

Part 1: Using Source Control

1. Why version control?

Scenario 1:

- Your program is working
- You change “just one thing”
- Your program breaks
- You change it back
- Your program is still broken--why?
- Has this ever happened to you?
- Your program worked well enough yesterday
- You made a lot of improvements last night but you haven't gotten them to work yet
- You need to turn in your program now
- Has this ever happened to you?

2. Version control for teams

Scenario 2:

- You change one part of a program--it works
- Your co-worker changes another part--it works
- You put them together--it doesn't work
- Some change in one part must have broken something in the other part
- What were all the changes?

Scenario 3:

- You make several improvements to a class
- Your co-worker makes many different improvements to the same class
- How can you merge these changes?

3. Version control systems

A version control system (often called a source code control system) does these things:

- Keeps multiple (older and newer) versions of everything (not just source code)
- Requests comments regarding every change
- Allows “check in” and “check out” of files so you know which files someone else is working on
- Displays differences between versions

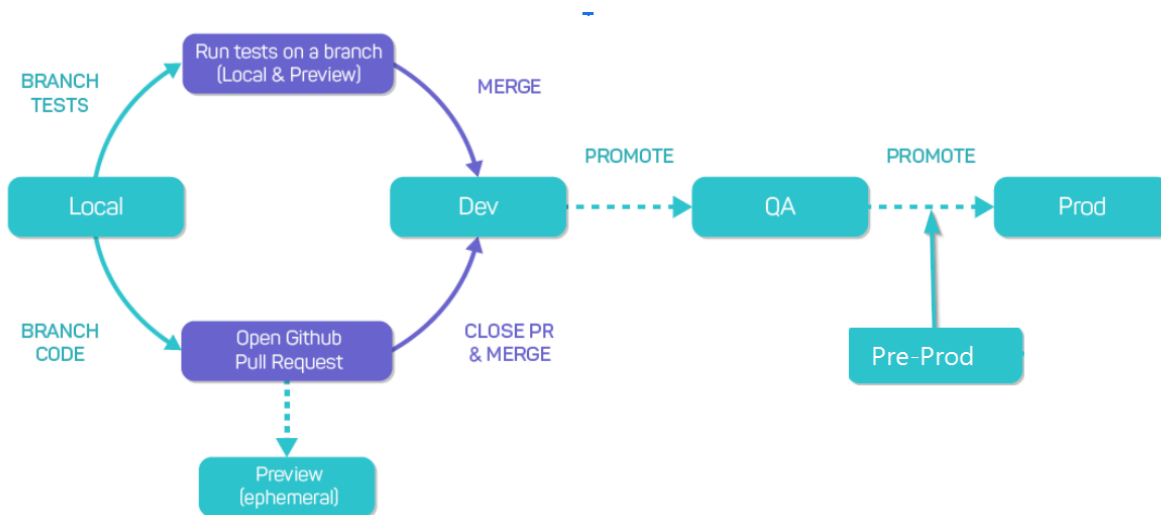
4. Benefits of version control

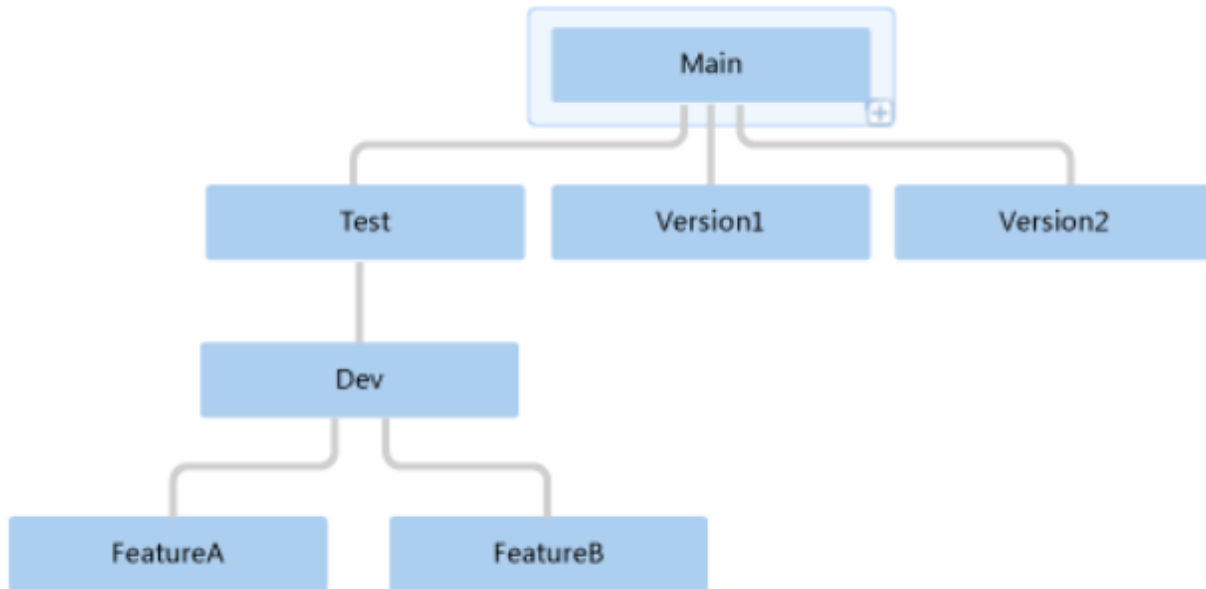
For working by yourself:

- Gives you a “time machine” for going back to earlier versions
- Gives you great support for different versions (standalone, web app, etc.) of the same basic project
- For working with others:
- Greatly simplifies concurrent work, merging changes

5. What are Git and GitHub

- Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency
- GitHub is a web-based Git repository hosting service, which offers all of the distributed revision control source code management (SCM) functionality of Git as well as adding its features.





6. Setup and Use Github steps

The first steps in starting with GitHub are to create an account, choose a product that fits your needs best, verify your email, set up two-factor authentication, and view your profile.

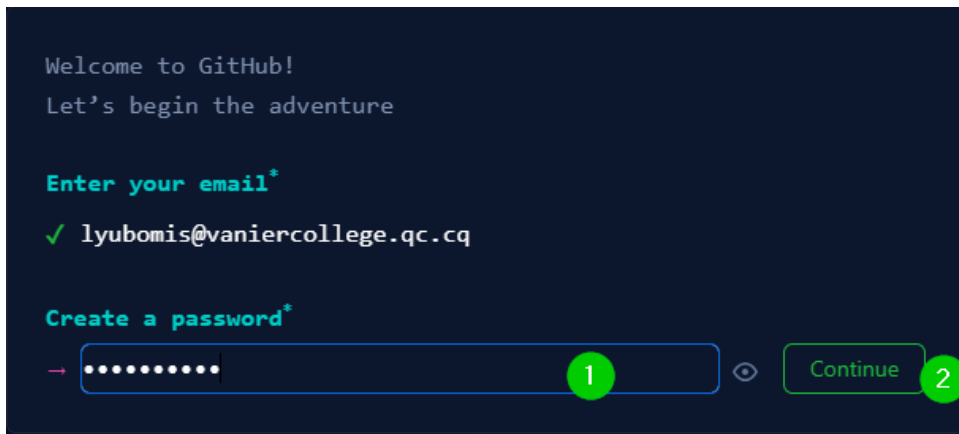
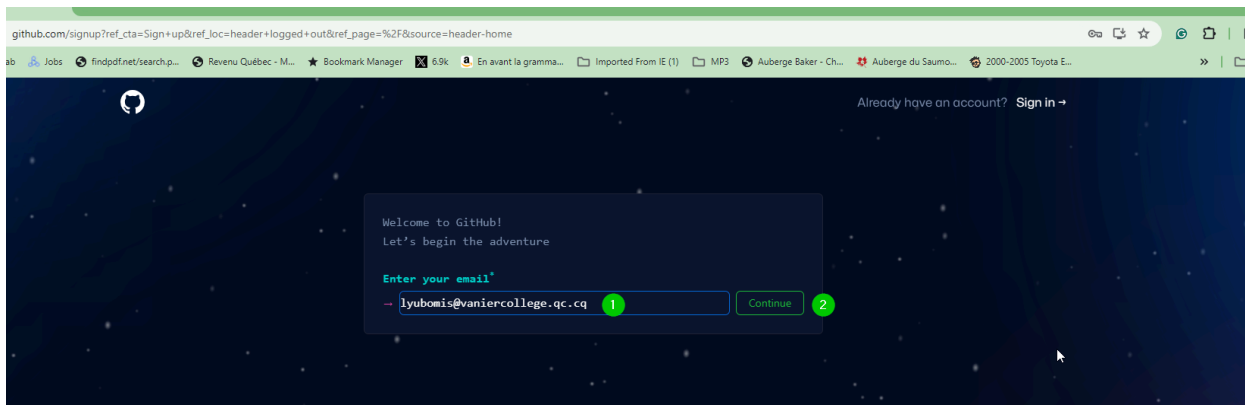
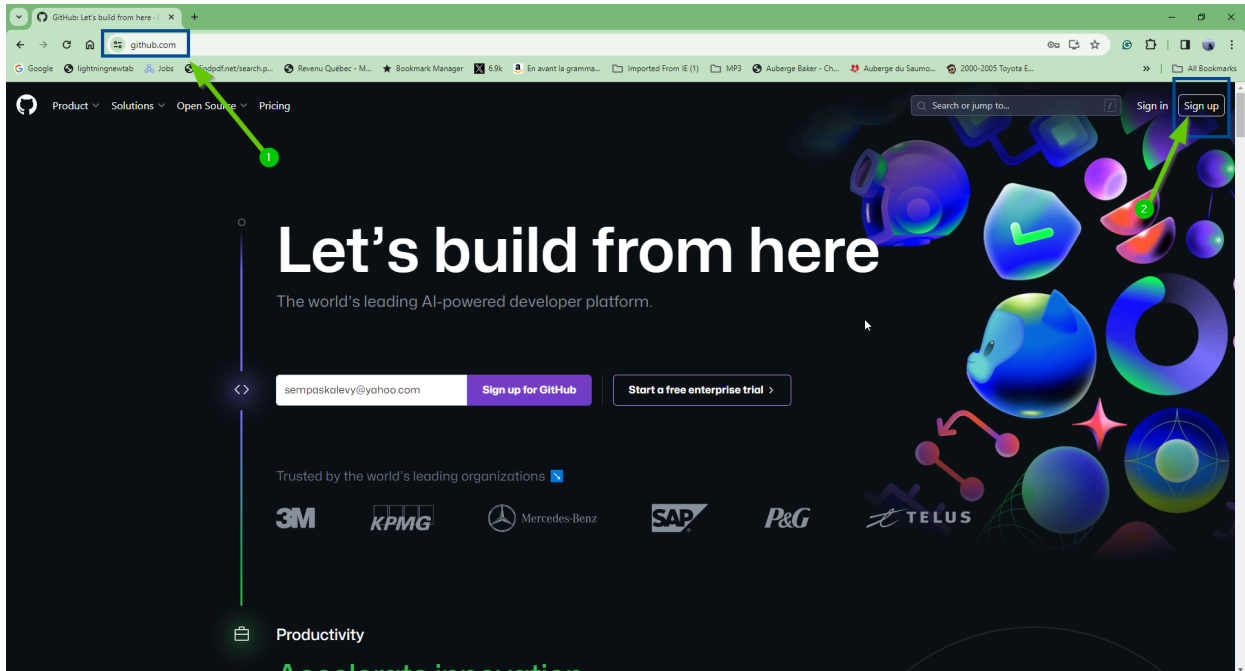
There are several types of accounts on GitHub. Every person who uses GitHub has an account, which can be part of multiple organizations and teams. Your account is your identity on GitHub.com and represents you as an individual.

7. Choosing your GitHub product

You can choose **GitHub Free** or GitHub Pro to get access to different features for your account. You can upgrade at any time if you are unsure at first which product you want.

8. Verifying your email address

To ensure you can use all the features in your GitHub plan, verify your email address after signing up for a new account. For more information, see "[Verifying your email address](#)."



Welcome to GitHub!
Let's begin the adventure

Enter your email*

✓ lyubomis@vaniercollege.qc.cq

Create a password*

✓ (.....)

Enter a username*

→ lyubomis 1

Continue 2

Welcome to GitHub!
Let's begin the adventure

Enter your email*

✓ lyubomis@vaniercollege.qc.cq

Create a password*

✓ (.....)

Enter a username*

✓ lyubomis

Email preferences

☐ Receive occasional product updates and announcements. optional

Continue

Welcome to GitHub!

Let's begin the adventure

Enter your email*

✓ lyubomis@vaniercollege.qc.ca

Create a password*

✓ (.....)

Enter a username*

✓ lyubomis

Email preferences

☐ Receive occasional product updates and announcements.

Verify your account

Protecting your account

Please solve this puzzle so we know you are a real person

Verify

Welcome to GitHub!
Let's begin the adventure

Enter your email*

✓ lyubomis@vaniercollege.qc.ca

Create a password*

✓ (.....)

Enter a username*


✓ lyubomis

Email preferences

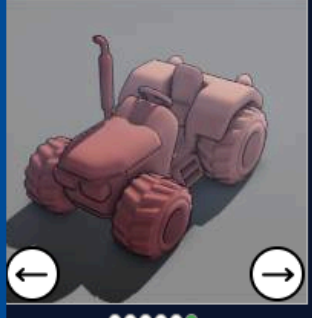
☐ Receive occasional product updates and announcements.

Verify your account 1

Use the arrows to rotate the object to face in the direction of the hand. (1 of 1)



Match this angle



Submit 2

9. Configuring two-factor authentication

Two-factor authentication, or 2FA, is an extra layer of security used when logging into websites or apps. We strongly urge you to configure 2FA for the safety of your account. For more information, see "[About two-factor authentication](#)."

Optionally, after you have configured 2FA, add a passkey to your account to enable a secure, passwordless login. For more information, see "[About passkeys](#)" and "[Managing your passkeys](#)."

10. [Viewing your GitHub profile and contribution graph](#)

Your GitHub profile tells people the story of your work through the repositories and lists you've pinned, the organization memberships you've chosen to publicize, the contributions you've made, and the projects you've created. For more information, see "[About your profile](#)" and "[Viewing contributions on your profile](#)."

Part 2: Using GitHub's tools and processes

To best use GitHub, you'll need to set up Git. Git is responsible for everything GitHub-related that happens locally on your computer. To effectively collaborate on GitHub, you'll write in issues and pull requests using GitHub Flavored Markdown.

1. Learning Git

GitHub's collaborative approach to development depends on publishing commits from your local repository to GitHub for other people to view, fetch, and update using Git. For more information about Git, see the "[Git Handbook](#)" guide. For more information about how Git is used on GitHub, see "[GitHub flow](#)."

2. Setting up Git

If you plan to use Git locally on your computer, whether through the command line, an IDE or text editor, you will need to install and set up Git. For more information, see "[Set up Git](#)."

If you prefer to use a visual interface, you can download and use GitHub Desktop. GitHub Desktop comes packaged with Git, so there is no need to install Git separately.

Once you install Git, you can connect to GitHub repositories from your local computer, whether your own repository or another user's fork. When you connect to a repository on GitHub.com from Git, you'll need to authenticate with GitHub using either HTTPS or SSH.

3. Choosing how to interact with GitHub

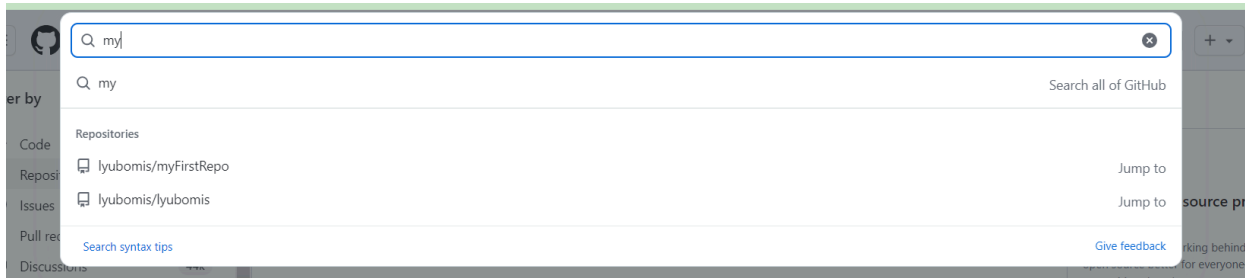
Everyone has their own unique workflow for interacting with GitHub; the interfaces and methods you use depend on your preference and what works best for your needs.

4. Writing on GitHub

To make your communication clear and organized in issues and pull requests, you can use GitHub Flavored Markdown for formatting, which combines an easy-to-read, easy-to-write syntax with some custom functionality.

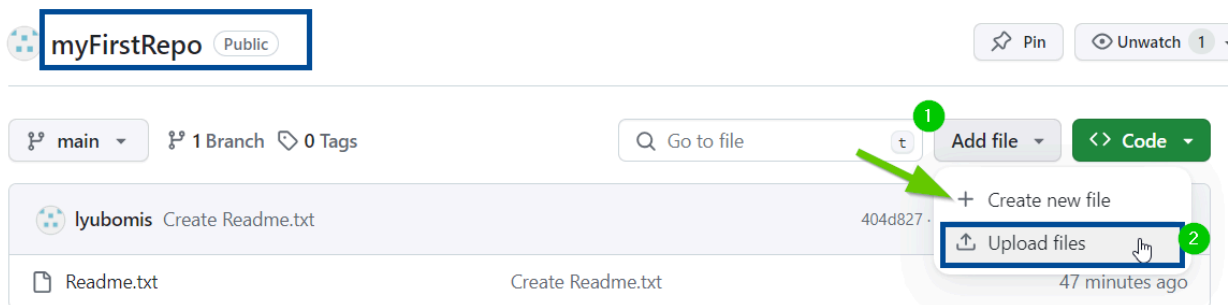
5. Searching on GitHub

Our integrated search allows you to find what you are looking for among the many repositories, users and lines of code on GitHub. You can search globally across all of GitHub or limit your search to a particular repository or organization.



6. Managing files on GitHub

With GitHub, you can create, edit, move and delete files in your repository or any repository you have write access to. You can also track the history of changes in a file line by line.



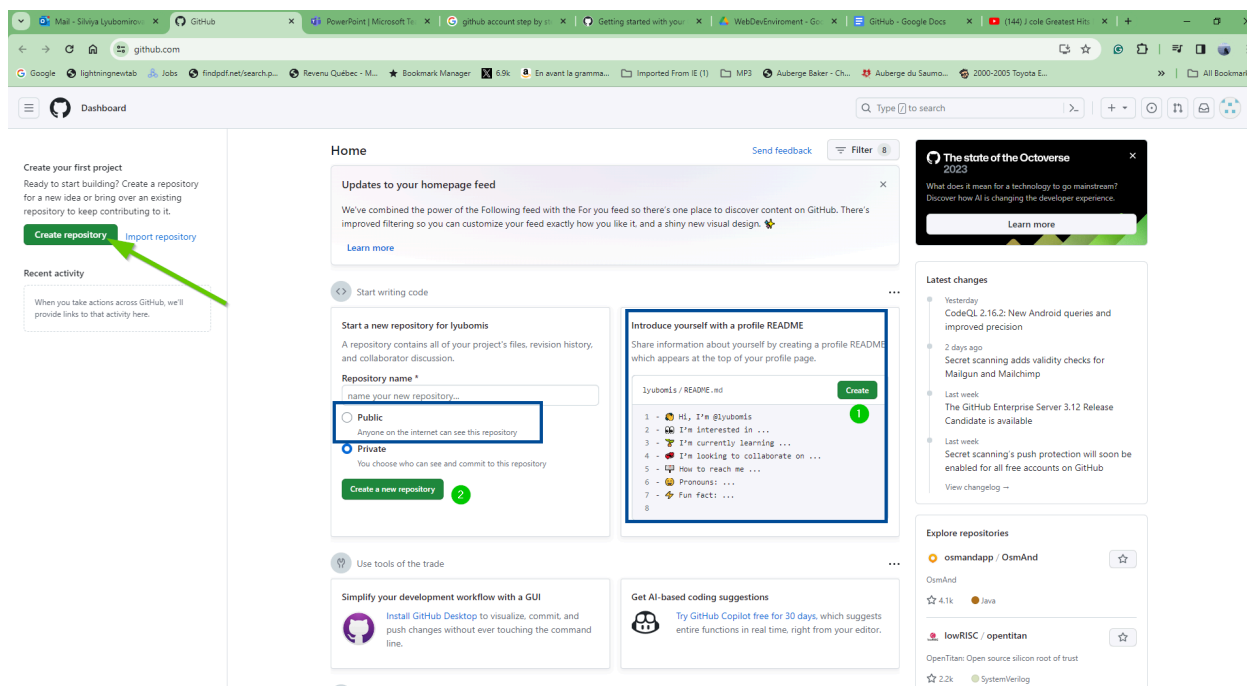
Part 3: Collaborating on GitHub

Any number of people can work together in repositories across GitHub. You can configure settings, create projects, and manage your notifications to encourage effective collaboration.

1. Working with repositories

Creating a repository

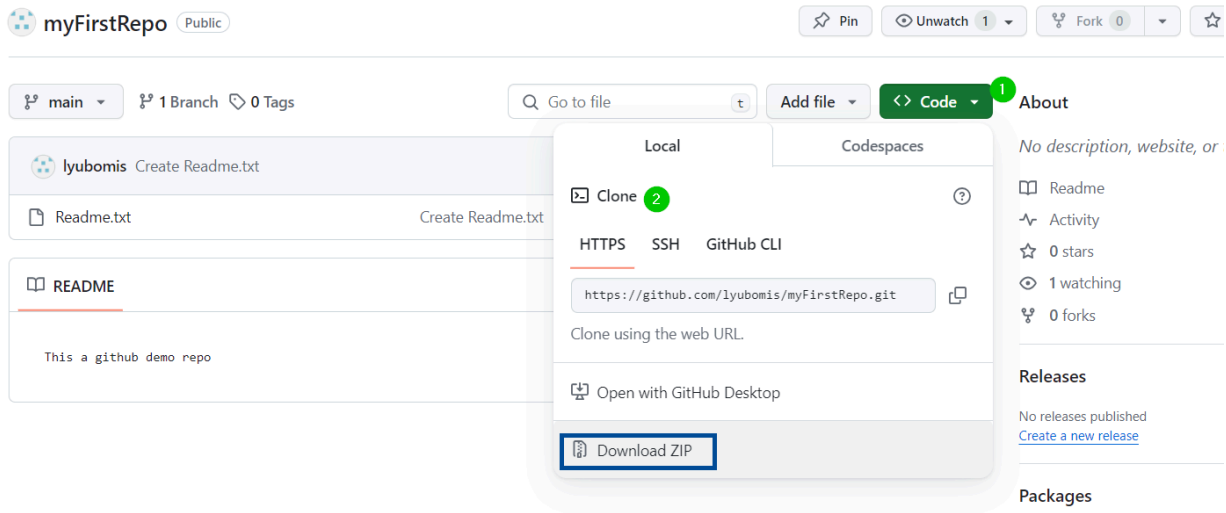
A repository is like a folder for your project. You can have any number of public and private repositories in your personal account. Repositories can contain folders and files, images, videos, spreadsheets, and data sets, as well as the revision history for all files in the repository. For more information, see "[About repositories](#)."



When you create a new repository, you should initialize the repository with a README file to let people know about your project. For more information, see "[Creating a new repository](#)."

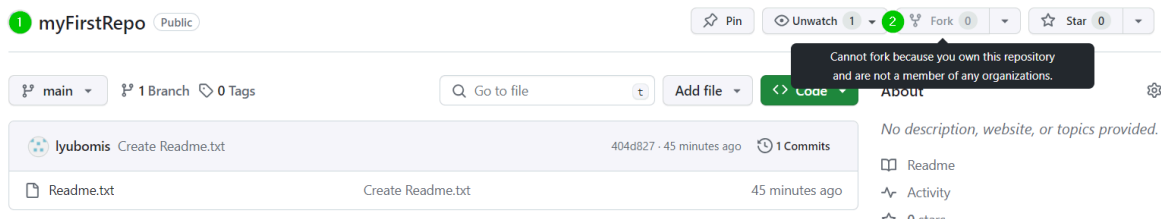
Cloning a repository

You can clone an existing repository from GitHub to your local computer, making it easier to add or remove files, fix merge conflicts, or make complex commits. Cloning a repository pulls down a full copy of all the repository data that GitHub has at that point in time, including all versions of every file and folder for the project.



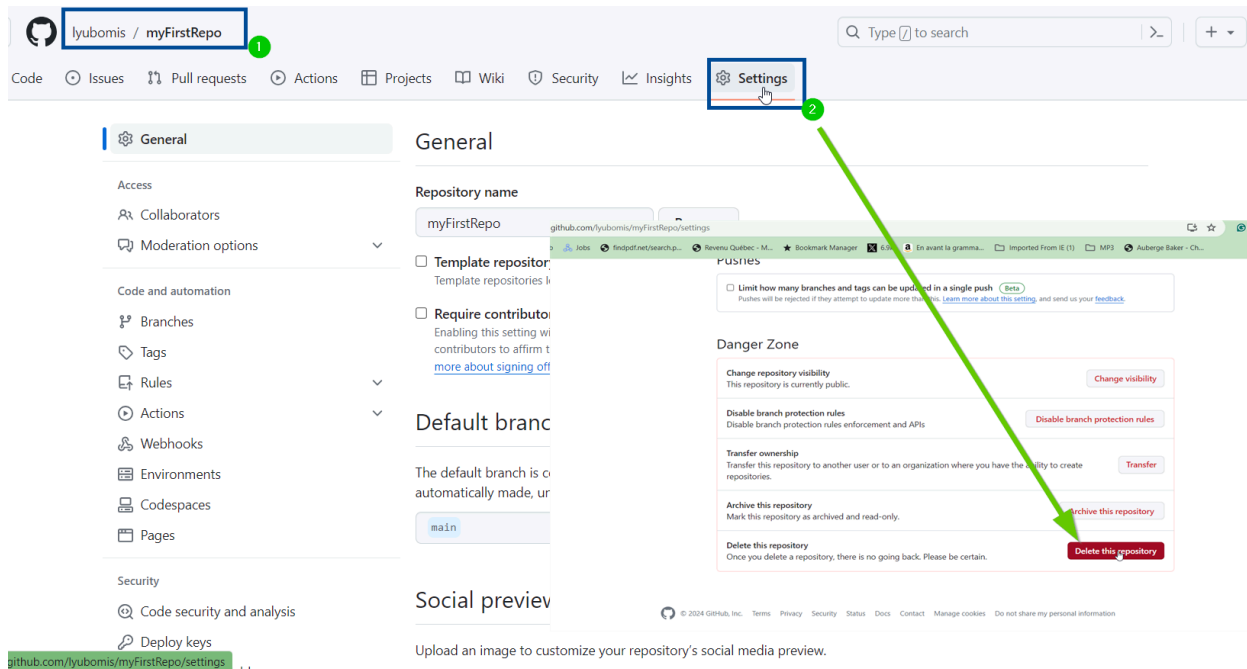
Forking a repository

A fork is a copy of a repository that you manage, where any changes you make will not affect the original repository unless you submit a pull request to the project owner. Most commonly, forks are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea.

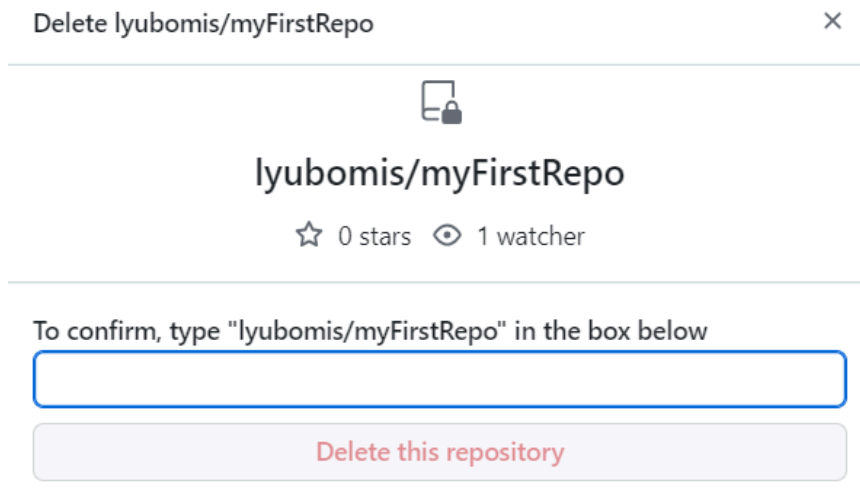


Deleting a repository

After you log in to the repository click on settings and after you scroll until the bottom of the page you will see the option to delete.



you will need to confirm your decision with the password.




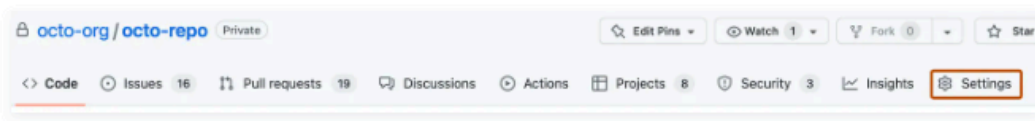
2. Importing your projects


If you have existing projects you'd like to move over to GitHub you can import projects using the GitHub Importer, the command line, or external migration tools. For more information, see "[Importing source code](#)."

3. Managing collaborators and permissions

You can collaborate on your project with others using your repository's issues, pull requests, and projects (classic). You can invite other people to your repository as collaborators from the Collaborators tab in the repository settings.

- 1 Ask for the username of the person you're inviting as a collaborator. If they don't have a username yet, they can sign up for GitHub. For more information, see "[Creating an account on GitHub](#)."
- 2 On GitHub.com, navigate to the main page of the repository.
- 3 Under your repository name, click  **Settings**. If you cannot see the "Settings" tab, select the ... dropdown menu, then click **Settings**.



- 4 In the "Access" section of the sidebar, click  **Collaborators**.
- 5 Click **Add people**.
- 6 In the search field, start typing the name of person you want to invite, then click a name in the list of matches.
- 7 Click **Add NAME to REPOSITORY**.
- 8 The user will receive an email inviting them to the repository. Once they accept your invitation, they will have collaborator access to your repository.

You are the owner of any repository you create in your personal account and have full control of the repository. Collaborators have write access to your repository, limiting what they have permission to do. For more information, see "[Permission levels for a personal account repository](#)."

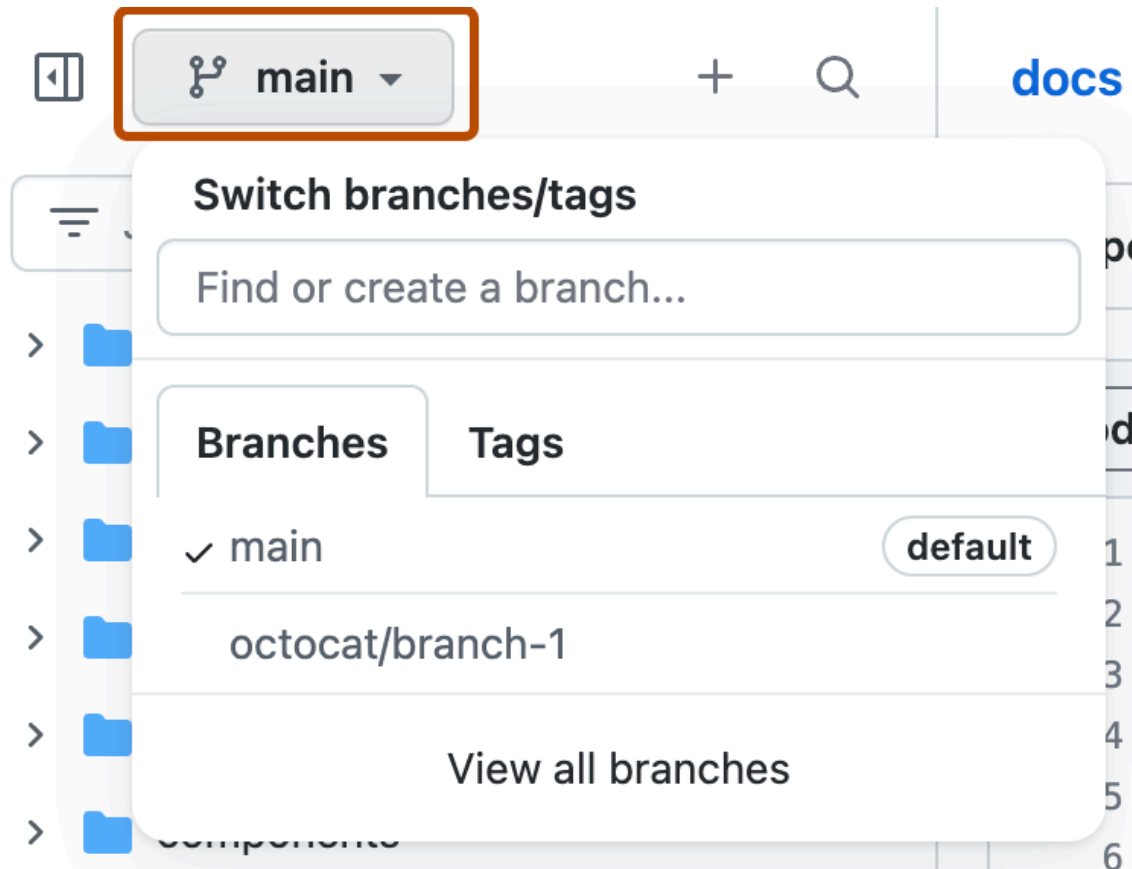
4. Managing repository settings

As the owner of a repository you can configure several settings, including the repository's visibility, topics, and social media preview. For more information, see "[Managing your repository's settings and features](#)."

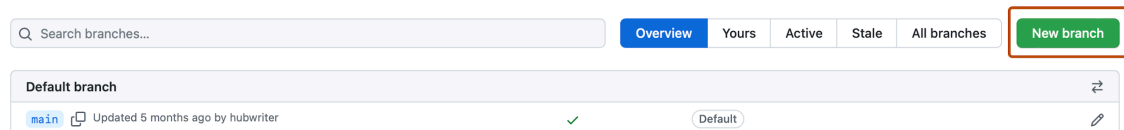
Branches

Creating a branch via the branches overview

On GitHub.com, navigate to the main page of the repository. From the file tree view on the left, select the branch dropdown menu, then click View all branches. You can also find the branch dropdown menu at the top of the integrated file editor.



Click New branch.



Under "Branch name", type a name for the branch.

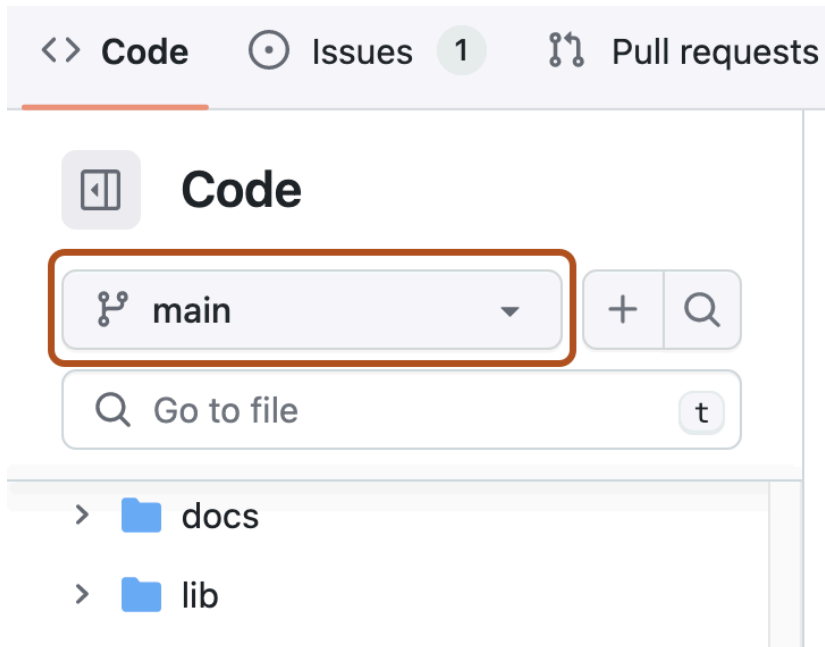
Under "Branch source", choose a source for your branch.

- If your repository is a fork, select the repository dropdown menu and click your fork or the upstream repository.
- Select the branch dropdown menu and click a branch.

Click Create branch.

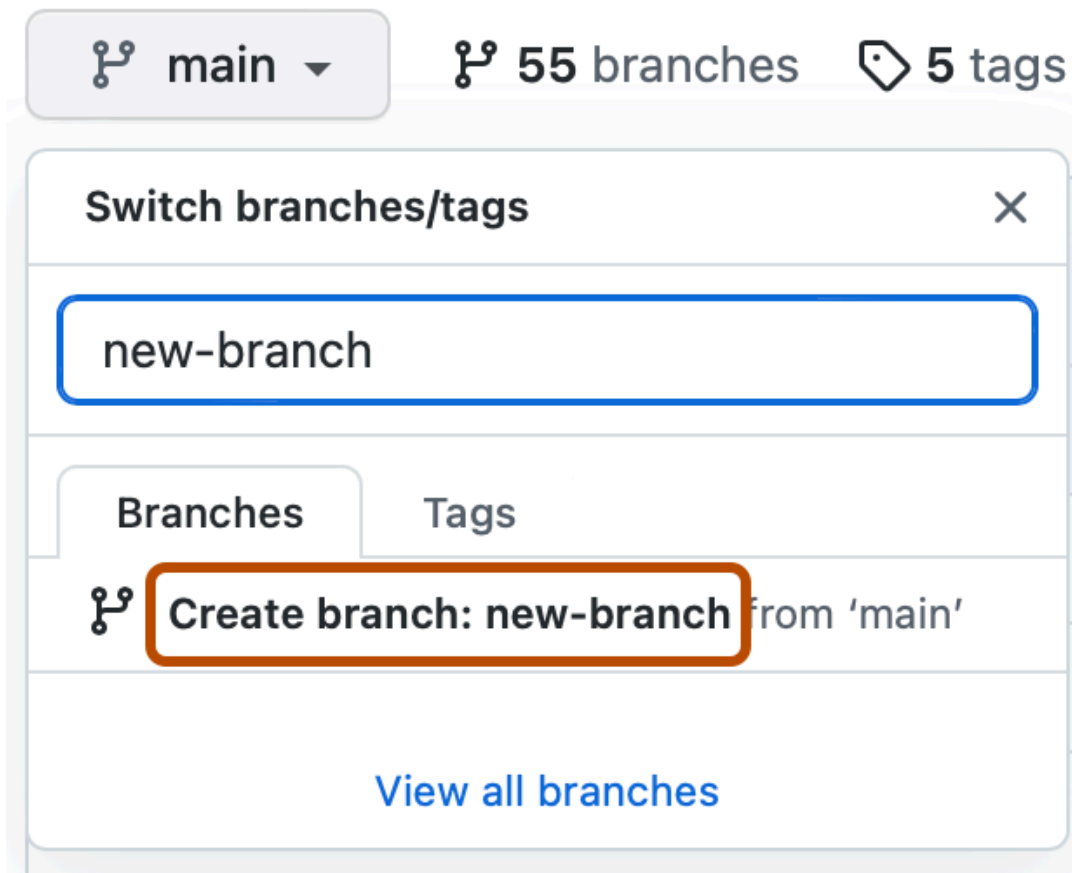
Creating a branch using the branch dropdown

On GitHub.com, navigate to the main page of the repository.
Select the branch dropdown menu, in the file tree view or at the top of the integrated file editor.



Optionally, if you want to create the new branch from a branch other than the default branch of the repository, click another branch, then select the branch dropdown menu again.

In the "Find or create a branch..." text field, type a unique name for your new branch, then click Create branch.



Creating a branch for an issue

You can create a branch to work on an issue directly from the issue page and get started right away. For more information, see "[Creating a branch to work on an issue](#)".

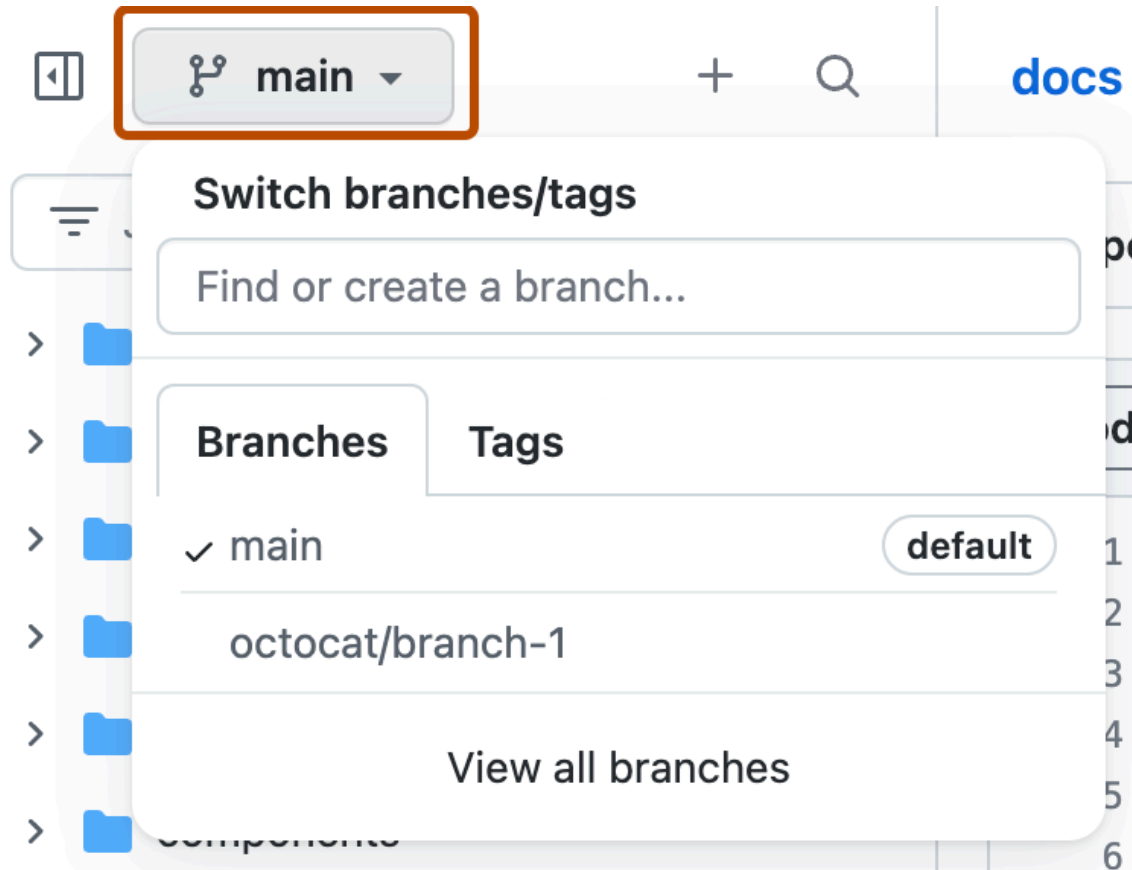
Deleting a branch

You can have head branches automatically deleted after pull requests are merged in your repository. For more information, see "[Managing the automatic deletion of branches](#)".

Note: If the branch you want to delete is the repository's default branch, you must choose a new default branch before deleting the branch. For more information, see "[Changing the default branch](#)".

If the branch you want to delete is associated with an open pull request, you must merge or close the pull request before deleting the branch. For more information, see "[Merging a pull request](#)" or "[Closing a pull request](#)".

On GitHub.com, navigate to the main page of the repository. From the file tree view on the left, select the branch dropdown menu, then click View all branches. You can also find the branch dropdown menu at the top of the integrated file editor.



Next to the branch that you want to delete, click.



If the branch is associated with at least one open pull request, deleting the branch will close the pull requests. Read the warning, then click Delete.

If you delete a head branch after its pull request has been merged, GitHub checks for any open pull requests in the same repository that specify the deleted branch as their base branch. GitHub automatically updates any such pull requests, changing their base branch to the merged pull request's base branch. For more information, see "[About branches](#)."