Git & Github

Clone

Let's take an example: Your company wants to develop a mobile app called "My-Mobile-App." Instead of starting the project from scratch, your teammate has already written some parts of the code. Now, your task is to continue building the project from that point. How would you approach integrating the code that your teammate wrote?

So here Come the concept of Github Cloning. The project's codebase is hosted on GitHub in a repository named "my-mobile-app." As a team member, you need to start working on the project and make your own contributions.

Here's how cloning helps:

- Accessing the Project: By cloning the "my-mobile-app" repository, you can create a local copy of the entire project on your computer. This allows you to access all the project files, including source code, assets, and configuration files.

- Making Changes Locally: Once you have cloned the repository, you can make changes to the code and experiment with new features without affecting the original project. You have the freedom to modify files, add new code, or even refactor existing code locally.

- Collaboration: Cloning enables effective collaboration. Imagine your teammate, Alice, has already made some updates to the project. By cloning the repository, you can fetch the latest changes made by Alice and merge them with your local copy. This way, you can keep your local version up to date with the latest developments in the project.

- Version Control: Cloning provides access to the entire version history of the project. You can view previous commits, switch between different branches, and explore the evolution of the codebase. This helps you understand the project's progress and allows you to roll back changes if necessary

- Pushing Changes: After making changes to your local copy, you can push those changes back to the remote repository on GitHub. This allows your team members, including Alice, to review your changes, provide feedback, and incorporate your contributions into the main project.

By cloning the "my-mobile-app" repository, you create a synchronized and independent copy of the project on your machine. This empowers you to work on the project, collaborate with teammates, and contribute your changes back to the main repository efficiently.
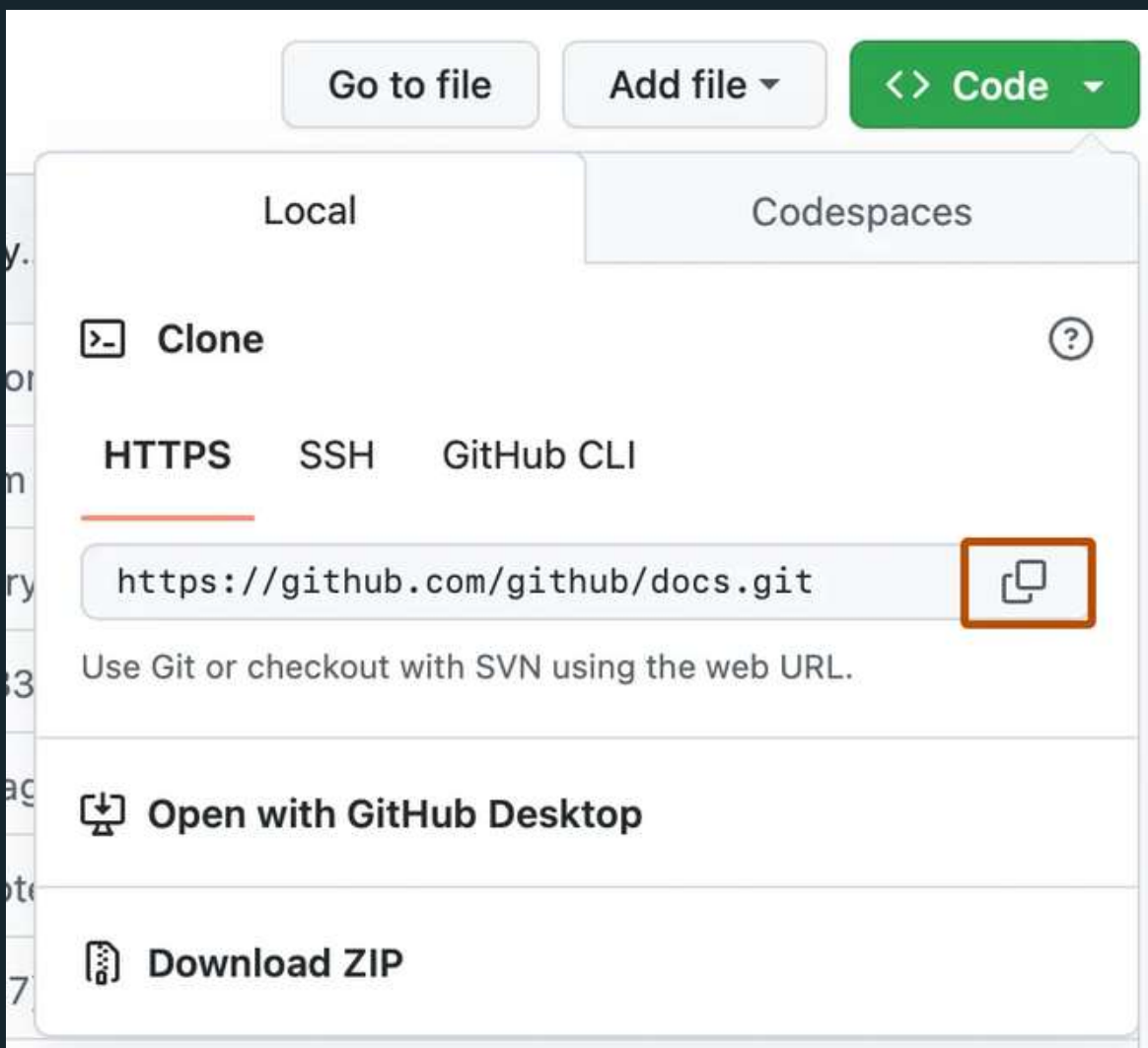
Cloning a repository

- On GitHub.com, navigate to the main page of the repository.

- Above the list of files, click <> Code.
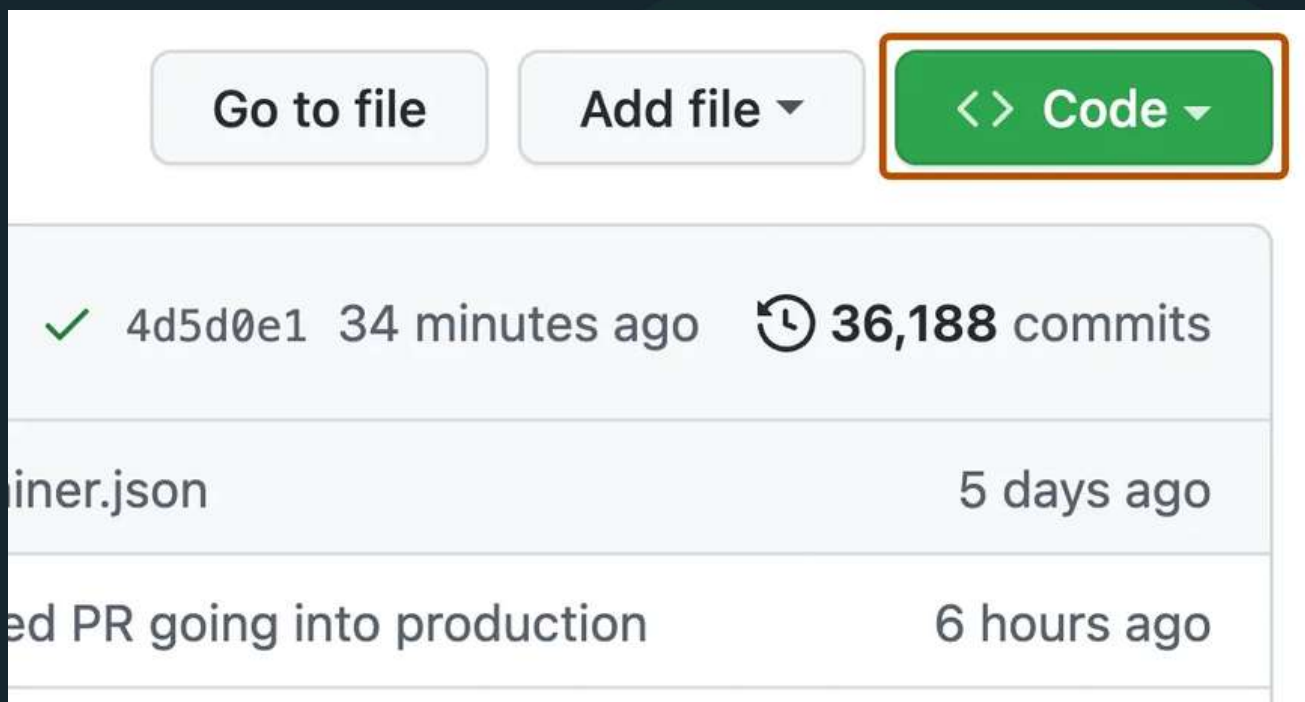
Copy the URL for the repository.

- To clone the repository using HTTPS, under "HTTPS", click .

- To clone the repository using an SSH key, including a certificate issued by your organization's SSH certificate authority, click SSH, then click .

- To clone a repository using GitHub CLI, click GitHub CLI, then click .

  - Open Git Bash.

  - Change the current working directory to the location where you want the cloned directory.

  - Type git clone, and then paste the URL you copied earlier

```
$ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY
```

- Press Enter to create your local clone.

```
$ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY
> Cloning into `Spoon-Knife`...
> remote: Counting objects: 10, done.
> remote: Compressing objects: 100% (8/8), done.
> remove: Total 10 (delta 1), reused 10 (delta 1)
> Unpacking objects: 100% (10/10), done.
```

Forking

Suppose you want to make changes or add features to a public code repository. If you have direct access to that repository, you can modify anything, which might not be ideal for the owner of that code. This is where the concept of forking comes into play. Forking allows us to create a separate copy of the public repository, enabling us to make changes without affecting the original codebase. Let's consider an example:

Let's say there is a popular open-source project called "AwesomeApp" hosted on GitHub. The repository URL is https://github.com/original-owner/AwesomeApp.

You are interested in the project and want to contribute to it or use it as a starting point for your own project.

To do this, you navigate to the "AwesomeApp" repository on GitHub.

- On the top-right corner of the repository page, you will find a button labeled "Fork." Clicking this button will create a copy of the entire repository under your own GitHub account.
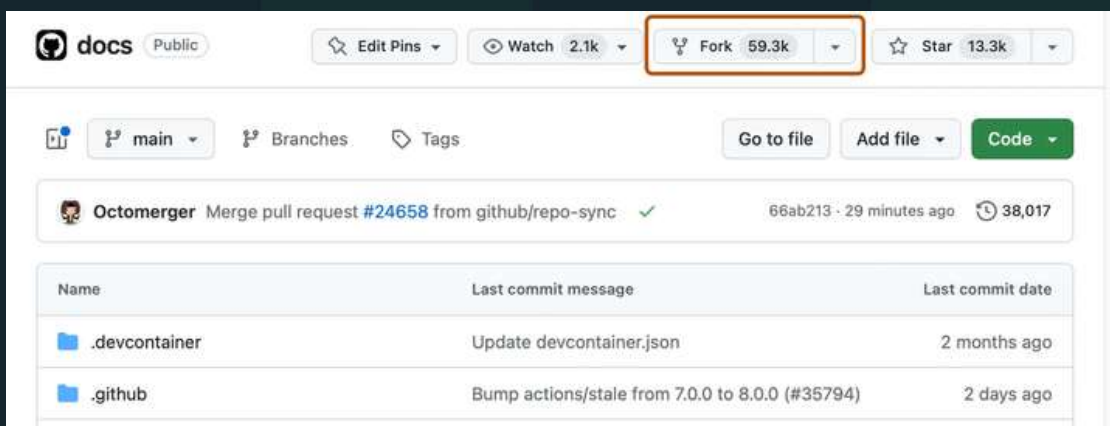
- After forking, you will be redirected to the URL of your forked repository, which will look like https://github.com/your-username/AwesomeApp.

- Now you have your own copy of the "AwesomeApp" repository on GitHub. You can clone this repository to your local machine using the steps I mentioned earlier.

- With the cloned repository, you can make changes, add new features, fix issues, or experiment with different approaches, all within your personal forked repository.

- Once you are satisfied with the changes or improvements you've made, you can propose those changes to the original repository for review and potential inclusion. This is done through a pull request.

- To submit your changes, go to your forked repository's page on GitHub, click on the "New pull request" button, and follow the prompts to create a pull request comparing your changes to the original repository.

- The owner of the original repository will review your changes and decide whether to accept them and merge them into the main project.

By forking a repository, you create an independent copy of the project under your GitHub account. This allows you to freely experiment, make changes, and contribute back to the original project. Forking is a powerful feature that fosters collaboration and enables individuals to participate in open-source projects or create their own variations based on existing codebases.

ReadMe

Suppose someone wants to run your project on their machine. If they visit your repository, how will they know about your project, how to use it, and how to run it? This is where the "README" file comes into play. The README file serves as a helpful documentation that provides instructions, explanations, and other details about your project. It helps users understand the purpose of the project, how to set it up, any dependencies required, and how to run it successfully.

Example of Readme

Contents of Readme File:

- Include Your Project's Title: This is the name of the project. It describes the whole project in few words and helps people understand the primary goal and aim.

- Write a Description: Your description is an essential part of your project. A well-maintained description allows you to show off your work to other developers as well as potential employers.

- How to Use Your Project: Provide instructions and examples so that users or contributors can use the project. This will make it easy for them so that if they encounter a problem, they will always have a place of reference.

- Include Credits: If you have worked on the project as a team, list your team members. You should also include their GitHub profiles.

Before diving deep into our project's readme, let's discuss markdown language.

Markdown is a lightweight markup language that allows us to style a digital text document using typical formatting techniques like headings, emphasis, lists, images, and links.

Markdown files have extensions .md or .markdown. We can convert Markdown into XHTML or HTML to display nicely in a browser. Some of the many uses of Markdown are:

1. readme files
2. Writing messages in online discussion forums
3. Creating rich text using a plain text editor, emails, and technical documentation.

Now, let's discuss elements of markdown language.

1. Headings:

Syntax:

```
# Heading level 1
## Heading level 2
### Heading level 3
#### Heading level 4
##### Heading level 5
###### Heading level 6
```

Output

Heading level 1

Heading level 2

Heading level 3

Heading level 4

Heading level 5

Heading level 6

2. Paragraphs:

Syntax:

```
Paragraph 1
Paragraph 2
```

3. Italics:

Syntax:

```
*one star on each side*
_This text is also italic_
```

4. Bold:

Syntax:

```
**two stars on each side**
__This text is also bold__
```

Output

two stars on each side
This text is also bold

5. Links:

Syntax:

```
[This text links to geekster](https://www.geekster.in/).
```

Output

This text links to geekster.

6. Images:

Syntax:

```
![Markdown symbol]
(https://upload.wikimedia.org/wikipedia/commons/4/48/Markdown-mark.svg)
```

Output

7. Unordered lists:

- Markdown allows you to format your lists with several different symbols: asterisks (*), hyphens (-), or plus signs (+).
- It's up to you to choose which symbol you prefer. The result you get is the same.

Syntax:

```
-Just add a dash first and then write a text.

-If you add another dash in the following line, you will have another
item in the list.
    - If you add four spaces or use a tab key, you will create an
indented list.
        - If you need sert an indenta list within an intended one, just
press a tab key again.

Sometimes you want bullet points:

*Start a line with a star
*Profit!
```

Output

- Just add a dash first and then write a text.
- If you add another dash in the following line, you will have another item in the list.
    - If you add four spaces or use a tab key, you will create an indented list.
        - If you need to insert an indented list within an intended one, just press a tab key again.

Sometimes you want bullet points:

- Start a line with a star
- Profit!

GitHub Pages

GitHub Pages is a static site hosting service that takes HTML, CSS, and JavaScript files straight from a repository on GitHub, optionally runs the files through a build process, and publishes a website

Creating your site

- On GitHub, navigate to your site's repository.

- Decide which publishing source you want to use.

- Create the entry file for your site. GitHub Pages will look for an index.html, index.md, or README.md file as the entry file for your site

  - Configure your publishing source

- Under your repository name, click Settings. If you cannot see the "Settings" tab, select the dropdown menu, then click Settings.
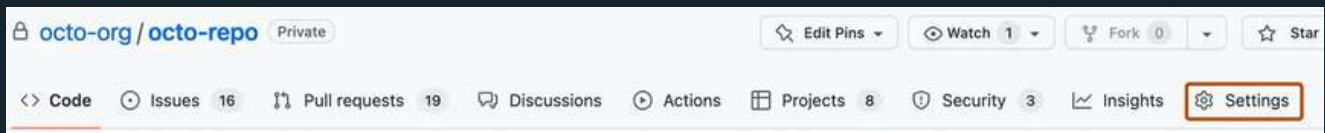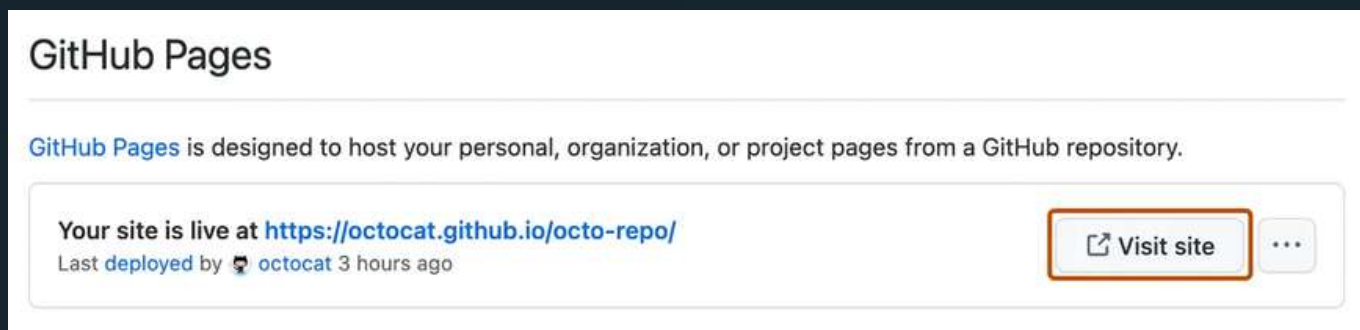

- In the "Code and automation" section of the sidebar, click Pages.
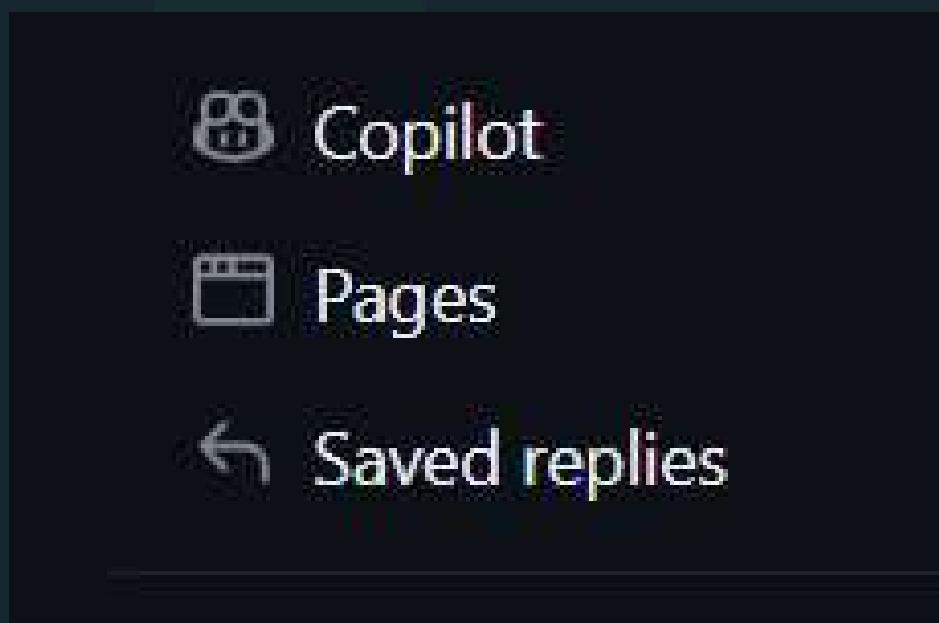
- After you have clicked on pages ,
- To see your published site, under "GitHub Pages", click Visit site.

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at https://octocat.github.io/octo-repo/
Last deployed by 🐙 octocat 3 hours ago

[Visit site]   ···

Once the deployment is complete, you can access your website at **https://<your-username>.github.io**. Replace <your-username> with your actual GitHub username.

If you followed the steps correctly, your website should display properly with the referenced images.