Introduction to GitHub

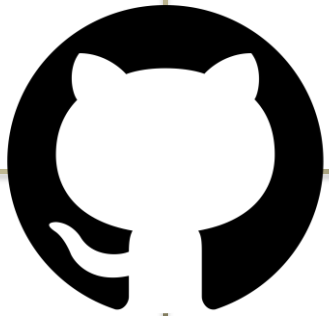


Image: https://github.com

Sebastian Schwindt
sschwindt@ucdavis.edu | https://sebastian-schwindt.org

Kenneth Larrieu
kglarrieu@ucdavis.edu

Prof. Gregory B. Pasternack lab – http://pasternack.ucdavis.edu

Department of Land, Air, and Water Resources
239 Veihmeyer Hall, University of California, Davis
Davis, CA 95616-8628

Davis, CA | August 28, 2019

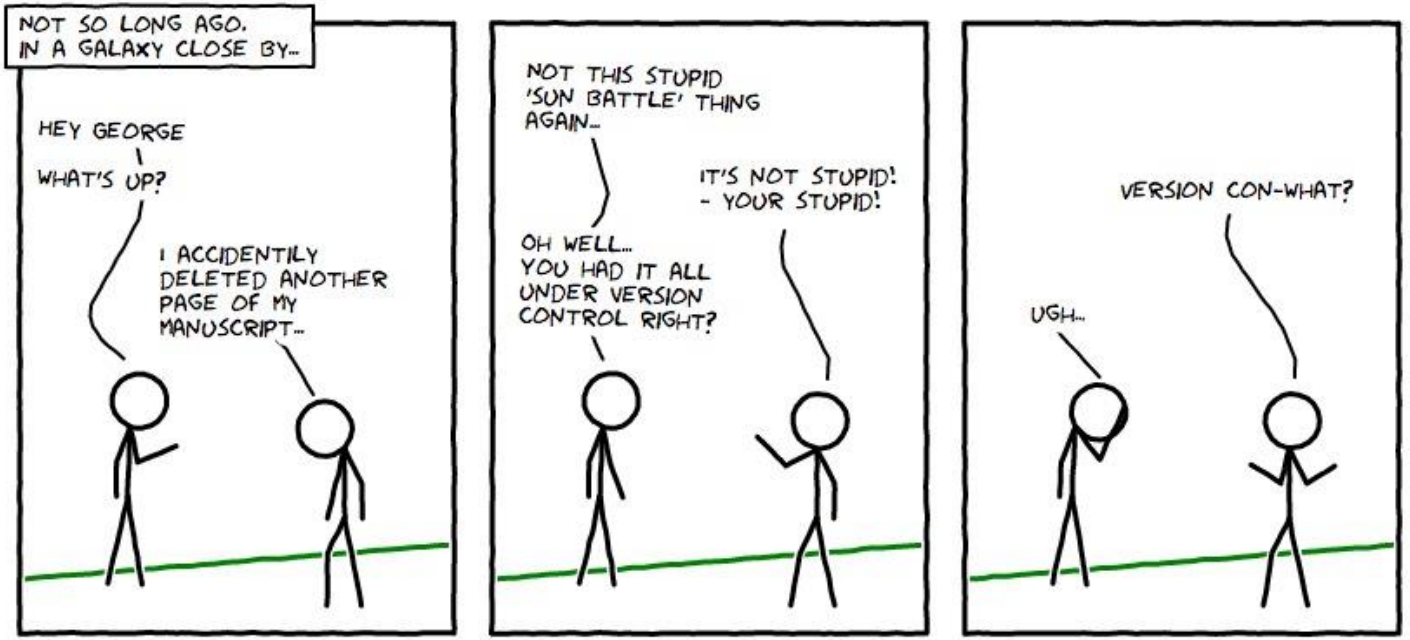
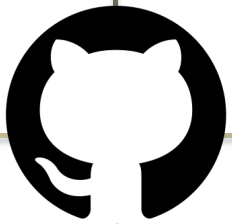
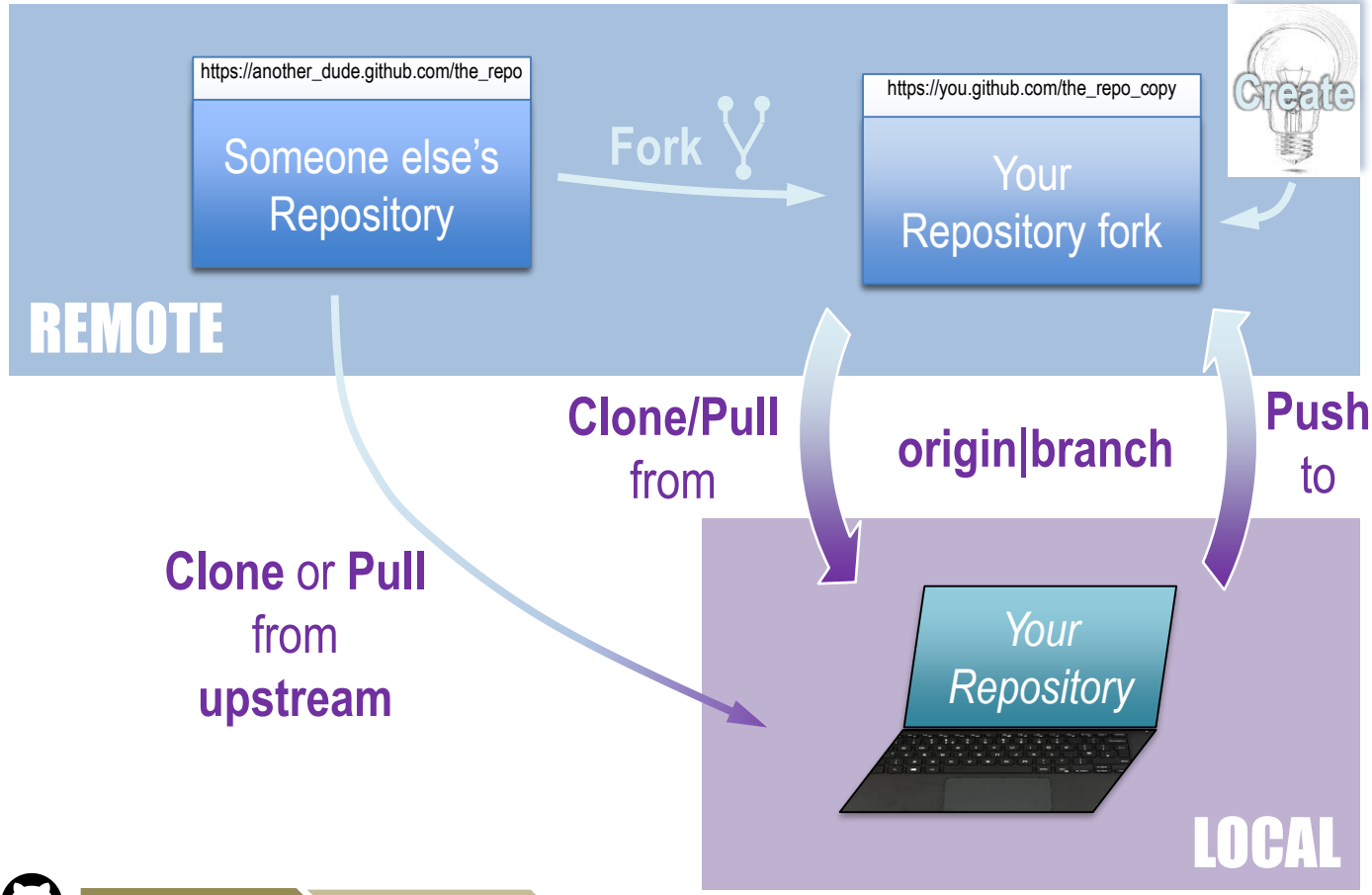


Table of contents

Creating, managing, and updating repositories



How it works & git vocabulary



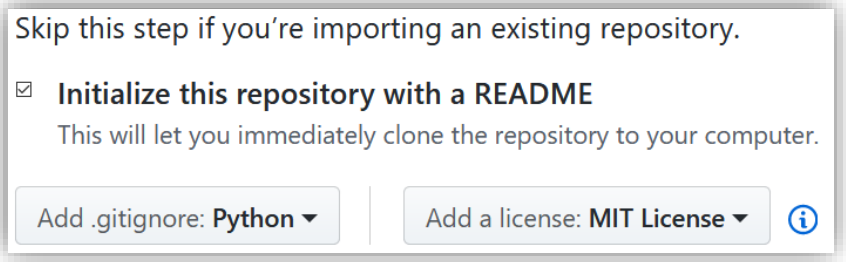
Create a remote Repository

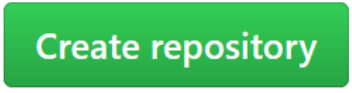
Option 1: Fork an existing repository.

Option 2: **Create a new repository**.

1. **Go to https://github.com and sign in.**
2. **In the upper left corner click on Repositories** 

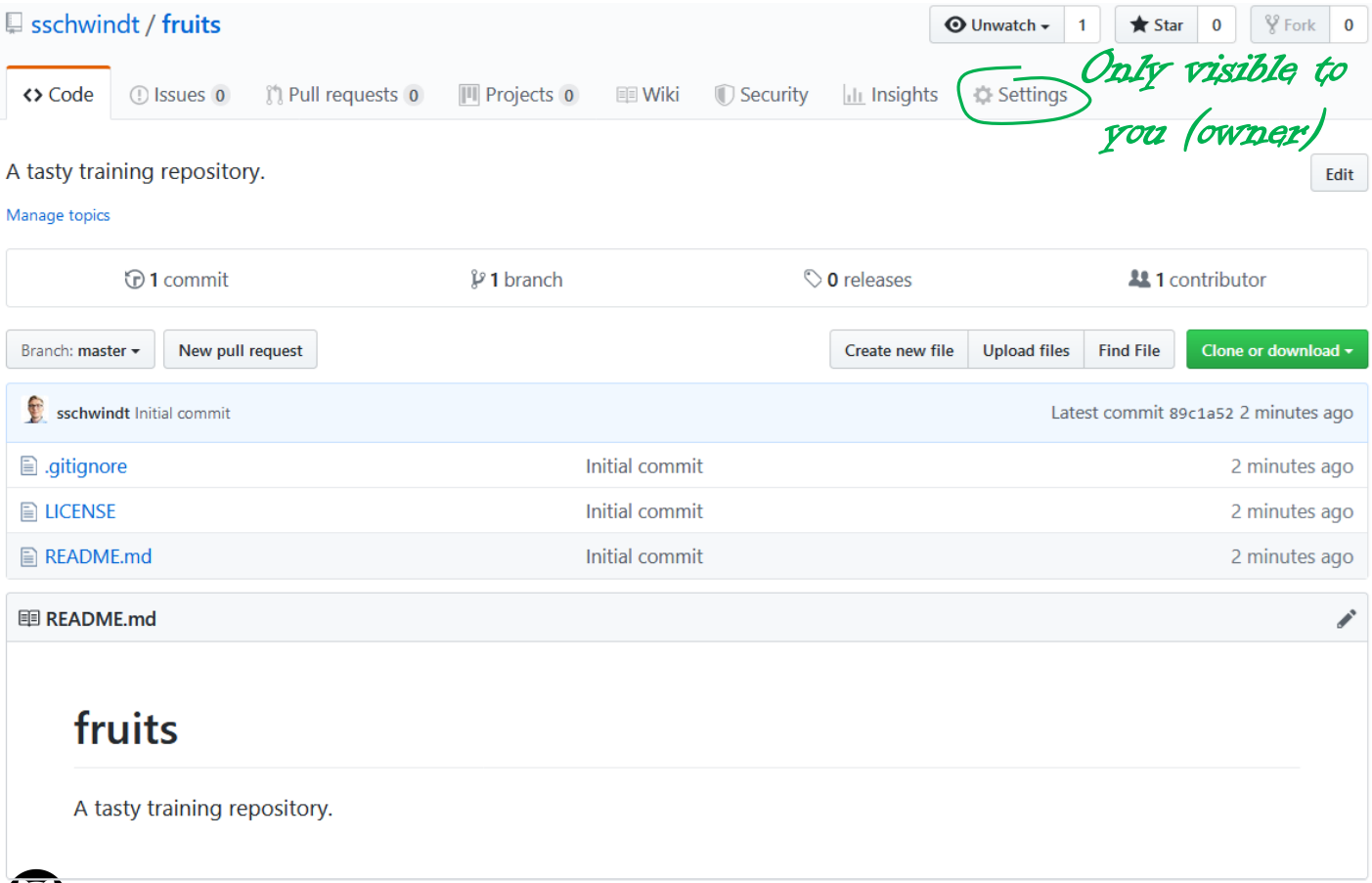
.
3. **Name your** *public* **repository (e.g.,** *juices*), **add a description, check** *Initialize … with a README*, **select** *Add .gitignore:…*, **and** *Add a license:…*



4. **Click on** 

.

Create a Repository



Clone a Repository (make local)

Option 1: Download as .zip file (quick-&-dirty).

Option 2: **Pull with git bash**.

1. **Create a new folder for GitHub repositories (e.g., *D:\GitHub*).**
2. **Open Git Bash (Windows) / command line (Linux).**
   → Requires that git is installed (https://git-scm.com/downloads)
3. **Navigate to the GitHub folder (git bash):** *cd "D:/GitHub/"*
4. **Clone the before created repository:**
   git clone https://github.com/sschwindt/fruits

Keep Git Bash open – we'll need it!

Verify that the folder *D:\GitHub\fruits* contains the repository (explorer).

Modify a Repository (local)

Modify *D:\GitHub\fruits \README.md* **by using any text editor:**

Push local changes to a remote Repository

Go back to Git Bash:

1. **Go to local copy:** *cd fruits*
2. **Verify status of local copy:** *git status*
   → Git Bash should prompt that README.md was modified (nothing else).
3. Optional: see what has been changed: *git diff*
4. **Add (stage) changes:** *git add .*
5. **Commit changes:**
   git commit –m "Modify Readme"

Determines what is added.
git add README.md would be sufficient here.
.gitignore provides enhanced control

Keep track of commits by adding a message (-m).

5. Optional: update local branch (more later on): *git pull --rebase*
6. **Push changes:** *git push* (user name and password will be queried)

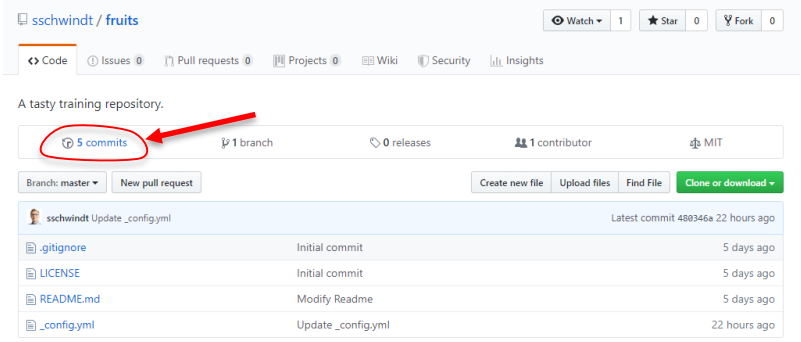
It may be necessary to set the upstream origin : --set-upstream origin master

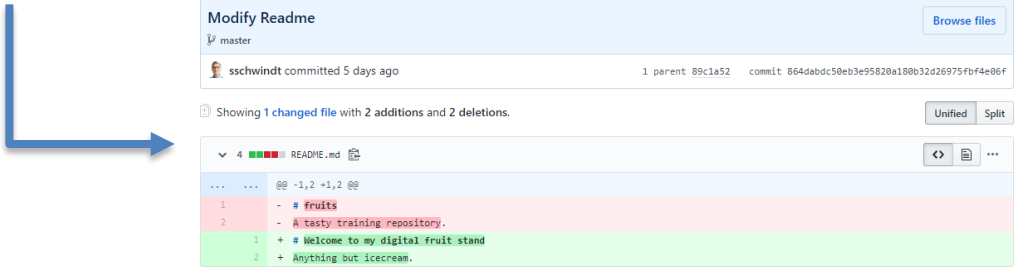
Push local changes to a remote Repository

7. Optional: View commit history: *git log*

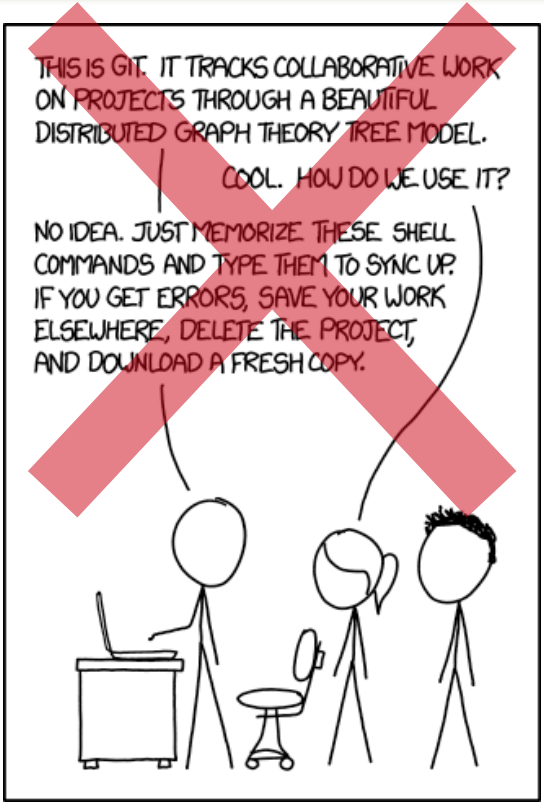
OR

View detailed commit history on GitHub





Updating & merging local repositories



No!

Let's do an exercise ...

Pull/merge changes from a remote Repository

Partner exercise

Partner A

Invite a partner to your repository

1. **Go to your repo on GitHub:** e.g., *https://github.com/klarrieu/veggies*
2. **Select** *Settings* → *Collaborators* **and add partner B.**

Partner B

Accept the invitation to collaborate & clone the repository

1. **Go to local folder:** *cd "D:/GitHub"*
2. **Clone the repository:** *git clone https://github.com/klarrieu/veggies*
3. **Go to local copy:** *cd veggies*

Now both partners should have a local copy of A's repository.

Pull/merge changes from a remote Repository

Both partner A and B

→ **Modify** *README.md* **in your local repository**

→ **Add (***git add .* **), commit (***git commit –m "I did…omg"***), and push (***git push***) your changes**

… and see what happens.

Pull/merge changes from a remote Repository

Modifying your local copy, it appears nothing is wrong…

```
C:\Users\Neth\Code\veggies>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")

C:\Users\Neth\Code\veggies>git add .

C:\Users\Neth\Code\veggies>git commit -m "ate my veggies"
[master 4764fa5] ate my veggies
 1 file changed, 4 insertions(+)
```

Pull/merge changes from a remote Repository

…but when you try to push your changes to remote, you see this!

```
C:\Users\Neth\Code\veggies>git push
To https://github.com/klarrieu/veggies.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/klarrieu/veggies.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

While we were modifying the local repo, changes were made to remote.

We need to pull the remote changes before we can integrate our local changes (*git pull --rebase*).

Pull/merge changes from a remote Repository

Merge conflict: Multiple contributors pushed changes to the same lines in the same file.

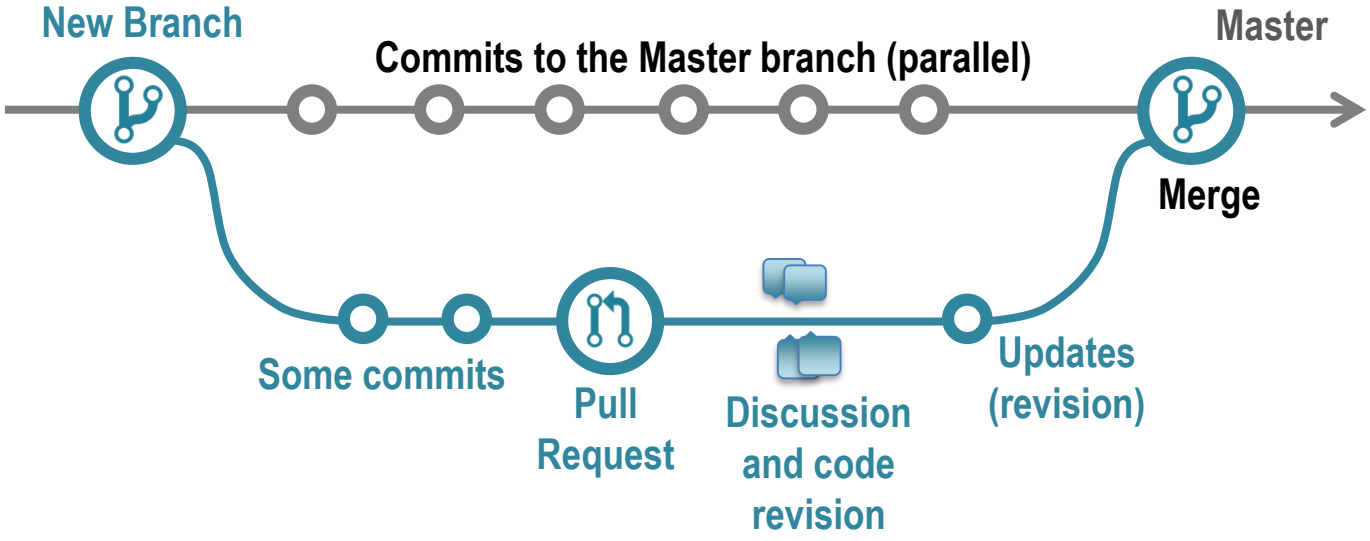
→ Git does not know which changes should be kept.

```
C:\Users\Neth\Code\veggies>git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/klarrieu/veggies
   f45ef6d..72938bd  master     -> origin/master
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
```

Remedy:

1. **Open the file where the merge conflict occurred** (README.md)
2. **Edit the file, deciding which lines of code to keep in the process.**
3. **Add** (*git add .*), **commit** (*git commit –m "Fixed merge conflict"*), **and push** (*git push*) **your changes.**

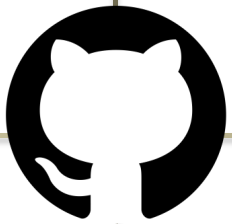
Working with branches



Further reading: https://help.github.com/en/articles/about-branches

Images: slidesharecdn.com, octicons.github.com

Markdown & Documentation



What is Markdown

Created in 2004, Markdown became a popular **markup** language.

Syntactically distinguishable computer text

Syntactic text

```
# Better than Word/Richtext
*OS*-independent
functionality:
- Avoid formatting of the
same kind of thing
redundantly (and
inconsistently)
- Backwards compatibility
- Formulae handling … etc.
```

Marked up

Better than Word/Richtext

OS-independent functionality:

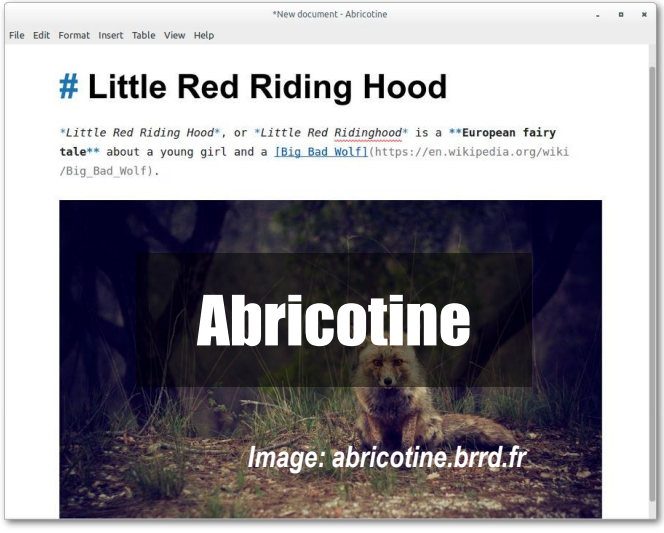
- *Avoid formatting of the same kind of thing redundantly (and inconsistently)*
- *Backwards compatibility*
- *Formulae handling … etc.*

- Popular markup languages: LaTeX, XML, HTML, **Markdown** (simplified HTML)
- Popular editors: Remarkable, Abricotine, Mark Text, Typora

Get your favorite ...

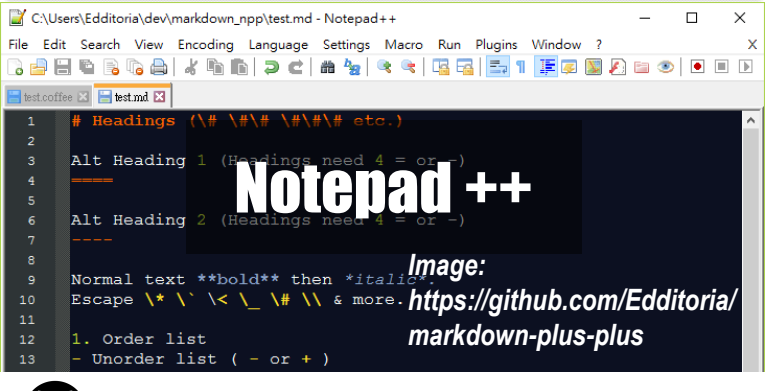
Remarkable

Image: remarkableapp.github.io



Abricotine

Image: abricotine.brrd.fr



Notepad ++

Image: https://github.com/Edditoria/markdown-plus-plus

Mark Text

Image: marktext.app

How to use Markdown (.md) (1)

```
*Italic*
**Bold**
~~Strikethrough~~

# Heading 1
## Heading 2

[Link](https://fruitsinfo.com)


![ImgName](https://raw.githubusercontent.com/RiverArchitect/Media/master/images/RA_icon.png)
```

Italic
Bold
~~Strikethrough~~

Heading 1
Heading 2

Link

How to use Markdown (.md) (2)

```
> Blockquote

* List item 1
* List item 2

1. Numbered list item 1
2. Numbered list item 2

*** (Horizontal Rule)


Embedded `inline code`

```
code block
print ('To be or not to be')
```
```

> Blockquote

- List item 1
- List item 2

1. Numbered list item 1
2. Numbered list item 2

(Horizontal Rule)

Embedded `inline code`

```
# code block
print ('To be or not to be')
```

How to use Markdown (.md) (3)

```
A table

| Fruit | Kingdom | Genus |
|-------|:-------:|------:|
|Banana | Plantae | Musa |
|Jackfruit|Plantae|Artocarpus|


More linking
[[InternalGithubPage]]
[InternalPageSection](InternalPageSection#Header-name)
```

Go to section *Space="-"*

```
[Defined Reference][1]
```

Define it!

```
[1]: https://wikipedia.org
```

A table

Fruit	***Kingdom***	***Genus***
Banana	*Plantae*	*Musa*
Jackfruit	*Plantae*	*Artocarpus*

More linking

InternalGithubPage

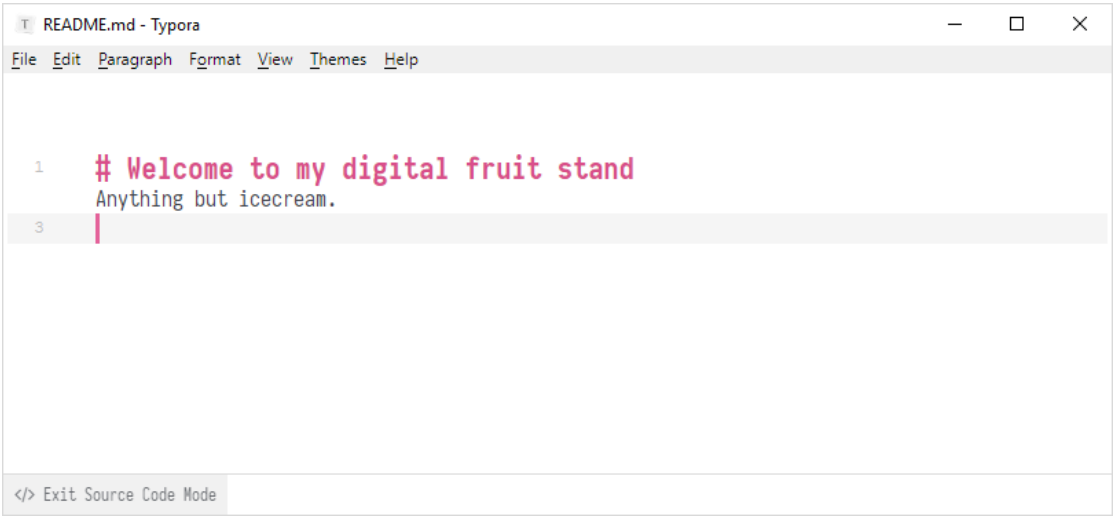
InternalPageSection

Defined Reference

Edit README.md

Go back to Git Bash :

1. **Go to local copy of Repository:** *cd fruits*
2. **Verify status of local copy:** *git status*
3. **In the explorer: Open README.md with Markdwon Editor and rewrite it**: README.md = your automated welcome page



Push local changes to remote

Go back to Git Bash :

1. **Verify status of local copy:** *git status*
   → Git Bash should prompt that README.md was modified (nothing else).
2. **Add (stage) changes:** *git add .*
3. **Commit changes:**

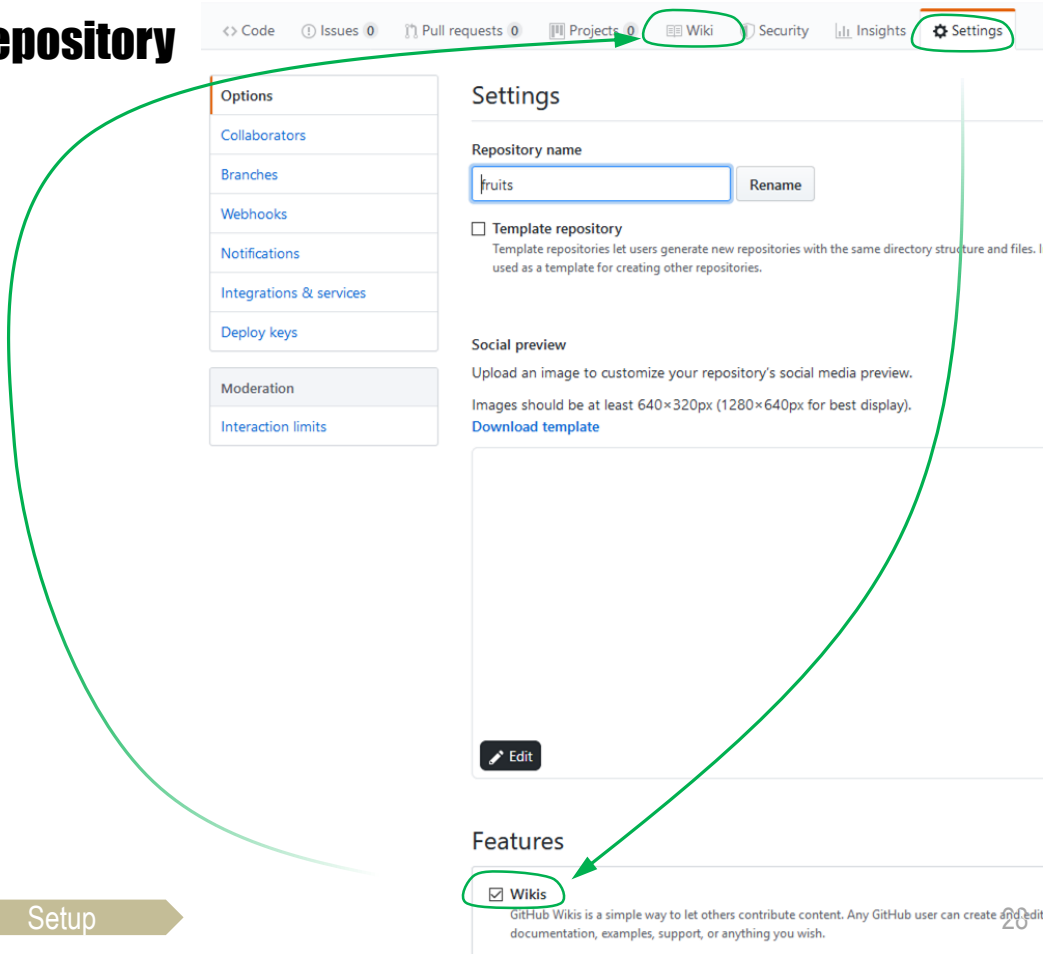
   git commit –m "Make Readme Great Again"

5. **Update local branch** (just good practice…): *git pull --rebase*
6. **Push changes:** *git push*

Working collaboratively is more than sharing codes …

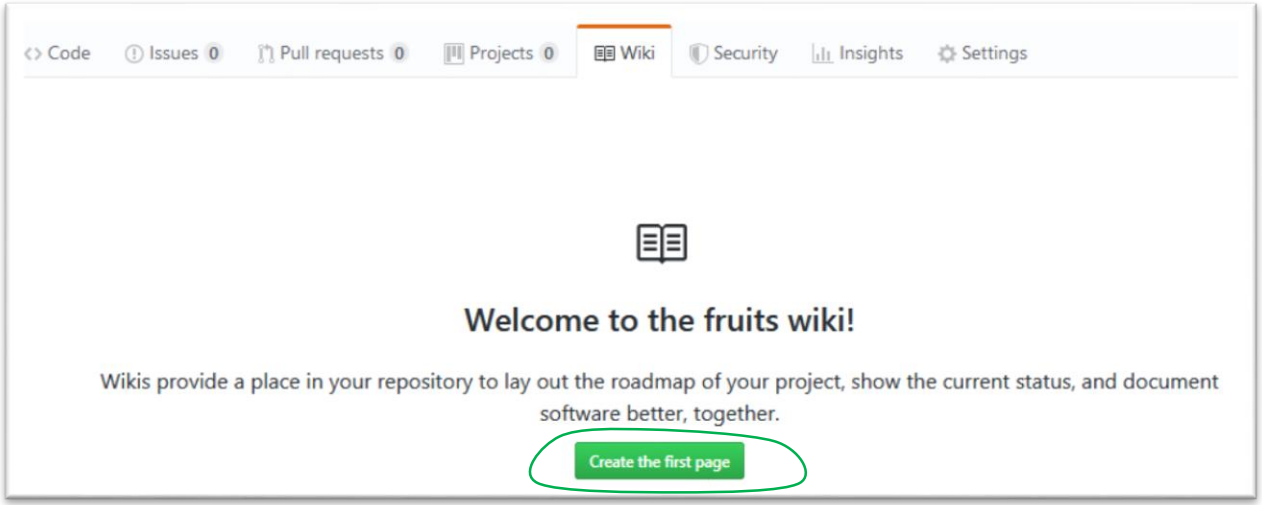
Wikis

Enable Wiki for a repository



Wikis

Get started

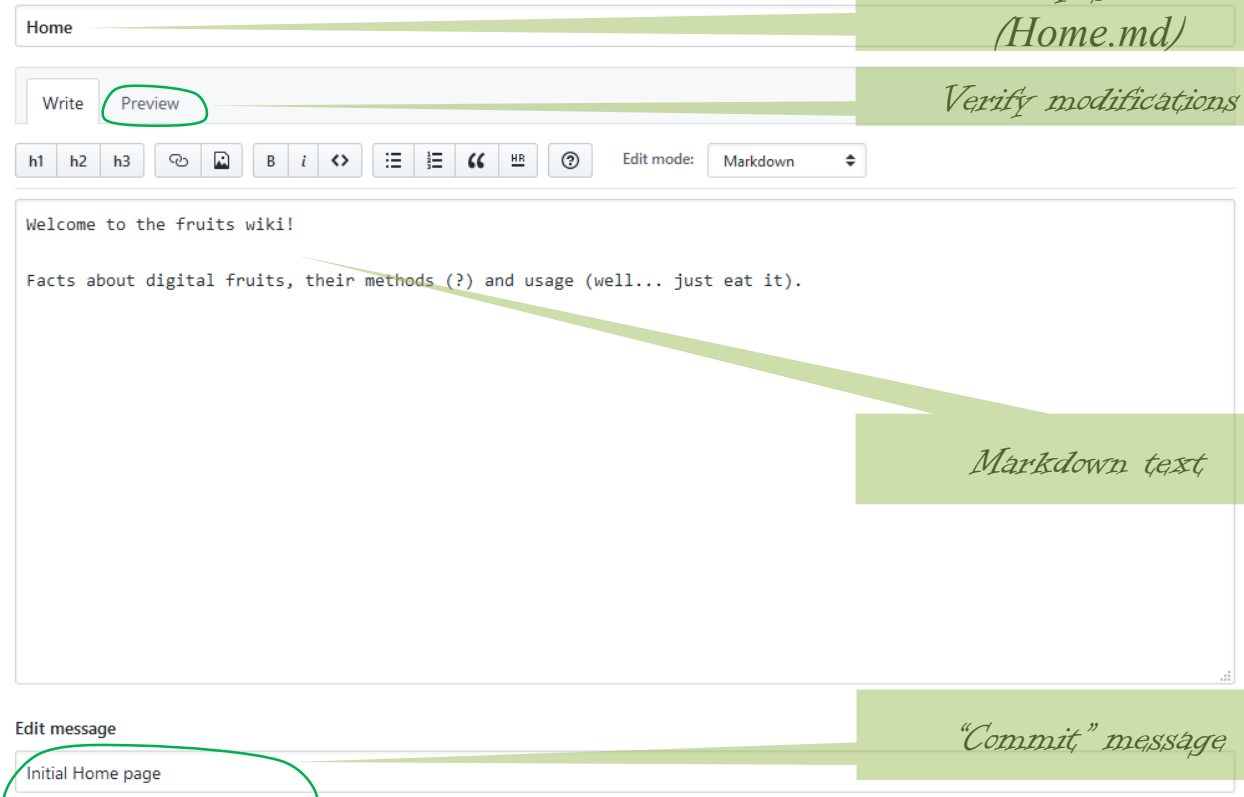


Note: The Wiki is a repository within the repository

→ It can be cloned like any other repository:

`https://github.com/YOUR_ID/REPO.wiki.git`

Get started: Create “Home”.md



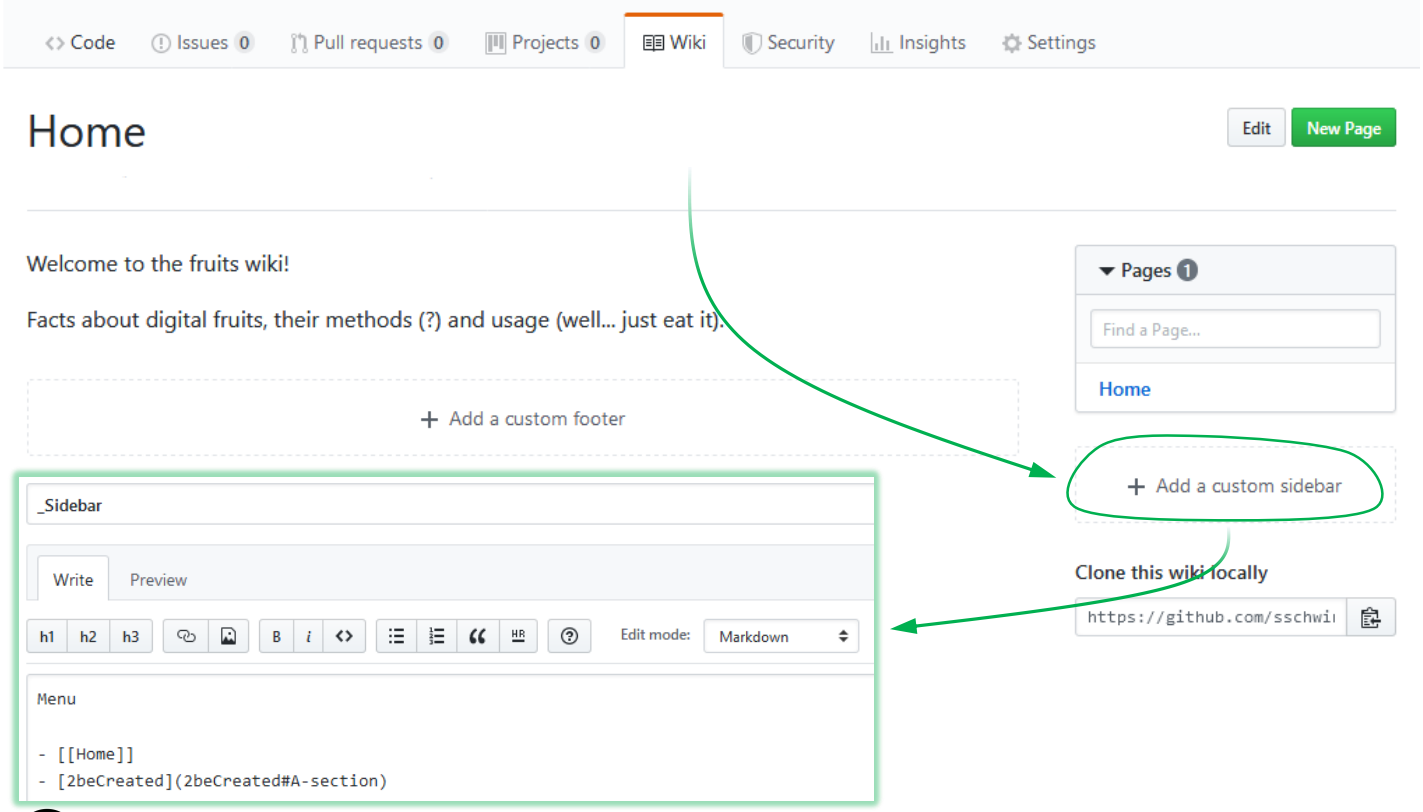
Define page name (Home.md)

Verify modifications

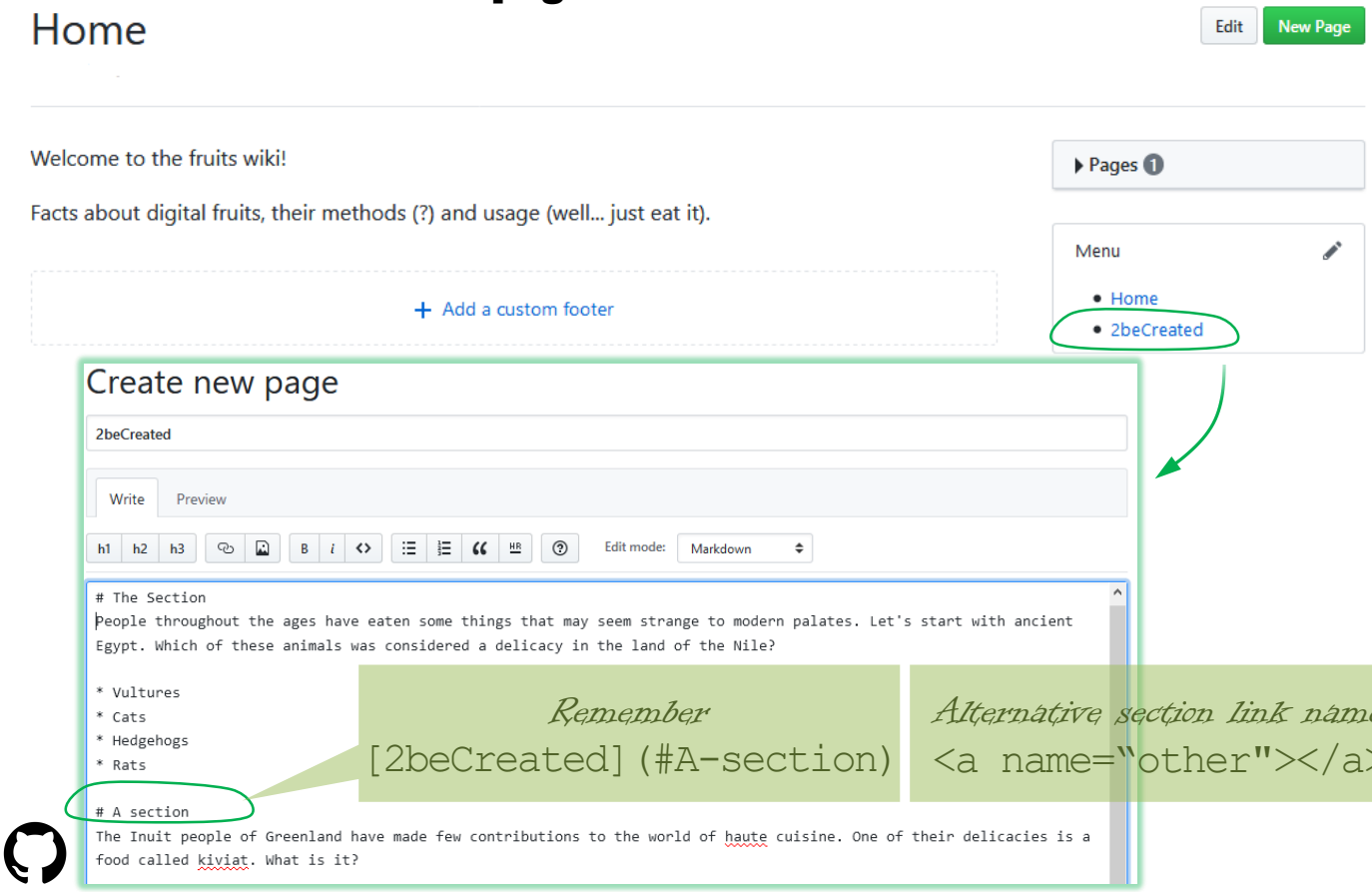
Markdown text

“Commit” message

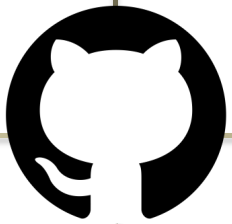
Get started: Add a sidebar



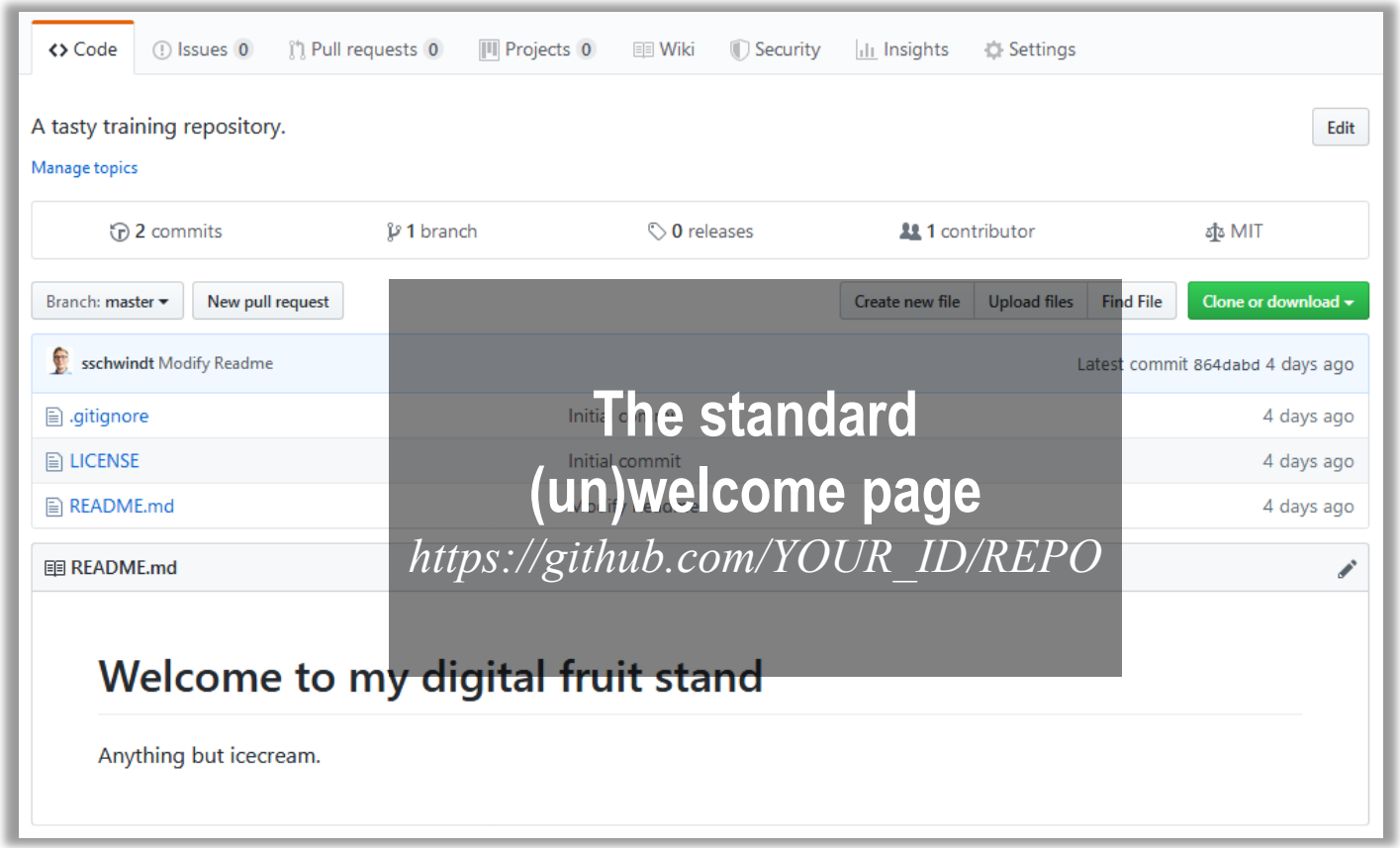
Get started: Add another page



GitHub pages



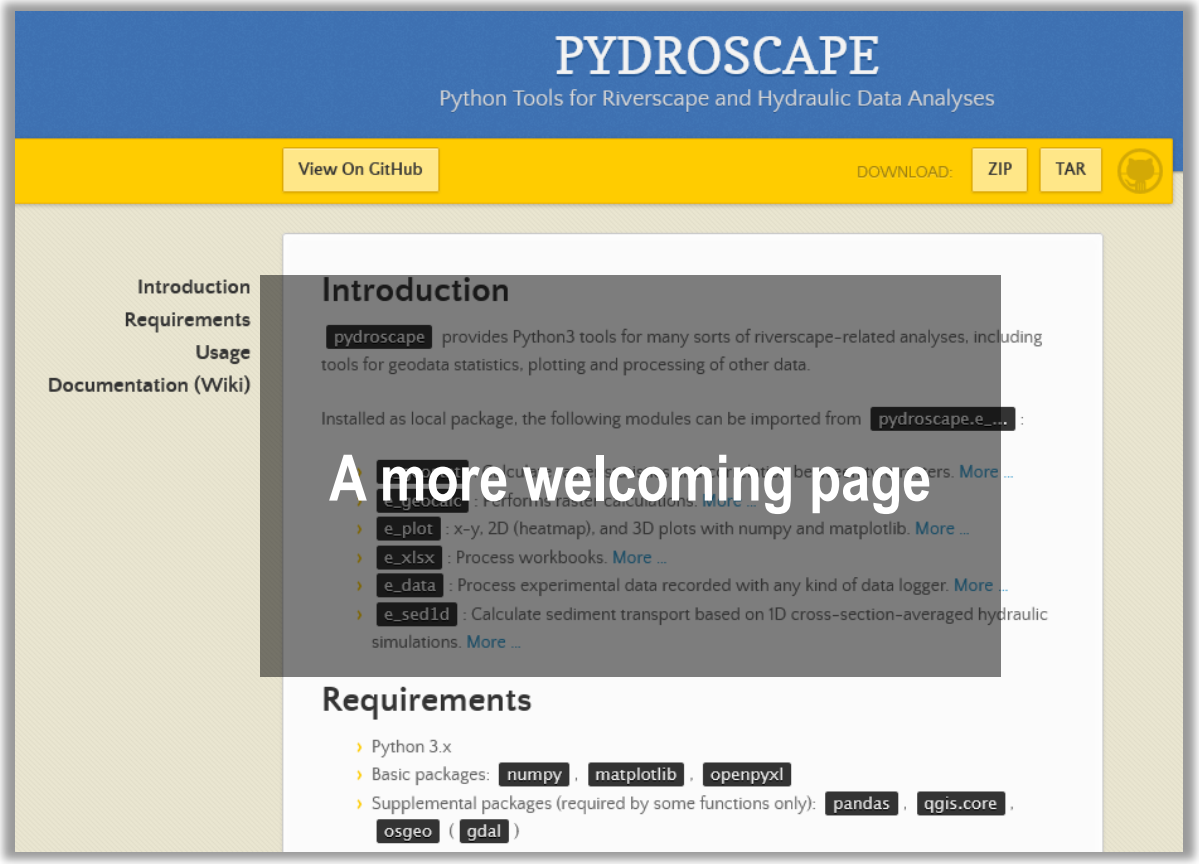
GitHub Pages



The standard (un)welcome page

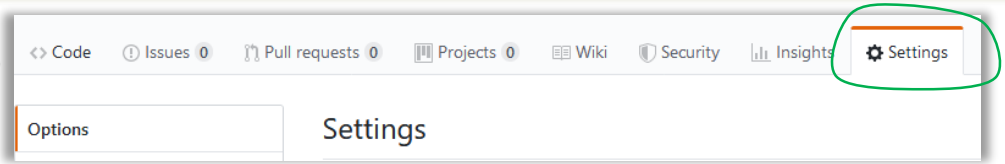
https://github.com/YOUR_ID/REPO

GitHub Pages

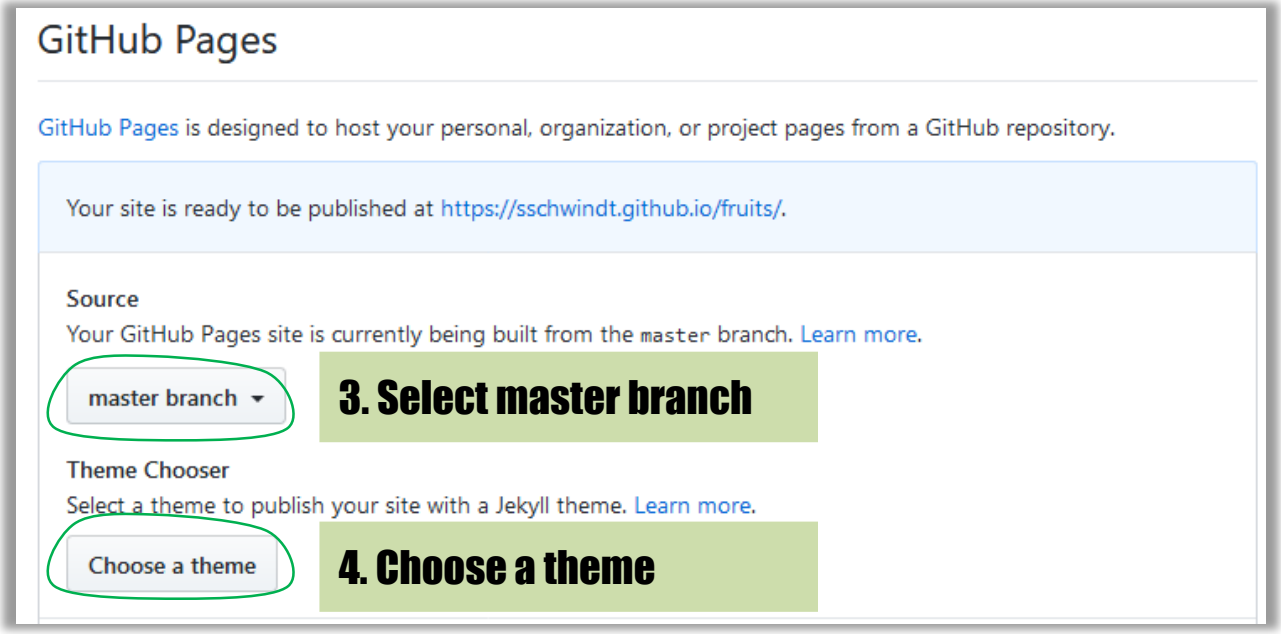


GitHub Pages

1. Go to Settings

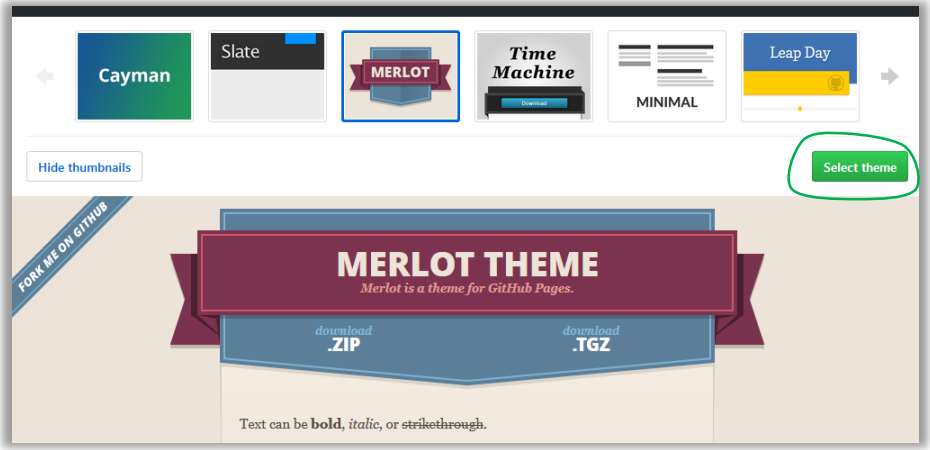


2. Scroll to GitHub Pages



GitHub Pages

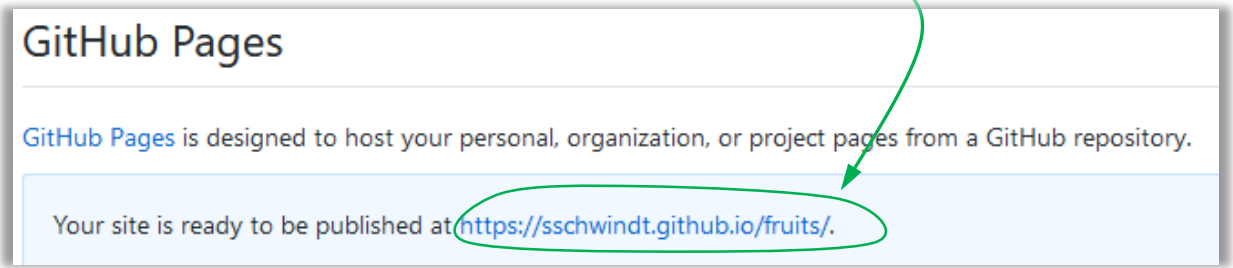
1. Select a Theme



More Themes are available (maybe tricky to install…)

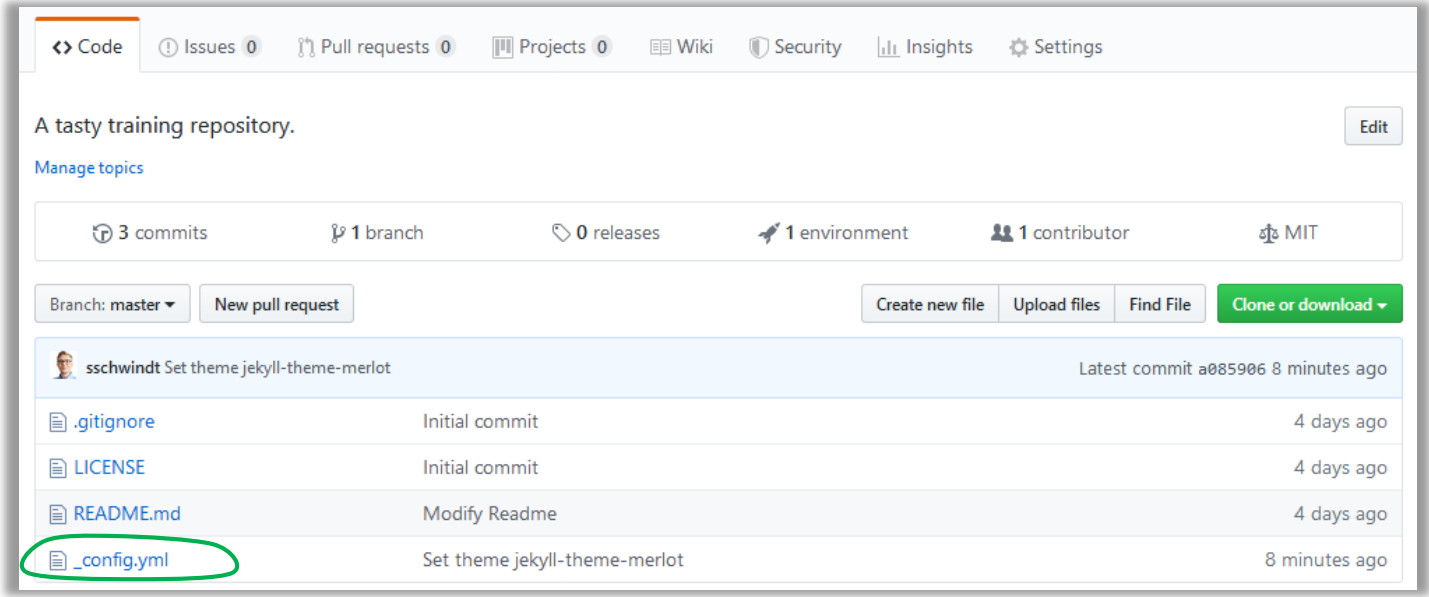
https://jekyllthemes.io/github-pages-templates

2. Wait 2-3 Minutes, then check out GitHub page

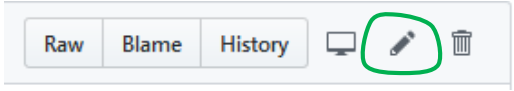


GitHub Pages

Looks nice, but there's room for optimization. The *_config.yml* file provides some options:



Open (click on) *_config.yml* & Start editing



GitHub Pages

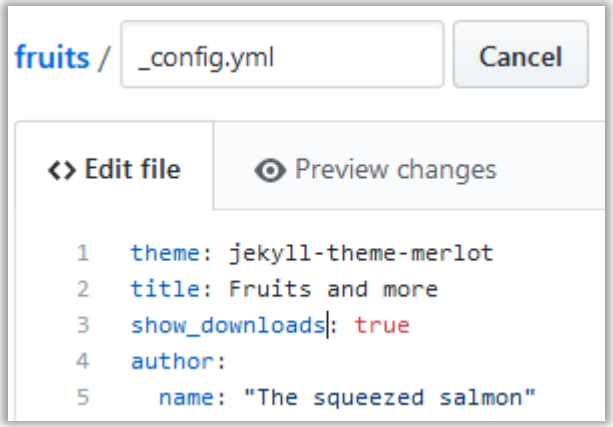
Modify *_config.yml* & adapt it to your needs. Some options:

Keyword:	Possible Values	Description
theme:	*"jekyll-theme-NAME"*	*Define / modify the installed theme.*
title:	*"Another title name"*	*Modify the title shown on github pages.*
description:	*"Text"*	*Modify the repository descriptions*
show_downloads:	*BOOL (true or false)*	*Enable download buttons.*
analytics: *google:*	*Google Analytics ID*	*Use SEO / webmaster tools (also available for bing)*
author: *name:*	*"Text"*	*E.g., "Ms. X"*
timezone:	*America/Los_Angeles*	*Uses http://en.wikipedia.org/wiki/List_of_tz_database_time_zones*

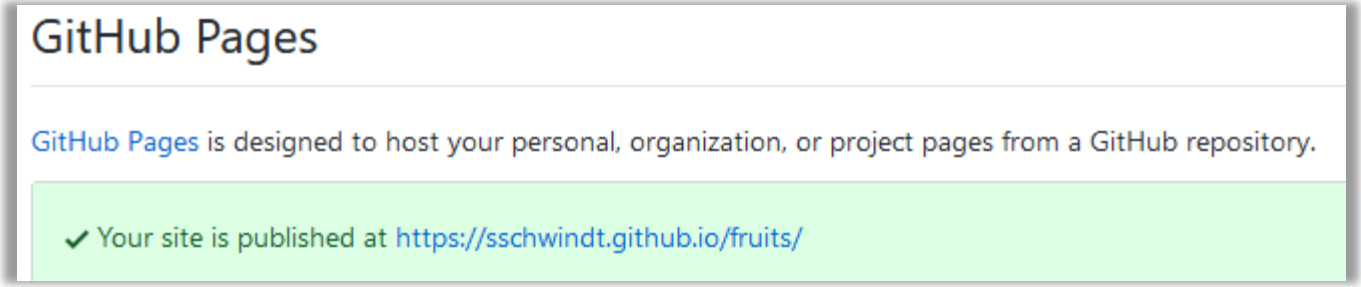
More: https://jekyllrb.com/docs/variables/

GitHub Pages

After editing, save and wait for 2-3 minutes.



Then refresh :



If variables were badly defined, you'll receive an email with a site generation error notice.

GitHub Pages

- There are more options to adapt templates

  Find the default.html of the selected Jekyll-theme and copy it into your repository :

  1. Create *_layouts/default.html*
  2. Edit *_layouts/default.html*

Note: HTML skill are required.

- Creating a “home” repository: *https://YOUR_ID.github.io*

This is how GitHub Pages can constitute you homepage, cheap and well referenced in google (SEO is another topic though).

1. Create a new repository called *YOUR_ID.github.io*
2. Install a GitHub pages theme
3. Modify & adapt it to your needs.

Thank you for listening.

Get inspired by the River Architect Repository:
https://riverarchitect.github.io

More information and contact:
https://sschwindt.github.io
https://github.com/klarrieu