

The screenshot shows the GitHub repository page for 'tiimgreen/github-cheat-sheet'. The repository name is 'tiimgreen / github-cheat-sheet'. The main navigation bar includes links for Code, Issues (19), Pull requests (16), Actions, Projects, Security, and Insights. Below the navigation is a dropdown menu for the 'master' branch. The main content area displays the 'github-cheat-sheet / README.md' file. A commit by 'tiimgreen' is shown, dated 6 months ago. The file has 1085 lines (752 loc) and is 41.3 KB. There are tabs for Preview, Code, and Blame, along with a Raw button and other file operations.

GitHub Cheat Sheet awesome

A collection of cool hidden and not so hidden features of Git and GitHub. This cheat sheet was inspired by [Zach Holman's Git and GitHub Secrets talk](#) at Aloha Ruby Conference 2012 ([slides](#)) and his [More Git and GitHub Secrets talk](#) at WDCNZ 2013 ([slides](#)).

Shortlink: <http://git.io/sheet>

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

GitHub Cheat Sheet is sponsored by [Snapshot: create interactive professional-quality product photos using AI](#)

Table of Contents

- [GitHub](#)
 - [Ignore Whitespace](#)
 - [Adjust Tab Space](#)
 - [Commit History by Author](#)
 - [Cloning a Repository](#)
 - [Branch](#)
 - [Compare all Branches to Another Branch](#)
 - [Comparing Branches](#)
 - [Compare Branches across Forked Repositories](#)
 - [Gists](#)
 - [Git.io](#)
 - [Keyboard Shortcuts](#)
 - [Line Highlighting in Repositories](#)
 - [Closing Issues via Commit Messages](#)

- [Cross-Link Issues](#)
 - [Locking Conversations](#)
 - [CI Status on Pull Requests](#)
 - [Filters](#)
 - [Syntax Highlighting in Markdown Files](#)
 - [Emojis](#)
 - [Images/GIFs](#)
 - [Embedding Images in GitHub Wiki](#)
 - [Quick Quoting](#)
 - [Pasting Clipboard Image to Comments](#)
 - [Quick Licensing](#)
 - [Task Lists](#)
 - [Task Lists in Markdown Documents](#)
 - [Relative Links](#)
 - [Metadata and Plugin Support for GitHub Pages](#)
 - [Viewing YAML Metadata in your Documents](#)
 - [Rendering Tabular Data](#)
 - [Rendering PDF](#)
 - [Revert a Pull Request](#)
 - [Diffs](#)
 - [Rendered Prose Diffs](#)
 - [Diffable Maps](#)
 - [Expanding Context in Diffs](#)
 - [Diff or Patch of Pull Request](#)
 - [Rendering and diffing images](#)
 - [Hub](#)
 - [Contribution Guidelines](#)
 - [CONTRIBUTING file](#)
 - [ISSUE_TEMPLATE file](#)
 - [PULL_REQUEST_TEMPLATE file](#)
 - [Octicons](#)
 - [GitHub Student Developer Pack](#)
 - [GitHub Resources](#)
 - [GitHub Talks](#)
 - [SSH keys](#)
 - [Profile Image](#)
 - [Repository Templates](#)
- [Git](#)
 - [Remove All Deleted Files from the Working Tree](#)

- [Previous Branch](#)
- [Stripspace](#)
- [Checking out Pull Requests](#)
- [Empty Commits](#)
- [Styled Git Status](#)
- [Styled Git Log](#)
- [Git Query](#)
- [Git Grep](#)
- [Merged Branches](#)
- [Fixup and Autosquash](#)
- [Web Server for Browsing Local Repositories](#)
- [Git Configurations](#)
 - [Aliases](#)
 - [Auto-Correct](#)
 - [Color](#)
- [Git Resources](#)
 - [Git Books](#)
 - [Git Videos](#)
 - [Git Articles](#)

GitHub

Ignore Whitespace

Adding `?w=1` to any diff URL will remove any changes only in whitespace, enabling you to see only the code that has changed.

[Diff without whitespace](#)

[*Read more about GitHub secrets.*](#)

Adjust Tab Space

Adding `?ts=4` to a diff or file URL will display tab characters as 4 spaces wide instead of the default 8. The number after `ts` can be adjusted to suit your preference. This does not work on Gists, or raw file views, but a [Chrome extension](#) can automate this.

Here is a Go source file before adding `?ts=4`:

```

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

```

...and this is after adding `?ts=4`:

```

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

```

Commit History by Author

To view all commits on a repo by author add `?author={user}` to the URL.

<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 [diff] [78a4884] [edit]
 - Stop promoting rack-cache usage at the moment (not so common or impor...
dhh authored 5 days ago [diff] [1302edf] [edit]
 - Show how to change the queuing backend for ActiveJob in production
dhh authored 5 days ago [diff] [6463495] [edit]
 - Set all asset options together
dhh authored 5 days ago [diff] [b9b28d8] [edit]
- Commits on Jan 9, 2015
 - Merge pull request #18413 from brainopia/automatic_inverse_of_for_bel...
dhh authored 9 days ago [diff] [6eb499f] [edit]

[Read more about the differences between commits views.](#)

Cloning a Repository

When cloning a repository the `.git` can be left off the end.

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

[Read more about the Git clone command.](#)

Branch

Compare all Branches to Another Branch

If you go to the repo's [Branches](#) page, next to the Commits button:

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

... you would see a list of all branches which are not merged into the main branch.

From here you can access the compare page or delete a branch with a click of a button.

Branch	Last Updated	Status	Compare
master	Updated an hour ago by fxn	✓	Default
4-2-stable	Updated 2 hours ago by sgrif	✗	Compare
4-1-stable	Updated 4 days ago by rafaelfranca	✓	Compare
fix_nested_transactions_for_re...	Updated 8 days ago by chancancode	✗	Compare
3-2-stable	Updated 12 days ago by rafaelfranca	✗	Compare
4-0-stable	Updated 12 days ago by rafaelfranca	✓	Compare

Comparing Branches

To use GitHub to compare branches, change the URL to look like this:

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

where {range} = master...4-1-stable

For example:

<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} can be changed to things like:

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

Here, dates are in the format YYYY-MM-DD

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

130 commits 123 files changed 5 comments 39 contributors

Commits Files changed Commit comments

Aug 07, 2013

- emre-basala Add tests to ActiveSupport::XmlMini#to_tag method ✓ 05d7cde

Nov 17, 2013

- iantropov Fix insertion of records for hmt association with scope, fix #3548 ✘ ec09280

Jan 07, 2014

- roderickvd Auto-generate stable fixture UUIDs on PostgreSQL. ... ✓ 9330631

Branches can also be compared in diff and patch views:

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

[Read more about comparing commits across time.](#)

Compare Branches across Forked Repositories

To use GitHub to compare branches across forked repositories, change the URL to look like this:

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



For example:

<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

Date	Author	Commit Message	Status
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](#) are an easy way to work with small bits of code without creating a fully fledged repository.

A screenshot of a GitHub Gist page. At the top, there's a header with "GitHub Gist" and a search bar. Below that, it says "PUBLIC" and shows the author "tiimgreen / app.rb" with a creation date of "Created 4 days ago". On the right, there are buttons for "Discover Gists", "rafaelchmiel", and "Star 0" and "Fork 0". The main content area shows a "Gist Detail" sidebar with "Revisions 1" and a "Download Gist" button. The main pane contains an "app.rb" file with one line of code: "puts 'Hello World'". To the right of the file, there's a "Ruby" language indicator and a copy icon. Below the file, a comment from "rafaelchmiel" says "Wow, dat is some engineering." and was posted "2 days ago". There are also "Write" and "Preview" tabs for comments, and a "Leave a comment" input field with "Comments are parsed with GitHub Flavored Markdown". A green "Add Comment" button is at the bottom right.

Add `.pibb` to the end of any Gist URL ([like this](#)) in order to get the *HTML-only* version suitable for embedding in any other site.

Gists can be treated as a repository so they can be cloned like any other:

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



A screenshot of a macOS terminal window titled "tim — bash — 80x17". The window shows the command \$ git clone https://gist.github.com/tiimgreen/10545817 being run. The output of the command is displayed, showing the progress of cloning the gist into a local directory named "10545817". The process includes counting objects, unpacking objects, and checking connectivity, all completed successfully.

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':
```



However, Gists do not support directories. All files need to be added to the repository root. [Read more about creating Gists.](#)

Git.io

[Git.io](#) is a simple URL shortener for GitHub.



You can also use it via pure HTTP using Curl:

```
$ 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/...
```



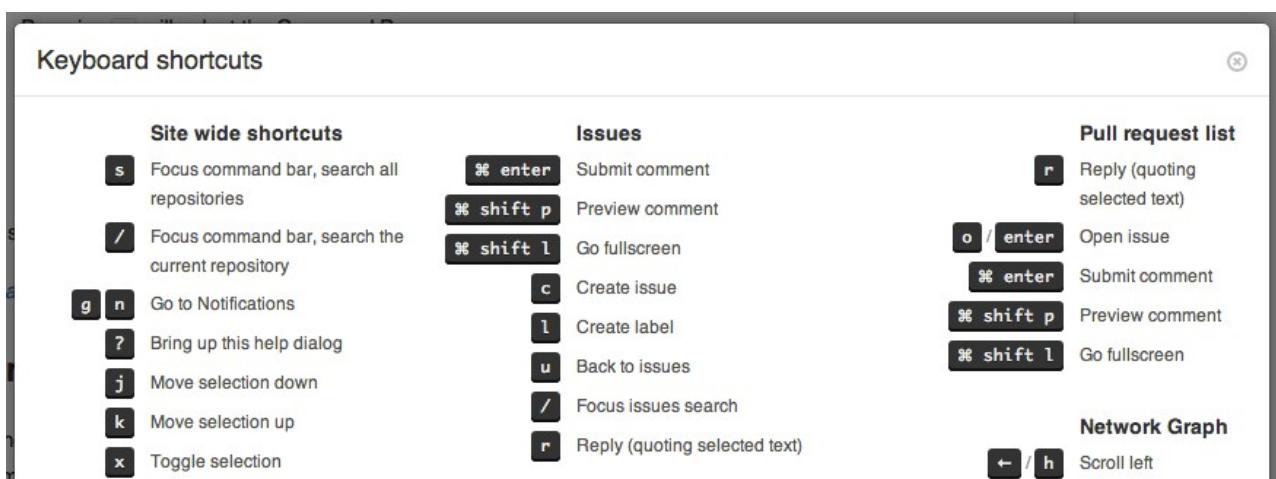
[Read more about Git.io.](#)

Keyboard Shortcuts

When on a repository page, keyboard shortcuts allow you to navigate easily.

- Pressing **t** will bring up a file explorer.
- Pressing **w** will bring up the branch selector.
- Pressing **s** will focus the search field for the current repository. Pressing **↓** to select the "All GitHub" option changes the field to search all of GitHub.
- Pressing **l** will edit labels on existing Issues.
- Pressing **y** when looking at a file (e.g., <https://github.com/tiimgreen/github-cheat-sheet/blob/master/README.md>) will change your URL to one which, in effect, freezes the page you are looking at. If this code changes, you will still be able to see what you saw at that current time.

To see all of the shortcuts for the current page press `:` :



[Read more about search syntax you can use.](#)

Line Highlighting in Repositories

Either adding, e.g., `#L52` to the end of a code file URL or simply clicking the line number will highlight that line number.

It also works with ranges, e.g., `#L53-L60`, to select ranges, hold `shift` and click two lines:

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   ActiveModel::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

```

Closing Issues via Commit Messages

If a particular commit fixes an issue, any of the keywords `fix/fixes/fixed`, `close/closes/closed` or `resolve/resolves/resolved`, followed by the issue number, will close the issue once it is committed to the repository's default branch.

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



This closes the issue and references the closing commit.

The screenshot shows a GitHub commit history. The first commit is from user 'tiimgreen' and has a grey background. It contains the message 'closed this issue from a commit 9 days ago'. Below this commit is a list of files changed, with the file 'Fix screwup, fixes #12' highlighted in blue. To the right of the commit message is the commit hash '36f4317'. The second commit is from user 'tiimgreen' and has a red background. It contains the message 'closed this in 36f4317 9 days ago'.

[Read more about closing Issues via commit messages.](#)

Cross-Link Issues

If you want to link to another issue in the same repository, simply type hash `#` then the issue number, and it will be auto-linked.

To link to an issue in another repository, `{user}/{repo}#ISSUE_NUMBER`, e.g., `tiimgreen/toc#12`.

[Cross-Link Issues](#)

Locking Conversations

Pull Requests and Issues can now be locked by owners or collaborators of the repo.

The screenshot shows a GitHub pull request interface. At the top, there are two buttons: 'Parsed as Markdown' and 'Edit in fullscreen'. To the right of these buttons is a 'Lock issue' button, which features a padlock icon and the text 'Lock issue'.

This means that users who are not collaborators on the project will no longer be able to comment.

The screenshot shows a GitHub conversation interface. On the left, there is a profile picture of a user and two tabs: 'Write' and 'Preview'. The main area contains a large text input field with a lock icon in its center. Below the input field, the text 'This conversation has been locked and limited to collaborators.' is displayed.

[Read more about locking conversations.](#)

CI Status on Pull Requests

If set up correctly, every time you receive a Pull Request, [Travis CI](#) will build that Pull Request just like it would every time you make a new commit. Read more about how to [get started with Travis CI](#).

The screenshot shows a GitHub pull request page for the octokit/octokit.rb repository. The pull request is titled "Lazy create test repos #452" and is from user joeyw. It has 2 commits and 1 file changed. The pull request has passed CI, as indicated by the green status bar at the bottom. The right sidebar shows labels (None yet), milestones (No milestone), assignees (No one assigned), and notifications (Subscribe). A comment from joeyw is visible, mentioning alternatives to mass repo creation and noting that it checks for and creates test repositories before recording new cassettes.

[Read more about the commit status API.](#)

Filters

Both issues and pull requests allow filtering in the user interface.

For the Rails repo: <https://github.com/rails/rails/issues>, the following filter is built by selecting the label "activerecord":

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

But, you can also find all issues that are NOT labeled activerecord:

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

Additionally, this also works for pull requests:

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

Github has tabs for displaying open or closed issues and pull requests but you can also see merged pull requests. Just put the following in the filter:

```
is:merged
```

[Read more about searching issues.](#)

Finally, github now allows you to filter by the Status API's status.

Pull requests with only successful statuses:

`status:success`

[Read more about searching on the Status API.](#)

Syntax Highlighting in Markdown Files

For example, to syntax highlight Ruby code in your Markdown files write:

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

```



This will produce:

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

```



GitHub uses [Linguist](#) to perform language detection and syntax highlighting. You can find out which keywords are valid by perusing the [languages YAML file](#).

[Read more about GitHub Flavored Markdown.](#)

Emojis

Emojis can be added to Pull Requests, Issues, commit messages, repository descriptions, etc. using `:name_of_emoji:` .

The full list of supported Emojis on GitHub can be found at [emoji-cheat-sheet.com](#) or [scotch-io/All-Github-Emoji-Icons](#). A handy emoji search engine can be found at [emoji.muan.co](#).

The top 5 used Emojis on GitHub are:

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

Images/GIFs

Images and GIFs can be added to comments, READMEs etc.:

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



Raw images from the repo can be used by calling them directly.:

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



All images are cached on GitHub, so if your host goes down, the image will remain available.

Embedding Images in GitHub Wiki

There are multiple ways of embedding images in Wiki pages. There's the standard Markdown syntax (shown above). But there's also a syntax that allows things like specifying the height or width of the image:

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



Which produces:

The screenshot shows a GitHub Wiki preview page. At the top, there are navigation links for 'Home', 'Pages', and 'History', and a 'New Page' button. On the right side, there is a sidebar with various icons for file operations. The main content area displays the following text and image:

Home (Preview)

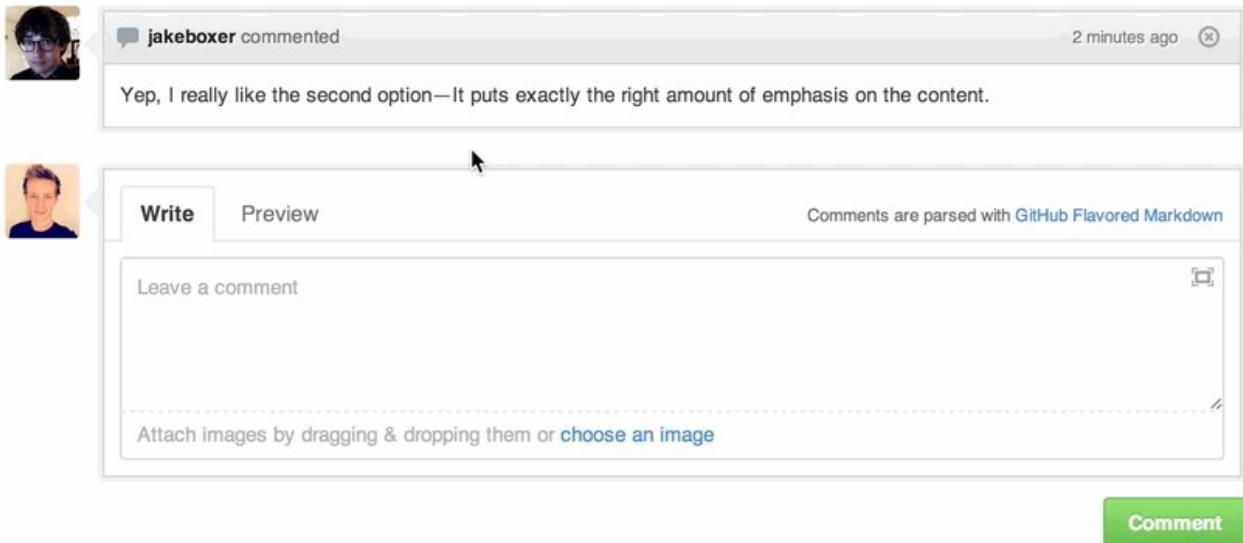


README not found

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.

Quick Quoting

When on a comment thread and you want to quote something someone previously said, highlight the text and press `r`, this will copy it into your text box in the block-quote format.



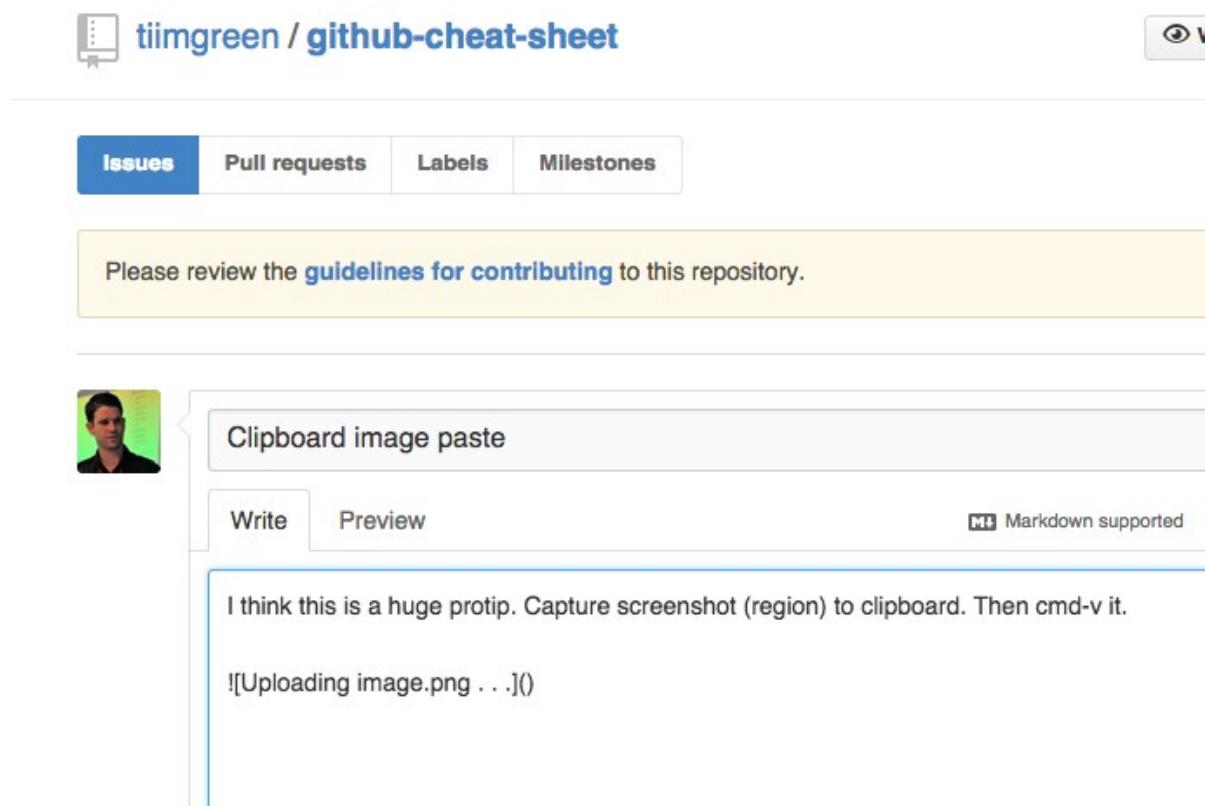
A screenshot of a GitHub comment section. At the top, a comment from user **jakeboxer** is shown, posted 2 minutes ago. The comment text is: "Yep, I really like the second option—It puts exactly the right amount of emphasis on the content." Below this is a larger comment input area. The tab "Write" is selected. The text area contains the placeholder "Leave a comment". Below the text area is a dashed line with the instruction "Attach images by dragging & dropping them or choose an image". A green "Comment" button is located at the bottom right of the input area.

[Read more about quick quoting.](#)

Pasting Clipboard Image to Comments

(Works on Chrome browsers only)

After taking a screenshot and adding it to the clipboard (mac: `cmd-ctrl-shift-4`), you can simply paste (`cmd-v` / `ctrl-v`) the image into the comment section and it will be auto-uploaded to github.



A screenshot of a GitHub repository page for **tiimgreen / github-cheat-sheet**. The "Issues" tab is selected. A yellow banner at the top says: "Please review the [guidelines for contributing](#) to this repository." In the comments section, a comment from user **tiimgreen** is shown with the subject "Clipboard image paste". The comment text is: "I think this is a huge protip. Capture screenshot (region) to clipboard. Then cmd-v it." Below the text is the image content: "![Uploading image.png . . .]()." The "Preview" tab is selected in the comment input area, and a note says "Markdown supported".

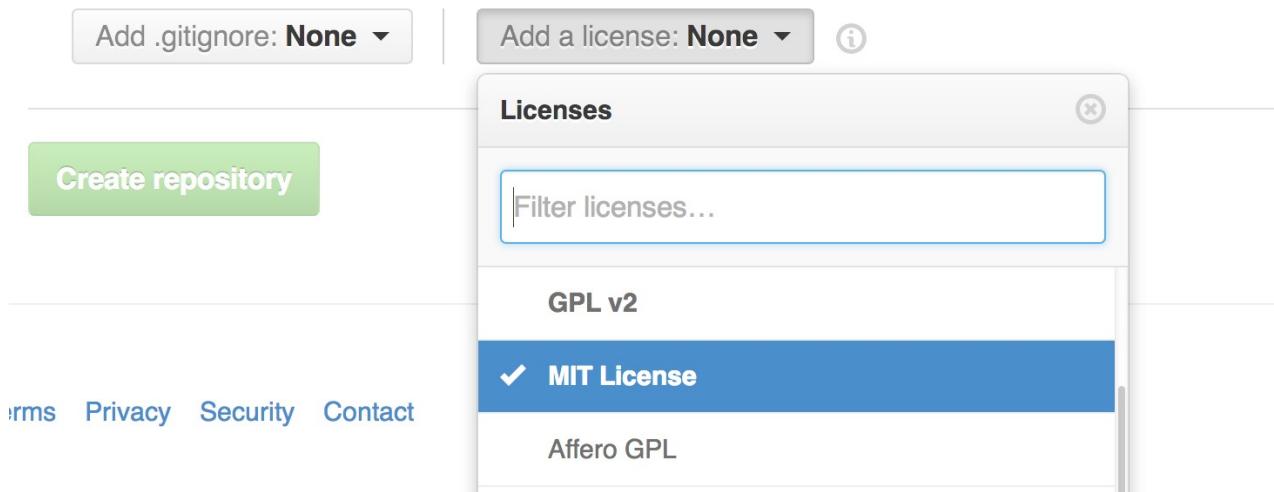
[Read more about issue attachments.](#)

Quick Licensing

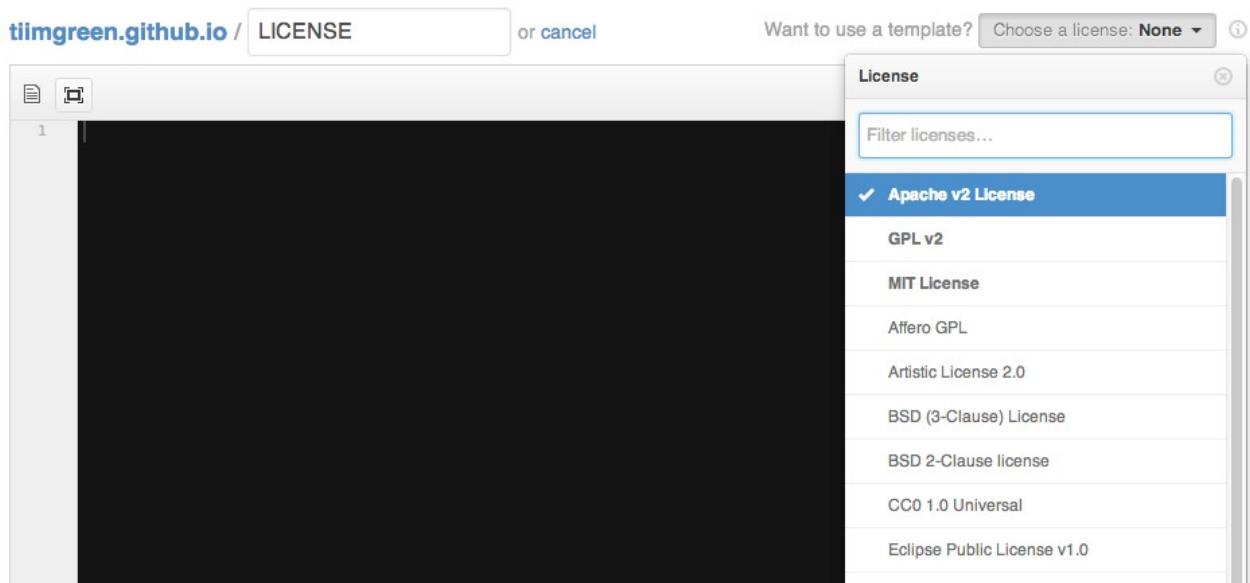
When creating a repository, GitHub gives you the option of adding in a pre-made license:

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`



You can also add them to existing repositories by creating a new file through the web interface. When the name `LICENSE` is typed in you will get an option to use a template:



Also works for `.gitignore`.

[Read more about open source licensing.](#)

Task Lists

In Issues and Pull requests check boxes can be added with the following syntax (notice the space):

- [] 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

When they are clicked, they will be updated in the pure Markdown:

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



[Read more about task lists.](#)

Task Lists in Markdown Documents

In full Markdown documents **read-only** checklists can now be added using the following syntax:

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



- Mercury
- Venus
- Earth

- Moon
- Mars
 - Deimos
 - Phobos

[Read more about task lists in markdown documents.](#)

Relative Links

Relative links are recommended in your Markdown files when linking to internal content.

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



Absolute links have to be updated whenever the URL changes (e.g., repository renamed, username changed, project forked). Using relative links makes your documentation easily stand on its own.

[Read more about relative links.](#)

Metadata and Plugin Support for GitHub Pages

Within Jekyll pages and posts, repository information is available within the `site.github` namespace, and can be displayed, for example, using `{{ site.github.project_title }}`.

The Jemoji and jekyll-mentions plugins enable [emoji](#) and [@mentions](#) in your Jekyll posts and pages to work just like you'd expect when interacting with a repository on GitHub.com.

[Read more about repository metadata and plugin support for GitHub Pages.](#)

Viewing YAML Metadata in your Documents

Many blogging websites, like [Jekyll](#) with [GitHub Pages](#), depend on some YAML-formatted metadata at the beginning of your post. GitHub will render this metadata as a horizontal table, for easier reading

[YAML metadata](#)

[Read more about viewing YAML metadata in your documents.](#)

Rendering Tabular Data

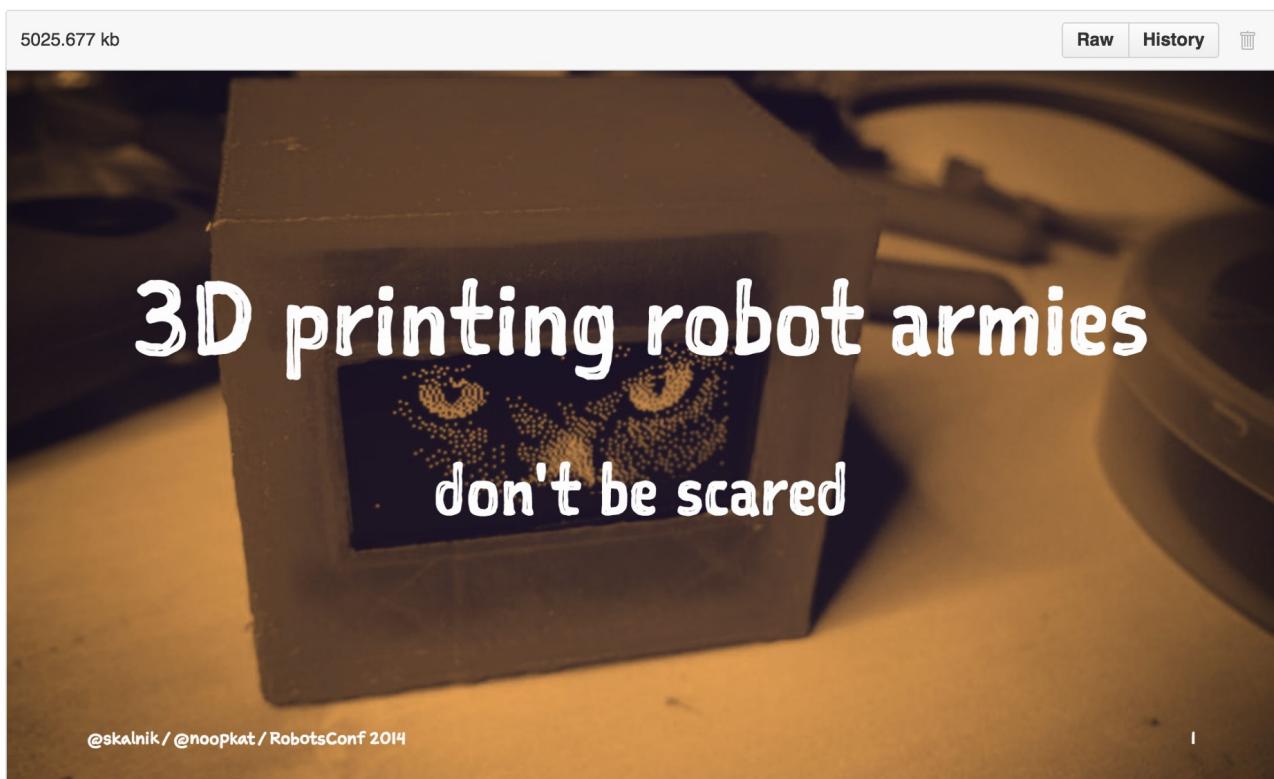
GitHub supports rendering tabular data in the form of `.csv` (comma-separated) and `.tsv` (tab-separated) files.

[Tabular data](#)

[Read more about rendering tabular data.](#)

Rendering PDF

GitHub supports rendering PDF:



[Read more about rendering PDF.](#)

Revert a Pull Request

After a pull request is merged, you may find it does not help anything or it was a bad decision to merge the pull request.

You can revert it by clicking the **Revert** button on the right side of a commit in the pull request page to create a pull request with reverted changes to this specific pull request.

[Revert button](#)

[Read more about reverting pull requests](#)

Diffs

Rendered Prose Diffs

Commits and pull requests, including rendered documents supported by GitHub (e.g., Markdown), feature *source* and *rendered* views.

[Source / Rendered view](#)

Click the "rendered" button to see the changes as they'll appear in the rendered document. Rendered prose view is handy when you're adding, removing, and editing text:

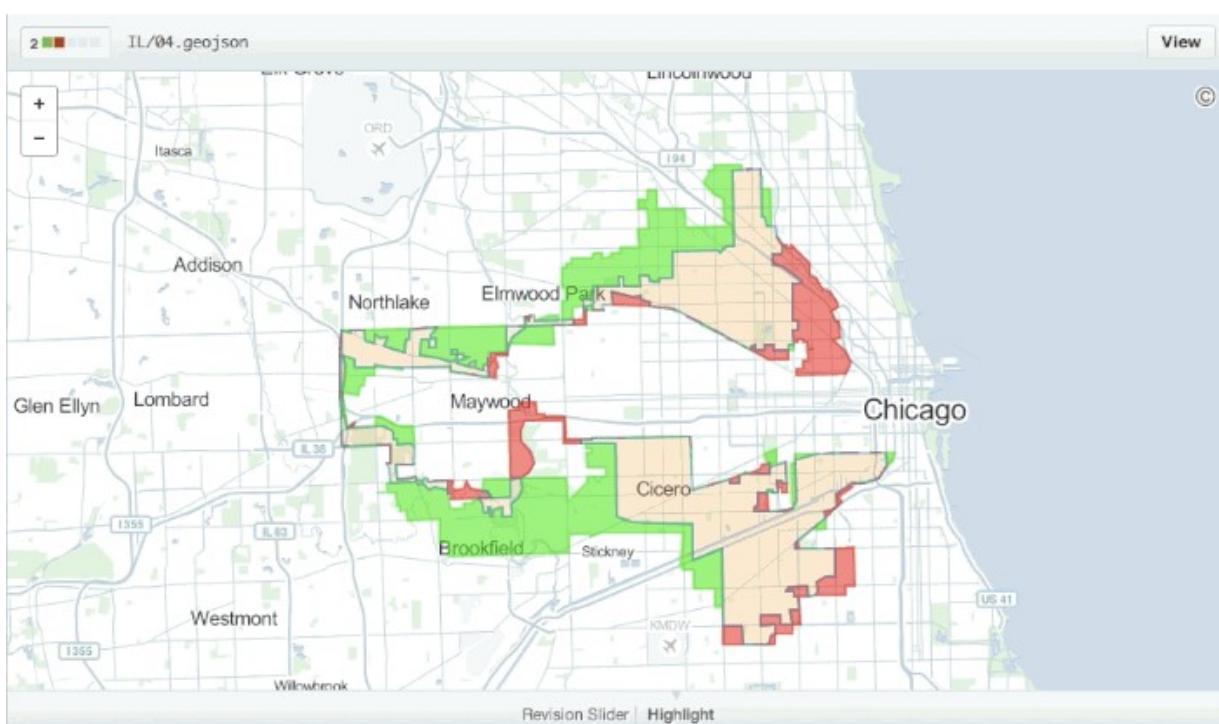
Obviously, you'll want to change `<your_client_id>` to match your actual Client ID.

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.

[Read more about rendered prose diffs.](#)

Diffable Maps

Any time you view a commit or pull request on GitHub that includes geodata, GitHub will render a visual representation of what was changed.



[Read more about diffable maps.](#)

Expanding Context in Diffs

Using the `unfold` button in the gutter of a diff, you can reveal additional lines of context with a click. You can keep clicking `unfold` until you've revealed the whole file, and the feature is available anywhere GitHub renders diffs.

```

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 //
85  @@ -241,11 +255,13 @@
86      NSURLConnection *userAgent = self.userAgent;
87      if (userAgent != nil) [self setDefaultHeader:@"User-Agent" value:userAgent];
88
89  207
90      - self.parameterEncoding = AFJSONParameterEncoding;
91      - [self setDefaultHeader:@"Accept" value:@"application/vnd.github.beta+json"];
92      -
93
94  258      [AFHTTPRequestOperation addAcceptableStatusCodes:[NSIndexSet indexSetWithIndex:OCTClientNotModifiedStatusCode]];
95
96  259      +
97      + NSString *contentType = [NSString stringWithFormat:@"application/vnd.github.%@+json", OCTClientAPIVersion];
98      + [self setDefaultHeader:@"Accept" value:contentType];
99      + [AFHTTPRequestOperation addAcceptableContentTypes:[NSSet setWithObject:contentType]];
100
101      +
102      + self.parameterEncoding = AFJSONParameterEncoding;
103
104  265      [self registerHTTPOperationClass:AFJSONRequestOperation.class];
105
106  266
107  267      return self;

```

[Read more about expanding context in diffs.](#)

Diff or Patch of Pull Request

You can get the diff of a Pull Request by adding a `.diff` or `.patch` extension to the end of the URL.
For example:

<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>

The `.diff` extension would give you this in plain text:

```

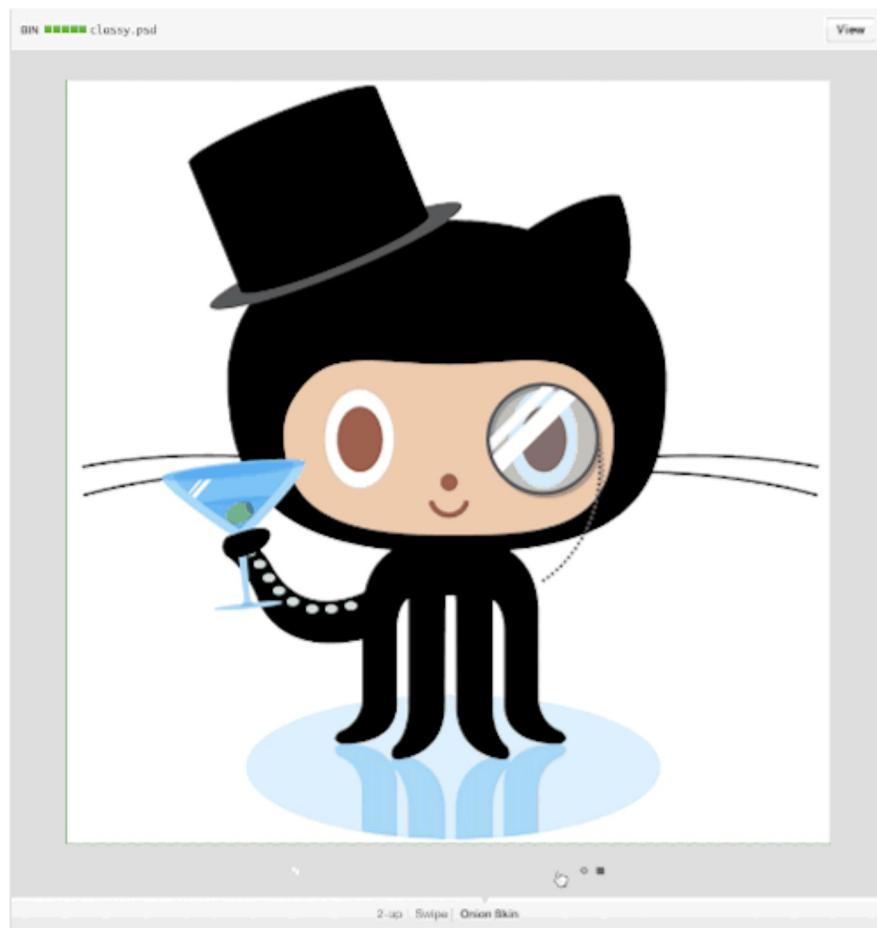
diff --git a/README.md b/README.md
index 88fcf69..8614873 100644
--- a/README.md
+++ b/README.md
@@ -28,6 +28,7 @@ All the hidden and not hidden features of Git and GitHub. This cheat
sheet was i
- [Merged Branches](#merged-branches)
- [Quick Licensing](#quick-licensing)
- [TODO Lists](#todo-lists)
+- [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

```

(...)

Rendering and diffing images

GitHub can display several common image formats, including PNG, JPG, GIF, and PSD. In addition, there are several ways to compare differences between versions of those image formats.



[Read more about rendering and diffing images.](#)

Hub

[Hub](#) is a command line Git wrapper that gives you extra features and commands that make working with GitHub easier.

This allows you to do things like:

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



[Check out some more cool commands Hub has to offer.](#)

Contribution Guidelines

GitHub supports adding 3 different files which help users contribute to your project. These files can either be placed in the root of your repository or a `.github` directory under the root.

CONTRIBUTING File

Adding a `CONTRIBUTING` or `CONTRIBUTING.md` file to either the root of your repository or a `.github` directory will add a link to your file when a contributor creates an Issue or opens a Pull Request.

[Contributing Guidelines](#)

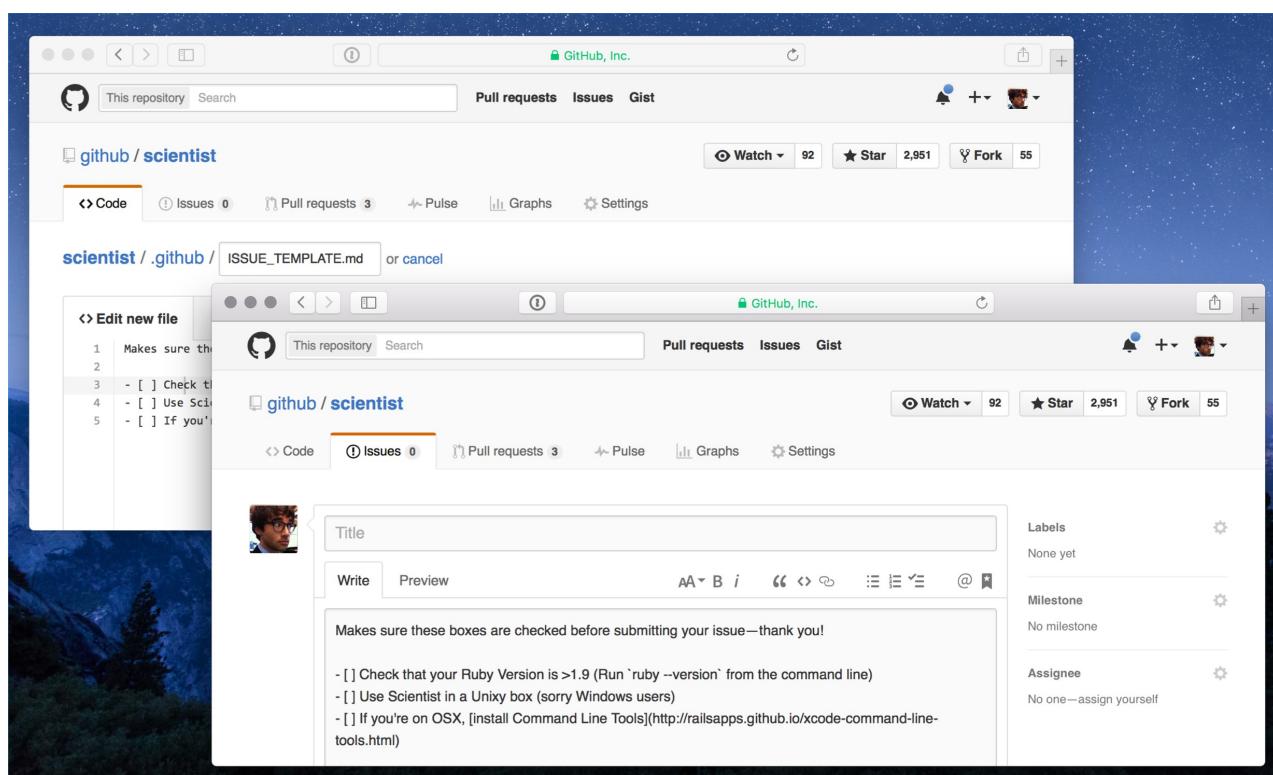
[Read more about contributing guidelines.](#)

ISSUE_TEMPLATE file

You can define a template for all new issues opened in your project. The content of this file will pre-populate the new issue box when users create new issues. Add an `ISSUE_TEMPLATE` or `ISSUE_TEMPLATE.md` file to either the root of your repository or a `.github` directory.

[Read more about issue templates.](#)

[Issue template file generator](#)



PULL_REQUEST_TEMPLATE file

You can define a template for all new pull requests opened in your project. The content of this file will pre-populate the text area when users create pull requests. Add a `PULL_REQUEST_TEMPLATE` or `PULL_REQUEST_TEMPLATE.md` file to either the root of your repository or a `.github` directory.

[Read more about pull request templates.](#)

[Pull request template file generator](#)

Octicons

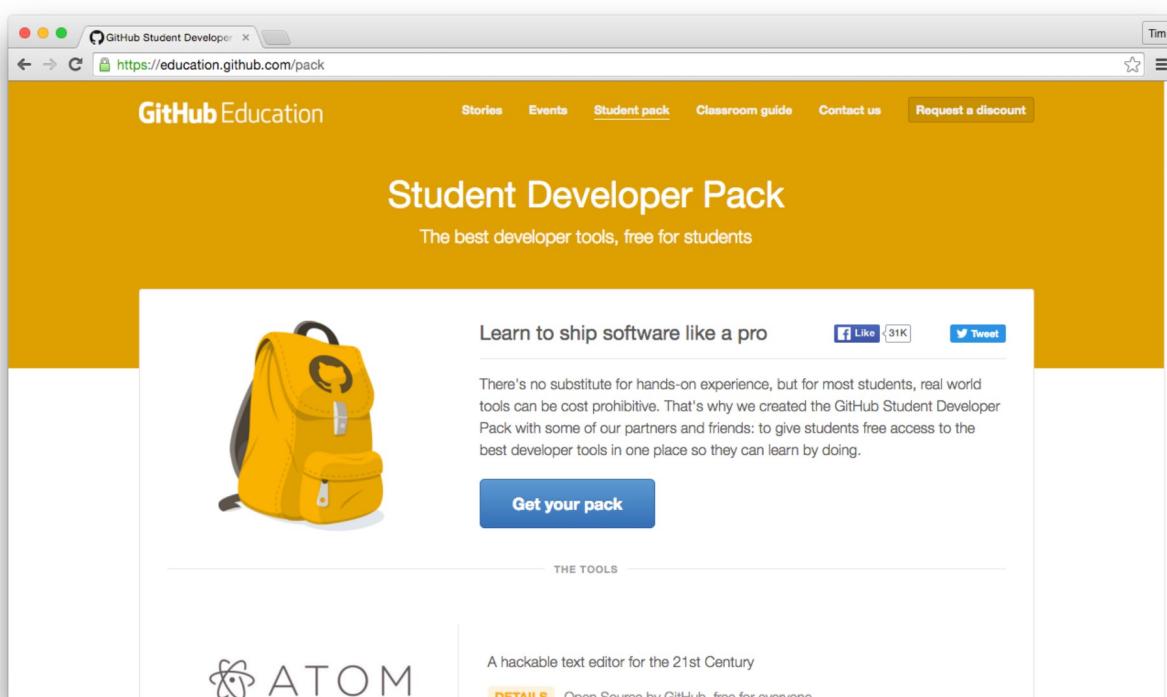
GitHubs icons (Octicons) have now been open sourced.

[Octicons](#)

[Read more about GitHub's Octicons](#)

GitHub Student Developer Pack

If you are a student you will be eligible for the GitHub Student Developer Pack. This gives you free credit, free trials and early access to software that will help you when developing.



[Read more about GitHub's Student Developer Pack](#)

GitHub Resources

| Title | Link |
|--|---|
| GitHub Explore | https://github.com/explore |
| GitHub Blog | https://github.com/blog |
| GitHub Help | https://help.github.com/ |
| GitHub Training | https://training.github.com/ |
| GitHub Developer | https://developer.github.com/ |
| Github Education (Free Micro Account and other stuff for students) | https://education.github.com/ |

| Title | Link |
|-----------------------|-------------------------------------|
| GitHub Best Practices | Best Practices List |

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 |

SSH keys

You can get a list of public ssh keys in plain text format by visiting:

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



e.g. <https://github.com/tiimgreen.keys>

[*Read more about accessing public ssh keys.*](#)

Profile Image

You can get a user's profile image by visiting:

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



e.g. <https://github.com/tiimgreen.png>

Repository Templates

You can enable templating on your repository which allows anyone to copy the directory structure and files, allowing them to instantly use the files (e.g. for a tutorial or if writing boilerplate code). This can be enabled in the settings of your repository.

The screenshot shows the 'Settings' page for the repository 'Machine-Learning-Bootcamp'. On the left, there's a sidebar with options like 'Code', 'Issues 0', 'Pull requests 0', 'Projects 0', 'Wiki', 'Security', 'Insights', and 'Settings'. The 'Settings' tab is active. In the main area, under 'Repository name', it says 'Machine-Learning-Bootcamp' with a 'Rename' button. Below that is a section for 'Template repository' with a checked checkbox and a note: 'Template repositories let users generate new repositories with the same directory structure and files. Indicate if SKhan97/Machine-Learning-Bootcamp can be used as a template for creating other repositories.' There's also a 'Social preview' link.

Changing to a template repository will give a new URL endpoint which can be shared and instantly allows users to use your repository as a template. Alternatively, they can go to your repository and click the 'Use as template' button.

The screenshot shows the main repository page for 'Machine-Learning-Bootcamp'. At the top, it says 'Template' next to the repository name. The page includes sections for 'Code', 'Issues 0', 'Pull requests 0', 'Projects 0', 'Wiki', 'Security', 'Insights', and 'Settings'. Below that is a description: 'Collection of data and analysis using Machine Learning. All code written in Python 3 using the Google Colaboratory cloud service.' A 'Manage topics' button is nearby. The repository stats are: 26 commits, 1 branch, 0 releases, 1 contributor, and MIT license. A 'Use this template' button is highlighted in green. The commit history table lists several files added, including 'Data Analysis', 'Linear Regression', 'Logistic Regression', 'Neural Networks', 'LICENSE', 'README.md', and 'Universal concepts, preamble & glossary.ipynb', all committed by 'SKhan97'.

[Read more about using repositories as templates](#)

Git

Remove All Deleted Files from the Working Tree

When you delete a lot of files using `/bin/rm` you can use the following command to remove them from the working tree and from the index, eliminating the need to remove each one individually:

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



For example:

```
$ 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
```



Previous Branch

To move to the previous branch in Git:

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

$ git checkout -
# Switched to branch 'next'

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



[Read more about Git branching.](#)

Stripspace

Git StripSpace:

- Strips trailing whitespace
- Collapses newlines
- Adds newline to end of file

A file must be passed when calling the command, e.g.:

```
$ git strip-space < README.md
```



[Read more about the Git strip-space command.](#)

Checking out Pull Requests

Pull Requests are special branches on the GitHub repository which can be retrieved locally in several ways:

Retrieve a specific Pull Request and store it temporarily in `FETCH_HEAD` for quickly `diff`-ing or `merge`-ing:

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



Acquire all Pull Request branches as local remote branches by refspec:

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



Or setup the remote to fetch Pull Requests automatically by adding these corresponding lines in your repository's `.git/config`:

```
[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/*
```



For Fork-based Pull Request contributions, it's useful to `checkout` a remote branch representing the Pull Request and create a local branch from it:

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



Or should you work on more repositories, you can globally configure fetching pull requests in the global `git config` instead.

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



This way, you can use the following short commands in all your repositories:

```
git fetch origin
```



```
git checkout pr/42
```



[Read more about checking out pull requests locally.](#)

Empty Commits

Commits can be pushed with no code changes by adding `--allow-empty`:

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



Some use-cases for this (that make sense), include:

- Annotating the start of a new bulk of work or a new feature.
- Documenting when you make changes to the project that aren't code related.
- Communicating with people using your repository.
- The first commit of a repository: `git commit -m "Initial commit" --allow-empty`.

Styled Git Status

Running:

```
$ git status
```



produces:

```
Tims-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")
Tims-MacBook-Pro:ghcs tim$
```

By adding `-sb`:

```
$ git status -sb
```



this is produced:

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

[Read more about the Git status command.](#)

Styled Git Log

Running:

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

produces:

```
Tim-MacBook-Pro:ghcs tim$ git log --all --graph --pretty=format:'%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr)' --abbrev-commit --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>
* fba6f99 - 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 6f5cfae (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>
| /
| * 6f5cfae - 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>
| \
| * 1eadfd88 - Update README.md (6 months ago) <ChelseaStats>
```

Credit to [Palesz](#)

This can be aliased using the instructions found [here](#).

[Read more about the Git Log command.](#)

Git Query

A Git query allows you to search all your previous commit messages and find the most recent one matching the query.

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



where `query` (case-sensitive) is the term you want to search, this then finds the last one and gives details on the lines that were changed.

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



```
Tims-MacBook-Pro:ghcs tims$ 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)
```

Press `q` to quit.

## Git Grep

Git Grep will return a list of lines matching a pattern.

Running:

```
$ git grep aliases
```



will show all the files containing the string `aliases`.

```

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: 0| 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*.
README.md:[git grep aliases](http://i.imgur.com/q3WkUgl.png?1)
README.md:Some useful aliases include:
README.zh-cn.md: - [Aliases](#aliases)
README.zh-tw.md: - [Aliases](#aliases)
Tims-MacBook-Pro:ghcs tim$

```

*Press `q` to quit.*

You can also use multiple flags for more advanced search. For example:

- `-e` The next parameter is the pattern (e.g., regex)
- `--and`, `--or` and `--not` Combine multiple patterns.

Use it like this:

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



[Read more about the Git grep command.](#)

## Merged Branches

Running:

\$ git branch --merged



will give you a list of all branches that have been merged into your current branch.

Conversely:

\$ git branch --no-merged



will give you a list of branches that have not been merged into your current branch.

[Read more about the Git branch command.](#)

## Fixup and Autosquash

If there is something wrong with a previous commit (can be one or more from HEAD), for example `abcde`, run the following command after you've amended the problem:

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



[Read more about the Git commit command.](#) [Read more about the Git rebase command.](#)

## Web Server for Browsing Local Repositories

Use the `git instaweb` command to instantly browse your working repository in `gitweb`. This command is a simple script to set up `gitweb` and a web server for browsing the local repository.

```
$ git instaweb
```



opens:

The screenshot shows a web-based interface for a Git repository. At the top, there's a header with 'projects / .git / summary'. Below it is a navigation bar with links for 'summary', 'shortlog', 'log', 'commit', 'committdiff', and 'tree'. There are also search and refresh buttons. The main content area has several sections:

- description:** Unnamed repository; edit this file 'description' to name the repository.
- owner:** Rafał Chmiel
- last change:** Mon, 14 Apr 2014 20:04:18 +0100 (20:04 +0100)
- shortlog:** A detailed list of commits, each with author, date, message, and a link to the commit details. Some messages include emojis like :star:, :trollface:, and :feelsgood:.
- heads:** Shows the 'master' branch with its last commit information.
- remotes:** Shows no remote branches.

[Read more about the Git instaweb command.](#)

## Git Configurations

Your `.gitconfig` file contains all your Git configurations.

### Aliases

Aliases are helpers that let you define your own git calls. For example you could set `git a` to run `git add --all`.

To add an alias, either navigate to `~/.gitconfig` and fill it out in the following format:

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

...or type in the command-line:

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

For example:

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

For an alias with multiple functions use quotes:

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

Some useful aliases include:

Alias	Command	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 cleanup	git branch --merged   grep -v '*'   xargs git branch -d	git config --global alias.cleanup "!git branch --merged   grep -v '*'   xargs git branch -d"
git remotes	git remote -v	git config --global alias.remotes 'remote -v'

Alias	Command	What to Type
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 --"

Some Aliases are taken from [@mathiasbynens](https://github.com/mathiasbynens/dotfiles/blob/master/.gitconfig) dotfiles: <https://github.com/mathiasbynens/dotfiles/blob/master/.gitconfig>

## Auto-Correct

Git gives suggestions for misspelled commands and if auto-correct is enabled the command can be fixed and executed automatically. Auto-correct is enabled by specifying an integer which is the delay in tenths of a second before git will run the corrected command. Zero is the default value where no correcting will take place, and a negative value will run the corrected command with no delay.

For example, if you type `git comit` you will get this:

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

Did you mean this?
commit
```



Auto-correct can be enabled like this (with a 1.5 second delay):

```
$ git config --global help.autocorrect 15
```



So now the command `git comit` will be auto-corrected to `git commit` like this:

```
$ 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 1.5 seconds automatically...
```



The delay before git will rerun the command is so the user has time to abort.

## Color

To add more color to your Git output:

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



[Read more about the `Git config` command.](#)

## Git Resources

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>
Git God	<a href="https://github.com/gorosgobe/git-god">https://github.com/gorosgobe/git-god</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 Visualization Playground	<a href="http://onlywei.github.io/explain-git-with-d3/#freestyle">http://onlywei.github.io/explain-git-with-d3/#freestyle</a>
Learn Git Branching	<a href="http://pcottle.github.io/learnGitBranching/">http://pcottle.github.io/learnGitBranching/</a>
A collection of useful <code>.gitignore</code> templates	<a href="https://github.com/github/gitignore">https://github.com/github/gitignore</a>
Unixorn's git-extra-commands collection of git scripts	<a href="https://github.com/unixorn/git-extra-commands">https://github.com/unixorn/git-extra-commands</a>

## Git Books

Title	Link
Pragmatic Version Control Using Git	<a href="https://pragprog.com/titles/tsgit/pragmatic-version-control-using-git">https://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.io/gitt/">http://cbx33.github.io/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="https://pragprog.com/titles/pg_git/pragmatic-guide-to-git">https://pragprog.com/titles/pg_git/pragmatic-guide-to-git</a>

Title	Link
Git: Version Control for Everyone	<a href="https://www.packtpub.com/application-development/git-version-control-everyone">https://www.packtpub.com/application-development/git-version-control-everyone</a>

## Git Videos

Title	Link
Linus Torvalds on Git	<a href="https://www.youtube.com/watch?v=4XpnKHJAok8">https://www.youtube.com/watch?v=4XpnKHJAok8</a>
Introduction to Git with Scott Chacon	<a href="https://www.youtube.com/watch?v=ZDR433b0HJY">https://www.youtube.com/watch?v=ZDR433b0HJY</a>
Git From the Bits Up	<a href="https://www.youtube.com/watch?v=MYP56QJpDr4">https://www.youtube.com/watch?v=MYP56QJpDr4</a>
Graphs, Hashes, and Compression, Oh My!	<a href="https://www.youtube.com/watch?v=ig5E8CcdM9g">https://www.youtube.com/watch?v=ig5E8CcdM9g</a>
GitHub Training & Guides	<a href="https://www.youtube.com/watch?list=PLg7s6cbtAD15G8INyoaYDuKZSKyJrgwB-&amp;v=FyfwLX4HAxM">https://www.youtube.com/watch?list=PLg7s6cbtAD15G8INyoaYDuKZSKyJrgwB-&amp;v=FyfwLX4HAxM</a>

## Git Articles

Title	Link
GitHub Flow	<a href="http://scottchacon.com/2011/08/31/github-flow.html">http://scottchacon.com/2011/08/31/github-flow.html</a>
Migrating to Git Large File Storage (Git LFS)	<a href="http://vooban.com/en/tips-articles-geek-stuff/migrating-to-git-lfs-for-developing-deep-learning-applications-with-large-files/">http://vooban.com/en/tips-articles-geek-stuff/migrating-to-git-lfs-for-developing-deep-learning-applications-with-large-files/</a>