First repository

Contents

Creating your first repository

Now that you have set up your environment, you are ready to start learning Git. As you progress along the guide, information and references about the concepts discussed will be provided, these will be critical in properly understanding and using Git in your workflow.

In this chapter, you will be familiarised with the basic ideas surrounding Git as well as starting to use Git.

Pre-requisite concepts

Git

Simply put, Git is a version control system. According to the article, a version control system is

> … a category of software tools that help a software team manage changes to source code over time. Version control software keeps track of every modification to the code in a special database

Git is not the first version control system but it is one of the most commonly used ones at present time. Other version control systems include SVN and Mercurial.

Essentially, a version control system tracks the "history" of changes to a piece of code over time. Git, in particular, is well-known as it allows developers to collaborate and work on a codebase without an active connection to a central code repository.

However, in order for collaboration to happen, Git requires a remote server to store this code history. There are plenty of free options available - one of which being Github which we will explore later - but there are also ways to create your own Git remote server (more details).

GitHub

GitHub is a company that provides hosting for Git. This means that any code that is uploaded to the Git server is managed and maintained by the GitHub team.

GitHub is one of the most popular version control sites out there. Other options include GitLab and BitBucket.

Now that we have gotten the basic terminology out of the way, let's get into creating your first repository.

First repository

There are two ways to create a repository. This guide will go through both in detail. There are specific use cases for either and it's best that you are familiar with them. When following along with the guide, execute the code in the first method.

GitHub first

The first approach is “GitHub first”. This is where the repository is created on GitHub (creating a remote repository first) before it is made available on your local machine. (Do not worry about the remote vs local idea for now, this will be explained in chapter 3).

1. First, login to GitHub and in the top-right corner of the menu bar, you should see a drop-down. Expand it and you will see an option for a “New Repository”. Select this option.

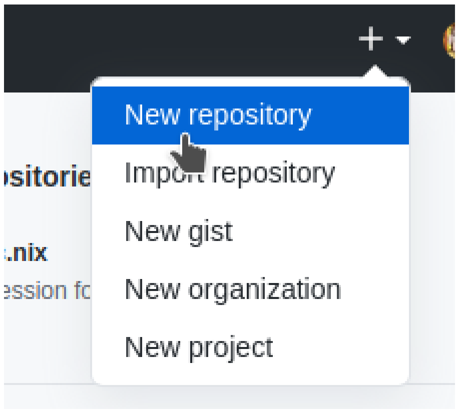
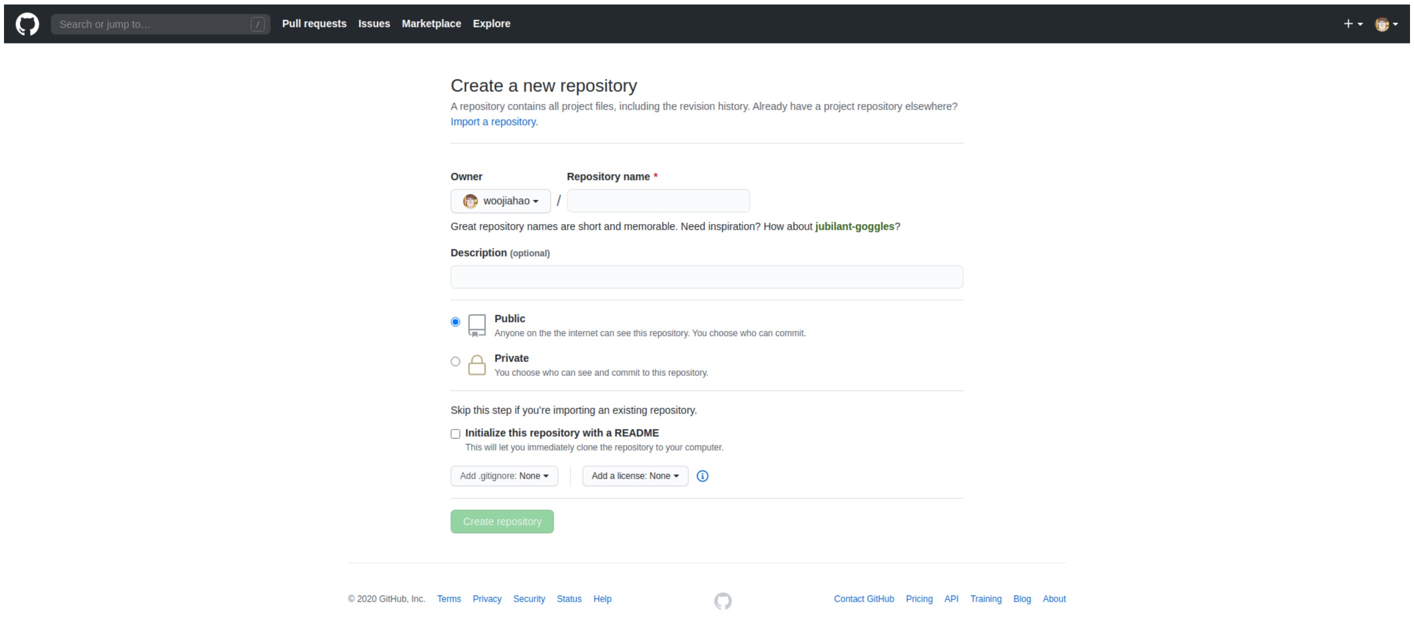


Figure 1: New repository option in GitHub

2. You will be brought to a page where you can “customise your repository”. You will have options to customise the following:



Search or jump to... Pull requests Issues Marketplace Explore

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: woojiahao / Repository name *

Great repository names are short and memorable. Need inspiration? How about jubilant-goggles?

Description (optional)

[✓ Public] Anyone on the the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Skip this step if you’re importing an existing repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None | Add a license: None

Create repository

© 2020 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

Figure 2: New repository page in GitHub

    1. Project/repository name (mandatory)
    2. Project/repository description (optional)
    3. Private vs public repository (mandatory, public by default)
    4. README file (optional)
    5. License (optional)

6. .gitignore (optional)

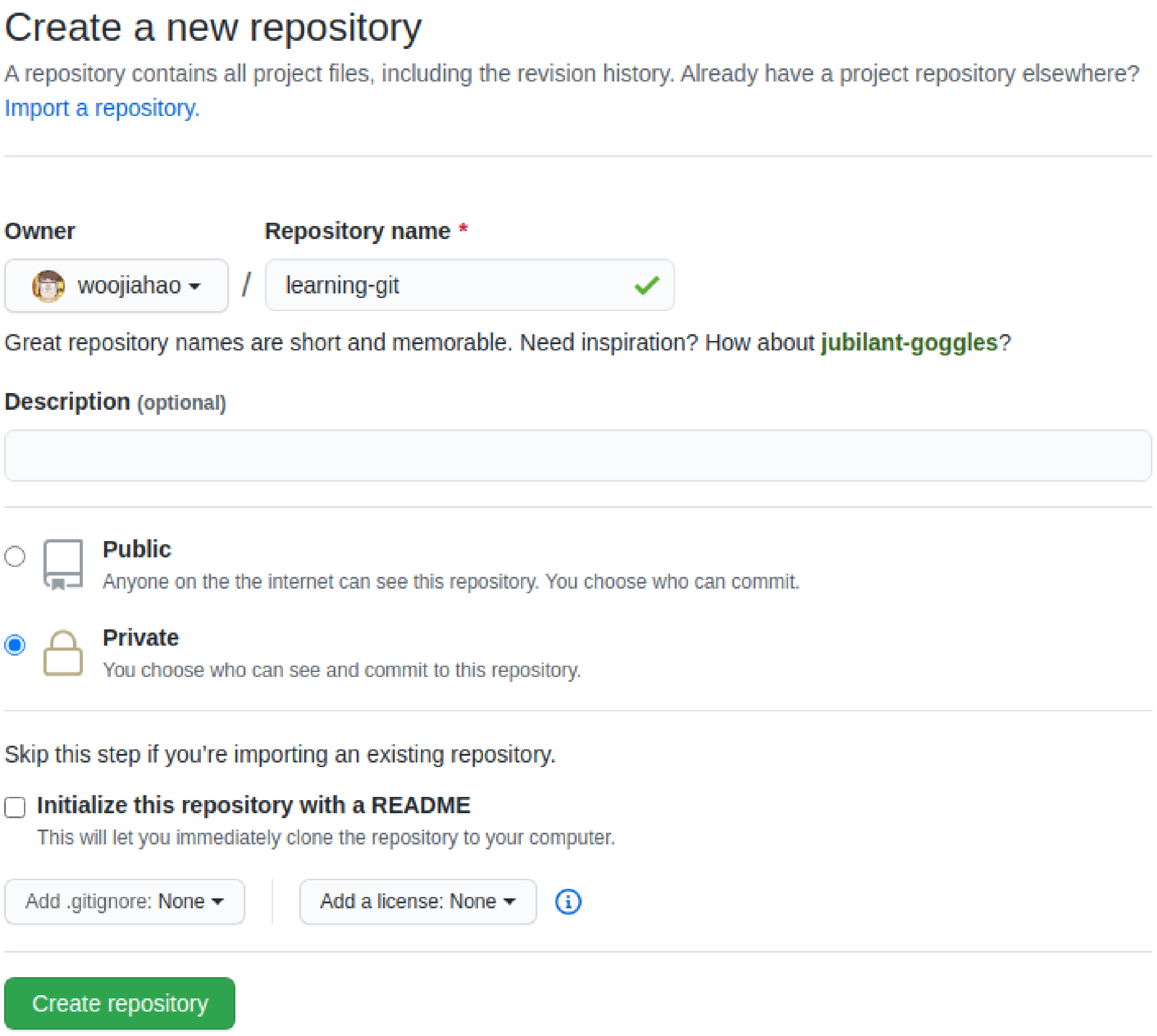
This chapter will explain the first three concepts. The remaining concepts will be elaborated upon in chapter 6.

Note* For the guide, we will be using the terms repository and project interchangeably.

Every time you wish to start a new project, you will create a new repository on GitHub. The repository's name is the name of the project that you will be working on. This can be any name that you wish to give it. However, common convention often has repository names using kebab naming convetion or pascal naming convention. For this guide, the project name chosen will be `learning-git`, but you can choose anything.

The project's description is used to provide a brief description of what the project is about. For now, you can choose to leave this alone.

Lastly, you can specify if the project is publicly-accessible or privately-accessible. For school projects or personal projects that you wish to keep under wraps, you can set the repository to private. This means that no one can view your repository unless you explicitly give them permission to do so. For open-source projects, you can set the repository to public so that other developers can view your code and use it. For this guide, this selection does not matter.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: woojiahao / Repository name *: learning-git ✓

Great repository names are short and memorable. Need inspiration? How about jubilant-goggles?

Description (optional)

○ Public — Anyone on the the internet can see this repository. You choose who can commit.

● Private — You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

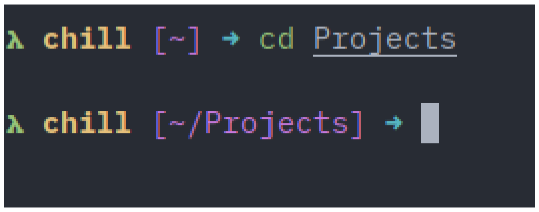
Add .gitignore: None | Add a license: None

Figure 3: New repository settings

3. Now that you have created a project on GitHub, you can then create a local copy of this project on your local machine. Open your terminal and navigate to a folder where you wish to store your project. For now, we will store all projects in a `Projects/` folder found on your user directory (`/home/<user>` for unix-based systems and `C:\Users\<user>\` for Windows).

   Note* If you are unfamiliar with using the terminal, a quick guide on the basic commands used in this guide can be found in chapter 7 so if you need to, quickly jump over to familiarise yourself with them.

   ```
   cd ~/Projects/
   ```



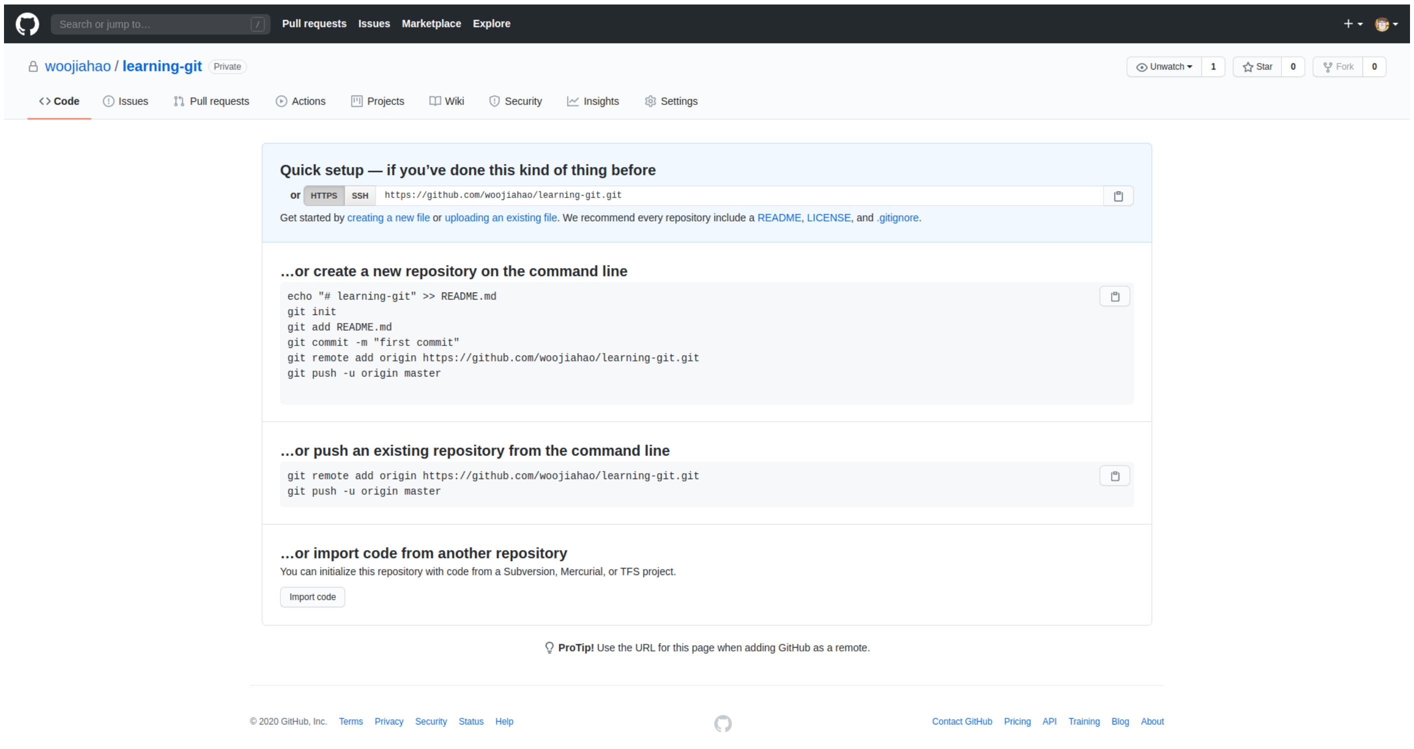
```
λ chill [~] → cd Projects
λ chill [~/Projects] →
```

Figure 4: cd

   The process of creating a local copy of the project is known as "cloning". Through cloning, you will obtain the latest version of the remote repository on your local machine. You can perform changes on the local repository and upload them to the remote repository when done.

   In your empty repository page, you will notice that a URL is provided. Copy this URL to your clipboard and go to the terminal. The URL should have the following format: `https://github.com/<username>/<project name>.git`. For example, `https://github.com/woojiahao/learning-git.git`.

   ```
   git clone https://github.com/woojiahao/learning-git.git
   ```



Quick setup — if you've done this kind of thing before

or HTTPS SSH `https://github.com/woojiahao/learning-git.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# learning-git" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/woojiahao/learning-git.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/woojiahao/learning-git.git
git push -u origin master
```

...or import code from another repository

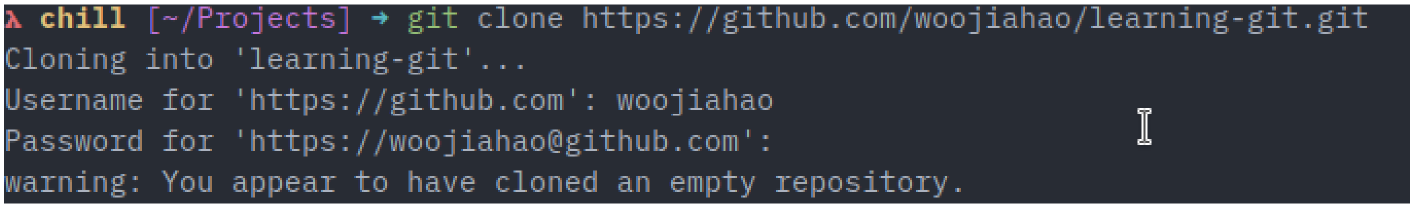
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

ProTip! Use the URL for this page when adding GitHub as a remote.

Figure 5: New repository created

   Note* You may be prompted for your GitHub credentials if you created a private repository.

You will observe some logging messages show up. Once completed, check that the project has been cloned. `ls` should display that a new folder, `learning-git`, has been created in the folder you have navigated to.



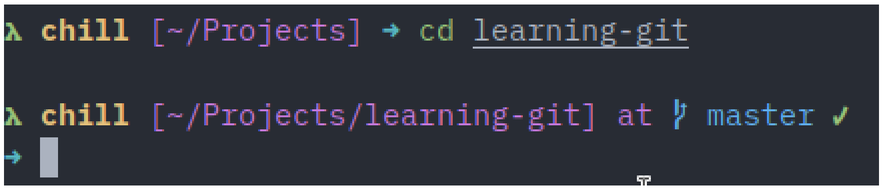
```
λ chill [~/Projects] ➜ git clone https://github.com/woojiahao/learning-git.git
Cloning into 'learning-git'...
Username for 'https://github.com': woojiahao
Password for 'https://woojiahao@github.com':
warning: You appear to have cloned an empty repository.
```

Figure 6: git clone

```
ls
```

4. Finally, you can navigate into your local copy.

```
cd learning-git/
```



```
λ chill [~/Projects] ➜ cd learning-git

λ chill [~/Projects/learning-git] at  master ✓
➜
```

Figure 7: cd into repository

Local machine first

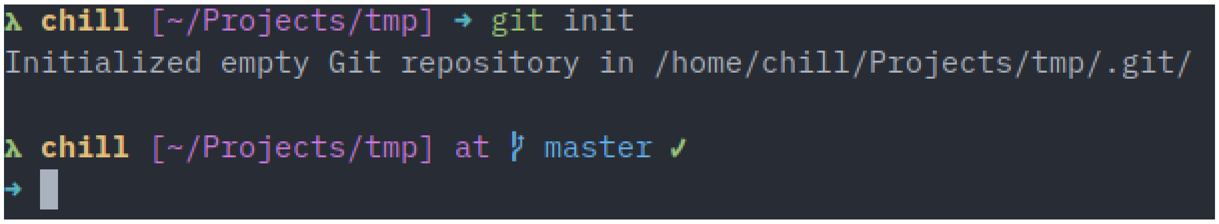
Opposed to the previous method of creating a repository, the “local machine first” approach is useful when you already have an existing project on your local machine that you wish to upload to GitHub. You are essentially performing the previous method but in reverse: starting with a local repository and then creating a remote repository based on it.

1. To start, navigate to an existing project folder on your local machine.

```
cd ~/Projects/existing-git-project/
```

2. When you’re inside the project folder, run the following command.

```
git init
```



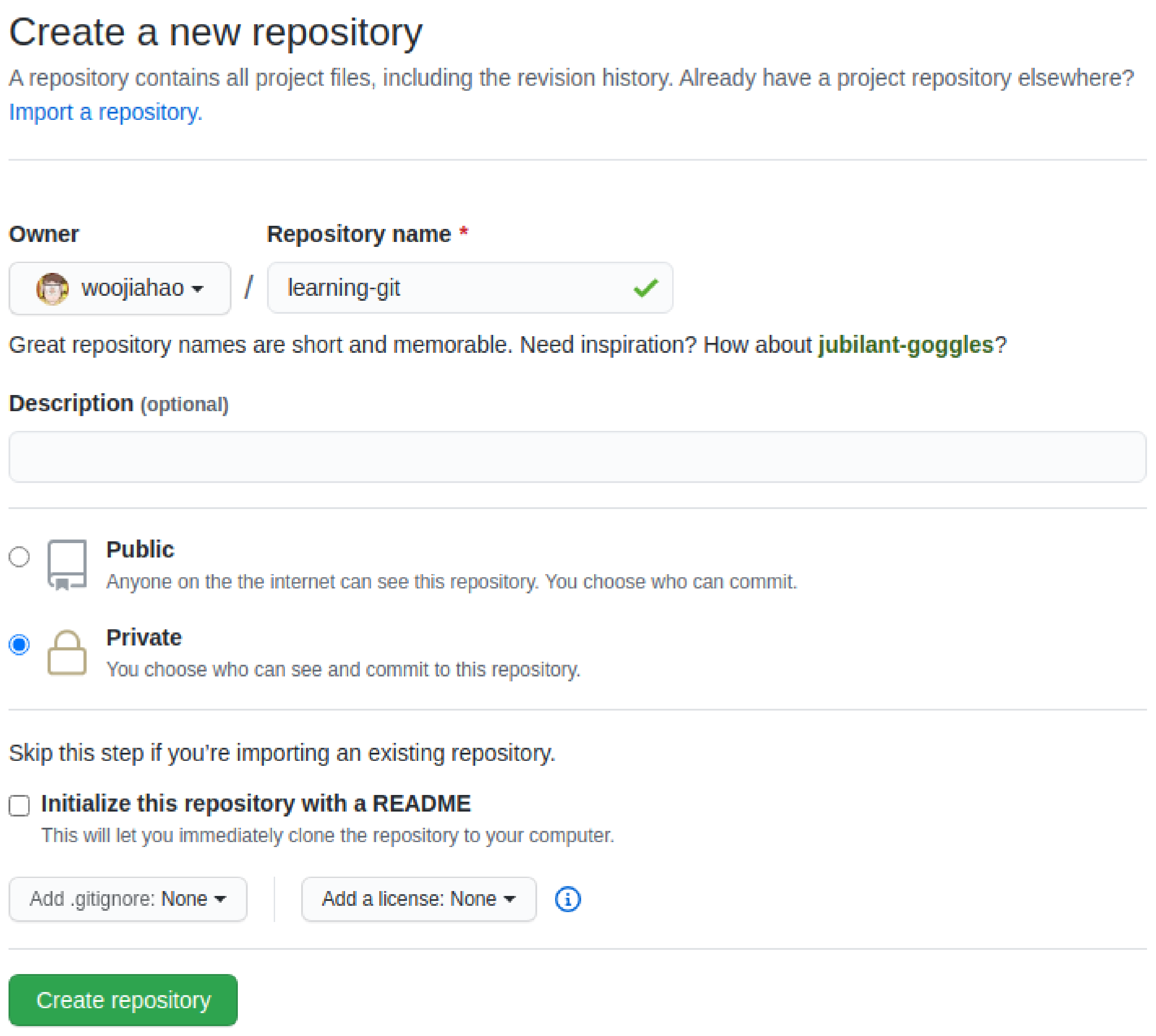
```
λ chill [~/Projects/tmp] ➜ git init
Initialized empty Git repository in /home/chill/Projects/tmp/.git/

λ chill [~/Projects/tmp] at  master ✓
➜
```

Figure 8: git init

`git init` configures the current folder to be a local Git repository. Once this is configured, you are set.

3. Now, visit GitHub and you can repeat steps 1 and 2 in the previous section. The project name can still be anything that you want. However, it is important to note that you should leave all other fields, aside from 1-3, as it is.


Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: woojiahao / Repository name *: learning-git

Great repository names are short and memorable. Need inspiration? How about **jubilant-goggles**?

Description (optional)

Public
Anyone on the the internet can see this repository. You choose who can commit.

[✓ Private]
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None | Add a license: None

Create repository

Figure 9: New repository settings

4. Once the repository is created, GitHub will again provide a URL for you to access the remote repository. Copy this URL to the clipboard and return to the terminal.
5. In the terminal, ensure that you within the local repository and execute the following commands.

```
git remote add origin <url>
git add .
git commit -m "Initial commit"
git push -u origin master
```

Let's break down these commands. More information about the specific commands will be elaborated on in the following chapters.

- `git remote add origin <url>` - links the local repository to the remote repository
- `git add .` - adds all current local files to the Git staging area
- `git commit -m "Initial commit"`- commits the staged (added) files to indicate that you are saving these changes and want to make them permanent within the history of the project
- `git push -u origin master` - uploads the current set of changes from your local repository to the remote repository, effectively making them available on GitHub. `-u` sets up a tracking branch (refer to the attached

resource for more information, this guide will not cover what tracking branches are)

And there you have it! You have successfully uploaded an existing project to GitHub. You may have some questions about what each command does; fret not, the following chapters will go over them and hopefully build on your understanding of Git.

Use case

To quickly recap the use cases of the repository creation patterns discussed above.

Pattern	Use case
GitHub first	When creating a brand new project where no code is available yet
Local repository first	When uploading an existing project to GitHub

Note that it is completely possible to use the local repository first pattern to create brand new projects. The use case mentioned above is the slightly more common approach to using the pattern.

Next