

Table of Contents

1. [Introduction](#)
2. [GitHub](#)
 - i. [忽略空白字符变化](#)
 - ii. [调整Tab字符所代表的空格数](#)
 - iii. [查看某个用户的Commit历史](#)
 - iv. [克隆某个仓库](#)
 - v. [将某个分支与其他所有分支进行对比](#)
 - vi. [比较分支](#)
 - vii. [比较不同派生库的分支](#)
 - viii. [Gists](#)
 - ix. [Git.io](#)
 - x. [键盘快捷键](#)
 - xi. [整行高亮](#)
 - xii. [用commit信息关闭Issue](#)
 - xiii. [链接其他仓库的Issue](#)
 - xiv. [设置CI对每条Pull Request都进行构建](#)
 - xv. [Markdown文件高亮语法](#)
 - xvi. [表情符](#)
 - xvii. [静态与动态图片](#)
 - i. [在GitHub Wiki中嵌入图片](#)
 - xviii. [快速引用](#)
 - xix. [快速添加许可证](#)
 - xx. [任务列表](#)
 - i. [Markdown文件中的任务列表](#)
 - xxi. [相对链接](#)
 - xxii. [GitHub Pages的元数据与插件支持](#)
 - xxiii. [查看YAML格式的元数据](#)
 - xxiv. [渲染表格数据](#)
 - xxv. [Diffs](#)
 - i. [可渲染文档的Diffs](#)
 - ii. [可变化地图](#)
 - iii. [在diff中折叠与扩展代码](#)
 - iv. [查看Pull Request的diff和patch](#)
 - v. [渲染图像发生的变动](#)
 - xxvi. [Hub](#)
 - xxvii. [贡献内容的自动检查](#)
 - xxviii. [贡献者指南](#)
 - xxix. [GitHub资源](#)
 - i. [GitHub讨论](#)
3. [Git](#)
 - i. [前一个分支](#)
 - ii. [StripSpace命令](#)
 - iii. [检出Pull Requests](#)
 - iv. [提交空改动 :trollface:](#)
 - v. [更直观的Git Status](#)
 - vi. [更直观的Git Log](#)
 - vii. [Git查询](#)
 - viii. [合并分支](#)
 - ix. [使用网页查看本地仓库](#)
 - x. [Git配置](#)
 - i. [Git命令自定义别名](#)
 - ii. [自动更正](#)
 - iii. [带颜色输出](#)

xi. [Git资源](#)

i. [Git参考书籍](#)

GitHub秘籍

本秘籍收录了一些Git和Github非常酷同时又少有人知的功能。灵感来自于Zach Holman在2012年Aloha Ruby Conference和2013年WDCNZ上所做的演讲：[Git and GitHub Secrets\(slides\)](#)和[More Git and GitHub Secrets\(slides\)](#)。

Read this in other languages: [English](#), [한국어](#), [日本語](#), [简体中文](#).

目录

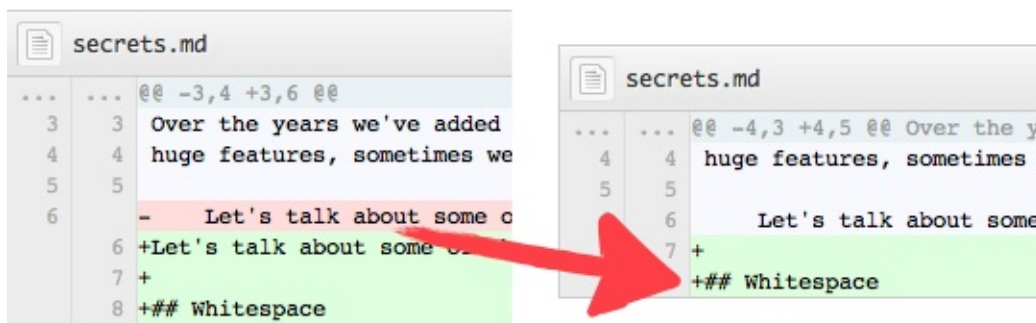
- [GitHub](#)
 - [忽略空白字符变化](#)
 - [调整Tab字符所代表的空格数](#)
 - [查看某个用户的Commit历史](#)
 - [克隆某个仓库](#)
 - [将某个分支与其他所有分支进行对比](#)
 - [比较分支](#)
 - [比较不同派生库的分支](#)
 - [Gists](#)
 - [Git.io](#)
 - [键盘快捷键](#)
 - [整行高亮](#)
 - [用commit信息关闭Issue](#)
 - [链接其他仓库的Issue](#)
 - [设置CI对每条Pull Request都进行构建](#)
 - [Markdown文件高亮语法](#)
 - [表情符](#)
 - [静态与动态图片](#)
 - [在GitHub Wiki中嵌入图片](#)
 - [快速引用](#)
 - [快速添加许可证](#)
 - [任务列表](#)
 - [Markdown文件中的任务列表](#)
 - [相对链接](#)
 - [GitHub Pages的元数据与插件支持](#)
 - [查看YAML格式的元数据](#)
 - [渲染表格数据](#)
 - [Diffs](#)
 - [可渲染文档的Diffs](#)
 - [可变化地图](#)
 - [在diff中折叠与扩展代码](#)
 - [查看Pull Request的diff和patch](#)
 - [渲染图像发生的变动](#)
 - [Hub](#)
 - [贡献内容的自动检查](#)
 - [贡献者指南](#)
 - [GitHub资源](#)
 - [GitHub讨论](#)
- [Git](#)
 - [前一个分支](#)
 - [StripSpace命令](#)
 - [检出Pull Requests](#)
 - [提交空改动 :trollface:](#)
 - [更直观的Git Status](#)
 - [更直观的Git Log](#)

- [Git查询](#)
- [合并分支](#)
- [使用网页查看本地仓库](#)
- [Git配置](#)
 - [Git命令自定义别名](#)
 - [自动更正](#)
 - [带颜色输出](#)
- [Git资源](#)
 - [Git参考书籍](#)

GitHub

忽略空白字符变化

在任意diff页面的URL后加上 `?w=1`，可以去掉那些只是空白字符的变化，使你能更专注于代码的变化。

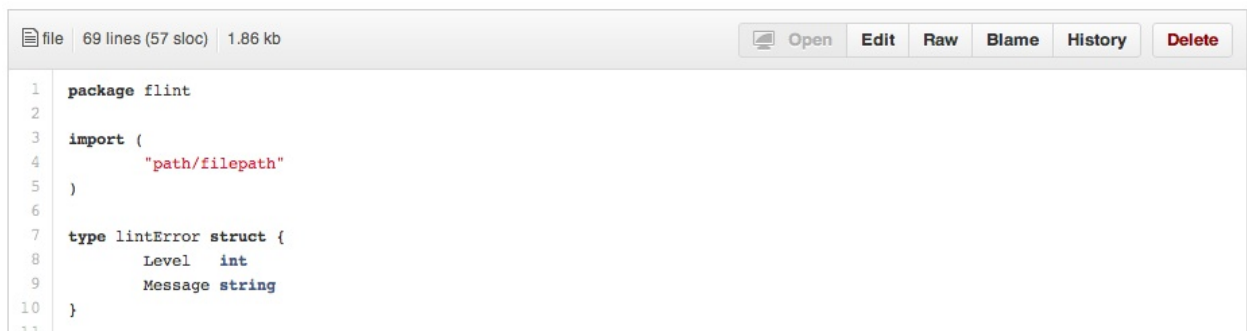


详见 [GitHub secrets](#).

调整Tab字符所代表的空格数

在diff或者file页面的URL后面加上 `?ts=4`，这样当显示tab字符的长度时就会是4个空格的长度，不再是默认的8个空格。`ts`后面的数字还可以根据你个人的偏好进行修改。不过，这个小诀窍在Gists页面和raw file页面不起作用。

下面是在Go语言的source file页面URL后加 `?ts=4` 前的例子：



然后是我们添加 `?ts=4` 后的例子：

```
file 69 lines (57 sloc) 1.86 kb
1 package flint
2
3 import (
4     "path/filepath"
5 )
6
7 type lintError struct {
8     Level int
9     Message string
10 }
```




查看某个用户的Commit历史

查看某个用户的所有提交历史，只需在commits页面URL后加上 `?author=username`。





```
https://github.com/rails/rails/commits/master?author=dhh
```

branch: master rails / Commits

Apr 08, 2014

-  **Dont abbreviate that which needs no abbreviation**
dhh authored 8 days ago 304d2f19c8 [Browse code](#)
-  **Dont encourage aliases now that we have variants**
dhh authored 8 days ago 10570cf5b [Browse code](#)
-  **Use short-form for the scaffold render calls and drop the needless test**
dhh authored 8 days ago 4b0c8a9467 [Browse code](#)

Mar 21, 2014

-  **Update test helper to use latest Digester API**
dhh authored a month ago 9d44b3f886 [Browse code](#)
-  **Digester should just rely on the finder to know about the format and ...** ...
dhh authored a month ago 637bb726ca [Browse code](#)
-  **Log the full path, including variant, that the digester is trying to ...** ...
dhh authored a month ago 4bca34750d [Browse code](#)
-  **Fix for digester to consider variants for partials -- this still need...** ...
dhh authored a month ago 06b4f01fca [Browse code](#)

深入了解提交视图之间的区别

克隆某个仓库

当我们克隆某一资源时，可以不要那个 `.git` 后缀。

```
$ git clone https://github.com/tiimgreen/github-cheat-sheet
```

更多对 `Git clone` 命令的介绍。

将某个分支与其他所有分支进行对比

当你点击某个仓库的分支（Branches）选项卡时

```
https://github.com/{user}/{repo}/branches
```

你会看到一个包含所有未合并的分支的列表。

你可以在这里查看比较（Compare）页面或点击删除某个分支。

Branches

Recently Active Stale

Showing 3 branches not merged into master. [View merged branches.](#)

master ✓ Last updated 3 days ago by mzgol.			Base branch
1.x-master ✓ Last updated 14 hours ago by mzgol.		547 ahead 743 behind	Compare
delegation ✓ Last updated 4 months ago by timmywil.		1 ahead 110 behind	Compare
1.9-stable Last updated a year ago by tomfuertes.		106 ahead 743 behind	Compare

有的时候我们需要将多个分支与一个非主分支（master）进行对比，此时可以通过在URL后加入要比较的分支名来实现：

```
https://github.com/{user}/{repo}/branches/{branch}
```

Branches

Recently Active Stale

Showing 2 branches not merged into 1.x-master. [View merged branches.](#)

1.x-master ✓ Last updated 14 hours ago by mzgol.			Base branch
master ✓ Last updated 3 days ago by mzgol.		743 ahead 547 behind	Compare
delegation ✓ Last updated 4 months ago by timmywil.		634 ahead 547 behind	Compare

可以在URL后加上 `?merged=1` 来查看已经合并了的分支。

Branches

Recently Active Stale

Showing 1 branch merged into 1.x-master. [View unmerged branches.](#)

1.x-master ✓ Last updated 14 hours ago by mzgol.			Base branch
1.9-stable Last updated a year ago by tomfuertes.		0 ahead 441 behind	Compare

你可以使用这个界面来替代命令行直接删除分支。

比较分支


如果我们想要比较两个分支，可以像下面一样修改URL：

```
https://github.com/user/repo/compare/{range}
```

其中 `{range} = master...4-1-stable`

例如：

<https://github.com/rails/rails/compare/master...4-1-stable>

 master ... 4-1-stable

Edit

Please review the [guidelines for contributing](#) to this repository.

Create Pull Request

Open a Pull Request for this comparison to discuss and review your changes with others.

?

247 commits

273 files changed







5 comments

36 contributors

Commits

Files changed


Commit comments

Feb 18, 2014		
	Update versions for 4.1.0.rc1	✗ 211ec1f
	Revert "Update versions for 4.1.0.rc1" -- old format for versions! ...	ccddc40
	Update versions for 4.1.0.rc1 (using new format)	✓ 78ba185
	Pointing README links to 4-1-stable [ci skip]	51bd49b
	`rails new --edge` should use the '4-1-stable' branch ...	✓ 24e1fff
	Merge pull request #14100 from chancancode/rails_new_edge ...	✓ e27b6fe

{range} 还可以使用下面的形式:

<https://github.com/rails/rails/compare/master@{1.day.ago}...master>
<https://github.com/rails/rails/compare/master@{2014-10-04}...master>

日期格式 YYYY-DD-MM

 master@{2014-10-04} ... master

Edit

130 commits

123 files changed

5 comments

39 contributors

Commits

Files changed

Commit comments

Aug 07, 2013		
	Add tests to ActiveSupport:XmlMini to_tag method	✓ 05d7cde
Nov 17, 2013		
	Fix insertion of records for hmt association with scope, fix #3548	✗ ec09280
Jan 07, 2014		
	Auto-generate stable fixture UUIDs on PostgreSQL. ...	✓ 9330631

...这样你就能查看master分支上一段时间或者指定日期内的改动。

[了解更多关于比较跨时间段的提交记录.](#)


比较不同派生库的分支

想要对派生仓库（Forked Repository）之间的分支进行比较，可以像下面这样修改URL实现：


https://github.com/user/repo/compare/{foreign-user}:{branch}...{own-branch}


例如：


https://github.com/rails/rails/compare/byroot:master...master


 byroot:master ... rails:master Edit


Please review the [guidelines for contributing](#) to this repository.

 **Create Pull Request** Open a Pull Request for this comparison to discuss and review your changes with others. ?

 6,802 commits

 309 files changed

 14 comments


 64 contributors







Commits

Files changed

Commit comments

This comparison is big! We're only showing the most recent 250 commits

 Apr 02, 2014

 kastiglione	PostgreSQL, Support for materialized views. [Dave Lee & Yves Senn] ...	✓ def6071
 rajcybage	We can conditional define the tests depending on the adapter or ...	✗ ee36af1
 rafaelfranca	Merge pull request #14565 from rajcybage/conditional_test_cases ...	✓ c82483a
 rwz	DRY AS::SafeBuffer a bit using existing helper	✓ 8482895
 alex88	Fixed small documentation typo ...	✗ 8ae3f24
 rafaelfranca	Merge pull request #14568 from alex88/patch-1 ...	✓ 3bcc51a





Gists



[Gists](#) 给我们提供了一种不需要创建一个完整的仓库，使小段代码也可以工作的简单方式。

GitHub Gist




Search...

Discover Gists

 rafaalchmiel   

PUBLIC   **tiimgreen / app.rb**

Created 4 days ago

  Star 0  Fork 0

Gist Detail

Revisions 1

Download Gist

Clone this gist

https://gist.githubub



Embed this gist

<script src="https:"




Link to this gist

https://gist.githubub

app.rb

Ruby  

1 puts 'Hello World'

 rafaalchmiel commented 2 days ago  


Wow, dat is some engineering.

Write

Preview

Comments are parsed with GitHub Flavored Markdown

Leave a comment

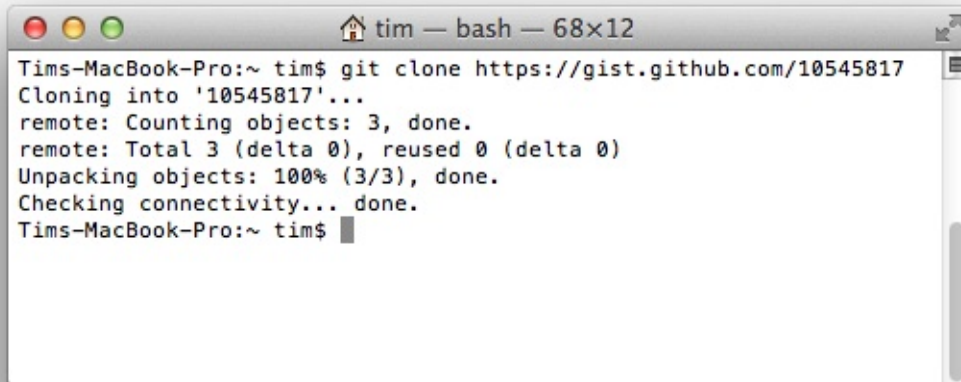


Add Comment

Gist的URL后加上 .pibb，可以得到更适合嵌入到其他网站的HTML版本。

Gists还可以像任何标准仓库一样被克隆。

```
$ git clone https://gist.github.com/tiimgreen/10545817
```

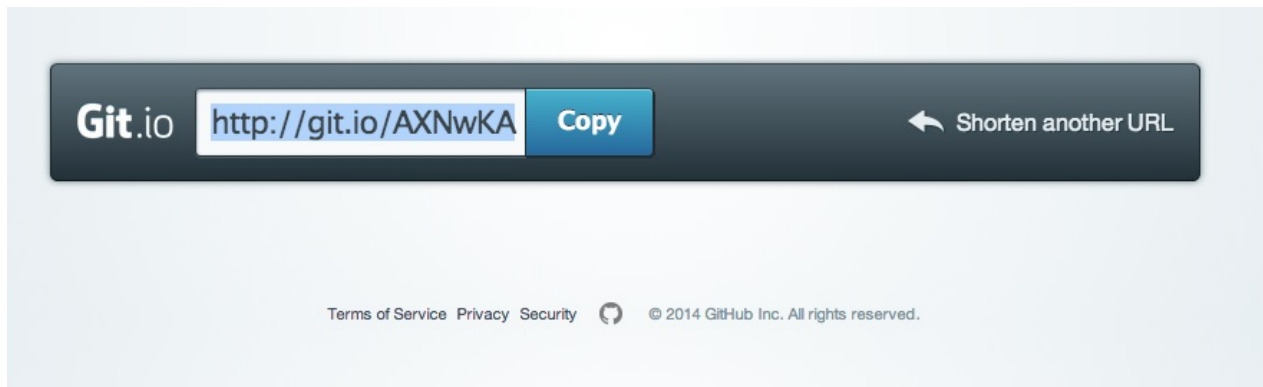
A terminal window titled 'tim — bash — 68x12' showing the execution of a git clone command. The output indicates that the repository was successfully cloned from a gist on GitHub.

```
Tims-MacBook-Pro:~ tim$ git clone https://gist.github.com/10545817
Cloning into '10545817'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
Checking connectivity... done.
Tims-MacBook-Pro:~ tim$
```

[进一步了解如何创建 gists.](#)

Git.io

Git.io是Github的短网址服务。



你可以通过Curl命令以普通HTTP协议使用它：

```
$ curl -i http://git.io -F "url=https://github.com/..."
HTTP/1.1 201 Created
Location: http://git.io/abc123

$ curl -i http://git.io/abc123
HTTP/1.1 302 Found
Location: https://github.com/...
```

[进一步了解 Git.io.](#)

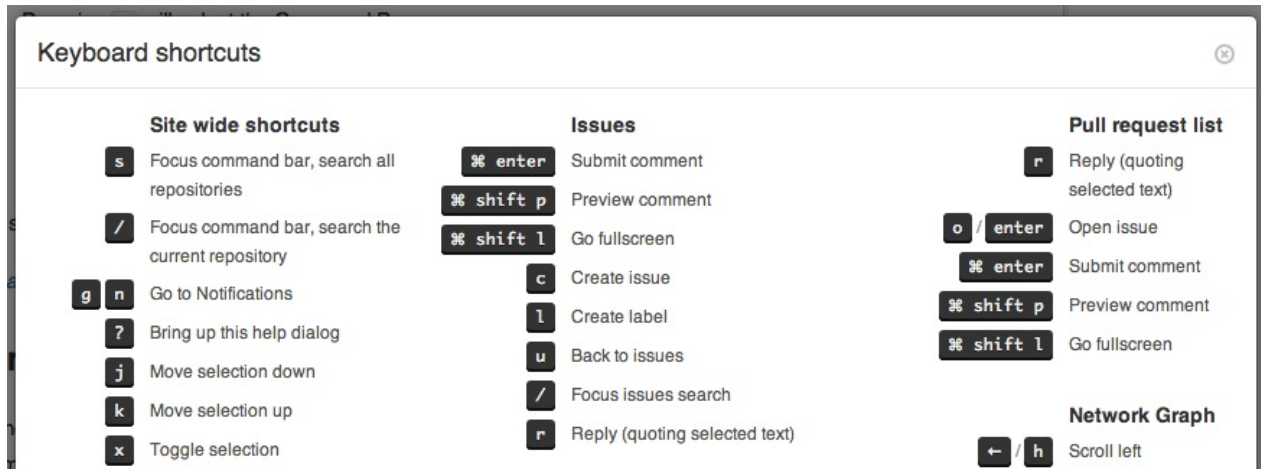
键盘快捷键

在仓库主页上提供了快捷键方便快速导航。

- 按 `t` 键会打开一个文件浏览器。

- 按 `w` 键会打开分支选择菜单。
- 按 `s` 键会激活顶端的命令栏 (Command Bar)。
- 按 `l` 键编辑Issue列表页的标签。
- 查看文件内容时（如：`https://github.com/tiimgreen/github-cheat-sheet/blob/master/README.md`），按 `y` 键将会冻结这个页面，这样就算代码被修改了也不会影响你当前看到的。

按 `?` 查看当前页面支持的快捷键列表：



进一步了解如何使用 *Command Bar*。

整行高亮

在代码文件地址后加上 `#L52` 或者单击行号52都会将第52行代码高亮显示。

多行高亮也可以，比如用 `#L53-L60` 选择范围，或者按住 `shift` 键，然后再点击选择的两行。

`https://github.com/rails/rails/blob/master/activemodel/lib/active_model.rb#L53-L60`

```

43   autoload :Serialization
44   autoload :TestCase
45   autoload :Translation
46   autoload :Validations
47   autoload :Validator
48
49   eager_autoload do
50     autoload :Errors
51   end
52
53   module Serializers
54     extend ActiveSupport::Autoload
55
56     eager_autoload do
57       autoload :JSON
58       autoload :Xml
59     end
60   end
61
62   def self.eager_load!
63     super
64     ActiveSupport::Serializers.eager_load!
65   end
66 end
67
68 ActiveSupport.on_load(:i18n) do
69   I18n.load_path << File.dirname(__FILE__) + '/active_model/locale/en.yml'
70 end

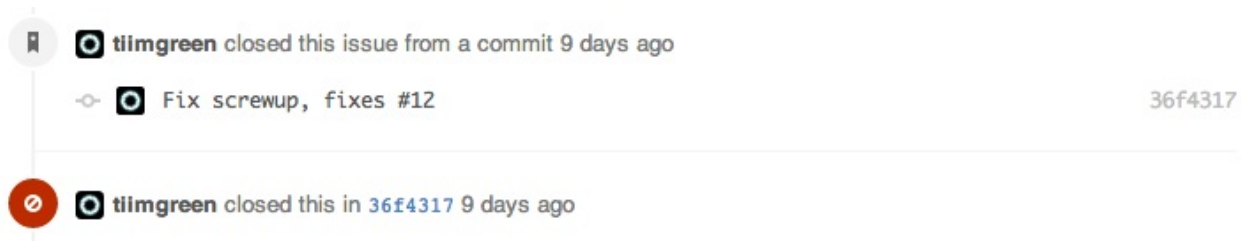
```

用commit信息关闭Issue

如果某个提交修复了一个Issue，当提交到master分支时，提交信息里可以使用 `fix/fixes/fixed`，`close/closes/closed` 或者 `resolve/resolves/resolved` 等关键词，后面再跟上Issue号，这样就会关闭这个Issue。

```
$ git commit -m "Fix screwup, fixes #12"
```

这将会关闭Issue #12, 并且在Issue讨论列表里关联引用这次提交。



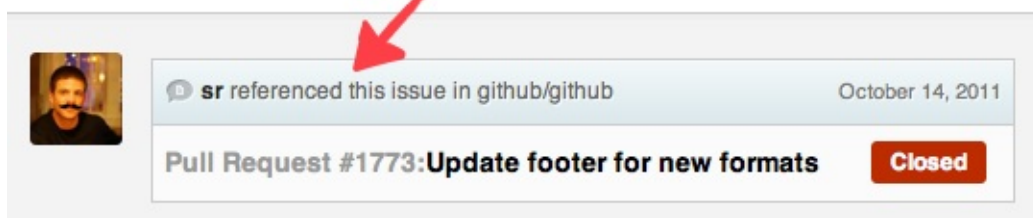
进一步了解通过提交信息关闭Issue.

链接其他仓库的Issue

如果你想引用到同一个仓库中的一个Issue, 只需使用井号 # 加上Issue号, 这样就会自动创建到此Issue的链接。

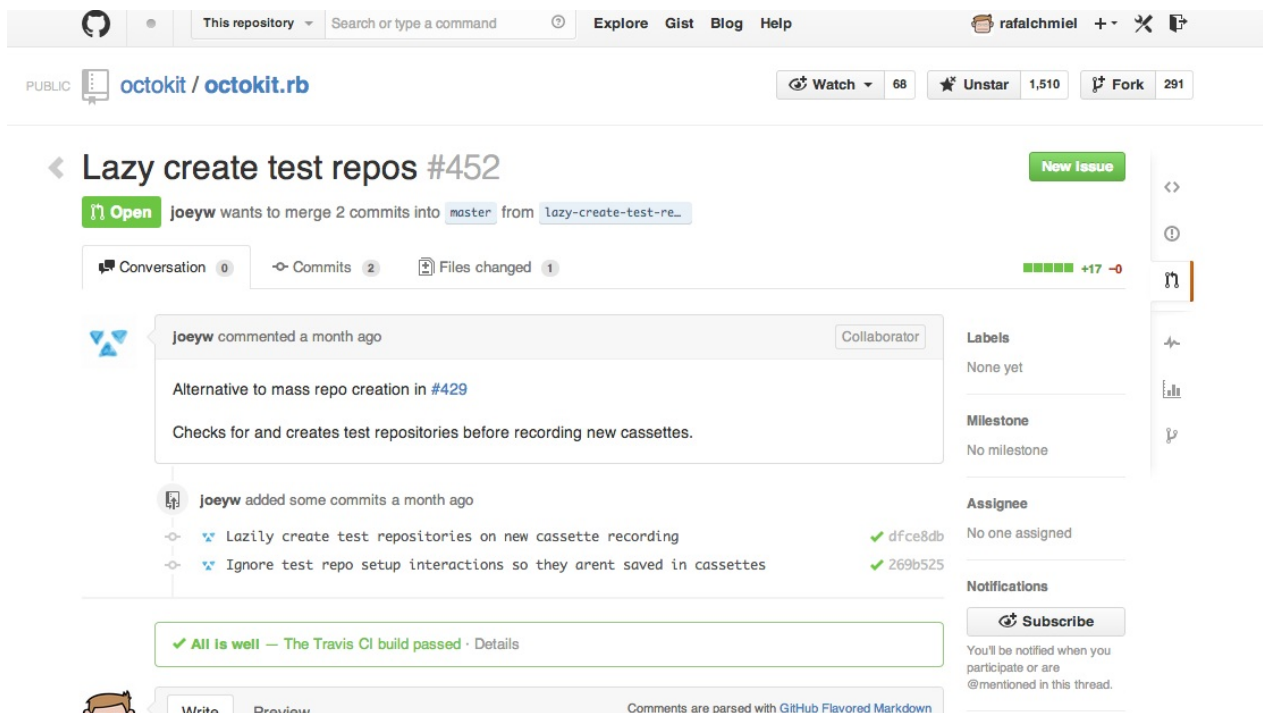
要链接到其他仓库的Issue, 就使用 `user_name/repo_name#ISSUE_NUMBER` 的方式, 例如 `tiimgreen/toc#12`。

We should probably handle this with [github/enterprise#59](#)



设置CI对每条Pull Request都进行构建

如果配置正确, Travis CI会为每个你收到的Pull Request执行构建, 就像每次提交也会触发构建一样。想了解更多关于Travis CI的信息, 请看 [Travis CI入门](#)。



进一步了解 [Commit status API](#).

Markdown文件高亮语法

例如，可以像下面这样在你的Markdown文件里为Ruby代码添加语法高亮：

```
```ruby
require 'tabbit'
table = Tabbit.new('Name', 'Email')
table.add_row('Tim Green', 'tiimgreen@gmail.com')
puts table.to_s
```
```

效果像下面这样：

```
require 'tabbit'
table = Tabbit.new('Name', 'Email')
table.add_row('Tim Green', 'tiimgreen@gmail.com')
puts table.to_s
```

Github使用 [Linguist](#) 做语言识别和语法高亮。你可以仔细阅读 [languages YAML file](#)，了解有哪些可用的关键字。

进一步了解 [GitHub Flavored Markdown](#).

表情符

可以在Pull Requests, Issues, 提交消息, Markdown文件里加入表情符。使用方法 `:name_of_emoji:`

```
:smile:
```

将输出一个笑脸：

```
:smile:
```

Github支持的完整表情符号列表详见[emoji-cheat-sheet.com](#) 或 [scotch-io/All-Github-Emoji-Icons](#)。

Github上使用最多的5个表情符号是：

1. `:shipit:` - `:shipit:`
2. `:sparkles:` - `:sparkles:`
3. `:-1:` - `:1:`
4. `:+1:` - `:+1:`
5. `:clap:` - `:clap:`

静态与动态图片

注释和README等文件里也可以使用图片和GIF动画：

```
![Alt Text](http://www.sheawong.com/wp-content/uploads/2013/08/keephatin.gif)
```



所有图片都缓存在Github，不用担心你的站点不能访问时就看不到图片了。

在GitHub Wiki中嵌入图片


有多种方法可以在Wiki页面里嵌入图片。既可以像上一条里那样使用标准的Markdown语法，也可以像下面这样指定图片的高度或宽度：

```
[[ http://www.sheawong.com/wp-content/uploads/2013/08/keephatin.gif | height = 100px ]]
```

结果：

[Home](#) [Pages](#) [History](#) [New Page](#)

Home (Preview)

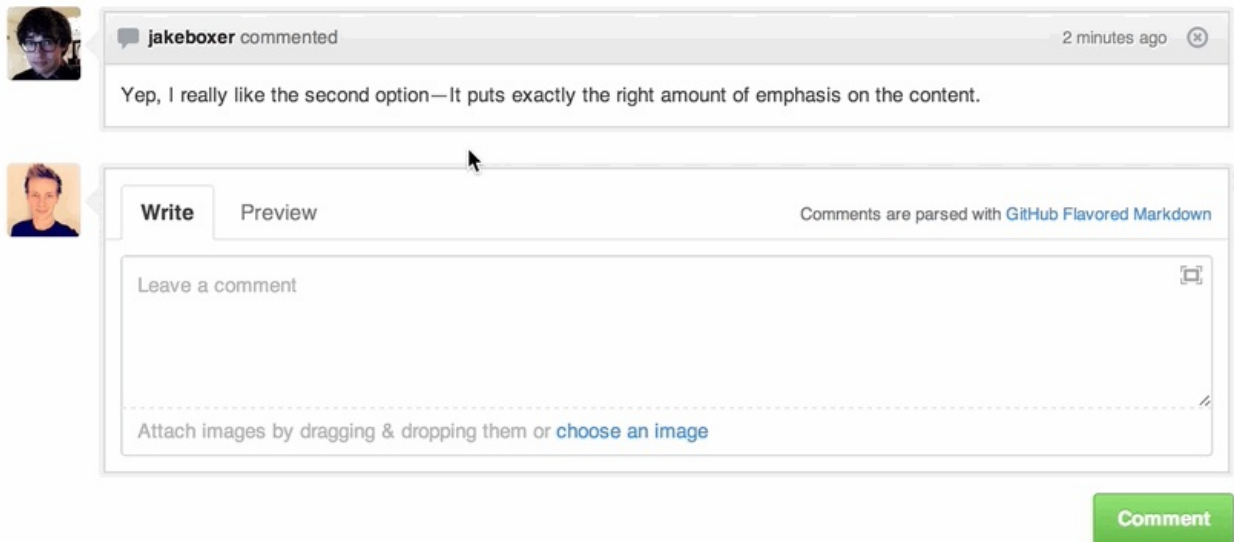


README not found

While a README isn't a required part of an open source project, it is a very good idea to have one. READMEs are a great place to describe your project or add some documentation such as how to install or use your project. You might want to include contact information - if your project becomes popular people will want to help you out. See [GitHub's article on creating repositories](#). See [Tom Preston-Werner's blog post on README driven development](#). Also see the [Wikipedia article on READMEs](#).

快速引用

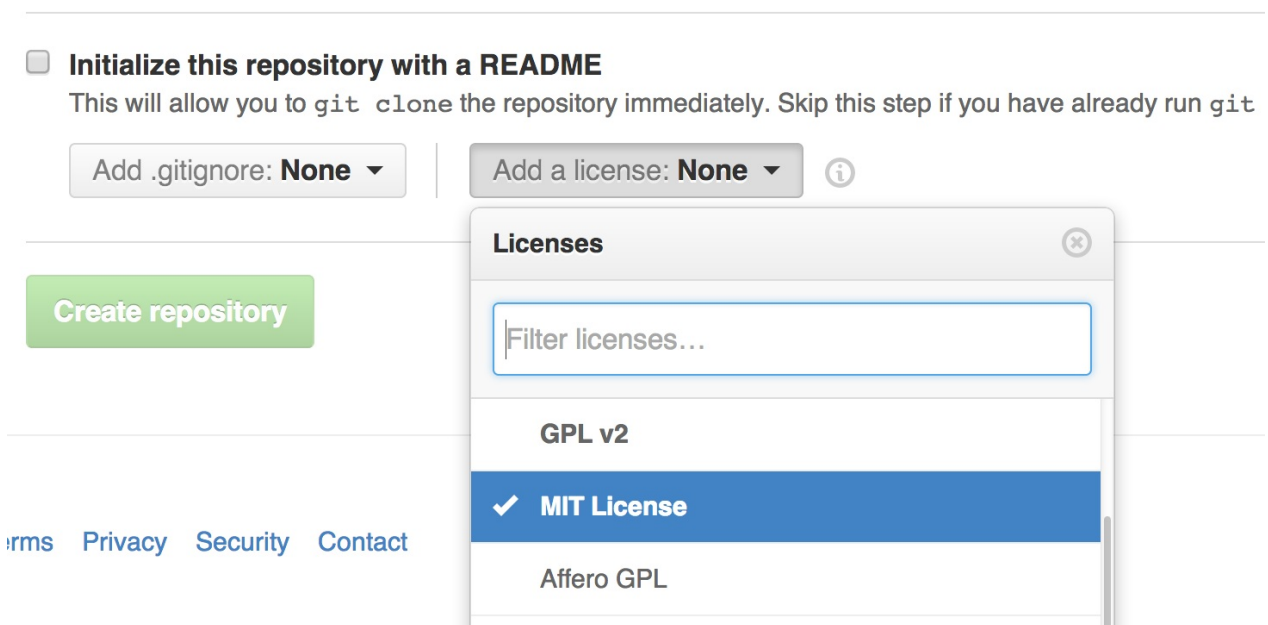
在注释话题里引用之前某个人所说的，只需选中文本，然后按 `r` 键，想要的就会以引用的形式复制到你的输入框里。



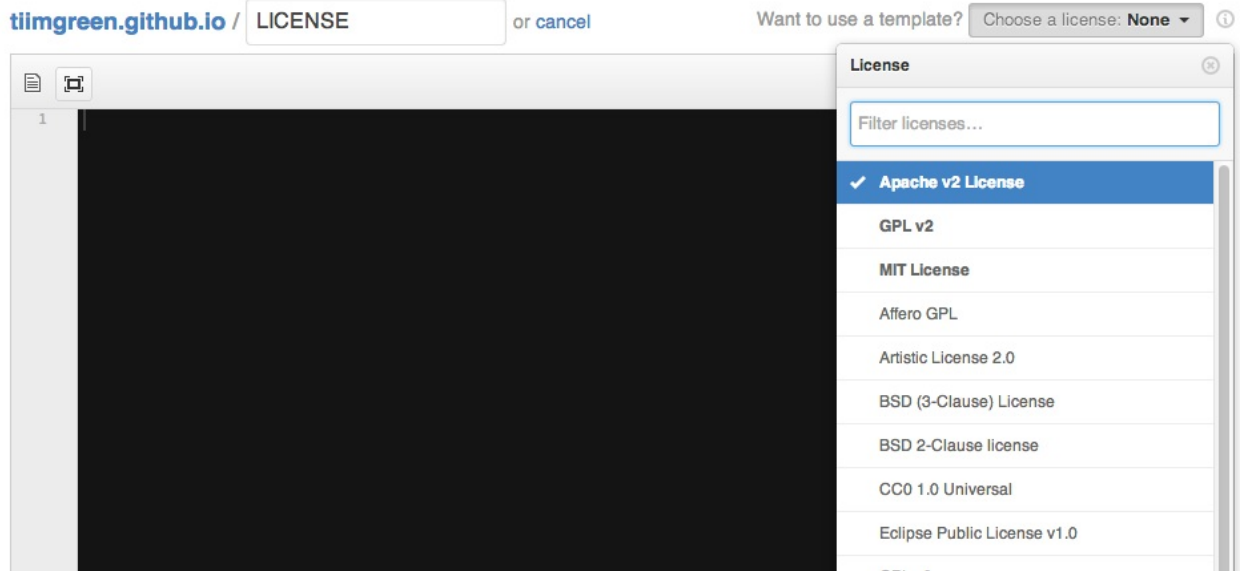
[进一步了解快速引用.](#)

快速添加许可证

创建一个仓库时，Github会为你提供一个预置的软件许可列表：



对于已有的仓库，可以通过web界面创建文件来添加软件许可。输入 LICENSE 作为文件名后，同样可以从预置的列表选择一个作为模板。



这个技巧也适用于 `.gitignore` 文件。

[进一步了解 open source licensing.](#)

任务列表

Issues和Pull requests里可以添加复选框，语法如下（注意空白符）：

```
- [ ] Be awesome
- [ ] Prepare dinner
  - [ ] Research recipe
  - [ ] Buy ingredients
  - [ ] Cook recipe
- [ ] Sleep
```

< TODO #1

Open AlexandreArpin opened this issue just now · 0 comments



AlexandreArpin commented just now

- ☐ Be awesome
- ☐ Prepare dinner
 - ☐ Research recipe
 - ☐ Buy ingredients
 - ☐ Cook recipe
- ☐ Sleep

当项目被选中时，它对应的Markdown源码也被更新了：

```
- [x] Be awesome
- [ ] Prepare dinner
- [x] Research recipe
- [x] Buy ingredients
- [ ] Cook recipe
- [ ] Sleep
```

[进一步了解任务列表.](#)

Markdown文件中的任务列表

在完全适配Markdown语法的文件中可以使用以下语法加入一个只读的任务列表

```
- [ ] Mercury
- [x] Venus
- [x] Earth
- [x] Moon
- [x] Mars
- [ ] Deimos
- [ ] Phobos
```

- ☐ Mercury
- ☒ Venus
- ☒ Earth
 - ☒ Moon
- ☒ Mars
 - ☐ Deimos
 - ☐ Phobos

[进一步了解Markdown文件中的任务列表](#)

相对链接

Markdown文件里链接到内部内容时推荐使用相对链接。

```
[Link to a header](#awesome-section)
[Link to a file](docs/readme)
```

绝对链接会在URL改变时（例如重命名仓库、用户名改变，建立分支项目）被更新。使用相对链接能够保证你的文档不受此影响。

[进一步了解相对链接.](#)

GitHub Pages的元数据与插件支持

在Jekyll页面和文章里，仓库信息可在 `site.github` 命名空间下找到，也可以显示出来，例如，使用 `{{ site.github.project_title }}` 显示项目标题。

Jemoji和jekyll-mentions插件为你的Jekyll文章和页面增加了emoji和@mentions功能。

[了解更多 GitHub Pages的元数据和插件支持.](#)

查看YAML格式的元数据

许多博客站点，比如基于Jekyll的GitHub Pages，都依赖于一些文章头部的YAML格式的元数据。Github会将其渲染成一个水平表格，方便阅读。

file | 61 lines (39 sloc) | 1.415 kb

OpenEditRawBlameHistoryDelete

| layout | title | description | tags | path | eventdate |
|--------|-------------------------------|-------------------------------|------------------|---|------------|
| bare | Git and GitHub Review for NYU | Git and GitHub Review for NYU | notesonlineclass | classnotes/2013-02-13-NYU-github-class.md | 2013-02-13 |

Your instructors for the evening are:

- Matthew McCullough (Twitter, GitHub)
- Tim Berglund (Twitter, GitHub)

进一步了解 在文档里 查看YAML元数据.

渲染表格数据

GitHub支持将 `.csv` (comma分隔)和 `.tsv` (tab分隔)格式的文件渲染成表格数据。

file | 869 lines (868 sloc) | 188.222 kb

EditRawBlameHistoryDelete

Search this file...

| | Title | Release Year | Locations | Fun Facts | Production Company |
|----|------------------------|--------------|---------------------------------------|---------------------------------------|---------------------------|
| 1 | 180 | 2011 | 555 Market St. | | SPI Cinemas |
| 2 | 180 | 2011 | Epic Roasthouse (399 Embarcade... | | SPI Cinemas |
| 3 | 180 | 2011 | Mason & California Streets (Nob Hill) | | SPI Cinemas |
| 4 | 180 | 2011 | Justin Herman Plaza | | SPI Cinemas |
| 5 | 180 | 2011 | 200 block Market Street | | SPI Cinemas |
| 6 | 180 | 2011 | City Hall | | SPI Cinemas |
| 7 | 180 | 2011 | Polk & Larkin Streets | | SPI Cinemas |
| 8 | 180 | 2011 | Randall Musuem | | SPI Cinemas |
| 9 | 24 Hours on Craigslist | 2005 | | | Yerba Buena Productions |
| 10 | 48 Hours | 1982 | | | Paramount Pictures |
| 11 | 50 First Dates | 2004 | Rainforest Café (145 Jefferson Str... | | Columbia Pictures Corpora |
| 12 | A Jitney Elopement | 1915 | Golden Gate Park | During San Francisco's Gold Rush... | The Essanay Film Manufact |
| 13 | A Jitney Elopement | 1915 | 20th and Folsom Streets | | The Essanay Film Manufact |
| 14 | A Night Full of Rain | 1978 | San Francisco Chronicle (901 Mis... | The San Francisco Zodiac Killer of... | Liberty Film |

进一步了解渲染表格数据.

Diffs

可渲染文档的Diffs

提交和Pull Requests里包含有Github支持的可渲染文档（比如Markdown）会提供source 和 *rendered* 两个视图功能。

21 rendered-prose-diffs.md

<>View

```
@@ -0,0 +1,21 @@
1  +# Rendered Prose Diffs
2  +
3  +Today we are making it easier to review and collaborate on prose documents. Commits and pull requests including
```

点击 "rendered" 按钮，看看改动在渲染后的显示效果。当你添加、删除或修改文本时，渲染纯文本视图非常方便。

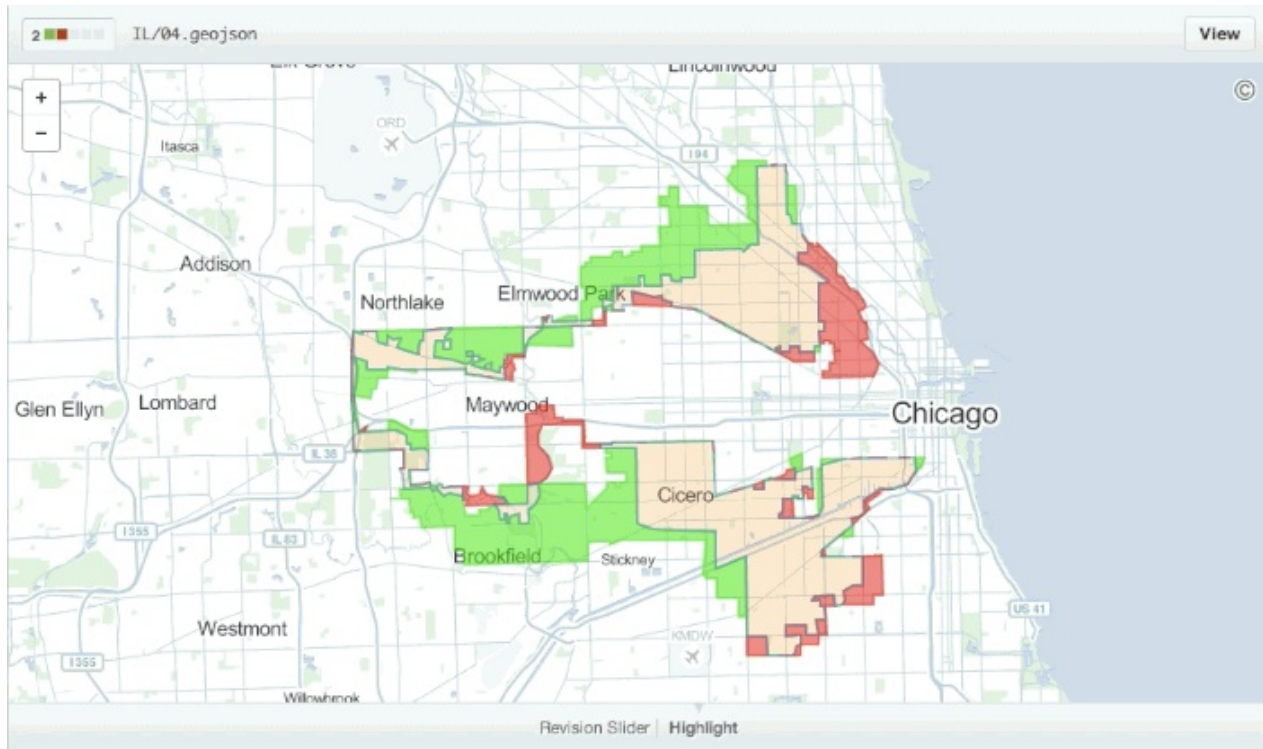
Obviously, you'll want to change `<your_client_id>` to match your actual Client ID.

Also, notice that the URL uses the `scope` query parameter to define the `scopes` requested by the application. For our application, we're requesting `user:email` scope for reading private email addresses.

[进一步了解渲染纯文本视图 *Diffs*.](#)

可变化地图

当你在GitHub上查看一个包含地理数据的提交或pull request时，Github可以显示数据变动的视觉表示。



[进一步了解可比较地图.](#)

在diff中折叠与扩展代码

你可以通过点击diff边栏里的 `unfold` 按钮来多显示几行上下文。你可以一直点击 `unfold` 按钮直到显示了文件的全部内容。这个功能在所有GitHub产生的diff界面都可以使用。

| | | |
|-----|-----|---|
| | 94 | +~ (RACSignal *)enqueueRequest:(NSURLRequest *)request fetchAllPages:(BOOL)fetchAllPages; |
| | 95 | + |
| 82 | 96 | // Enqueues a request to fetch information about the current user by accessing |
| 83 | 97 | // a path relative to the user object. |
| 84 | 98 | // |
| ✚ | | @@ -241,11 +255,13 @@ - (id)initWithServer:(OCTServer *)server { |
| 241 | 255 | NSString *userAgent = self.class.userAgent; |
| 242 | 256 | if (userAgent != nil) [self setDefaultHeader:@"User-Agent" value:userAgent]; |
| 243 | 257 | - self.parameterEncoding = AFJSONParameterEncoding; |
| 244 | | - [self setDefaultHeader:@"Accept" value:@"application/vnd.github.beta+json"]; |
| 245 | | - |
| 246 | | |
| 247 | 258 | [AFHTTPRequestOperation addAcceptableStatusCodes:[NSIndexSet indexSetWithIndex:OCTClientNotModifiedStatusCode]] |
| 248 | | - [AFJSONRequestOperation addAcceptableContentTypes:[NSSet setWithObject:@"application/vnd.github.beta+json"]]; |
| | 259 | + |
| | 260 | + NSString *contentType = [NSString stringWithFormat:@"application/vnd.github.%@+json", OCTClientAPIVersion]; |
| | 261 | + [self setDefaultHeader:@"Accept" value:contentType]; |
| | 262 | + [AFJSONRequestOperation addAcceptableContentTypes:[NSSet setWithObject:contentType]]; |
| | 263 | + |
| | 264 | + self.parameterEncoding = AFJSONParameterEncoding; |
| 249 | 265 | [self registerHTTPOperationClass:AFJSONRequestOperation.class]; |
| 250 | 266 | |
| 251 | 267 | return self; |

[进一步了解扩展Diff上下文。](#)

查看Pull Request的diff和patch

在Pull Request的URL后面加上 `.diff` 或 `.patch` 的扩展名就可以得到它的diff或patch文件，例如：

```
https://github.com/tiimgreen/github-cheat-sheet/pull/15
https://github.com/tiimgreen/github-cheat-sheet/pull/15.diff
https://github.com/tiimgreen/github-cheat-sheet/pull/15.patch
```

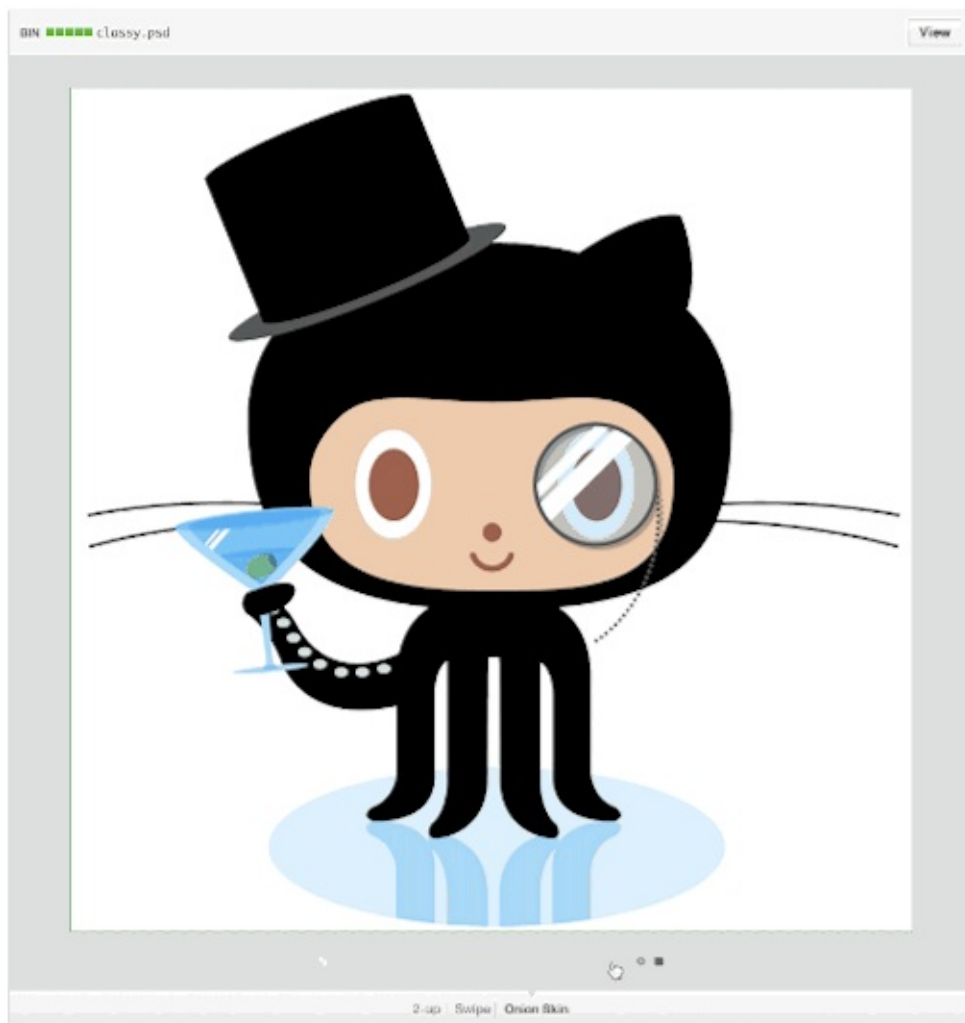
`.diff` 扩展会使用普通文本格式显示如下内容：

```
diff --git a/README.md b/README.md
index 88fcf69..8614873 100644
--- a/README.md
+++ b/README.md
@@ -28,6 +28,7 @@ All the hidden and not hidden features of Git and GitHub. This cheat sheet was i
- [Merged Branches](#merged-branches)
- [Quick Licensing](#quick-licensing)
- [TODO Lists](#todo-lists)
+- [Relative Links](#relative-links)
- [.gitconfig Recommendations](#gitconfig-recommendations)
- [Aliases](#aliases)
- [Auto-correct](#auto-correct)
@@ -381,6 +382,19 @@ When they are clicked, they will be updated in the pure Markdown:
- [ ] Sleep

(...)
```

渲染图像发生的变动

GitHub可以显示包括PNG、JPG、GIF、PSD在内的多种图片格式并提供了几种方式来比较这些格式的图片文件版本间的不同。



[查看更多关于渲染图像变动的内容](#)

Hub

Hub是一个对Git进行了封装的命令行工具，可以帮助你更方便的使用Github。

这使得你可以像下面这样进行克隆：

```
$ hub clone tiimgreen/toc
```

[查看更多Hub提供的超酷命令.](#)

贡献内容的自动检查

假设你想人们使用你的项目并给你的项目做出贡献，你往往需要回答他们常见问题。这个项目是干什么用的？我如何使用它？允许我怎样使用？我如何为项目出力？我怎样配置开发环境？我怎么能保证新功能不会破坏已有的功能？

Friction是一个命令行脚本，用来检查你的项目是否[回答了这些问题](#)。下面是示例输出：


```
Tims-MacBook-Pro:test tim$ friction
/test
[ERROR] CONTRIBUTING guide not found (see http://git.io/g_0mVQ)
[ERROR] LICENSE not found (see http://git.io/pFMQMQ)
[ERROR] Bootstrap script not found (see http://git.io/jZoRYA)
[ERROR] Test script not found (see http://git.io/oo21Jw)
[ERROR] .gitignore not found (see http://git.io/pevJkA)
Tims-MacBook-Pro:test tim$
```

Friction 支持 MRI 2.1.0, MRI 2.0.0 和 MRI 1.9.3.

贡献者指南

在你的仓库的根目录添加一个名为 CONTRIBUTING 的文件后，贡献者在新建Issue或Pull Request时会看到这个文件的链接。

Browse Issues Milestones Search:



Please review the [guidelines for contributing](#) to this repository.

Write Preview

Comments are parsed with [GitHub Flavored Markdown](#)

[进一步了解贡献者指南.](#)

GitHub资源

| Title | Link |
|------------------|---|
| GitHub Explore | https://github.com/explore |
| GitHub Blog | https://github.com/blog |
| GitHub Help | https://help.github.com/ |
| GitHub Training | http://training.github.com/ |
| GitHub Developer | https://developer.github.com/ |

GitHub讨论

| Title | Link |
|---|---|
| How GitHub Uses GitHub to Build GitHub | https://www.youtube.com/watch?v=qyz3jkOBbQY |
| Introduction to Git with Scott Chacon of GitHub | https://www.youtube.com/watch?v=ZDR433b0HJY |
| How GitHub No Longer Works | https://www.youtube.com/watch?v=gXD1ITW7iZI |

| | |
|-----------------------------|---|
| Git and GitHub Secrets | https://www.youtube.com/watch?v=Foz9yvMkvIA |
| More Git and GitHub Secrets | https://www.youtube.com/watch?v=p50xsL-iVgU |

Git

前一个分支

快速检出上一个分支：

```
$ git checkout -  
# Switched to branch 'master'  
  
$ git checkout -  
# Switched to branch 'next'  
  
$ git checkout -  
# Switched to branch 'master'
```

[进一步了解 Git 分支.](#)

Stripspace命令

Git Stripspace命令可以:

- 去掉行尾空白符
- 多个空行压缩成一行
- 必要时在文件末尾增加一个空行

使用此命令时必须传入一个文件，像这样：

```
$ git stripspace < README.md
```

[进一步了解 Git stripspace 命令.](#)

检出Pull Requests

Pull Request是一种GitHub上可以通过以下多种方式在本地被检索的特别分支：

检索某个分支并临时储存在本地的 `FETCH_HEAD` 中以便快速查看更改(diff)以及合并(merge)：

```
$ git fetch origin refs/pull/[PR-Number]/head
```

通过refspec获取所有的Pull Request为本地分支：

```
$ git fetch origin '+refs/pull/*:head:refs/remotes/origin/pr/*'
```

或在仓库的 `.git/config` 中加入下列设置来自动获取远程仓库中的Pull Request

```
[remote "origin"]  
  fetch = +refs/heads/*:refs/remotes/origin/*  
  url = git@github.com:tiimgreen/github-cheat-sheet.git
```



```
[remote "origin"]
  fetch = +refs/heads/*:refs/remotes/origin/*
  url = git@github.com:tiimgreen/github-cheat-sheet.git
  fetch = +refs/pull/*:head:refs/remotes/origin/pr/*
```

对基于派生库的Pull Request，可以通过先 checkout 代表此Pull Request的远端分支再由此分支建立一个本地分支：

```
$ git checkout pr/42 pr-42
```

[进一步了解如何检出pull request到本地.](#)

提交空改动 :trollface:

可以使用 `--allow-empty` 选项强制创建一个没有任何改动的提交：

```
$ git commit -m "Big-ass commit" --allow-empty
```

这样做在如下几种情况下是有意义的：

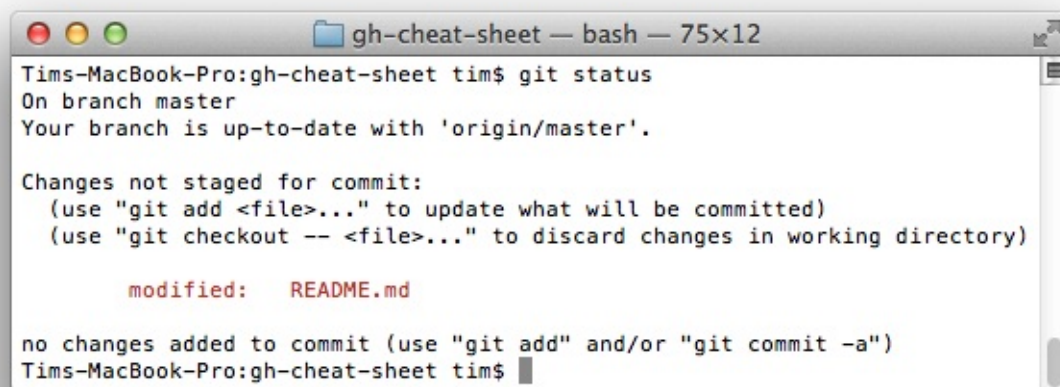
- 标记一批工作或一个新功能的开始。
- 记录你对项目进行了跟代码无关的改动。
- 跟使用你仓库的其他人交流。
- 作为仓库的第一次提交，因为第一次提交日后是不能被rebase的：`git commit -m "init repo" --allow-empty`。

更直观的Git Status

在命令行输入如下命令：

```
$ git status
```

可以看到：

A screenshot of a macOS terminal window titled "gh-cheat-sheet — bash — 75x12". The terminal shows the output of the "git status" command. The output indicates the user is on the "master" branch and it is up-to-date with "origin/master". It lists "Changes not staged for commit" with instructions on how to use "git add" and "git checkout". It shows "modified: README.md" in red text. Finally, it states "no changes added to commit" and provides instructions for using "git add" and "git commit -a". The prompt "Tims-MacBook-Pro:gh-cheat-sheet tim\$" is visible at the bottom.

```
Tims-MacBook-Pro:gh-cheat-sheet tim$ 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)

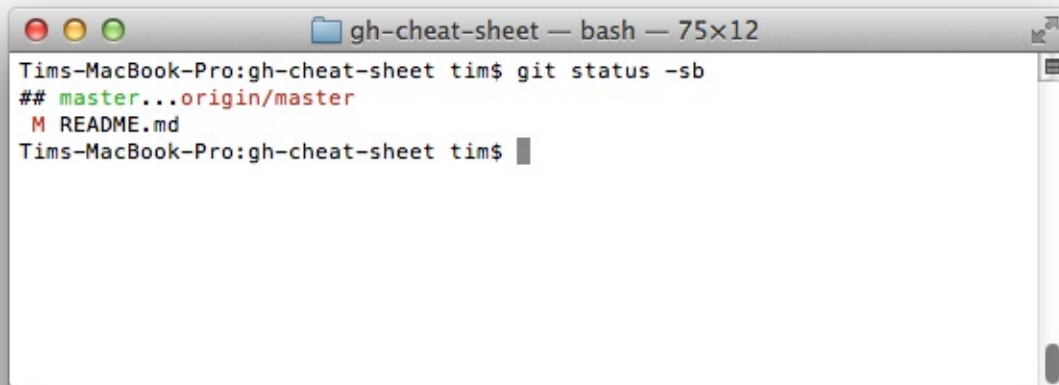
       modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
Tims-MacBook-Pro:gh-cheat-sheet tim$
```

加上 `-sb` 选项：

```
$ git status -sb
```

这回得到:



```
gh-cheat-sheet — bash — 75x12
Tims-MacBook-Pro:gh-cheat-sheet tim$ git status -sb
## master...origin/master
M README.md
Tims-MacBook-Pro:gh-cheat-sheet tim$
```

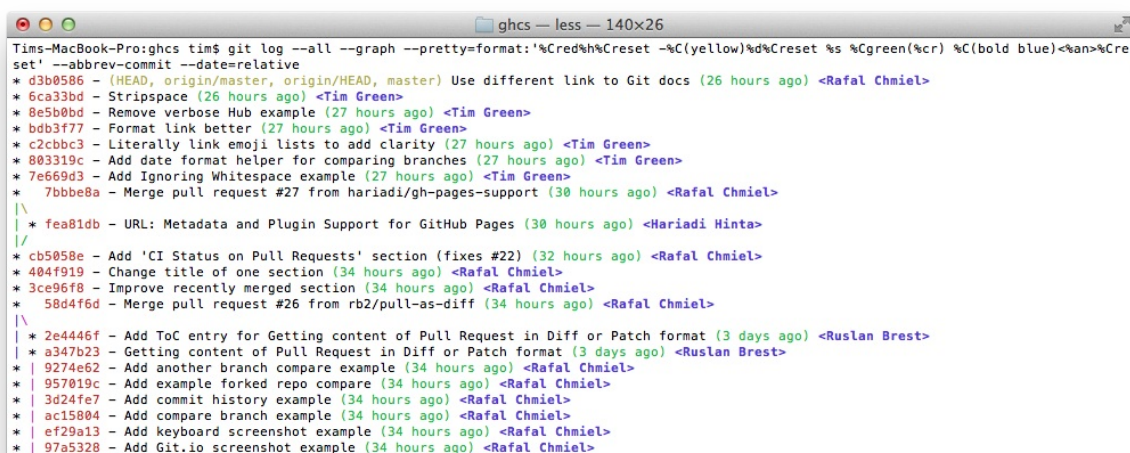
进一步了解 `Git status` 命令.

更直观的Git Log

输入如下命令:

```
$ git log --all --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<an>%Creset' --at
```

可以看到:



```
ghcs — less — 140x26
Tims-MacBook-Pro:ghcs tim$ git log --all --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<an>%Creset' --at
set' --abbrev-commit --date=relative
* d3b0586 - (HEAD, origin/master, origin/HEAD, master) Use different link to Git docs (26 hours ago) <Rafal Chmiel>
* 6ca33bd - Stripspace (26 hours ago) <Tim Green>
* 0e5b0bd - Remove verbose Hub example (27 hours ago) <Tim Green>
* bdb3f77 - Format link better (27 hours ago) <Tim Green>
* a2cbbc3 - Literally link emoji lists to add clarity (27 hours ago) <Tim Green>
* 803319c - Add date format helper for comparing branches (27 hours ago) <Tim Green>
* 7e669d3 - Add Ignoring Whitespace example (27 hours ago) <Tim Green>
* 7b8be8a - Merge pull request #27 from hariadi/gh-pages-support (30 hours ago) <Rafal Chmiel>
|
| * fea81db - URL: Metadata and Plugin Support for GitHub Pages (30 hours ago) <Hariadi Hint>
|
| * cb5058e - Add 'CI Status on Pull Requests' section (fixes #22) (32 hours ago) <Rafal Chmiel>
* 404f919 - Change title of one section (34 hours ago) <Rafal Chmiel>
* 3ce96f8 - Improve recently merged section (34 hours ago) <Rafal Chmiel>
* 58d4f6d - Merge pull request #26 from rb2/pull-as-diff (34 hours ago) <Rafal Chmiel>
|
| * 2e4446f - Add ToC entry for Getting content of Pull Request in Diff or Patch format (3 days ago) <Ruslan Brest>
* a347b23 - Getting content of Pull Request in Diff or Patch format (3 days ago) <Ruslan Brest>
* 9274e62 - Add another branch compare example (34 hours ago) <Rafal Chmiel>
* 957019c - Add example forked repo compare (34 hours ago) <Rafal Chmiel>
* 3d24fe7 - Add commit history example (34 hours ago) <Rafal Chmiel>
* ac15804 - Add compare branch example (34 hours ago) <Rafal Chmiel>
* ef29a13 - Add keyboard screenshot example (34 hours ago) <Rafal Chmiel>
* 97a5328 - Add Git.io screenshot example (34 hours ago) <Rafal Chmiel>
```

这要归功于[Palesz](#)在stackoverflow的回答。

这个命令可以被用作别名, 详细做法见[这里](#)。

进一步了解 `Git log` 命令.

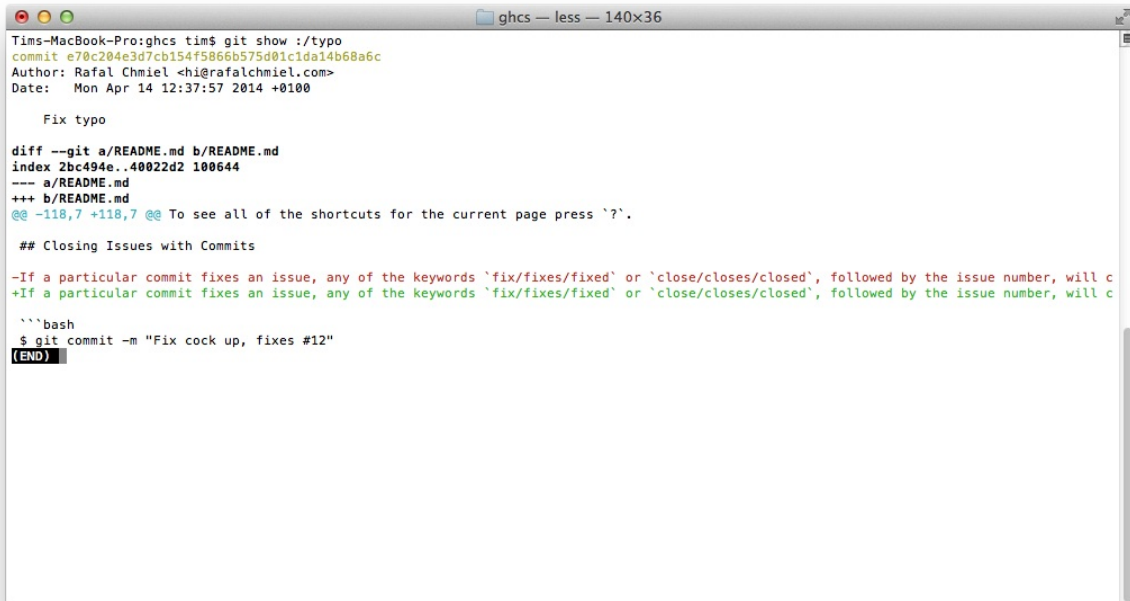
Git 查询

Git 查询运行你在之前的所有提交信息里进行搜索，找到其中和搜索条件相匹配的最近的一条。

```
$ git show :/query
```

这里 `query`（区别大小写）是你想要搜索的词语，这条命令会找到包含这个词语的最后那个提交并显示变动详情。

```
$ git show :/typo
```



```
Tims-MacBook-Pro:ghcs tim$ git show :/typo
commit e70c204e3d7cb154f5866b575d01c1da14b68a6c
Author: Rafal Chmiel <hi@rafalchmiel.com>
Date:   Mon Apr 14 12:37:57 2014 +0100

    Fix typo

diff --git a/README.md b/README.md
index 2bc494e..40022d2 100644
--- a/README.md
+++ b/README.md
@@ -118,7 +118,7 @@ To see all of the shortcuts for the current page press `?`.

## Closing Issues with Commits

-If a particular commit fixes an issue, any of the keywords `fix/fixes/fixed`, followed by the issue number, will c
+If a particular commit fixes an issue, any of the keywords `fix/fixes/fixed` or `close/closes/closed`, followed by the issue number, will c

```bash
$ git commit -m "Fix cock up, fixes #12"
(END)
```

- 按 `q` 键退出命令。\*

## 合并分支

输入命令：

```
$ git branch --merged
```

这会显示所有已经合并到你当前分支的分支列表。

相反地：

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

会显示所有还没有合并到你当前分支的分支列表。

[进一步了解 Git branch 命令.](#)

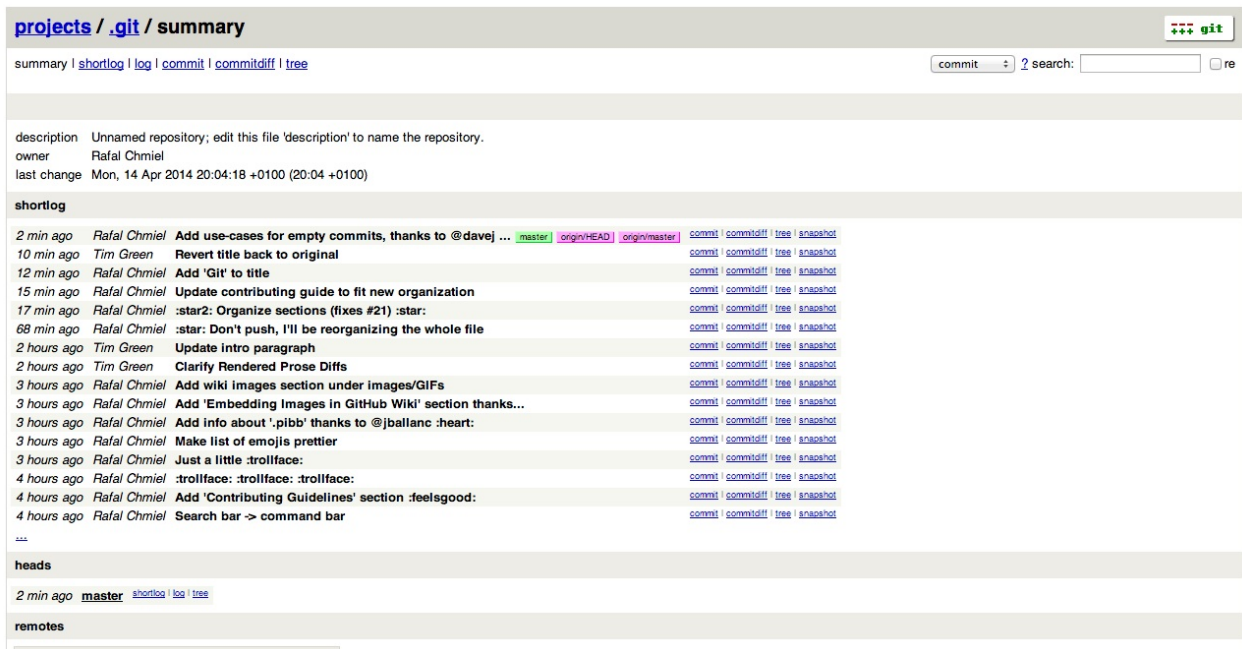
## 使用网页查看本地仓库

使用Git的 `instaweb` 可以立即在 `gitweb` 中浏览你的工作仓库。这个命令是个简单的脚步，配置了 `gitweb` 和用来浏览本地仓库

的Web服务器。（译者注：默认需要`lighttpd`支持）

```
$ git instaweb
```

执行后打开：



The screenshot shows the Git instaweb web interface. At the top, there's a navigation bar with 'projects / .git / summary' and a 'git' logo. Below the navigation bar, there's a search bar and a 'commit' button. The main content area displays repository information: 'description: Unnamed repository; edit this file 'description' to name the repository.', 'owner: Rafal Chmiel', and 'last change: Mon, 14 Apr 2014 20:04:18 +0100 (20:04 +0100)'. Below this, there's a 'shortlog' section showing a list of recent commits with their authors, messages, and links to the commit, diff, tree, and snapshot. The 'heads' section shows the 'master' branch. The 'remotes' section is empty.

进一步了解 `Git instaweb` 命令.

## Git配置

所有Git配置都保存在你的 `.gitconfig` 文件中。

## Git命令自定义 别名

别名用来帮助你定义自己的git命令。比如你可以定义 `git a` 来运行 `git add --all`。

要添加一个别名，一种方法是打开 `~/.gitconfig` 文件并添加如下内容：

```
[alias]
co = checkout
cm = commit
p = push
Show verbose output about tags, branches or remotes
tags = tag -l
branches = branch -a
remotes = remote -v
```

...或者在命令行里键入：

```
$ git config --global alias.new_alias git_function
```

例如：

```
$ git config --global alias.cm commit
```

指向多个命令的别名可以用引号来定义：

```
$ git config --global alias.ac 'add -A . && commit'
```

下面列出了一些有用的别名：

| 别名<br>Alias  | 命令 Command                                                                                                                                | 如何设置 What to Type                                                                                                                                                    |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| git cm       | git commit                                                                                                                                | git config --global alias.cm commit                                                                                                                                  |
| git co       | git checkout                                                                                                                              | git config --global alias.co checkout                                                                                                                                |
| git ac       | git add . -A git commit                                                                                                                   | git config --global alias.ac '!git add -A && git commit'                                                                                                             |
| git st       | git status -sb                                                                                                                            | git config --global alias.st 'status -sb'                                                                                                                            |
| git tags     | git tag -l                                                                                                                                | git config --global alias.tags 'tag -l'                                                                                                                              |
| git branches | git branch -a                                                                                                                             | git config --global alias.branches 'branch -a'                                                                                                                       |
| git remotes  | git remote -v                                                                                                                             | git config --global alias.remotes 'remote -v'                                                                                                                        |
| git lg       | git log --color --graph --pretty=format:%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --abbrev-commit -- | git config --global alias.lg "log --color --graph --pretty=format:%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --abbrev-commit --" |

## 自动更正

如果键入 `git comit` 你会看到如下输出：

```
$ git comit -m "Message"
git: 'comit' is not a git command. See 'git --help'.

Did you mean this?
commit
```

为了在键入 `comit` 调用 `commit` 命令，只需启用自动纠错功能：

```
$ git config --global help.autocorrect 1
```

现在你就会看到：

```
$ git comit -m "Message"
WARNING: You called a Git command named 'comit', which does not exist.
Continuing under the assumption that you meant 'commit'
in 0.1 seconds automatically...
```

## 带颜色输出

要在你的Git命令输出里加上颜色的话，可以用如下命令：

```
$ git config --global color.ui 1
```

进一步了解 `Git config` 命令.

## Git资源

| Title | Link |
|-------|------|
|-------|------|

|                                           |                                                                                                                               |
|-------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Official Git Site                         | <a href="http://git-scm.com/">http://git-scm.com/</a>                                                                         |
| Official Git Video Tutorials              | <a href="http://git-scm.com/videos">http://git-scm.com/videos</a>                                                             |
| Code School Try Git                       | <a href="http://try.github.com/">http://try.github.com/</a>                                                                   |
| Introductory Reference & Tutorial for Git | <a href="http://gitref.org/">http://gitref.org/</a>                                                                           |
| Official Git Tutorial                     | <a href="http://git-scm.com/docs/gittutorial">http://git-scm.com/docs/gittutorial</a>                                         |
| Everyday Git                              | <a href="http://git-scm.com/docs/everyday">http://git-scm.com/docs/everyday</a>                                               |
| Git Immersion                             | <a href="http://gitimmersion.com/">http://gitimmersion.com/</a>                                                               |
| Ry's Git Tutorial                         | <a href="http://rypress.com/tutorials/git/index.html">http://rypress.com/tutorials/git/index.html</a>                         |
| Git for Designer                          | <a href="http://hoth.entp.com/output/git_for_designers.html">http://hoth.entp.com/output/git_for_designers.html</a>           |
| Git for Computer Scientists               | <a href="http://eagain.net/articles/git-for-computer-scientists/">http://eagain.net/articles/git-for-computer-scientists/</a> |
| Git Magic                                 | <a href="http://www-cs-students.stanford.edu/~blynn/gitmagic/">http://www-cs-students.stanford.edu/~blynn/gitmagic/</a>       |

## Git参考书籍

| Title                               | Link                                                                                                                                                                              |
|-------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Pragmatic Version Control Using Git | <a href="http://www.pragprog.com/titles/tsgit/pragmatic-version-control-using-git">http://www.pragprog.com/titles/tsgit/pragmatic-version-control-using-git</a>                   |
| Pro Git                             | <a href="http://git-scm.com/book">http://git-scm.com/book</a>                                                                                                                     |
| Git Internals Peepcode              | <a href="http://peepcode.com/products/git-internals-pdf">http://peepcode.com/products/git-internals-pdf</a>                                                                       |
| Git in the Trenches                 | <a href="http://cbx33.github.com/gitt/">http://cbx33.github.com/gitt/</a>                                                                                                         |
| Version Control with Git            | <a href="http://www.amazon.com/Version-Control-Git-collaborative-development/dp/1449316387">http://www.amazon.com/Version-Control-Git-collaborative-development/dp/1449316387</a> |
| Pragmatic Guide to Git              | <a href="http://www.pragprog.com/titles/pg_git/pragmatic-guide-to-git">http://www.pragprog.com/titles/pg_git/pragmatic-guide-to-git</a>                                           |
| Git: Version Control for Everyone   | <a href="http://www.packtpub.com/git-version-control-for-everyone/book">http://www.packtpub.com/git-version-control-for-everyone/book</a>                                         |