Git

Version control with Git and GitHub

1.1

Introduction

- Evan Giles and Rebecca Blundell

  evan@catalyst.net.nz

  rebeccablundell@catalyst.net.nz
- Please interrupt to ask questions

Overview

- These slides are online: https://cat-training-2020.github.io
- Tutorial format (with some lecturing thrown in)
- Tutorial will be in phases (time permitting)
  1. Hands-on with GitHub and Atom
  2. Questions
  3. Discussion of GitHub workflows
  4. Hands-on with Git on the command line

Git

 https://git-scm.org/

Git

Git is a **version control system**:

software that manages different versions of files

Git

Why is it useful?

1. Keep track of changes to files
2. Review and revert back to old versions
3. Synchronise files between different locations
4. Test changes without losing the original copy
5. Facilitates teamwork

Git

Why Git and not some other software?

1. Performance
2. Flexibility
3. **Popularity**

GitHub

 https://github.com/

GitHub

GitHub is a **hosting site for Git repositories**

(with lots of extra features)

GitHub

Why is GitHub useful?

1. Free hosting for open source projects
2. Interface for browsing and editing code
3. Workflows for collaborating with others
4. Create a static website with GitHub pages
5. Comprehensive documentation:
   https://docs.github.com/en

GitHub

Why GitHub and not some other software?

1. **Popularity**

Atom

 https://atom.io/

Atom

Atom is a **text editor**

with some useful Git and GitHub-related features

Let's get started

Using Git's primary workflow

Overview

1. **Create** a project (on GitHub)
2. **Clone** the project (onto your computer)
3. **Change** some files
4. **Commit** the changes
5. **Push** the changes (to GitHub)

1. Create a project

Initialise a new "repository" from within GitHub

Create a project

- Go to https://github.com/
  - If you have an account, log in
  - If you don't, create one
    - You may need to verify your email address
- Click **New**
  - Or navigate to https://github.com/new

Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 50 million developers.

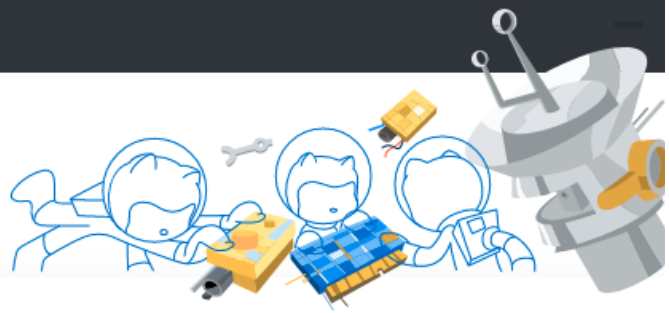
Email

Password

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. Learn more.

Sign up for GitHub

By clicking “Sign up for GitHub”, you agree to our Terms of Service and Privacy Statement. We'll occasionally send you account related emails.



There were problems creating your account.

Username *

myaccountname12333

Email address *

[redacted]

Password *

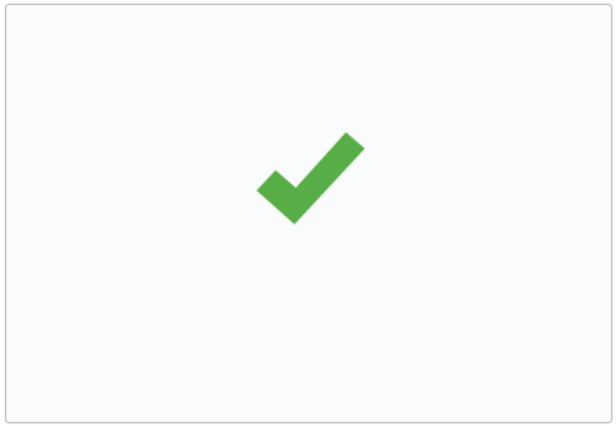
••••••••••••••••••••

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. Learn more.

Email preferences

Send me occasional product updates, announcements, and offers.

Verify your account



Create account

By creating an account, you agree to the Terms of Service. For more information about GitHub's privacy practices, see the GitHub Privacy Statement. We'll occasionally send you account-related emails.

ected plan: Free

Welcome to GitHub

Woohoo! You've joined millions of developers who are doing their best work on GitHub. Tell us what you're interested in. We'll help you get there.

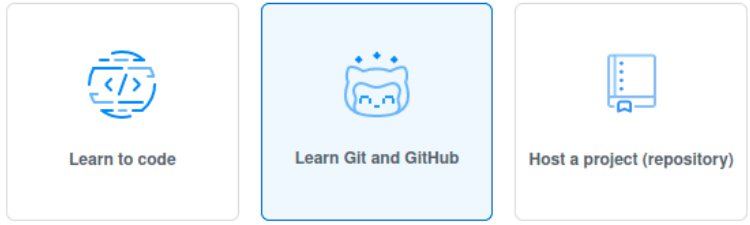
What kind of work do you do, mainly?

- Software Engineer — I write code
- Student — I go to school
- Product Manager — I write specs
- UX & Design — I draw interfaces
- Data & Analytics — I write queries
- Marketing & Sales — I look at charts
- Teacher — I educate people
- Other — I do my own thing

How much programming experience do you have?

- None — I don't program at all
- A little — I'm new to programming
- A moderate amount — I'm somewhat experienced
- A lot — I'm very experienced

(Select up to 3)



I am interested in:

languages, frameworks, industries

We'll connect you with communities and projects that fit your interests.

For example: `fish` `spring` `webauthn`

Complete setup

Please verify your email address

Before you can contribute on GitHub, we need you to verify your email address.

An email containing verification instructions was sent to [redacted].

Resend verification email | Change your email settings

Sign in to GitHub

Username or email address

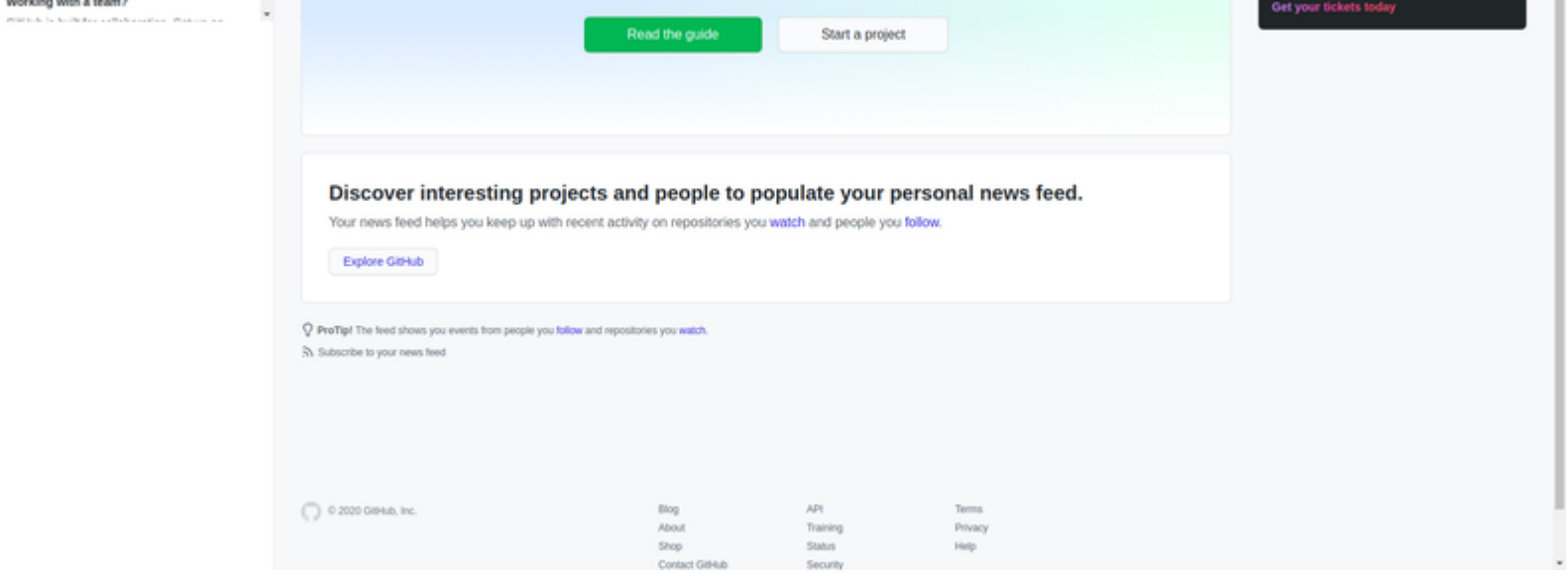
Password Forgot password?

Sign in

New to GitHub? Create an account.

Create a project

- Enter repository name **cs4hs**
- Check **Add a README file**
- Click **Create repository**


Working with a team?
Read the guide
Start a project
Get your tickets today
Discover interesting projects and people to populate your personal news feed.
Your news feed helps you keep up with recent activity on repositories you watch and people you follow.
Explore GitHub
ProTip! The feed shows you events from people you follow and repositories you watch.
Subscribe to your news feed
© 2020 GitHub, Inc.
Blog
About
Shop
Contact GitHub
API
Training
Status
Security
Terms
Privacy
Help

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
Import a repository.

Owner * **Repository name** *

[redacted] / cs4hs

cs4hs is available.

Great repository name emorable. Need inspiration? How about **special-octo-fortnight**?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. Learn more.

Add .gitignore
Choose which files not to track from a list of templates. Learn more.

Choose a license
A license tells others what they can and can't do with your code. Learn more.

This will set `main` as the default branch. Change the default name in your settings.

Create repository

main 1 branch 0 tags

Go to file | Add file | Code

Initial commit a77a16c now 1 commits

README.md	Initial commit	now

README.md

cs4hs

About

No description, website, or topics provided.

Readme

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Remote cs4hs repository

- Exists on GitHub's servers
  - It is a "remote" repository
- Publically accessible
  - Anyone can see it
  - Only you can modify it

2. "Clone" the project

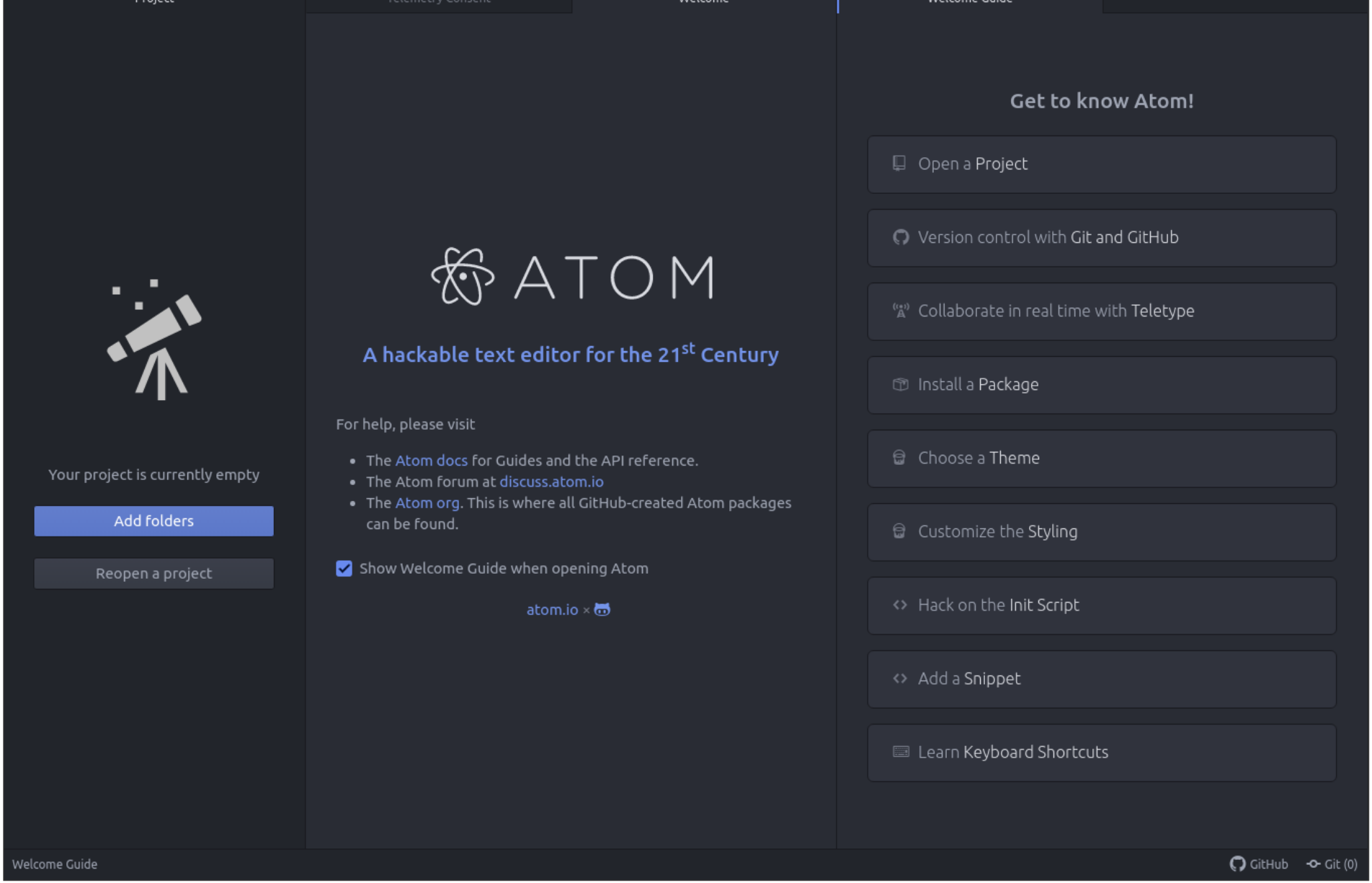
Copy the project so you can edit it

"Clone"

- Given the location of a repository
- Makes a copy of the project
- Stores it on your computer
- Links the two copies

Clone the cs4hs repository

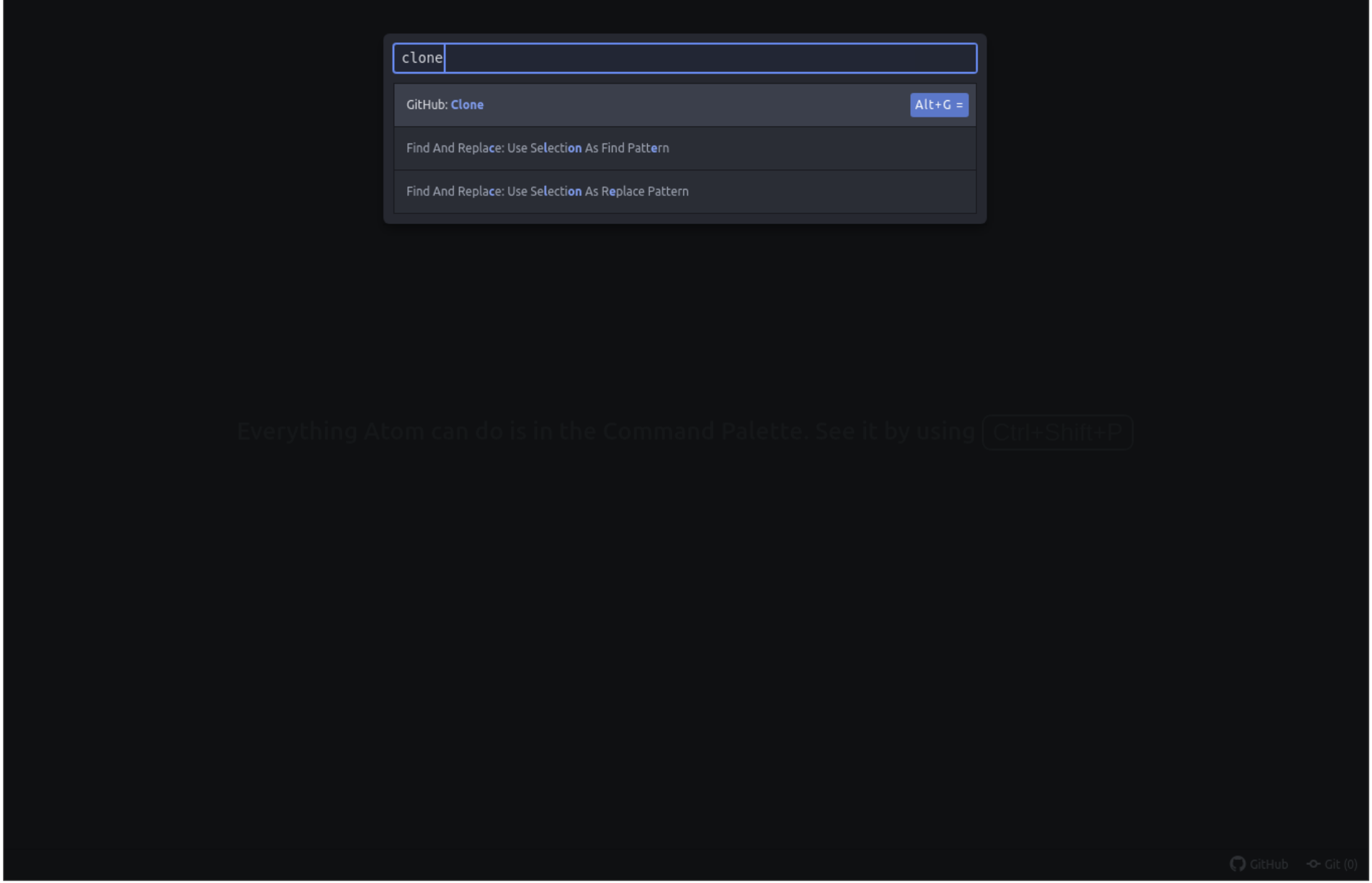
- Open Atom
  - Close any "welcome" tabs

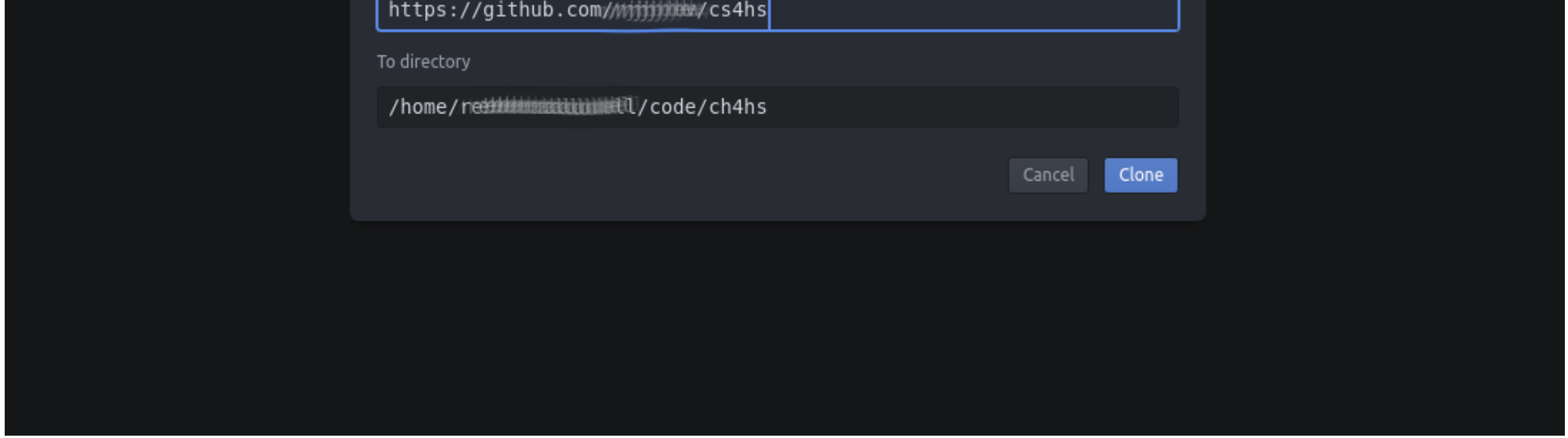

Project
Welcome
Welcome Guide
Your project is currently empty
Add folders
Reopen a project
ATOM
A hackable text editor for the 21st Century
For help, please visit
The Atom docs for Guides and the API reference.
The Atom forum at discuss.atom.io
The Atom org. This is where all GitHub-created Atom packages can be found.
Show Welcome Guide when opening Atom
atom.io
Get to know Atom!
Open a Project
Version control with Git and GitHub
Collaborate in real time with Teletype
Install a Package
Choose a Theme
Customize the Styling
Hack on the Init Script
Add a Snippet
Learn Keyboard Shortcuts
Welcome Guide
GitHub
Git (0)

Clone the cs4hs repository

- Press **Control-Shift-P** to enter a command
  - Type **GitHub: Clone** and press **Enter**
- Choose file locations
  - **Clone from:** your GitHub repository URL

    https://github.com/username/cs4hs
  - **To directory:** wherever you like, e.g. home/yourname /code/cs4hs
  - To view folder pane on left in Atom, press Alt + \ or View -> 'Toggle tree view'


clone
GitHub: Clone
Alt+G =
Find And Replace: Use Selection As Find Pattern
Find And Replace: Use Selection As Replace Pattern
Everything Atom can do is in the Command Palette. See it by using Ctrl+Shift+P
GitHub
Git (0)


https://github.com/[redacted]/cs4hs
To directory
/home/[redacted]/code/ch4hs
Cancel
Clone

Local cs4hs repository

- Exists on your computer
  - It is a "local" repository
- Files and folders behave as usual
- Linked to the version on GitHub

3. Change some files

Now you can make changes to your local copy

Edit the README

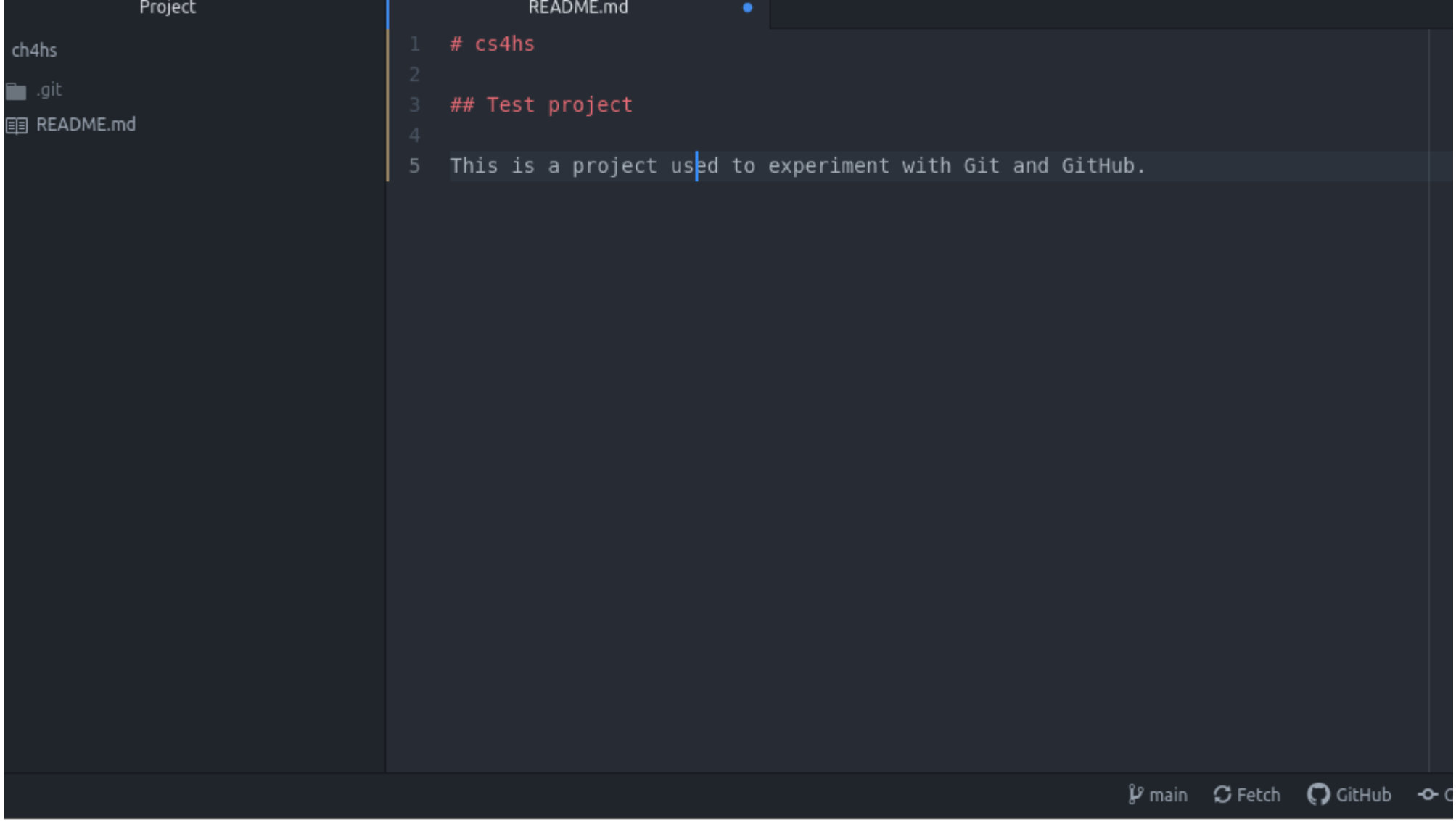
- Edit file "README.md"

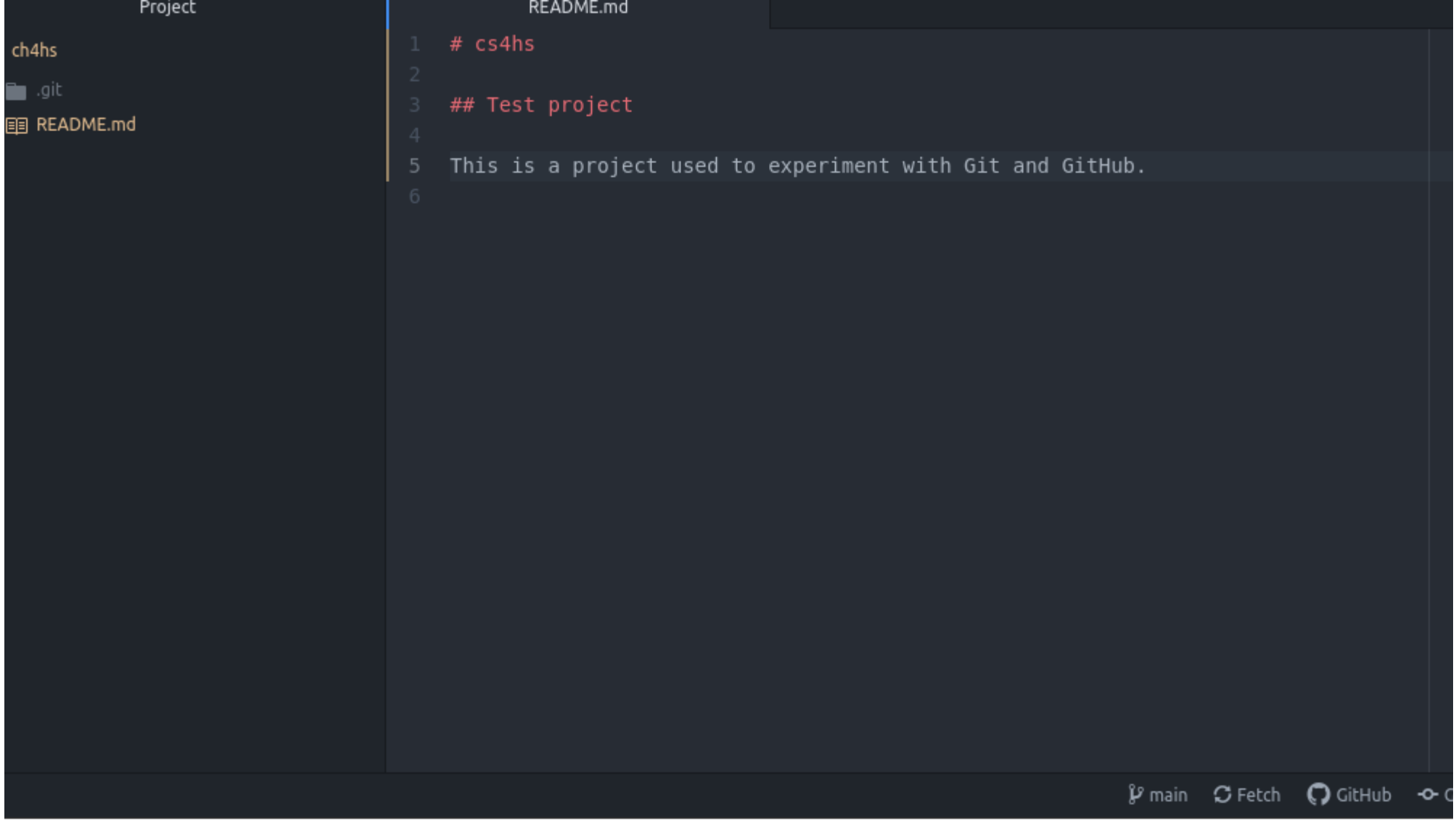
```
# cs4hs

## Test project

This is a project used to experiment with Git and GitHub.
```

- Save with **Control-S** (or from the **File** menu)


Project
ch4hs
.git
README.md
README.md
1 # cs4hs
2
3 ## Test project
4
5 This is a project used to experiment with Git and GitHub.
main
Fetch
GitHub

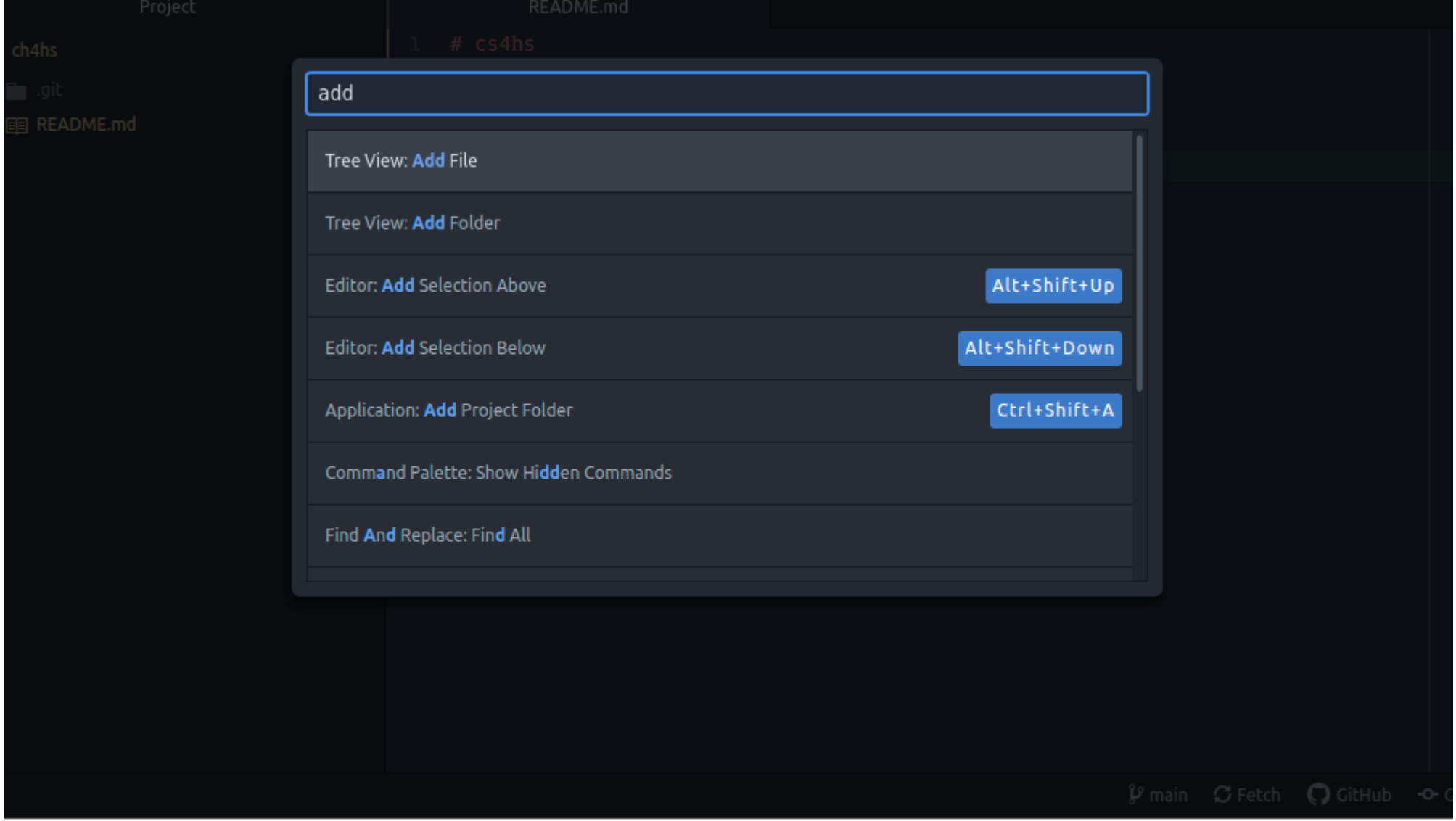

Project
README.md
ch4hs
.git
README.md
1 # cs4hs
2
3 ## Test project
4
5 This is a project used to experiment with Git and GitHub.
6
main
Fetch
GitHub

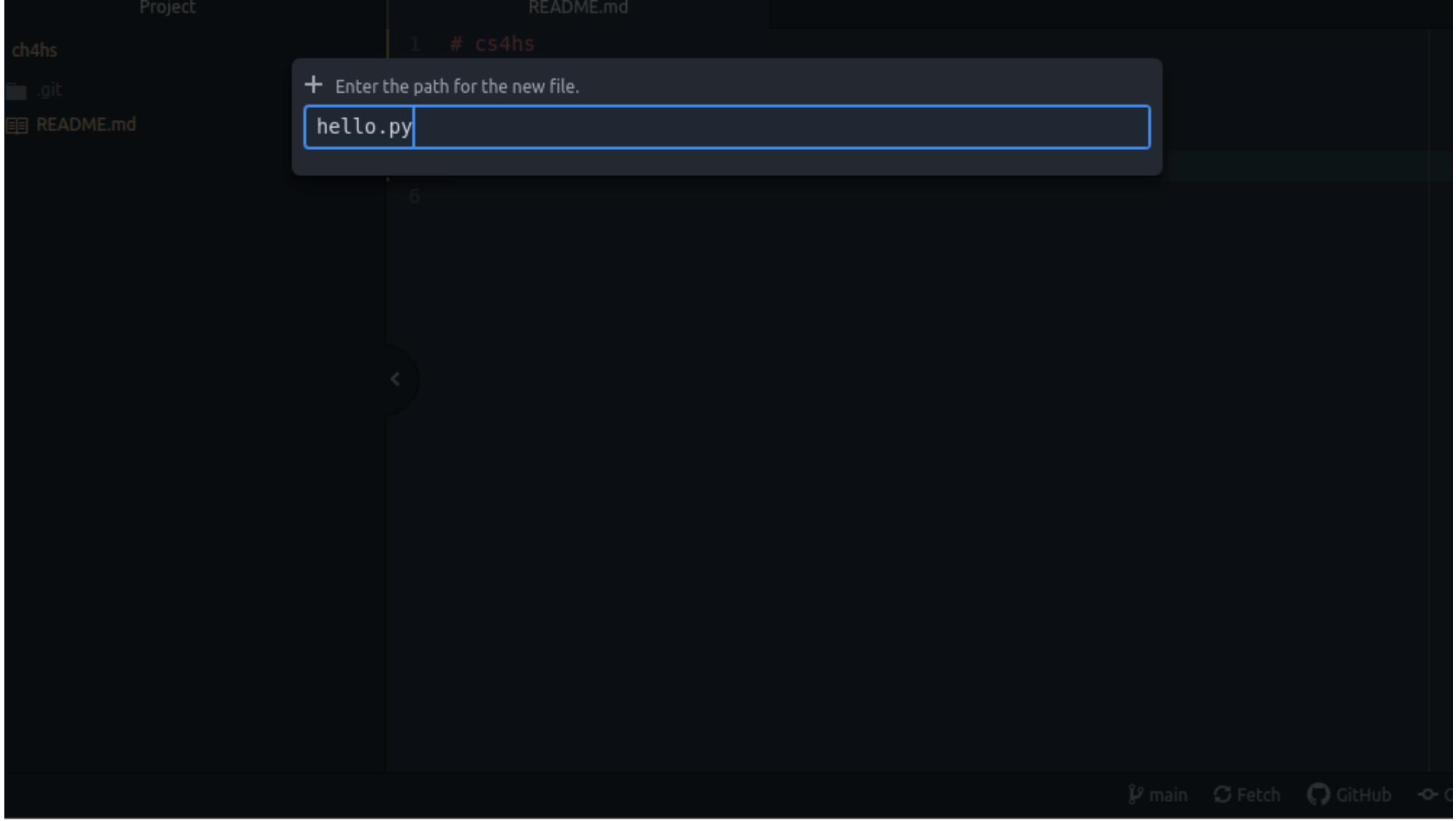
Add a Python program

- Enter **Control-Shift-P** and **Add File**
- Enter the path for the new file: **hello.py**

```
print("Kia ora koutou,")
print("")
print("This is a simple python program.")
print("What does it do? Just prints out this message.")
print("")
print("Ngā mihi, Evan & Rebecca")
```

- Remember to save with **Control-S**


Project
README.md
ch4hs
.git
README.md
1 # cs4hs
add
Tree View: Add File
Tree View: Add Folder
Editor: Add Selection Above
Alt+Shift+Up
Editor: Add Selection Below
Alt+Shift+Down
Application: Add Project Folder
Ctrl+Shift+A
Command Palette: Show Hidden Commands
Find And Replace: Find All
main
Fetch
GitHub


Project
README.md
ch4hs
.git
README.md
1 # cs4hs
Enter the path for the new file.
hello.py
6
main
Fetch
GitHub

Sidenote: the ".git" folder

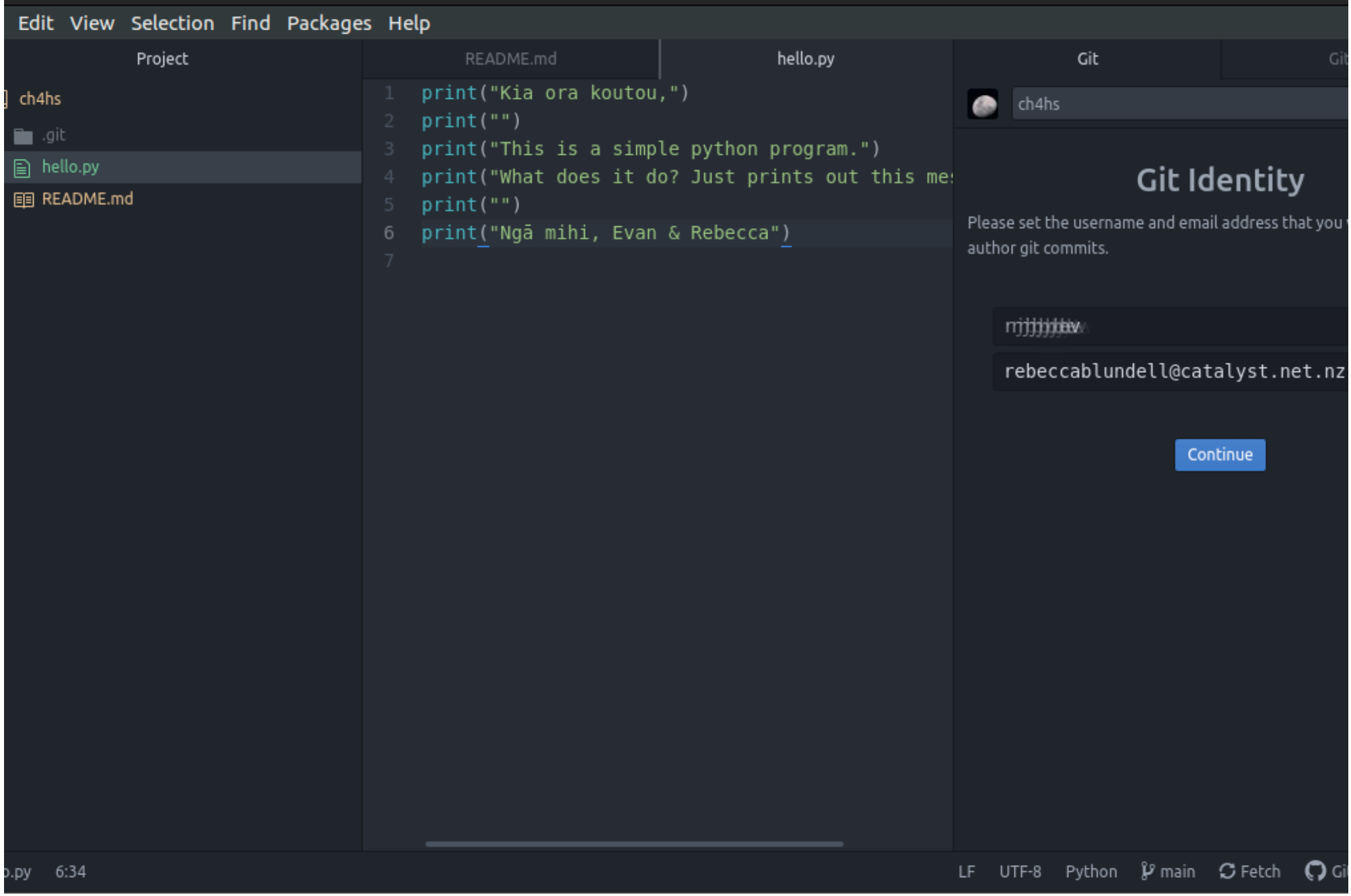
- The ".git" folder contains Git's data files
- They can be edited but not saved in your repository

Local changes

- You now have:
  1. Some edits to a file
  2. An entirely new file
- These changes only exist locally
- They have not been "saved" to your repository yet!

3.1 Tell Git who you are

- Click the **Git** button on the lower right corner of your Atom window
  - You should see a **Git Identity** pane
- Check the username and email you want to use for commits are correct
- Click **Continue**


Edit View Selection Find Packages Help
Project
ch4hs
.git
hello.py
README.md
README.md
hello.py
1 print("Kia ora koutou,")
2 print("")
3 print("This is a simple python program.")
4 print("What does it do? Just prints out this me
5 print("")
6 print("Ngā mihi, Evan & Rebecca")
7
Git
ch4hs
Git Identity
Please set the username and email address that you
author git commits.
rebeccablundell@catalyst.net.nz
Continue
6:34
LF
UTF-8
Python
main
Fetch

Tell Git who you are

An alternative to setting your **Git Identity** as above, is editing the config file:

- Edit file ".git/config"
- Add to the end of the file:

```
[user]
  name = Your Name
  email = you@example.com
```

4. "Stage" and "Commit" the changes

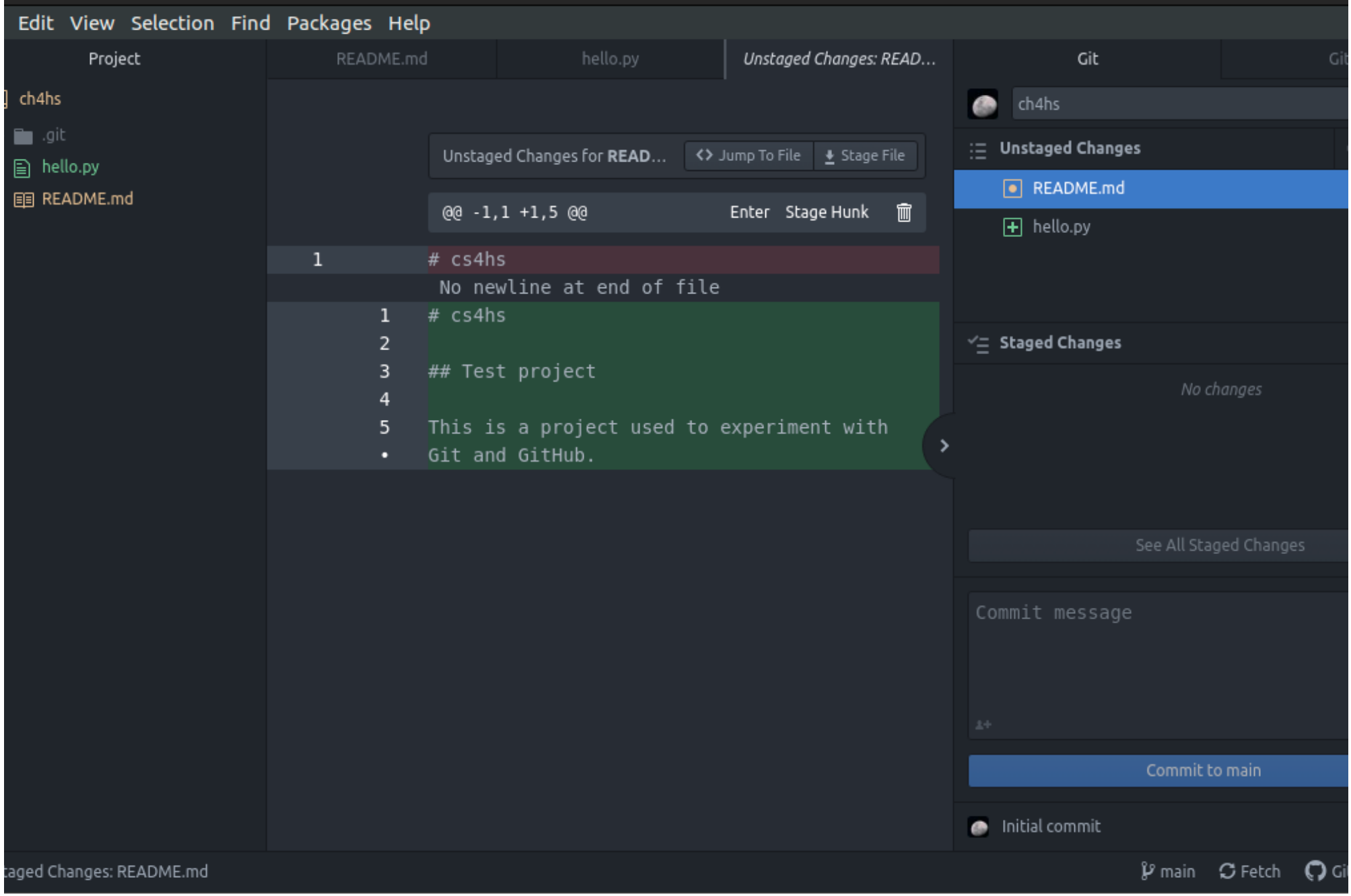
Save your edits to a new version of the project

"Commit"

- Saving changes is a two-step process:
  1. Add them to the "staging area"
     - Tells Git to include the file in the next commit
  2. "Commit" the changes
     - Tells Git to save a new version of the project
- A "commit" is a version
  - Includes a message and attribution information

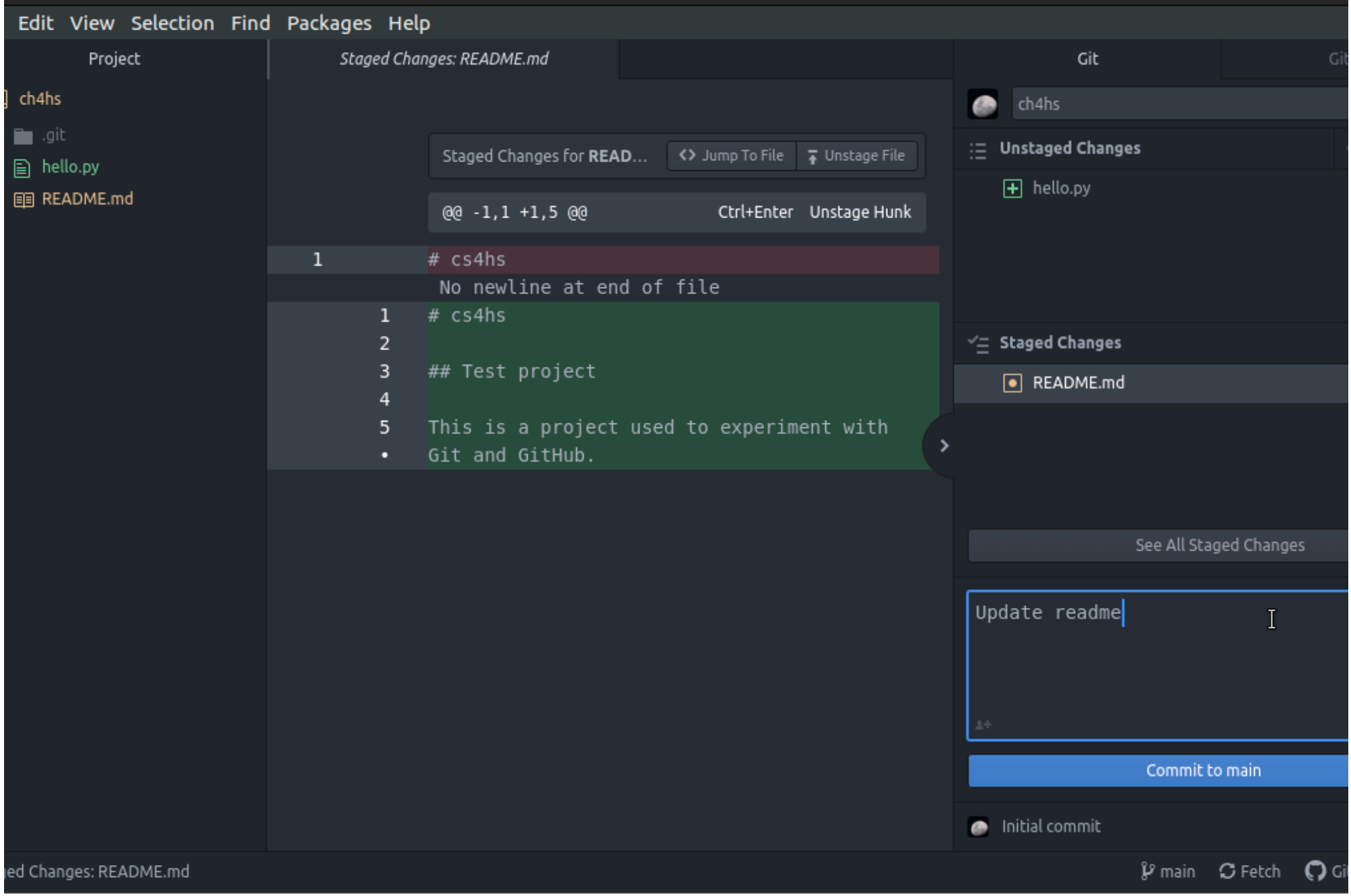
Stage the edited file

- Open the "Git" pane
- Your changes are currently "unstaged"
- **Right-click** and **Stage** "README.md"


Edit View Selection Find Packages Help
Project
README.md
hello.py
Unstaged Changes: READ...
Git
ch4hs
.git
hello.py
README.md
Unstaged Changes for READ...
Jump To File
Stage File
@@ -1,1 +1,5 @@
Enter Stage Hunk
1 # cs4hs
No newline at end of file
1 # cs4hs
2
3 ## Test project
4
5 This is a project used to experiment with
• Git and GitHub.
ch4hs
Unstaged Changes
README.md
hello.py
Staged Changes
No changes
See All Staged Changes
Commit message
Commit to main
Initial commit
taged Changes: README.md
main
Fetch

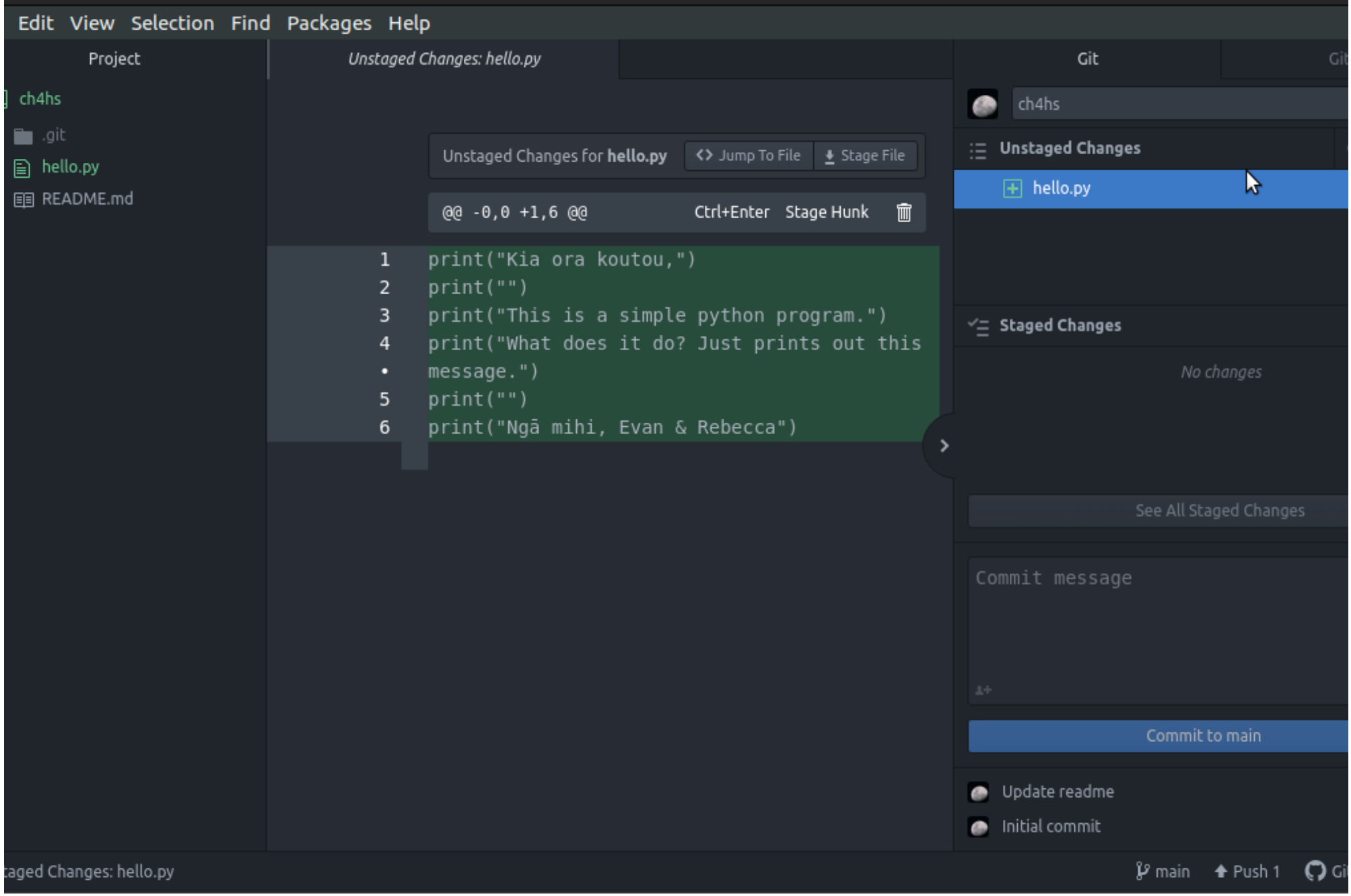
Commit the edited file

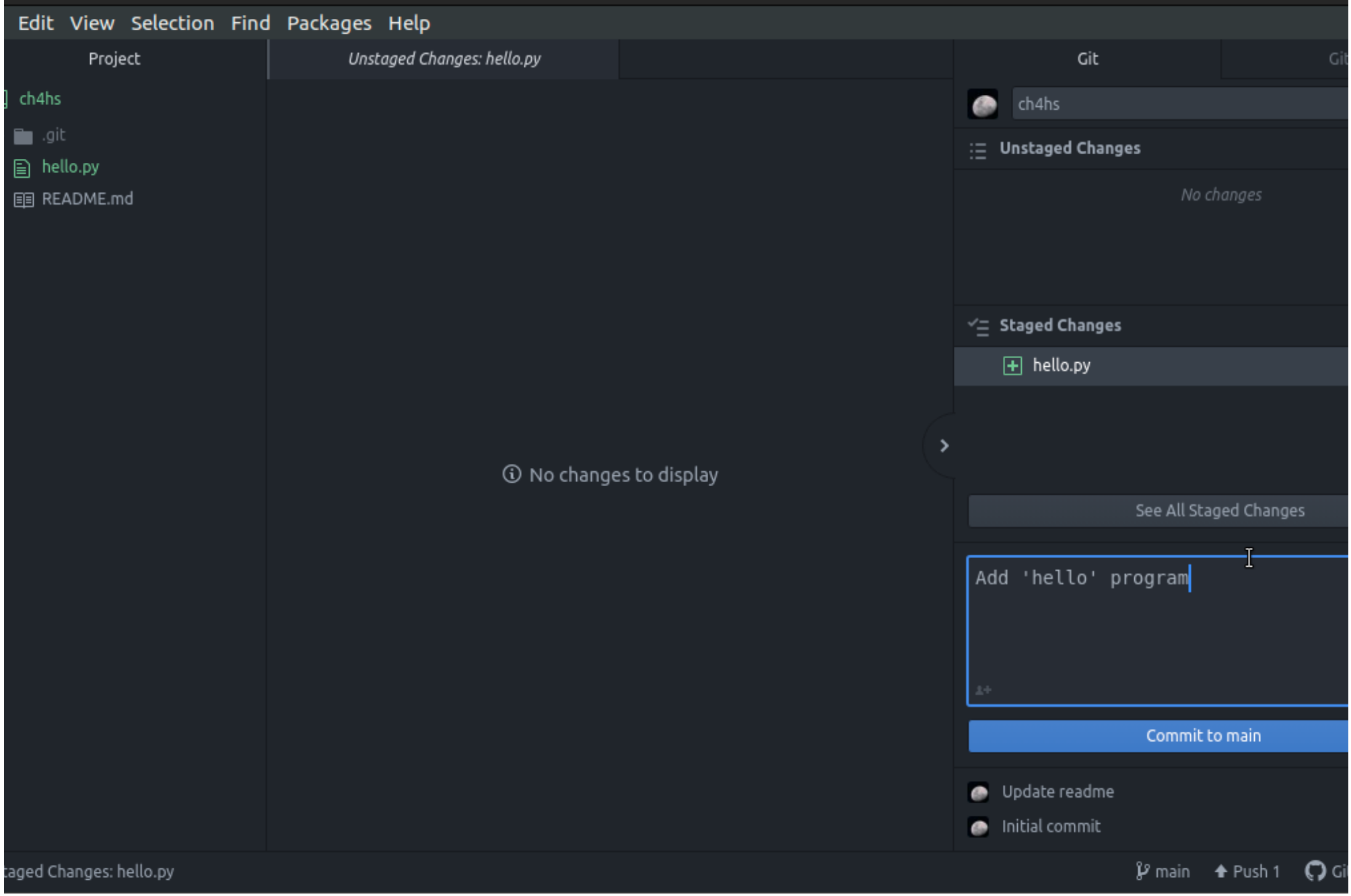
- Enter a **Commit message**: something descriptive
- Click **Commit to main**


Edit View Selection Find Packages Help
Project
Staged Changes: README.md
Git
ch4hs
.git
hello.py
README.md
Staged Changes for READ...
Jump To File
Unstage File
@@ -1,1 +1,5 @@
Ctrl+Enter
Unstage Hunk
1
cs4hs
No newline at end of file
1
cs4hs
2
3
Test project
4
5
This is a project used to experiment with
•
Git and GitHub.
Unstaged Changes
hello.py
Staged Changes
README.md
See All Staged Changes
Update readme
Commit to main
Initial commit
ed Changes: README.md
main
Fetch

Stage and commit the new file

Use the same process, this time for "hello.py"


Edit View Selection Find Packages Help
Project
Unstaged Changes: hello.py
Git
ch4hs
.git
hello.py
README.md
Unstaged Changes for hello.py
Jump To File
Stage File
@@ -0,0 +1,6 @@
Ctrl+Enter
Stage Hunk
1 print("Kia ora koutou,")
2 print("")
3 print("This is a simple python program.")
4 print("What does it do? Just prints out this
• message.")
5 print("")
6 print("Ngā mihi, Evan & Rebecca")
ch4hs
Unstaged Changes
hello.py
Staged Changes
No changes
See All Staged Changes
Commit message
Commit to main
Update readme
Initial commit
taged Changes: hello.py
main
Push 1


Edit View Selection Find Packages Help
Project
Unstaged Changes: hello.py
Git
ch4hs
.git
hello.py
README.md
ch4hs
Unstaged Changes
No changes
Staged Changes
hello.py
No changes to display
See All Staged Changes
Add 'hello' program
Commit to main
Update readme
Initial commit
taged Changes: hello.py
main
Push 1

Local commit history

- You now have a series of changes
- This is your project's "commit history"
- It tells you who changed what, how, and when
- You can "revert" to previous versions

5. "Push" the changes

Now you can publish these new commits on GitHub

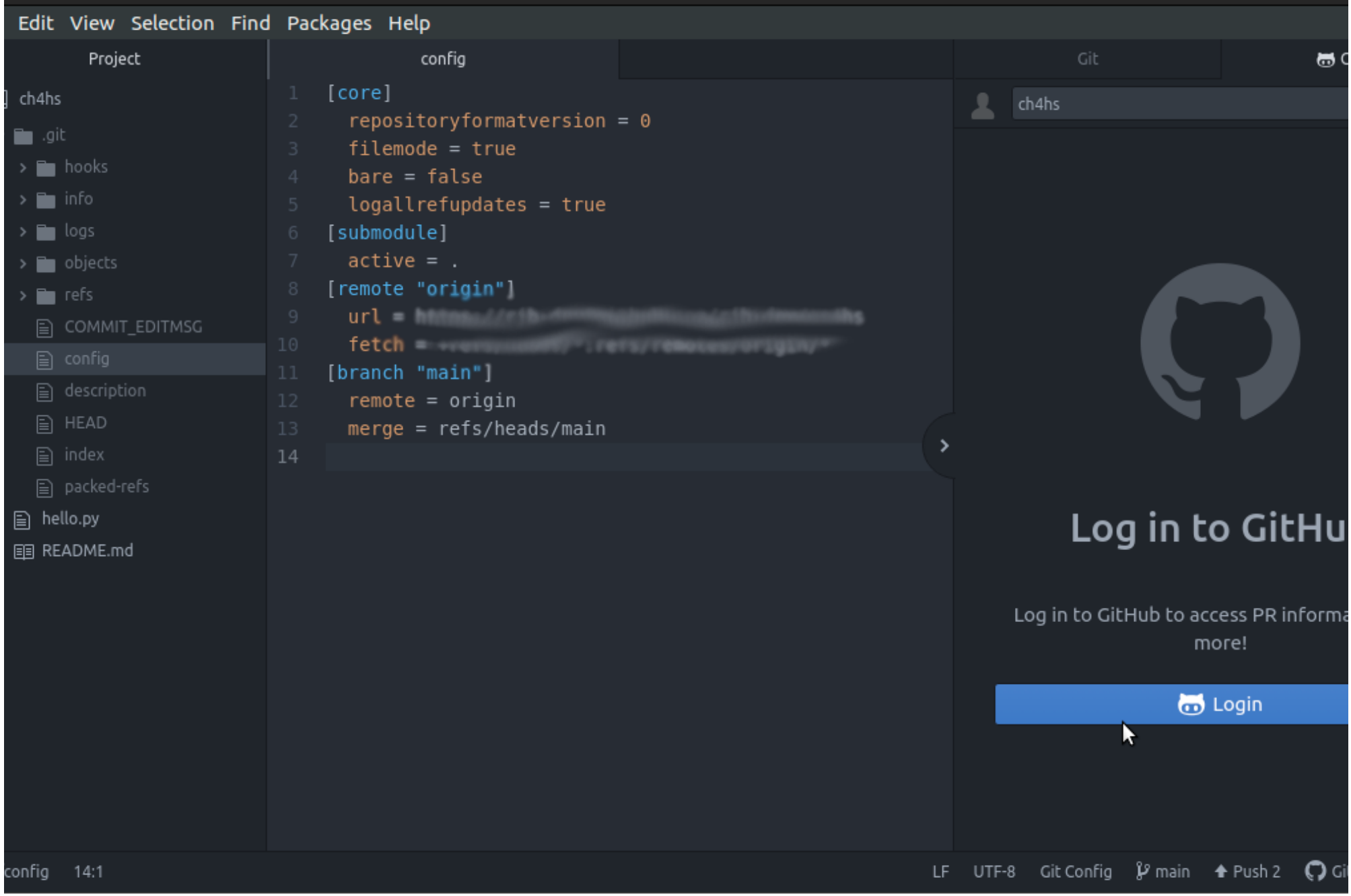
Log in to GitHub

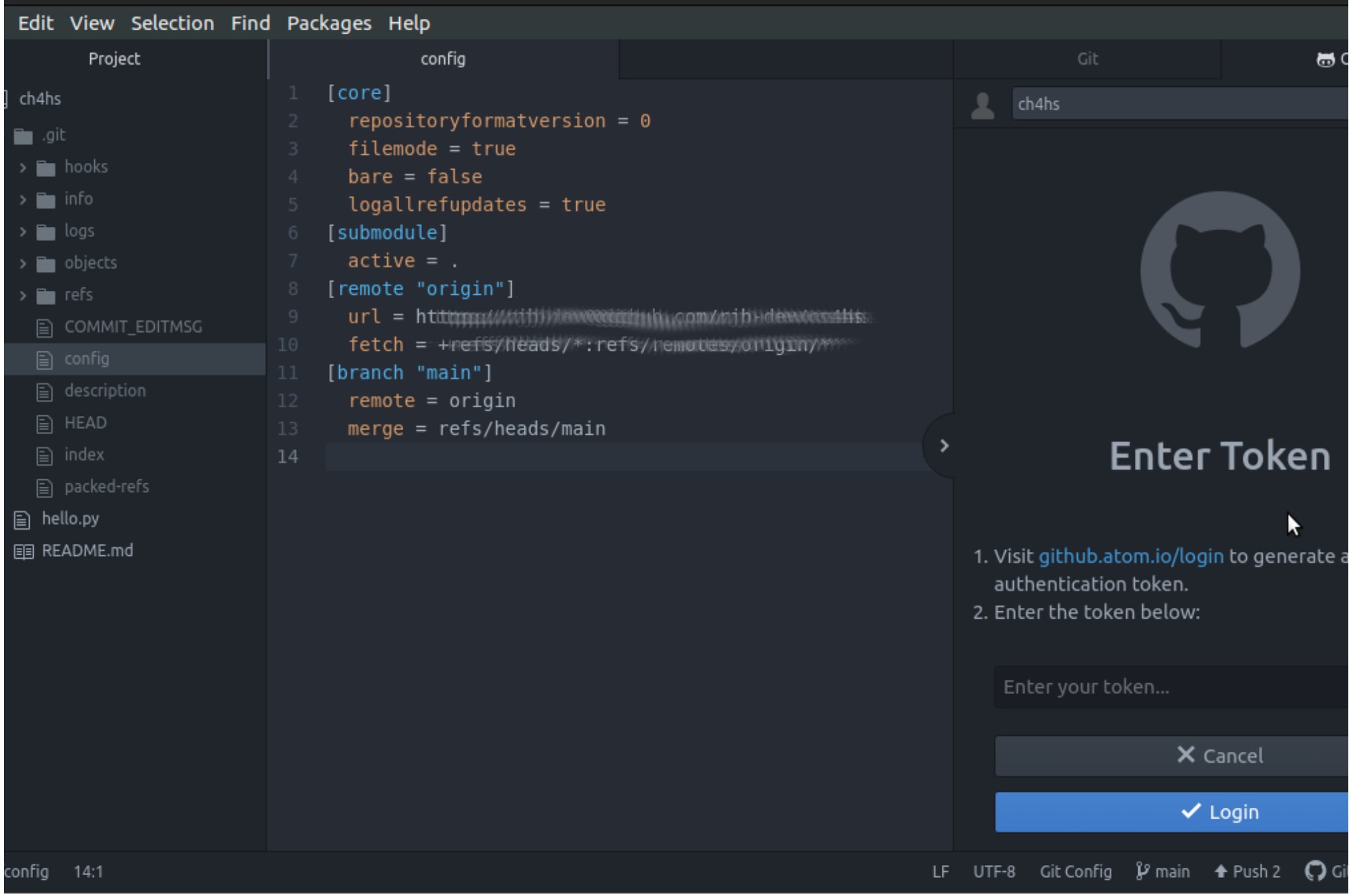
- Edit file ".git/config"
- Add your username to the cloned URL

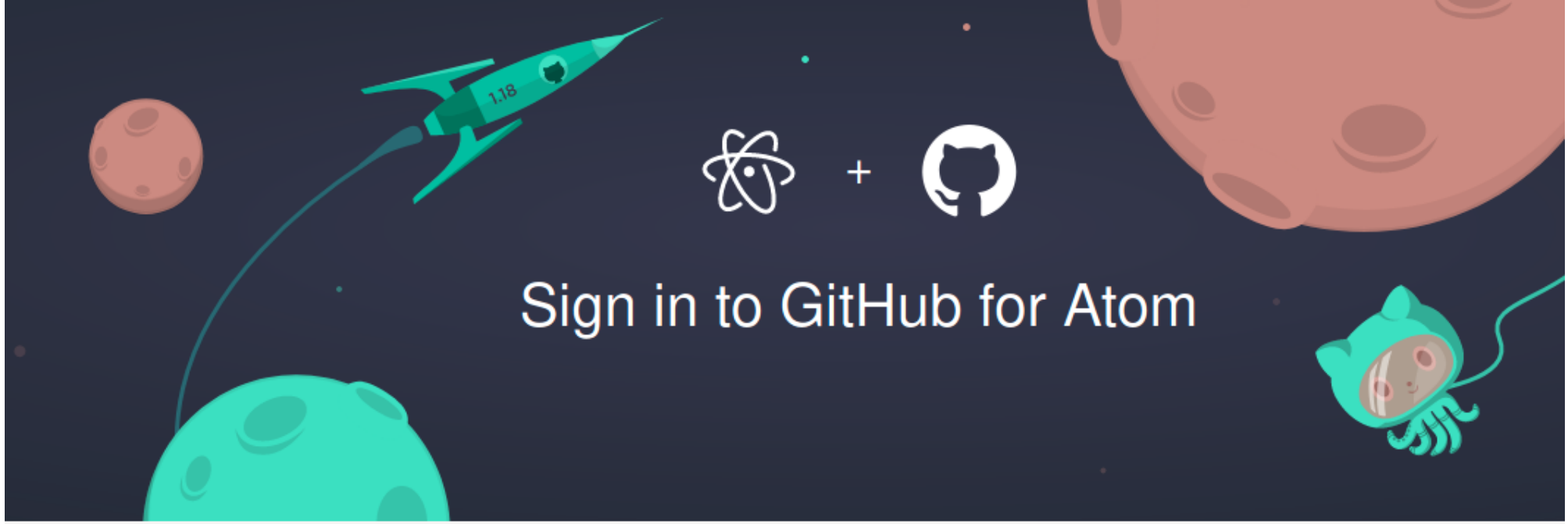
```
[remote "origin"]
  url = https://<username>@github.com/<username>/cs4hs.git
```

Push your commits

- Open the "GitHub" pane
- Click **Login**
- Follow the link to https://github.atom.io/login
- Copy the generated token
- Paste it into Atom under **Enter Token**
- Click **Login**


Edit View Selection Find Packages Help
Project
ch4hs
.git
hooks
info
logs
objects
refs
COMMIT_EDITMSG
config
description
HEAD
index
packed-refs
hello.py
README.md
config
1 [core]
2 repositoryformatversion = 0
3 filemode = true
4 bare = false
5 logallrefupdates = true
6 [submodule]
7 active = .
8 [remote "origin"]
9 url =
10 fetch =
11 [branch "main"]
12 remote = origin
13 merge = refs/heads/main
14
Git
ch4hs
Log in to GitHu
Log in to GitHub to access PR informa
more!
Login
config 14:1
LF UTF-8 Git Config main Push 2


Edit View Selection Find Packages Help
Project
ch4hs
.git
hooks
info
logs
objects
refs
COMMIT_EDITMSG
config
description
HEAD
index
packed-refs
hello.py
README.md
config
1 [core]
2 repositoryformatversion = 0
3 filemode = true
4 bare = false
5 logallrefupdates = true
6 [submodule]
7 active = .
8 [remote "origin"]
9 url = [redacted]
10 fetch = [redacted]
11 [branch "main"]
12 remote = origin
13 merge = refs/heads/main
14
Git
ch4hs
Enter Token
1. Visit github.atom.io/login to generate a authentication token.
2. Enter the token below:
Enter your token...
Cancel
Login
config 14:1
LF UTF-8 Git Config main Push 2



Your GitHub token

[redacted] Copy token

To finish signing in, copy your token and paste it into the sign-in form in Atom.

<> with ♥ by **GitHub**

Push your commits

- Open the "Git" pane
- Click **Push 2** on the bottom right in Atom
- Visit GitHub to view your published changes

Review

1. Created a project (on GitHub)
2. Cloned the project (onto your computer)
3. Changed some files
4. Committed the changes
5. Pushed the changes (to GitHub)

Next Steps

Reviewing and modifying commits

Reviewing changes

Your project now has a "history" you can review

Your project's history

- A series of commits
- Every commit has:
  1. A message
  2. An author
  3. A date
  4. A set of changes
  5. A "parent" commit
  6. A unique ID
     - This is also known as its "hash" or "SHA1"

main

rjb-dev committed 28 minutes ago 2 3

1 parent b0c73a6 5 commit c6faf289047362810f770861d6f7366af52 6

howing **1 changed file** with **6 additions** and **0 deletions**.

Unified

6 ■■■■■ hello.py 4

```
@@ -0,0 +1,6 @@
+ print("Kia ora koutou,")
+ print("")
+ print("This is a simple python program.")
+ print("What does it do? Just prints out this message.")
+ print("")
+ print("Ngā mihi, Evan & Rebecca")
```

mments on commit `c6faf28`

Lock conv

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Comment on this com

Fix-ups

What to do when you've made a mistake?

Undo a commit

- Open the "Git" pane
- Click **Undo** on the most recent commit
  - The commit is discarded
  - Its changes are moved back into the staging area
  - Its message is preserved

Fix a commit

- Edit "hello.py" and add two lines at the end

```
print("Kia ora koutou,")
print("")
print("This is a simple python program.")
print("What does it do? Just prints out this message.")
print("")
print("Ngā mihi, Evan & Rebecca")
print("")
print("PS. I'm learning Git")
```

  - Save the file
    - The new changes are now *unstaged*

Fix a commit

- Stage the new changes
- Commit the result
  - You have just replaced the old commit with a new one

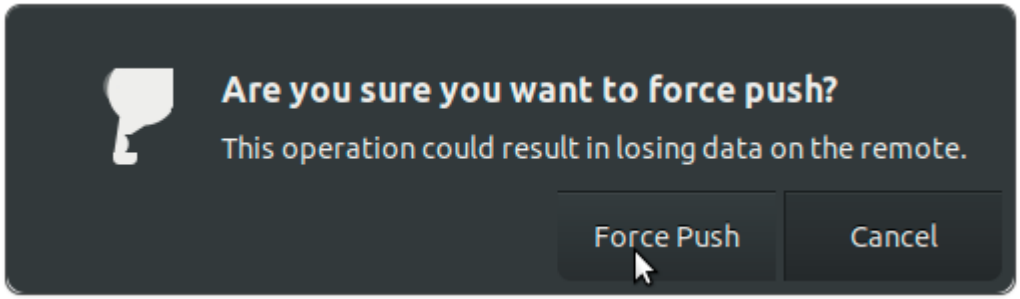
Push the commit

- Hover over the button you used to push
  - What does it say now?

Push the commit

- To push these commits you must "force push"
- **Right-Click** and choose **Force Push**

You will see a warning message as this overwrites data on the remote repository!



Branches

Isolating and sharing specific sets of changes

Branches

- Each "branch" indicates a separate version
- Multiple branches in a single repository
- Let you work without interfering with others
- Eventually combined or "merged"
- The default branch is called "main"

Overview

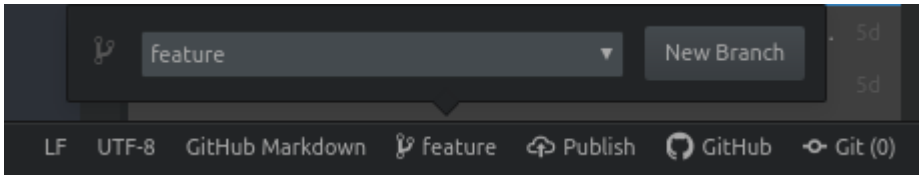
1. Create a new **branch**
2. **Commit** to the branch
3. **Push** the branch (to GitHub)
4. Create a **pull request** (on GitHub)
5. **Accept** the pull request (on GitHub)

1. Create a new branch

Starts a new set of changes

Create a branch

- Open the "Git" pane
- Click **main**
  - Click **New Branch**
  - Type **feature1**
  - Click **New Branch** (again)
- You have a new "feature1" branch



2. Commit to the branch

New commits are added to the active branch

Create a new commit

- Edit the file "README.md"

```
# cs4hs

## Test project

This is a project used to experiment with Git and GitHub.

I changed this line in the 'feature1' branch
```

- Save and commit the change as usual

Isolated changes

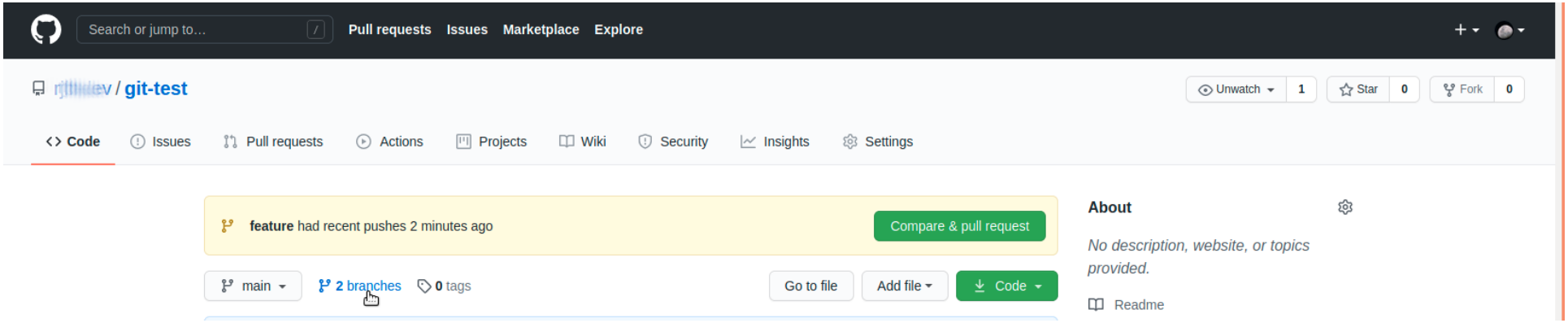
- Click **feature1** to open the branch picker
  - Click **main** to switch branches
- README.md has changed back
  - The edit has been made to the **feature1** branch
  - The **main** branch remains unchanged

3. Push the branch

Send the new branch to GitHub

Push the branch

- Open the "Git" pane
- Switch back to **feature1**
- Click **Publish**
- Visit your project on GitHub
  - The **feature1** branch has been pushed



4. Create a pull request

Request your branch to be "merged"

Request a merge

- Click **Compare & pull request**
- Enter some information
- Click **Create pull request**

Issues Pull requests Actions Projects Wiki Security Insights Settings

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: main ← compare: feature ✓ **Able to merge.** These branches can be automatically merged.

New feature

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

5. Accept the pull request

Merge the "feature1" branch into "main"

Merge the branch

- Make sure you're happy with the changes!
- Click **Merge pull request**
  - Add any information you'd like to include
  - Click **Confirm merge**
- Click **Delete branch**
  - Removes it from the remote repository

New feature #2

Edit | Open with

Open | rjjjjjjev wants to merge 1 commit into `main` from `feature`

Conversation 0 | Commits 1 | Checks 0 | Files changed 1 | +1 −4

rjb-dev commented now — Owner

No description provided.

New feature 9a2f0ff

Add more commits by pushing to the **`feature`** branch on rjjjjjjev/**git-test**.

Continuous integration has not been set up

GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request | or view command line instructions.

Write | Preview

Leave a comment

Reviewers

No reviews

Still in progress? Convert to draft

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked issues

Successfully merging this pull request may close these issues.

None yet

Notifications Customize

Review the new commits

- The two branches have been combined
- There are two new commits:
  1. Your own
  2. A "merge commit"
- The combination happened on GitHub
  - GitHub is the *remote* repository
  - Your *local* repository is now out of date

Update your local repo

- Back in Atom, open the "Git" pane
- Open the branch picker
  - Switch back to **main**
- Click **Fetch**
- Click **Pull 2**

Update your local repo

- The two new commits are pulled from GitHub
- You are back up to date with "origin"

Review

1. Created a new branch
2. Committed to the branch
3. Pushed the branch (to GitHub)
4. Created a pull request (on GitHub)
5. Accepted the pull request (on GitHub)

GitHub Pages

Publish a static webpage with Github pages

Create a (very) basic website

Add a file called **index.html** to your repo:

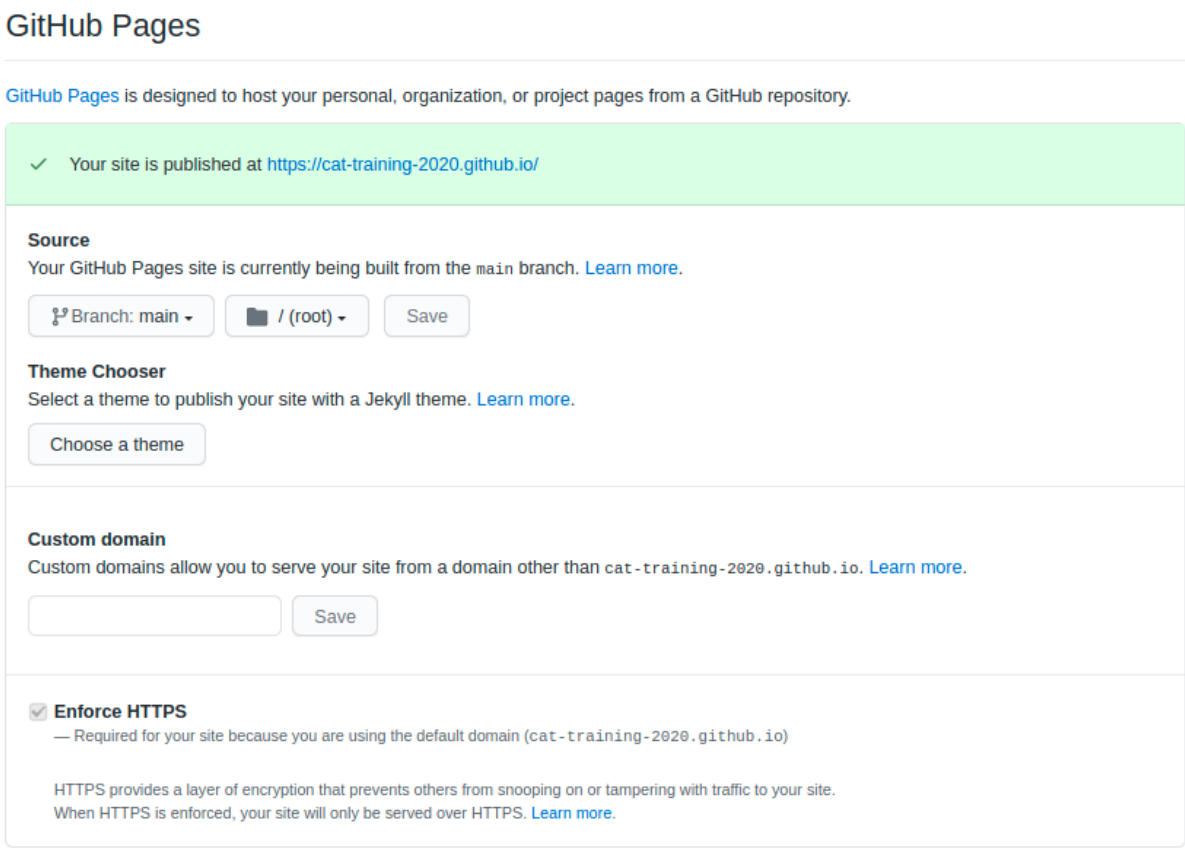
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>My page</title>
  </head>
  <body>
    <p>Here is the content of my page!</p>
  </body>
</html>
```

Deploy your page

- Choose Settings in your GitHub menu
- Under **GitHub Pages** choose the branch and the folder to serve the page from:
  - main
  - root
- Click **Save**
- Soon GitHub will deploy your page at `https://<username>.github.io/<repo-name>`

Review settings

We put our slides on GitHub pages:



View your page

- Visit the url
- You will see:
  Here is the content of my page!
- To start with more functionality, use a framework like **Jekyll**
- Read more here: https://docs.github.com/en/free-pro-team@latest/github/working-with-github-pages

Merge Conflicts

When two branches can't be combined safely

Overview

1. Create two **incompatible commits**
   - One locally
   - One on GitHub
2. **Pull** the remote change (from GitHub)
3. **Resolve** the resulting conflict

1. Create two commits

Changing the same line but with different content

Edit a file locally

- Open "README.md" and change the last line

```
# cs4hs

## Test project

This is a project used to experiment with Git and GitHub.

This is some important information about the project.
```

  - Make it read "important information"
  - Commit the result

Edit a file remotely

- Visit your repository on GitHub
- Click the **pencil icon** on README.md

```
# cs4hs

## Test project

This is a project used to experiment with Git and GitHub.

This is some unimportant information about the project.
```

- Make it read "unimportant information"
- Enter a commit message
- Click **Commit changes**

2. Pull the remote change

Try (and fail) to combine the two branches

Fetch changes

- **Right-click** the **Push 1** button
- Click **Fetch**
  - This retrieves the new commits
- Click **Pull 1**
  - This attempts to merge the changes
- You should hit **Merge conflicts**

3. Resolve the conflict

Indicate how the two changes should be combined

Choose a version

- The two different versions are coloured
- Click **Use me** for the version that is "correct"
- You can edit the text if neither one is right

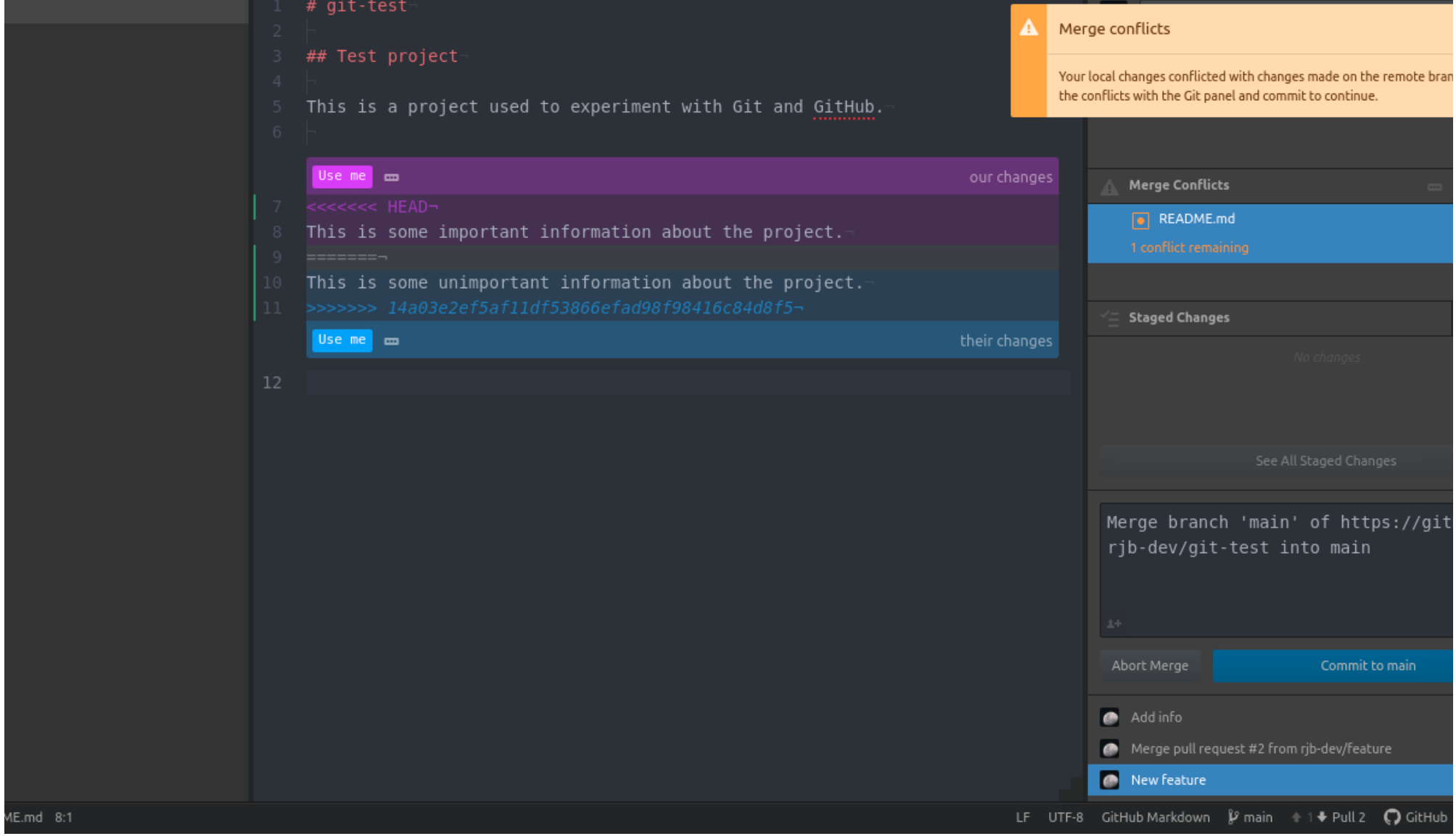
```
# cs4hs

## Test project

This is a project used to experiment with Git and GitHub.

This is some potentially important information about the project.
```

- Remember to save the file when finished


Merge conflicts
Your local changes conflicted with changes made on the remote bran
the conflicts with the Git panel and commit to continue.
git-test
Test project
This is a project used to experiment with Git and GitHub.
Use me
our changes
<<<<<<< HEAD
This is some important information about the project.
=======
This is some unimportant information about the project.
>>>>>>> 14a03e2ef5af11df53866efad98f98416c84d8f5
Use me
their changes
Merge Conflicts
README.md
1 conflict remaining
Staged Changes
No changes
See All Staged Changes
Merge branch 'main' of https://git
rjb-dev/git-test into main
Abort Merge
Commit to main
Add info
Merge pull request #2 from rjb-dev/feature
New feature
ME.md 8:1
LF
UTF-8
GitHub Markdown
main
1 Pull 2
GitHub

Commit the result

- Stage the file
- Click **Commit to main**
- The conflict is resolved locally

Share the result

- We still need to push the resolution
- Click **Push 2**
- Now everything is back in sync

Review

1. Created two **incompatible commits**
2. **Pulled** to combine the two
    - Encountered a merge conflict
3. **Resolved** the conflict

Questions?

Command Line

Using the Git program from the terminal

Open a Terminal

- This depends on your Operating System
  - Windows: Git Bash
  - macOS: Terminal
  - Linux: Terminal

Using the Terminal

- Quick command line how-to

The Git Command

- The **git** command (lower case) runs Git
- Enter **`git --version`** to see whether it works

Configuring Git

- `git config core.editor "atom --wait"`
- `git config user.name "Rebecca Blundell"`
- `git config user.email "rebeccablundell@catalyst.net.nz"`

Cloning a project

- `git clone <url>`
- `cd cs4hs`
- `git status`

Making commits

- `atom README.md`
- `git status`
- `git add README.md`
- `git commit -m "message goes here"`

Reviewing history

- `git log`
- `git show`
- `git show <commit>`

Fixing commits

- `git reset <commit>`
- `git reset --hard <commit>`
- `git add`
- `git commit`

Branching

- `git branch`
- `git branch <name>`
- `git checkout <name>`

Questions?

Some Git terms

- **Repo**: Repository, your git project
- **Clone**: Copy a repo
- **Remote**: The remote or cloud computer
- **Local**: Your computer
- **Fork**: Make changes to a copied repo
- **Rebase**: Similar to **Merge** - combine changes

More git terms

- **Feature**: Branch dedicated to specific feature
- **Pull Request/Merge Request**: Ask to combine your changes into a someone else's remote repo
- **Squash**: Combining many commits into one
- **Hotfix**: Branch dedicated to a single bug
- **CI**: Testing new commits automatically
- Others?

Thanks!

- Evan Giles & Rebecca Blundell
- evanh@catalyst.net.nz
- rebeccablundell@catalyst.net.nz
- Slides will be emailed out
- Don't hesitate to contact us!

freedom to innovate