Workflow with Git, Gitpod & Github

AN INTRODUCTION TO VERSION CONTROL

PLEASE NOTE THAT THIS ZOOM CALL IS BEING RECORDED AND WILL BE POSTED IN SLACK

Git-what???

First let's take a look at all the lingo:

GIT - Git is a version control system (VCS) that makes it easier to track changes to files. so you have a record of what has been done, and you can revert to specific versions should you ever need to. Git also makes collaboration easier, allowing changes by multiple people to all be merged into one source.

REPOSITORY - A repository (or "repo" for short) is a location where all the files for a particular project are stored.

GITHUB - For the User Centric module you'll be using Github to store your repositories.

Github is a free git repository hosting service.

Each project you build will have its own 'repo', and you can access it with a unique URL, which will follow the format of: *https://github.com/username/project-name*

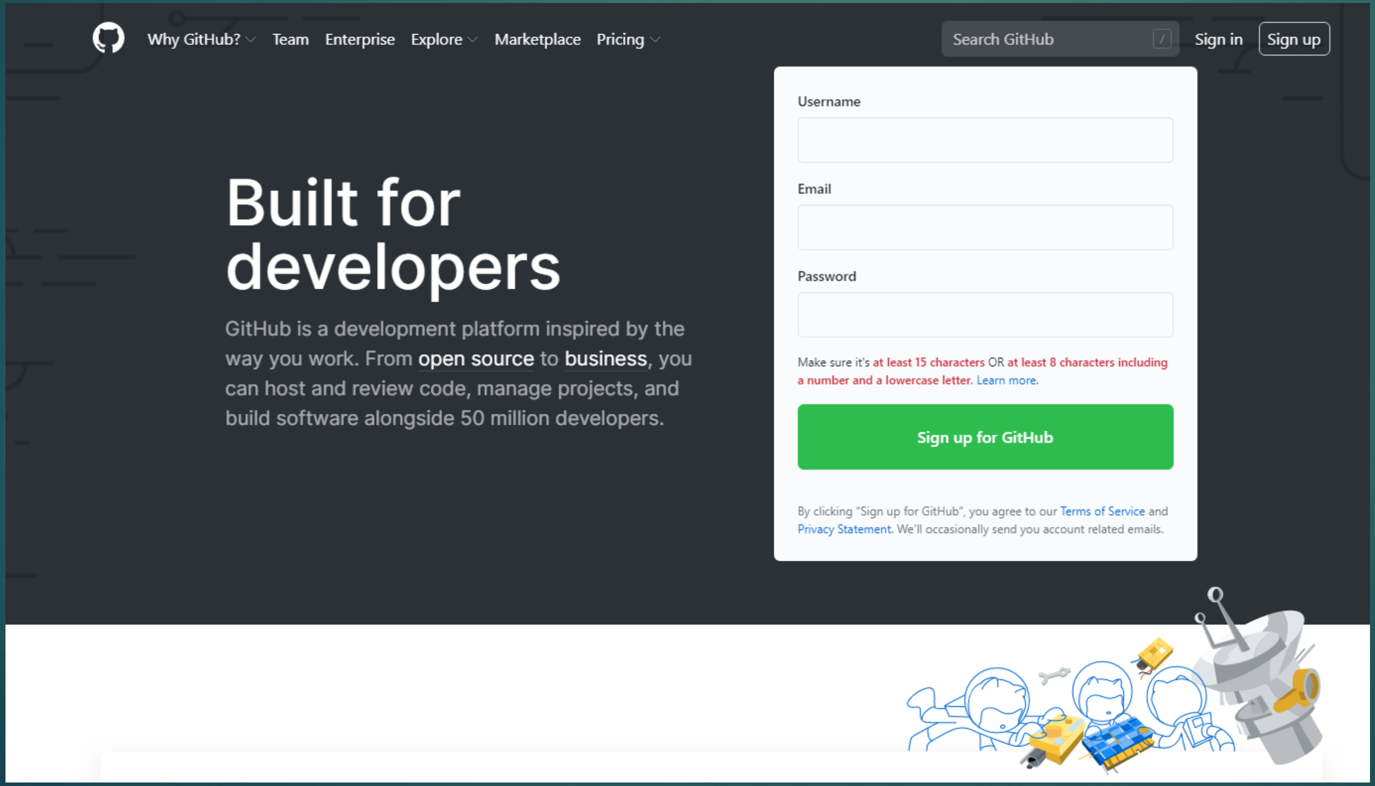
GITPOD - Cloud-based IDE (Integrated Development Environment). This is essentially your 'workspace' where you'll write your code; and use Git to push commits to your repo on Github.

GITHUB PAGES - Github also includes a free website hosting service called Github pages, where you can view your site live.

1. Setting up Github

Where to start? You need an account of course:

→ Head over to https://github.com/ - Create an account - Log in.



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.

Username

Email

Password

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

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

2. Creating a 'Repo'

Gitpod Extension – To open your repo in Gitpod from Github, you'll need to add the browser extension

Chrome

https://chrome.google.com/webstore/detail/gitpod-dev-environments-i/dodmmooeoklaejobgleioelladacbeki

Firefox - https://addons.mozilla.org/en-GB/firefox/addon/gitpod/

Gitpod Template - Code Institute has created a great template to get you started, including necessary tools and handy extensions.

You can find this at https://github.com/Code-Institute-Org/gitpod-full-template

Creating a 'Repo' - You can either:

- Head over to the Gitpod Template link above and press "Use this Template";

or

- Click the Octocat logo and then press the **"New"** button. You'll need to click the 'Repository template' dropdown and select the *Code-Institute-Org/gitpod-full-template* now listed there. Check the box next to "Include all branches".

3. Creating a ‘Repo’(cont..)

- **Name your Repo** - Name should be semantic, in lower case & separate words with a dash (i.e. my-first-repo). The repo name will appear on your repository and in the url.

- **Description** - A brief description of your repository/project (Milestone 1 Project for the Fullstack Software Development Diploma through the Code Institute)

- **Public/Private?** - Set to Public in order to deploy your site on Github pages.

- **Checkboxes**
  - Add a README file – Adds template README to customise
  - Add .gitignore
  - Choosing a license – *Don’t worry about this for now*

(Ps. When selecting the CI Template these <u>checkboxes will vanish</u> as the template includes everything you need).

4. Opening Repo in Gitpod

➡ **Gitpod button** - Once the Gitpod browser extension is installed, each new repo will have a green **Gitpod** button. Click this to open your repo in Gitpod.

➡ **Important:** - **Only click this ONCE when opening your repo initially.** (if clicking again later on, it will open a new instance of the repo).

➡ **Returning** - To go back to your workspace, you'll need to bookmark https://gitpod.io/workspaces/ and open it from the list.

(Unused workspaces are automatically deleted after 14 days of inactivity.)

➡ **Workspace** - The workspace is now ready (can take a minute or two to load up).

5. Git Commands

Add - First you need to Add a file for staging. To add all, you can:

* Add all - git add . (remember the space between add and the dot)

* Add individual files - git add index.html

* Add multiple files - git add index.html style.css myphoto.png

Commit - Once added, you can commit the files with a commit message:

i.e. - git commit –m “*Added bootstrap navigation bar and customised links*”

- git commit –m “*Added site structure elements in html*”

Note: Commit regularly, when a particular instance or material change has been completed

Push - Once commited you can push by using the command git push. All of your commits (since your last git push) will not be ‘pushed’ to your repository. In this case Github.

Note: When you push is entirely up to you. Some push after each commit, some when leaving the computer for a break, and some at the end of the day. Bear in mind that once you push your commits, the changes will now be safely amended in your repository…. You cannot ‘push’ too often.

6. Committing to committing

Commit often, but don't Commit half-done work

You should only commit code when a logical component is added/completed.

Write semantic Git messages

- A well-crafted Git commit message lets people quickly understand changes without having to read code (including you!).
- Capitalised, short (50 chars or less) summary in general.
- More detailed explanatory text, can be bumped to about 72 characters.

Using Branches

When working on a new feature, it is advisable to create a branch to work on. This allow you freedom to solve bugs/issues in the new feature without affecting the master.

https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging

7. Get Familiar

A few tips to keep in mind:

Start somewhere - i.e. Get your navbar up, style it, make sure its responsive. Add, commit, push. The best way to learn Git and become familiar with it, is to use it often.

Play! - Don’t be scared to try things. Open a Test repo and play around with Git commands.

- Add, commit and push
- Revert a commit to a previous commit
- Create a branch, work on that branch and then merge to master

Google - It’s there ALL the time, use it! You’ll find many answers to questions on Google and in in particular, Stack Overflow. Copy the error message in the console to Google to search.

- It’s good to get used to searching for resolutions to issues as this is a common practice as a developer, no matter how advanced.

Additional Resources

Check out these awesome resources:

Repos Setup guide - https://jimlynx.github.io/repo-setup-for-gitpod/

Git cheat sheet - https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet <-download PDF

Emmet cheat sheet - https://devhints.io/emmet

Emmet is a set of plug-ins for text editors that allow for high-speed coding and editing code formats via content assist. Think of shortcuts for 'coding auto-complete'.

Extra Extensions - The CI template comes with a few built in extensions, but you can more as you wish

- Bracket Pair Colorizer
- Indent Rainbow
- Beautify

Note: When installing extensions, you can choose whether to install for a particular workspace or globally

Slack – **Join calls!** This is an excellent way to interact and feel 'part' of the community

#in-it-together - Community group – daily meetup calls at 11am

#study-group - Thursday evening (8pm UK time) – Pre-determined topics and code-play