Creating your first repository using GitHub Desktop

You can use GitHub Desktop to quickly get to work with a Git repository without needing to use the command line.

Introduction

This guide will walk you through the process of using GitHub Desktop to work on a Git repository. GitHub Desktop extends and simplifies your GitHub.com workflow, using a visual interface instead of text commands on the command line. 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.com or GitHub Enterprise Server.

After downloading 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 an editor, creating a branch, making a commit, pushing to GitHub.com, and creating a pull request. The tutorial is available as long as you don't have any repositories on GitHub Desktop yet.

Step 1. Install and sign into GitHub Desktop

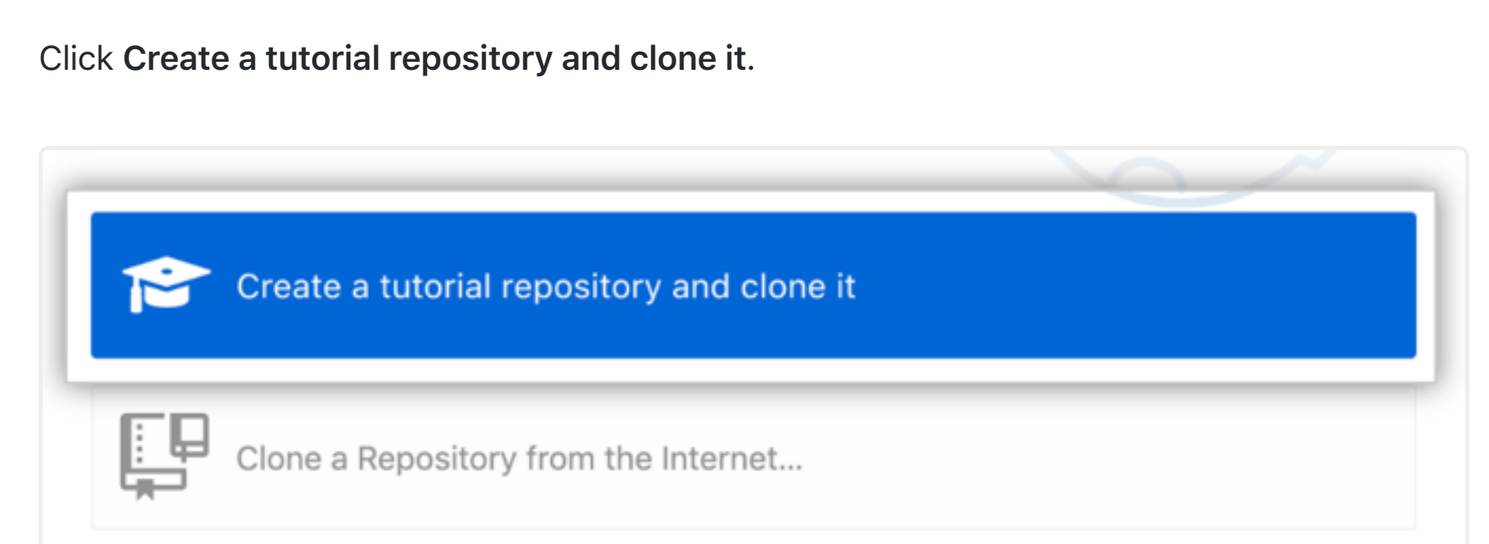
1. Download GitHub Desktop from https://desktop.github.com/. GitHub Desktop supports recent versions of Windows and macOS. For specific installation instructions for your operating system, see "Installing GitHub Desktop."
2. Launch GitHub Desktop and follow the initial welcome flow to sign into your GitHub account. You'll see a "Configure Git" step, where you can set your name and email address. To ensure your commits are correctly attributed to your GitHub account, use the email address associated with your GitHub account. For more information about commit attribution, see "Setting your commit email address."

Step 2. Create a new repository

You'll see a "Let's get started!" view, where you can choose to create and clone a tutorial repository, clone an existing repository, create a new repository, or add an existing repository.

Create and clone a tutorial repository

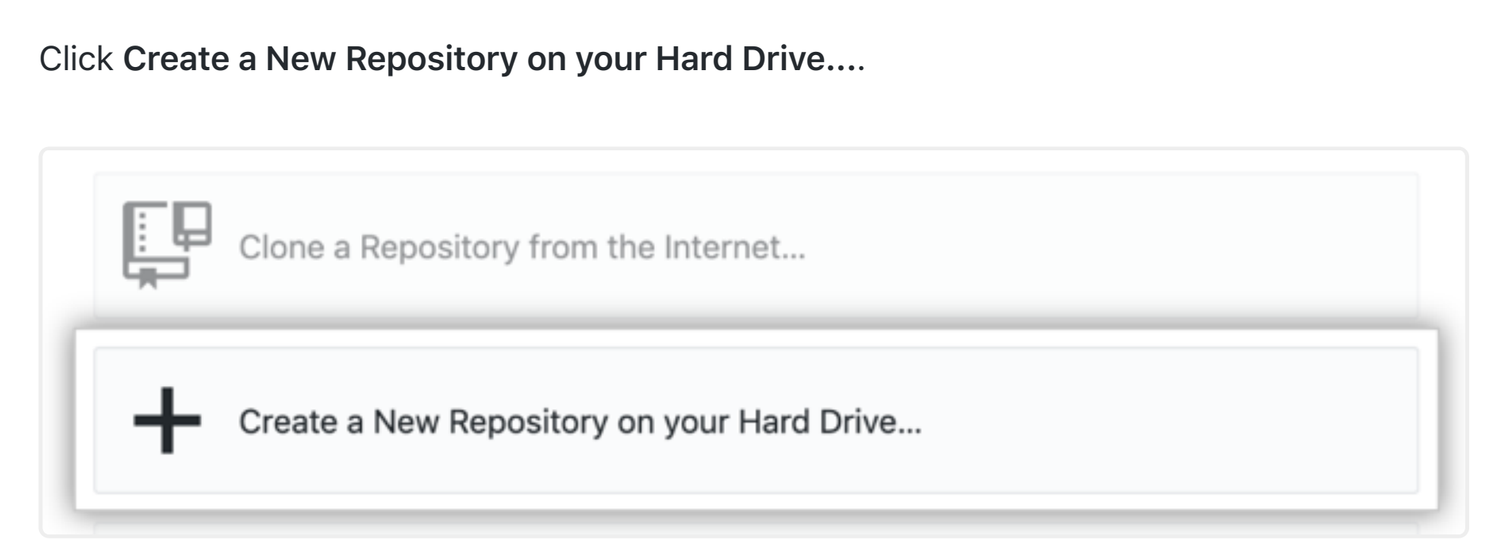
1. Click **Create a tutorial repository and clone it**.



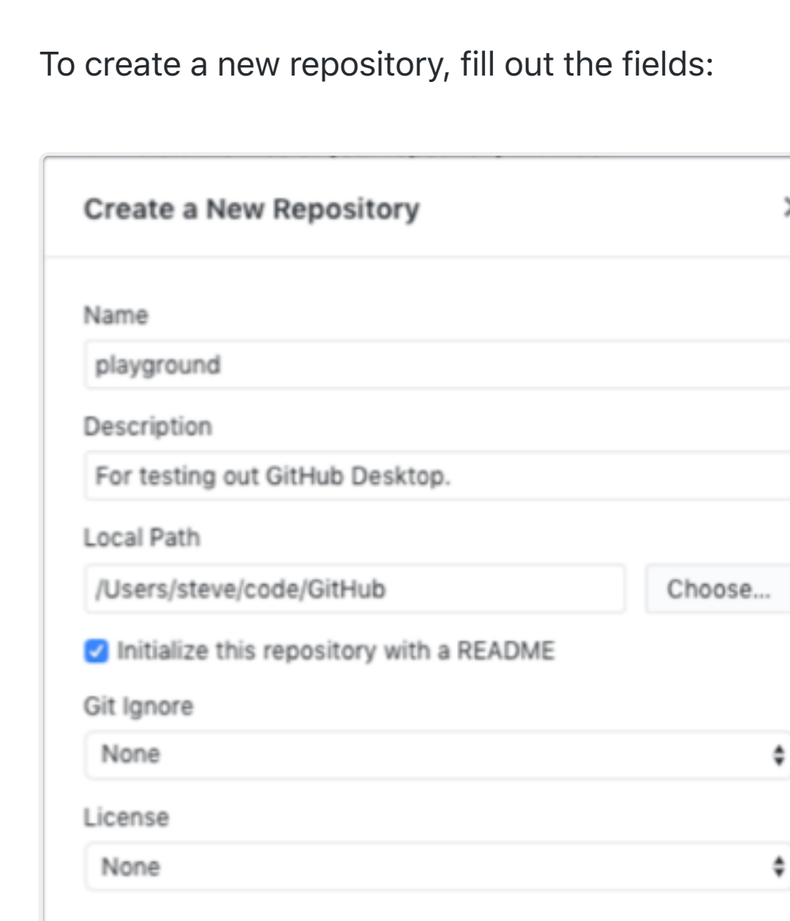
2. Follow the prompts in the tutorial.

Create a new repository

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



2. To create a new repository, fill out the fields:



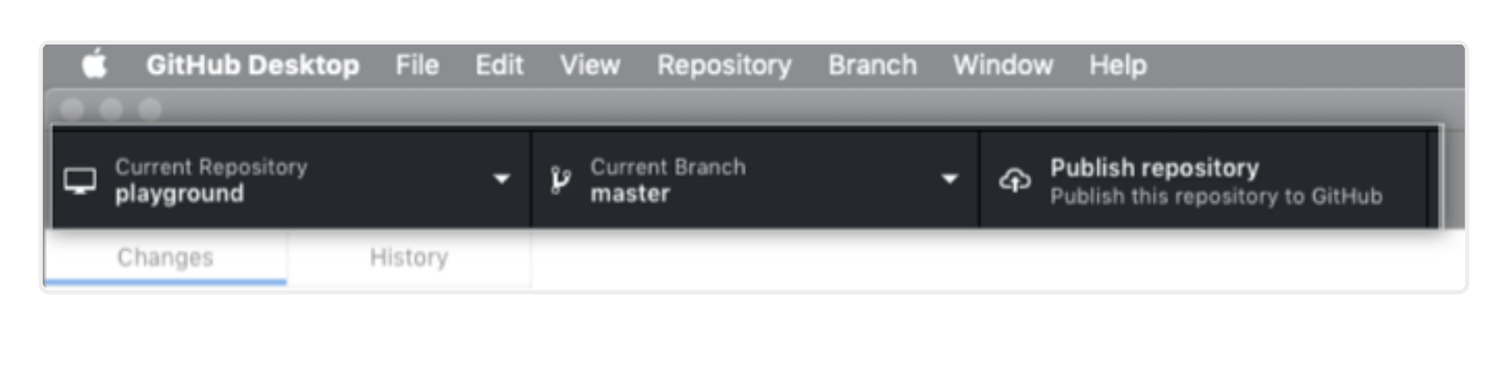
   - "Name" defines the name of your repository both locally and on GitHub.
   - "Description" is an optional field that you can use to provide more information about the purpose of your repository.
   - "Local path" sets the location of your repository on your computer. By default, GitHub Desktop creates 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, 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 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**.

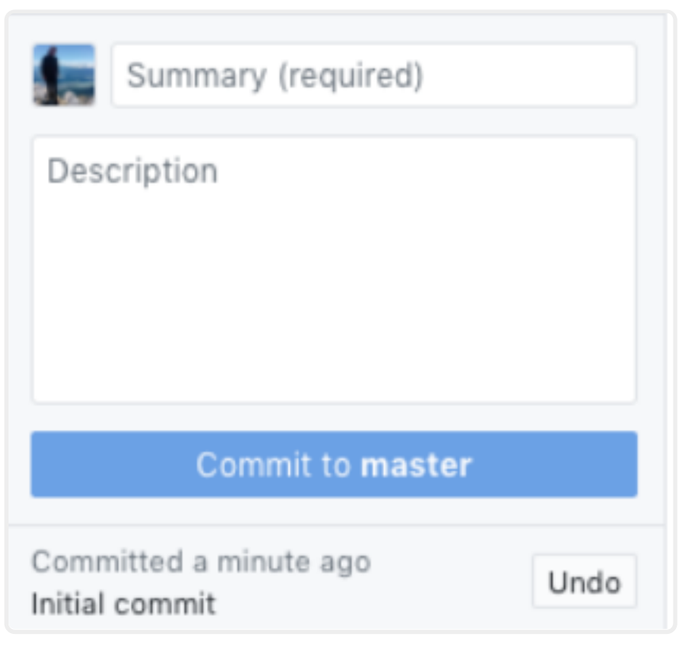
Step 3. Explore GitHub Desktop

Now that you've created a repository, you'll see the file menu at the top of the screen. This is where 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 "Keyboard shortcuts in GitHub Desktop."

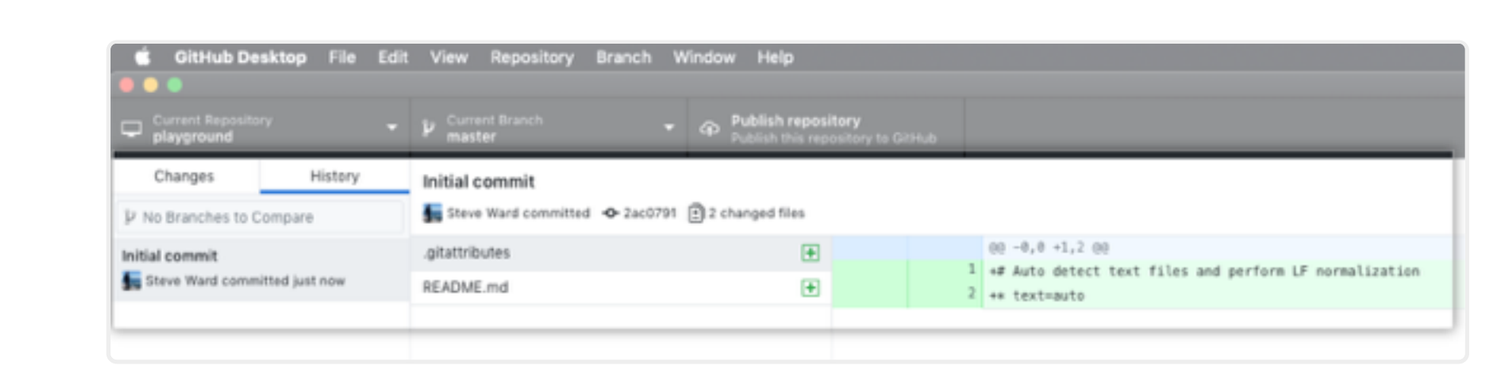
1. Below the menu is a bar that shows the current state of your repository in GitHub Desktop:
   - **Current repository** shows the name of the repository you're working on. You can click **Current repository** to switch to a different repository in GitHub Desktop.
   - **Current branch** shows the name 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 yet, which you'll do later in the next step.



2. 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, you'll also notice a box with "Summary" and "Description" text boxes and a **Commit to master** button. This is where you'll commit new changes. The **Commit** button lets you know which branch you're committing your changes to.



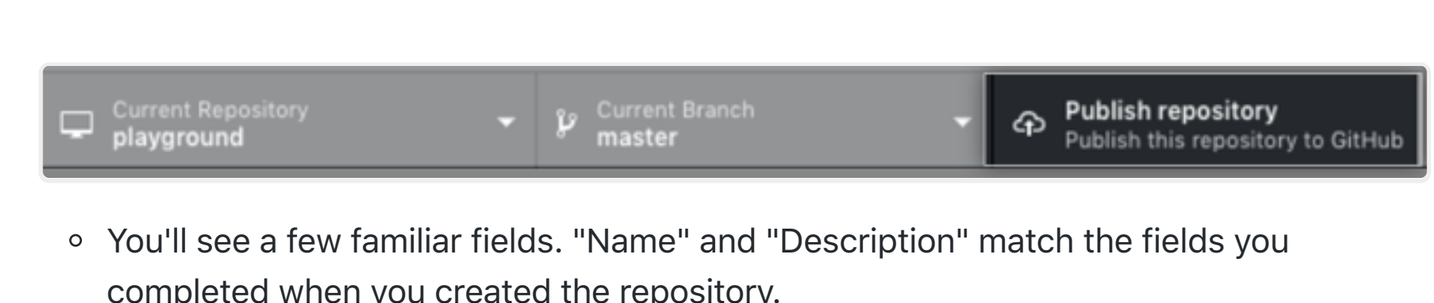
   - The **History** view shows the previous commits on the current branch of your repository. You should see an "Initial commit" that was created by GitHub Desktop when you created your repository. To the right of the commit, depending on the options you selected while creating your repository, you may 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.



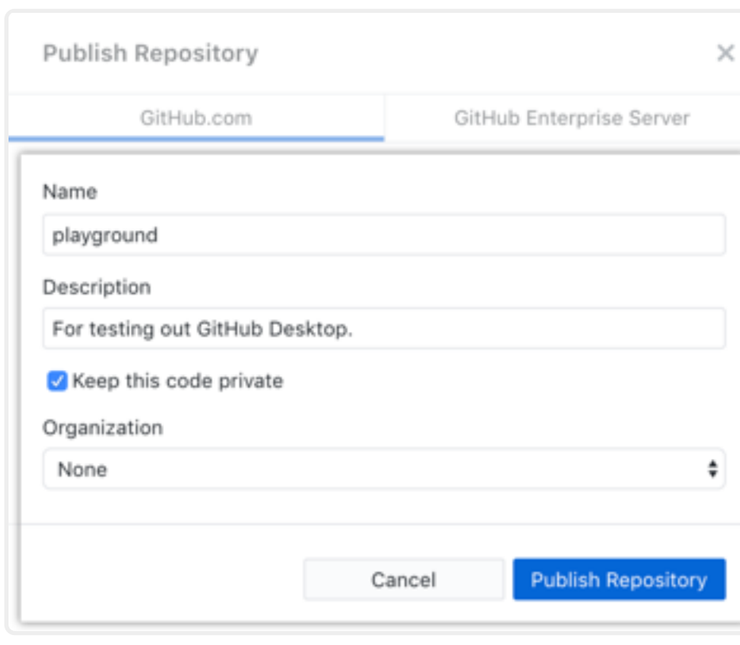
Step 4. Push your repository to GitHub

Currently, your repository only exists on your computer, and you're the only one who can access the repository. Publishing your repository to GitHub keeps it synchronized across multiple computers and team members on the same project. To publish the repository, you'll "push" it to GitHub, which makes it available on GitHub.com as well.

1. Click **Publish repository**.



   - You'll see a few familiar fields. "Name" and "Description" match the fields you completed when you created the repository.
   - You'll see the option to **Keep this code private**. Select this option if you don't want to share your code publicly with other users on GitHub.
   - The **Organization** dropdown, if present, lets you publish your repository to a specific organization that you belong to on GitHub. It's okay if you're not a member of an organization yet!



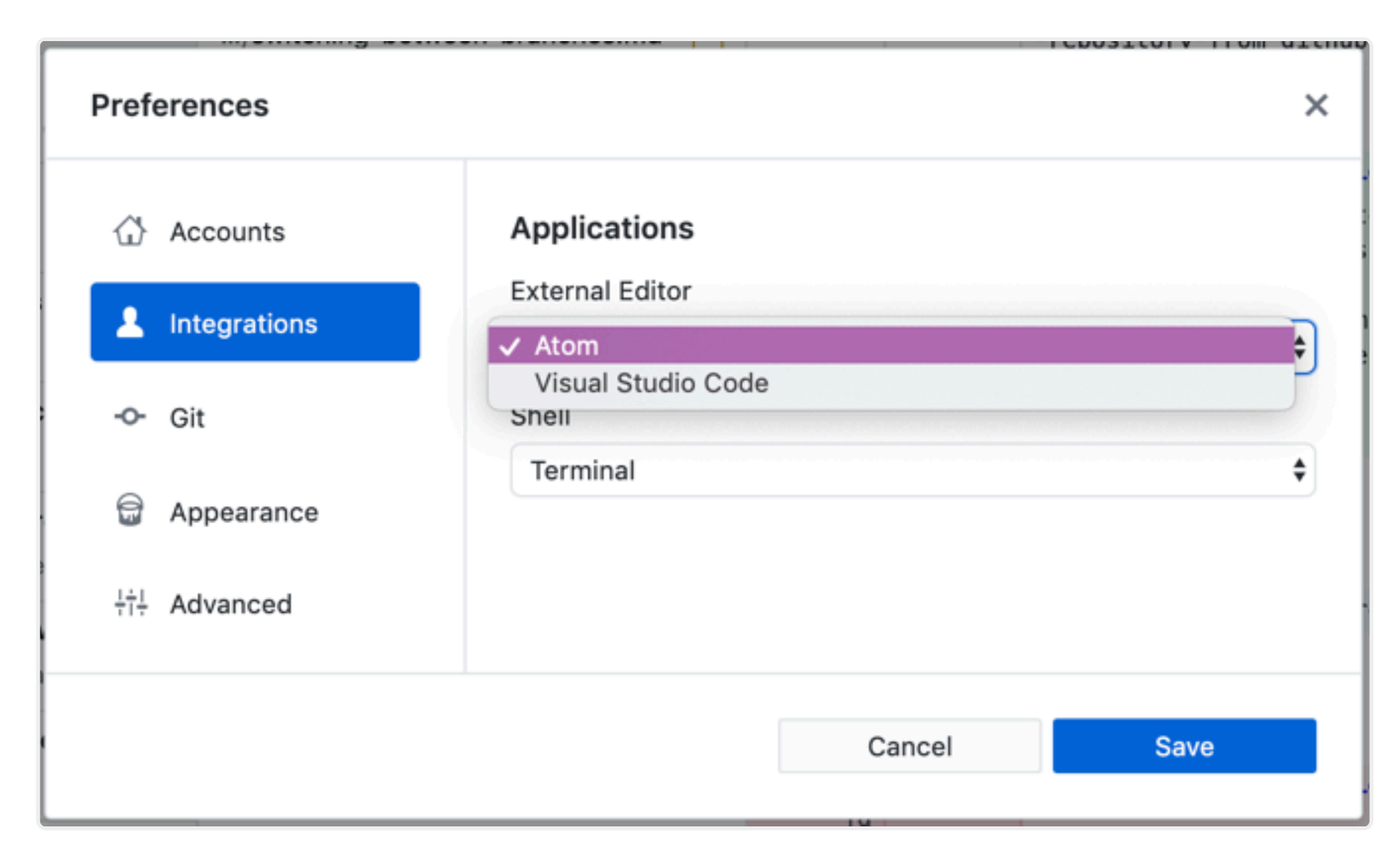
2. Click **Publish repository**.
3. You can access the repository on GitHub.com from within GitHub Desktop. In the file menu, click **Repository**, then click **View on GitHub**. This will take you directly to the repository in your default browser.

Now that your repository is published, let's go back to GitHub Desktop and make more changes to your local repository. First, we'll walk through setting up a default text editor.

Step 5. Set up a text editor

To reduce time spent setting up your development environment, you can launch a number of text editors and integrated development environments (IDEs) directly from GitHub Desktop. From a repository in GitHub Desktop, you can seamlessly open the project folder in your favorite text editor.

1. Click **File**, then click **Options**, and then click **Advanced**.
2. Use the **External editor** dropdown menu and select an editor from the list. You should see any editors you have installed in the list. If you don't see any editors, install a supported editor like Atom. For a list of supported editors, see "Open External Editor" integration in the GitHub Desktop repository.

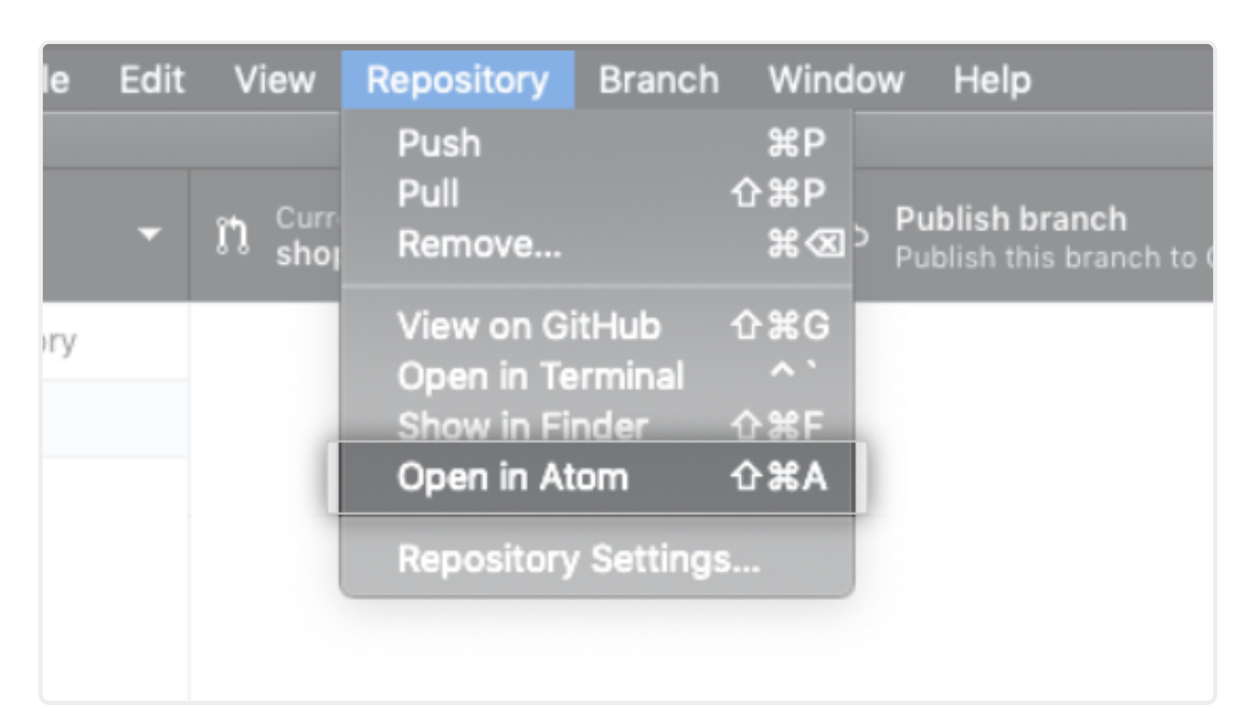


3. If you installed a new editor, restart GitHub Desktop to make the editor available in the **External editor** dropdown menu.

Step 6. Make, commit, and push changes

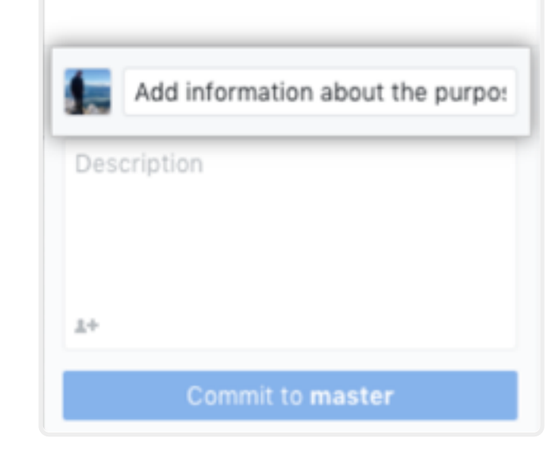
Now that you've configured a default editor, you're ready to make changes to your project and start crafting the first commit of your own to your repository.

1. To launch your external editor from within GitHub Desktop, click **Repository** and then click **Open in *EDITOR***.

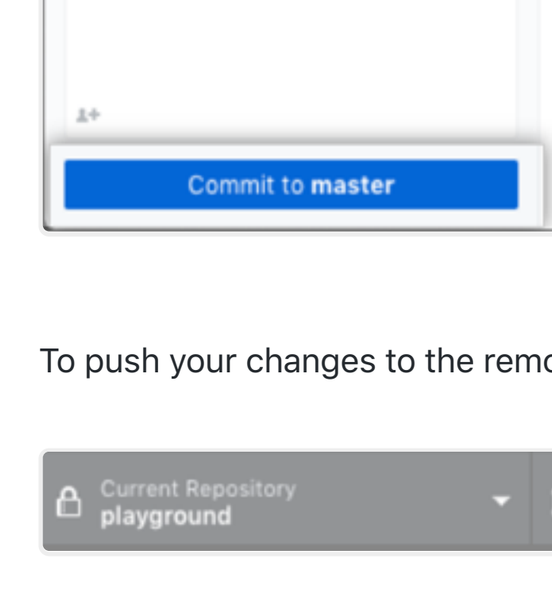


2. Start by making some changes to the *README.md* file that you previously created. Add information that describes the project like what it does and why it's useful. Remember that this is the first interaction people will have with your project. Now you're ready to make your first commit!
3. Switch from your text editor back to GitHub Desktop and navigate to the **Changes** tab. In the file list, you should see your *README.md*. The checkmark by 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. GitHub Desktop allows you to select specific changes you want to commit.

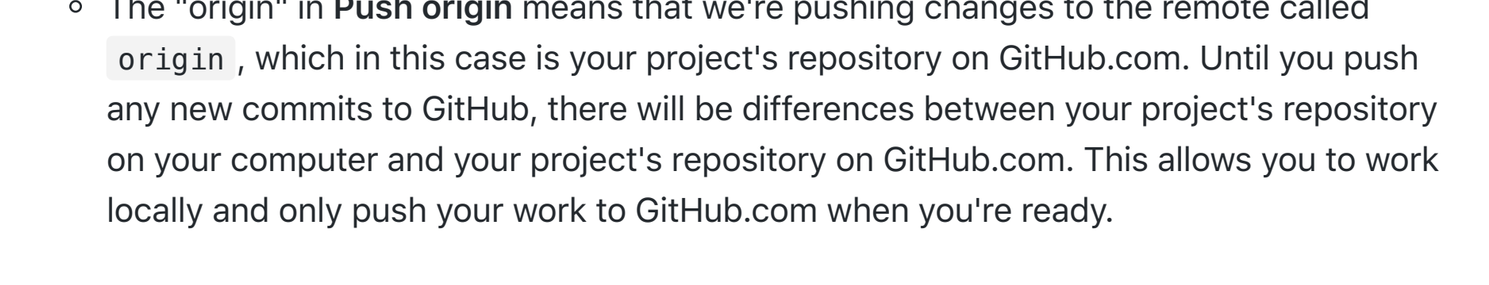
4. At the bottom of the **Changes** list, enter a commit message. To the right of your profile picture, type a short description of the commit. Since we're changing the *README.md* file, "Add information about purpose of project" would be a good commit summary. Below the summary, you'll see a "Description" 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. Click **Commit to master**. The commit button shows your current branch, which in this case is `master`, so you can be sure to commit to the branch you want.

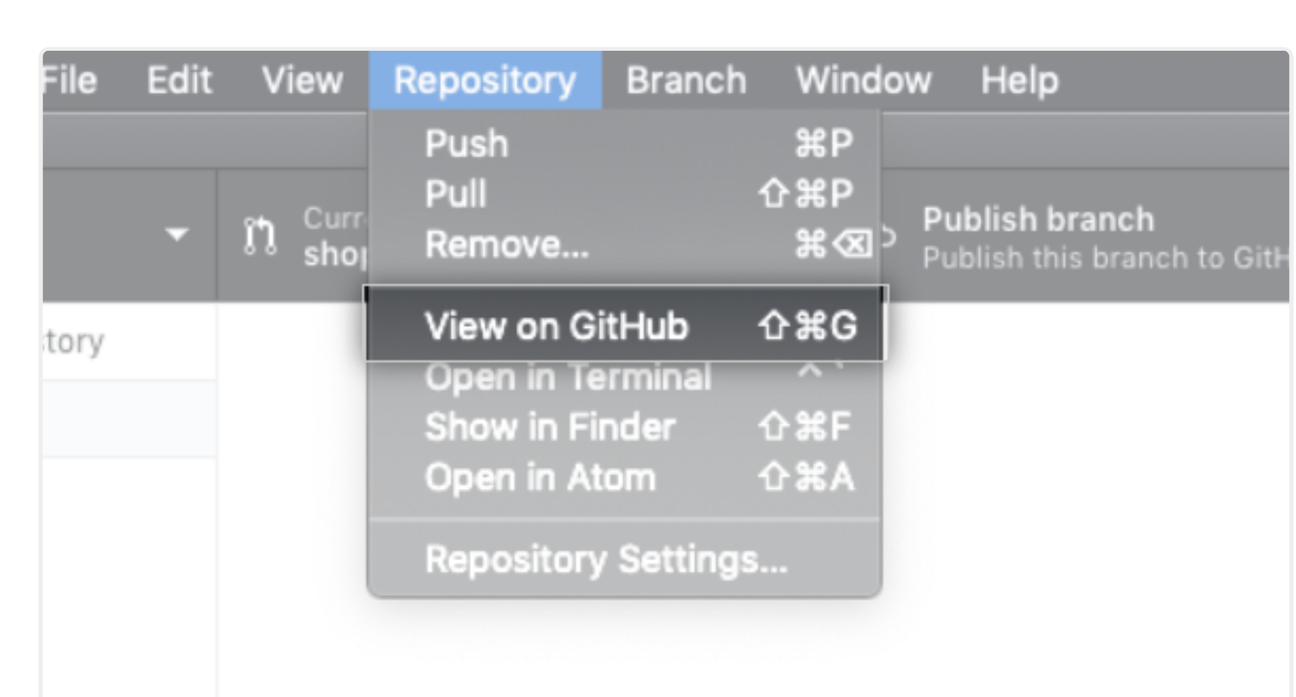


6. To push your changes to the remote repository on GitHub, click **Push origin**.

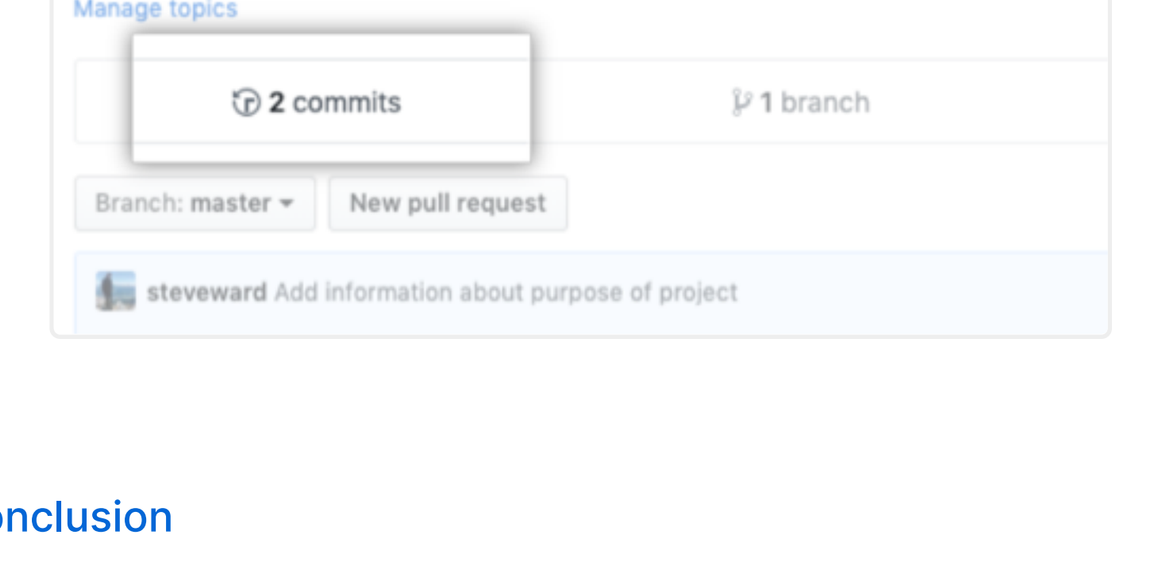


   - Remember the **Publish** button that you used to publish your repository to GitHub? It should now say `Push origin` instead, with a `1` next to it, indicating that there is one commit that has not been pushed up to GitHub.
   - The "origin" in **Push origin** means that we're pushing changes to the remote called `origin`, which in this case is your project's repository on GitHub.com. Until you push any new commits to GitHub, 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 work to GitHub.com when you're ready.

7. In the open area next to the **Changes** tab, you'll see suggestions for things you can do next. To open the repository on GitHub in your browser, click **View on GitHub**.



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



Conclusion

Congratulations! You've now created a repository, published the repository to GitHub, made a commit, and pushed your changes. We've only scratched the surface of all the things you can do with GitHub and GitHub Desktop. We hope this exercise has you excited to explore further!