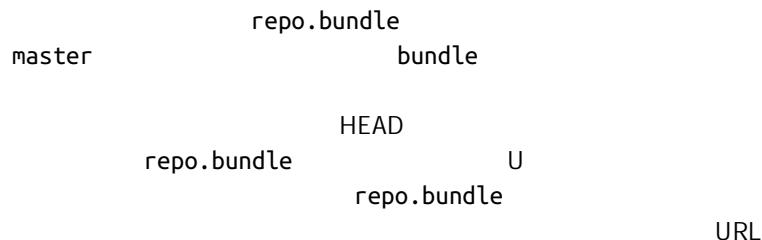
    second commit

commit b1ec3248f39900d2a406049d762aa68e9641be25
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Mar 10 07:34:01 2010 -0800

    first commit
```

```
git bundle create
```

```
$ git bundle create repo.bundle HEAD master
Counting objects: 6, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 441 bytes, done.
Total 6 (delta 0), reused 0 (delta 0)
```



```
$ git clone repo.bundle repo
Initialized empty Git repository in /private/tmp/bundle/repo/.git/
$ cd repo
$ git log --oneline
9a466c5 second commit
b1ec324 first commit
```



```
$ git log --oneline
71b84da last commit - second repo
c99cf5b fourth commit - second repo
7011d3d third commit - second repo
9a466c5 second commit
b1ec324 first commit
```

—

```
          "          "
origin/
master..master      master ^origin/master           3
          master                                log
```

```
$ git log --oneline master ^origin/master
71b84da last commit - second repo
c99cf5b fourth commit - second repo
7011d3d third commit - second repo
```

git bundle create

```
$ git bundle create commits.bundle master ^9a466c5
Counting objects: 11, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (9/9), 775 bytes, done.
Total 9 (delta 0), reused 0 (delta 0)
```

commits.bundle

bundle verify

Git

```
$ git bundle verify ../commits.bundle
The bundle contains 1 ref
71b84daaf49abed142a373b6e5c59a22dc6560dc refs/heads/master
The bundle requires these 1 ref
9a466c572fe88b195efd356c3f2bbeccdb504102 second commit
../commits.bundle is okay
```

verify

```
$ git bundle verify ../commits-bad.bundle
error: Repository lacks these prerequisite commits:
error: 7011d3d8fc200abe0ad561c011c3852a4b7bbe95 third commit - second repo
```

```
$ git bundle list-heads ../commits.bundle
71b84daaf49abed142a373b6e5c59a22dc6560dc refs/heads/master
```

verify

| fetch  | pull                |
|--------|---------------------|
| master | <i>other-master</i> |

```
$ git fetch ../commits.bundle master:other-master
From ../commits.bundle
 * [new branch]      master      -> other-master
```

*master**other-master*

```
$ git log --oneline --decorate --graph --all
* 8255d41 (HEAD, master) third commit - first repo
| * 71b84da (other-master) last commit - second repo
| * c99cf5b fourth commit - second repo
| * 7011d3d third commit - second repo
|/
* 9a466c5 second commit
* b1ec324 first commit
```

git bundle

## 替换

Git

Git

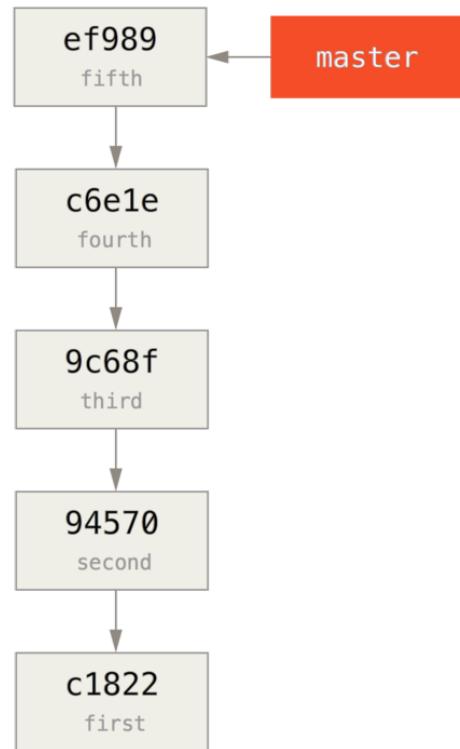
replace Git "

replace

SHA

SHA replace  
5

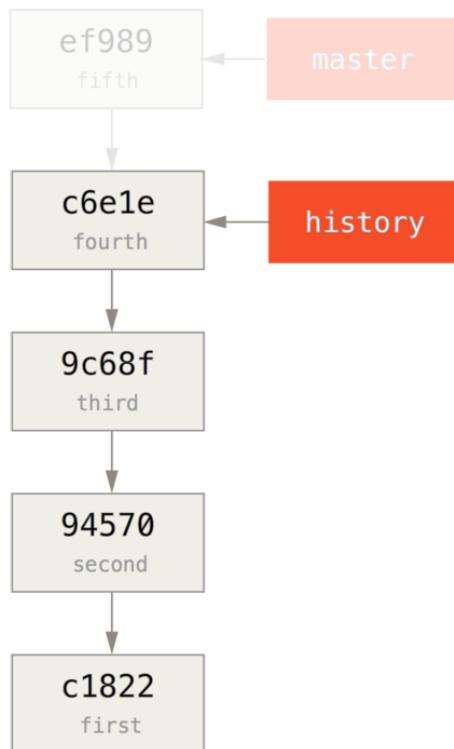
```
$ git log --oneline
ef989d8 fifth commit
c6e1e95 fourth commit
9c68fdc third commit
945704c second commit
c1822cf first commit
```

**FIGURE 7-28**

master

```
$ git branch history c6e1e95
$ git log --oneline --decorate
ef989d8 (HEAD, master) fifth commit
c6e1e95 (history) fourth commit
9c68fdc third commit
945704c second commit
c1822cf first commit
```

**FIGURE 7-29**



history

master

```
$ git remote add project-history https://github.com/schacon/project-history
$ git push project-history history:master
Counting objects: 12, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (12/12), 907 bytes, done.
Total 12 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (12/12), done.
To git@github.com:schacon/project-history.git
 * [new branch]      history -> master
```

```
$ git log --oneline --decorate  
ef989d8 (HEAD, master) fifth commit  
c6e1e95 (history) fourth commit  
9c68fdc third commit  
945704c second commit  
c1822cf first commit
```

SHA 9c68fdc  
commit-tree  
SHA

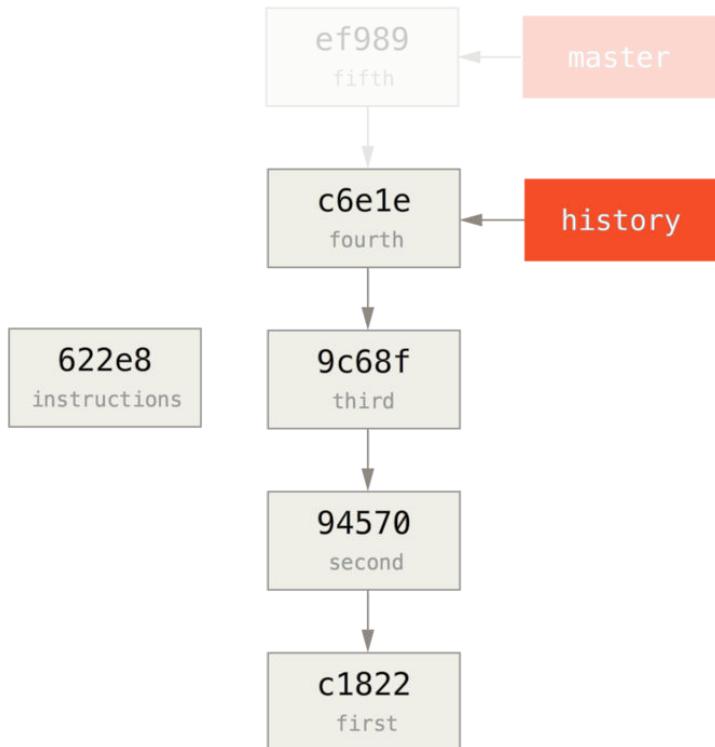
```
$ echo 'get history from blah blah blah' | git commit-tree 9c68fdc^{tree}  
622e88e9cbfbacfb75b5279245b9fb38dfa10cf
```

---

commit-tree 命令属于底层指令。有许多指令并非直接使用，而是被 **其他的** Git 命令用来做更小一些的工作。有时当我们做一些像这样的奇怪事情时，它们允许我们做一些不适用于日常使用但真正底层的东西。更多关于底层命令的内容请参见“**底层命令和高层命令**”

---

**FIGURE 7-30**

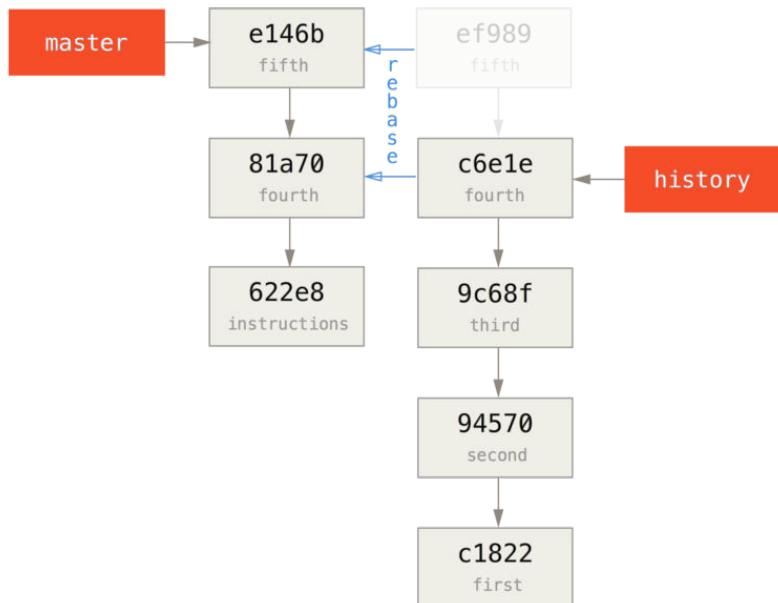


git rebase --onto  
--onto commit-tree

SHA

9c68fdc

```
$ git rebase --onto 622e88 9c68fdc
First, rewinding head to replay your work on top of it...
Applying: fourth commit
Applying: fifth commit
```

**FIGURE 7-31**

```
$ git clone https://github.com/schacon/project
$ cd project

$ git log --oneline master
e146b5f fifth commit
81a708d fourth commit
622e88e get history from blah blah blah

$ git remote add project-history https://github.com/schacon/project-history
$ git fetch project-history
From https://github.com/schacon/project-history
 * [new branch]      master      -> project-history/master
```

master project-  
history/master

```
$ git log --oneline master
e146b5f fifth commit
81a708d fourth commit
622e88e get history from blah blah blah

$ git log --oneline project-history/master
c6e1e95 fourth commit
9c68fdc third commit
945704c second commit
c1822cf first commit
```

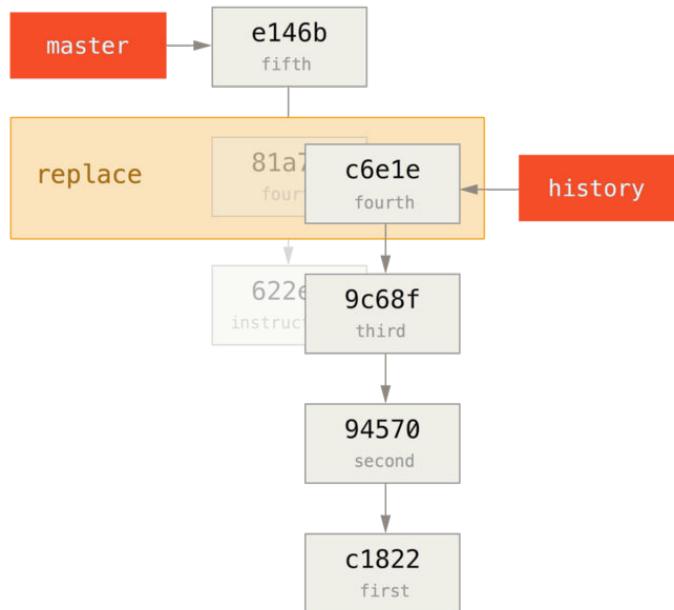
git replace  
master  
project-history/master " "

```
$ git replace 81a708d c6e1e95
```

master

```
$ git log --oneline master
e146b5f fifth commit
81a708d fourth commit
9c68fdc third commit
945704c second commit
c1822cf first commit
```

SHA-1  
bisect blame

**FIGURE 7-32**

c6e1e95  
81a708d                          cat-file                          SHA-1

```

$ git cat-file -p 81a708d
tree 7bc544cf438903b65ca9104a1e30345eee6c083d
parent 9c68fdceee073230f19ebb8b5e7fc71b479c0252
author Scott Chacon <schacon@gmail.com> 1268712581 -0700
committer Scott Chacon <schacon@gmail.com> 1268712581 -0700

fourth commit
  
```

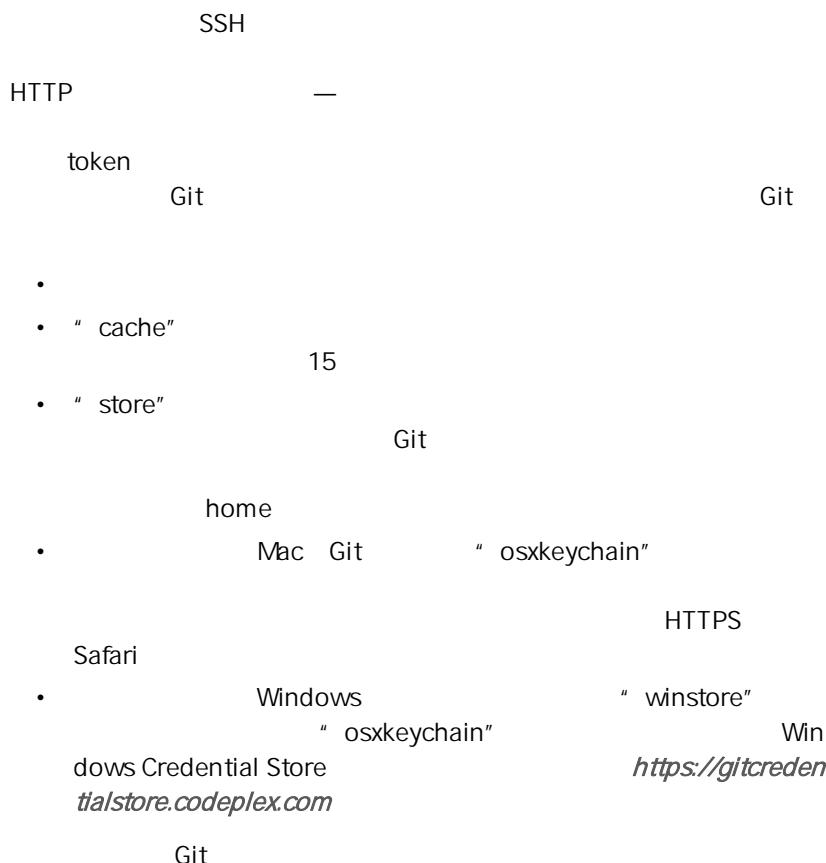
81a708d                          622e882  
9c68fdce

```

$ git for-each-ref
e146b5f14e79d4935160c0e83fb9ebe526b8da0d commit refs/heads/master
c6e1e95051d41771a649f3145423f8809d1a74d4 commit refs/remotes/history/master
  
```

```
e146b5f14e79d4935160c0e83fb9ebe526b8da0d commit refs/remotes/origin/HEAD  
e146b5f14e79d4935160c0e83fb9ebe526b8da0d commit refs/remotes/origin/master  
c6e1e95051d41771a649f3145423f8809d1a74d4 commit refs/replace/81a708dd0e167a3f691541c7a646334
```

## 凭证存储



```
$ git config --global credential.helper cache
```

```

        " store"          --file <path>
        " cache"          --timeout <seconds>
        " 900"           15           " store"

```

```
$ git config --global credential.helper store --file ~/.my-credentials
```

Git

Git  
Git

.gitconfig

## [credential]

```

helper = store --file /mnt/thumbdrive/.git-credentials
helper = cache --timeout 30000

```

Git

git credential

Git

mygithost

" fill"

```

$ git credential fill ❶
protocol=https ❷
host=mygithost
❸
protocol=https ❹
host=mygithost
username=bob
password=s3cre7
$ git credential fill ❺
protocol=https
host=unknownhost

Username for 'https://unknownhost': bob
Password for 'https://bob@unknownhost':
protocol=https
host=unknownhost

```

```
username=bob  
password=s3cre7
```

①

② Git-credential

③

④ Git-credential

⑤ Git

Git  
`credential.helper`

---

---

```
foo                               git-credential-foo  
foo -a --opt=bcd                 git-credential-foo -a --  
                                    opt=bcd  
/absolute/path/foo -xyz           /absolute/path/foo -xyz  
!f() { echo "pass- !"  
word=s3cre7"; }; f               shell
```

---

git-credential-cache git-  
credential-store

“ git-credential-foo [args] <action>.” / git-  
credential

- `get`
- `store`
- `erase`

store erase  
get Git

Git

git-credential  
git-credential-store:

```
$ git credential-store --file ~/git.store store ①
protocol=https
host=mygithost
username=bob
password=s3cre7
$ git credential-store --file ~/git.store get ②
protocol=https
host=mygithost

username=bob ③
password=s3cre7
```

|          |                                   |   |                        |
|----------|-----------------------------------|---|------------------------|
| <b>①</b> | <code>git-credential-store</code> | <code>https://mygi-</code>                |                        |
|          | <code>thost</code>                | <code>" bob"</code>                       | <code>" s3cre7"</code> |
| <b>②</b> | <code>git-credential-store</code> | <code>https://mygi-</code>                |                        |
|          | <code>thost</code>                |   |                        |
| <b>③</b> | <code>git-credential-store</code> |   |                        |
|          | <code>~/git.store</code>          |   |                        |
|          |                                   | <code>https://bob:s3cre7@mygithost</code> |                        |

|                  |                          |                       |
|------------------|--------------------------|-----------------------|
| <code>URL</code> | <code>osxkeychain</code> | <code>winstore</code> |
|                  | <code>cache</code>       |                       |

|                                   |                  |
|-----------------------------------|------------------|
| <code>git-credential-store</code> | <code>Git</code> |
| <code>Git</code>                  | <code>Git</code> |

1. `get` `store` `erase`

2. git-credential-store

3.

Ruby

Git

```
#!/usr/bin/env ruby

require 'optparse'

path = File.expand_path '~/.git-credentials' ❶
OptionParser.new do |opts|
  opts.banner = 'USAGE: git-credential-read-only [options] <action>'
  opts.on('-f', '--file PATH', 'Specify path for backing store') do |argpath|
    path = File.expand_path argpath
  end
end.parse!

exit(0) unless ARGV[0].downcase == 'get' ❷
exit(0) unless File.exists? path

known = {} ❸
while line = STDIN.gets
  break if line.strip == ''
  k,v = line.strip.split '=', 2
  known[k] = v
end

File.readlines(path).each do |fileline| ❹
  prot,user,pass,host = fileline.scan(/^(.*):(\/\/(.*):(.*?)@(.*)$)/).first
  if prot == known['protocol'] and host == known['host'] then
    puts "protocol=#{prot}"
    puts "host=#{host}"
    puts "username=#{user}"
    puts "password=#{pass}"
    exit(0)
  end
end

❶ ~/.git-credentials.
❷ get
```

**③**

known

**④**

known

git-credential-read-only

PATH

```
$ git credential-read-only --file=/mnt/shared/creds get
protocol=https
host=mygithost

protocol=https
host=mygithost
username=bob
password=s3cre7
```

" git-"

```
$ git config --global credential.helper read-only --file /mnt/shared/creds
```

## 总结

Git

# 自定义 Git 8

Git  
Git  
Git

## 配置 Git

Chapter 1 git config Git

```
$ git config --global user.name "John Doe"  
$ git config --global user.email johndoe@example.com
```

Git  
/etc/gitconfig  
--system git config  
  
Git  
~/.config/git/config  
--global Git  
  
Git  
.git/config  
  
.git/config /etc/gitconfig

---

Git 的配置文件是纯文本的，所以你可以直接手动编辑这些配置文件，输入合乎语法的值。但是运行 `git config` 命令会更简单些。

---

Git

—

Git

Git

```
$ man git-config
```

*http://git-scm.com/docs/git-config.html*

#### CORE.EDITOR

|             |          |          |
|-------------|----------|----------|
| Git         | \$VISUAL | \$EDITOR |
| vi          |          |          |
| core.editor |          |          |

```
$ git config --global core.editor emacs
```

Git      Emacs

#### COMMIT.TEMPLATE

```
~/.gitmessage.txt  
  
subject line  
  
what happened  
  
[ticket: X]
```

|     |                 |
|-----|-----------------|
| Git | git commit      |
|     | commit.template |

```
$ git config --global commit.template ~/.gitmessage.txt  
$ git commit
```

```
subject line

what happened

[ticket: X]
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
# modified:   lib/test.rb
#
~
~
".git/COMMIT_EDITMSG" 14L, 297C
```

## Git

### CORE.PAGER

|      |     |      |
|------|-----|------|
| Git  | log | diff |
| more |     | less |

```
$ git config --global core.pager ''
```

## Git

### USER.SIGNINGKEY

|     |     |    |
|-----|-----|----|
| GPG | " " | ID |
|-----|-----|----|

```
$ git config --global user.signingkey <gpg-key-id>
```

git tag

```
$ git tag -s <tag-name>
```

## CORE.EXCLUDESFILE

```

"          "          .gitignore
Git          git add

OS X           .DS_Store      Emacs   Vim
~              ~

          .gitignore
~/ .gitignore_global

*~
.DS_Store

... ...       git config --global core.excludesfile ~/ .gi-
tignore_global Git

```

## HELP.AUTOCORRECT

```
$ git chekcout master
git : 'chekcout' 不是一个 git 命令。参见 'git --help'。
```

您指的是这个么？  
checkout

|                |   |                         |
|----------------|---|-------------------------|
| Git<br>correct | 1 | help.autocorrect<br>Git |
|----------------|---|-------------------------|

```
$ git chekcout master
警告：您运行一个不存在的 git 命令 'chekcout'。继续执行假定您要要运行的是 'checkout'
在 0.1 秒钟后自动运行...
```

" 0.1 " help.autocorrect

50, Git

5

## Git

Git

COLOR.UI

Git

Git

```
$ git config --global color.ui false
```

auto

always

--color      Git

COLOR.\*

true    false    always

color.branch

color.diff

color.interactive

color.status

diff

```
$ git config --global color.diff.meta "blue black bold"
```

normal    black    red    green    yellow    blue

magenta    cyan    white

bold    dim    ul

blink    reverse



Code      Git  
        Git

diff.external

```
$ git config --global merge.tool extMerge
$ git config --global mergetool.extMerge.cmd \
  'extMerge \"$BASE\" \"$LOCAL\" \"$REMOTE\" \"$MERGED\"'
$ git config --global mergetool.extMerge.trustExitCode false
$ git config --global diff.external extDiff
```

~/.gitconfig

```
[merge]
  tool = extMerge
[mergetool "extMerge"]
  cmd = extMerge "$BASE" "$LOCAL" "$REMOTE" "$MERGED"
  trustExitCode = false
[diff]
  external = extDiff
```

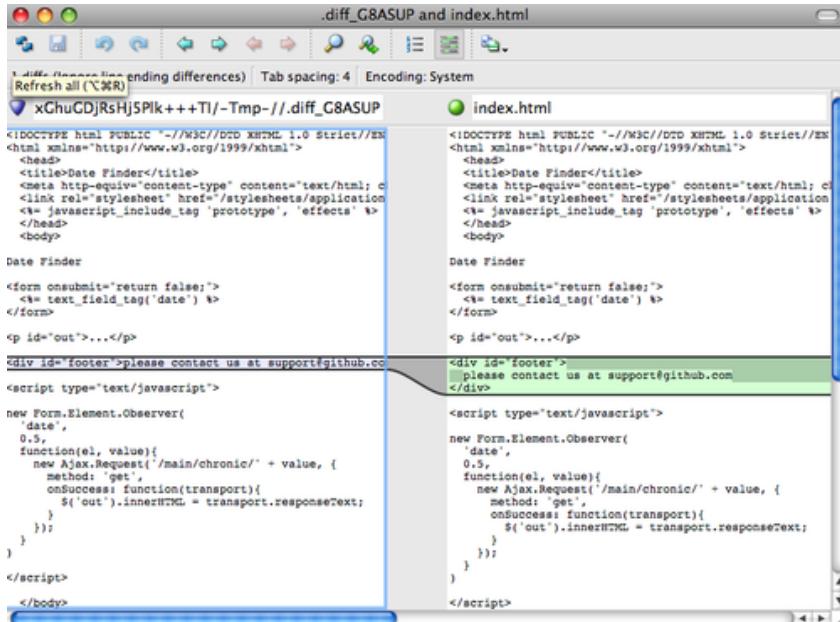
dif

```
$ git diff 32d1776b1^ 32d1776b1
```

Git      P4Merge

**FIGURE 8-1**

P4Merge.



git merge-  
 tool Git P4Merge  
 dif merge  
 extDiff extMerge KDiff3  
 extMerge

```
$ cat /usr/local/bin/extMerge
#!/bin/sh
/Applications/kdiff3.app/Contents/MacOS/kdiff3 $*
```

Git KDiff3  
 Git

```
$ git mergetool --tool-help
'git mergetool --tool=<tool>' may be set to one of the following:
  emerge
  gvimdiff
  gvimdiff2
  opendiff
  p4merge
```

```
vimdiff
vimdiff2
```

The following tools are valid, but not currently available:

```
araxis
bc3
codecompare
deltawalker
diffmerge
diffuse
ecmerge
kdiff3
meld
tkdiff
tortoisenmerge
xxdiff
```

Some of the tools listed above only work in a windowed environment. If run in a terminal-only session, they will fail.

KDif 3

kdif 3

```
$ git config --global merge.tool kdiff3
```

KDif 3

diff

extMerge extDiff Git

Windows

Git

## CORE.AUTOCRLF

Windows

CRLF

LF

Windows

Mac

Linux

CR

LF

Windows

Enter

Git

Windows

core.autocrlf  
true

```
$ git config --global core.autocrlf true
```

Linux Mac

Git

Git  
Git

core.autocrlf input

```
$ git config --global core.autocrlf input
```

Windows

Mac Linux

Windows  
false

Windows

```
$ git config --global core.autocrlf false
```

## CORE.WHITESPACE

Git

—

at-eof

blank-at-eol  
space-before-tabblank-  
tabtabwidth  
tab cr-at-eol  
core.whitespace

indent-with-non-tab

tab-in-indent

Git

-

cr-at-eol

```
$ git config --global core.whitespace \
trailing-space,space-before-tab,indent-with-non-tab
```

git diff

Git

git apply

Git

```
$ git apply --whitespace=warn <patch>
```

Git

```
$ git apply --whitespace=fix <patch>
```

git rebase

git rebase --whitespace=fix

Git

Git

## RECEIVE.FSCKOBJECTS

Git

SHA-1

Git

Git

receive.fsckObjects true

```
$ git config --system receive.fsckObjects true
```

Git

## RECEIVE.DENYNONFASTFORWARDS

push -f force-  
update force-pushes receive.de-  
nyNonFastForwards

```
$ git config --system receive.denyNonFastForwards true
```

non-fast-  
forwards

## RECEIVE.DENYDELETES

denyNonFastForwards receive.denyDeletes  
true

```
$ git config --system receive.denyDeletes true
```

—

ACL

u

## Git 属性

Git  
Git  
.gitattributes .git/info/  
attributes

Git

Git

Git      Git

Git

## 识别二进制文件

|       |      |            |     |
|-------|------|------------|-----|
| Xcode |      | .pbxproj   | Mac |
| IDE   | JSON | Javascript |     |
|       |      | UTF-8      |     |
|       |      |            | —   |
|       |      | dif        |     |

Git      pbxproj      .gitattributes

\*.pbxproj binary

|          |          |     |
|----------|----------|-----|
| Git      | CRLF     |     |
| git show | git diff | Git |

## 比较二进制文件

|          |                |           |
|----------|----------------|-----------|
| Word     | Microsoft Word | Microsoft |
|          | Word           |           |
| Git      |                |           |
| git diff |                |           |

```
$ git diff
diff --git a/chapter1.docx b/chapter1.docx
index 88839c4..4afcb7c 100644
Binary files a/chapter1.docx and b/chapter1.docx differ
```

```

Word          Git
.gitattributes

*.docx diff=word

Git           .docx
" word"      " word"
Git           docx2txt   Word
docx2txt     http://docx2txt.sourceforge.net
INSTALL
Git
docx2txt

```

```

#!/bin/bash
docx2txt.pl $1 -

```

```

chmod a+x
Git

```

```
$ git config diff.word.textconv docx2txt
```

```

Git           .docx
" word"      docx2txt   Word
Word         Git
git diff

```

```

$ git diff
diff --git a/chapter1.docx b/chapter1.docx
index 0b013ca..ba25db5 100644
--- a/chapter1.docx
+++ b/chapter1.docx
@@ -2,6 +2,7 @@
    This chapter will be about getting started with Git. We will begin at the beginning
    1.1. About Version Control
    What is "version control", and why should you care? Version control is a system to
    +Testing: 1, 2, 3.
    If you are a graphic or web designer and want to keep every version of an image or
    1.1.1. Local Version Control Systems
    Many people's version-control method of choice is to copy files into another direc

```

Git                                  " Testing: 1, 2, 3."

—  
EXIF                                  —  
exiftool

```
$ echo '*.* diff=exif' >> .gitattributes
$ git config diff.exif.textconv exiftool
```

git diff

```
diff --git a/image.png b/image.png
index 88839c4..4afcb7c 100644
--- a/image.png
+++ b/image.png
@@ -1,12 +1,12 @@
    ExifTool Version Number      : 7.74
-File Size                   : 70 kB
-File Modification Date/Time : 2009:04:21 07:02:45-07:00
+File Size                   : 94 kB
+File Modification Date/Time : 2009:04:21 07:02:43-07:00
    File Type                  : PNG
    MIME Type                  : image/png
-Image Width                 : 1058
-Image Height                : 889
+Image Width                 : 1056
+Image Height                : 827
    Bit Depth                  : 8
    Color Type                 : RGB with Alpha
```

SVN    CVS

keyword expansion

Git

Git

Git

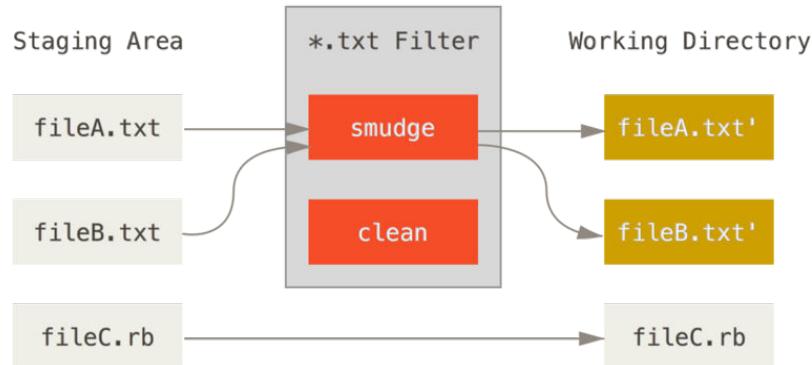
Git

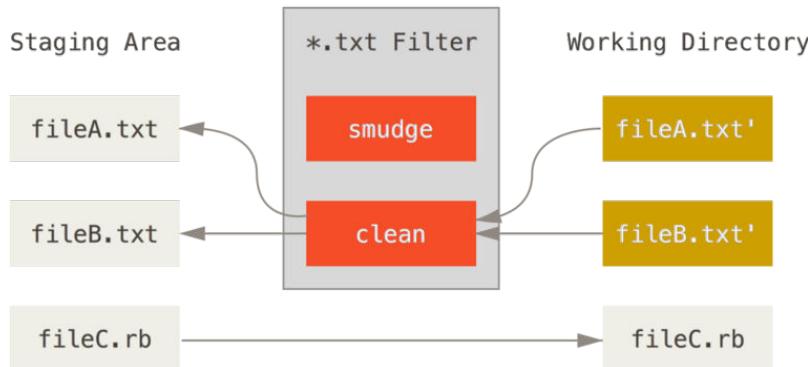
Git

Git

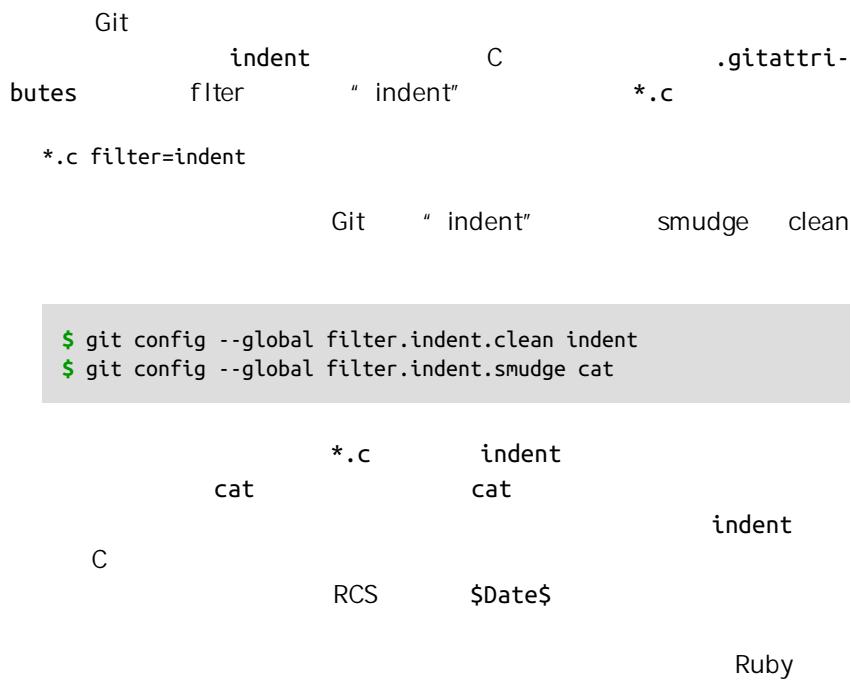
**FIGURE 8-2**

“smudge”过滤器会在文件被检出时触发



**FIGURE 8-3**

“clean”过滤器会在文件被暂存时触发



```
#!/usr/bin/env ruby
data = STDIN.read
last_date = `git log --pretty=format:"%ad" -1`
puts data.gsub('$Date$', '$Date: ' + last_date.to_s + '$')
```

```
git log
$Date$ —
```



```
test/ export-ignore
```

```
    git archive
```

EXPORT-SUBST

```
git log
export-subst
LAST_COMMIT
git archive
```

```
$ echo 'Last commit date: $Format:%cd by %aN$' > LAST_COMMIT
$ echo "LAST_COMMIT export-subst" >> .gitattributes
$ git add LAST_COMMIT .gitattributes
$ git commit -am 'adding LAST_COMMIT file for archives'
```

```
git archive
```

```
$ git archive HEAD | tar xcf ../deployment-testing -
$ cat ../deployment-testing/LAST_COMMIT
Last commit date: Tue Apr 21 08:38:48 2009 -0700 by Scott Chacon
```

git

git log

```
$ echo '$Format:Last commit: %h by %aN at %cd%w(76,6,9)%B$' > LAST_COMMIT
$ git commit -am 'export-subst 使用 git log 的自定义格式化工具'
```

git archive 直接使用 git log 的 `pretty=format:` 处理器，并在输出中移除两侧的 `\$Format:` 和 `\$` 标记。

```
$ git archive @ | tar xf0 - LAST_COMMIT
Last commit: 312ccc8 by Jim Hill at Fri May 8 09:14:04 2015 -0700
export-subst 使用 git log 的自定义格式化工具
```

git archive 直接使用 git log 的 `pretty=format:` 处理器，并在输出中移除两侧的 `\$Format:` 和 `\$` 标记。

Git

Git

database.xml

database.xml merge=ours

ours

```
$ git config --global merge.ours.driver true
```

database.xml

```
$ git merge topic
Auto-merging database.xml
Merge made by recursive.
```

database.xml

## Git 钩子

Git

Git                hooks  
.git/hooks        git init        Git

Perl              shell  
—



---

需要注意的是，克隆某个版本库时，它的客户端钩子 **并不** 随同复制。如果需要靠这些脚本来强制维持某种策略，建议你在服务器端实现这一功能。（请参照“使用强制策略的一个例子”中的例子。）

---

## 提交工作流钩子

```
pre-commit
```

```
          Git  
verify  
          git commit --no-  
          lint
```

```
prepare-commit-msg
```

```
          SHA-1
```

```
commit-msg
```

```
          Git
```

```
post-commit
```

```
          git log -1 HEAD
```

## 电子邮件工作流钩子



Git  
pre-auto-gc                          git gc --auto

## PRE-RECEIVE

pre-receive  
non-fast-forward

## UPDATE

update        pre-receive  
receive                              pre-  
                                       update  
                                       SHA-1  
SHA-1            update

## POST-RECEIVE

post-receive  
pre-receive  
continuous integration  
ticket-tracking system —  
ticket

## 使用强制策略的一个例子

Git

```

Ruby
Ruby
      Git          Perl    Bash
      hooks       update   up-
date
      •
      •           revision
      •           revision
      SSH
      " git"
shell
      $USER
update

#!/usr/bin/env ruby

$refname = ARGV[0]
$oldrev = ARGV[1]
$newrev = ARGV[2]
$user = ENV['USER']

puts "Enforcing Policies..."
puts "($refname) ($oldrev[0,6]) ($newrev[0,6])"

```

—

指定特殊的提交信息格式

" ref: 1234"

## ticketing system

|  |  |   |
|--|--|---|
| <code>\$newrev \$oldrev<br/>git log</code> |  | <code>git rev-list Git<br/>SHA-1<br/>SHA-1<br/>SHA-1</code> |
|--|--|---|

```
$ git rev-list 538c33..d14fc7  
d14fc7c847ab946ec39590d87783c69b031bdfb7  
9f585da4401b0a3999e84113824d15245c13f0be  
234071a1be950e2a8d078e6141f5cd20c1e61ad3  
dfa04c9ef3d5197182f13fb5b9b1fb7717d2222a  
17716ec0f1ff5c77eff40b7fe912f9f6cf0e475
```

SHA-1

|                                      |  |                       |
|--------------------------------------|--|-----------------------|
| <code>git cat-file Chapter 10</code> |  | <code>git cat-</code> |
|--------------------------------------|--|-----------------------|

```
$ git cat-file commit ca82a6  
tree cfda3bf379e4f8dba8717dee55aab78aef7f4daf  
parent 085bb3bc608e1e8451d4b2432f8ecbe6306e7e7  
author Scott Chacon <schacon@gmail.com> 1205815931 -0700  
committer Scott Chacon <schacon@gmail.com> 1240030591 -0700
```

changed the version number

|       |  |          |
|-------|--|----------|
| SHA-1 |  | Unix sed |
|-------|--|----------|

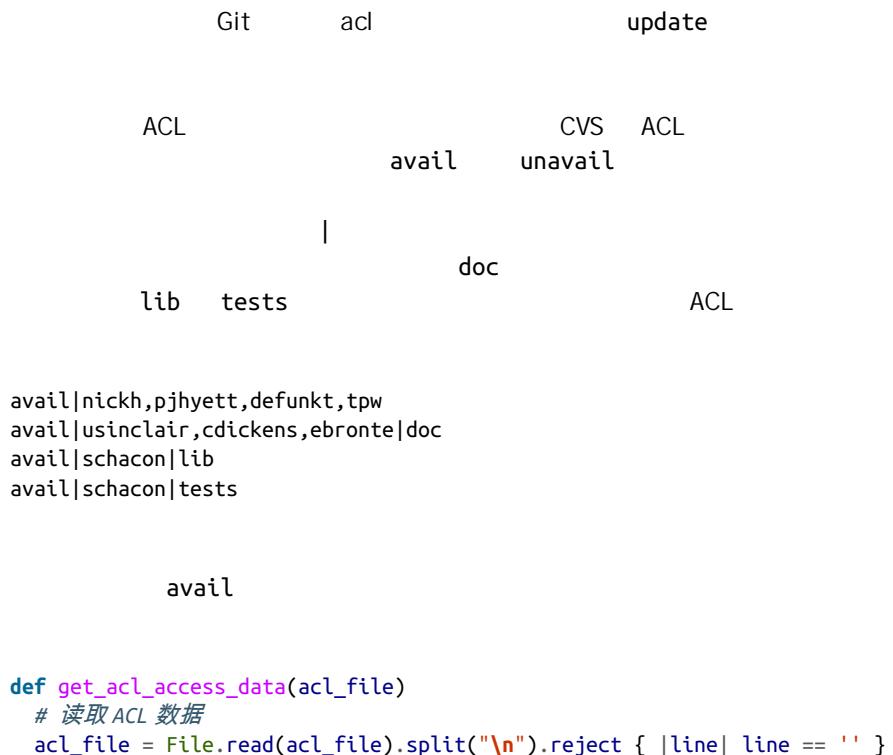
```
$ git cat-file commit ca82a6 | sed '1,/^\$/d'  
changed the version number
```

```
$regex = /\[ref: (\d+)\]/

# 指定自定义的提交信息格式
def check_message_format
  missed_revs = `git rev-list ${oldrev}..${newrev}`.split("\n")
  missed_revs.each do |rev|
    message = `git cat-file commit ${rev} | sed '1,/^\$/d'`
    if !$regex.match(message)
      puts "[POLICY] Your message is not formatted correctly"
      exit 1
    end
  end
end
check_message_format

update
```

## 指定基于用户的访问权限控制列表 (ACL) 系统



```

access = []
acl_file.each do |line|
  avail, users, path = line.split('||')
  next unless avail == 'avail'
  users.split(',').each do |user|
    access[user] ||= []
    access[user] << path
  end
end
access
end

```

ACL

get\_acl\_access\_data

```

{"defunkt"=>[nil],
 "tpw"=>[nil],
 "nickh"=>[nil],
 "pjhyett"=>[nil],
 "schacon"=>["lib", "tests"],
 "cdickens"=>["doc"],
 "usinclair"=>["doc"],
 "ebronte"=>["doc"]}

```

git log --name-only

```
$ git log -1 --name-only --pretty=format:'' 9f585d
```

```
README
lib/test.rb
```

get\_acl\_access\_data      ACL

:

```

# 仅允许特定用户修改项目中的特定子目录
def check_directory_perms
  access = get_acl_access_data('acl')

  # 检查是否有人在向他没有权限的地方推送内容
  new_commits = `git rev-list #{$oldrev}..#{$newrev}`.split("\n")
  new_commits.each do |rev|
    files_modified = `git log -1 --name-only --pretty=format:'' #{rev}`.split("\n")
    files_modified.each do |path|

```

```

next if path.size == 0
has_file_access = false
access[$user].each do |access_path|
  if !access_path # 用户拥有完全访问权限
    || (path.start_with? access_path) # 或者对此路径有访问权限
    has_file_access = true
  end
end
if !has_file_access
  puts "[POLICY] You do not have access to push to #{path}"
  exit 1
end
end
end
end

check_directory_perms

git rev-list

```

测试一下

|                      |         |
|----------------------|---------|
| .git/hooks/update    | chmod u |
| +x .git/hooks/update |         |

```

$ git push -f origin master
Counting objects: 5, done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 323 bytes, done.
Total 3 (delta 1), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
Enforcing Policies...
(refs/heads/master) (8338c5) (c5b616)
[POLICY] Your message is not formatted correctly
error: hooks/update exited with error code 1
error: hook declined to update refs/heads/master
To git@gitserver:project.git
  ! [remote rejected] master -> master (hook declined)
error: failed to push some refs to 'git@gitserver:project.git'

```

```
Enforcing Policies...
(refs/heads/master) (fb8c72) (c56860)
```

update

```
[POLICY] Your message is not formatted correctly
error: hooks/update exited with error code 1
error: hook declined to update refs/heads/master
```

Git update

```
To git@gitserver:project.git
! [remote rejected] master -> master (hook declined)
error: failed to push some refs to 'git@gitserver:project.git'
```

remote rejected

lib

```
[POLICY] You do not have access to push to lib/test.rb
```

update

.git/hooks

Git

commit-msg

## Git

```

#!/usr/bin/env ruby
message_file = ARGV[0]
message = File.read(message_file)

$regex = /\\[ref: (\d+)\]/

if !$regex.match(message)
  puts "[POLICY] Your message is not formatted correctly"
  exit 1
end

(.git/hooks/commit-msg)

```

```
$ git commit -am 'test'
[POLICY] Your message is not formatted correctly
```

## Git

```
$ git commit -am 'test [ref: 132]'
[master e05c914] test [ref: 132]
 1 file changed, 1 insertions(+), 0 deletions(-)
```

ACL   .git  
  pre-commit

```

#!/usr/bin/env ruby

$user      = ENV['USER']

# [ 插入上文中的 get_acl_access_data 方法 ]

# 仅允许特定用户修改项目中的特定子目录
def check_directory_perms
  access = get_acl_access_data('.git/acl')

  files_modified = `git diff-index --cached --name-only HEAD`.split("\n")
  files_modified.each do |path|
    next if path.size == 0
    has_file_access = false

```

```
access[$user].each do |access_path|
  if !access_path || (path.index(access_path) == 0)
    has_file_access = true
  end
  if !has_file_access
    puts "[POLICY] You do not have access to push to #{path}"
    exit 1
  end
end
end

check_directory_perms
```

ACL  
.git      ACL

```
access = get_acl_access_data('acl')
```

```
access = get_acl_access_data('.git/acl')
```

```
files_modified = `git log -1 --name-only --pretty=format:'' #{ref}`
```

```
files_modified = `git diff-index --cached --name-only HEAD`
```

—

\$user

non-fast-forward  
fast-forward

```
receive.denyDe-
letes    receive.denyNonFastForwards
```

pre-rebase

reachable

```

#!/usr/bin/env ruby

base_branch = ARGV[0]
if ARGV[1]
  topic_branch = ARGV[1]
else
  topic_branch = "HEAD"
end

target_shas = `git rev-list #{base_branch}..#{topic_branch}`.split("\n")
remote_refs = `git branch -r`.split("\n").map { |r| r.strip }

target_shas.each do |sha|
  remote_refs.each do |remote_ref|
    shas_pushed = `git rev-list ^#{sha}^@ refs/remotes/#{remote_ref}`
    if shas_pushed.split("\n").include?(sha)
      puts "[POLICY] Commit #{sha} has already been pushed to #{remote_ref}"
      exit 1
    end
  end
end

```

" "

`git rev-list ^#{sha}^@ refs/remotes/#{remote\_ref}`

SHA^@

SHA-1

—

—

-f

## 总结

Git

Git

# 9

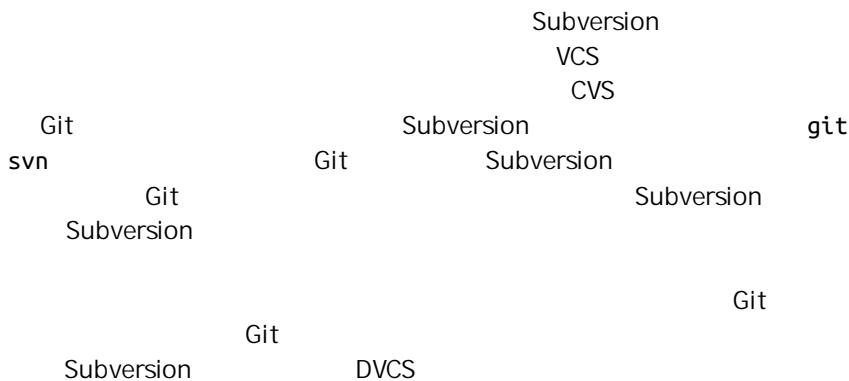
## Git 与其他系统

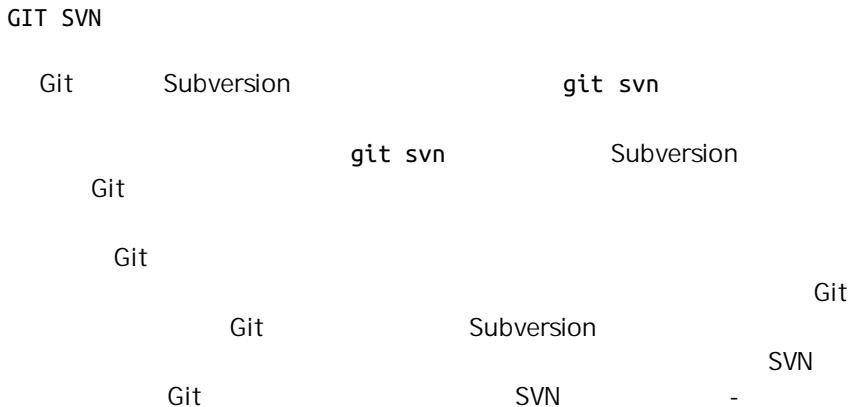


### 作为客户端的 Git

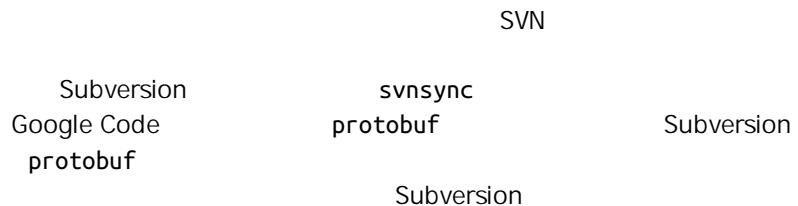


### Git Subversion





## 设置



```
$ mkdir /tmp/test-svn
$ svnadmin create /tmp/test-svn
```

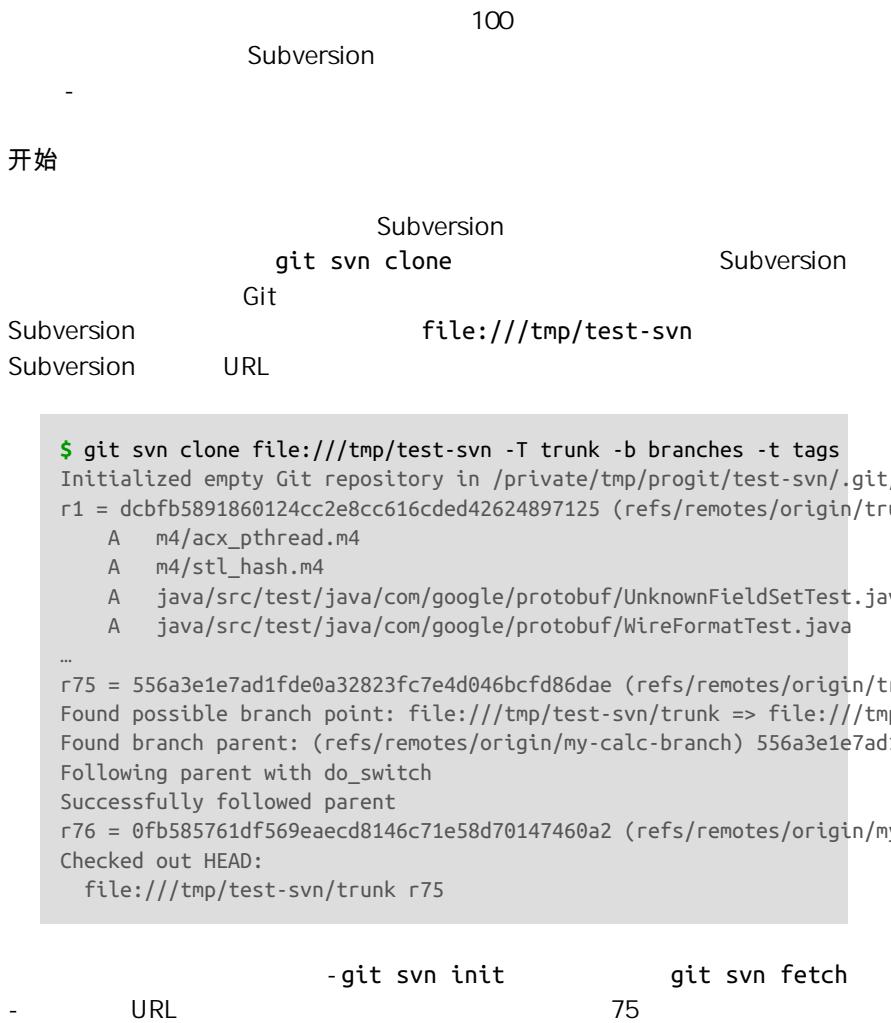
pre-revprop-change

```
$ cat /tmp/test-svn/hooks/pre-revprop-change
#!/bin/sh
exit 0;
$ chmod +x /tmp/test-svn/hooks/pre-revprop-change
```

svnsync init

```
$ svnsync init file:///tmp/test-svn \
http://progit-example.googlecode.com/svn/
```

```
$ svnsync sync file:///tmp/test-svn
Committed revision 1.
Copied properties for revision 1.
Transmitting file data .....[...]
Committed revision 2.
Copied properties for revision 2.
[...]
```



-T trunk -b branches -t tags                          Git Subversion

-S

```
$ git svn clone file:///tmp/test-svn -s
```

Git

```
$ git branch -a
* master
  remotes/origin/my-calc-branch
  remotes/origin/tags/2.0.2
  remotes/origin/tags/release-2.0.1
  remotes/origin/tags/release-2.0.2
  remotes/origin/tags/release-2.0.2rc1
  remotes/origin/trunk
```

Subversion  
Git                          show-ref

```
$ git show-ref
556a3e1e7ad1fde0a32823fc7e4d046bcfd86dae refs/heads/master
0fb585761df569eaecd8146c71e58d70147460a2 refs/remotes/origin/my-calc-branch
bfd2d79303166789fc73af4046651a4b35c12f0b refs/remotes/origin/tags/2.0.2
285c2b2e36e467dd4d91c8e3c0e1750b3fe8ca refs/remotes/origin/tags/release-2.0.1
cbda99cb45d9abcb9793db1d4f70ae562a969f1e refs/remotes/origin/tags/release-2.0.2
a9f074aa89e826d6f9d30808ce5ae3ffe711fed4 refs/remotes/origin/tags/release-2.0.2rc1
556a3e1e7ad1fde0a32823fc7e4d046bcfd86dae refs/remotes/origin/trunk
```

Git                  Git

```
$ git show-ref
c3dcbe8488c6240392e8a5d7553bbffcb0f94ef0 refs/remotes/origin/master
32ef1d1c7cc8c603ab78416262cc421b80a8c2df refs/remotes/origin/branch-1
75f703a3580a9b81ead89fe1138e6da858c5ba18 refs/remotes/origin/branch-2
23f8588dde934e8f33c263c6d8359b2ae095f863 refs/tags/v0.1.0
7064938bd5e7ef47bfd79a685a62c1e2649e2ce7 refs/tags/v0.2.0
6dcb09b5b57875f334f61aebed695e2e4193db5e refs/tags/v1.0.0
```

Git                    refs/tags

### 提交回 SUBVERSION

Git        SVN

Git

Subversion

```
$ git commit -am 'Adding git-svn instructions to the README'
[master 4af61fd] Adding git-svn instructions to the README
 1 file changed, 5 insertions(+)
```

    sion        -

Subversion

Subver  
Subversion

git svn dcommit

```
$ git svn dcommit
Committing to file:///tmp/test-svn/trunk ...
 M README.txt
Committed r77
 M README.txt
r77 = 95e0222ba6399739834380eb10afcd73e0670bc5 (refs/remotes/origin/trunk)
No changes between 4af61fd05045e07598c553167e0f31c84fd6ffe1 and refs/remotes/origin/trunk
Resetting to the latest refs/remotes/origin/trunk
```

Subversion  
Subversion

Git  
SHA-1  
Git

Subversion  
git-svn-id

```
$ git log -1
commit 95e0222ba6399739834380eb10afcd73e0670bc5
Author: ben <ben@0b684db3-b064-4277-89d1-21af03df0a68>
Date:   Thu Jul 24 03:08:36 2014 +0000
```

Adding git-svn instructions to the README

git-svn-id: file:///tmp/test-svn/trunk@77 0b684db3-b064-4277-89d1-21af03df0a68

|                         |                       |                      |
|-------------------------|-----------------------|----------------------|
| SHA-1<br><b>95e0222</b> | Git<br><b>dcommit</b> | Subver<br>Subversion |
|-------------------------|-----------------------|----------------------|

**拉取新改动****git svn**

```
$ git svn dcommit
Committing to file:///tmp/test-svn/trunk ...

ERROR from SVN:
Transaction is out of date: File '/trunk/README.txt' is out of date
W: d5837c4b461b7c0e018b49d12398769d2bfc240a and refs/remotes/origin/trunk differ,
:100644 100644 f414c433af0fd6734428cf9d2a9fd8ba00ada145 c80b6127dd04f5fcda218730dc
Current branch master is up to date.
ERROR: Not all changes have been committed into SVN, however the committed
ones (if any) seem to be successfully integrated into the working tree.
Please see the above messages for details.
```

**git svn rebase**

```
$ git svn rebase
Committing to file:///tmp/test-svn/trunk ...

ERROR from SVN:
Transaction is out of date: File '/trunk/README.txt' is out of date
W: eaa029d99f87c5c822c5c29039d19111ff32ef46 and refs/remotes/origin/trunk differ,
:100644 100644 65536c6e30d263495c17d781962cff12422693a b34372b25ccf4945fe5658fa38
First, rewinding head to replay your work on top of it...
Applying: update foo
Using index info to reconstruct a base tree...
M README.txt
Falling back to patching base and 3-way merge...
Auto-merging README.txt
ERROR: Not all changes have been committed into SVN, however the committed
ones (if any) seem to be successfully integrated into the working tree.
Please see the above messages for details.
```

Subversion  
**dcommit**

```
$ git svn dcommit
Committing to file:///tmp/test-svn/trunk ...
    M README.txt
Committed r85
    M README.txt
r85 = 9c29704cc0bbbed7bd58160cfb66cb9191835cd8 (refs/remotes/origin/trunk)
No changes between 5762f56732a958d6cfda681b661d2a239cc53ef5 and refs/remotes/origin/trunk
Resetting to the latest refs/remotes/origin/trunk
```

|                   |                          |
|-------------------|--------------------------|
| Git<br><b>svn</b> | <b>git</b><br>Subversion |
| <b>dcommit</b>    |                          |

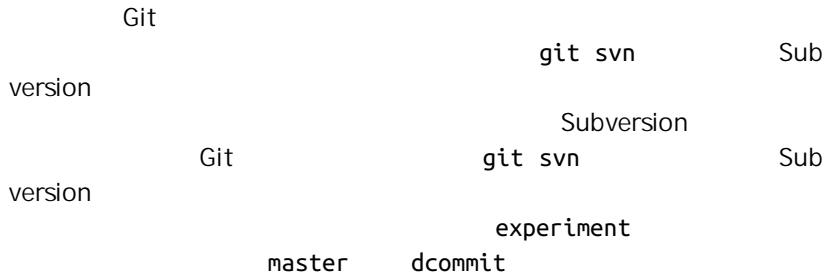
```
$ git svn dcommit
Committing to file:///tmp/test-svn/trunk ...
    M configure.ac
Committed r87
    M autogen.sh
r86 = d8450bab8a77228a644b7dc0e95977ffc61adff7 (refs/remotes/origin/trunk)
    M configure.ac
r87 = f3653ea40cb4e26b6281cec102e35dcba1fe17c4 (refs/remotes/origin/trunk)
W: a0253d06732169107aa020390d9fefdb2b1d92806 and refs/remotes/origin/trunk differ, using rebase
:100755 100755 efa5a59965fb5b2b0a12890f1b351bb5493c18 e757b59a9439312d80d5d43bb65d4a7d0389
First, rewinding head to replay your work on top of it...
```

|                   |       |                      |
|-------------------|-------|----------------------|
| Git<br><b>SVN</b> | - Git | Subversion           |
|                   |       | <b>git svn fetch</b> |
| <b>base</b>       |       | <b>git svn re-</b>   |

```
$ git svn rebase
    M autogen.sh
r88 = c9c5f83c64bd755368784b444bc7a0216cc1e17b (refs/remotes/origin/trunk)
First, rewinding head to replay your work on top of it...
Fast-forwarded master to refs/remotes/origin/trunk.
```

```
git svn rebase
```

```
git svn rebase
```

**GIT 分支问题**

```
$ git svn dcommit
Committing to file:///tmp/test-svn/trunk ...
M   CHANGES.txt
Committed r89
M   CHANGES.txt
r89 = 89d492c884ea7c834353563d5d913c6adf933981 (refs/remotes/origin/trunk)
M   COPYING.txt
M   INSTALL.txt
Committed r90
M   INSTALL.txt
M   COPYING.txt
r90 = cb522197870e61467473391799148f6721bcf9a0 (refs/remotes/origin/trunk)
No changes between 71af502c214ba13123992338569f4669877f55fd and refs/remotes/origin/trunk
Resetting to the latest refs/remotes/origin/trunk
```

|     |   |            |
|-----|---|------------|
| Git | - | dcommit    |
|     |   | experiment |
|     |   | SVN        |

```
git merge --squash
```

**SUBVERSION 分支**

|            |     |         |            |
|------------|-----|---------|------------|
| Subversion | Git | git svn | Subversion |
|------------|-----|---------|------------|

## 创建一个新的 SVN 分支

Subversion    `git svn branch [branch-name]`

```
$ git svn branch opera
Copying file:///tmp/test-svn/trunk at r90 to file:///tmp/test-svn/branches/opera...
Found possible branch point: file:///tmp/test-svn/trunk => file:///tmp/test-svn/branches/opera
Found branch parent: (refs/remotes/origin/opera) cb522197870e61467473391799148f6721bcf9a0
Following parent with do_switch
Successfully followed parent
r91 = f1b64a3855d3c8dd84ee0ef10fa89d27f1584302 (refs/remotes/origin/opera)
```

Subversion    `svn copy trunk branches/opera`  
 Subversion    `trunk`                                  `opera`  
 era

## 切换活动分支

Git    Subversion  
 -  
`svn-id`

Subversion    `dcommit`                                  Subversion  
 opera

```
$ git branch opera remotes/origin/opera
```

Subversion    `opera`                                  `trunk`                                  `master`  
`git merge`                                         `-m`  
    " Merge branch opera"  
`git merge`

Subversion    `Git`

Subversion    `Git`

Subversion    `Subversion`

`Git`

`dcommit`

`- dcommit`

`git merge`                                         `git merge --squash`

- Subversion

`opera`

### SUBVERSION 命令

`git svn`

Git

Subversion

Subversion

SVN

Subversion

`git svn log`

SVN

SVN

```
$ git svn log
```

```
r87 | schacon | 2014-05-02 16:07:37 -0700 (Sat, 02 May 2014) | 2 lines
```

```
autogen change
```

```
r86 | schacon | 2014-05-02 16:00:21 -0700 (Sat, 02 May 2014) | 2 lines
```

```
Merge branch 'experiment'
```

```
r85 | schacon | 2014-05-02 16:00:09 -0700 (Sat, 02 May 2014) | 2 lines
```

```
updated the changelog
```

`git svn log`

`svn log`

Subversion

Subversion

dcommit

Git

Subversion

Subversion

SVN

`git svn log`

`svn log`

`git svn`

`blame [FILE]`

`svn annotate`

```
$ git svn blame README.txt
```

```
2 temporal Protocol Buffers - Google's data interchange format
```

```
2 temporal Copyright 2008 Google Inc.
```

```
2 temporal http://code.google.com/apis/protocolbuffers/
```

```
2 temporal
```

```

22 temporal C++ Installation - Unix
22 temporal =====
2 temporal
79 schacon Committing in git-svn.
78 schacon
2 temporal To build and install the C++ Protocol Buffer runtime and the Protocol
2 temporal Buffer compiler (protoc) execute the following:
2 temporal

```

## Git

Subversion  
SVN

`git svn info`      `svn info`

```

$ git svn info
Path: .
URL: https://schacon-test.googlecode.com/svn/trunk
Repository Root: https://schacon-test.googlecode.com/svn
Repository UUID: 4c93b258-373f-11de-be05-5f7a86268029
Revision: 87
Node Kind: directory
Schedule: normal
Last Changed Author: schacon
Last Changed Rev: 87
Last Changed Date: 2009-05-02 16:07:37 -0700 (Sat, 02 May 2009)

```

## Subversion

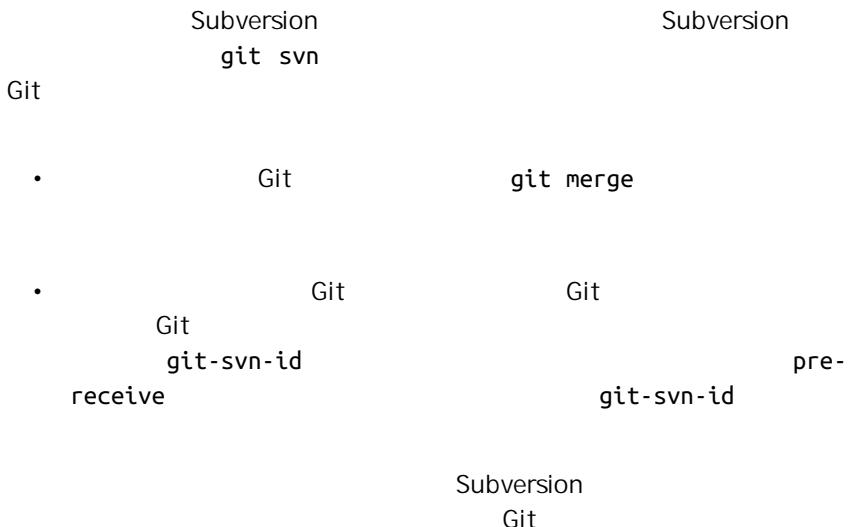
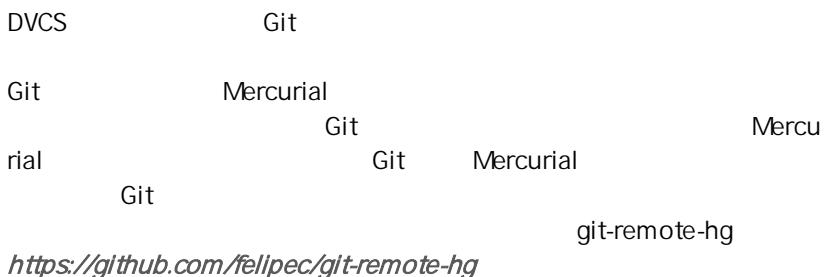
`blame`    `log`  
Subversion

|  |  |
|--|--|
| <code>svn:ignore</code><br><code>.gitignore</code><br><code>git svn</code><br><code>svn create-ignore</code> | <b>Subversion</b><br><code>git</code><br><code>.gitignore</code> |
|--|--|

`git svn show-ignore`      `.gitignore`

```
$ git svn show-ignore > .git/info/exclude
```

|   |   |
|---|---|
| <code>.gitignore</code><br><code>Subver</code><br><code>sion</code> | <code>Git</code><br><code>.gitignore</code> |
|---|---|

**GIT-SVN 总结****Git Mercurial****GIT-REMOTE-HG**

```

git-remote-hg PATH
$ curl -o ~/bin/git-remote-hg \
  https://raw.githubusercontent.com/felipec/git-remote-hg/master/git-remote-hg
$ chmod +x ~/bin/git-remote-hg
  
```

|   |  |   |                    |
|---|--|---|--------------------|
| <code>~/bin</code><br><code>curial</code> | <code>\$PATH</code><br><code>Python</code> | <code>Git-remote-hg</code><br><code>Python</code> | <code>mer -</code> |
|---|--|---|--------------------|

```
$ pip install mercurial
```

Python      <https://www.python.org/>  
Mercurial  
<http://mercurial.selenic.com/>

Mer  
curial                  Mercurial  
Mercurial    " hello world"

```
$ hg clone http://selenic.com/repo/hello /tmp/hello
```

## 开始

" server-side"

## Git

```
$ git clone hg::/tmp/hello /tmp/hello-git
```

```
$ cd /tmp/hello-git
```

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

```
* ac7955c (HEAD, origin/master, origin/branches/default, origin/HEAD, refs/hg/origin/branches/default) Create a standard "hello, world" program
* 65bb417 Create a standard "hello, world" program
```

Mercurial                  git clone  
git-remote-hg                GitHTTP/S  
                              Git    Mercurial

## log

.git

```
$ tree .git/refs
.git/refs
├── heads
│   └── master
└── hg
    └── origin
        ├── bookmarks
        │   └── master
        └── branches
            └── default
```

```

├── notes
│   └── hg
├── remotes
│   └── origin
│       └── HEAD
└── tags

9 directories, 5 files

```

| Git-remote-hg | Git                                   |
|---------------|---------------------------------------|
|               | refs/hg                               |
|               | refs/hg/origin/branches/default       |
| " ac7955c"    | SHA-1      Git      master            |
| refs/hg       | refs/remotes/origin                   |
| notes/hg      | git-remote-hg      Git      Mercurial |
| ID            |                                       |

```

$ cat notes/hg
d4c10386...

$ git cat-file -p d4c10386...
tree 1781c96...
author remote-hg <> 1408066400 -0800
committer remote-hg <> 1408066400 -0800

Notes for master

$ git ls-tree 1781c96...
100644 blob ac9117f... 65bb417...
100644 blob 485e178... ac7955c...

$ git cat-file -p ac9117f
0a04b987be5ae354b710cefeba0e2d9de7ad41a9

```

| refs/notes/hg | Git                              |
|---------------|----------------------------------|
|               | git ls-tree      tree            |
|               | tree                             |
|               | " ac9117f"      blob      master |
| SHA-1         | " 0a04b98"      default      Mer |
| curial        | ID                               |
|               | Git                              |



```
$ cp .hgignore .git/info/exclude
```

|                   |            |
|-------------------|------------|
| .git/info/exclude | .gitignore |
|-------------------|------------|

## 工作流程

master

```
$ git log --oneline --graph --decorate
* ba04a2a (HEAD, master) Update makefile
* d25d16f Goodbye
* ac7955c (origin/master, origin/branches/default, origin/HEAD, refs/hg/origin/branches/default)
* 65bb417 Create a standard "hello, world" program
```

|        |               |
|--------|---------------|
| master | origin/master |
|--------|---------------|

```
$ git fetch
From hg:///tmp/hello
  ac7955c..df85e87  master      -> origin/master
  ac7955c..df85e87  branches/default -> origin/branches/default
$ git log --oneline --graph --decorate --all
* 7b07969 (refs/notes/hg) Notes for default
* d4c1038 Notes for master
* df85e87 (origin/master, origin/branches/default, origin/HEAD, refs/hg/origin/branches/default)
| * ba04a2a (HEAD, master) Update makefile
| * d25d16f Goodbye
|/
* ac7955c Create a makefile
* 65bb417 Create a standard "hello, world" program
```

|                   |               |         |
|-------------------|---------------|---------|
| --all<br>" notes" | git-remote-hg | origin/ |
|-------------------|---------------|---------|

master

Mercurial

```
$ git merge origin/master
Auto-merging hello.c
Merge made by the 'recursive' strategy.
 hello.c | 2 ++
 1 file changed, 1 insertion(+), 1 deletion(-)
$ git log --oneline --graph --decorate
*   0c64627 (HEAD, master) Merge remote-tracking branch 'origin/master'
|\ \
| * df85e87 (origin/master, origin/branches/default, origin/HEAD, refs/hg/origin/b
* | ba04a2a Update makefile
* | d25d16f Goodbye
|/
* ac7955c Create a makefile
* 65bb417 Create a standard "hello, world" program
```

```
$ git push
To hg::/tmp/hello
 df85e87..0c64627  master -> master
```

Mercurial

```
$ hg log -G --style compact
o 5[tip]:4,2 dc8fa4f932b8 2014-08-14 19:33 -0700 ben
|\ Merge remote-tracking branch 'origin/master'
|
| o 4 64f27bcefc35 2014-08-14 19:27 -0700 ben
| | Update makefile
|
| o 3:1 4256fc29598f 2014-08-14 19:27 -0700 ben
| | Goodbye
|
@ | 2 7db0b4848b3c 2014-08-14 19:30 -0700 ben
|/ Add some documentation
|
o 1 82e55d328c8c 2005-08-26 01:21 -0700 mpm
| Create a makefile
|
```

```
o 0 0a04b987be5a 2005-08-26 01:20 -0700 mpm
  Create a standard "hello, world" program
```

|                    |           |          |   |
|--------------------|-----------|----------|---|
| 2<br>git-remote-hg | Mercurial | 3<br>Git | 4 |
|--------------------|-----------|----------|---|

## 分支与书签

|                |             |     |
|----------------|-------------|-----|
| Git            | Mercurial   |     |
|                | " bookmark" |     |
| Mercurial      | " branch"   | Git |
| <b>develop</b> |             |     |

```
$ hg log -l 1
changeset: 6:8f65e5e02793
branch: develop
tag: tip
user: Ben Straub <ben@straub.cc>
date: Thu Aug 14 20:06:38 2014 -0700
summary: More documentation
```

|           |           |     |      |
|-----------|-----------|-----|------|
| " branch" | Git       |     |      |
|           |           | Git |      |
| remote-hg | Mercurial |     | git- |
| Mercurial | Git       | Git |      |

```
$ git checkout -b featureA
Switched to a new branch 'featureA'
$ git push origin featureA
To hg::/tmp/hello
 * [new branch]      featureA -> featureA
```

Mercurial

```
$ hg bookmarks
  featureA          5:bd5ac26f11f9
$ hg log --style compact -G
@ 6[tip] 8f65e5e02793 2014-08-14 20:06 -0700 ben
| More documentation
|
o 5[featureA]:4,2 bd5ac26f11f9 2014-08-14 20:02 -0700 ben
|\ Merge remote-tracking branch 'origin/master'
```

```

| |
| o 4 0434aaa6b91f 2014-08-14 20:01 -0700 ben
| | update makefile
|
| o 3:1 318914536c86 2014-08-14 20:00 -0700 ben
| | goodbye
|
o | 2 f098c7f45c4f 2014-08-14 20:01 -0700 ben
|/ Add some documentation
|
o 1 82e55d328c8c 2005-08-26 01:21 -0700 mpm
| Create a makefile
|
o 0 0a04b987be5a 2005-08-26 01:20 -0700 mpm
| Create a standard "hello, world" program

```



```

$ git checkout -b branches/permanent
Switched to a new branch 'branches/permanent'
$ vi Makefile
$ git commit -am 'A permanent change'
$ git push origin branches/permanent
To hg::/tmp/hello
 * [new branch] branches/permanent -> branches/permanent

```

## Mercurial

```

$ hg branches
permanent                               7:a4529d07aad4
develop                                6:8f65e5e02793
default                                 5:bd5ac26f11f9 (inactive)
$ hg log -G
o changeset: 7:a4529d07aad4
| branch: permanent
| tag: tip
| parent: 5:bd5ac26f11f9
| user: Ben Straub <ben@straub.cc>
| date: Thu Aug 14 20:21:09 2014 -0700
| summary: A permanent change
|
| @ changeset: 6:8f65e5e02793

```

```

|/ branch:      develop
| user:        Ben Straub <ben@straub.cc>
| date:        Thu Aug 14 20:06:38 2014 -0700
| summary:     More documentation
|
o  changeset:  5:bd5ac26f11f9
|\ bookmark:   featureA
| | parent:    4:0434aaa6b91f
| | parent:    2:f098c7f45c4f
| | user:      Ben Straub <ben@straub.cc>
| | date:      Thu Aug 14 20:02:21 2014 -0700
| | summary:   Merge remote-tracking branch 'origin/master'
[...]

```

“ permanent”

7

Git

Mercurial

Mercurial

```

$ hg log --style compact -G
o 10[tip]  99611176cbc9  2014-08-14 20:21 -0700  ben
| A permanent change
|
o 9   f23e12f939c3  2014-08-14 20:01 -0700  ben
| Add some documentation
|
o 8:1  c16971d33922  2014-08-14 20:00 -0700  ben
| goodbye
|
| o 7:5  a4529d07aad4  2014-08-14 20:21 -0700  ben
| | A permanent change
|
| |
| | @ 6   8f65e5e02793  2014-08-14 20:06 -0700  ben
| | / More documentation
|
| |
| o 5[featureA]:4,2  bd5ac26f11f9  2014-08-14 20:02 -0700  ben
| |\ Merge remote-tracking branch 'origin/master'
| |
| | o 4   0434aaa6b91f  2014-08-14 20:01 -0700  ben
| | | update makefile
| |
+--o 3:1  318914536c86  2014-08-14 20:00 -0700  ben
| | goodbye
|
| |
| o 2   f098c7f45c4f  2014-08-14 20:01 -0700  ben
| / Add some documentation

```

```

| 
o 1  82e55d328c8c  2005-08-26 01:21 -0700  mpm
|   Create a makefile
|
o 0  0a04b987be5a  2005-08-26 01:20 -0700  mpm
    Create a standard "hello, world" program

```

8 9 10

permanent

Mercurial

**MERCURIAL 总结**

Git Mercurial

Mercurial

**Git Perforce**

Perforce

1995

**GIT FUSION**

Perforce

*[www.perforce.com/git-fusion](http://www.perforce.com/git-fusion)*

Git

Git Fusion

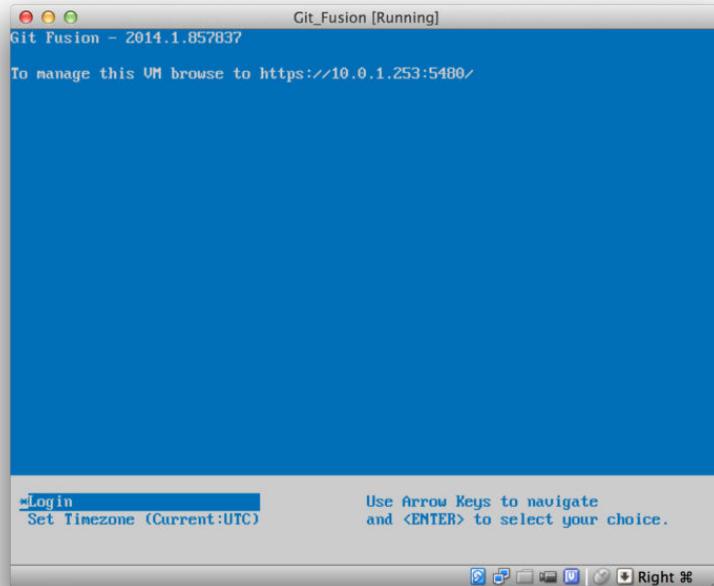
*<http://>*

Perforce



force git

Linux root perf-



**FIGURE 9-1**

Git Fusion 虚拟机启动屏幕。

IP  
Perforce  
SSH

" Login"

root

```
$ p4 -p localhost:1666 -u super user -f john
$ p4 -p localhost:1666 -u john passwd
$ exit
```

VI  
:wq

|       |     |                     |
|-------|-----|---------------------|
| Git   | SSL | Git Fusion          |
|       | IP  | Perforce Git Fusion |
| HTTPS |     |                     |

```
$ export GIT_SSL_NO_VERIFY=true
```

```
$ git clone https://10.0.1.254/Talkhouse
Cloning into 'Talkhouse'...
Username for 'https://10.0.1.254': john
Password for 'https://john@10.0.1.254':
remote: Counting objects: 630, done.
remote: Compressing objects: 100% (581/581), done.
remote: Total 630 (delta 172), reused 0 (delta 0)
Receiving objects: 100% (630/630), 1.22 MiB | 0 bytes/s, done.
Resolving deltas: 100% (172/172), done.
Checking connectivity... done.
```

|        |            |          |         |
|--------|------------|----------|---------|
| john   | HTTPS      | Git      |         |
| Fusion | Git Fusion | Perforce | //.git- |
|        |            | Perforce | fusion  |

```
$ tree
.
├── objects
│   └── repos
│       └── [...]
│   └── trees
│       └── [...]
|
└── p4gf_config
└── repos
    └── Talkhouse
        └── p4gf_config
└── users
    └── p4gf_usermap
```

```
498 directories, 287 files
```

objects      GitFusion

Perforce      Git

p4gf\_config

Git Fusion

-

**[repo-creation]**

charset = utf8

**[git-to-perforce]**

change-owner = author

enable-git-branch-creation = yes

enable-swarm-reviews = yes

enable-git-merge-commits = yes

enable-git-submodules = yes

preflight-commit = none

ignore-author-permissions = no

read-permission-check = none

git-merge-avoidance-after-change-num = 12107

**[perforce-to-git]**

http-url = none

ssh-url = none

**[@features]**

imports = False

chunked-push = False

matrix2 = False

parallel-push = False

**[authentication]**

email-case-sensitivity = no

INI

Git

repos/Talkhouse/p4gf\_config

[@repo]

**[Talkhouse-master]**

git-branch-name = master

view = //depot/Talkhouse/main-dev/... ...

Perforce

Git

git-branch-name

```
Git                               view
                                Perforce      Git
```

```
[multi-project-mapping]
git-branch-name = master
view = //depot/project1/main/... project1/...
//depot/project2/mainline/... project2/...
```

```
Git
      users/p4gf_usermap   Perforce
      Git Fusion           Perforce
      Git     author/committer
      Git     author          Perforce
```

```
john john@example.com "John Doe"
john johnny@appleseed.net "John Doe"
bob employeeX@example.com "Anon X. Mouse"
joe employeeY@example.com "Anon Y. Mouse"
```

```
<user> <email> "<full name>"           Perforce
                                         Git
                                         Perforce
                                         Git
                                         Git
                                         Git
                                         Git
```

```
Perforce Git Fusion      Perforce  Git
      Git                           " Jam"
```

```
$ git clone https://10.0.1.254/Jam
Cloning into 'Jam'...
Username for 'https://10.0.1.254': john
Password for 'https://ben@10.0.1.254':
remote: Counting objects: 2070, done.
remote: Compressing objects: 100% (1704/1704), done.
```

```

Receiving objects: 100% (2070/2070), 1.21 MiB | 0 bytes/s, done.
remote: Total 2070 (delta 1242), reused 0 (delta 0)
Resolving deltas: 100% (1242/1242), done.
Checking connectivity... done.
$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/master
  remotes/origin/rel2.1
$ git log --oneline --decorate --graph --all
* 0a38c33 (origin/rel2.1) Create Jam 2.1 release branch.
| * d254865 (HEAD, origin/master, origin/HEAD, master) Upgrade to latest metrowerks on Beos
| * bd2f54a Put in fix for jam's NT handle leak.
| * c0f29e7 Fix URL in a jam doc
| * cc644ac Radstone's lynx port.
[...]

```

## Git Fusion

Perforce

Git



```

# ...
$ git log --oneline --decorate --graph --all
* cfd46ab (HEAD, master) Add documentation for new feature
* a730d77 Whitespace
* d254865 (origin/master, origin/HEAD) Upgrade to latest metrowerks on Beos -- the Intel one
* bd2f54a Put in fix for jam's NT handle leak.
[...]

```

```

$ git fetch
remote: Counting objects: 5, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
From https://10.0.1.254/Jam
  d254865..6afeb15  master      -> origin/master
$ git log --oneline --decorate --graph --all
* 6afeb15 (origin/master, origin/HEAD) Update copyright
| * cfd46ab (HEAD, master) Add documentation for new feature

```

```
| * a730d77 Whitespace
|/
* d254865 Upgrade to latest metrowerks on Beos -- the Intel one.
* bd2f54a Put in fix for jam's NT handle leak.
[...]
```

6afeb15

Perforce

Git

Perforce

```
$ git merge origin/master
Auto-merging README
Merge made by the 'recursive' strategy.
 README | 2 ++
 1 file changed, 1 insertion(+), 1 deletion(-)
$ git push
Counting objects: 9, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 917 bytes | 0 bytes/s, done.
Total 9 (delta 6), reused 0 (delta 0)
remote: Perforce: 100% (3/3) Loading commit tree into memory...
remote: Perforce: 100% (5/5) Finding child commits...
remote: Perforce: Running git fast-export...
remote: Perforce: 100% (3/3) Checking commits...
remote: Perforce: Processing will continue even if connection is closed.
remote: Perforce: 100% (3/3) Copying changelists...
remote: Perforce: Submitting new Git commit objects to Perforce: 4
To https://10.0.1.254/Jam
 6afeb15..89cba2b master -> master
```

Git

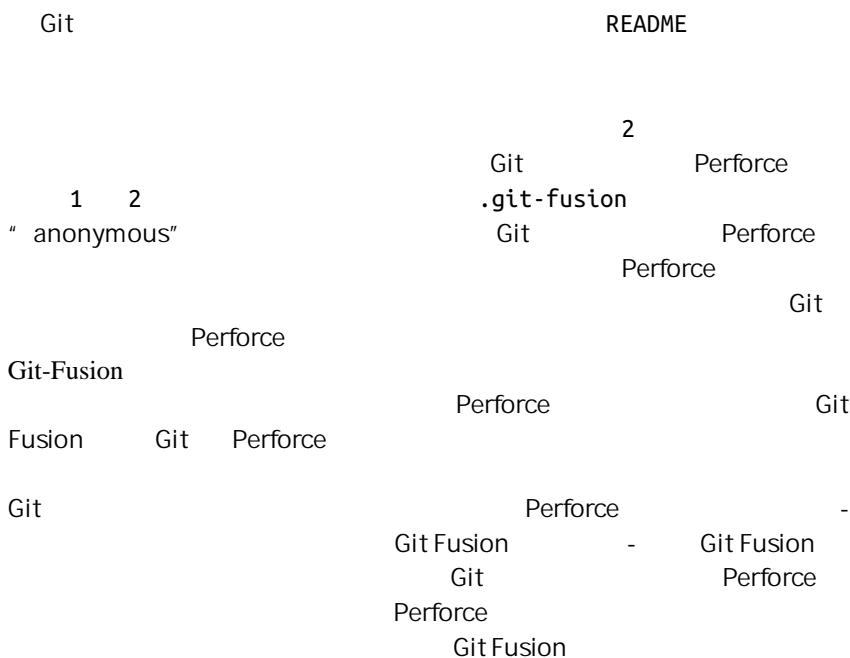
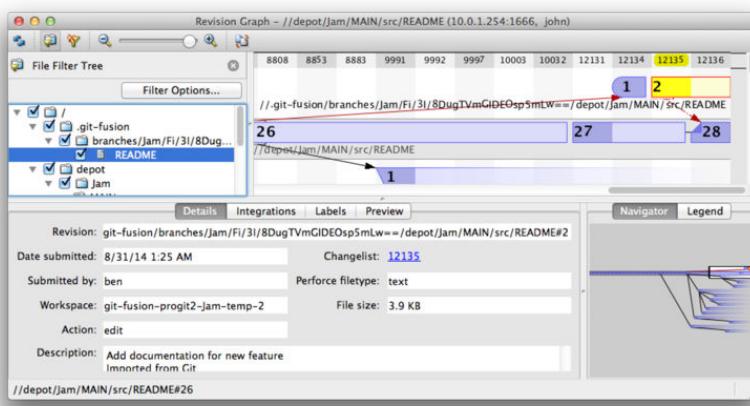
Perforce

README

p4v

**FIGURE 9-2**

Git 推送后的 Perforce 版本图



**GIT-P4**


---

为了与 git-p4 一起工作需要在你的 PATH 环境变量中的某个目录中有 p4 工具。在写这篇文章的时候，它可以在 <http://www.perforce.com/downloads/Perforce/zo-User> 免费获得。

---



```
$ export P4PORT=10.0.1.254:1666
$ export P4USER=john
```

**Git**

```
$ git p4 clone //depot/www/live www-shallow
Importing from //depot/www/live into www-shallow
Initialized empty Git repository in /private/tmp/www-shallow/.git/
Doing initial import of //depot/www/live/ from revision #head into refs/remotes/p4
```

**Git**

```
$ cd myproject
$ git log --oneline --all --graph --decorate
* 70eaf78 (HEAD, p4/master, p4/HEAD, master) Initial import of //depot/www/live/ f
```

" p4"              Perforce

```
$ git remote -v
```

Git-p4

git log

Git

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

- \* 018467c (HEAD, master) Change page title

- \* c0fb617 Update link

- \* 70eaf78 (p4/master, p4/HEAD) Initial import of //depot/www/live/ from the state at revision

Perforce

```
$ git p4 sync
```

git p4 sync

Performing incremental import into refs/remotes/p4/master git branch

Depot paths: //depot/www/live/

Import destination: refs/remotes/p4/master

Importing revision 12142 (100%)

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

- \* 75cd059 (p4/master, p4/HEAD) Update copyright

- | \* 018467c (HEAD, master) Change page title

- | \* c0fb617 Update link

- ||

- \* 70eaf78 Initial import of //depot/www/live/ from the state at revision #head

master p4/master

Git

Perforce

Git-p4

```
$ git p4 rebase
```

Performing incremental import into refs/remotes/p4/master git branch

Depot paths: //depot/www/live/

No changes to import!

Rebasing the current branch onto remotes/p4/master

First, rewinding head to replay your work on top of it...

Applying: Update link

Applying: Change page title

```
index.html | 2 +-  
1 file changed, 1 insertion(+), 1 deletion(-)  
  
git p4 rebase    git p4 sync      git re-  
base p4/master
```

|              |                           |           |        |
|--------------|---------------------------|-----------|--------|
| force<br>Git | git p4 submit<br>Perforce | p4/master | master |
|--------------|---------------------------|-----------|--------|

```
# A Perforce Change Specification.  
#  
# Change:      The change number. 'new' on a new changelist.  
# Date:        The date this specification was last modified.  
# Client:      The client on which the changelist was created. Read-only.  
# User:        The user who created the changelist.  
# Status:      Either 'pending' or 'submitted'. Read-only.  
# Type:        Either 'public' or 'restricted'. Default is 'public'.  
# Description: Comments about the changelist. Required.  
# Jobs:        What opened jobs are to be closed by this changelist.  
#               You may delete jobs from this list. (New changelists only.)  
# Files:       What opened files from the default changelist are to be added  
#               to this changelist. You may delete files from this list.  
#               (New changelists only.)  
  
Change:  new  
  
Client:  john_bens-mbp_8487  
  
User:   john  
  
Status:  new  
  
Description:  
  Update link  
  
Files:  
  //depot/www/live/index.html  # edit  
  
##### git author ben@straub.cc does not match your p4 account.  
##### Use option --preserve-user to modify authorship.  
##### Variable git-p4.skipUserNameCheck hides this message.  
##### everything below this line is just the diff #####  
--- //depot/www/live/index.html  2014-08-31 18:26:05.000000000 0000  
+++ /Users/ben/john_bens-mbp_8487/john_bens-mbp_8487/depot/www/live/index.html 2014-08-31 18:26:05.000000000 0000
```

```
@@ -60,7 +60,7 @@
 </td>
 <td valign=top>
 Source and documentation for
-<a href="http://www.perforce.com/jam/jam.html">
+<a href="jam.html">
 Jam/MR</a>,
 a software build tool.
 </td>
```



```
$ git p4 submit
Perforce checkout for depot path //depot/www/live/ located at /Users/ben/john_bens-mbp_8487/
Synchronizing p4 checkout...
... - file(s) up-to-date.
Applying dbac45b Update link
//depot/www/live/index.html#4 - opened for edit
Change 12143 created with 1 open file(s).
Submitting change 12143.
Locking 1 files ...
edit //depot/www/live/index.html#5
Change 12143 submitted.
Applying 905ec6a Change page title
//depot/www/live/index.html#5 - opened for edit
Change 12144 created with 1 open file(s).
Submitting change 12144.
Locking 1 files ...
edit //depot/www/live/index.html#6
Change 12144 submitted.
All commits applied!
Performing incremental import into refs/remotes/p4/master git branch
Depot paths: //depot/www/live/
Import destination: refs/remotes/p4/master
Importing revision 12144 (100%)
Rebasing the current branch onto remotes/p4/master
First, rewinding head to replay your work on top of it...
$ git log --oneline --all --graph --decorate
* 775a46f (HEAD, p4/master, p4/HEAD, master) Change page title
* 05f1ade Update link
```

```
* 75cd059 Update copyright
* 70eaf78 Initial import of //depot/www/live/ from the state at revision #head
```

**git push**

Git

Perforce

**git p4 submit**

SHA-1

git-p4

```
$ git log -1
commit 775a46f630d8b46535fc9983cf3ebe6b9aa53145
Author: John Doe <john@example.com>
Date:   Sun Aug 31 10:31:44 2014 -0800

    Change page title

[git-p4: depot-paths = "//depot/www/live/": change = 12144]
```

```
$ git log --oneline --all --graph --decorate
* 3be6fd8 (HEAD, master) Correct email address
* 1dcbf21 Merge remote-tracking branch 'p4/master'
  \\
  | * c4689fc (p4/master, p4/HEAD) Grammar fix
  * | cbacd0a Table borders: yes please
  * | b4959b6 Trademark
  //
* 775a46f Change page title
* 05f1ade Update link
* 75cd059 Update copyright
* 70eaf78 Initial import of //depot/www/live/ from the state at revision #head
```

Git      Perforce      775a46f      Git  
           Perforce  
           Perforce

```
$ git p4 submit -n
```

```
Perforce checkout for depot path //depot/www/live/ located at /Users/ben/john_bens
```

```
Would synchronize p4 checkout in /Users/ben/john_bens-mbp_8487/john_bens-mbp_8487/depot/www/
Would apply
b4959b6 Trademark
cbacd0a Table borders: yes please
3be6fd8 Correct email address
```

-n      --dry-run  
 Perforce  
 Perforce

```
$ git p4 submit
[...]
$ git log --oneline --all --graph --decorate
* dadbd89 (HEAD, p4/master, p4/HEAD, master) Correct email address
* 1b79a80 Table borders: yes please
* 0097235 Trademark
* c4689fc Grammar fix
* 775a46f Change page title
* 05f1ade Update link
* 75cd059 Update copyright
* 70eaf78 Initial import of //depot/www/live/ from the state at revision #head
```

Git  
 Perforce  
 Perforce  
 Perforce  
 Git                          git-p4  
 Perforce

```
//depot
└── project
    ├── main
    └── dev

    dev

//depot/project/main/... //depot/project/dev/...

Git-p4
```

```
$ git p4 clone --detect-branches //depot/project@all
Importing from //depot/project@all into project
Initialized empty Git repository in /private/tmp/project/.git/
Importing revision 20 (50%)
    Importing new branch project/dev

        Resuming with change 20
Importing revision 22 (100%)
Updated branches: main dev
$ cd project; git log --oneline --all --graph --decorate
* eae77ae (HEAD, p4/master, p4/HEAD, master) main
| * 10d55fb (p4/project/dev) dev
| * a43cfaf Populate //depot/project/main/... //depot/project/dev/....
|/
* 2b83451 Project init
```

" @ all"

git-p4

Git

|                                 |                              |                                |
|---------------------------------|------------------------------|--------------------------------|
| <b>--detect-branches</b><br>Git | git-p4<br>Perforce<br>git-p4 | Perforce<br>Perforce<br>git-p4 |
|---------------------------------|------------------------------|--------------------------------|

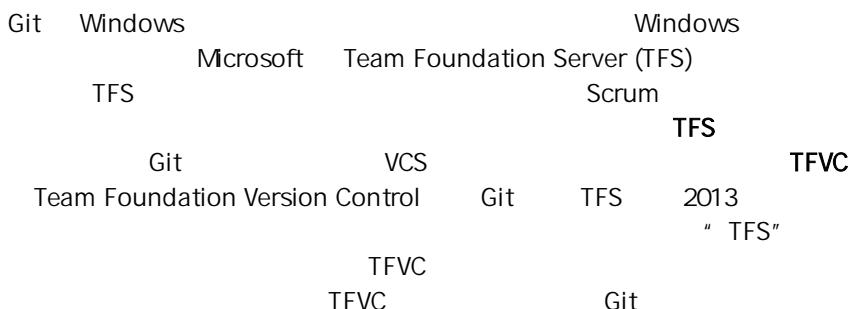
```
$ git init project
Initialized empty Git repository in /tmp/project/.git/
$ cd project
$ git config git-p4.branchList main:dev
$ git clone --detect-branches //depot/project@all .
```

|   |  |                    |
|---|--|--------------------|
| <b>git-p4.branchList</b><br>" main"      " dev" | main:dev<br>git checkout -b dev p4/project/dev | git-p4<br>Perforce |
| <b>git p4 submit</b><br>git-p4                  | git-p4<br>shallow                              | Git-p4<br>Git      |
| <b>git p4 clone</b>                             |  |                    |

## GIT 与 PERFORCE 总结



## Git TFS



## 选择哪个工具



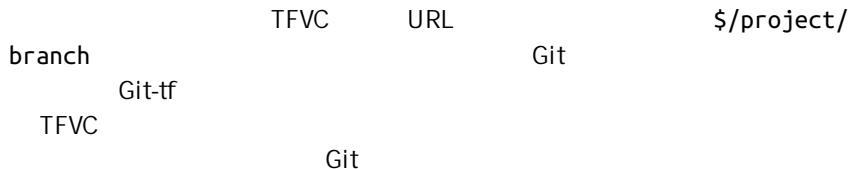
你需要有一个基于 TFVC 的仓库来执行后续的指令。现实中它们并没有 Git 或 Subversion 仓库那样多，所以你可能需要创建一个你自己的仓库。Codeplex (<https://www.codeplex.com>) 或 Visual Studio Online (<http://www.visualstudio.com>) 都是非常好的选择。

### 使用 : GIT-TF

Git

git-tf

```
$ git tf clone https://tfs.codeplex.com:443/tfs/TFS13 $/myproject/Main project_git
```



```
$ cd project_git
$ git log --all --oneline --decorate
512e75a (HEAD, tag: TFS_C35190, origin_tfs/tfs, master) Checkin message
```

TFVC  
git-tf

--deep

```
$ git tf clone https://tfs.codeplex.com:443/tfs/TFS13 $/myproject/Main \
  project_git --deep
Username: domain\user
Password:
Connecting to TFS...
Cloning $/myproject into /tmp/project_git: 100%, done.
Cloned 4 changesets. Cloned last changeset 35190 as d44b17a
$ cd project_git
$ git log --all --oneline --decorate
d44b17a (HEAD, tag: TFS_C35190, origin_tfs/tfs, master) Goodbye
126aa7b (tag: TFS_C35189)
8f77431 (tag: TFS_C35178) FIRST
```

```
0745a25 (tag: TFS_C35177) Created team project folder $/tfvctest via the \
Team Project Creation Wizard
```

|             |                                     |     |
|-------------|-------------------------------------|-----|
| TFVC<br>log | TFS_C35189                          | Git |
|             | TFVC<br>git config git-tf.tag false |     |
|             | - git-tf .git/git-tf                |     |

### 使用 : GIT-TFS

#### Git-tfs

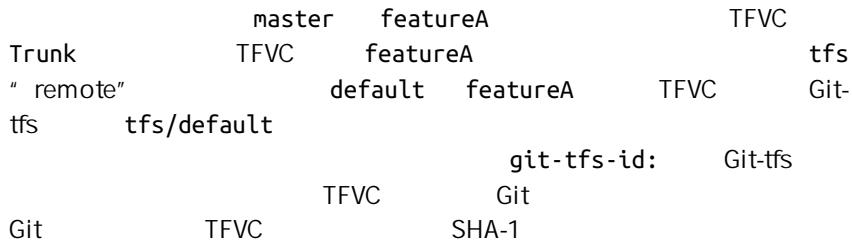
```
PS> git tfs clone --with-branches \
    https://username.visualstudio.com/DefaultCollection \
    $/project/Trunk project_git
Initialized empty Git repository in C:/Users/ben/project_git/.git/
C15 = b75da1aba1ffb359d00e85c52acb261e4586b0c9
C16 = c403405f4989d73a2c3c119e79021cb2104ce44a
Tfs branches found:
- $/tfvc-test/featureA
The name of the local branch will be : featureA
C17 = d202b53f67bde32171d5078968c644e562f1c439
C18 = 44cd729d8df868a8be20438fdeeffb961958b674
```

|     |                 |         |      |     |
|-----|-----------------|---------|------|-----|
| TFS | --with-branches | Git-tfs | TFVC | Git |
|     |                 | TFVC    |      |     |
|     |                 | -       |      |     |
|     |                 |         | "    |     |
|     |                 |         |      | "   |
|     |                 |         |      |     |
|     |                 | git-tfs |      |     |
|     |                 | Git     |      |     |

```
PS> git log --oneline --graph --decorate --all
* 44cd729 (tfs/featureA, featureA) Goodbye
* d202b53 Branched from $/tfvc-test/Trunk
* c403405 (HEAD, tfs/default, master) Hello
* b75da1a New project
PS> git log -1
commit c403405f4989d73a2c3c119e79021cb2104ce44a
Author: Ben Straub <ben@straub.cc>
Date:   Fri Aug 1 03:41:59 2014 +0000
```

Hello

git-tfs-id: [https://username.visualstudio.com/DefaultCollection]\$/**myproject**/Trunk;C16



### GIT-TF[S] 工作流程

无论你使用哪个工具，都需要先设置几个 Git 配置选项来避免一些问题。

```
$ git config set --local core.ignorecase=true
$ git config set --local core.autocrlf=false
```

TFVC TFS

#### 1. TFVC

TFVC Git

#### 2.

TFVC " "

#### 3. TFS

" " TFS -  
TFVC " shelve"  
git-tf  
checkin tool git-tfs

### 工作流程 : GIT-TF

master Git  
Git

```
$ git log --oneline --graph --decorate --all
* 4178a82 (HEAD, master) update code
* 9df2ae3 update readme
* d44b17a (tag: TFS_C35190, origin_tfs/tfs) Goodbye
* 126aa7b (tag: TFS_C35189)
* 8f77431 (tag: TFS_C35178) FIRST
```

```
* 0745a25 (tag: TFS_C35177) Created team project folder $/tfvctest via the \
  Team Project Creation Wizard
```

4178a82 TFVC

```
$ git tf fetch
Username: domain\user
Password:
Connecting to TFS...
Fetching $/myproject at latest changeset: 100%, done.
Downloaded changeset 35320 as commit 8ef06a8. Updated FETCH_HEAD.
$ git log --oneline --graph --decorate --all
* 8ef06a8 (tag: TFS_C35320, origin_tfs/tfs) just some text
| * 4178a82 (HEAD, master) update code
| * 9df2ae3 update readme
|/
* d44b17a (tag: TFS_C35190) Goodbye
* 126aa7b (tag: TFS_C35189)
* 8f77431 (tag: TFS_C35178) FIRST
* 0745a25 (tag: TFS_C35177) Created team project folder $/tfvctest via the \
  Team Project Creation Wizard
```

Git

1.        Git    git  
pull            git-tf    git tf pull  
    TFVC

2.        Git    git-tf    git  
    git-tf    git  
tf pull --rebase

```
$ git rebase FETCH_HEAD
First, rewinding head to replay your work on top of it...
Applying: update readme
Applying: update code
$ git log --oneline --graph --decorate --all
* 5a0e25e (HEAD, master) update code
* 6eb3eb5 update readme
```

```
* 8ef06a8 (tag: TFS_C35320, origin_tfs/tfs) just some text
* d44b17a (tag: TFS_C35190) Goodbye
* 126aa7b (tag: TFS_C35189)
* 8f77431 (tag: TFS_C35178) FIRST
* 0745a25 (tag: TFS_C35177) Created team project folder $/tfvctest via the \
    Team Project Creation Wizard
```

|           |        |
|-----------|--------|
| TFVC      | Git-tf |
| --shallow |        |
| Git       | --deep |

```
$ git tf checkin -m 'Updating readme and code'
Username: domain\user
Password:
Connecting to TFS...
Checking in to $/myproject: 100%, done.
Checked commit 5a0e25e in as changeset 35348
$ git log --oneline --graph --decorate --all
* 5a0e25e (HEAD, tag: TFS_C35348, origin_tfs/tfs, master) update code
* 6eb3eb5 update readme
* 8ef06a8 (tag: TFS_C35320) just some text
* d44b17a (tag: TFS_C35190) Goodbye
* 126aa7b (tag: TFS_C35189)
* 8f77431 (tag: TFS_C35178) FIRST
* 0745a25 (tag: TFS_C35177) Created team project folder $/tfvctest via the \
    Team Project Creation Wizard
```

|            |         |       |      |
|------------|---------|-------|------|
| TFS_C35348 | TFVC    | Git   | TFVC |
| 5a0e25e    | TFVC    | Git   | Git  |
|            | 6eb3eb5 |       |      |
| •          | Git-tf  | TFVC  | Git  |
| •          | TFVC    | Git   | TFVC |
|            | git-tf  | SHA-1 |      |
| •          |         | Git   | TFVC |
|            | Git     | TFVC  |      |

### 工作流程 : GIT-TFS

|         |     |        |
|---------|-----|--------|
| git-tfs | Git | master |
|---------|-----|--------|

```
PS> git log --oneline --graph --all --decorate
* c3bd3ae (HEAD, master) update code
* d85e5a2 update readme
| * 44cd729 (tfss/featureA, featureA) Goodbye
| * d202b53 Branched from $/tfvc-test/Trunk
|
* c403405 (tfss/default) Hello
* b75da1a New project
```

```
PS> git tfs fetch
C19 = aea74a0313de0a391940c999e51c5c15c381d91d
PS> git log --all --oneline --graph --decorate
* aea74a0 (tfss/default) update documentation
| * c3bd3ae (HEAD, master) update code
| * d85e5a2 update readme
|
| * 44cd729 (tfss/featureA, featureA) Goodbye
| * d202b53 Branched from $/tfvc-test/Trunk
|
* c403405 Hello
* b75da1a New project
```

TFVC

aea74a0        tfss/default  
git-tf

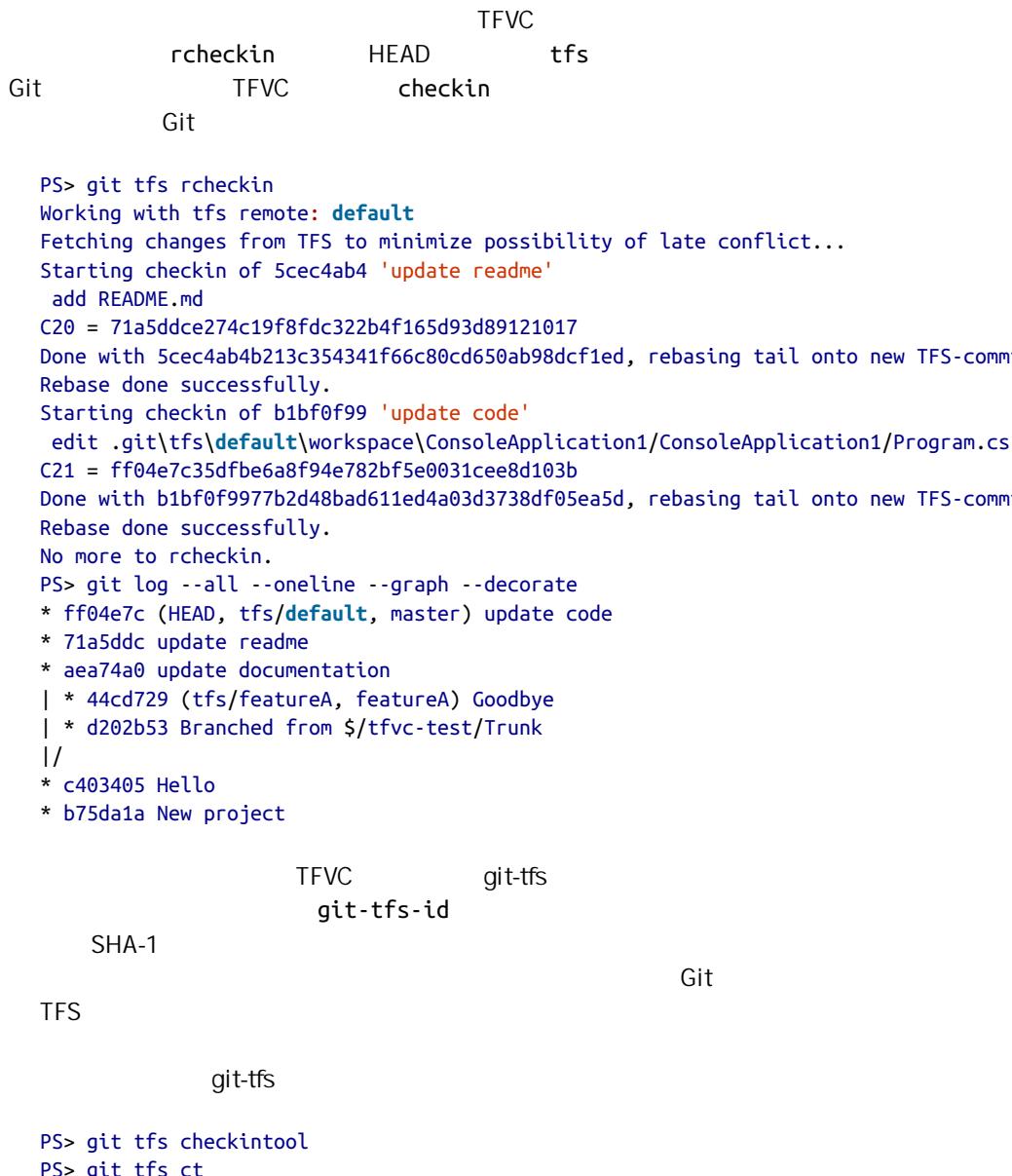
1.

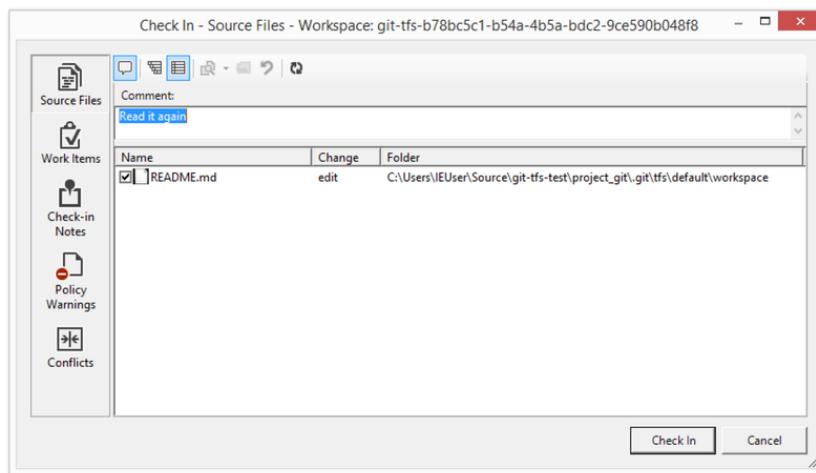
2.

Git

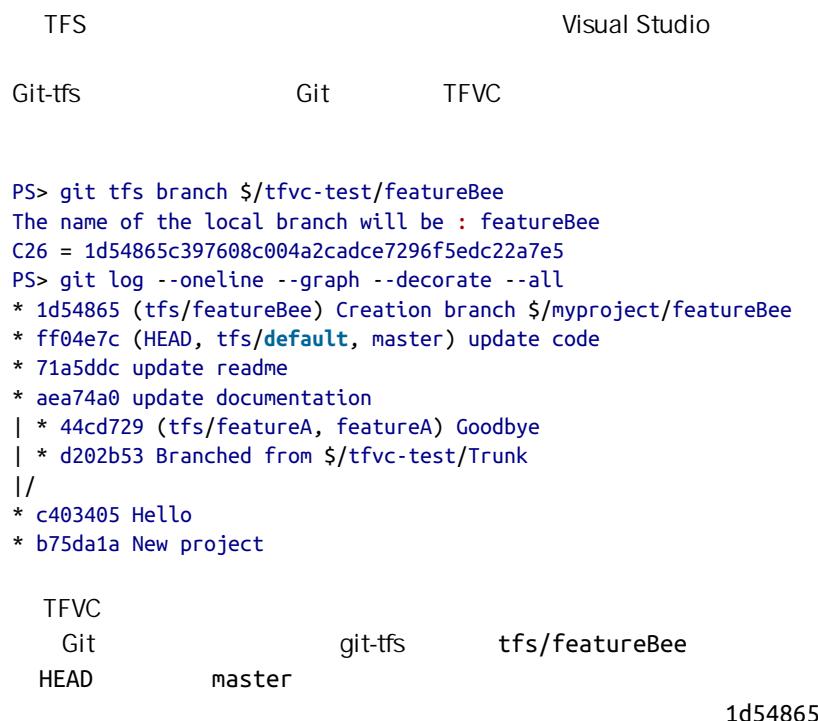
TFVC

```
PS> git rebase tfss/default
First, rewinding head to replay your work on top of it...
Applying: update readme
Applying: update code
PS> git log --all --oneline --graph --decorate
* 10a75ac (HEAD, master) update code
* 5cec4ab update readme
* aea74a0 (tfss/default) update documentation
| * 44cd729 (tfss/featureA, featureA) Goodbye
| * d202b53 Branched from $/tfvc-test/Trunk
|
* c403405 Hello
* b75da1a New project
```

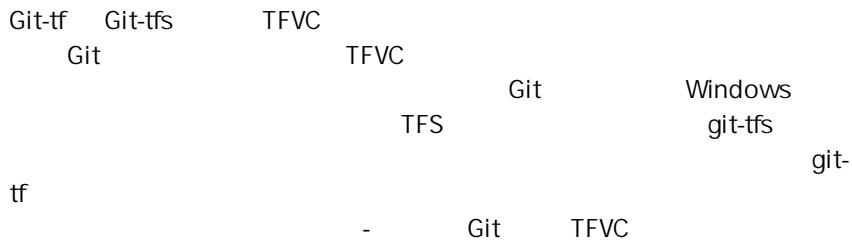


**FIGURE 9-3**

git-tfs 检入工具。



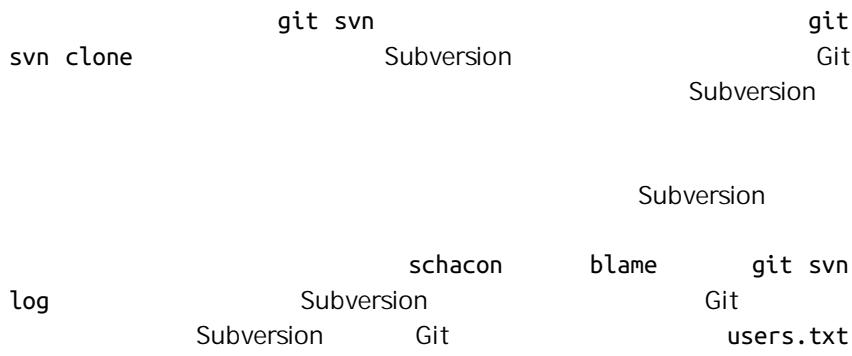
## GIT 与 TFS 总结



## 迁移到 Git



## Subversion



```

schacon = Scott Chacon <schacon@geemail.com>
selse = Someo Nelse <selse@geemail.com>
  
```

SVN

```

$ svn log --xml | grep author | sort -u | \
perl -pe 's/.*/>(.*)<.*$/1 = /'
  
```

|            |                             |                          |         |
|------------|-----------------------------|--------------------------|---------|
| XML        | grep sort perl<br>users.txt | Git                      |         |
|            |                             | git svn                  |         |
|            |                             | --no-metadata clone init | git svn |
| Subversion |                             |                          | import  |

```
$ git svn clone http://my-project.googlecode.com/svn/ \
--authors-file=users.txt --no-metadata -s my_project
```

my\_project

Subversion

```
commit 37efa680e8473b615de980fa935944215428a35a
Author: schacon <schacon@4c93b258-373f-11de-be05-5f7a86268029>
Date:   Sun May 3 00:12:22 2009 +0000

fixed install - go to trunk

git-svn-id: https://my-project.googlecode.com/svn/trunk@94 4c93b258-373f-11de-
be05-5f7a86268029
```

```
commit 03a8785f44c8ea5cdb0e8834b7c8e6c469be2ff2
Author: Scott Chacon <schacon@geemail.com>
Date:   Sun May 3 00:12:22 2009 +0000
```

fixed install - go to trunk

Author

git-svn-id

git svn

Git

```
$ cp -Rf .git/refs/remotes/origin/tags/* .git/refs/tags/
$ rm -Rf .git/refs/remotes/origin/tags
```

remotes/origin/tags/

refs/remotes

```
$ cp -Rf .git/refs/remotes/* .git/refs/heads/
$ rm -Rf .git/refs/remotes
```

Git

Git

Git

```
$ git remote add origin git@my-git-server:myrepository.git
```

```
$ git push origin --all
```

Git

## Mercurial

Mercurial    Git

Mercurial        Git

" hg-fast-export"

```
$ git clone http://repo.or.cz/r/fast-export.git /tmp/fast-export
```

Mercurial

```
$ hg clone <remote repo URL> /tmp/hg-repo
```

Git

Mercurial

bash

```
$ cd /tmp/hg-repo
$ hg log | grep user: | sort | uniq | sed 's/user: *//' > ../authors
```

/tmp/authors

```

bob
bob@localhost
bob <bob@company.com>
bob jones <bob <AT> company <DOT> com>
Bob Jones <bob@company.com>
Joe Smith <joe@company.com>
```

|     |                               |                |
|-----|-------------------------------|----------------|
| Bob | Git                           | Hg-fast-export |
|     | ={new name and email address} |                |

```

bob=Bob Jones <bob@company.com>
bob@localhost=Bob Jones <bob@company.com>
bob jones <bob <AT> company <DOT> com>=Bob Jones <bob@company.com>
bob <bob@company.com>=Bob Jones <bob@company.com>
```

Git

```
$ git init /tmp/converted
$ cd /tmp/converted
$ ./hg-fast-export/hg-fast-export.sh -r /tmp/hg-repo -A /tmp/authors
```

|                          |   |                            |
|--------------------------|---|----------------------------|
| - <b>r</b><br>- <b>A</b> | hg-fast-export<br><br><b>Git</b> “ fast-import” | Mercurial<br><br>Mercurial |
|--------------------------|---|----------------------------|

```
$ ./hg-fast-export/hg-fast-export.sh -r /tmp/hg-repo -A /tmp/authors
Loaded 4 authors
master: Exporting full revision 1/22208 with 13/0/0 added/changed/removed files
master: Exporting simple delta revision 2/22208 with 1/1/0 added/changed/removed files
master: Exporting simple delta revision 3/22208 with 0/1/0 added/changed/removed files
[...]
master: Exporting simple delta revision 22206/22208 with 0/4/0 added/changed/removed files
master: Exporting simple delta revision 22207/22208 with 0/2/0 added/changed/removed files
master: Exporting thorough delta revision 22208/22208 with 3/213/0 added/changed/removed files
Exporting tag [0.4c] at [hg r9] [git :10]
```

```

Exporting tag [0.4d] at [hg r16] [git :17]
[...]
Exporting tag [3.1-rc] at [hg r21926] [git :21927]
Exporting tag [3.1] at [hg r21973] [git :21974]
Issued 22315 commands
git-fast-import statistics:
-----
Alloc'd objects: 120000
Total objects: 115032 ( 208171 duplicates ) 39602
    blobs : 40504 ( 205320 duplicates ) 47599
    trees : 52320 ( 2851 duplicates ) 0
    commits: 22208 ( 0 duplicates ) 0
    tags : 0 ( 0 duplicates ) 0
Total branches: 109 ( 2 loads )
marks: 1048576 ( 22208 unique )
atoms: 1952
Memory total: 7860 KiB
    pools: 2235 KiB
    objects: 5625 KiB
-----
pack_report: getpagesize() = 4096
pack_report: core.packedGitWindowSize = 1073741824
pack_report: core.packedGitLimit = 8589934592
pack_report: pack_used_ctr = 90430
pack_report: pack_mmap_calls = 46771
pack_report: pack_open_windows = 1 /
pack_report: pack_mapped = 340852700 / 340852700
-----
$ git shortlog -sn
 369 Bob Jones
 365 Joe Smith

```

|                  |     |           |
|------------------|-----|-----------|
| Mercurial<br>Git | Git | Mercurial |
|------------------|-----|-----------|

```

$ git remote add origin git@my-git-server:myrepository.git
$ git push origin --all

```

## Perforce

|                 |                    |                                 |
|-----------------|--------------------|---------------------------------|
| Perforce<br>Git | Perforce<br>git-p4 | Perforce<br>Perforce Git Fusion |
|-----------------|--------------------|---------------------------------|

## PERFORCE GIT FUSION

Git Fusion

“ GitFusion”

Git

Fusion

Git

Perforce

Git

### GIT-P4

Git-p4

Perforce

Jam

Perforce

P4PORT

```
$ export P4PORT=public.perforce.com:1666
```

---

为了继续后续步骤，需要连接到 Perforce 仓库。在我们的例子中将会使用在 public.perforce.com 的公开仓库，但是你可以使用任何你有权限的仓库。

---

git p4 clone

Perforce

Jam

```
$ git-p4 clone //guest/perforce_software/jam@all p4import
Importing from //guest/perforce_software/jam@all into p4import
Initialized empty Git repository in /private/tmp/p4import/.git/
Import destination: refs/remotes/p4/master
Importing revision 9957 (100%)
```

clone

--detect-branches

“ “

git p4

p4import

git log

```
$ git log -2
commit e5da1c909e5db3036475419f6379f2c73710c4e6
Author: giles <giles@giles@perforce.com>
Date:   Wed Feb 8 03:13:27 2012 -0800
```

Correction to line 355; change </UL> to </OL>.

```
[git-p4: depot-paths = "//public/jam/src/": change = 8068]

commit aa21359a0a135dda85c50a7f7cf249e4f7b8fd98
Author: kwirth <kwirth@perforce.com>
Date:   Tue Jul 7 01:35:51 2009 -0800

    Fix spelling error on Jam doc page (cummulative -> cumulative).

[git-p4: depot-paths = "//public/jam/src/": change = 7304]
```

git-p4  
Perforce

git filter-branch

```
$ git filter-branch --msg-filter 'sed -e "/^\\[git-p4:/d"
Rewrite e5da1c909e5db3036475419f6379f2c73710c4e6 (125/125)
Ref 'refs/heads/master' was rewritten
```

git log SHA-1  
git-p4

```
$ git log -2
commit b17341801ed838d97f7800a54a6f9b95750839b7
Author: giles <giles@giles@perforce.com>
Date:   Wed Feb 8 03:13:27 2012 -0800

    Correction to line 355; change </UL> to </OL>.

commit 3e68c2e26cd89cb983eb52c024ecdfba1d6b3fff
Author: kwirth <kwirth@perforce.com>
Date:   Tue Jul 7 01:35:51 2009 -0800

    Fix spelling error on Jam doc page (cummulative -> cumulative).
```

Git

TFS

|      |         |
|------|---------|
| TFVC | Git     |
|      | git-tfs |

```
git-tf          git-tfs      git-tfs
    git-tf
```

---

这是一个单向转换。这意味着 Git 仓库无法连接到原始的 TFVC 项目。

---

```
TFVC
Git          tf
```

```
PS> tf history $/myproject -recursive > AUTHORS_TMP
```

```
AUTHORS_TMP
User
cut          11-20
```

```
PS> cat AUTHORS_TMP | cut -b 11-20 | tail -n+3 | uniq | sort > AUTHORS
```

```
cut          11      22      tail
           ASCII
uniq        AUTOHRS
git-tfs
```

```
DOMAIN\username = User Name <email@address.com>
```

```
TFVC      " User"      Git
```

```
TFVC
```

```
PS> git tfs clone --with-branches --authors=AUTHORS https://username.visualstudio.com/Default
```

```
git-tfs-id
```

```
PS> git filter-branch -f --msg-filter 'sed "s/^git-tfs-id:.*$//g"' -- --all
```

```
Git          sed          " git-tfs-id:"
Git
```

```
Git
```

CVS Clear Case Visual Source  
Safe  
import  
Chapter 10

Git                          Git

git fast-import  
current  
back\_YYYY\_MM\_DD

Git

```
$ ls /opt/import_from
back_2014_01_02
back_2014_01_04
back_2014_01_14
back_2014_02_03
current
```

Git                          Git

Git  
fast-import

Ruby

标准

Windows  
- git fast-import  
dows                          CRLF

LF                          Win

```
last_mark = nil

# loop through the directories
Dir.chdir(argv[0]) do
  Dir.glob("*").each do |dir|
    next if File.file?(dir)
```

```
# move into the target directory
Dir.chdir(dir) do
  last_mark = print_export(dir, last_mark)
end
end
end

print_export

"      "
fast-import

print_export

mark = convert_dir_to_mark(dir)

$marks = []
def convert_dir_to_mark(dir)
  if !$marks.include?(dir)
    $marks << dir
  end
  ($marks.index(dir) + 1).to_s
end

print_export

date = convert_dir_to_date(dir)

convert_dir_to_date

def convert_dir_to_date(dir)
  if dir == 'current'
    return Time.now().to_i
  else
    dir = dir.gsub('back_', '')
    (year, month, day) = dir.split('_')
    return Time.local(year, month, day).to_i
  end
end
```

```
$author = 'John Doe <john@example.com>'
```

```
# print the import information
puts 'commit refs/heads/master'
puts 'mark :' + mark
puts "committer #{$author} #{date} -0700"
export_data('imported from ' + dir)
puts 'from :' + last_mark if last_mark
```

-0700

data (size)\n(contents)

export\_data

```
def export_data(string)
  print "data #[string.size]\n#{string}"
end
```

deleteall

Git

```
puts 'deleteall'
Dir.glob("**/*").each do |file|
  next if !File.file?(file)
  inline_data(file)
end
```

fast-import

Git

fast-import man

```

M 644 inline path/to/file
data (size)
(file contents)

      644
755    inline           inline_data

def inline_data(file, code = 'M', mode = '644')
  content = File.read(file)
  puts "#{code} #{mode} inline #{file}"
  export_data(content)
end

  export_data

return mark

```

---

如果在 Windows 上还需要确保增加一个额外步骤。正如之前提到的，Windows 使用 CRLF 作为换行符而 git fast-import 只接受 LF。为了修正这个问题使 git fast-import 正常工作，你需要告诉 ruby 使用 LF 代替 CRLF：

```
$stdout.binmode
```

---

```

#!/usr/bin/env ruby

$stdout.binmode
$author = "John Doe <john@example.com>"

$marks = []
def convert_dir_to_mark(dir)
  if !$marks.include?(dir)
    $marks << dir
  end
  ($marks.index(dir)+1).to_s
end

def convert_dir_to_date(dir)
  if dir == 'current'
    return Time.now().to_i
  else
    dir = dir.gsub('back_', '')
  end

```

```

        (year, month, day) = dir.split('_')
        return Time.local(year, month, day).to_i
    end
end

def export_data(string)
    print "data #{string.size}\n#{string}"
end

def inline_data(file, code='M', mode='644')
    content = File.read(file)
    puts "#{code} #{mode} inline #{file}"
    export_data(content)
end

def print_export(dir, last_mark)
    date = convert_dir_to_date(dir)
    mark = convert_dir_to_mark(dir)

    puts 'commit refs/heads/master'
    puts "mark :#{mark}"
    puts "committer #{author} #{date} -0700"
    export_data("imported from #{dir}")
    puts "from :#{last_mark}" if last_mark

    puts 'deleteall'
    Dir.glob("**/*").each do |file|
        next if !File.file?(file)
        inline_data(file)
    end
    mark
end

# Loop through the directories
last_mark = nil
Dir.chdir(argv[0]) do
    Dir.glob("*").each do |dir|
        next if File.file?(dir)

        # move into the target directory
        Dir.chdir(dir) do
            last_mark = print_export(dir, last_mark)
        end
    end
end

```

```
$ ruby import.rb /opt/import_from
commit refs/heads/master
mark :1
committer John Doe <john@example.com> 1388649600 -0700
data 29
imported from back_2014_01_02deleteall
M 644 inline README.md
data 28
# Hello

This is my readme.
commit refs/heads/master
mark :2
committer John Doe <john@example.com> 1388822400 -0700
data 29
imported from back_2014_01_04from :1
deleteall
M 644 inline main.rb
data 34
#!/bin/env ruby

puts "Hey there"
M 644 inline README.md
(...)
```

git fast-import

git init

```
$ git init
Initialized empty Git repository in /opt/import_to/.git/
$ ruby import.rb /opt/import_from | git fast-import
git-fast-import statistics:
-----
Alloc'd objects:      5000
Total objects:        13 (       6 duplicates          )
                      blobs :      5 (       4 duplicates          ) 3 deltas of 5 attempts)
                      trees :      4 (       1 duplicates          ) 0 deltas of 4 attempts)
                      commits:     4 (       1 duplicates          ) 0 deltas of 0 attempts)
                      tags :       0 (       0 duplicates          ) 0 deltas of 0 attempts)
-----
```

|                 |                                 |  |
|-----------------|---------------------------------|--|
| Total branches: | 1 (       1 loads          )    |  |
| marks:          | 1024 (      5 unique          ) |  |
| atoms:          | 2                               |  |
| Memory total:   | 2344 KiB                        |  |
| pools:          | 2110 KiB                        |  |
| objects:        | 234 KiB                         |  |

-----

```

pack_report: getpagesize()          =      4096
pack_report: core.packedGitWindowSize = 1073741824
pack_report: core.packedGitLimit    = 8589934592
pack_report: pack_used_ctr        =       10
pack_report: pack_mmap_calls     =        5
pack_report: pack_open_windows   =       2 /       2
pack_report: pack_mapped         = 1457 / 1457
-----
```

13                  4                  1

`git log`

```

$ git log -2
commit 3caa046d4aac682a55867132ccdfbe0d3fdee498
Author: John Doe <john@example.com>
Date:   Tue Jul 29 19:39:04 2014 -0700

    imported from current

commit 4afc2b945d0d3c8cd00556fbe2e8224569dc9def
Author: John Doe <john@example.com>
Date:   Mon Feb 3 01:00:00 2014 -0700

    imported from back_2014_02_03
```

-                  Git

-                  master

```

$ ls
$ git reset --hard master
HEAD is now at 3caa046 imported from current
$ ls
README.md main.rb
```

fast-import        -

Git                  contrib/fast-import

## 总结

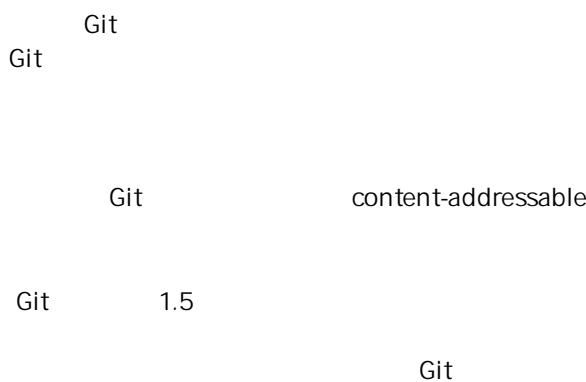
Git

Git

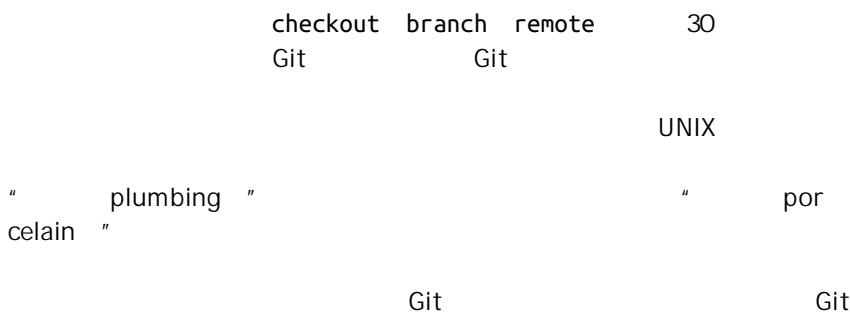
Git



# Git 内部原理 10



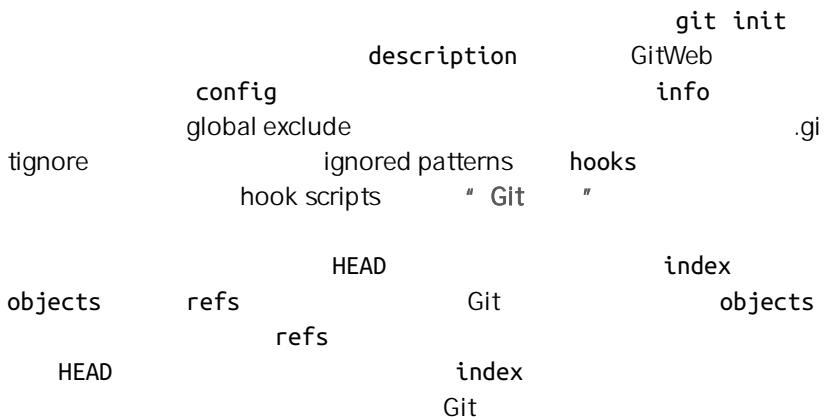
## 底层命令和高层命令



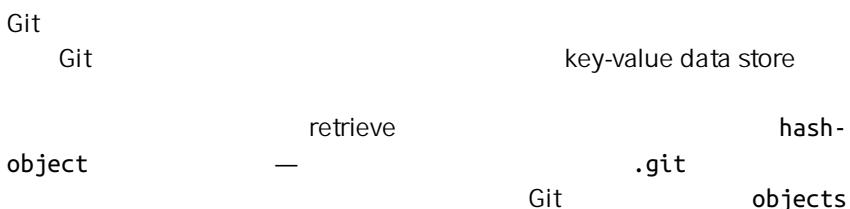
```

$ ls -F1
HEAD
config*
description
hooks/
info/
objects/
refs/

```



## Git 对象



```

$ git init test
Initialized empty Git repository in /tmp/test/.git/
$ cd test
$ find .git/objects
.git/objects

```

```
.git/objects/info  
.git/objects/pack  
$ find .git/objects -type f
```

```
Git    objects          pack    info  
      Git
```

```
$ echo 'test content' | git hash-object -w --stdin  
d670460b4b4aece5915caf5c68d12f560a9fe3e4
```

```
-w      hash-object  
       --stdin  
  
40           SHA-1      —  
header        SHA-1  
               Git
```

```
$ find .git/objects -type f  
.git/objects/d6/70460b4b4aece5915caf5c68d12f560a9fe3e4
```

```
objects          Git  
—  
               SHA-1  
38  
  
cat-file      Git  
Git            cat-file -p
```

```
$ git cat-file -p d670460b4b4aece5915caf5c68d12f560a9fe3e4  
test content
```

```
Git
```

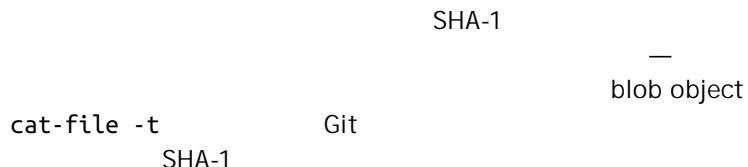
```
$ echo 'version 1' > test.txt  
$ git hash-object -w test.txt  
83baae61804e65cc73a7201a7252750c76066a30
```

```
$ echo 'version 2' > test.txt  
$ git hash-object -w test.txt  
1f7a7a472abf3dd9643fd615f6da379c4acb3e3a
```

```
$ find .git/objects -type f  
.git/objects/1f/7a7a472abf3dd9643fd615f6da379c4acb3e3a  
.git/objects/83/baae61804e65cc73a7201a7252750c76066a30  
.git/objects/d6/70460b4b4aece5915caf5c68d12f560a9fe3e4
```

```
$ git cat-file -p 83baae61804e65cc73a7201a7252750c76066a30 > test.txt  
$ cat test.txt  
version 1
```

```
$ git cat-file -p 1f7a7a472abf3dd9643fd615f6da379c4acb3e3a > test.txt  
$ cat test.txt  
version 2
```



```
$ git cat-file -t 1f7a7a472abf3dd9643fd615f6da379c4acb3e3a  
blob
```

tree object

Git

UNIX

UNIX

|               |                  |
|---------------|------------------|
| inodes<br>try | tree en<br>SHA-1 |
|---------------|------------------|

```
$ git cat-file -p master^{tree}
100644 blob a906cb2a4a904a152e80877d4088654daad0c859      README
100644 blob 8f94139338f9404f26296befa88755fc2598c289      Rakefile
040000 tree 99f1a6d12cb4b6f19c8655fca46c3ecf317074e0      lib
```

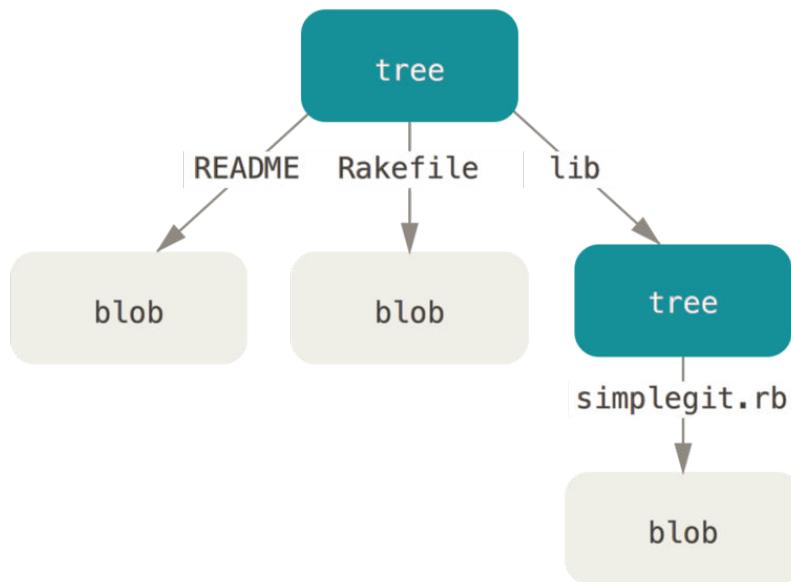
master^{tree}                    master  
      lib

```
$ git cat-file -p 99f1a6d12cb4b6f19c8655fca46c3ecf317074e0
100644 blob 47c6340d6459e05787f644c2447d2595f5d3a54b      simplegit.rb
```

Git

**FIGURE 10-1**

简化版的 Git 数据模型。



Git  
in  
dex

```
update-index      —      test.txt      —  
                  test.txt  
                  --add
```

```
cacheinfo          Git  
                  SHA-1
```

```
$ git update-index --add --cacheinfo 100644 \  
83baae61804e65cc73a7201a7252750c76066a30 test.txt
```

```
100644  
100755      120000  
              UNIX  
Git
```

```
write-tree  
-w      —      write-  
tree
```

```
$ git write-tree  
d8329fc1cc938780ffdd9f94e0d364e0ea74f579  
$ git cat-file -p d8329fc1cc938780ffdd9f94e0d364e0ea74f579  
100644 blob 83baae61804e65cc73a7201a7252750c76066a30      test.txt
```

```
$ git cat-file -t d8329fc1cc938780ffdd9f94e0d364e0ea74f579  
tree
```

test.txt

```
$ echo 'new file' > new.txt  
$ git update-index test.txt  
$ git update-index --add new.txt
```

test.txt

new.txt

```
$ git write-tree  
0155eb4229851634a0f03eb265b69f5a2d56f341  
$ git cat-file -p 0155eb4229851634a0f03eb265b69f5a2d56f341  
100644 blob fa49b077972391ad58037050f2a75f74e3671e92      new.txt  
100644 blob 1f7a7a472abf3dd9643fd615f6da379c4acb3e3a      test.txt
```

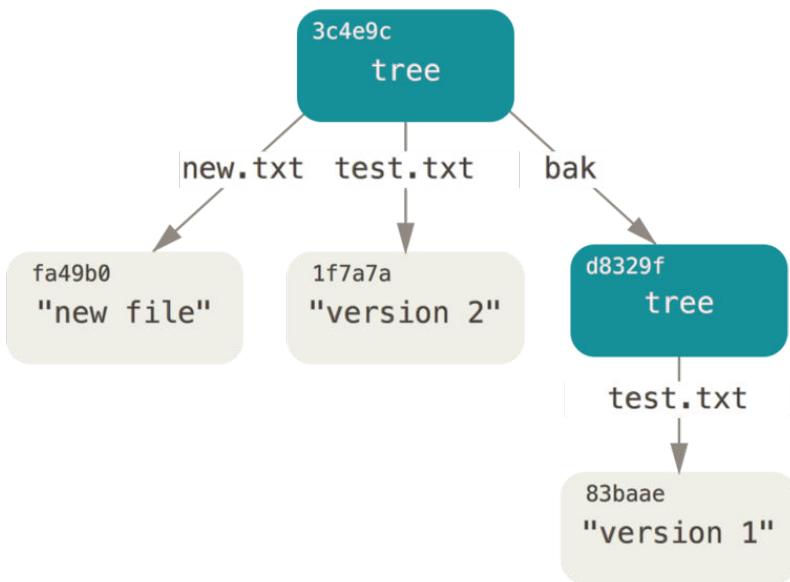
```
test.txt      SHA-1  
1f7a7a          "  
                   "  
                   read-  
tree           read-tree  
--prefix
```

```
$ git read-tree --prefix=bak d8329fc1cc938780ffdd9f94e0d364e0ea74f579  
$ git write-tree  
3c4e9cd789d88d8d89c1073707c3585e41b0e614  
$ git cat-file -p 3c4e9cd789d88d8d89c1073707c3585e41b0e614  
040000 tree d8329fc1cc938780ffdd9f94e0d364e0ea74f579      bak  
100644 blob fa49b077972391ad58037050f2a75f74e3671e92      new.txt  
100644 blob 1f7a7a472abf3dd9643fd615f6da379c4acb3e3a      test.txt
```

```
bak          test.txt  
Git
```

**FIGURE 10-2**

当前 Git 的数据内容  
结构。



SHA-1

commit object

commit-tree

SHA-1

```
$ echo 'first commit' | git commit-tree d8329f
fdf4fc3344e67ab068f836878b6c4951e3b15f3d
```

cat-file

```
$ git cat-file -p fdf4fc3
tree d8329fc1cc938780ffdd9f94e0d364e0ea74f579
author Scott Chacon <schacon@gmail.com> 1243040974 -0700
committer Scott Chacon <schacon@gmail.com> 1243040974 -0700
```

```
first commit
```

```
/ user.name user.email
```

```
$ echo 'second commit' | git commit-tree 0155eb -p fdf4fc3  
cac0cab538b970a37ea1e769cbbde608743bc96d  
$ echo 'third commit' | git commit-tree 3c4e9c -p cac0cab  
1a410efbd13591db07496601ebc7a059dd55cfe9
```

```
SHA-1      git log  
git log      Git
```

```
$ git log --stat 1a410e  
commit 1a410efbd13591db07496601ebc7a059dd55cfe9  
Author: Scott Chacon <schacon@gmail.com>  
Date:   Fri May 22 18:15:24 2009 -0700
```

```
third commit
```

```
bak/test.txt | 1 +  
1 file changed, 1 insertion(+)
```

```
commit cac0cab538b970a37ea1e769cbbde608743bc96d  
Author: Scott Chacon <schacon@gmail.com>  
Date:   Fri May 22 18:14:29 2009 -0700
```

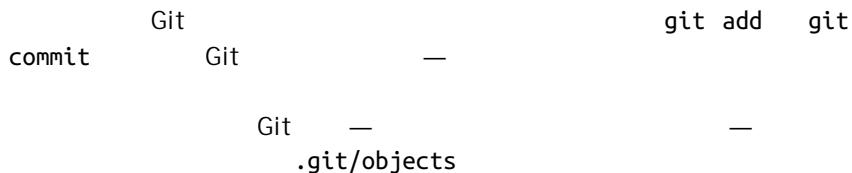
```
second commit
```

```
new.txt  | 1 +  
test.txt | 2 +-  
2 files changed, 2 insertions(+), 1 deletion(-)
```

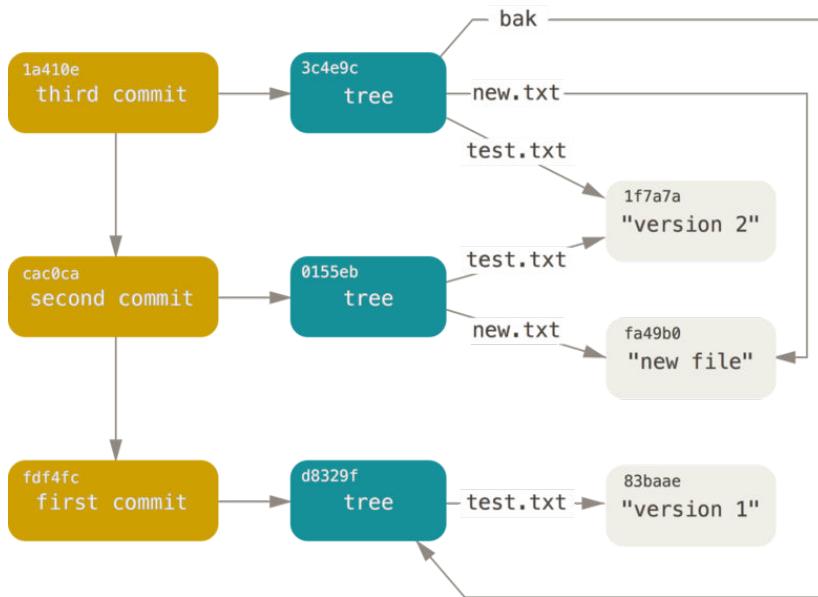
```
commit fdf4fc3344e67ab068f836878b6c4951e3b15f3d  
Author: Scott Chacon <schacon@gmail.com>  
Date:   Fri May 22 18:09:34 2009 -0700
```

```
first commit
```

```
test.txt | 1 +  
1 file changed, 1 insertion(+)
```



```
$ find .git/objects -type f  
.git/objects/01/55eb4229851634a0f03eb265b69f5a2d56f341 # tree 2  
.git/objects/1a/410efbd13591db07496601ebc7a059dd55cf9 # commit 3  
.git/objects/1f/7a7a472abf3dd9643fd615f6da379c4acb3e3a # test.txt v2  
.git/objects/3c/4e9cd789d88d8d89c1073707c3585e41b0e614 # tree 3  
.git/objects/83/baae61804e65cc73a7201a7252750c76066a30 # test.txt v1  
.git/objects/ca/c0cab538b970a37ea1e769cbbde608743bc96d # commit 2  
.git/objects/d6/70460b4b4aece5915caf5c68d12f560a9fe3e4 # 'test content'  
.git/objects/d8/329fc1cc938780ffdd9f94e0d364e0ea74f579 # tree 1  
.git/objects/fa/49b077972391ad58037050f2a75f74e3671e92 # new.txt  
.git/objects/fd/f4fc3344e67ab068f836878b6c4951e3b15f3d # commit 1
```

**FIGURE 10-3**

你的 Git 目录下的所有对象。

Git

Ruby

— " what is up, doc? " —

irb

Ruby

```
$ irb
>> content = "what is up, doc?"
=> "what is up, doc?"
```

Git

" blob"

Git

null byte

```
>> header = "blob #{content.length}\0"
=> "blob 16\0000"
```

|              |      |                         |
|--------------|------|-------------------------|
| Git          |      |                         |
| SHA-1        | Ruby | SHA-1 — require         |
| SHA-1 digest |      | Digest::SHA1hexdigest() |
| digest()     |      |                         |

```
>> store = header + content
=> "blob 16\u0000what is up, doc?"
>> require 'digest/sha1'
=> true
>> sha1 = Digest::SHA1hexdigest(store)
=> "bd9dbf5aae1a3862dd1526723246b20206e5fc37"
```

|     |      |      |                         |
|-----|------|------|-------------------------|
| Git | zlib | Ruby | zlib                    |
|     |      |      | zlib::Deflate.deflate() |

```
>> require 'zlib'
=> true
>> zlib_content = Zlib::Deflate.deflate(store)
=> "x\x9CK\xCA\xC90R04c(\xC9F,Q\xC8,V(-\xD0QH\xC90\xB6\aa\x00_\x1C\aa\x9D"
```

|                     |  |       |             |
|---------------------|--|-------|-------------|
| zlib                |  | SHA-1 | 38          |
|                     |  |       | Ruby        |
| FileUtils.mkdir_p() |  |       | File.open() |
|                     |  |       | write()     |
| zlib                |  |       |             |

```
>> path = '.git/objects/' + sha1[0,2] + '/' + sha1[2,38]
=> ".git/objects/bd/9dbf5aae1a3862dd1526723246b20206e5fc37"
>> require 'fileutils'
=> true
>> FileUtils.mkdir_p(File.dirname(path))
=> ".git/objects/bd"
>> File.open(path, 'w') { |f| f.write zlib_content }
=> 32
```

|                      |     |         |
|----------------------|-----|---------|
| —                    | Git | Git     |
|                      |     |         |
| " commit"    " tree" |     | " blob" |

## Git 引用

```
git log 1a410e  
1a410e  
SHA-1  
SHA-1  
Git " references refs "  
.git/refs SHA-1
```

```
$ find .git/refs  
.git/refs  
.git/refs/heads  
.git/refs/tags  
$ find .git/refs -type f
```

```
$ echo "1a410efbd13591db07496601ebc7a059dd55cfe9" > .git/refs/heads/master
```

```
Git SHA-1
```

```
$ git log --pretty=oneline master  
1a410efbd13591db07496601ebc7a059dd55cfe9 third commit  
cac0cab538b970a37ea1e769cbde608743bc96d second commit  
fdf4fc3344e67ab068f836878b6c4951e3b15f3d first commit
```

```
Git  
update-ref
```

```
$ git update-ref refs/heads/master 1a410efbd13591db07496601ebc7a059dd55cfe9
```

```
Git
```

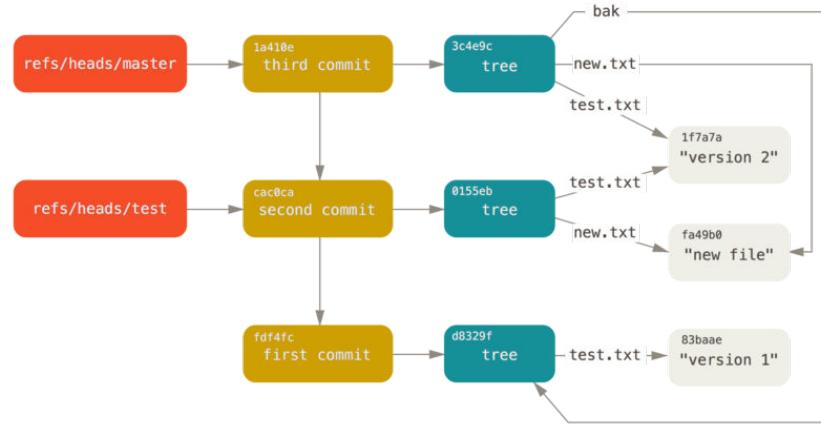
```
$ git update-ref refs/heads/test cac0ca
```

```
$ git log --pretty=oneline test
cac0cab538b970a37ea1e769ccbde608743bc96d second commit
fdf4fc3344e67ab068f836878b6c4951e3b15f3d first commit
```

## Git

**FIGURE 10-4**

包含分支引用的 Git 目录对象。



git branch (branchname)  
update-ref    Git  
  SHA-1

HEAD

|               |   |
|---------------|---|
| SHA-1<br>HEAD | git branch (branchname)<br>HEAD    Git<br>symbolic reference<br>SHA-1 —<br>HEAD |
|---------------|---|

```
$ cat .git/HEAD
ref: refs/heads/master
```

```
git checkout test  Git          HEAD
```

```
$ cat .git/HEAD  
ref: refs/heads/test
```

```
git commit          HEAD  
SHA-1
```

```
symbolic-ref      HEAD
```

```
$ git symbolic-ref HEAD  
refs/heads/master
```

```
HEAD
```

```
$ git symbolic-ref HEAD refs/heads/test  
$ cat .git/HEAD  
ref: refs/heads/test
```

```
$ git symbolic-ref HEAD test  
fatal: Refusing to point HEAD outside of refs/
```

Git

tag object —

—

## Chapter 2

```
$ git update-ref refs/tags/v1.0 cac0cab538b970a37ea1e769cbbde608743bc96d
```

—  
Git

-a

```
$ git tag -a v1.1 1a410efbd13591db07496601ebc7a059dd55cfe9 -m 'test tag'
```

SHA-1

```
$ cat .git/refs/tags/v1.1  
9585191f37f7b0fb9444f35a9bf50de191beadc2
```

SHA-1      cat-file

```
$ git cat-file -p 9585191f37f7b0fb9444f35a9bf50de191beadc2  
object 1a410efbd13591db07496601ebc7a059dd55cfe9  
type commit  
tag v1.1  
tagger Scott Chacon <schacon@gmail.com> Sat May 23 16:48:58 2009 -0700  
  
test tag
```

object                    SHA-1

Git                        Git                        GPG

Git

```
$ git cat-file blob junio-gpg-pub
```

Linux

remote reference

Git

refs/remotes

origin

master

```
$ git remote add origin git@github.com:schacon/simplegit-progit.git
$ git push origin master
Counting objects: 11, done.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 716 bytes, done.
Total 7 (delta 2), reused 4 (delta 1)
To git@github.com:schacon/simplegit-progit.git
  a11bef0..ca82a6d  master -> master
```

|        |                            |       |        |
|--------|----------------------------|-------|--------|
|        | refs/remotes/origin/master |       | origin |
| master |                            | SHA-1 |        |
| master |                            | SHA-1 |        |

```
$ cat .git/refs/remotes/origin/master
ca82a6dff817ec66f44342007202690a93763949
```

|     |            |              |        |
|-----|------------|--------------|--------|
|     | refs/heads |              |        |
|     |            | git checkout |        |
| Git | HEAD       |              | commit |
|     |            | Git          |        |

## 包文件

|     |   |   |    |
|-----|---|---|----|
| Git |   |   | 11 |
| — 4 | 3 | 3 | 1  |

```
$ find .git/objects -type f
.git/objects/01/55eb4229851634a0f03eb265b69f5a2d56f341 # tree 2
.git/objects/1a/410efbd13591db07496601ebc7a059dd55cfe9 # commit 3
.git/objects/1f/7a7a472abf3dd9643fd615f6da379c4acb3e3a # test.txt v2
.git/objects/3c/4e9cd789d88d8d89c1073707c3585e41b0e614 # tree 3
.git/objects/83/baae61804e65cc73a7201a7252750c76066a30 # test.txt v1
.git/objects/95/85191f37f7b0fb9444f35a9bf50de191beadc2 # tag
.git/objects/ca/c0cab538b970a37ea1e769cbbde608743bc96d # commit 2
.git/objects/d6/70460b4b4aece5915caf5c68d12f560a9fe3e4 # 'test content'
.git/objects/d8/329fc1cc938780ffdd9f94e0d364e0ea74f579 # tree 1
.git/objects/fa/49b077972391ad58037050f2a75f74e3671e92 # new.txt
.git/objects/fd/f4fc3344e67ab068f836878b6c4951e3b15f3d # commit 1
```

Git zlib



repo.rb

400

Git

```
$ git cat-file -s b042a60ef7dff760008df33cee372b945b6e884e  
22054
```

22K

Git

```
          Git          Git          Git  
"      loose   "          packfile   "  
          "          "          "          git gc  
          Git          Git  
git gc      Git
```

```
$ git gc  
Counting objects: 18, done.  
Delta compression using up to 8 threads.  
Compressing objects: 100% (14/14), done.  
Writing objects: 100% (18/18), done.  
Total 18 (delta 3), reused 0 (delta 0)
```

objects

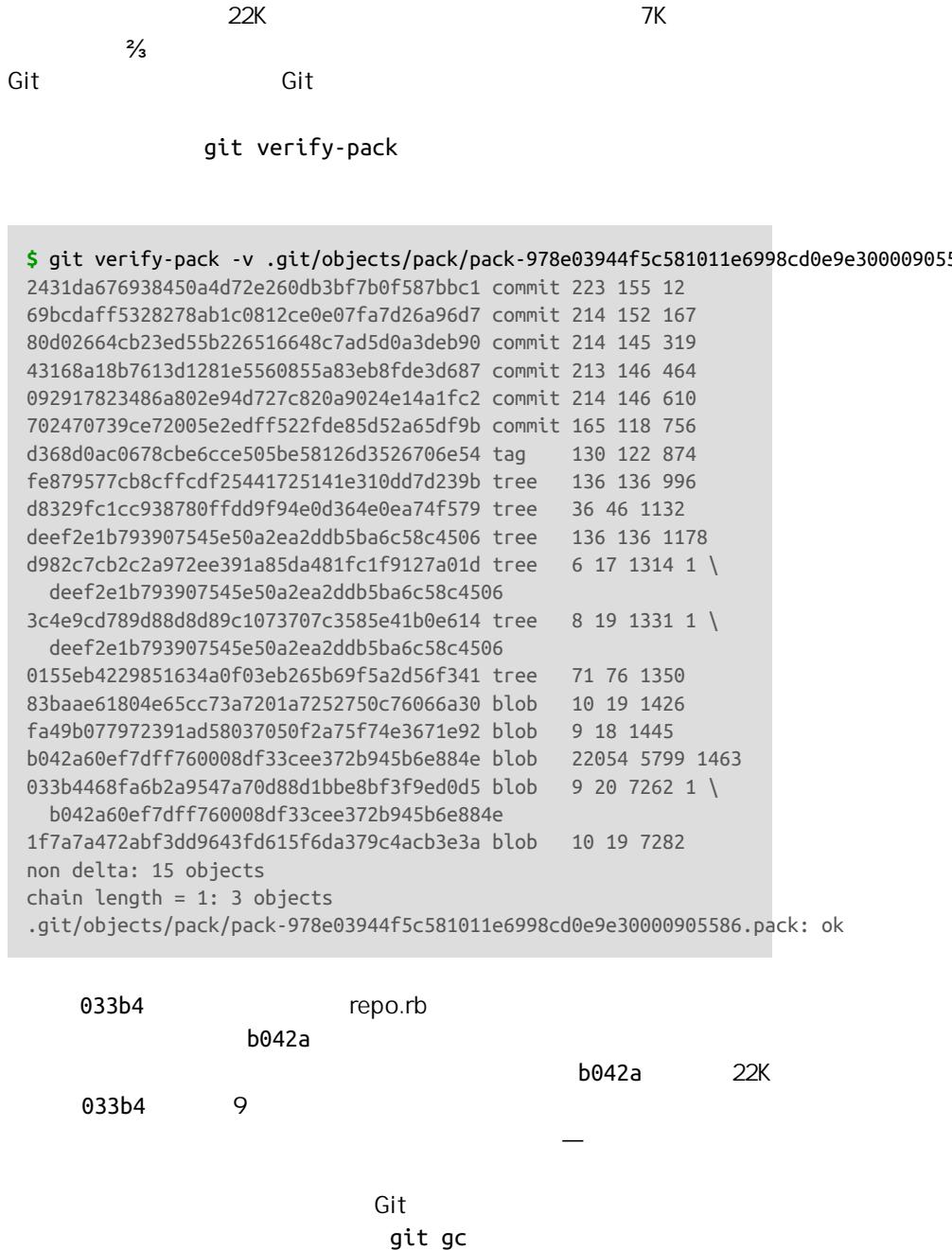
```
$ find .git/objects -type f  
.git/objects/bd/9dbf5aae1a3862dd1526723246b20206e5fc37  
.git/objects/d6/70460b4b4aece5915caf5c68d12f560a9fe3e4  
.git/objects/info/packs  
.git/objects/pack/pack-978e03944f5c581011e6998cd0e9e30000905586.idx  
.git/objects/pack/pack-978e03944f5c581011e6998cd0e9e30000905586.pack
```

“ what is up, doc?” “ test content”

Git

dangling

gc



## 引用规格

```
$ git remote add origin https://github.com/schacon/simplegit-progit
```

.git/config  
origin URL re  
fspec

```
[remote "origin"]  
url = https://github.com/schacon/simplegit-progit  
fetch = +refs/heads/*:refs/remotes/origin/*
```

<src> + <src>:<dst>  
pattern + <dst>  
+ Git

git remote add Git  
refs/heads/ refs/remotes/  
origin/ master

```
$ git log origin/master  
$ git log remotes/origin/master  
$ git log refs/remotes/origin/master
```

Git refs/remotes/  
origin/master master

```
fetch = +refs/heads/master:refs/remotes/origin/master
```

git fetch

master origin/mymaster

```
$ git fetch origin master:refs/remotes/origin/mymaster
```

```
$ git fetch origin master:refs/remotes/origin/mymaster \
    topic:refs/remotes/origin/topic
From git@github.com:schacon/simplegit
! [rejected]      master      -> origin/mymaster (non fast forward)
* [new branch]    topic       -> origin/topic
```



QA                    master                    qa/master

```
$ git push origin master:refs/heads/qa/master
```

Git                git push origin  
                    push

```
[remote "origin"]  
url = https://github.com/schacon/simplegit-progit  
fetch = +refs/heads/*:refs/remotes/origin/*  
push = refs/heads/master:refs/heads/qa/master
```

git push origin                    master  
qa/master

```
$ git push origin :topic
```

<src>:<dst>                    <src>  
topic

## 传输协议

Git                        "                    dumb     "  
                      smart     "

HTTP

Git

Git

HTTP    GET

---

现在已经很少使用哑协议了。使用哑协议的版本库很难保证安全性和私有化，所以大多数 Git 服务器宿主（包括云端和本地）都会拒绝使用它。一般情况下都建议使用智能协议，我们会在后面进行介绍。

---



```
$ git cat-file -p ca82a6dff817ec66f44342007202690a93763949
tree cfda3bf379e4f8dba8717dee55aab78aef7f4daf
parent 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
author Scott Chacon <schacon@gmail.com> 1205815931 -0700
committer Scott Chacon <schacon@gmail.com> 1240030591 -0700
changed the version number
```

cfda3b  
085bb3

```
=> GET objects/08/5bb3bcb608e1e8451d4b2432f8ecbe6306e7e7  
(179 bytes of data)
```

```
=> GET objects/cf/da3bf379e4f8dba8717dee55aab78aef7f4daf  
(404 - Not Found)
```

```
—  
404                   HTTP  
—                      Git
```

```
=> GET objects/info/http-alternates  
(empty file)
```

```
URL                   Git
```

```
—
```

```
objects/info/packs  
update-server-info
```

```
=> GET objects/info/packs  
P pack-816a9b2334da9953e530f27bcac22082a9f5b835.pack
```

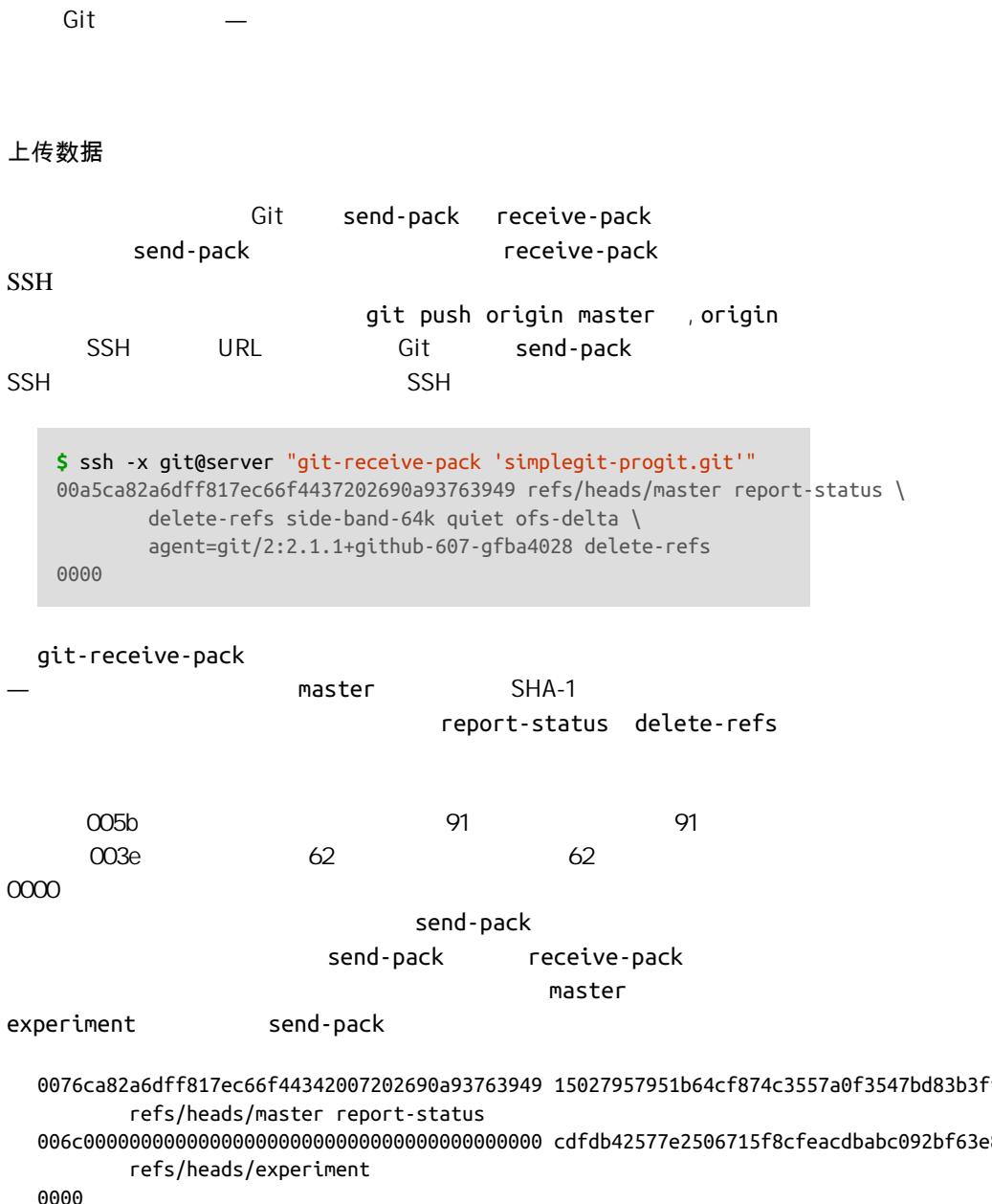
```
=> GET objects/pack/pack-816a9b2334da9953e530f27bcac22082a9f5b835.idx  
(4k of binary data)
```

```
—
```

```
SHA-1
```

```
=> GET objects/pack/pack-816a9b2334da9953e530f27bcac22082a9f5b835.pack  
(13k of binary data)
```

```
Git                   HEAD                   master
```





000eunpack ok



```

=> GET http://server/simplegit-progit.git/info/refs?service=git-receive-pack
001f# service=git-receive-pack
00ab6c5f0e45abd7832bf23074a333f739977c9e8188 refs/heads/master report-status \
  delete-refs side-band-64k quiet ofs-delta \
  agent=git/2:2.1.1~vmpg-bitmaps-bugaloo-608-g116744e
0000
  
```

POST git-upload-pack

=> POST http://server/simplegit-progit.git/git-receive-pack

POST send-pack  
HTTP

### 下载数据

|            |             |
|------------|-------------|
| fetch-pack | upload-pack |
| fetch-pack | upload-pack |

SSH

SSH fetch-pack

```
$ ssh -x git@server "git-upload-pack 'simplegit-progit.git'"
```

fetch-pack upload-pack

```
00dfca82a6dff817ec66f44342007202690a93763949 HEAD multi_ack thin-pack \
    side-band side-band-64k ofs-delta shallow no-progress include-tag \
    multi_ack_detailed symref=HEAD:refs/heads/master \
    agent=git/2:2.1.1+github-607-gfba4028
003fe2409a098dc3e53539a9028a94b6224db9d6a6b6 refs/heads/master
0000
```

|               |  |                         |
|---------------|--|-------------------------|
| receive-pack  |  |                         |
| HEAD          |  | symref=HEAD:refs/heads/ |
| <b>master</b> |  |                         |
| fetch-pack    |  | " want"                 |
| SHA-1         |  | " have"                 |
| SHA-1         |  | " done" upload-pack     |

```
003cwant ca82a6dff817ec66f44342007202690a93763949 ofs-delta
0032have 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
0009done
0000
```

HTTP(S)

HTTP

GET

```
=> GET $GIT_URL/info/refs?service=git-upload-pack
001e# service=git-upload-pack
00e7ca82a6dff817ec66f44342007202690a93763949 HEAD multi_ack thin-pack \
    side-band side-band-64k ofs-delta shallow no-progress include-tag \
    multi_ack_detailed no-done symref=HEAD:refs/heads/master \
    agent=git/2:2.1.1+github-607-gfba4028
003fc82a6dff817ec66f44342007202690a93763949 refs/heads/master
0000
```

SSH git-upload-pack

```
=> POST $GIT_URL/git-upload-pack HTTP/1.0
0032want 0a53e9ddeaddad63ad106860237bbf53411d11a7
0032have 441b40d833fdfa93eb2908e52742248faf0ee993
0000
```

```
multi_ack    side-band
```

Git

## 维护与数据恢复

Git " auto gc"

Git git gc " gc"

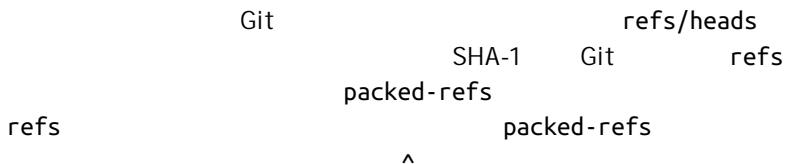
```
$ git gc --auto
```

50 Git 7000  
gc.auto gc.autopacklimit  
gc

```
$ find .git/refs -type f  
.git/refs/heads/experiment  
.git/refs/heads/master  
.git/refs/tags/v1.0  
.git/refs/tags/v1.1
```

git gc refs  
Git .git/packed-refs

```
$ cat .git/packed-refs
# pack-refs with: peeled fully-peeled
cac0cab538b970a37ea1e769cbbde608743bc96d refs/heads/experiment
ab1afef80fac8e34258ff41fc1b867c702daa24b refs/heads/master
cac0cab538b970a37ea1e769cbbde608743bc96d refs/tags/v1.0
9585191f37f7b0fb9444f35a9bf50de191beadc2 refs/tags/v1.1
^1a410efbd13591db07496601ebc7a059dd55cfe9
```



Git

master

```
$ git log --pretty=oneline
ab1afef80fac8e34258ff41fc1b867c702daa24b modified repo a bit
484a59275031909e19aad7c92262719cfcdf19a added repo.rb
1a410efbd13591db07496601ebc7a059dd55cfe9 third commit
cac0cab538b970a37ea1e769cbbde608743bc96d second commit
fdf4fc3344e67ab068f836878b6c4951e3b15f3d first commit
```

master

```
$ git reset --hard 1a410efbd13591db07496601ebc7a059dd55cfe9
HEAD is now at 1a410ef third commit
$ git log --pretty=oneline
1a410efbd13591db07496601ebc7a059dd55cfe9 third commit
cac0cab538b970a37ea1e769cbbde608743bc96d second commit
fdf4fc3344e67ab068f836878b6c4951e3b15f3d first commit
```

```
SHA-1  
SHA-1 -  
          git reflog  
          HEAD  
          reflog  
Git           " Git      "  
  
git update-ref  
          SHA-1  
git reflog
```

```
$ git reflog  
1a410ef HEAD@{0}: reset: moving to 1a410ef  
ab1afef HEAD@{1}: commit: modified repo.rb a bit  
484a592 HEAD@{2}: commit: added repo.rb
```

git log -g

```
$ git log -g  
commit 1a410efbd13591db07496601ebc7a059dd55cfe9  
Reflog: HEAD@{0} (Scott Chacon <schacon@gmail.com>)  
Reflog message: updating HEAD  
Author: Scott Chacon <schacon@gmail.com>  
Date:   Fri May 22 18:22:37 2009 -0700  
  
            third commit  
  
commit ab1afef80fac8e34258ff41fc1b867c702daa24b  
Reflog: HEAD@{1} (Scott Chacon <schacon@gmail.com>)  
Reflog message: updating HEAD  
Author: Scott Chacon <schacon@gmail.com>  
Date:   Fri May 22 18:15:24 2009 -0700  
  
            modified repo.rb a bit
```

recover-branch

ab1afef

```
$ git branch recover-branch ab1afef  
$ git log --pretty=oneline recover-branch  
ab1afef80fac8e34258ff41fc1b867c702daa24b modified repo a bit
```

```
484a59275031909e19aadb7c92262719cfcdf19a added repo.rb
1a410efbd13591db07496601ebc7a059dd55cf9 third commit
cac0cab538b970a37ea1e769cbbde608743bc96d second commit
fdf4fc3344e67ab068f836878b6c4951e3b15f3d first commit
```

recover-branch master

- recover-branch

```
$ git branch -D recover-branch
$ rm -Rf .git/logs/
```

.git/logs/  
git fsck  
--full

```
$ git fsck --full
Checking object directories: 100% (256/256), done.
Checking objects: 100% (18/18), done.
dangling blob d670460b4b4aece5915caf5c68d12f560a9fe3e4
dangling commit ab1afef80fac8e34258ff41fc1b867c702daa24b
dangling tree aea790b9a58f6cf6f2804eeac9f0abbef9631e4c9
dangling blob 7108f7ecb345ee9d0084193f147cdad4d2998293
```

" dangling commit"

Git

git clone

Git

Subversion Perforce Git

```
$ curl https://www.kernel.org/pub/software/scm/git/git-2.1.0.tar.gz > git.tgz
$ git add git.tgz
$ git commit -m 'add git tarball'
[master 7b30847] add git tarball
 1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 git.tgz
```

```
$ git rm git.tgz
rm 'git.tgz'
$ git commit -m 'oops - removed large tarball'
[master dadf725] oops - removed large tarball
 1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100644 git.tgz
```

## gc

```
$ git gc
Counting objects: 17, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (13/13), done.
Writing objects: 100% (17/17), done.
Total 17 (delta 1), reused 10 (delta 0)
```

## count-objects

```
$ git count-objects -v
count: 7
size: 32
```

```
in-pack: 17
packs: 1
size-pack: 4868
prune-packable: 0
garbage: 0
size-garbage: 0
```

|                  |       |
|------------------|-------|
| <b>size-pack</b> | KB    |
| 5MB              | 2KB - |
| 5MB              |       |

|                    |      |
|--------------------|------|
| <b>git gc</b>      | git  |
| <b>verify-pack</b> | tail |

```
$ git verify-pack -v .git/objects/pack/pack-29...69.idx \
| sort -k 3 -n \
| tail -3
dadf7258d699da2c8d89b09ef6670edb7d5f91b4 commit 229 159 12
033b4468fa6b2a9547a70d88d1bbe8bf3f9ed0d5 blob 22044 5792 4977696
82c99a3e86bb1267b236a4b6eff7868d97489af1 blob 4975916 4976258 1438
```

|                 |          |
|-----------------|----------|
| <b>rev-list</b> | 5MB      |
| "               | "        |
| --objects       | rev-list |
| SHA-1           | SHA-1    |

```
$ git rev-list --objects --all | grep 82c99a3
82c99a3e86bb1267b236a4b6eff7868d97489af1 git.tgz
```

```
$ git log --oneline --branches -- git.tgz
dadf725 oops - removed large tarball
7b30847 add git tarball
```

7b30847

Git  
filter-branch

```
" " "  
$ git filter-branch --index-filter \  
'git rm --ignore-unmatch --cached git.tgz' -- 7b30847^..  
Rewrite 7b30847d080183a1ab7d18fb202473b3096e9f34 (1/2)rm 'git.tgz'  
Rewrite dadf7258d699da2c8d89b09ef6670edb7d5f91b4 (2/2)  
Ref 'refs/heads/master' was rewritten
```

--index-filter " " --tree-  
filter

```
git rm --cached " "  
file - - Git --tree-filter  
git rm --ignore-unmatch  
filter-branch  
7b30847
```

.git/refs/original filter-branch

```
$ rm -Rf .git/refs/original  
$ rm -Rf .git/logs/  
$ git gc  
Counting objects: 15, done.  
Delta compression using up to 8 threads.  
Compressing objects: 100% (11/11), done.  
Writing objects: 100% (15/15), done.  
Total 15 (delta 1), reused 12 (delta 0)
```

```
$ git count-objects -v  
count: 11  
size: 4904  
in-pack: 15  
packs: 1
```

```
size-pack: 8
prune-packable: 0
garbage: 0
size-garbage: 0
```

8K 5MB

size

--  
expire git prune

```
$ git prune --expire now
$ git count-objects -v
count: 0
size: 0
in-pack: 15
packs: 1
size-pack: 8
prune-packable: 0
garbage: 0
size-garbage: 0
```

## 环境变量

Git bash shell

shell  
Git

Git

Git

**GIT\_EXEC\_PATH** Git

git-commit, git-diff

git --exec-path

HOME

Git

Git

Git shell

HOME

Git \$PREFIX/etc/

**PREFIX**

gitconfig

**GIT\_CONFIG\_NOSYSTEM**

```
GIT_PAGER  
    PAGER .  
GIT_EDITOR                               Git  
    EDITOR
```

```
Git  
GIT_DIR   .git      .          Git  
        .git           ~   /  
GIT_CEILING_DIRECTORIES     .git
```

```
Git          shell          Git  
GIT_WORK_TREE  
$GIT_DIR  
  GIT_INDEX_FILE  
  GIT_OBJECT_DIRECTORY       .git/objects  
  GIT_ALTERNATE_OBJECT_DIRECTORIES          (   
  /dir/one:/dir/two:...)      Git          GIT_OBJECT_DI-  
RECTORY
```

```
" pathspec          Git  
.gitignore          git add *.c  
GIT_GLOB_PATHSPECS and GIT_NOGLOB_PATHSPECS  
  GIT_GLOB_PATHSPECS      1,  
  ;      GIT_NOGLOB_PATHSPECS      1,  
  *.c            " *.C"          .c  
  :(glob)    :(literal)  
  :(glob)*.c  
GIT_LITERAL_PATHSPECS
```

```
GIT_ICASE_PATHSPECS
```

```
Git          git-commit-tree          git-commit-  
tree
```

```

GIT_AUTHOR_NAME    " author"
GIT_AUTHOR_EMAIL   " author"
GIT_AUTHOR_DATE    " author"
GIT_COMMITTER_NAME " committer"
GIT_COMMITTER_EMAIL " committer"
GIT_COMMITTER_DATE " committer"
    user.email           EMAIL
    Git

```

```

Git      curl      HTTP          GIT_CURL_VERBOSE
        Git
GIT_SSL_NO_VERIFY     Git      SSL
                           HTTPS     Git
        Git
                           GIT_HTTP_LOW_SPEED_LIMIT
GIT_HTTP_LOW_SPEED_TIME     Git
                           http.lowSpeedLimit  http.lowSpeedTime
GIT_HTTP_USER_AGENT       Git      HTTP
        agent            git/2.0.0      user-

```

```

GIT_DIFF_OPTS          -u<n>    --
unified=<n>           git diff
GIT_EXTERNAL_DIFF       diff.external
                        Gitgit diff  Git
GIT_DIFF_PATH_COUNTER  GIT_DIFF_PATH_TOTAL  GIT_EXTER-
NAL_DIFF  diff.external
                        1
GIT_MERGE_VERBOSITY

```

- 0
- 1
- 2
- 3
- 4
- 5

Git                    ?Git

- " true" , " 1" , " 2" -
- / -

### GIT\_TRACE

```
$ GIT_TRACE=true git lga
20:12:49.877982 git.c:554
20:12:49.878369 run-command.c:341
20:12:49.879529 git.c:282
20:12:49.879885 git.c:349
20:12:49.899217 run-command.c:341
20:12:49.899675 run-command.c:192
```

```
trace: exec: 'git-lga'
trace: run_command: 'git-lga'
trace: alias expansion: lga => 'log' '--graph' '--pretty=oneline'
trace: built-in: git 'log' '--graph' '--pretty=oneline'
trace: run_command: 'less'
trace: exec: 'less'
```

### GIT\_TRACE\_PACK\_ACCESS

```
$ GIT_TRACE_PACK_ACCESS=true git status
20:10:12.081397 sha1_file.c:2088
20:10:12.081886 sha1_file.c:2088
20:10:12.082115 sha1_file.c:2088
# [...]
20:10:12.087398 sha1_file.c:2088
20:10:12.087419 sha1_file.c:2088
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
```

```
.git/objects/pack/pack-c3fa...291e.pack 12
.git/objects/pack/pack-c3fa...291e.pack 34662
.git/objects/pack/pack-c3fa...291e.pack 35175
.git/objects/pack/pack-e80e...e3d2.pack 56914983
.git/objects/pack/pack-e80e...e3d2.pack 14303666
```

### GIT\_TRACE\_PACKET

```
$ GIT_TRACE_PACKET=true git ls-remote origin
20:15:14.867043 pkt-line.c:46      packet:      git< # service=git-upload-pack
20:15:14.867071 pkt-line.c:46      packet:      git< 0000
20:15:14.867079 pkt-line.c:46      packet:      git< 97b8860c071898d9e162678ea1035a
20:15:14.867088 pkt-line.c:46      packet:      git< 0f20ae29889d61f2e93ae00fd34f1c
```

```
20:15:14.867094 pkt-line.c:46          packet:      git< 36dc827bc9d17f80ed4f
# [...]
```

**GIT\_TRACE\_PERFORMANCE**

Git

```
$ GIT_TRACE_PERFORMANCE=true git gc
20:18:19.499676 trace.c:414           performance: 0.374835000 s: git command: '
20:18:19.845585 trace.c:414           performance: 0.343020000 s: git command: '
Counting objects: 170994, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (43413/43413), done.
Writing objects: 100% (170994/170994), done.
Total 170994 (delta 126176), reused 170524 (delta 125706)
20:18:23.567927 trace.c:414           performance: 3.715349000 s: git command: '
20:18:23.584728 trace.c:414           performance: 0.000910000 s: git command: '
20:18:23.605218 trace.c:414           performance: 0.017972000 s: git command: '
20:18:23.606342 trace.c:414           performance: 3.756312000 s: git command: '
Checking connectivity: 170994, done.
20:18:25.225424 trace.c:414           performance: 1.616423000 s: git command: '
20:18:25.232403 trace.c:414           performance: 0.001051000 s: git command: '
20:18:25.233159 trace.c:414           performance: 6.112217000 s: git command: '
```

**GIT\_TRACE\_SETUP** Git

```
$ GIT_TRACE_SETUP=true git status
20:19:47.086765 trace.c:315           setup: git_dir: .git
20:19:47.087184 trace.c:316           setup: worktree: /Users/ben/src/git
20:19:47.087191 trace.c:317           setup: cwd: /Users/ben/src/git
20:19:47.087194 trace.c:318           setup: prefix: (null)
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
```

**GIT\_SSH** Git SSH

ssh

\$GIT\_SSH [username@]host [-p &lt;port&gt;] &lt;command&gt;

ssh

GIT\_SSH

~/.ssh/config

```
GIT_ASKPASS      core.askpass          Git
                  ( " " _ )           stdout
GIT_NAMESPACE    --namespace
fork,
GIT_FLUSH        Git               I/O
1   Git           0
GIT_REFLOG_ACTION reflog
```

```
$ GIT_REFLOG_ACTION="my action" git commit --allow-empty -m 'my message'
[master 9e3d55a] my message
$ git reflog -1
9e3d55a HEAD@{0}: my action: my message
```

## 总结

```
Git
Git
Git
Git
Git
Git
Git
```



# 其它环境中的 Git A

Git

Git

Git

Git

## 图形界面

Git

Git

Git

" " "

**gitk git-gui**

Git  
gitk  
git grep

Gitk

gitk git-gui  
git log

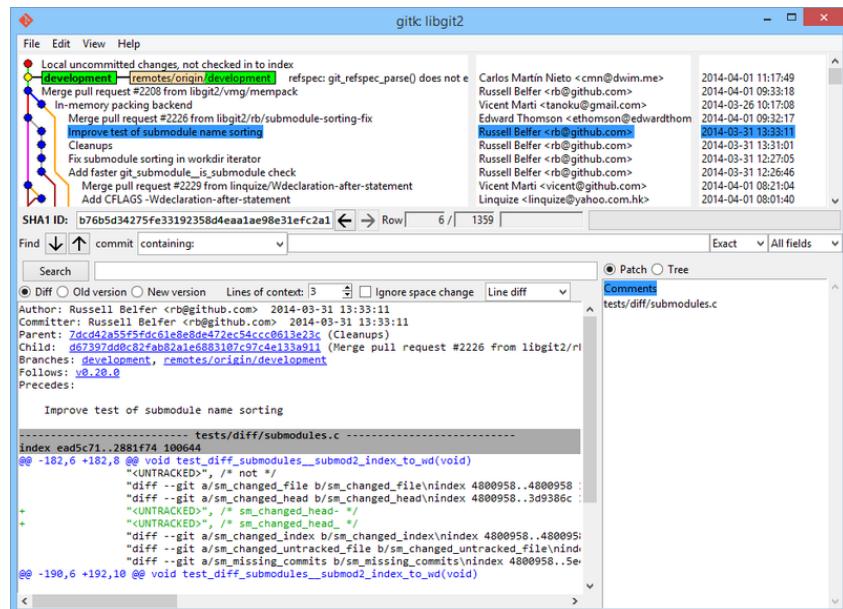
cd Git

**Figure 1-1.**

gitk 历史查看器。

```
$ gitk [git log options]
```

Gitk  
log            --all            ,            gitk  
HEAD        Gitk



git log --graph

HEAD

git-gui

```
$ git gui
```

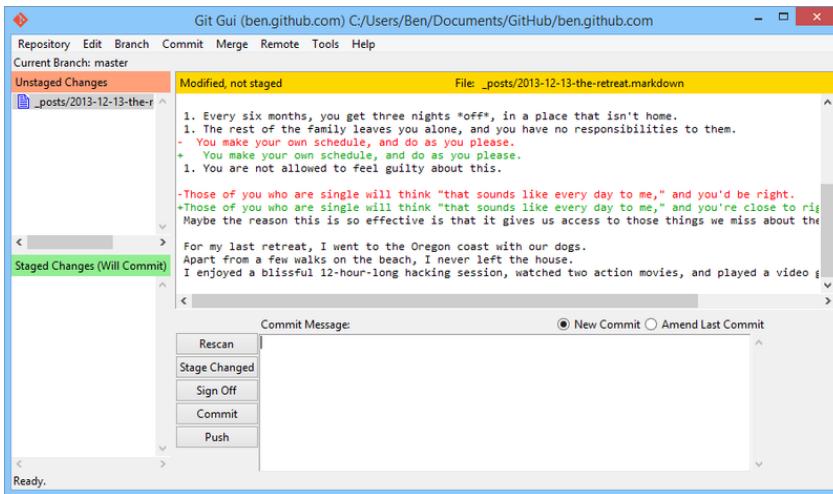
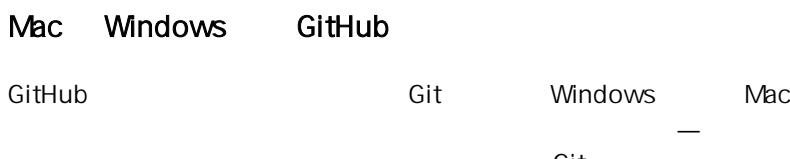


Figure 1-2.

git-gui 提交工具。

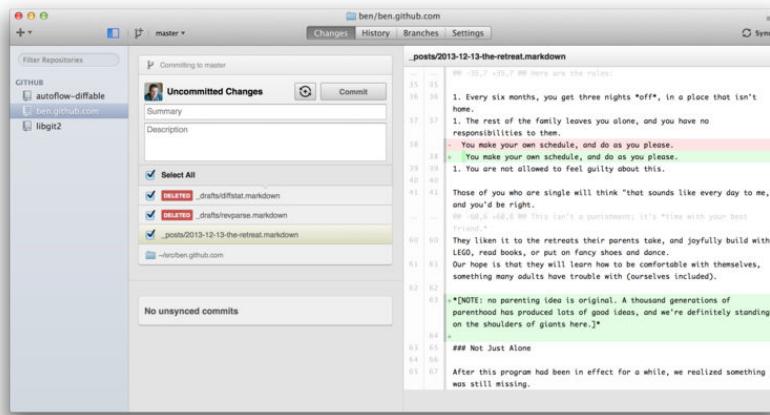
dif

" " git commit  
" " "  
"  
"  
"  
"  
gitk git-gui



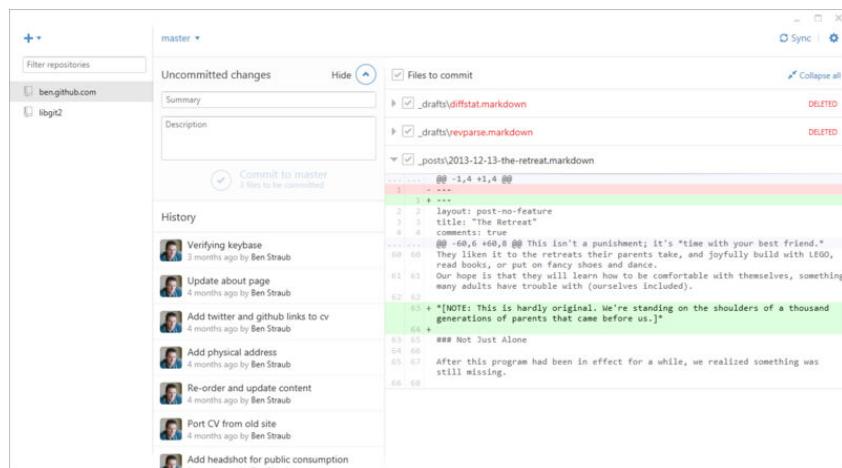
**Figure 1-3.**

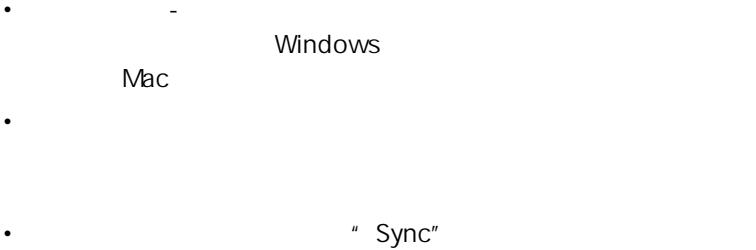
GitHub Mac 客户端。



**Figure 1-4.**

GitHub Windows 客户端。



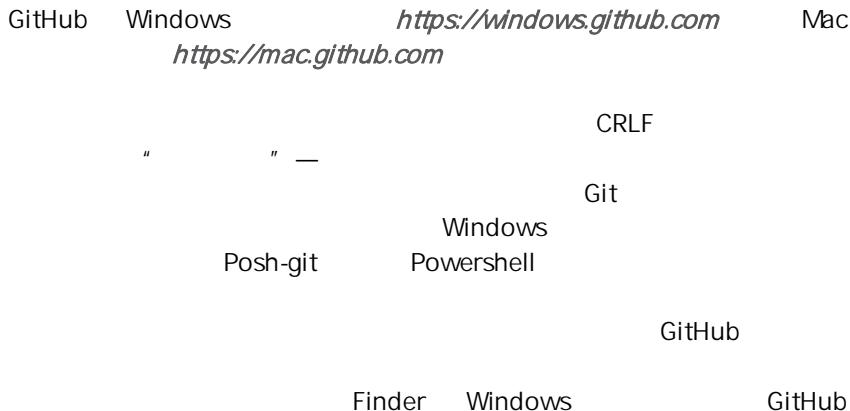


---

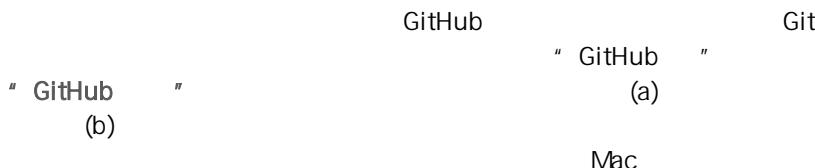
你不需要注册 GitHub 账号也可以使用这些工具。尽管它们是按照 GitHub 推荐的工作流程来设计的，并突出提升了一些 GitHub 的服务体验，但它们可以在任何 Git 仓库上工作良好，也可以通过网络连接到任意 Git 主机。

---

## 安装

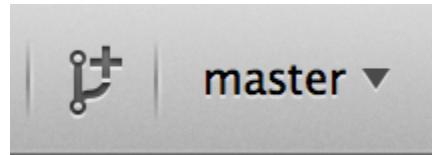


## 推荐的工作流程



**Figure 1-5.**

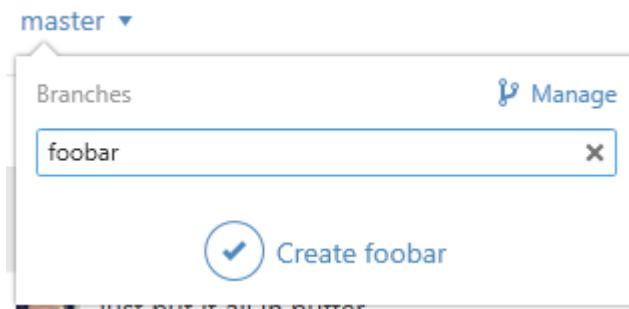
Mac 上的“创建分支”按钮。



Windows

**Figure 1-6.**

在 Windows 上创建分支。



GitHub

" " ctrl-enter -enter  
" " push  
fetch merge rebase Git , GitHub

1. git pull --rebase  
      git pull --no-rebase  
2. git push

## 小结

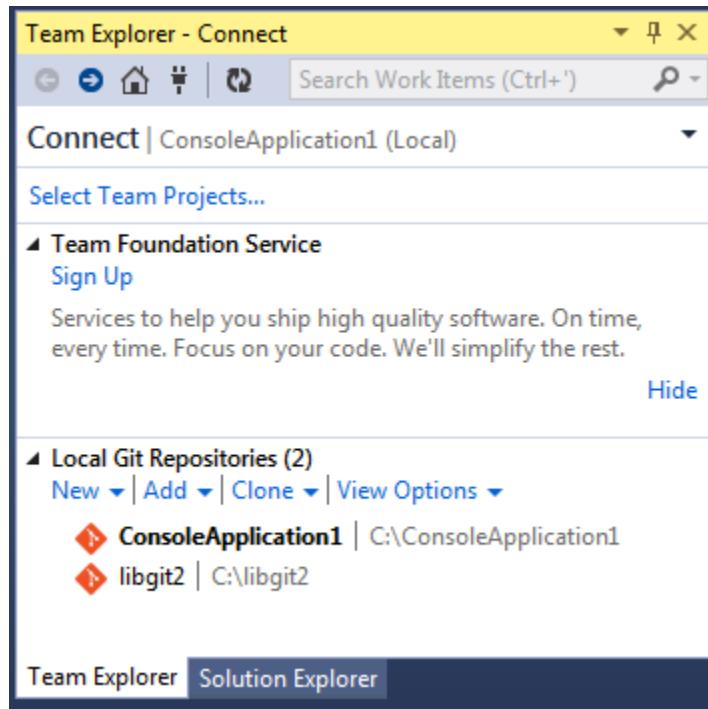
Git  
Git  
<http://git-scm.com/downloads/guis> Git  
[https://git.wiki.kernel.org/index.php/Interfaces,\\_frontends,\\_and\\_tools#Graphical\\_Interfaces](https://git.wiki.kernel.org/index.php/Interfaces,_frontends,_and_tools#Graphical_Interfaces).

## Visual Studio 中的 Git



**Figure 1-7.**

从 Team Explorer 中  
连接 Git 仓库。



Visual Studio

Figure A-8

Git

" Add"

Git

" Home"

Git

" Changes"

" Unsynced Commits"

" Branches"

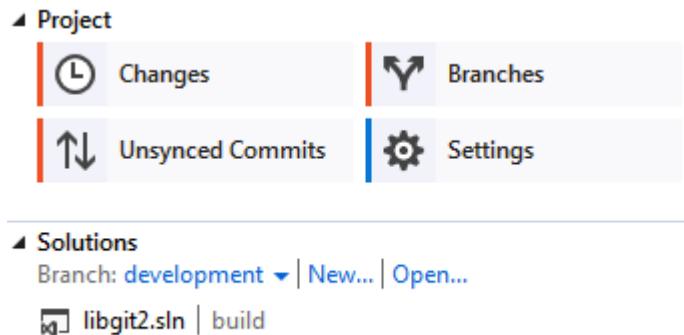


Figure 1-8.

Visual Studio 中的  
Git 仓库的“Home”  
视图。

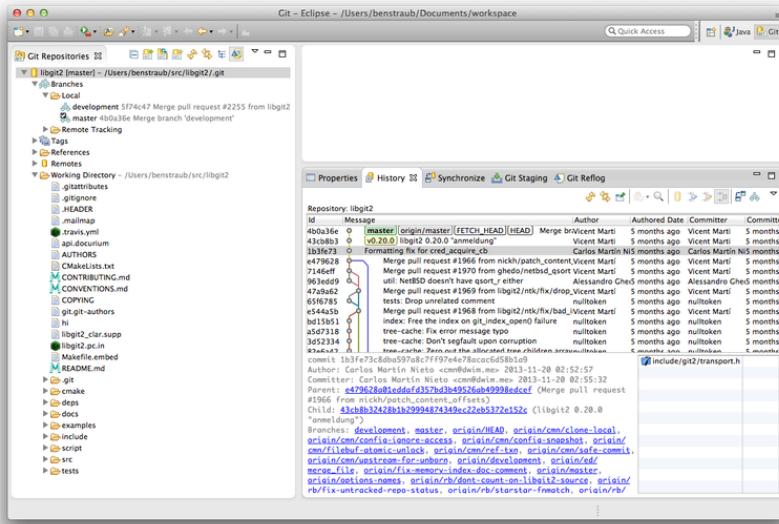
Visual Studio  
dif  
[http://msdn.microsoft.com/  
en-us/library/hh850437.aspx](http://msdn.microsoft.com/en-us/library/hh850437.aspx)

## Eclipse 中的 Git

Eclipse  
spective > Other...      Egit  
                                Git  
                                " Git"  
                                Git  
(Window > Open Per

**Figure 1-9.**

Eclipse 中 EGit 的界面环境。

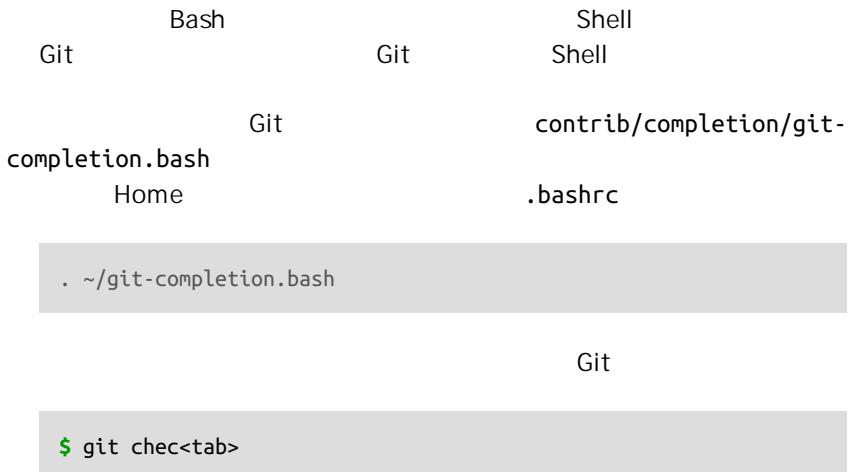


## EGit

Help > Help Contents  
tion”

“ EGit Documentation

## Bash 中的 Git



```

....      Bash          git checkout
          Git           prompt
Git

prompt      Git          con-
trib/completion/git-prompt.sh   Home   (
)           .bashrc

.

. ~/git-prompt.sh
export GIT_PS1_SHOWDIRTYSTATE=1
export PS1='\w$(__git_ps1 " (%s)")\$ '


\w          \$      prompt
__git_ps1 " (%s)"      %s      git-prompt.sh
                        Bash

prompt     Git

```

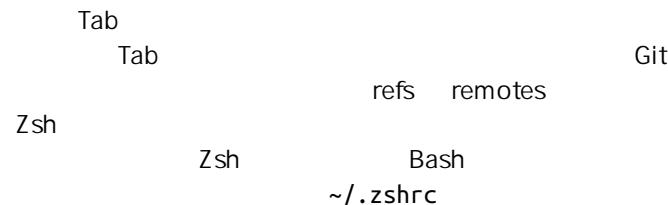
Figure 1-10.  
自定义的 bash 提示符 (prompt).

git-completion.bash  
git-prompt.sh

## Zsh 中的 Git

Git Zsh Tab contrib/completion/git-
completion.zsh home .zshrc source
Bash Zsh

```
$ git che<Tab>
check-attr      -- 显示 gitattributes 信息
check-ref-format -- 检查引用名称是否符合规范
checkout        -- 从工作区中检出分支或路径
checkout-index   -- 从暂存区拷贝文件至工作目录
cherry          -- 查找没有被合并至上游的提交
cherry-pick     -- 从一些已存在的提交中应用更改
```

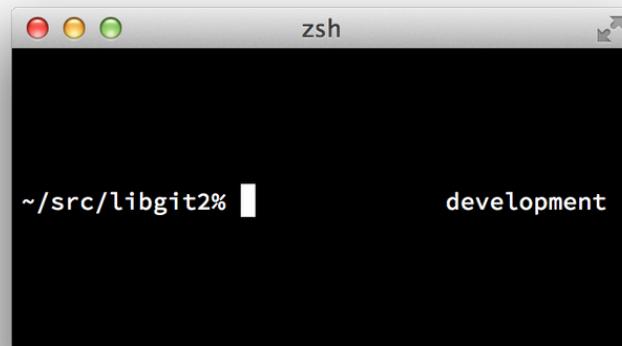


```
setopt prompt_subst
. ~/git-prompt.sh
export RPROMPT=$'$(__git_ps1 "%s")'
```

Git

**Figure 1-11.**

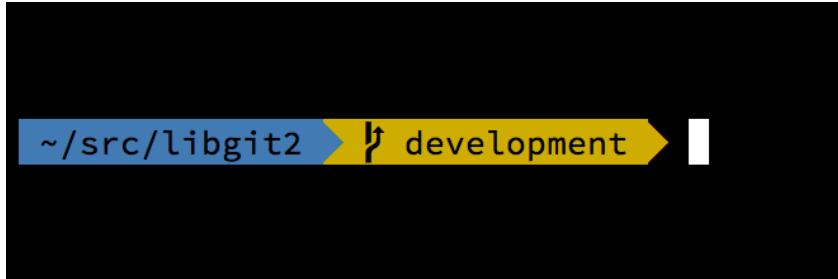
自定义 zsh 提示符.



Zsh  
ussell/oh-my-zsh " oh-my-zsh" https://github.com/robbbyr  
" " Git Tab

**Figure 1-12.**

## 一个oh-my-zsh 主题的示例.



## Powershell 中的 Git

Windows (cmd.exe) Git

Powershell

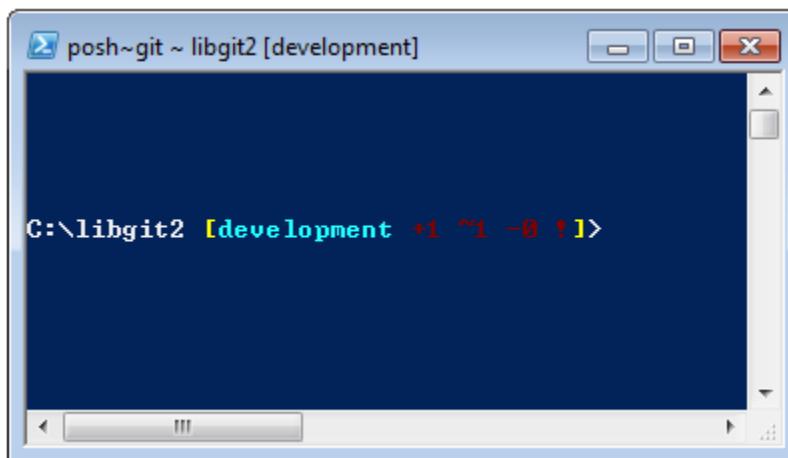
(<https://github.com/dahlbyk/posh-git>)

Git

Posh-Git  
ab

**Figure 1-13.**

附带了 Posh-Git 扩展包的 Powershell。



```
Windows          GitHub   Posh-Git
profile.ps1      (       C:\Users\<user-
name>\Documents\WindowsPowerShell);

. (Resolve-Path "$env:LOCALAPPDATA\GitHub\shell.ps1")
. $env:github_posh_git\profile.example.ps1

Windows          GitHub   (https://github.com/
dahlbyk\posh-git)    Posh-Git   WindowsPower -
shell           Powershell

> Set-ExecutionPolicy RemoteSigned -Scope CurrentUser -Confirm
> cd ~\Documents\WindowsPowerShell\posh-git
> .\install.ps1

profile.ps1          Posh-Git
```

## 总结

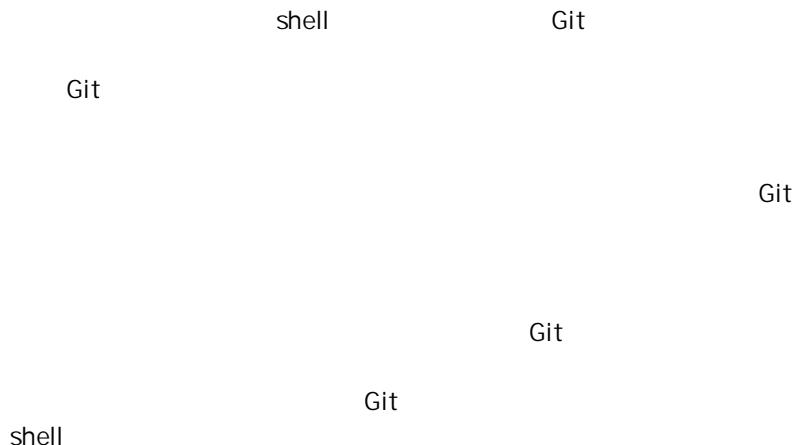
Git  
Git

B

# 将 Git 嵌入你的应用



## 命令行 Git 方式



## Libgit2

```
Libgit2    Libgit2      Git          Git  
          Git          API          http://  
libgit2github.com          C API  
  
// 打开一个版本库  
git_repository *repo;  
int error = git_repository_open(&repo, "/path/to/repository");  
  
// 逆向引用 HEAD 到一个提交  
git_object *head_commit;  
error = git_revparse_single(&head_commit, repo, "HEAD^{commit}");  
git_commit *commit = (git_commit*)head_commit;  
  
// 显示这个提交的一些详情  
printf("%s", git_commit_message(commit));  
const git_signature *author = git_commit_author(commit);  
printf("%s <%s>\n", author->name, author->email);  
const git_oid *tree_id = git_commit_tree_id(commit);  
  
// 清理现场  
git_commit_free(commit);  
git_repository_free(repo);  
  
Git          git_repository  
  
          .git  
git_repository_open_ext          git_clone  
                                  git_repository_init  
  
          rev-parse          ""          "  
HEAD          git_object  
          ""          ""          git_object  
          Git          git_object  
          ""          ""          git_object  
  
          git_object_type(commit)          GIT_OBJ_COMMIT  
          git_commit  
                                  git_oid  
          Libgit2          SHA-1
```

- Libgit2
  - int 0
- Libgit2
- Libgit2 const
- C
  - Libgit2 C
- Git
  - Rugged
- Libgit2 Ruby
- <https://github.com/libgit2/rugged>

```

repo = Rugged::Repository.new('path/to/repository')
commit = repo.head.target
puts commit.message
puts "#{commit.author[:name]} <#{commit.author[:email]}>"
tree = commit.tree

Rugged
ConfigError ObjectError
Ruby

```

```

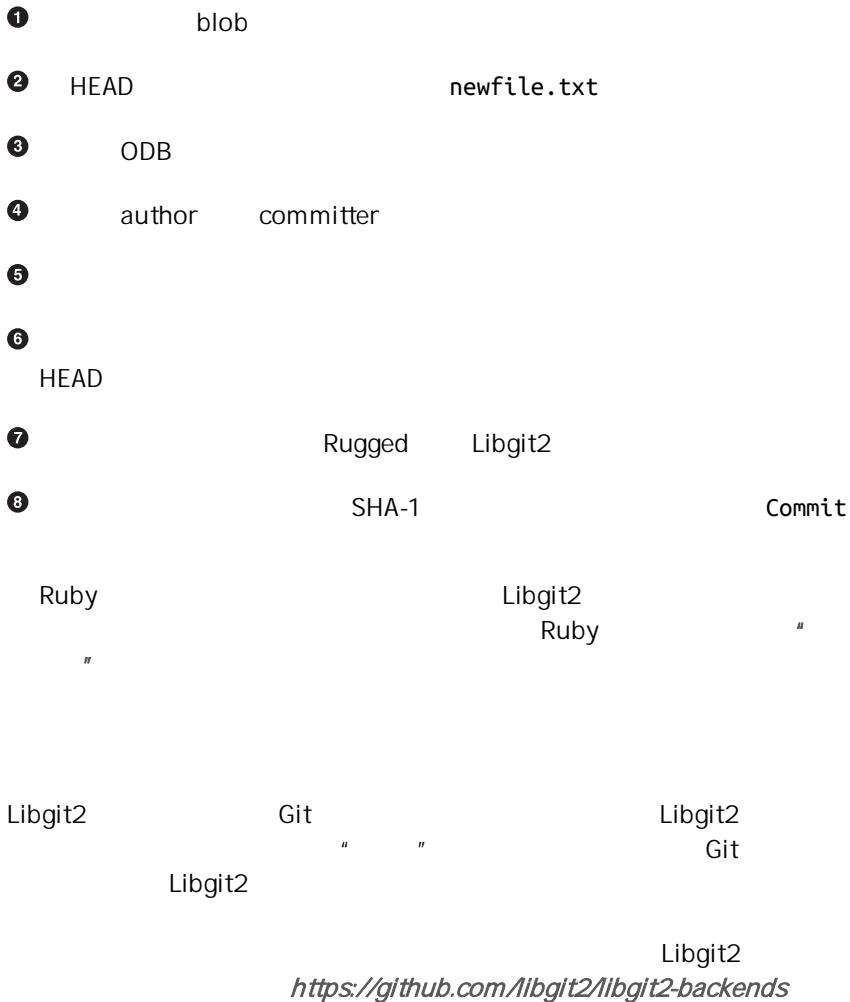
blob_id = repo.write("Blob contents", :blob) ❶

index = repo.index
index.read_tree(repo.head.target.tree)
index.add(:path => 'newfile.txt', :oid => blob_id) ❷

sig = {
  :email => "bob@example.com",
  :name => "Bob User",
  :time => Time.now,
}

commit_id = Rugged::Commit.create(repo,
  :tree => index.write_tree(repo), ❸
  :author => sig,
  :committer => sig, ❹
  :message => "Add newfile.txt", ❺
  :parents => repo.empty? ? [] : [repo.head.target].compact, ❻
  :update_ref => 'HEAD', ❼
)
commit = repo.lookup(commit_id) ❽

```



```

git_odb *odb;
int error = git_odb_new(&odb); ①

git_odb_backend *my_backend;
error = git_odb_backend_mine(&my_backend, /*...*/); ②

error = git_odb_add_backend(odb, my_backend, 1); ③

git_repository *repo;
error = git_repository_open(&repo, "some-path");
error = git_repository_set_odb(odb); ④

```

```

(
)

①          ODB      "      "
"          "

②          ODB

③

④          ODB

    git_odb_backend_mine
ODB
    git_odb_backend

typedef struct {
    git_odb_backend parent;

    // 其它的一些东西
    void *custom_context;
} my_backend_struct;

int git_odb_backend_mine(git_odb_backend **backend_out, /*...*/)
{
    my_backend_struct *backend;

    backend = calloc(1, sizeof (my_backend_struct));

    backend->custom_context = ...;

    backend->parent.read = &my_backend__read;
    backend->parent.read_prefix = &my_backend__read_prefix;
    backend->parent.read_header = &my_backend__read_header;
    // .....

    *backend_out = (git_odb_backend *) backend;

    return GIT_SUCCESS;
}

my_backend_struct          git_odb_backend
                           Libgit2

```

parent    Libgit2        include/git2/sys/

odb\_backend.h

Libgit2

Go Node.js Erlang JVM

<https://github.com/libgit2/>

HEAD

( git log -1 )

C++

**LIBGIT2SHARP**

.NET Mono  
[github.com/libgit2/libgit2sharp](https://github.com/libgit2/libgit2sharp))

LibGit2Sharp (<https://github.com/libgit2/libgit2sharp>)

C#

CLR API

Lib

git2

```
new Repository(@"C:\path\to\repo").Head.Tip.Message;
```

Windows

NuGet

**OBJECTIVE-GIT**

Apple  
Objective-Git (<https://github.com/libgit2/objective-git>)  
Libgit2

Objective-C

```
GTRepository *repo =  
    [[[GTRepository alloc] initWithURL:[NSURL fileURLWithPath:@" /path/to/repo"] error:&error]  
    NSString *msg = [[[repo headReferenceWithError:NULL] resolvedTarget] message];
```

Objective-git Swift

Objective-C

**PYGIT2**

Python Libgit2

Pygit2

<http://www.pygit2.org/>

```
pygit2.Repository("/path/to/repo") # 打开版本库  
    .head # get the current branch  
    .peel(pygit2.Commit) # walk down to the commit  
    .message # read the message
```

Libgit2  
Libgit2 API <https://libgit2.github.com/>  
*libgit2* <https://libgit2.github.com/docs>.  
README

## JGit

JGit Java JGit Git Git Git Java  
JGit Java JGit Eclipse Git Java  
<http://www.eclipse.org/jgit>

JGit Maven pom.xml <dependencies>

```
<dependency>
  <groupId>org.eclipse.jgit</groupId>
  <artifactId>org.eclipse.jgit</artifactId>
  <version>3.5.0.201409260305-r</version>
</dependency>
```

version <http://mvnrepository.com/artifact/org.eclipse.jgit/org.eclipse.jgit>  
Maven  
JGit  
<http://www.eclipse.org/jgit/download> JGit

```
javac -cp .:org.eclipse.jgit-3.5.0.201409260305-r.jar App.java
java -cp .:org.eclipse.jgit-3.5.0.201409260305-r.jar App
```

```

JGit API
Git      JGit
          API
          Git
          API
          Repository
          JGit
          FileRepositoryBuilder

// 创建一个新仓库
Repository newlyCreatedRepo = FileRepositoryBuilder.create(
    new File("/tmp/new_repo/.git"));
newlyCreatedRepo.create();

// 打开一个存在的仓库
Repository existingRepo = new FileRepositoryBuilder()
    .setGitDir(new File("my_repo/.git"))
    .build();

builder     API
.readEn-   .setWork-
vironment() .git
Tree(...).findGitDir() , .git

Repository

// 获取引用
Ref master = repo.getRef("master");

// 获取该引用所指向的对象
ObjectId masterTip = master.getObjectId();

// Rev-parse
ObjectId obj = repo.resolve("HEAD^{tree}");

// 装载对象原始内容
ObjectLoader loader = repo.open(masterTip);
loader.copyTo(System.out);

// 创建分支
RefUpdate createBranch1 = repo.updateRef("refs/heads/branch1");
createBranch1.setNewObjectId(masterTip);
createBranch1.update();

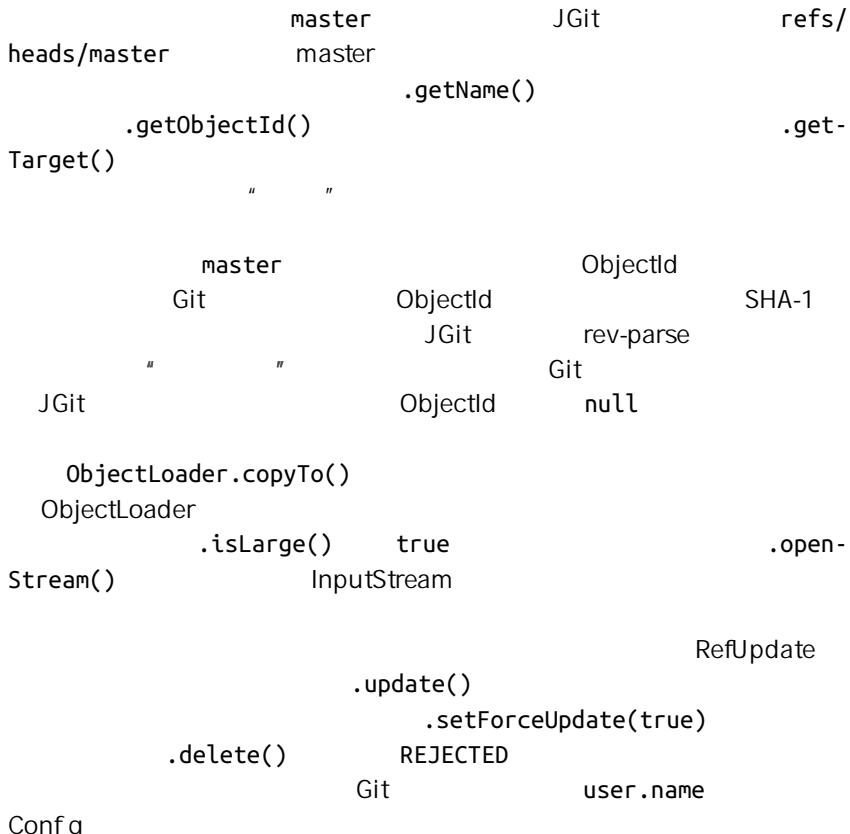
// 删除分支

```

```
RefUpdate deleteBranch1 = repo.updateRef("refs/heads/branch1");
deleteBranch1.setForceUpdate(true);
deleteBranch1.delete();
```

// 配置

```
Config cfg = repo.getConfig();
String name = cfg.getString("user", null, "name");
```



API

JGit

API

API

Git

```
Repository repo;  
// 构建仓库。。。  
Git git = new Git(repo);
```

Git

—

git ls-remote

```
CredentialsProvider cp = new UsernamePasswordCredentialsProvider("username", "p4ssw0rd");  
Collection<Ref> remoteRefs = git.lsRemote()  
    .setCredentialsProvider(cp)  
    .setRemote("origin")  
    .setTags(true)  
    .setHeads(false)  
    .call();  
for (Ref ref : remoteRefs) {  
    System.out.println(ref.getName() + " -> " + ref.getObjectId().name());  
}
```

Git

.call()

origin

CredentialsProvider

Git

add blame

commit clean push rebase revert reset

JGit

- JGit API  
*http://download.eclipse.org/jgit/docs/latest/apidocs*  
Javadoc  
JVMIDE
- JGit Cookbook    *https://github.com/centic9/jgit-cookbook*  
JGit
- *http://stackoverflow.com/questions/6861881*

# Git 命令

C

Git

## 设置与配置

Git  
config help

git config

Git  
Git  
Git

git config  
" " Git " " Git  
" " Git " " Git  
" " " git pull --rebase  
" " " " HTTP  
" " " " Git  
smudge clean

"Git"

## git help

`git help` Git

`git help <command>`

```
git help
```

git shell

## 获取与创建项目

Git

git init

git init

Git

Git

$\mu$        $\mu$

git clone

git clone

git init

Git

add URL

## origin

## git remote

git checkout

git clone

Git

- -hare

Git

“ ” “ ” --  
recursive

## 快照基础

### git add

```
git add                               index  
          git commit  
git add  
          Git  
  
          “ ” “ ” git add  
          “ ” “ ”  
          “ ” “ ”  
          “ ” “ ”
```

### git status

```
git status
```

```
“ ” “ ” status  
status  
          git
```

### git dif

```
          git diff  
          git diff  
          git diff --staged  
git diff master branchB
```

git diff

--check

git diff A...B

-b  
--theirs --ours --base

--submodule

## git dif tool

git diff git difftool

## git commit

git commit git add

-a git add

-m  
--amend

git commit -S

git commit

## git reset

git reset HEAD index --hard

```
          "          "          git reset  
          "          "          git add  
          "          "          git reset --hard  
          "          "          git merge --abort      git reset
```

## git rm

```
git rm   Git  
          git add  
          "          "          git rm  
          "          "          --cached  
          "          "          git rm  
          git filter-branch          --ignore-unmatch
```

## git mv

```
git mv  
add          git rm  
          "          "
```

## git clean

```
git clean  
          "          "          clean
```

## 分支与合并

Git

## git branch

```
git branch
```

## Chapter 3

branch

git branch -u

Git

## git checkout

git checkout

git branch

--track

--conflict=diff3

git reset

HEAD

## git merge

git merge

git merge

merge

git

merge <branch>

squashed merge

Git

Xignore-space-change

--abort

GPG

## git mergetool

Git

git mergetool

## git log

git log

```
git log --pretty=oneline --decorate --graph  
git log branchA..branchB  
git log <_merge_log>> --left-right --merge  
git log --cc  
git log -g  
git log -S -L  
git log --show-signature
```

## git stash

git stash

git stash

**git tag**

**git tag**

“ “ “  
“ “ “ -s GPG  
“ “ -v

## 项目分享与更新

Git

**git fetch**

**git fetch**

“ “ “  
“ “ “  
“ “ “  
“ “ “  
“ “ “  
“ “ “  
“ “ “ refspec git  
fetch

**git pull**

**git pull**

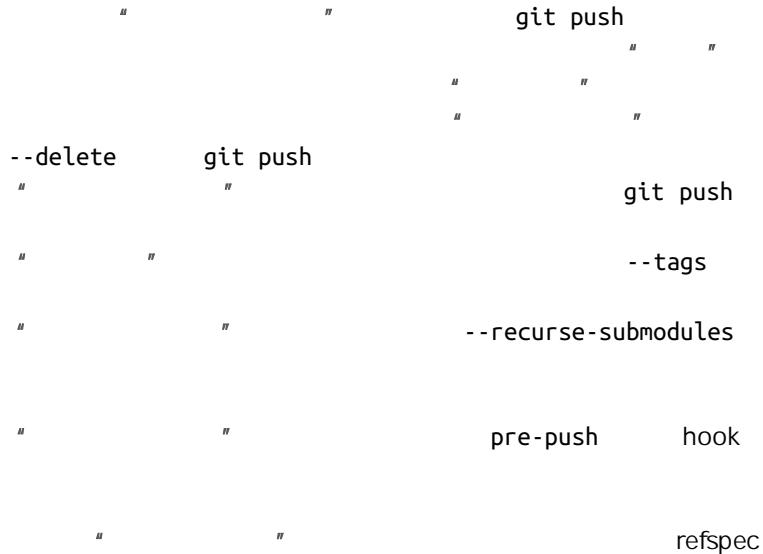
git fetch git merge

Git

“ “ “  
“ “ “  
“ “ “  
“ “ “  
“ “ “ URL  
“ “ “ --  
verify-signatures GPG

## **git push**

**git push**



## **git remote**

**git remote**

    URL                          origin  
                                git remote

    "                          "

add <name> <url>                          git remote

## **git archive**

**git archive**

    "                          "

                                git archive

**git submodule**

git submodule  
submodule , add up-  
date sync  
" " "

## 检查与比较

**git show**

git show Git

" " "  
" " "

git show " "  
" " "

**git shortlog**

git shortlog git log  
git log

" " "

changelog

**git describe**

git describe

SHA-1  
" " " " " git de-  
scribe

## 调试

Git

**git bisect**

git bisect

bug

" "

**git blame**

git blame

" "

**git grep**

git grep

" Git Grep"

**补丁**

Git

**git cherry-pick**

git cherry-pick

" "

Cherry picking

**git rebase**

git rebase

cherry-pick

picks

" "

cherry-

" "

--onto

" Rerere"  
" " -i

git revert

git revert

git cherry-pick

" " "

邮件

Git

Git

,Git

git apply

git apply

git diff

GNU dif

" " "

git am

git am

mbox

" am " "

--resolved -i -3

" " "

hooks

git am

" " "

GitHub

git format-patch

git format-patch

mbox

patch

### git imap-send

git imap-send                  git format-patch                  IMAP  
"                                "                                 git  
imap-send

### git send-email

git send-mail                  git format-patch  
"                                "  
email                            git send-  
email

### git request-pull

git request-pull

"                                "  
"                                git request-pull

## 外部系统

Git

### git svn

git svn                  Git                          Subversion  
"                            Git                            Subversion  
"    Git    Subversion"

### git fast-import

import                          Git                            git fast-  
"                                "  
"                                "

## 管理

Git  
Git

**git gc**

git gc " garbage collection"

" "

**git fsck**

git fsck

" "

dangling object

**git reflog**

git reflog

" "

git log -g git log

" "

**git filter-branch**

git filter-branch

"

"

--commit-filter --subdirectory-filter --  
tree-filter  
" Git-p4" " TFS"

## 底层命令

" " ls-remote  
" " Rerere" " "  
files " rev-parse  
" SHA-1  
Chapter 10



# Index

## Symbols

\$EDITOR, 350  
\$VISUAL  
    see \$EDITOR, 350  
.gitignore, 352  
.NET, 502

## A

aliases, 73  
Apache, 130  
Apple, 502  
archiving, 366  
attributes, 360  
autocorrect, 352

## B

bash, 492  
binary files, 361  
BitKeeper, 31  
bitnami, 134  
branches, 75  
    basic workflow, 83  
    creating, 77  
    deleting remote, 105  
    diffing, 169  
    long-running, 93  
    managing, 91  
    merging, 87  
    remote, 96, 169  
    switching, 79  
    topic, 94, 165  
    tracking, 104  
        upstream, 104  
build numbers, 178

## C

C#, 502

Cocoa, 502

color, 353  
commit templates, 350  
contributing, 142  
    private managed team, 152  
    private small team, 145  
    public large project, 161  
    public small project, 157  
credential caching, 37  
credentials, 343  
CRLF, 37  
crlf, 357  
CVS, 28

## D

dif tool, 354  
distributed git, 139

## E

Eclipse, 491  
editor  
    changing default, 51  
email, 163  
    applying patches from, 165  
excludes, 352, 442

## F

files  
    moving, 54  
    removing, 53  
forking, 141, 187

## G

Git as a client, 381  
git commands  
    add, 44, 45, 45  
    am, 166

apply, 165  
archive, 178  
branch, 77, 91  
checkout, 79  
cherry-pick, 175  
clone, 42  
    bare, 122  
commit, 51, 76  
config, 38, 40, 51, 73, 163, 349  
credential, 343  
daemon, 128  
describe, 178  
dif, 48  
    check, 143  
fast-import, 432  
fetch, 66  
fetch-pack, 467  
filter-branch, 430  
format-patch, 162  
gitk, 483  
gui, 483  
help, 40, 128  
http-backend, 130  
init, 41, 45  
    bare, 123, 126  
instaweb, 132  
log, 55  
merge, 85  
    squash, 161  
mergetool, 90  
p4, 408, 429  
pull, 66  
push, 67, 72, 102  
rebase, 107  
receive-pack, 466  
remote, 64, 65, 67, 68  
request-pull, 158  
rerere, 176  
send-pack, 466  
shortlog, 179  
show, 71  
show-ref, 384  
status, 43, 51  
svn, 381  
tag, 69, 70, 71  
upload-pack, 467  
git-svn, 381  
git-tf, 415  
git-tfs, 415  
GitHub, 181  
    API, 227

Flow, 187  
organizations, 219  
pull requests, 190  
user accounts, 181  
GitHub for Mac, 485  
GitHub for Windows, 485  
gitk, 483  
GitLab, 133  
GitWeb, 131  
GPG, 351  
Graphical tools, 483  
GUIs, 483

## H

hooks, 368  
    post-update, 120

## I

Importing  
    from Mercurial, 426  
    from others, 432  
    from Perforce, 428  
    from Subversion, 424  
    from TFS, 430  
integrating work, 171  
Interoperation with other VCSs  
    Mercurial, 392  
    Perforce, 400  
    Subversion, 381  
    TFS, 415  
IRC, 40

## J

java, 503  
jgit, 503

## K

keyword expansion, 363

## L

libgit2, 498  
line endings, 357  
Linus Torvalds, 31  
Linux, 31  
    installing, 35  
log filtering, 61  
log formatting, 58

## M

Mac  
installing, 36  
maintaining a project, 164  
master, 77  
Mercurial, 392, 426  
mergetool, 354  
merging, 87  
  conflicts, 89  
  strategies, 368  
  vs. rebasing, 115  
Migrating to Git, 424  
Mono, 502

## O

Objective-C, 502  
origin, 97

## P

pager, 351  
Perforce, 28, 31, 400, 428  
  Git Fusion, 400  
policy example, 372  
posh-git, 495  
Powershell, 37  
powershell, 495  
protocols  
  dumb HTTP, 119  
  git, 122  
  local, 117  
  smart HTTP, 119  
  SSH, 121  
pulling, 105  
pushing, 102  
Python, 502

## R

rebasing, 106  
  perils of, 111  
  vs. merging, 115  
references  
  remote, 96  
releasing, 178  
rerere, 176  
Ruby, 499

## S

serving repositories, 117

git protocol, 128  
GitLab, 133  
GitWeb, 131  
HTTP, 130  
SSH, 124  
SHA-1, 33  
shell prompts  
  bash, 492  
  powershell, 495  
  zsh, 493  
SSH keys, 124  
  with GitHub, 182  
staging area  
  skipping, 52  
Subversion, 28, 31, 140, 381, 424

## T

tab completion  
  bash, 492  
  powershell, 495  
  zsh, 493  
tags, 69, 177  
  annotated, 70  
  lightweight, 70  
  signing, 177  
TFS, 415, 430  
TFVC (see TFS)

## V

version control, 27  
  centralized, 28  
  distributed, 29  
  local, 27  
Visual Studio, 489

## W

whitespace, 357  
Windows  
  installing, 37  
workflows, 139  
  centralized, 139  
  dictator and lieutenants, 141  
  integration manager, 140  
  merging, 171  
  merging (large), 173  
  rebasing and cherry-picking, 175

**X**  
Xcode, 36

**Z**  
zsh, 493