Creating your first repository using GitHub Desktop

In this article

You can use GitHub Desktop to create and manage a Git repository without using the command line.

Introduction 🔗

GitHub Desktop is a free, open source application that helps you to work with code hosted on GitHub or other Git hosting services. With GitHub Desktop, you can perform Git commands, such as committing and pushing changes, in a graphical user interface, rather than using the command line. For more information, see "About GitHub Desktop."

By the end of this guide, you'll have used GitHub Desktop to create a repository, make changes to the repository, and publish the changes to GitHub Enterprise Server.

After installing GitHub Desktop and signing into GitHub or GitHub Enterprise you can create and clone a tutorial repository. The tutorial will introduce the basics of working with Git and GitHub, including installing a text editor, creating a branch, making a commit, pushing to GitHub.com, and opening a pull request. The tutorial is available if you do not have any repositories on GitHub Desktop yet.

We recommend completing the tutorial, but if you want to explore GitHub Desktop by creating a new repository, this guide will walk you through using GitHub Desktop to work on a Git repository.

Part 1: Installing GitHub Desktop and authenticating your account 🔗

You can install GitHub Desktop on any supported operating system. After you install the app, you will need to sign in and authenticate your account on GitHub or GitHub Enterprise before you can create and clone a tutorial repository.

For more information on installing and authenticating, see "Setting up GitHub Desktop."

Part 2: Creating a new repository 🔗

If you do not have any repositories associated with GitHub Desktop, you will see a "Let's get started!" view, where you can choose to create and clone a tutorial repository, clone an existing repository from the Internet, create a new repository, or add an existing repository from your hard drive.

Let's get started!

Add a repository to GitHub Desktop to start collaborating

🎓 Create a Tutorial Repository...

🖥️ Clone a Repository from the Internet...

➕ Create a New Repository on your Hard Drive...

📁 Add an Existing Repository from your Hard Drive...
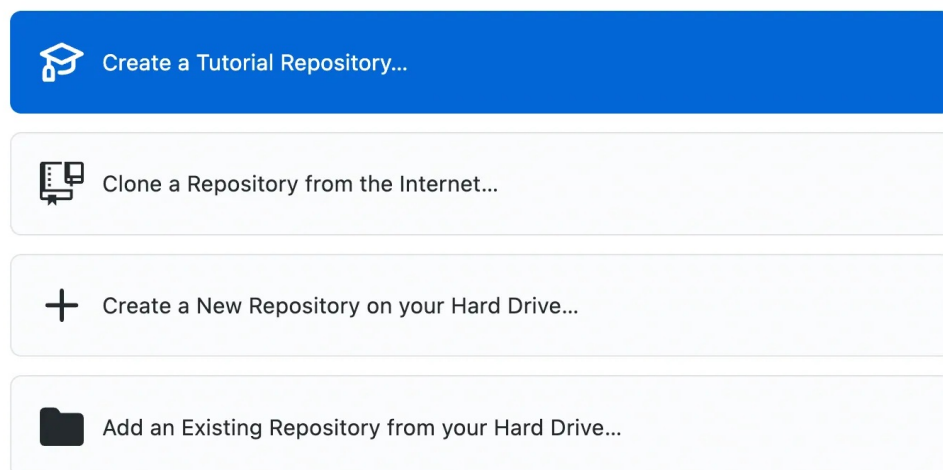
Creating and cloning a tutorial repository 🔗

We recommend that you create and clone a tutorial repository as your first project to practice using GitHub Desktop.

1 Click **Create a Tutorial Repository...**.

2 Follow the prompts in the tutorial to install a text editor, create a branch, edit a file, make a commit, publish to GitHub, and open a pull request.

Creating a new repository 🔗

If you do not wish to create and clone a tutorial repository, you can create a new repository.

1 Click **Create a New Repository on your Hard Drive...**.

2 In the "Create a New Repository" window, fill in the fields and select your preferred options.

   - "Name" defines the name of your repository both locally and on GitHub Enterprise Server.
   - "Description" is an optional field that you can use to provide more information about the purpose of your repository.
   - "Local path" sets the location of your repository on your computer. By default, GitHub Desktop creates a *GitHub* folder inside your *Documents* folder to store your repositories, but you can choose any location on your computer. Your new repository will be a folder inside the chosen location. For example, if you name your repository `Tutorial`, a folder named *Tutorial* is created inside the folder you selected for your local path. GitHub Desktop remembers your chosen location the next time you create or clone a new repository.

- **Initialize this repository with a README** creates an initial commit with a *README.md* file. READMEs helps people understand the purpose of your project, so we recommend selecting this and filling it out with helpful information. When someone visits your repository on GitHub Enterprise Server, the README is the first thing they'll see as they learn about your project. For more information, see "[About READMEs](#)."

    - The **Git ignore** drop-down menu lets you add a custom file to ignore specific files in your local repository that you don't want to store in version control. If there's a specific language or framework that you'll be using, you can select an option from the available list. If you're just getting started, feel free to skip this selection. For more information, see "[Ignoring files](#)."

    - The **License** drop-down menu lets you add an open-source license to a LICENSE file in your repository. You don't need to worry about adding a license right away. For more information about available open-source licenses and how to add them to your repository, see "[Licensing a repository](#)."

3. Click **Create repository**.

Part 3: Exploring GitHub Desktop 🔗

In the file menu at the top of the screen, you can access settings and actions that you can perform in GitHub Desktop. Most actions also have keyboard shortcuts to help you work more efficiently. For a full list of keyboard shortcuts, see "[GitHub Desktop keyboard shortcuts](#)."

## The GitHub Desktop repository bar 🔗

At the top of the GitHub Desktop app, you will see a bar that shows the current state of your repository.
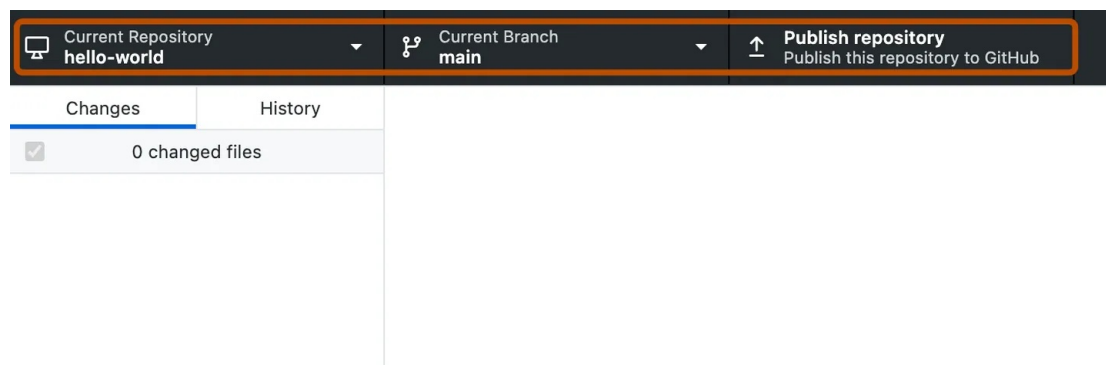
- **Current repository** shows the name of the repository you're working on. You can click **Current repository** to switch to a different repository in GitHub Desktop.
- **Current branch** shows the name of the branch you're working on. You can click **Current branch** to view all the branches in your repository, switch to a different branch, or create a new branch. Once you create pull requests in your repository, you can also view these by clicking on **Current branch**.
- **Publish repository** appears because you haven't published your repository to GitHub Enterprise Server yet, which you'll do later in the next step. This section of the bar will change based on the status of your current branch and repository. Different context dependent actions will be available that let you exchange data between your local and remote repositories.

Changes and History 🔗

In the left sidebar, you'll find the **Changes** and **History** views.
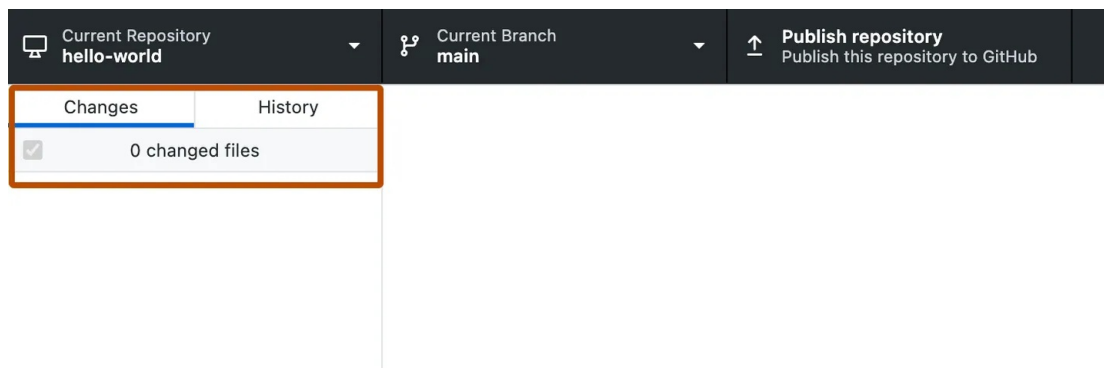
- The **Changes** view shows changes you've made to files in your current branch but haven't committed to your local repository. At the bottom, there is a box with "Summary" and "Description" text boxes and a **Commit to BRANCH** button. This is where you'll commit new changes. The **Commit to BRANCH** button is dynamic and will display which branch you're committing your changes to.
- The **History** view shows the previous commits on the current branch of your repository. You should see an "Initial commit" that was created by GitHub Desktop when you created your repository. To the right of the commit, depending on the options you selected while creating your repository, you may see *.gitattributes*, *.gitignore*, LICENSE, or README files. You can click each file to see a diff for that file, which is the changes made to the file in that commit. The diff only shows the parts of the file that have changed, not the entire contents of the file

Part 4: Publishing your repository to GitHub Enterprise Server 🔗

When you create a new repository, it only exists on your computer and you are the only one who can access the repository. You can publish your repository to GitHub Enterprise Server to keep it synchronized across multiple computers and allow other people to access it. To publish your repository, push your local changes to GitHub Enterprise Server.

1 In the repository bar, click **Publish repository**.
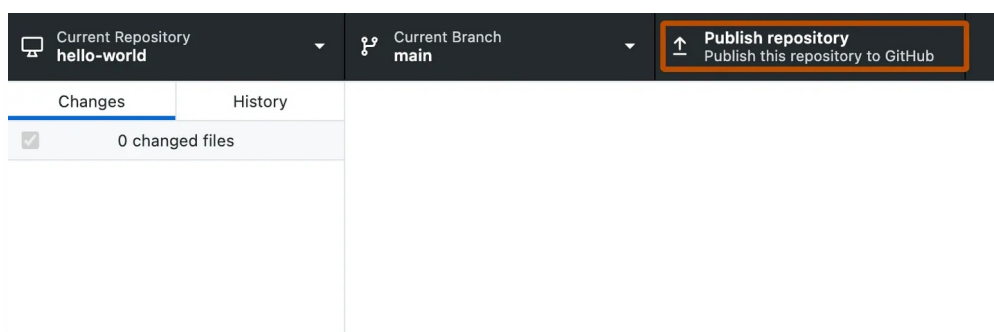
2 In the "Publish Repository" window, enter details for your new repository.

- GitHub Desktop automatically fills the "Name" and "Description" fields with the information you entered when you created the repository.
- **Keep this code private** lets you control who can view your project. If you leave this option unselected, other users on GitHub Enterprise Server will be able to view your code. If you select this option, your code will not be publicly available.
- The **Organization** drop-down menu, if present, lets you publish your repository to a specific organization that you belong to on GitHub Enterprise Server.

a.   Click **Publish Repository**.

b. You can access the repository on GitHub.com from within GitHub Desktop. In the file menu, click **Repository**, then click **View on GitHub**. This will take you directly to the repository in your default browser.

Part 5: Making, committing, and pushing changes 🔗

Now that you've created and published your repository, you're ready to make changes to your project and start crafting your first commit to your repository.

1. To launch your external editor from within GitHub Desktop, in the "GitHub Desktop" menu bar, select **Repository**, then click **Open in EDITOR**. For more information, see "Configuring a default editor in GitHub Desktop."
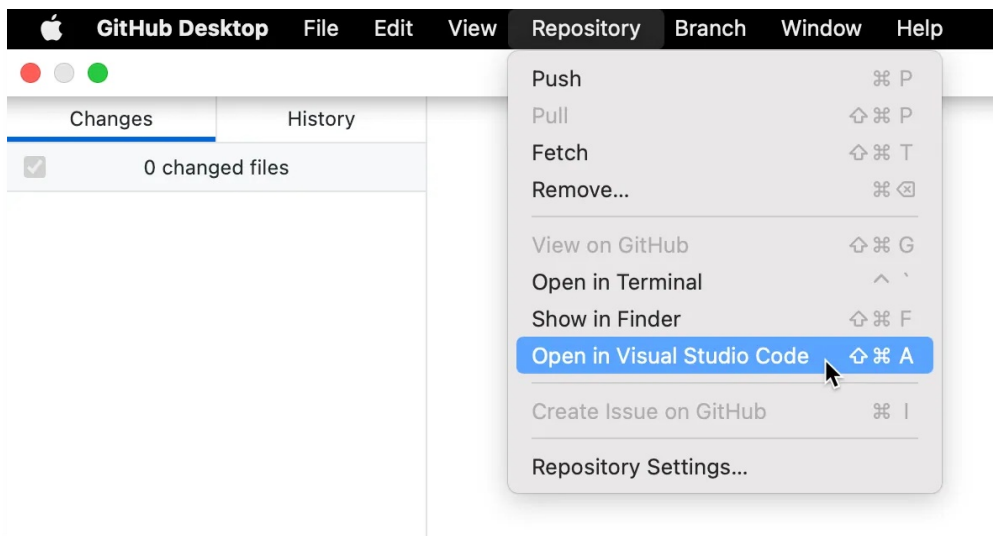

2. Make some changes to the *README.md* file that you previously created. You can add information that describes your project, like what it does and why it is useful. When you are satisfied with your changes, save them in your text editor.

3. In GitHub Desktop, navigate to the **Changes** view. In the file list, you should see your *README.md*. The checkbox to the left of the *README.md* file indicates that the changes you've made to the file will be part of the commit you make. In the future, you might make changes to multiple files but only want to commit the changes you've made to some of the files. If you click the checkbox next to a file, that file will not be included in the commit.
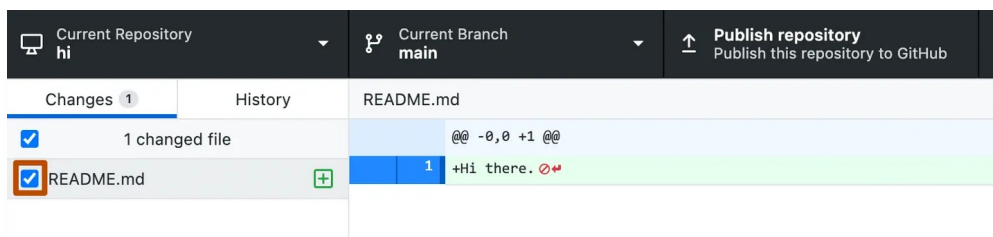

4. At the bottom of the **Changes** list, enter a commit message. To the right of your profile picture, type a short description of the commit. Since we're changing the *README.md* file, "Add information about purpose of project" would be a good commit summary. Below the summary, you'll see a "Description" text field where you can type a longer description of the changes in the commit, which is helpful when looking back at the history of a project and understanding why changes were made. Since you're making a basic update of a *README.md* file, you can skip the

description.
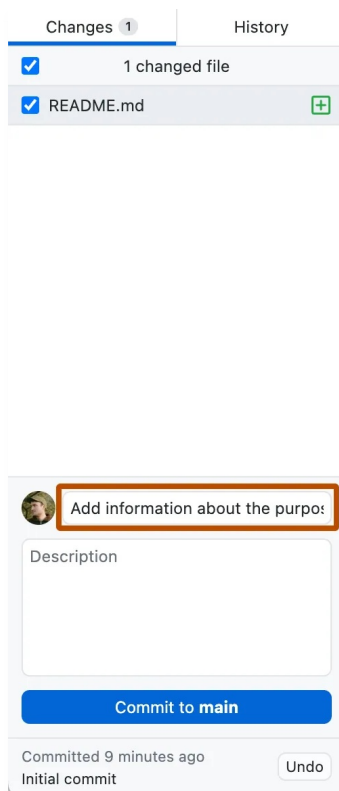
⑤ Below your commit message, click **Commit to BRANCH NAME**. The commit button shows your current branch so you can be sure to commit to the branch you want.

⑥ To push your changes to the remote repository on GitHub Enterprise Server, click **Push origin**.
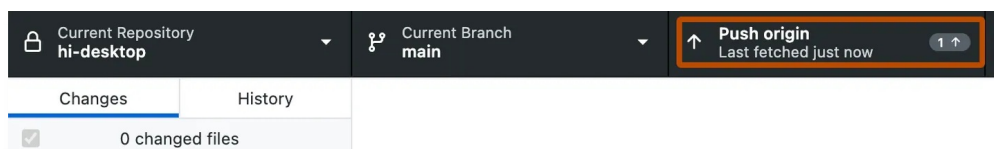
- The **Push origin** button is the same one that you clicked to publish your repository to GitHub Enterprise Server. This button changes contextually based on where you are at in the Git workflow. It should now say `Push origin` with a `1` next to it, indicating that there is one commit that has not been pushed up to GitHub Enterprise Server.
- The "origin" in **Push origin** means that you are pushing changes to the remote called `origin`, which in this case is your project's repository on GitHub.com. Until you push any new commits to GitHub Enterprise Server, there will be differences between your project's repository on your computer and your project's repository on GitHub.com. This allows you to work locally and only push your changes to GitHub.com when you're ready.

⑦ In the window to the right of the **Changes** view, you'll see suggestions for actions you can do next. To open the repository on GitHub Enterprise Server in your browser, click **View on GitHub Enterprise Server**.

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.

Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

Push origin

Open the repository in your external editor
Select your editor in Preferences
Repository menu or ⌘ ⇧ A

Open in Visual Studio Code

View the files of your repository in Finder
Repository menu or ⌘ ⇧ F

Show in Finder

Open the repository page on GitHub in your browser
Repository menu or ⌘ ⇧ G

View on GitHub

8. In your browser, click **2 commits**. You'll see a list of the commits in this repository on GitHub Enterprise Server. The first commit should be the commit you just made in GitHub Desktop.

hi-desktop Private

main ⌄   Branches   Tags   Go to file   Add file ⌄   <> Code ⌄

octocat Add information about the purpose of the project   c3fc541 12 minutes ago   ⟳ 2 commits

.gitattributes   Initial commit   27 minutes ago

README.md   Add information about the purpose of the project   12 minutes ago

Conclusion 🔗

You've now created a repository, published the repository to GitHub Enterprise Server, made a commit, and pushed your changes to GitHub Enterprise Server. You can follow this same workflow when contributing to other projects that you create or collaborate on.

Further reading 🔗

- "[Getting started with Git](#)"
- "[Learning about GitHub](#)"
- "[Get started with GitHub documentation](#)"