

GitHub

SURP 2023 Python Bootcamp
Ohio State Astronomy
Slides by: James W. Johnson

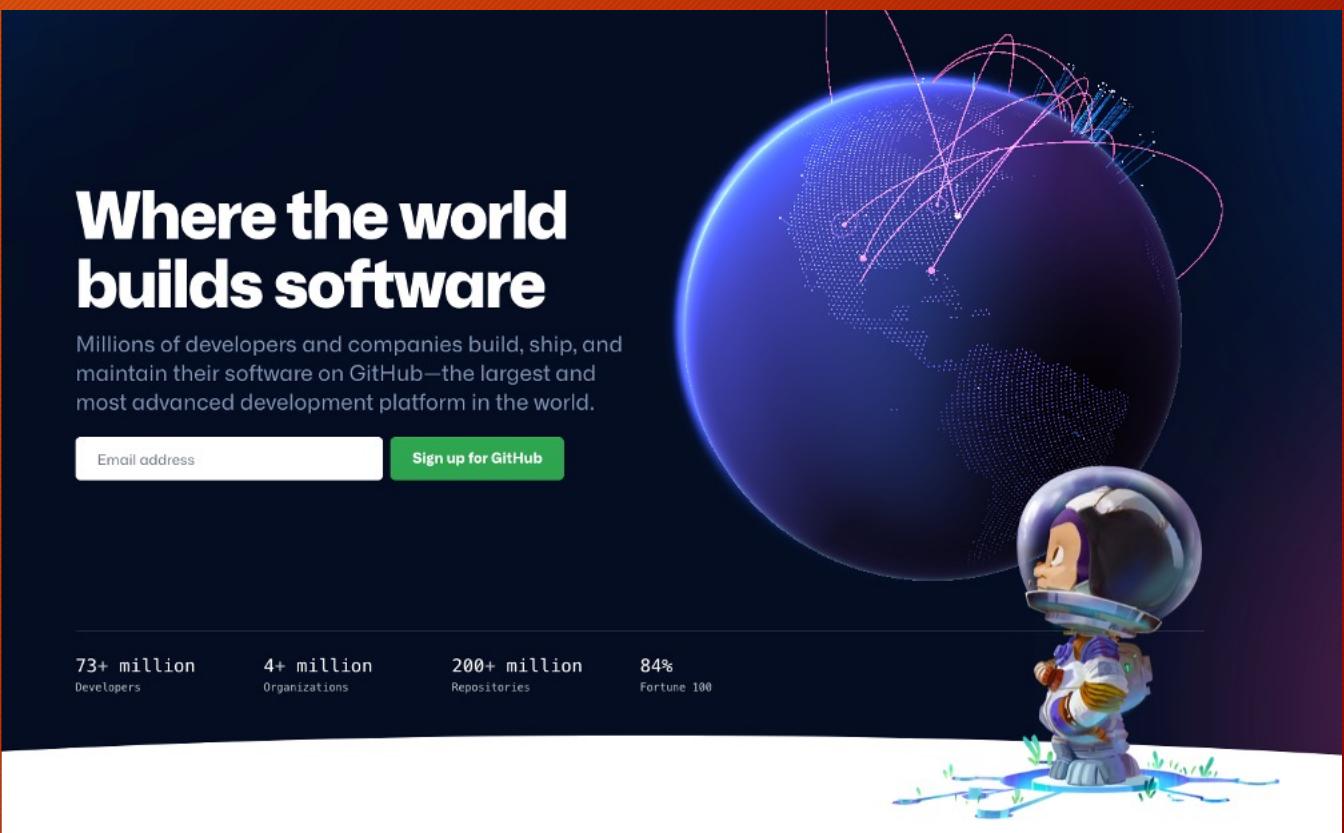
What is GitHub?

A website for hosting files in the cloud

More fundamentally: version control

- Tracks changes to your files over time
- Can host multiple versions of the same file(s) under the same name(s)

Free to use, with some features enabled with a premium account



The image shows the GitHub homepage. At the top, the text "Where the world builds software" is displayed in large white letters. Below it, a subtitle reads: "Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world." There is a input field for "Email address" and a green "Sign up for GitHub" button. At the bottom, there are four statistics: "73+ million Developers", "4+ million Organizations", "200+ million Repositories", and "84% Fortune 100". To the right of the stats is a large blue globe with glowing lines representing network connections. A small, colorful cartoon character wearing a space helmet stands at the bottom right, surrounded by stylized water and plants.

Where the world
builds software

Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.

Email address [Sign up for GitHub](#)

73+ million Developers

4+ million Organizations

200+ million Repositories

84% Fortune 100

What can you do with GitHub?

For researchers:

- Share files with collaborators
- Manage copies between multiple computers
- Backup copy of your work

For software developers:

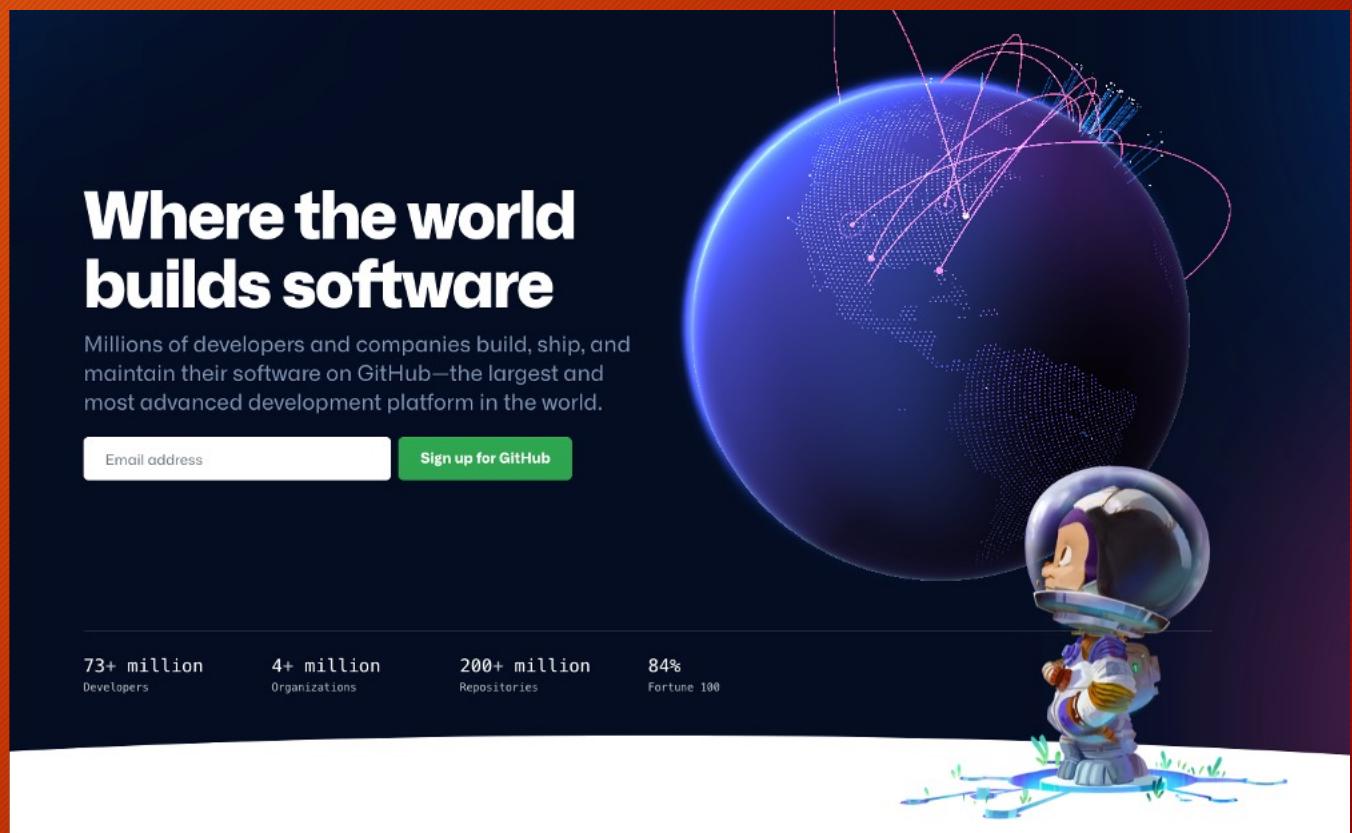
- Track changes to code over time
- Collaborate with other developers seamlessly

The image shows the GitHub homepage. At the top, the text "Where the world builds software" is displayed in large white letters. Below it, a subtitle reads: "Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world." There is a input field for "Email address" and a green "Sign up for GitHub" button. At the bottom, there are four statistics: "73+ million Developers", "4+ million Organizations", "200+ million Repositories", and "84% Fortune 100". To the right of the stats is a large blue globe with glowing purple lines representing network connections. A small, colorful cartoon character wearing a space helmet stands at the bottom right, surrounded by stylized water and plants.

Why learn it now?

A majority of astronomers and physicists eventually leave academia for data science or other private sector jobs

In these positions, knowledge of version control tools like GitHub is *expected*.



The image shows the GitHub homepage. At the top, the tagline "Where the world builds software" is displayed in large white font. Below it, a sub-tagline reads: "Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world." There is a input field for "Email address" and a green "Sign up for GitHub" button. At the bottom, there are four statistics: "73+ million Developers", "4+ million Organizations", "200+ million Repositories", and "84% Fortune 100". To the right of the stats is a large blue globe with glowing purple lines representing network connections. A cartoon astronaut in a space suit stands at the bottom right, looking up at the globe.

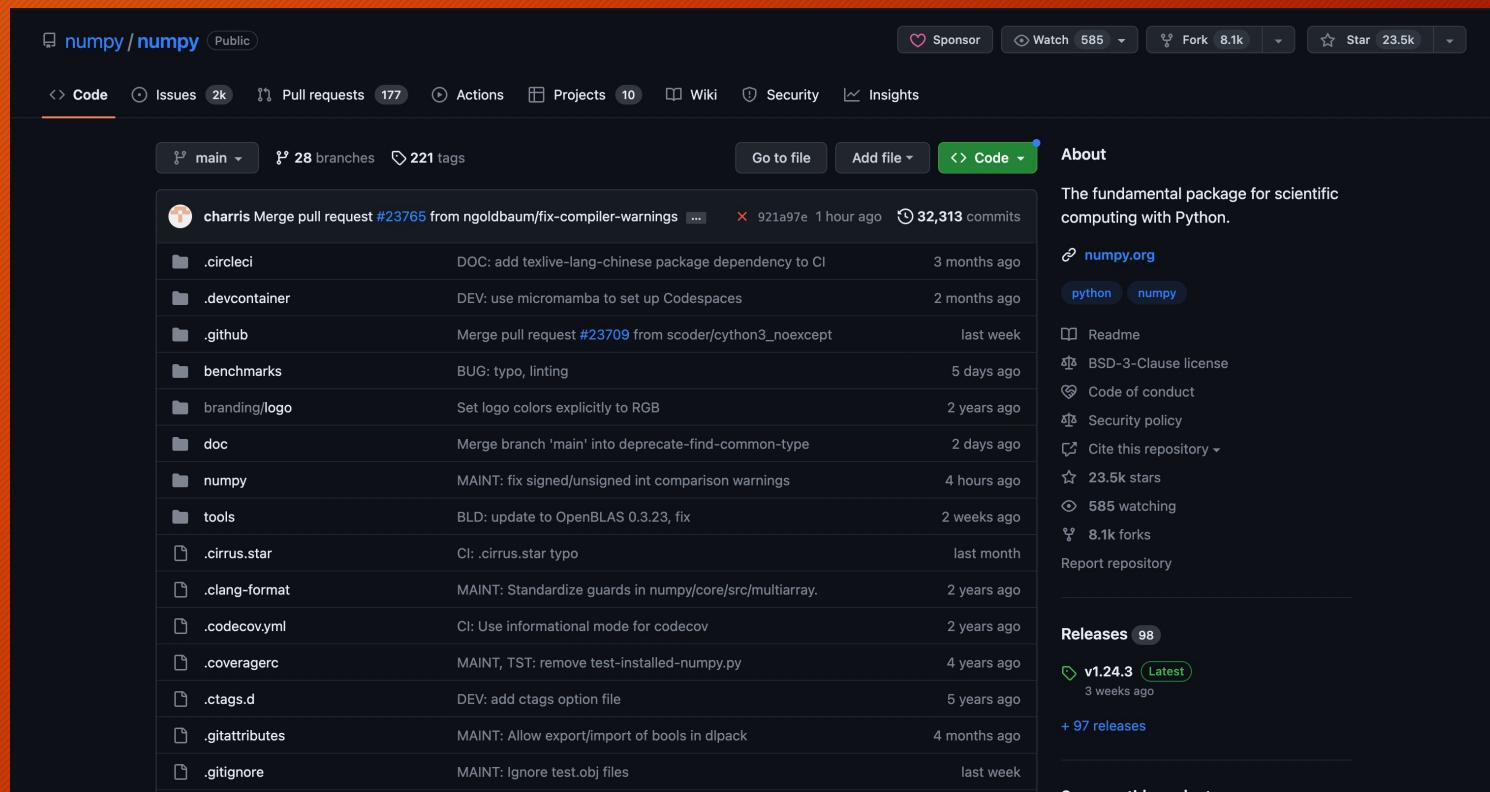
Repositories

A directory containing files and other directories that is stored on GitHub

Example: NumPy, SciPy, and matplotlib each host their source code on GitHub

Files can be anything

- Code (including jupyter notebooks)
- Documents
- Images



Repositories: Local vs. Remote

Local: this file system *on your computer*

Remote: this file system *on GitHub's website*

Remote will always have the url
format *github.com/<user>/<repo>*

The screenshot shows the GitHub repository page for numpy/numpy. The page has a dark theme. At the top, there are navigation links for Code, Issues (2k), Pull requests (177), Actions, Projects (10), Wiki, Security, and Insights. Below these are buttons for Sponsor, Watch (585), Fork (8.1k), and Star (23.5k). The main content area displays a list of recent commits. Each commit includes the author's profile picture, the commit message, a link to the pull request, the commit hash, the time since the commit, and the number of commits. On the right side, there is a sidebar with sections for About, Readme, BSD-3-Clause license, Code of conduct, Security policy, Cite this repository, 23.5k stars, 585 watching, 8.1k forks, Report repository, Releases (98), v1.24.3 (Latest), and + 97 releases. At the bottom, there is a button for Sponsor this project.

Commit Details	Date	Author
charris Merge pull request #23765 from ngoldbaum/fix-compiler-warnings ...	1 hour ago	921a97e
.circleci DOC: add texlive-lang-chinese package dependency to CI	3 months ago	
.devcontainer DEV: use micromamba to set up Codespaces	2 months ago	
.github Merge pull request #23709 from scoder/cython3_noexcept	last week	
benchmarks BUG: typo, linting	5 days ago	
branding/logo Set logo colors explicitly to RGB	2 years ago	
doc Merge branch 'main' into deprecate-find-common-type	2 days ago	
numpy MAINT: fix signed/unsigned int comparison warnings	4 hours ago	
tools BLD: update to OpenBLAS 0.3.23, fix	2 weeks ago	
.cirrus.star Cl: .cirrus.star typo	last month	
.clang-format MAINT: Standardize guards in numpy/core/src/multiarray.	2 years ago	
.codecov.yml Cl: Use informational mode for codecov	2 years ago	
.coveragerc MAINT, TST: remove test-installed-numpy.py	4 years ago	
.ctags.d DEV: add ctags option file	5 years ago	
.gitattributes MAINT: Allow export/import of bools in dlpack	4 months ago	
.gitignore MAINT: Ignore test.obj files	last week	

Objectives

- Commits: how to make changes to a remote repository
- *git*: terminal application for interacting with a repository on GitHub
- Branching: how to store the same file(s) in different states
- GitHub websites
- Setting up a repository: GitHub itself walks you through this

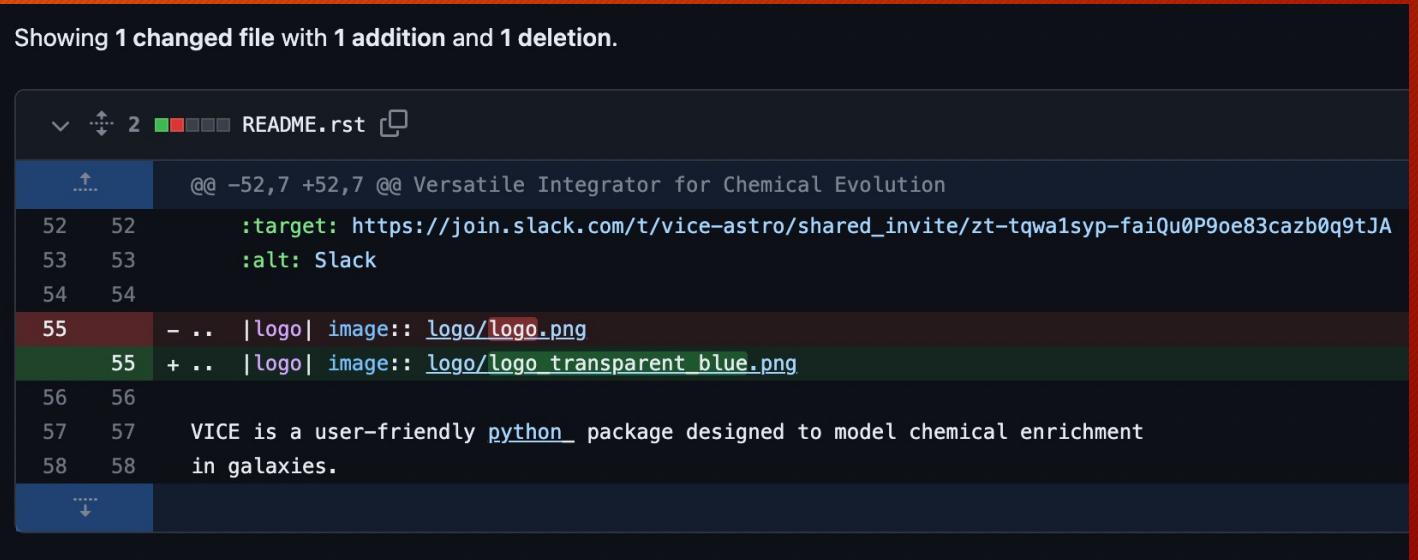
Commits

Bread and butter for making changes
to a repository

A series of +'s and -'s within files

- Current state of repository = sum of all commits across all files

Tracks bit-by-bit changes to binary
files (e.g., PDFs)



The screenshot shows a terminal window displaying a git diff output for the file README.rst. The title bar indicates "Showing 1 changed file with 1 addition and 1 deletion." The diff output shows the following changes:

```
diff --git a/README.rst b/README.rst
@@ -52,7 +52,7 @@ Versatile Integrator for Chemical Evolution
 52 52      :target: https://join.slack.com/t/vice-astro/shared_invite/zt-tqwaisyp-faiQu0P9oe83cazb0q9tJA
 53 53      :alt: Slack
 54 54
 55 - .. |logo| image:: logo/logo.png
 55 + .. |logo| image:: logo/logo_transparent_blue.png
 56 56
 57 57      VICE is a user-friendly python_ package designed to model chemical enrichment
 58 58      in galaxies.
```

Commits

That's about all there is to it!

All of this happens *after* you're ready to upload changes to your files

```
Showing 1 changed file with 1 addition and 1 deletion.

diff --git a/README.rst b/README.rst
@@ -52,7 +52,7 @@ Versatile Integrator for Chemical Evolution
 52 52      :target: https://join.slack.com/t/vice-astro/shared_invite/zt-tqwaisyp-faiQu0P9oe83cazb0q9tJA
 53 53      :alt: Slack
 54 54
 55 - .. |logo| image:: logo/logo.png
 55 + .. |logo| image:: logo/logo_transparent_blue.png
 56 56
 57 57      VICE is a user-friendly python_ package designed to model chemical enrichment
 58 58      in galaxies.
```

git: Managing Repositories

Terminal application for managing a repository, including making commits

- Generally pre-installed
- Uploading: *git add*, *git commit*, *git push*
- Downloading: *git pull*

git --help prints an API reference in the terminal

No connection to GitHub as a website : *git* is an open-source version control program

- Tracks changes in any set of computer files for projects of any size
- Other websites built around *git*: GitLab, Bitbucket

git mv and *git rm*

Both fulfill the same functionality as *mv* and *rm* otherwise do

git mv: rename the file

- A bare *mv* without *git* will make the new version show up as an untracked file
- The file takes its commit history with it
- If you mess up, *mv* followed by *git mv* is fine

git rm: delete the file

- Change will be reflected when others run *git pull*

git status: Which files have updates?

Displays the relative paths to files that

- are new and not yet stored within the repository (i.e., “untracked” files)
- are already stored within the repository, but have local updates not yet added to the next commit
- have local updates that have been added to the next commit

```
(python3.10) jamesjohnson@Binary slides % git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   github.pptx

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ~$github.pptx

no changes added to commit (use "git add" and/or "git commit -a")
(python3.10) jamesjohnson@Binary slides % _
```

The *.gitignore* file

git looks for a plain-text file called *.gitignore* in the top-level directory of all repositories

- Contains paths and glob patterns for files that it should skip

```
1 * .so
2 * .pyc
3 * .pdf
4 * .sublime-workspace
5 * .sublime-project
6
7
```

git add and *git restore*

git add: tells *git* which file(s) to include in the next commit

- Can be as many files as you want/need
- Doesn't have to be *every* file with changes

git restore [--staged]: the “undo” button

```
(python3.10) jamesjohnson@Binary slides % git add github.pptx
(python3.10) jamesjohnson@Binary slides % git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   github.pptx

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    images/github.gitignore.png
    images/github.gitstatus.png
    ~$github.pptx
```

git commit

Creates the commit itself

Command will open the *vi* text-editor and prompt you for a message to attach to the commit

- Press *A* to enter edit mode
- Press *Esc* to enter command mode, then “*:wq*” and *Enter* will save and quit

git commit -m “message here” takes the message in quotes without opening *vi*

```
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
# Your branch is up to date with 'origin/main'.
#
# Changes to be committed:
#       modified:   github.pptx
#
# Changes not staged for commit:
#       modified:   github.pptx
#
# Untracked files:
#       images/github.gitadd.png
#       images/github.gitignore.png
#       images/github.gitstatus.png
#       ~$github.pptx
#
# ~
# ~
# ~
# ~
# ~
# ~
```

git push and *git pull*

git push: uploads the new commit to the remote repository on GitHub

- The first time you do this, it will likely prompt you to run with extra flags to tell it the branch you're pushing to
git push --set-upstream origin main

git pull: downloads new commits from the remote repository

Display remote URL: *git remote -v*

Set remote URL: *git remote set-url origin [url]*

git gud: for when coding is as hard as Dark Souls

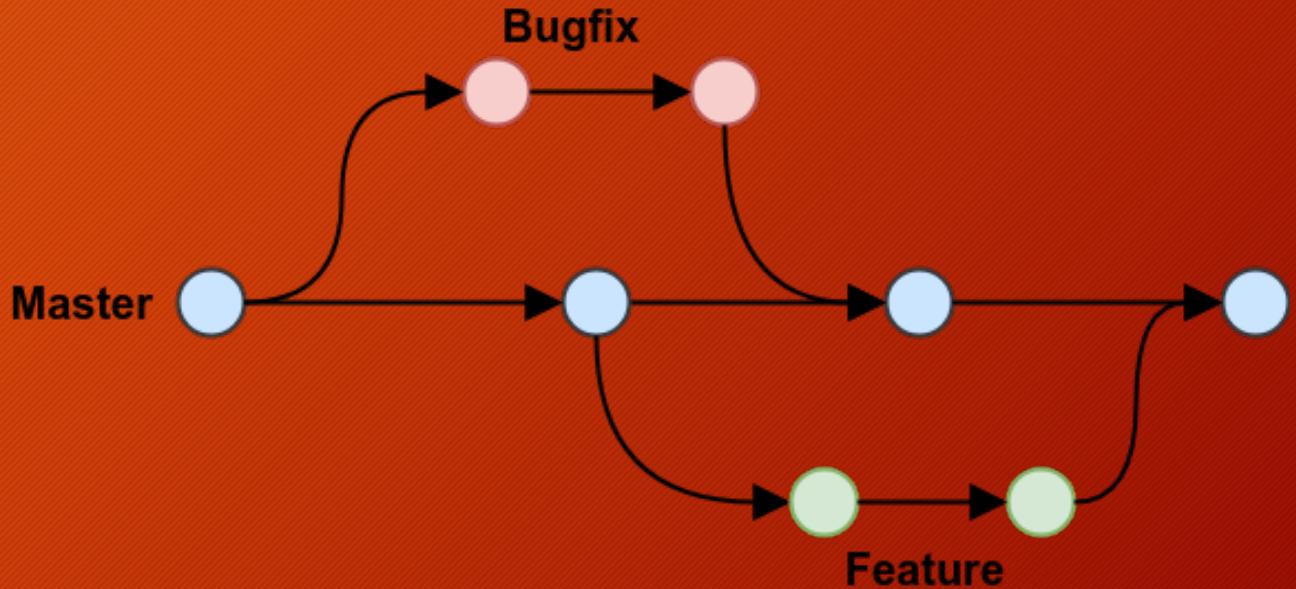


Branches

Different branches track different commit histories

- Current state of file(s) = sum of commits *on a given branch*

Of particular importance if you are developing software that will be distributed/released



Note: default name of the “primary” branch x to be *master* but is now *main*

git branch: Displays which branch you're on

Branching in production-level code:

- Development
- Different features on their own branches
- Recently released versions
- Main branch usually latest release

Branching for researchers:

- Can be used to store the same files in different states
- More useful if your code base gets large

```
(python3.10) jamesjohnson@Binary VICE % git branch
* development
  development-openmp
  development-openmp-modeling
  development-openmp-windowscompat
  main
  v1.1.x
  v1.2.x
  v1.3.x
  v1.3.x-patch
```

git switch

Changes between branches locally

Can also be done with *git checkout*

- *checkout* can also be used to copy file(s) from another branch

Will prompt you to commit your changes if it detects updates in tracked files.

```
(python3.10) jamesjohnson@Binary VICE % git branch
* development
  development-openmp
  development-openmp-modeling
  development-openmp-windowscompat
  main
  v1.1.x
  v1.2.x
  v1.3.x
  v1.3.x-patch
(python3.10) jamesjohnson@Binary VICE % git switch main --quiet
(python3.10) jamesjohnson@Binary VICE % git branch
  development
  development-openmp
  development-openmp-modeling
  development-openmp-windowscompat
* main
  v1.1.x
  v1.2.x
  v1.3.x
  v1.3.x-patch
(python3.10) jamesjohnson@Binary VICE % git checkout development --quiet
(python3.10) jamesjohnson@Binary VICE % git branch
* development
  development-openmp
  development-openmp-modeling
  development-openmp-windowscompat
  main
  v1.1.x
  v1.2.x
  v1.3.x
  v1.3.x-patch
```

git branch: A new branch

git branch <new_branch> will create a new branch with whatever name you give it

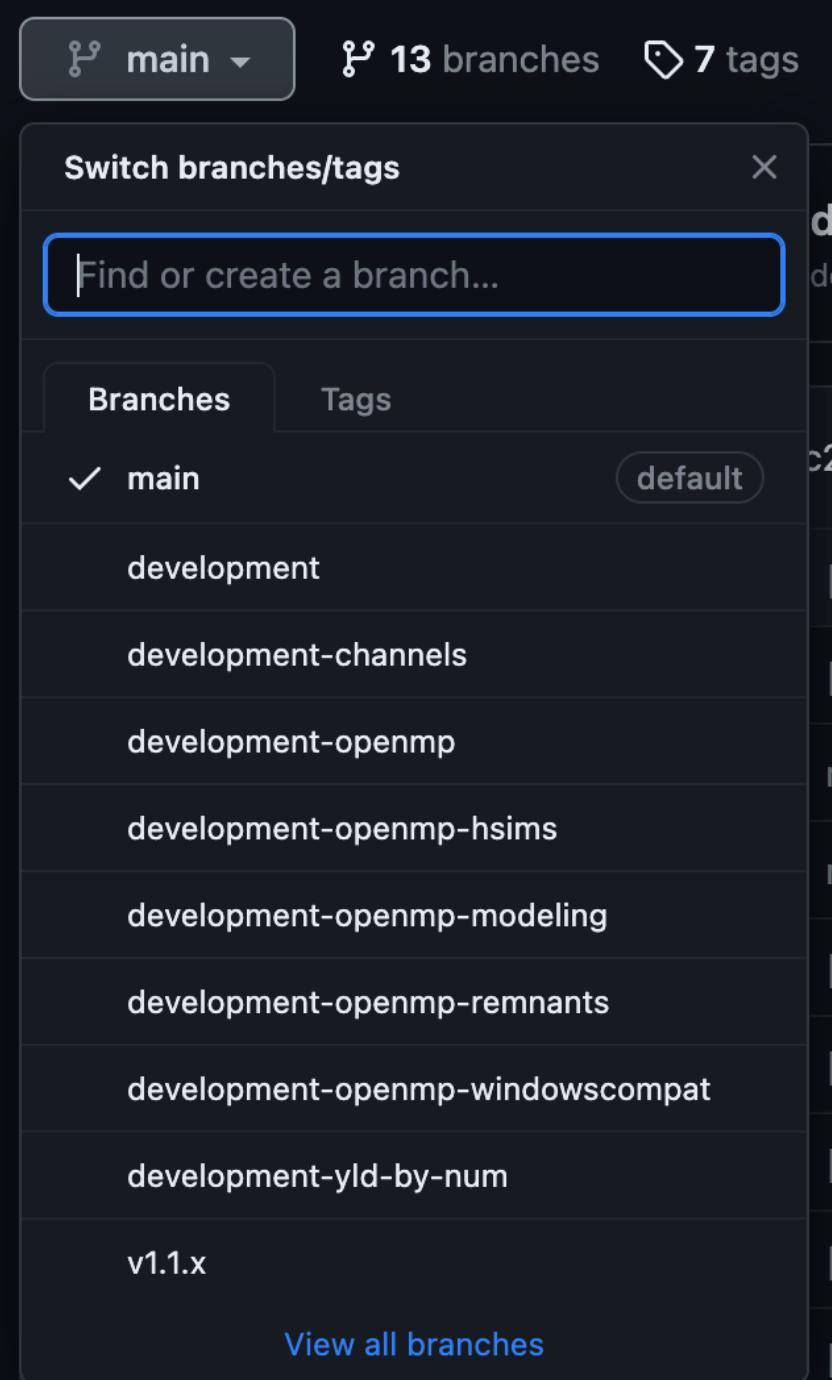
- *git push --set-upstream origin <new_branch>*

You can also create a new branch on GitHub's webpage for your repository, then

git pull

git switch <branch_name>

git branch --delete <branch_name> will delete the local copy of the branch



Merging branches

Copies the commits on one branch into another branch

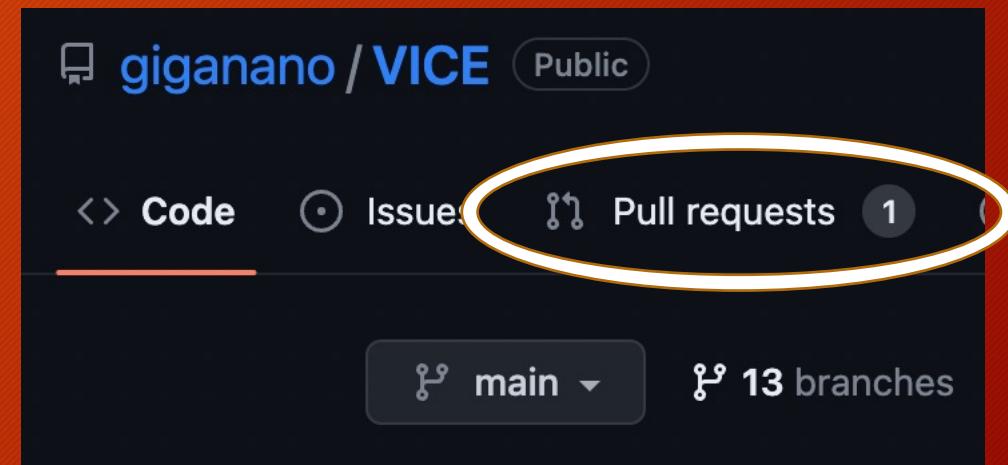
When collaborating, open a *pull request*. Merge will be handled on GitHub's website.

Locally, you can use:

`git switch <into_branch>`

`git merge <from_branch>`

`git push`



How Researchers Benefit from GitHub

At the end of the day, or whenever you need it: *git add + git commit + git push*

Later, or simply on a different computer: *git pull*

- No need to handle transfers yourself!
- The other system *can* be a supercomputer (e.g., Ohio Supercomputer Center)
- If all you're looking for is managing files between systems, then *add*, *commit*, *push*, and *pull* are all you *really* need

You're backing up your work in the process

“Can you send me a copy of your code?”

“Sure, here’s a link that won’t change even if the code changes.”

GitHub Pages: An easy way to deploy websites

GitHub will host *static* websites for free!

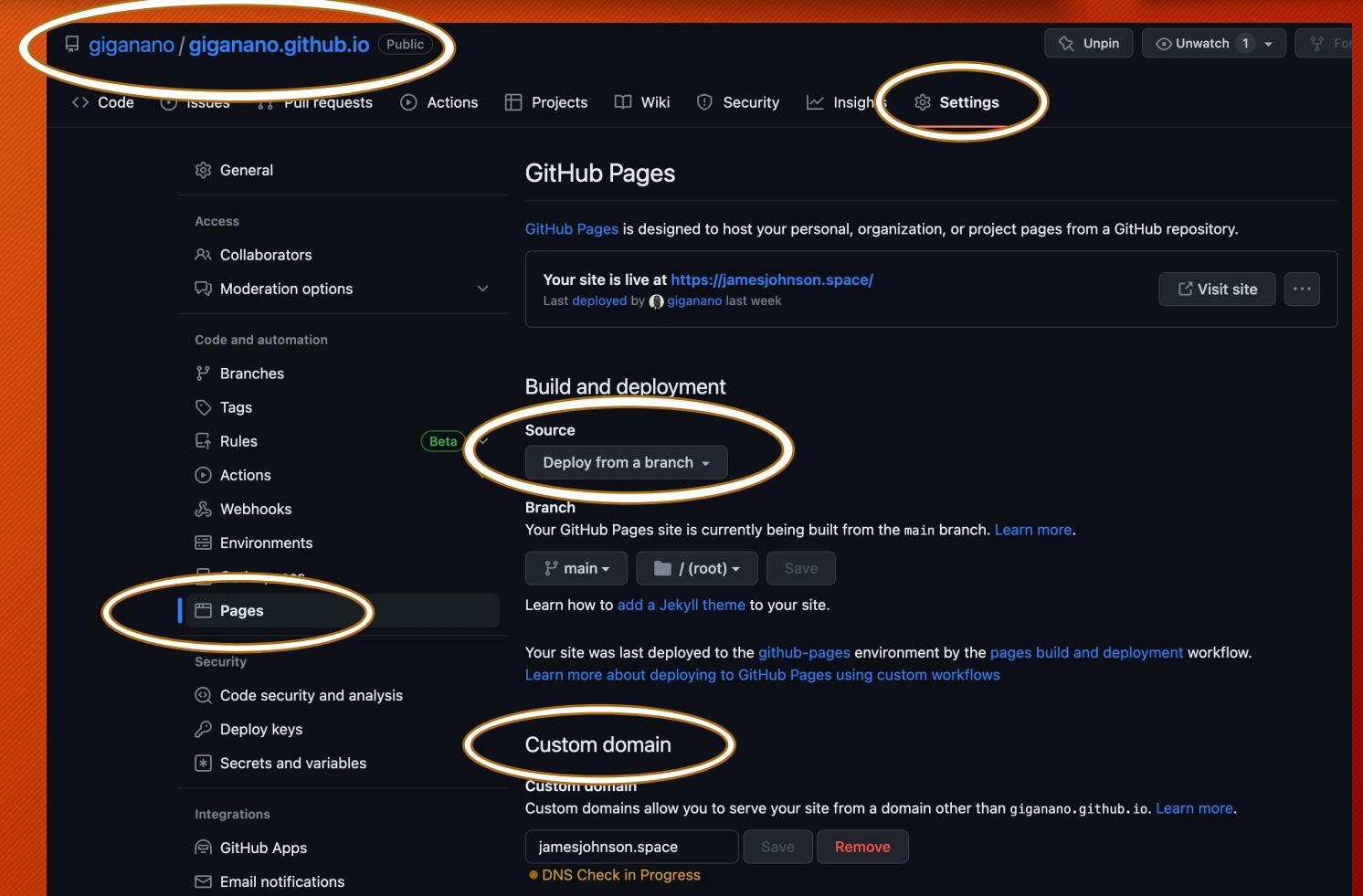
- *Static*: built using HTML/Markdown as opposed to JavaScript (*dynamic*)
- Popular among academics → more freedom than, e.g., google sites
- <https://username.github.io> → URLs of this format are GitHub websites

Can also be deployed at any URL that you own the rights to

- Example: <https://giganano.github.io> redirects to <https://jamesjohnson.space>
- The source material for this bootcamp is hosted on a GitHub website
 - <https://jamesjohnson.space/bootcamp> is really just <https://giganano.github.io/bootcamp>

GitHub Pages: An easy way to deploy websites

In repository, navigate to Settings, then Pages, and tell it which branch to deploy from (usually *main*)



In general, the name of the repo should end with *.github.io*

GitHub Pages: An easy way to deploy websites

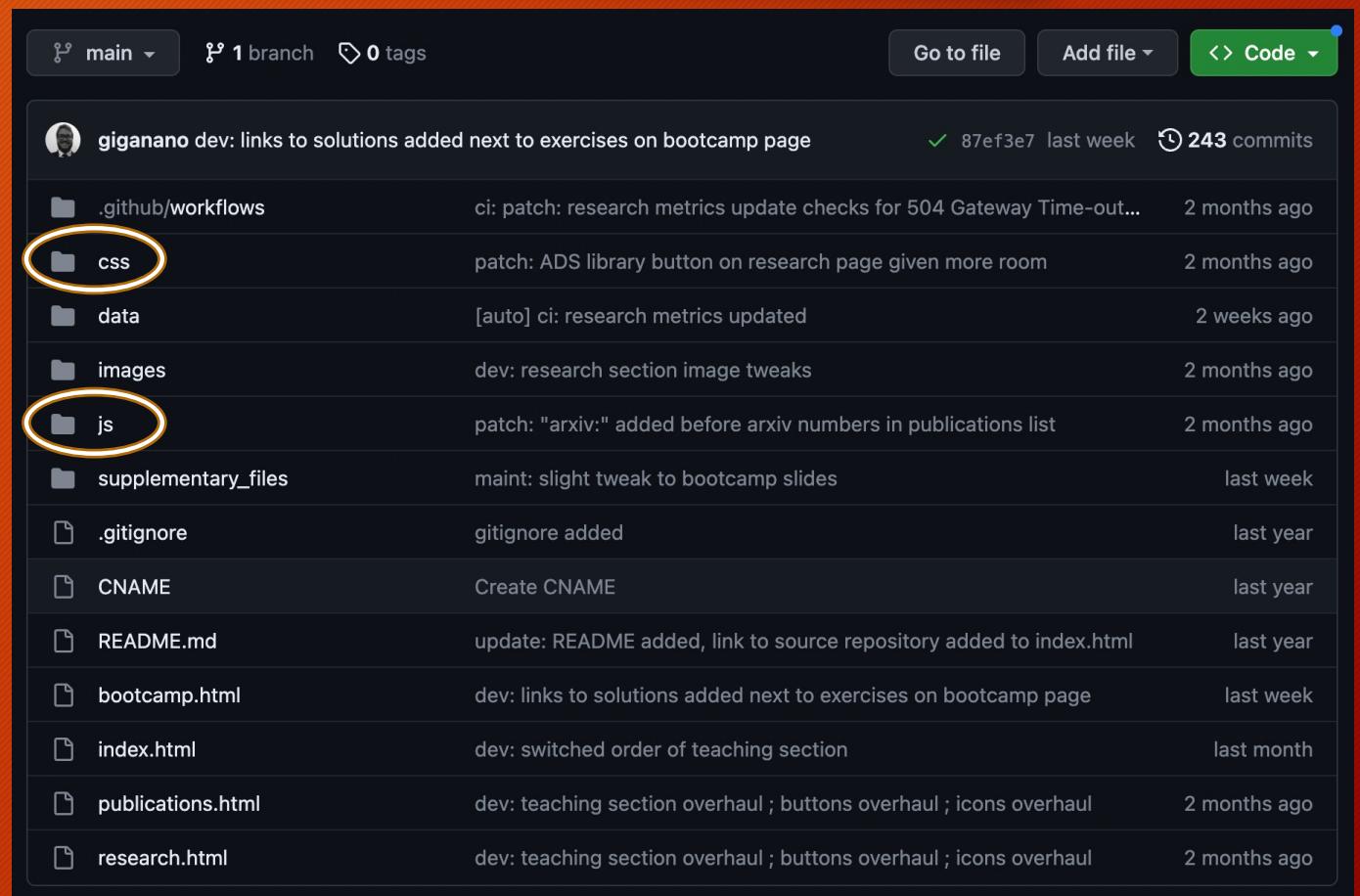
Built using HTML or Markdown

- *index.html* or *index.md* will be the website's front page

Can use your own JavaScript or CSS
(Cascading Style Sheets)

Anatomy of a website:

- HTML = bones
- JavaScript = muscles
- CSS = skin



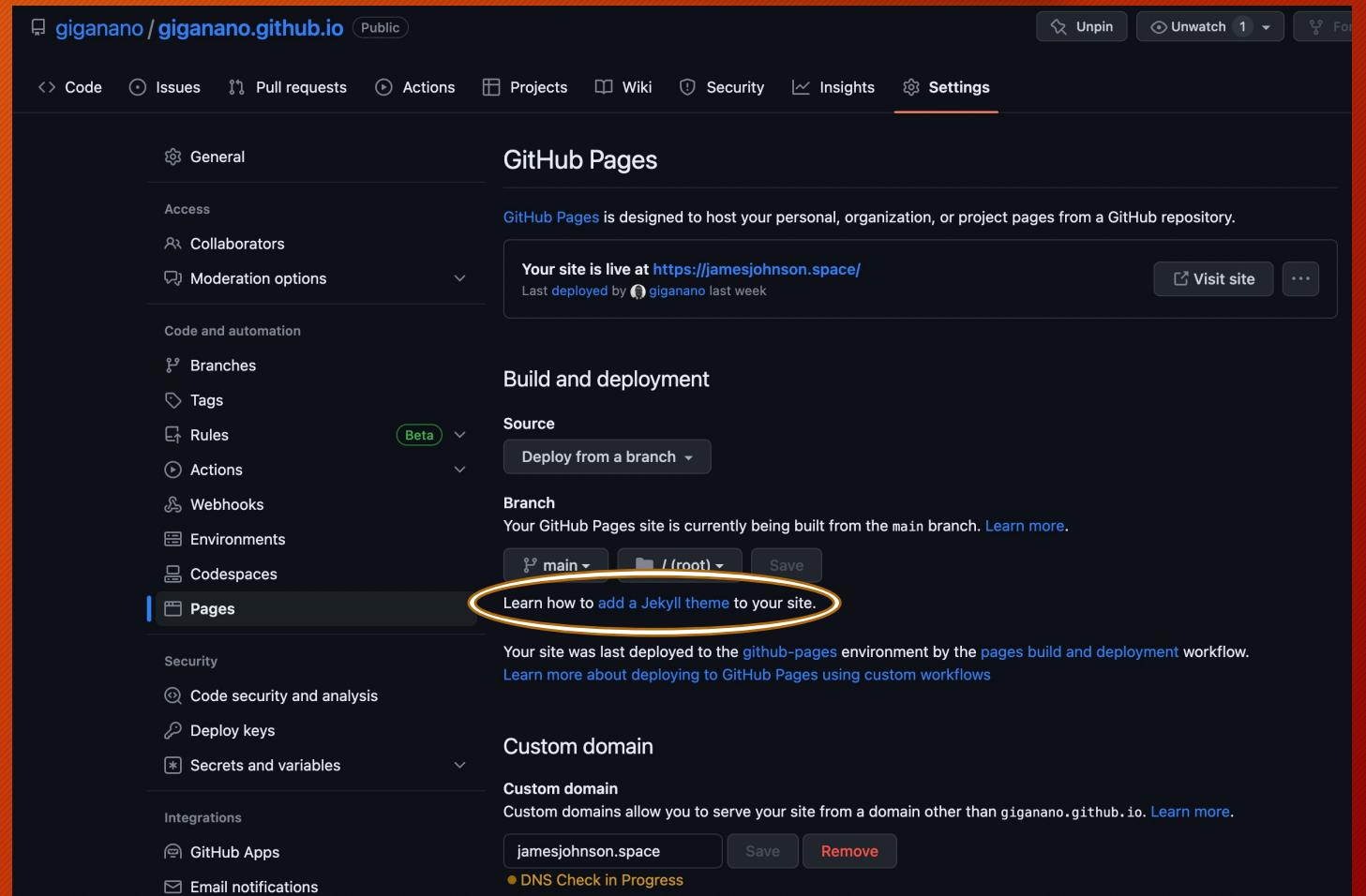
GitHub Pages: An easy way to deploy websites

Built using HTML or Markdown

- *index.html* or *index.md* will be the website's front page

You can simply choose an “off-the-shelf” theme to expedite styling

- Regardless of HTML/Markdown



Creating a repository

Easy if you simply use GitHub's website

The screenshot shows a GitHub user profile for James W. Johnson. On the left is a circular profile picture of a smiling man with glasses and a beard. Below it, the name "James W. Johnson" and the handle "giganano" are displayed. The main area shows two repositories: "outflows" (Public) and "PythonBootcamp" (Public). The "outflows" repository has a description: "an investigation into the impact of galactic outflows on chemical evolution models". It was last updated 4 hours ago via Jupyter Notebook. The "PythonBootcamp" repository has a description: "A Python bootcamp aimed at astronomers and other scientists with a background in functional programming". It was last updated yesterday via Python. At the top, there is a navigation bar with tabs: Overview, Repositories (16), Projects, Packages, Stars (2). Below the tabs is a search bar with placeholder text "Find a repository...". To the right of the search bar are filters for Type, Language, and Sort, followed by a green "New" button with a plus sign icon, which is circled in orange. A small green line graph is visible on the far right.

James W. Johnson
giganano

outflows Public

an investigation into the impact of galactic outflows on chemical evolution models

Jupyter Notebook Updated 4 hours ago

PythonBootcamp Public

A Python bootcamp aimed at astronomers and other scientists with a background in functional programming

Python ⭐ 6 2 Updated yesterday

Overview Repositories 16 Projects Packages Stars 2

Find a repository... Type Language Sort New

Creating a repository

You'll land at a page that looks like this

- Name it
- Private repos require premium

README, *.gitignore*, and *license* can be added at any time

- Leave them out if this is your first repository

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner *



giganano

Repository name *



Great repository names are short and memorable. Need inspiration? How about [animated-octo-eureka](#)?

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:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs](#).

Add *.gitignore*



Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license



A license tells others what they can and can't do with your code. [Learn more about licenses](#).

You are creating a public repository in your personal account.

Create repository

Creating a repository

You'll land at a page that looks like this

- Name it
- Private repos require premium

README, *.gitignore*, and *license* can be added at any time

- Leave them out if this is your first repository

The suggested name is usually so random that it's funny...

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner *



giganano

Repository name *



/

Great repository names are short and memorable. Need inspiration? How about [animated-octo-eureka?](#)

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:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs](#).

Add *.gitignore*

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

You are creating a public repository in your personal account.

[Create repository](#)

Creating a repository

If you leave out the *README*, *.gitignore*, and *license*, then the landing page of an empty repository will walk you through how to make your first push

From here on, everything proceeds according to the earlier slides

The screenshot shows the GitHub repository creation interface. At the top, there are two sections: "Set up GitHub Copilot" and "Invite collaborators". Below these is a "Quick setup" section with options for "Set up in Desktop" (selected), "HTTPS", "SSH", and a URL. It also includes a note about including a *README*, *LICENSE*, and *.gitignore*. A large orange arrow points from the text above to the "...or create a new repository on the command line" section. This section contains a command-line script:

```
echo "# animated-octo-eureka" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/giganano/animated-octo-eureka.git
git push -u origin main
```

Below this is another command-line section for pushing an existing repository:

```
git remote add origin https://github.com/giganano/animated-octo-eureka.git
git branch -M main
git push -u origin main
```

At the bottom, there's a section for importing code from another repository with a "Import code" button.

Thank you!