

Hello World

Follow this Hello World exercise to get started with GitHub.

In this article

[Introduction](#)

[Creating a repository](#)

[Creating a branch](#)

[Create a branch](#)

[Making and committing changes](#)

[Opening a pull request](#)

[Merging your pull request](#)

[Next steps](#)

Introduction

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

This tutorial teaches you GitHub essentials like repositories, branches, commits, and pull requests. You'll create your own Hello World repository and learn GitHub's pull request workflow, a popular way to create and review code.

In this quickstart guide, you will:

- Create and use a repository
- Start and manage a new branch
- Make changes to a file and push them to GitHub as commits
- Open and merge a pull request

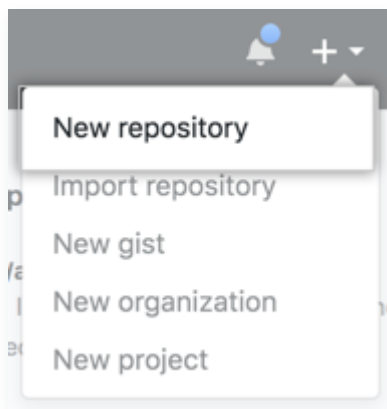
To complete this tutorial, you need a [GitHub account](#) and Internet access. You don't need to know how to code, use the command line, or install Git (the version control software that GitHub is built on).

Creating a repository

A repository is usually used to organize a single project. Repositories can contain folders and files, images, videos, spreadsheets, and data sets -- anything your project needs. Often, repositories include a `README` file, a file with information about your project. GitHub makes it easy to add one at the same time you create your new repository. It also offers other common options such as a license file.

Your `hello-world` repository can be a place where you store ideas, resources, or even share and discuss things with others.


- 1 In the upper-right corner of any page, use the **+** drop-down menu, and select **New repository**.



- 2 In the **Repository name** box, enter `hello-world`.
- 3 In the **Description** box, write a short description.
- 4 Select **Add a README file**.
- 5 Click **Create repository**.

Owner *

Repository name *

 octocat ▾


 /


hello-world ✓

Great repository names are short and memorable. Need inspiration? How about [ubiquitous-system?](#)

Description (optional)

My first repository

☐  **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

Creating a branch

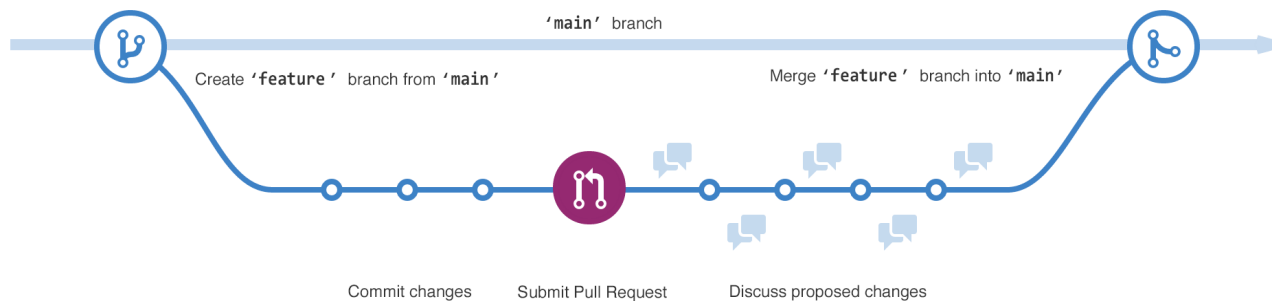
Branching lets you have different versions of a repository at one time.

By default, your repository has one branch named `main` that is considered to be the definitive branch. You can use branches to experiment and make edits before committing them to `main`.

When you create a branch off the `main` branch, you're making a copy, or snapshot, of `main` as it was at that point in time. If someone else made changes to the `main` branch while you were working on your branch, you could pull in those updates.

This diagram shows:

- The `main` branch
- A new branch called `feature`
- The journey that `feature` takes before it's merged into `main`



Have you ever saved different versions of a file? Something like:

- `story.txt`

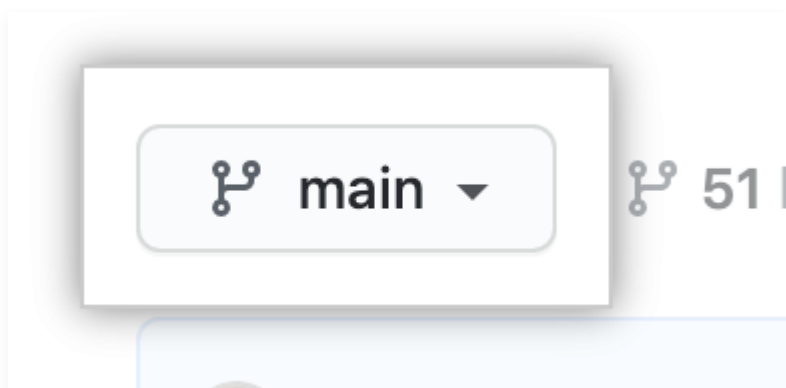
- `story-joe-edit.txt`
- `story-joe-edit-reviewed.txt`

Branches accomplish similar goals in GitHub repositories.

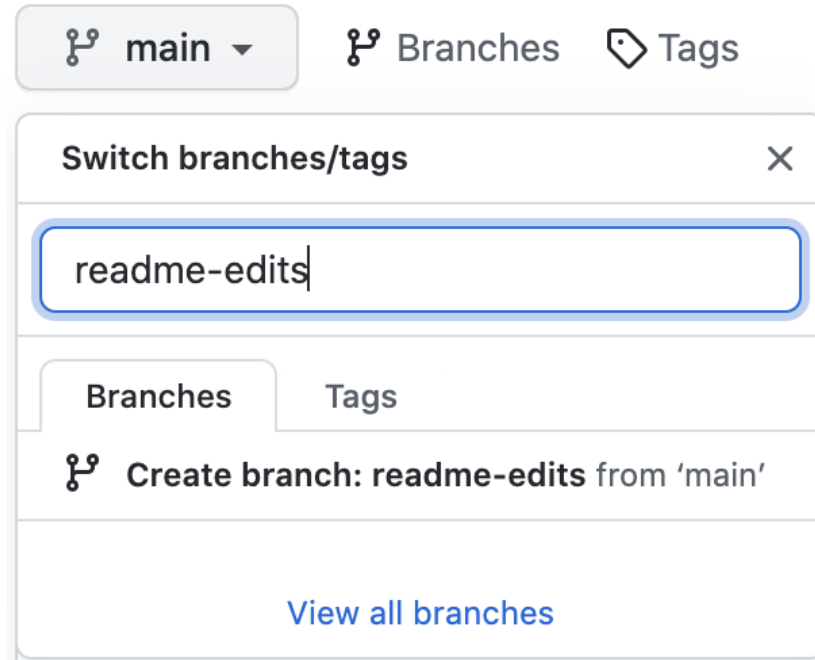
Here at GitHub, our developers, writers, and designers use branches for keeping bug fixes and feature work separate from our `main` (production) branch. When a change is ready, they merge their branch into `main`.

Create a branch

- 1 Click the **Code** tab of your `hello-world` repository.
- 2 Click the drop down at the top of the file list that says **main**.



- 3 Type a branch name, `readme-edits`, into the text box.
- 4 Click **Create branch: readme-edits from main**.




Now you have two branches, `main` and `readme-edits`. Right now, they look exactly the same. Next you'll add changes to the new branch.

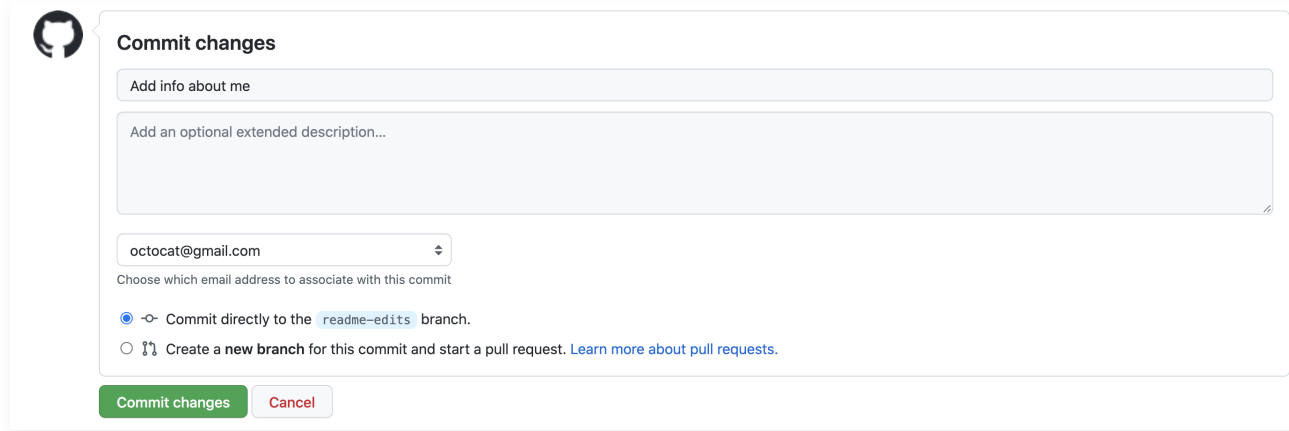
Making and committing changes

When you created a new branch in the previous step, GitHub brought you to the code page for your new `readme-edits` branch, which is a copy of `main`.

You can make and save changes to the files in your repository. On GitHub, saved changes are called commits. Each commit has an associated commit message, which is a description explaining why a particular change was made. Commit messages capture the history of your changes so that other contributors can understand what you've done and why.

- 1 Click the `README.md` file.

- 2 Click  to edit the file.
- 3 In the editor, write a bit about yourself.
- 4 In the **Commit changes** box, write a commit message that describes your changes.
- 5 Click **Commit changes**.



The screenshot shows the GitHub 'Commit changes' dialog box. It features the GitHub logo on the left. The main area contains a text input field for 'Add info about me', a larger text area for 'Add an optional extended description...', and a dropdown menu showing 'octocat@gmail.com'. Below the dropdown, it says 'Choose which email address to associate with this commit'. There are two radio button options: 'Commit directly to the `readme-edits` branch.' (which is selected) and 'Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)'. At the bottom, there are two buttons: 'Commit changes' (green) and 'Cancel' (red).

These changes will be made only to the README file on your `readme-edits` branch, so now this branch contains content that's different from `main`.

Opening a pull request

Now that you have changes in a branch off of `main`, you can open a pull request.

Pull requests are the heart of collaboration on GitHub. When you open a pull request, you're proposing your changes and requesting that someone review and pull in your contribution and

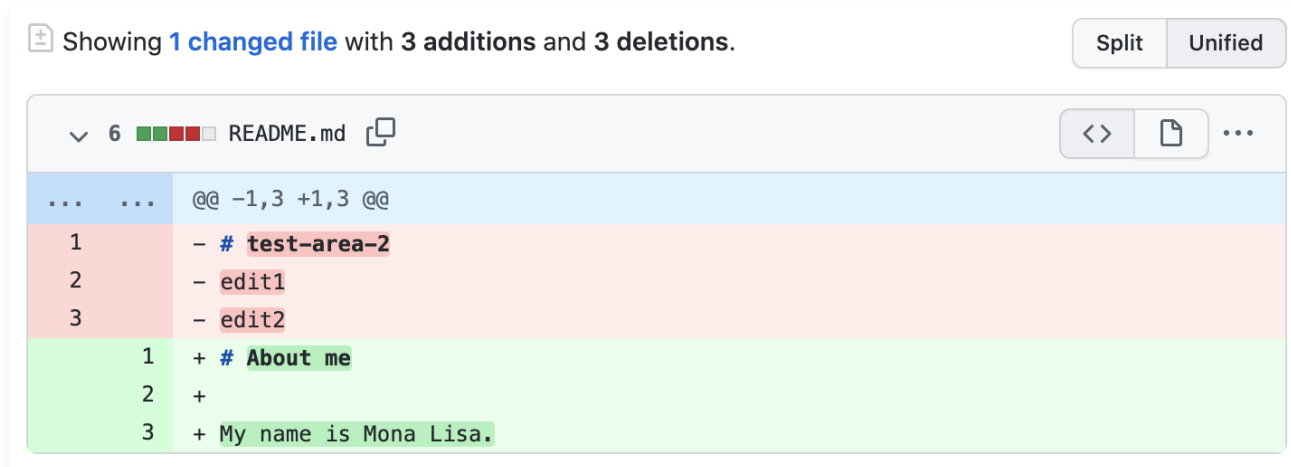
merge them into their branch. Pull requests show diffs, or differences, of the content from both branches. The changes, additions, and subtractions are shown in different colors.

As soon as you make a commit, you can open a pull request and start a discussion, even before the code is finished.

By using GitHub's `@mention` feature in your pull request message, you can ask for feedback from specific people or teams, whether they're down the hall or 10 time zones away.

You can even open pull requests in your own repository and merge them yourself. It's a great way to learn the GitHub flow before working on larger projects.

- 1 Click the **Pull requests** tab of your `hello-world` repository.
- 2 Click **New pull request**
- 3 In the **Example Comparisons** box, select the branch you made, `readme-edits`, to compare with `main` (the original).
- 4 Look over your changes in the diffs on the Compare page, make sure they're what you want to submit.



- 5 Click **Create pull request**.
- 6 Give your pull request a title and write a brief description of your changes. You can include emojis and drag and drop images and gifs.
- 7 Click **Create pull request**.

Your collaborators can now review your edits and make suggestions.

Merging your pull request

In this final step, you will merge your `readme-edits` branch into the `main` branch.

- 1 Click **Merge pull request** to merge the changes into `main`.
- 2 Click **Confirm merge**.
- 3 Go ahead and delete the branch, since its changes have been incorporated, by clicking **Delete branch**.

Next steps

By completing this tutorial, you've learned to create a project and make a pull request on GitHub.

Here's what you accomplished in this tutorial:

- Created an open source repository

- Started and managed a new branch
- Changed a file and committed those changes to GitHub
- Opened and merged a pull request

Take a look at your GitHub profile and you'll see your work reflected on your contribution graph.

For more information about the power of branches and pull requests, see "[GitHub flow](#)." For more information about getting started with GitHub, see the other guides in the [getting started quickstart](#).