

GitHubカンニング・ペーパー



これはGitやGitHubの隠された機能やよく知られていない機能の一覧だ。Zach HolmanによるAloha Ruby Conference 2012でのGit and GitHub Secrets（スライド）とWDCNZ 2013でのMore Git and GitHub Secrets（スライド）の二つのトーカーを元にしている。

短縮URL: <http://git.io/sheet>

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

目次

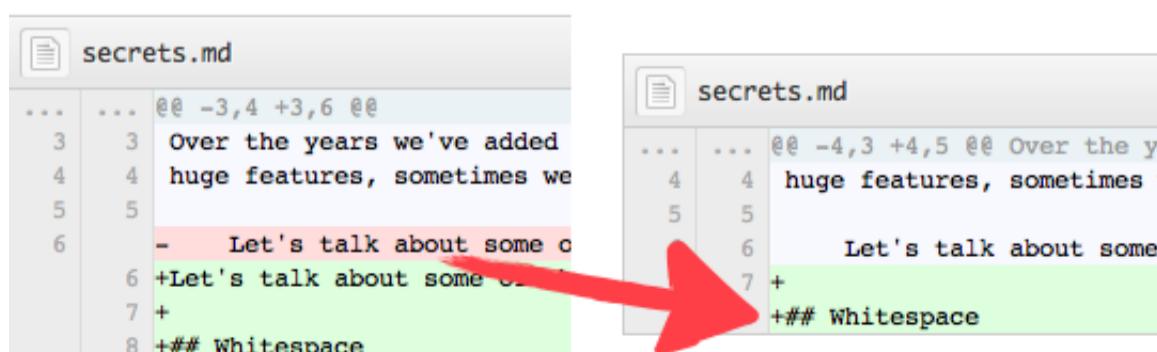
- GitHub
 - 空白の無視
 - タブ幅の調節
 - 特定のユーザーによるコミット履歴
 - リポジトリのクローン
 - ブランチ
 - 全ブランチを別のブランチと比較
 - ブランチ同士の比較
 - フォークされたリポジトリ間でのブランチ比較
 - Gists
 - Git.io
 - キーボード・ショートカット
 - コードの指定行の強調
 - コミットからissueを閉じる
 - issueの相互リンク
 - コメントのロツク
 - pull requestでのCI結果の表示
 - 絞り込み検索
 - Markdownファイルでの構文強調
 - Emoji
 - 画像及びアニメーションGIF
 - GitHub Wikiへの画像の添付
 - 素早く引用
 - コメントへのクリップボードの画像の追加
 - 設定済みライセンスの追加
 - タスクリスト
 - Markdownファイルでのタスクリスト
 - 相対リンク
 - GitHub Pagesでのメタデータとプラグインのサポート
 - 文書内のYAMLメタデータ
 - 表形式のデータ表示
 - PDF表示
 - pull requestの取り消し
 - 差分の表示
 - レンダリング済みの差分表示

- マップ差分の可視化
- 差分表示の前後を表示
- pull requestの内容をDIFFまたはPATCH形式で取得
- 画像の表示とDIFF
- Hub
- 開発参加のガイドライン
- Octicons
- GitHub情報
 - GitHub Talks
- Git
 - すべての削除済みファイルをワーキング・ツリーから削除する
 - 直前のブランチ
 - 空白の削除
 - SSH公開鍵
 - pull requestのチェックアウト
 - 空のコミット
 - Gitステータスのスタイル
 - Gitログのスタイル
 - コミットログの検索
 - Git Grep
 - マージ済みブランチ
 - FixupとAutosquash
 - ローカル・リポジトリを参照するウェブサーバー
 - Gitの設定
 - エイリアス
 - コマンドの自動修正
 - 色設定
 - Git情報
 - Git Books
- 訳注

GitHub

空白の無視

GitHub上で差分ページを表示している時、そのURLに ?w=1 を加えると、空白の変化によるできた差分は表示されなくなり、コード上の変化だけを参照することができる。



タブ幅の調節

差分やファイルを表示している時、URLに ?ts=4 を追加するとタブを空白4つの幅で表示する。デフォルトは8つだ。 ts に指定した数で表示されるということだ。これはGistやrawファイルを表示している時には適用されないが、[ChromeとOperaの拡張をインストールすることにより](#)、対応することができる。

例えばGoのソースファイルを表示している時、 ?ts=4 を追加する前はこのように表示されるが：

A screenshot of a GitHub raw file viewer. The top bar shows 'file' (69 lines, 57 sloc, 1.86 kb) and buttons for Open, Edit, Raw, Blame, History, and Delete. The code area contains the following Go code:

```
1 package flint
2
3 import (
4     "path/filepath"
5 )
6
7 type lintError struct {
8     Level    int
9     Message  string
10 }
11
```

?ts=4 を追加するとこのように表示される：

A screenshot of a GitHub raw file viewer, identical to the previous one but with the URL parameter ?ts=4 added. The code is displayed with wider tabs, reflecting the increased tab width.

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

特定のユーザーによるコミット履歴

特定のユーザーによるあるリポジトリへのコミット履歴のみを参照したい場合は、 ?author={user} をURLの末尾に付ける。

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

branch: master ▾

Commits on Jan 13, 2015

	Merge pull request #18476 from Alamoz/scaffold_index_view_grammar dhh authored 5 days ago	... 78a4884	 78a4884	
	Stop promoting rack-cache usage at the moment (not so common or impor... dhh authored 5 days ago	... 1302edf	 1302edf	
	Show how to change the queuing backend for ActiveJob in production dhh authored 5 days ago	... 6463495	 6463495	
	Set all asset options together dhh authored 5 days ago	... b9b28d8	 b9b28d8	

Commits on Jan 9, 2015

	Merge pull request #18413 from brainopia/automatic_inverse_of_for_bel... dhh authored 9 days ago	... 6eb499f	 6eb499f	
---	---	---------------------------------	---	---

コミット・ビューの違いについてもつと詳しく

リポジトリのクローン

リポジトリをクローンする時、URLの末尾の .git は無くても構わない。

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

Gitの clone コマンドについてもつと詳しく

ブランチ

全ブランチを別のブランチと比較

コミット一覧ボタンの隣からブランチ一覧ページに行くと：

<https://github.com/{user}/{repo}/branches>

メイン・ブランチにマージされていないブランチの一覧が表示される。

このページからボタンをクリックしてブランチ比較ページへ移動したりブランチを削除することができる。

[Overview](#) [Active](#) [Stale](#) [All branches](#)
 Search branches...

Default branch

master Updated an hour ago by fxn



Default

Active branches

4-2-stable Updated 2 hours ago by sgrif



1147 | 269

[Compare](#)

4-1-stable Updated 4 days ago by rafaelfranca



6892 | 672

[Compare](#)

fix_nested_transactions_for_re... Updated 8 days ago by chancancode



149 | 3

[Compare](#)

3-2-stable Updated 12 days ago by rafaelfranca



22478 | 1757

[Compare](#)

4-0-stable Updated 12 days ago by rafaelfranca



11974 | 1081

[Compare](#)[View more active branches >](#)

プランチ同士の比較

GitHubのプランチ比較は以下のようなURLで提供されている:

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

{range} を master...4-1-stable に変更する。

例えば:

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

We're showing branches in this repository, but you can also compare across forks.

base: [master](#) ... compare: [4-1-stable](#)[Create pull request](#)

Discuss and review the changes in this comparison with others.

[?](#)[Commits](#) 672[Files changed](#) 509[Commit comments](#) 5

45 contributors

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

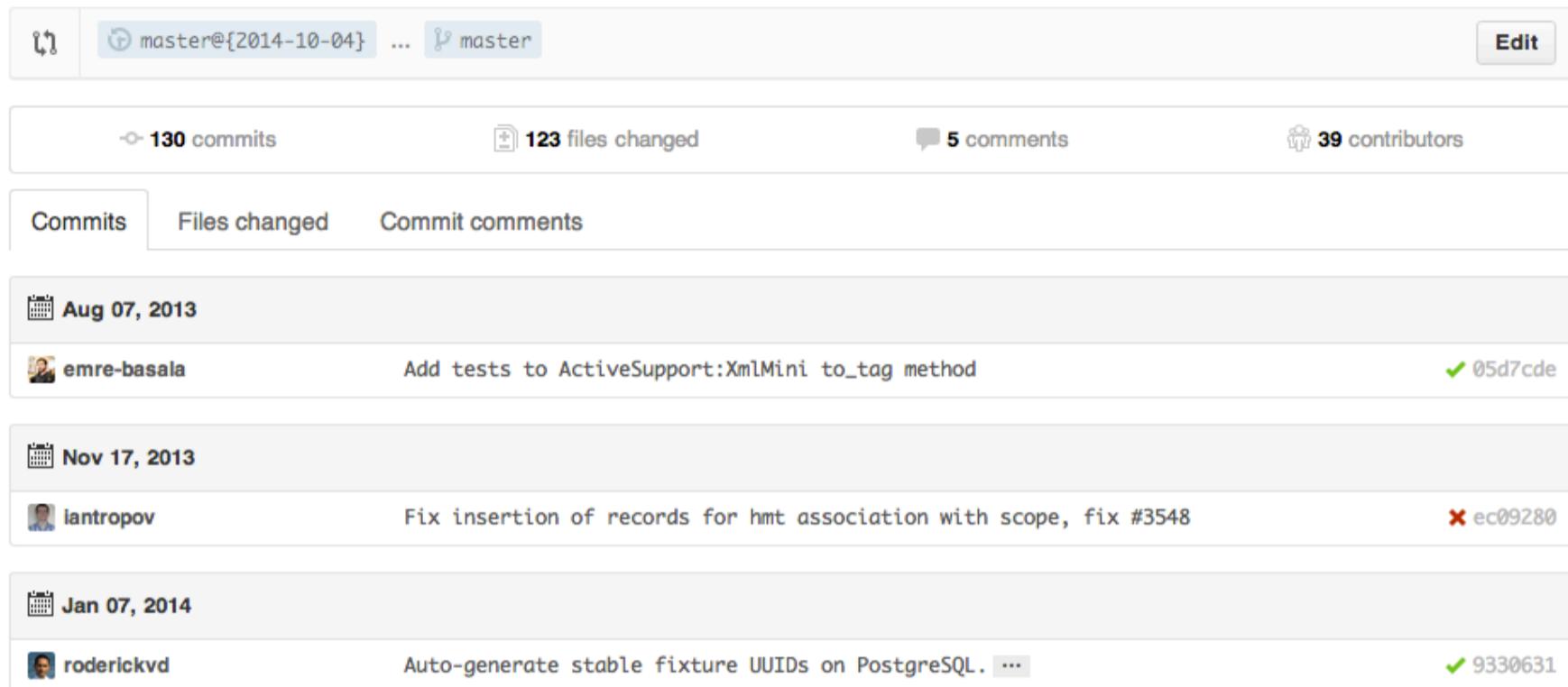
Commits on Jun 19, 2014

-  rafaelfranca Merge pull request #15813 from DNNX/valid-action-name-refactoring ... ✓ 6413eb4
-  rafaelfranca Fix has_and_belongs_to_many in a namespaced model pointing to a non n... ✓ 0769465
-  rafaelfranca Fix has_and_belongs_to_many in a namespaced model pointing to a non n... ✓ 1a12bee
-  rafaelfranca Merge pull request #15772 from nbudin/sti_through_bug ... ✓ 8f76511
-  rafaelfranca Merge pull request #15772 from nbudin/sti_through_bug ... ✓ a44feed
-  rafaelfranca Merge pull request #15729 from sgrif/sg-double-save-hm-t-4-1-stable ... ✓ ae0d952

{range} には以下のように変更することもできる:

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

日付の形式は YYYY-MM-DD だ。



The screenshot shows a GitHub commit comparison page. At the top, it displays two branches: 'master@{2014-10-04}' and 'master'. Below this, key statistics are shown: 130 commits, 123 files changed, 5 comments, and 39 contributors. Navigation tabs include 'Commits' (selected), 'Files changed', and 'Commit comments'. The main area lists commits grouped by date:

- Aug 07, 2013**: emre-basala - Add tests to ActiveSupport::XmlMini to_tag method (green checkmark, commit hash 05d7cde)
- Nov 17, 2013**: iantropov - Fix insertion of records for hmt association with scope, fix #3548 (red X, commit hash ec09280)
- Jan 07, 2014**: roderickvd - Auto-generate stable fixture UUIDs on PostgreSQL. ... (green checkmark, commit hash 9330631)

diff や patch のビューでもブランチを利用して比較することができる:

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

時間を指定してのブランチ比較についてもつと詳しく

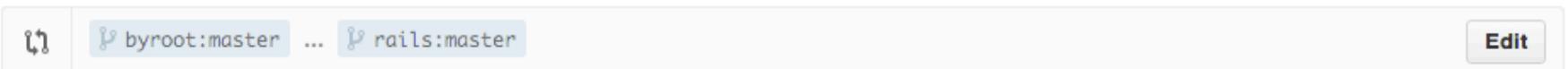
フォークされたリポジトリ間でのブランチ比較

GitHubでフォークされたリポジトリ同士でブランチを比較する場合、以下のようなURLを変更する:

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

例:

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



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は少量のコード群を管理する最適な手段だ。ちゃんとしたリポジトリをいちいち作成する必要はない。

[GitHub Gist](#) Search... Discover Gists rafalchmiel [Edit](#) [Raw](#) [View](#)

PUBLIC Star 0 Fork 0

A simple Ruby program.

Gist Detail **app.rb** Ruby

Revisions 1

```
1 puts 'Hello World'
```

[Download Gist](#) rafalchmiel commented 2 days ago
Wow, dat is some engineering.

Clone this gist <https://gist.github.com/tiimgreen/app.rb> Embed this gist <script src='https://gist.github.com/tiimgreen/app.rb'> Link to this gist <https://gist.github.com/tiimgreen/app.rb>

Write **Preview** Comments are parsed with GitHub Flavored Markdown

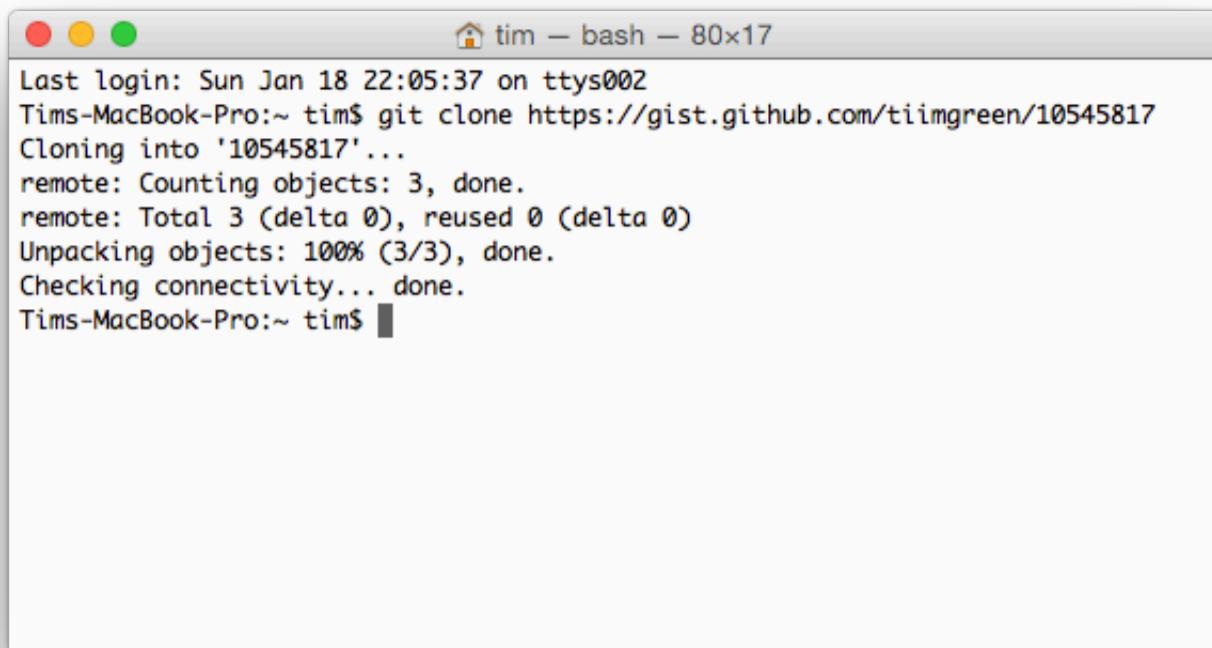
Leave a comment

[Add Comment](#)

GistのURLの最後に .pibb を付ける(例)とHTMLのみのバージョンが表示されるので、そのソースは他のウェブサイトに貼り付けるにはもってこいだろう。

簡単なものとはいえ、Gitリポジトリとして機能するため、以下のようにすれば普通のGitリポジトリと同じようにクローンすることができる:

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



```
Last login: Sun Jan 18 22:05:37 on ttys002
Tims-MacBook-Pro:~ tim$ git clone https://gist.github.com/tiimgreen/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$
```

This means you also can modify and push updates to Gists:

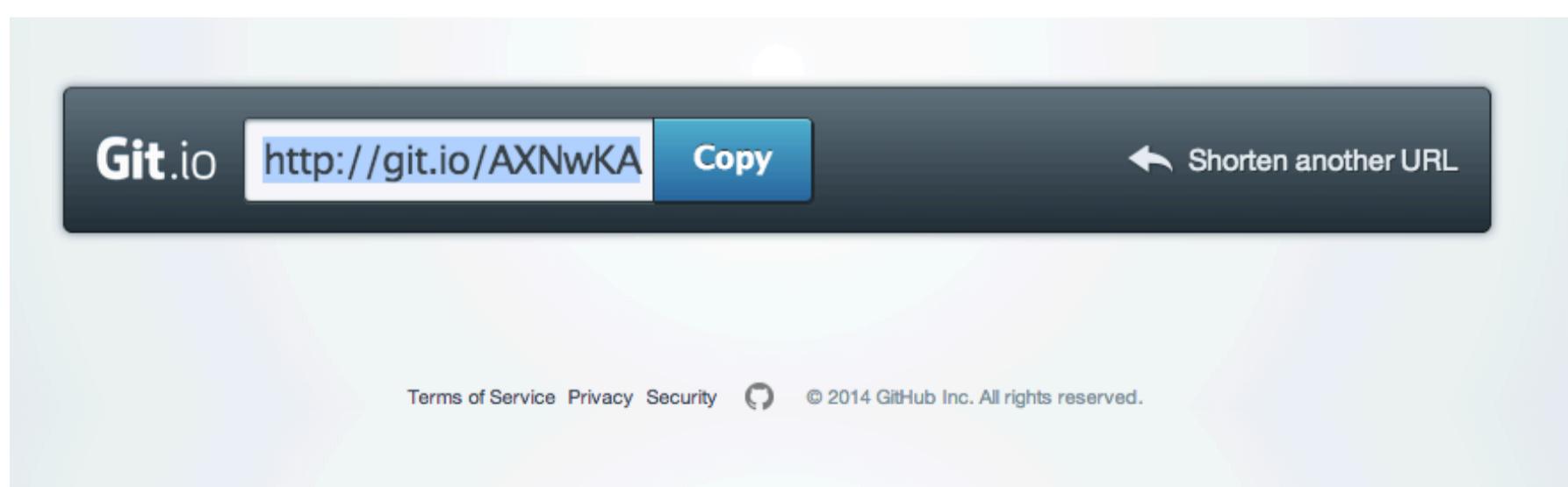
```
$ git commit
$ git push
Username for 'https://gist.github.com':
Password for 'https://tiimgreen@gist.github.com':
```

しかしながら、Gistではディレクトリーがサポートされていない。全てのファイルはリポジトリのルートに置かれている必要がある。

Gistの作成についてもつと詳しく

Git.io

Git.ioはGitHubの提供するGitHub専用のシンプルな短縮URLサービスだ。



cURLを使って利用することができる:

```
$ 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/...
```

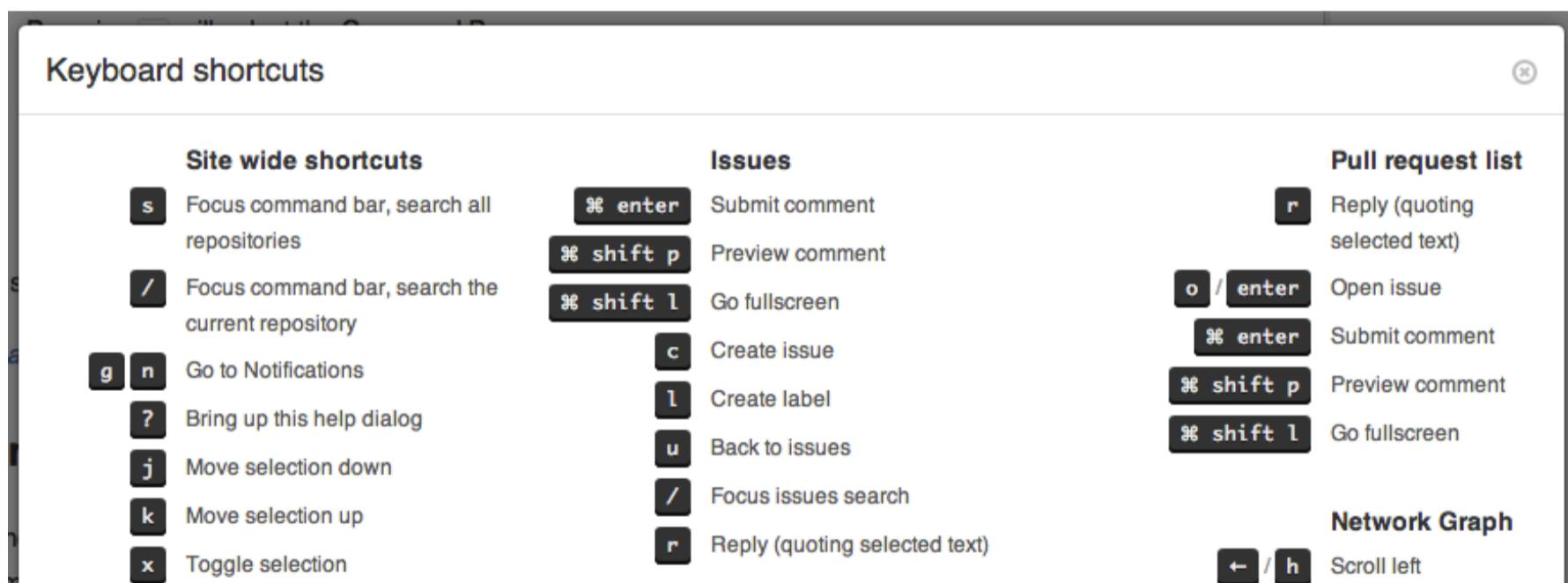
Git.ioについてもつと詳しく

キー・ボード・ショートカット

リポジトリをブラウザーで開いている時は、ショートカットを利用して様々な機能へ簡単にアクセスできるようになっている。

- t を押すとファイルの検索インターフェイスが起動する。
- w を押すとブランチ選択インターフェイスが起動する。
- s を押すと現在閲覧しているリポジトリから検索するフォームにフォーカスが当たる。ここでBackspaceを押し「This repository」という文字列を消すことでGitHub全体からの検索へと切り替えることができる
- issue画面で l を押すとラベルの編集インターフェイスが開かれる。
- ファイルを参照している時（例: <https://github.com/tiimgreen/github-cheat-sheet/blob/master/README.md>）に y を押すと、参照している時の状態で固定されるURLに変更される。つまりそのファイルのコードが後に変化したとしても、そのURLでは今とまったく同じ状態で表示されるということだ。

? を押すとそのページで使える全ショートカットが表示されるだろう。



検索機能についてもつと詳しく

コードの指定行の強調

コードのURLの末尾に #L52 と付けるか行番号をクリックすると、その行が強調表示される。

これは範囲指定も可能だ（例: #L53-L60）。こういった範囲を選択するには shift を押しながら二つの行をクリックしても良い：

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   ActiveRecord::Serializers.eager_load!
65 end
66
67
68 ActiveSupport.on_load(:i18n) do
69   I18n.load_path << File.dirname(__FILE__) + '/active_model/locale/en.yml'
70 end

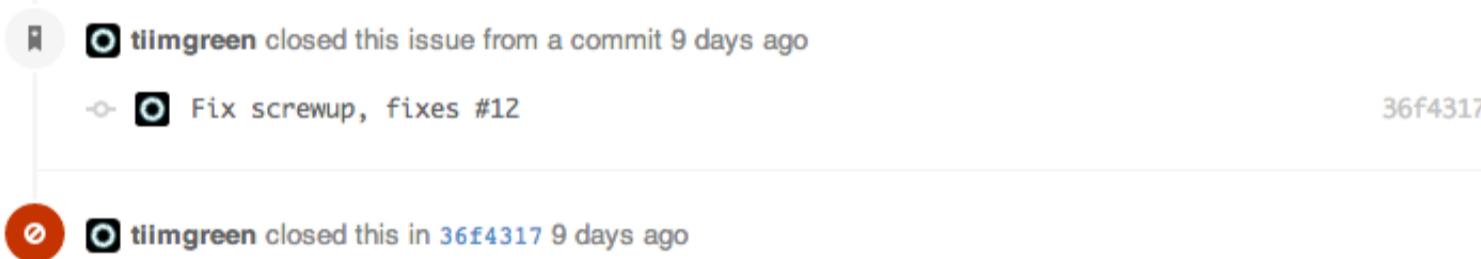
```

コミットからissueを閉じる

あるコミットでissueを解決した場合、コミットメッセージで fix/fixed/fixed や close/closes/closed 、 resolve/resolves/resolved に続けてissue番号を指定すると、そのコミットがmasterブランチにpushされると同時に指定issueが閉じられるだろう。

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

こうするとissue#12が閉じられ、閉じたissueにはそのコミットへの参照が自動的に追加される。



tiimgreen closed this issue from a commit 9 days ago
 Fix screwup, fixes #12 36f4317

tiimgreen closed this in 36f4317 9 days ago

コミット・メッセージからissueを閉じる方法についてもつと詳しく

issueの相互リンク

同じリポジトリの違うissueへリンクを張り参照させたい場合、 # に続けてissue番号を指定する。そうすると自動的にリンクが作成されるだろう。

別のリポジトリのissueの場合は {user}/{repo}#ISSUE_NUMBER とすれば良い（例: tiimgreen/toc#12）。

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

A screenshot of a GitHub pull request page. At the top, there's a profile picture of a man with a mustache. Below it, a comment from user 'sr' is shown with a red arrow pointing to the text 'sr referenced this issue in github/github'. To the right of the comment is the date 'October 14, 2011'. Further down, the pull request summary 'Pull Request #1773: Update footer for new formats' is listed next to a red 'Closed' button.

コメントのロック

リポジトリのオーナーや共同開発者ならば、pull requestやissueへのコメントをロックできるようになった。

A screenshot of a GitHub issue comment form. At the top, there are two buttons: 'Parsed as Markdown' and 'Edit in fullscreen'. On the right, there's a 'Lock issue' button with a padlock icon. Below these, there are two tabs: 'Write' and 'Preview'. In the main area, there's a large text input field containing a padlock icon and the text 'This conversation has been locked and limited to collaborators.'

つまりそのプロジェクトの共同開発者ではないユーザーはコメントをすることができないということだ。

A screenshot of a GitHub issue comment form. At the top, there are two tabs: 'Write' and 'Preview'. In the main area, there's a large text input field containing a padlock icon and the text 'This conversation has been locked and limited to collaborators.'

コメントのロックについてもつと詳しく

pull requestでのCI結果の表示

適切に設定を行えばpull requestを受け取るたびに、通常のコミットと同じようにTravis CIがそのpull requestをビルドするだろう。どう設定するかはTravis CI: Getting startedを読むと良い。

This repository Search or type a command Explore Gist Blog Help rafalchmiel + × ↗

PUBLIC octokit / octokit.rb Watch 68 Unstar 1,510 Fork 291

Lazy create test repos #452

Open joeyw wants to merge 2 commits into master from lazy-create-test-re...

Conversation 0 Commits 2 Files changed 1

joeyw commented a month ago Alternative to mass repo creation in #429 Checks for and creates test repositories before recording new cassettes.

joeyw added some commits a month ago

- Lazily create test repositories on new cassette recording ✓ dfce8db
- Ignore test repo setup interactions so they aren't saved in cassettes ✓ 269b525

All is well — The Travis CI build passed · Details

Write Preview Comments are parsed with GitHub Flavored Markdown

Labels None yet

Milestone No milestone

Assignee No one assigned

Notifications

Subscribe You'll be notified when you participate or are @mentioned in this thread.

コミット・ステータスAPIについてもつと詳しく

絞り込み検索

issueとpull requestの検索インターフェイスでは、絞り込みをかけることが可能だ。

例えばRailsのリポジトリでは、以下の様な絞り込み検索により"activerecord"というラベルがついたissueのみを表示することができる:

```
is:issue label:activerecord
```

逆に"activerecord"というラベルがついていないissueのみを表示することもできる:

```
is:issue -label:activerecord
```

そして、この絞り込み検索はpull requestに対しても行うことができる:

```
is:pr -label:activerecord
```

GitHubでは開かれているissueやpull requestのみを表示するタブ、または既に閉じられたそれらのみを表示するタブが用意されているが、絞り込み検索によってマージ済みのpull requestのみを表示することもできる。以下のようなフィルターを使えば良いだろう:

```
is:merged
```

issueの検索についてもつと詳しく

そしてGitHubはStatus APIの結果を使ってフィルターすることもできるようになった。

Status APIでsuccessが設定されたpull requestのみ:

```
status:success
```

Status APIを使った絞り込み検索についてもつと詳しく

Markdownファイルでの構文強調

例えばMarkdownファイルでRubyのコードを構文強調したいならば以下のようにする:

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

```

こうすると以下のように表示されることになる:

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

```

GitHubではLinguistを使って言語を判別し構文強調を行っている。構文強調がサポートされている言語の一覧は言語定義YAMLファイルを参照すればわかるだろう。

*GitHub Flavored Markdown*についてもつと詳しく

Emoji

Emojiはpull requestやissue、コミット・メッセージ、リポジトリの概要などで :name_of_emoji: と書くと利用できる:

GitHubでサポートされているEmojiの完全なリストはEmoji cheat sheet for Campfire and GitHubかAll-Github-Emoji-Iconsで確認できる。素敵なemojiの検索はemoji.muan.co。

GitHubで使われているEmojiのトップ5は以下の通りだ:

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

画像及びアニメーションGIF

画像やアニメーションGIFはコメントやREADMEなどで利用できる:

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

リポジトリにある画像も直接参照することが出来る:

![Alt Text](<https://github.com/{user}/{repo}/raw/master/path/to/image.gif>)



あらゆる画像はGitHubでキャッシュされるので、画像のホスティング先が落ちていたとしても変わらず表示されるだろう。

GitHub Wikiへの画像の添付

GitHub Wikiで画像を追加する方法がいくつかある。通常のMarkdown記法（前節を参照）はもちろん使える。しかしそれだけではなく、画像の幅と高さを指定する記法も使うことができる:

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

こうすると以下のようになる:

Home Pages History New Page

Home (Preview)

[\[http://www.sheawong.com/wp-content/uploads/2013/08/keepthatin.gif | height = 100px\]](#)

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](#).

素早く引用

issueのスレッドで他の人のコメントを引用してコメントしたい場合、引用したい文章を選択した状態で `r` を押すと、ブロック引用の記法を使ってテキストエリアにコピーされる。



rafalchmiel commented 44 minutes ago

Collaborator

Ahhh, I found what the issue was and also found a solution. Some people had [similar issues](#). Adding `noglob` to the beginning of this command seems to work just fine. You might want to add that into your PR:

```
→ github-cheat-sheet git:(master) noglob git fetch origin +refs/pull/*/head:refs
remote: Counting objects: 14, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 2), reused 5 (delta 1)
Unpacking objects: 100% (6/6), done.
From github.com:tiimgreen/github-cheat-sheet
 * [new ref]      refs/pull/1/head -> refs/pull/1
 * [new ref]      refs/pull/2/head -> refs/pull/2
 * [new ref]      refs/pull/3/head -> refs/pull/3
```



This pull request can be automatically merged.

You can also merge branches on the [command line](#).



[Merge pull request](#)



[Write](#) [Preview](#)

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

> Ahhh, I found what the issue was and also found a solution.



Attach images by dragging & dropping, [selecting them](#), or pasting from the clipboard.

ProTip Add comments to specific lines under Files changed.

[Close & Comment](#)

[Comment](#)

素早く引用する方法についてもつと詳しく

コメントへのクリップボードの画像の追加

(Chrome系のブラウザーのみで動作)

スクリーンショットをクリップボードに保存(mac: cmd+ctrl+shift+4)した後、その画像はコメント投稿フォームで貼り付け(cmd+v または ctrl+v)ことができ、自動的にGitHubへアップロードされます。

[Issues](#) [Pull requests](#) [Labels](#) [Milestones](#)

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



Clipboard image paste

Write

Preview

Markdown supported

I think this is a huge protip. Capture screenshot (region) to clipboard. Then cmd-v it.

![Uploading image.png . . .]()

コメントへのクリップボードの画像の追加についてもと詳しく

設定済みライセンスの追加

GitHub上でリポジトリを作成する時、あらかじめ設定されているライセンスを追加することもできる:

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

Add .gitignore: **None** ▾

Add a license: **None** ▾

Create repository

Licenses

Filter licenses...

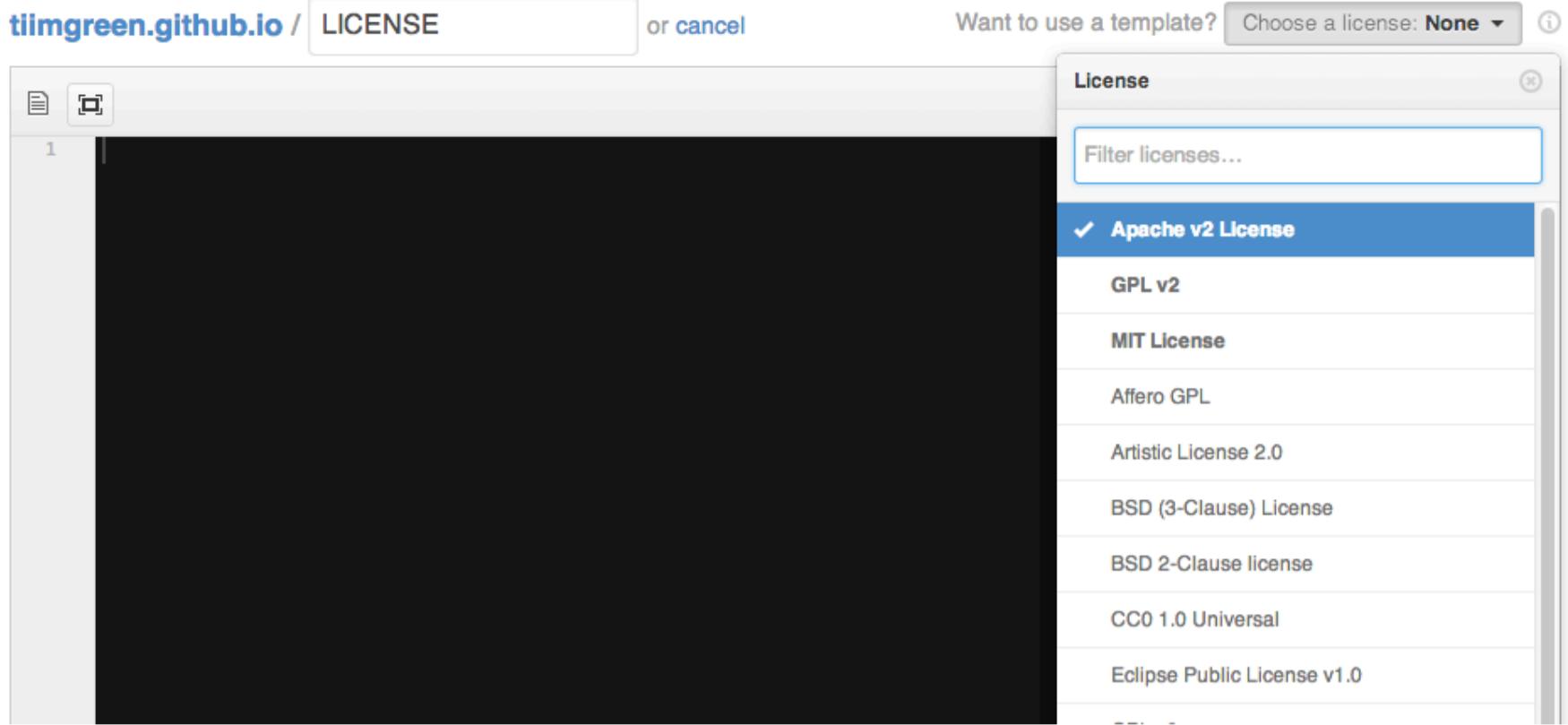
GPL v2

MIT License

Affero GPL

[Terms](#) [Privacy](#) [Security](#) [Contact](#)

既に存在するリポジトリであってもウェブ上のインターフェイスからファイルを作成することで追加できる。 LICENSE というファイル名にした場合、ライセンスを選択するオプションが表示されるのだ:



.gitignore も同じように作成時に追加することも、後で追加することもできる。

オープンソース・ライセンスについてもつと詳しく

タスクリスト

issueやpull requestでは以下のように（空白に注意）書くとチェックボックスを作成することができる：

- [] 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

これらチェックボックスにチェックが入れられると、同時にMarkdownソースも更新される：

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

タスク・リストについてもつと詳しく

Markdownファイルでのタスクリスト

通常のMarkdownファイルでも読み取り専用のチェックリストを以下のような記法で追加することができる:

- [] Mercury
- [x] Venus
- [x] Earth
- [x] Mars
- [] Jupiter

Mercury

Venus

Earth

Mars

Jupiter

Markdownファイルでのタスクリストについてもつと詳しく

相対リンク

Markdownファイルでリポジトリ内のコンテンツへ張る場合、相対リンクを利用することが推奨されている。

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

絶対リンクはURLの変更（例: リポジトリのリネーム、ユーザー名の変更、プロジェクトのフォーク）により更新される。相対リンクを利用すれば、そのままうまく機能するはずだ。

相対リンクについてもつと詳しく

GitHub Pagesでのメタデータとプラグインのサポート

Jekyllのページや投稿ではリポジトリの情報が `site.github` という名前空間に格納されており、例えば `{} site.github.project_title {}` などと書けば表示することができる。

また、`Jemoji`と`jekyll-mentions`というプラグインがインストールされているので、`Emoji`や`@mentions`はJekyllの投稿やページでGitHub.com上と同じように動作する。

[GitHub Page](#)でのメタデータとプラグインのサポートについてもつと詳しく

文書内のYAMLメタデータ

JekyllをGitHub Pagesで利用している場合など、多くのブログではYAML形式のメタデータをその記事の先頭に書く必要がある。GitHubではこういったメタデータを読みやすいように表として表示してくれる:

The screenshot shows a GitHub file page for a file named 'classnotes/2013-02-13-NYU-github-class.md'. The file contains 61 lines (39 sloc) and is 1.415 kb in size. The page includes standard GitHub navigation buttons: Open, Edit, Raw, Blame, History, and Delete. Below the navigation, there is a table displaying the YAML metadata:

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

Below the table, the main content of the page is displayed:

Your instructors for the evening are:

- Matthew McCullough ([Twitter](#), [GitHub](#))
- Tim Berglund ([Twitter](#), [GitHub](#))

文書内のYAMLメタデータの表示についてもつと詳しく

表形式のデータ表示

GitHubでは .csv (カンマ区切り) と .tsv (タブ区切り) の形式で書かれた表を整形して表示する機能をサポートしている。

The screenshot shows a GitHub file page for a file named 'locations.csv'. The file contains 869 lines (868 sloc) and is 188.222 kb in size. The page includes standard GitHub navigation buttons: Edit, Raw, Blame, History, and Delete. Below the navigation, there is a search bar labeled 'Search this file...' and a table displaying movie location data:

| | Title | Release Year | Locations | Fun Facts | Production Company |
|----|------------------------|--------------|--|---------------------------------------|---------------------------|
| 1 | 180 | 2011 | 555 Market St. | | SPI Cinemas |
| 2 | 180 | 2011 | Epic Roasthouse (399 Embarcadero...) | | 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 Manufac |
| 13 | A Jitney Elopement | 1915 | 20th and Folsom Streets | | The Essanay Film Manufac |
| 14 | A Night Full of Rain | 1978 | San Francisco Chronicle (901 Mis...) | The San Francisco Zodiac Killer of... | Liberty Film |

表形式のデータ表示についてもつと詳しく

PDF表示

GitHubではPDFの表示をサポートしている:



PDF表示についてもつと詳しく

pull requestの取り消し

pull requestをマージした後、意味がわからなかったことがわかつたり、そのpull requestをマージしたのは間違いだったことがわかることがあるだろう。

その取り消しは、pull requestのページに表示されているマージ・コミットの右端にある**Revert**ボタンをクリックすることにより、そのpull requestで行われた変更を取り消すpull requestを作成することによって行うことができる。

mdo and others added some commits 5 hours ago

- Auto-merged master into code_list_cleanup on deployment ✓ 9a8163c
- fix bg colors ✘ 5b880a0
- linting ✓ 98be15c

octocat merged commit `6b79af4` into `master` from `code_list_cleanup` 5 hours ago

octocat closed this 5 hours ago

Revert

Create a new pull request to revert these changes

pull requestの取り消しについてもつと詳しく

差分の表示

レンダリング済みの差分表示

コミットやpull requestにGitHubでレンダリングされて表示されるもの（例: Markdown）が含まれる場合、そのソースとレンダリング済みの両方の差分を見ることができる。

```
21 ████ rendered-prose-diffs.md
...
... @@ -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
```

レンダリングされた状態での差分を表示したい場合は「Rendered」ボタンをクリックする。レンダリング済みの差分表示では文章の追加や削除、編集がよりわかりやすい：

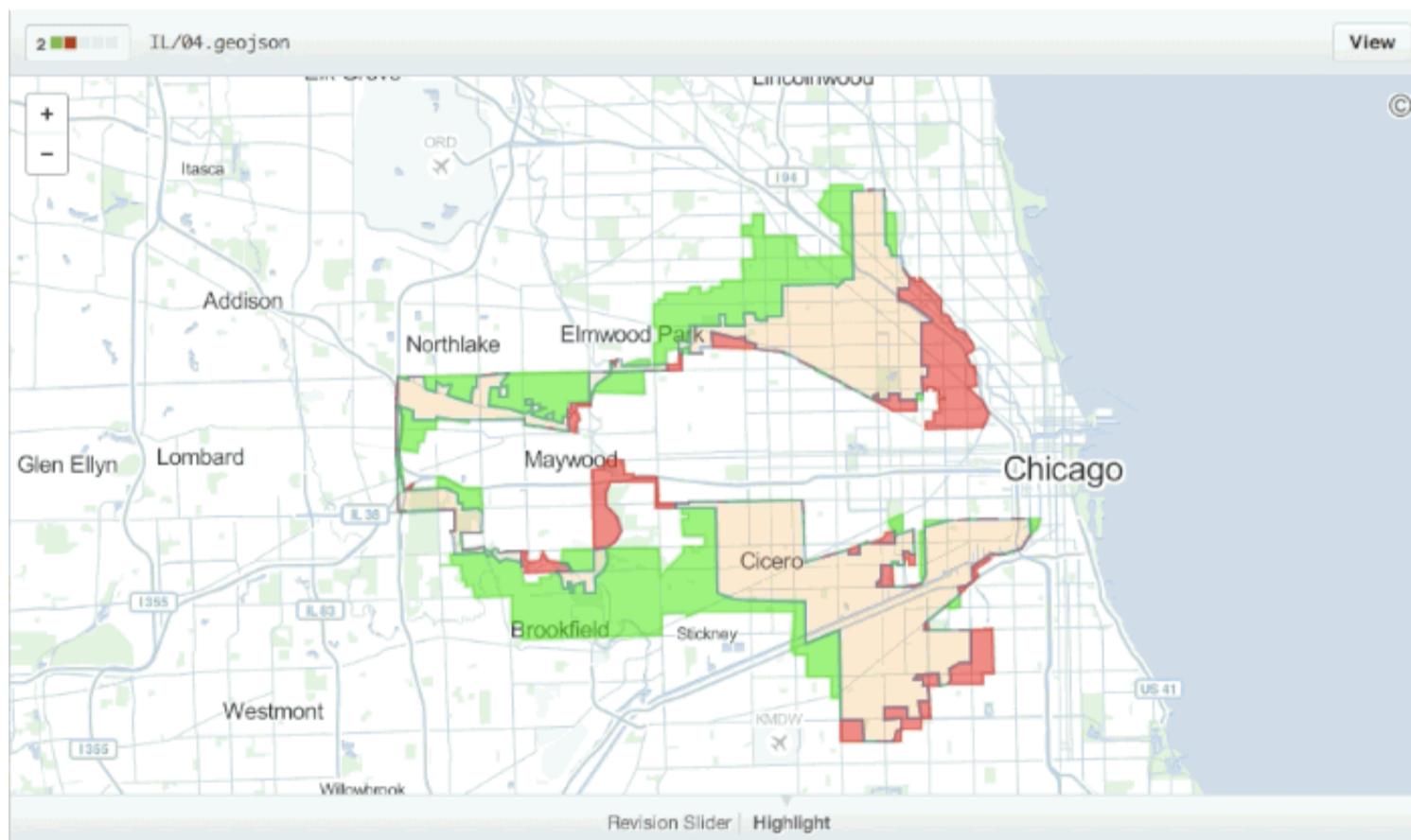
~~Obviously, you'll want to change <your_client_id> to match your actual Client ID.~~

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.

レンダリング済みの差分表示についてもつと詳しく

マップ差分の可視化

コミットやpull requestにジオデータの変更が含まれている場合はいつも、GitHubではそのジオデータの変化を可視化してくれるだろう。



マップ差分の可視化についてもつと詳しく

差分表示の前後を表示

差分表示の行番号付近にある展開ボタンを使うと、その前後の行をクリックして表示させることができる。展開ボタンを押し続けることによってファイル全体を表示することもできるし、またこの機能はあらゆるGitHubの差分表示ビューに用意されている。

```

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 //
99 @@ -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;
244 258     - [self setDefaultHeader:@"Accept" value:@"application/vnd.github.beta+json"];
245 259     -
246 260     [AFHTTPRequestOperation addAcceptableStatusCodes:[NSIndexSet indexSetWithIndex:OCTClientNotModifiedStatusCode]]
247 261     - [AFJSONRequestOperation addAcceptableContentTypes:[NSSet setWithObject:@"application/vnd.github.beta+json"]];
248 262     +
249 263     + NSString *contentType = [NSString stringWithFormat:@"application/vnd.github.%@+json", OCTClientAPIVersion];
250 264     + [self setDefaultHeader:@"Accept" value:contentType];
251 265     + [AFJSONRequestOperation addAcceptableContentTypes:[NSSet setWithObject:contentType]];
252 266     +
253 267     + self.parameterEncoding = AFJSONParameterEncoding;
254 268     [self registerHTTPOperationClass:AFJSONRequestOperation.class];
255 269
256 270     return self;

```

差分表示の前後を表示についてもつと詳しく

pull requestの内容をDIFFまたはPATCH形式で取得

pull requestによる差分はそのURLの末尾に .diff または .patch を追加すると、それぞれの形式で取得することができる。例えば:

```

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

```

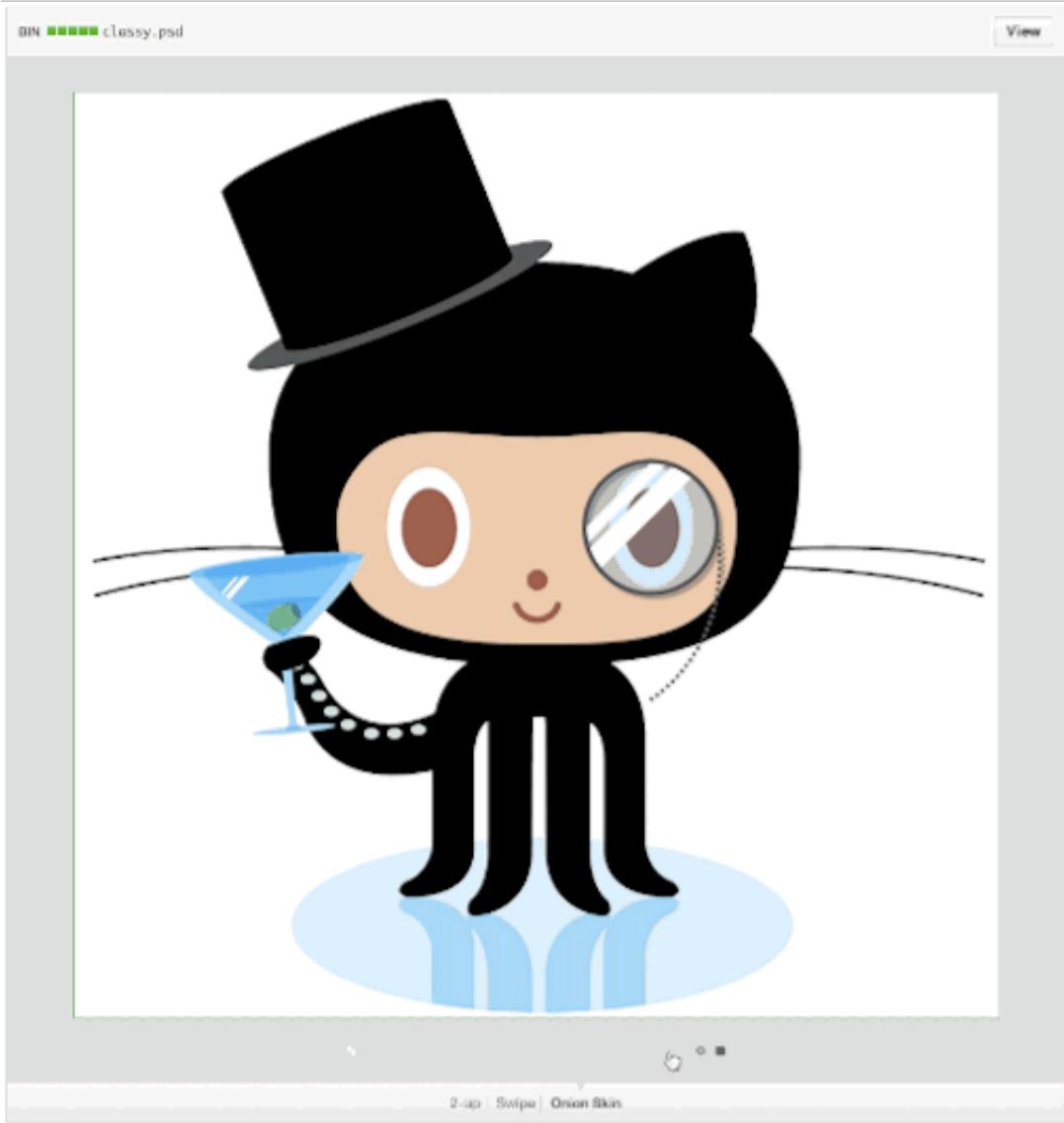
拡張子 .diff を追加した場合、このようなプレーンテキストで表示されるだろう:

```

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
 - [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
(...)
```

画像の表示とDIFF

GitHubは、PNGやJPG、GIF、PSDといった多くの一般的な画像形式の表示をサポートしている。それに加え、様々な方法でこれら画像形式のバージョンごとの差分を比較することもできる。



画像の表示とDIFFについてもつと詳しく

Hub

HubはGitのラッパーとして機能するコマンドライン・ツールで、これを利用するとGitHubをコマンドラインからとても簡単に扱えるようになる。

例えば以下のようにしてリポジトリのクローンが行える:

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

これが以下のコマンドの代わりというわけだ:

```
$ git clone https://github.com/tiimgreen/toc.git
```

Hubが提供する便利な機能についてもつと詳しく

開発参加のガイドライン

リポジトリのルートに CONTRIBUTING という名前のファイルを置くと、issueやpull requestを作成しようとした時にそれへのリンクが表示されるようになる。

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

Houston, we have a problem!

Write Preview Comments are parsed with GitHub Flavored Markdown

Leave a comment

開発参加のガイドラインについてもつと詳しく

Octicons

GitHubで使われているアイコン（Octicons）はオープンソース化された。



GitHubのOcticonsについてもつと詳しく

GitHub情報

| 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 Talks

| 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

すべての削除済みファイルをワーキング・ツリーから削除する

例えば `/bin/rm` を使って大量のファイルを削除した場合、以下のコマンドを使えばワーキング・ツリー、そしてインデックスからも削除することができ、ファイルごとにそれぞれGitコマンドを実行する必要がなくなる：

```
$ git rm $(git ls-files -d)
```

例えば以下のように実行される：

```
$ git status
On branch master
Changes not staged for commit:
  deleted:    a
  deleted:    c
```

```
$ git rm $(git ls-files -d)
rm 'a'
rm 'c'
```

```
$ git status
On branch master
Changes to be committed:
  deleted:    a
  deleted:    c
```

直前のブランチ

Gitで直前のブランチへ移動するには：

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

$ git checkout -
# Switched to branch 'next'

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

空白の削除

Gitの `stripspace` コマンドは以下の作業を行う:

- 行末の空白文字の削除
- 空白行の取りまとめ
- ファイル末尾への改行の追加

このコマンドを呼ぶ時はファイルを渡さねばならない。例:

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

Gitの `stripspace` コマンドについてもつと詳しく

SSH公開鍵

以下のURLにアクセスすることで、SSH公開鍵の一覧をテキスト形式で取得することができる:

<https://github.com/{user}.keys>

例: <https://github.com/tiimgreen.keys>

SSH公開鍵についてもつと詳しく

pull requestのチェックアウト

pull requestはGitHubのリポジトリでは特別なブランチであり、様々な方法でローカルに取り込むことができる:

特定のpull requestを取り込み、一時的に `FETCH_HEAD` として保存すると、素早く差分を確認してマージすることができる:

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

参照仕様を使うとすべてのpull requestをローカル・ブランチとして取り込むことができる:

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

リポジトリの `.git/config` に以下の行を追加すれば自動的にpull requestを落とすようにもできるだろう:

```
[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/*
```

フォークされてから送ってきたpull requestに対しては、そのpull requestを参照するリモート・ブランチから直接ローカル・ブランチとしてチェックアウトすると便利だろう:

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

もしくは様々なリポジトリで作業をするのなら、代わりにグローバルのGit設定で行うことにより、pull requestの取得をグローバルに設定すると良いだろう。

```
git config --global --add remote.origin.fetch "+refs/pull/*:refs/remotes/origin/pr/*"
```

こうすると以下の様な短いコマンドを利用することが可能になる:

```
git fetch origin
```

```
git checkout pr/42
```

*pull request*のチェックアウトについてもつと詳しく

空のコミット

--allow-empty オプションを付けると、コードの変化がなくてもコミットを作成することができる:

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

この機能の使い方（便利なもの）としては以下のようものが挙げられる:

- 新たな機能や大きな変更を事前に通知する時
- コード上に現れないような変更をプロジェクトに加えた時
- リポジトリを利用して誰かと連絡を取りたい時
- リポジトリへの最初のコミットをやり直しできるようにしたい時: `git commit -m "init repo" --allow-empty` .

Gitステータスのスタイリング

普通に実行すると:

```
$ git status
```

このように表示されるが:

```
Tim's-MacBook-Pro:ghcs 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")
Tim's-MacBook-Pro:ghcs tim$
```

-sb を追加することによって:

```
$ git status -sb
```

このように表示することもできる:

```
Tim's-MacBook-Pro:ghcs tim$ git status -sb
## master...origin/master
 M README.md
Tim's-MacBook-Pro:ghcs tim$
```

Gitのstatus コマンドについてもつと詳しく

Gitログのスタイル

以下のように実行すると:

```
$ git log --all --graph --pretty=format:'%Cred%h%Creset --%C(auto)%d%Creset %s %Cgreen(%cr) %C(bol
```

このように表示される:

```
ghcs - less - 155x33
Tims-MacBook-Pro:ghcs tim$ git log --all --graph --pretty=format:'%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --abbrev-com
mit --date=relative
* f122fd1 - (HEAD, origin/master, origin/HEAD, master) Remove note about Emojis in Markdown files, closes #93 (25 minutes ago) <Tim Green>
* 704d07f - CONTRIBUTING.md (29 minutes ago) <Tim Green>
* 3b04f63 - Pasting Clipboard Image to Comments, closes #92 (33 minutes ago) <Tim Green>
* f0fcf20 - Remove out-dated feature, fixes #87 (48 minutes ago) <Tim Green>
* 57f1b82 - Update links/images (54 minutes ago) <Tim Green>
* fba6fa9 - Merge pull request #91 from paulirish/patch-1 (73 minutes ago) <Tim Green>
|\ 
| * 1abd8af - Chrome extension for GH tab size (89 minutes ago) <Paul Irish>
|/
* 77ece52 - Merge pull request #89 from hail2u/ja-translation (5 months ago) <Tim Green>
|\
| * abdffef - Update Ja translation to 6f5cf8e (5 months ago) <Kyo Nagashima>
* | edea1aa - Merge pull request #88 from marocchino/ko-update (5 months ago) <Tim Green>
|\
|/
|/
* ee65723 - Update to 6c9cf82 (5 months ago) <Shim Tw>
|/
* 6c9cf82 - Merge pull request #86 from tonyxue/master (6 months ago) <Tim Green>
|\ 
| * 3961f0b - zh-cn update for 9c1f837 (6 months ago) <Tony Xue>
|/
* 6f5cf8e - Needs more :trollface: (6 months ago) <Rafal Chmiel>
* 9c1f837 - Merge pull request #85 from gWorldz/master (6 months ago) <Rafal Chmiel>
|\
| * b957df8 - Update README.md (6 months ago) <Doc gWorldz>
| * aa8d100 - Update README.md (6 months ago) <Doc gWorldz>
|/
* 6a1e7aa - Merge pull request #84 from TheCellarRoom/master (6 months ago) <Tim Green>
|\ 
| * 10afdf88 - Update README.md (6 months ago) <Che'senStats>
```

この設定はPaleszが考えたものだ。

これは後述の手順に従つてエイリアスへ追加することもできる。

Gitの log コマンドについてもつと詳しく

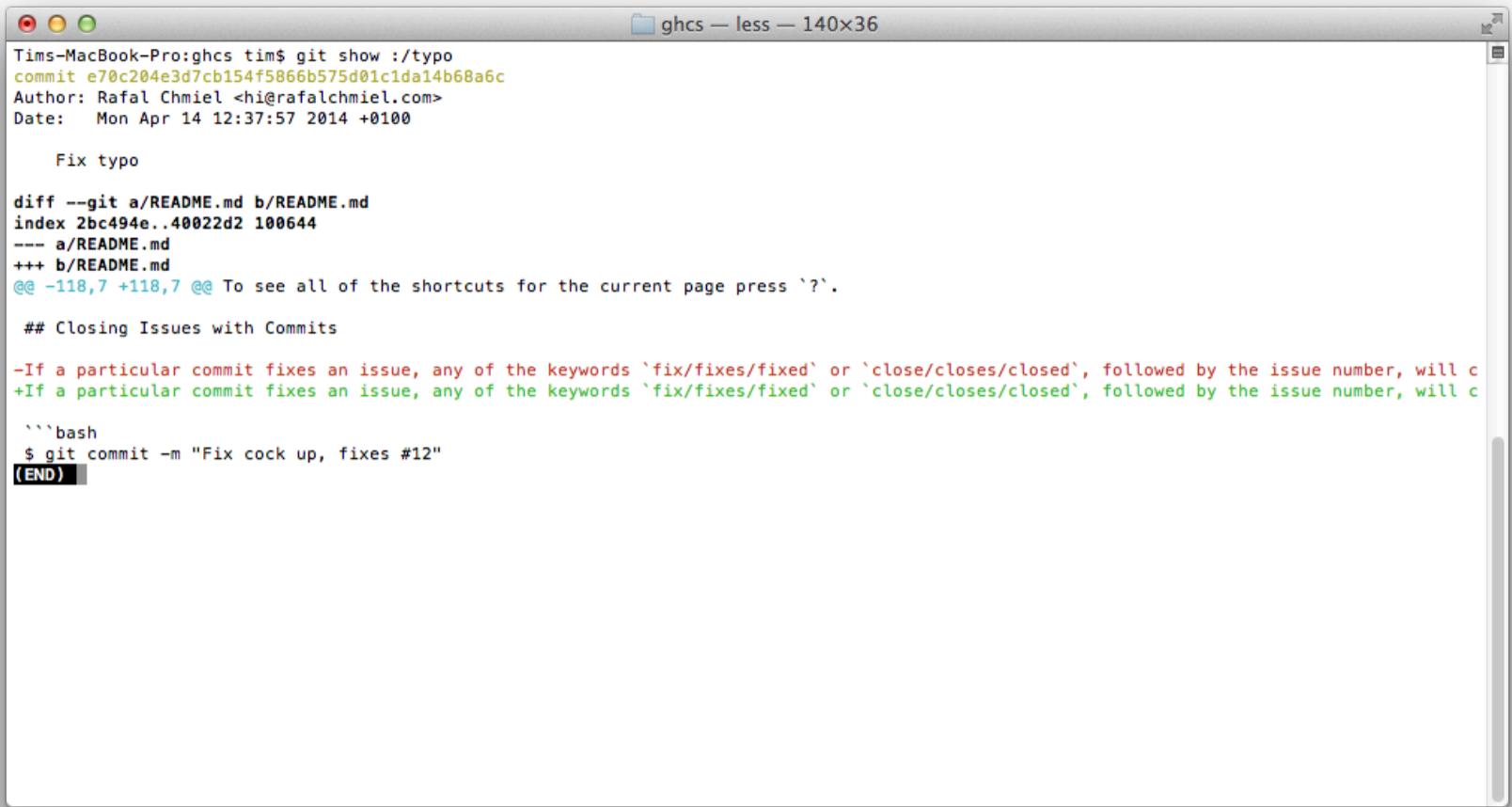
コミットログの検索

指定した文字列を今までのコミット・メッセージから検索して、もつとも新しいものを表示することができる。

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

query を検索したい文字列（大文字と小文字を区別する）で置き換えると、最新のコミットがそのコミットにおける差分と同時に表示される。

```
$ 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` or `close/closes/closed`, 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)

注: 終了するには q を押す。

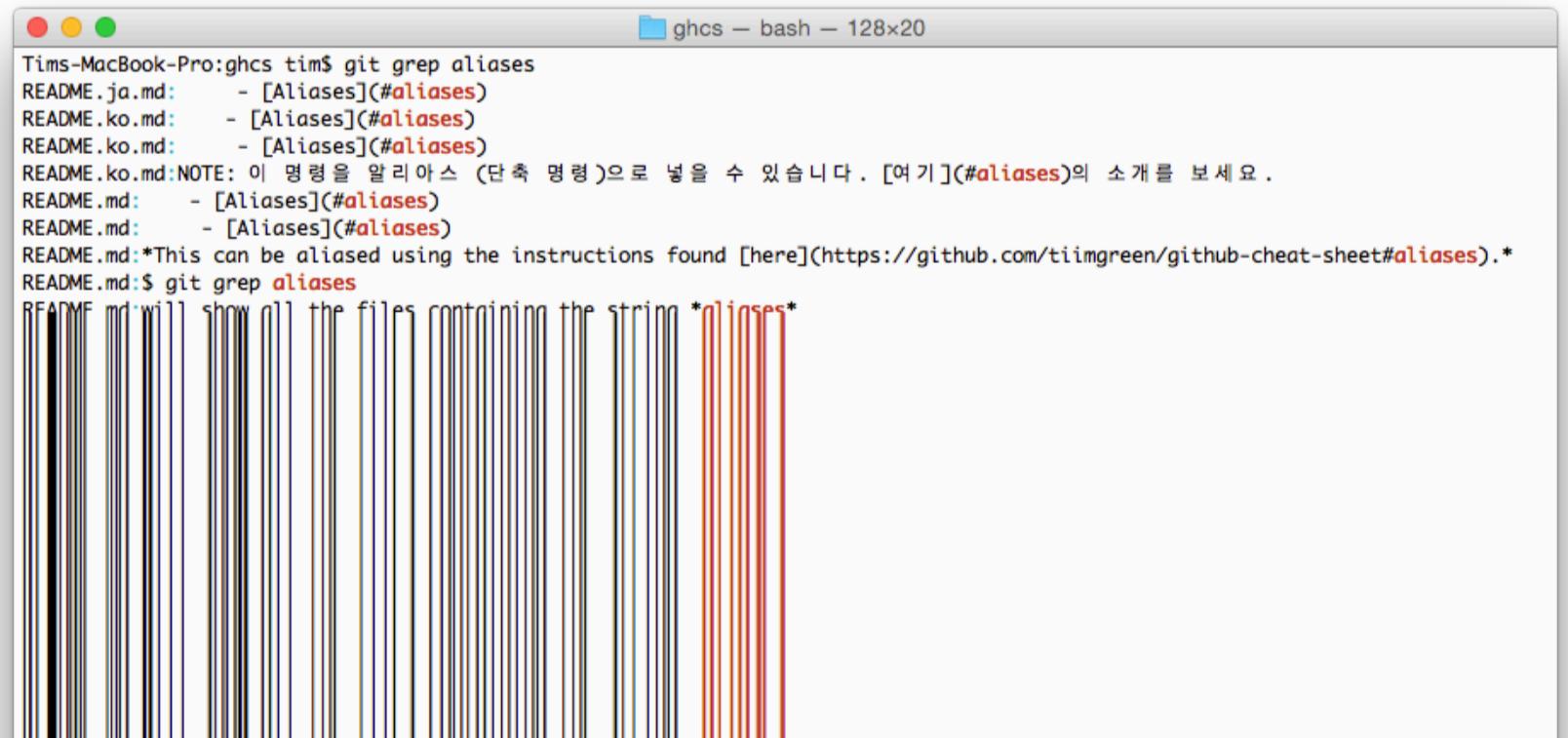
## Git Grep

Git Grepは指定したパターンに一致する行の一覧を返してくれる。

実行すると:

```
$ git grep aliases
```

aliasesという文字を含むファイルを全て表示することだろう。



Tims-MacBook-Pro:ghcs tim\$ git grep aliases  
README.ja.md: - [Aliases](#aliases)  
README.ko.md: - [Aliases](#aliases)  
README.ko.md: - [Aliases](#aliases)  
README.ko.md:NOTE: 이 명령을 알리아스 (단축 명령)으로 넣을 수 있습니다. [여기](#aliases)의 소개를 보세요.  
README.md: - [Aliases](#aliases)  
README.md: - [Aliases](#aliases)  
README.md:\*This can be aliased using the instructions found [here](https://github.com/tiimgreen/github-cheat-sheet#aliases).\*  
README.md:\$ git grep aliases  
README.md:will show all the files containing the string \*aliases\*

注: 終了するには q を押す。

複数のフラグを組み合わせることで、より高度な検索を行うことができる。例えば:

- `-e` 次のパラメーターをパターンとする(例: `regex`)
- `--and`、`--or`と`--not`複数のパターンを組み合わせる

以下のようにして使う:

```
$ git grep -e pattern --and -e anotherpattern
```

*Gitのgrepコマンドについてもつと詳しく*

## マージ済みブランチ

以下のように実行すると:

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

現在のブランチに既にマージされたブランチの一覧が表示される。

逆に:

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

こうするとまだマージされていないブランチが表示されるだろう。

*Gitのbranchコマンドについてもつと詳しく*

## FixupとAutosquash

これまでのコミット(HEADの直前でなくても構わない)、例えば`abcde`というチェックサムのコミットで何か問題を見つけた場合、以下の様なコマンドで問題の修正を行うことができる:

```
$ git commit --fixup=abcde
$ git rebase abcde^ --autosquash -i
```

*Gitのcommitコマンドについてもつと詳しく Gitのrebaseコマンドについてもつと詳しく*

## ローカル・リポジトリを参照するウェブサーバー

Gitの`instaweb`コマンドを利用すると、自分の作業リポジトリを`gitweb`で参照することができる。このコマンドは`gitweb`とウェブサーバーをセットアップしてローカル・リポジトリをブラウザーで開けるようにする簡単なスクリプトだ。

```
$ git instaweb
```

以下のようなページが開かれる:

summary | shortlog | log | commit | commitdiff | tree

commit ↻ 2 search:  re

description Unnamed repository; edit this file 'description' to name the repository.  
 owner Rafal Chmiel  
 last change Mon, 14 Apr 2014 20:04:18 +0100 (20:04 +0100)

**shortlog**

|             |              |                                                         |                                                                                      |                                                                                                       |
|-------------|--------------|---------------------------------------------------------|--------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| 2 min ago   | Rafal Chmiel | Add use-cases for empty commits, thanks to @davej ...   | <a href="#">master</a>   <a href="#">origin/HEAD</a>   <a href="#">origin/master</a> | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 10 min ago  | Tim Green    | Revert title back to original                           |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 12 min ago  | Rafal Chmiel | Add 'Git' to title                                      |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 15 min ago  | Rafal Chmiel | Update contributing guide to fit new organization       |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 17 min ago  | Rafal Chmiel | :star2: Organize sections (fixes #21) :star:            |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 68 min ago  | Rafal Chmiel | :star: Don't push, I'll be reorganizing the whole file  |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 2 hours ago | Tim Green    | Update intro paragraph                                  |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 2 hours ago | Tim Green    | Clarify Rendered Prose Diffs                            |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 3 hours ago | Rafal Chmiel | Add wiki images section under images/GIFs               |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 3 hours ago | Rafal Chmiel | Add 'Embedding Images in GitHub Wiki' section thanks... |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 3 hours ago | Rafal Chmiel | Add info about '.pibb' thanks to @jballanc :heart:      |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 3 hours ago | Rafal Chmiel | Make list of emojis prettier                            |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 3 hours ago | Rafal Chmiel | Just a little :trollface:                               |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 4 hours ago | Rafal Chmiel | :trollface: :trollface: :trollface:                     |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 4 hours ago | Rafal Chmiel | Add 'Contributing Guidelines' section :feelsgood:       |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| 4 hours ago | Rafal Chmiel | Search bar -> command bar                               |                                                                                      | <a href="#">commit</a>   <a href="#">commitdiff</a>   <a href="#">tree</a>   <a href="#">snapshot</a> |
| ...         |              |                                                         |                                                                                      |                                                                                                       |

**heads**2 min ago [master](#) | [shortlog](#) | [log](#) | [tree](#)**remotes**

Gitのinstawebコマンドについてもつと詳しく

## Gitの設定

.gitconfigとはあらゆる設定が書き込まれるファイルだ。

## エイリアス

エイリアスはGitの呼び出し方を自分で好きなように定義できるヘルパー機能だ。例えばgit aでgit add --allを実行するようにすることができる。

エイリアスを追加するには~/.gitconfigを開き、以下のような形式で記述していく:

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

またはコマンドラインからも設定できる:

\$ git config --global alias.new\_alias git\_function

例:

\$ git config --global alias.cm commit

注: エイリアスが複数のコマンドからなる場合はワオートで括る必要がある:

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

おすすめの設定を挙げておこう:

| エイリアス        | コマンド                                                                                                                                       | 設定方法                                                                                                                                                                  |                               |                                                           |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------|-----------------------------------------------------------|
| 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 cleanup  | `git branch --merged                                                                                                                       | grep -v `"                                                                                                                                                            | xargs<br>git<br>branch<br>-d` | `git config --global alias.cleanup "!git branch --merged` |
| git cleanup  | `git branch --merged                                                                                                                       | grep -v `"                                                                                                                                                            | xargs<br>git<br>branch<br>-d` | `git config --global alias.cleanup "!git branch --merged` |
| 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 --" |                               |                                                           |

## コマンドの自動修正

多分今は git comit とタイプした場合、以下のような出力を得ることだろう:

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

Did you mean this?
commit
```

これを comit とタイプした時に commit を実行させたい場合、自動修正を有効にすれば良い:

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

すると以下のような出力を得るようになるだろう:

```
$ 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...
```

## 色設定

Gitの出力をカラフルにするには以下のようないい:

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

Gitの config コマンドについてもつと詳しく

## Git情報

| 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 Designers                         | <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>       |
| GitHub Training Kit                       | <a href="http://training.github.com/kit">http://training.github.com/kit</a>                                                   |
| Git Visualization Playground              | <a href="http://onlywei.github.io/explain-git-with-d3/#freeplay">http://onlywei.github.io/explain-git-with-d3/#freeplay</a>   |

## Git Books

| 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 PluralSight           | <a href="https://github.com/pluralsight/git-internals-pdf">https://github.com/pluralsight/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>                                         |

## 訳注

---

これはGitHub Cheat Sheetの日本語訳である。