# GitHub Safe Landing Tutorial

## Introduction

Welcome to the exciting world of GitHub, a platform that will play a crucial role in your journey as a computer science student. GitHub is a hub for collaborative coding, version control, and open-source development. In this tutorial, we'll embark on a gentle journey to ensure you have a smooth landing in this fantastic realm of coding.

We want to make sure that your initial steps into GitHub are not just informative but also enjoyable. Let's get started!

## What is GitHub?

GitHub is a web-based platform serving as a central software development and collaboration hub. It provides a range of tools and features for developers to work on projects efficiently. Here are some critical aspects of GitHub:

- **Version Control**: GitHub uses a system called Git to manage and track changes to code over time. This ensures that multiple contributors can collaborate on a project without conflicts.
- **Repositories**: GitHub repositories are where code and other project files are stored. They function as a digital home for your project, and you can think of them as folders for your code.
- **Collaboration**: GitHub makes it easy for developers to work together on projects, whether they are across the room or on the other side of the world. Multiple people can contribute to a project simultaneously.
- **Transparency**: Most repositories on GitHub are public, which means anyone can see and contribute to them. This openness encourages knowledge-sharing and collaborative coding.
- **Issue Tracking**: GitHub provides a system for tracking and managing issues, which can be bug reports, feature requests, or any task related to the project.
- **Pull Requests**: Developers can propose changes to a project by creating a pull request. This requests the project's maintainers to review and merge the changes into the main codebase.

### Where are the files on GitHub saved?

All the files and code you work on in GitHub are saved in repositories. These repositories are stored on GitHub's servers, making them accessible from anywhere with an internet connection. However, you can also clone (copy) a repository to your local machine to make changes and interact with the code offline. When you're ready to share your changes or collaborate with others, you can push those changes back to the repository on GitHub.

### GitHub's Origins

GitHub was founded by **Tom Preston-Werner**, **Chris Wanstrath**, and **PJ Hyett** in April 2008. It was created to simplify and enhance the collaboration process for developers and make version control more accessible.

### Who Uses GitHub?

A diverse range of individuals, teams, and organizations widely use GitHub. It has become the backbone of open-source software development, but it's also used for private projects. Here's a snapshot of who uses GitHub:

- **Developers**: all levels use GitHub to host, collaborate, and showcase their coding projects.
- **Open-Source Communities**: Many open-source projects are hosted on GitHub. This includes some of the most popular software libraries and applications.
- **Companies**: Tech companies, startups, and businesses of all sizes use GitHub to manage their software projects, both internally and as open-source contributions.
- **Academia**: Many academic institutions and researchers use GitHub for sharing code related to their research.
- **Nonprofits**: Many nonprofit organizations host their projects on GitHub to facilitate collaboration and transparency.
- **Students**: Students like you use GitHub to work on coding assignments, showcase their skills, and collaborate on class projects.

As you continue your computer science and coding journey, you'll discover that GitHub is a valuable tool for individual and collaborative development, making it an essential platform in your coding toolkit.

## Prerequisites

Before we begin, let's make sure you have everything you need to start this adventure:

- A GitHub account (if you don't have one, you can create it here: GitHub Signup).
- Git installed on your computer (if it's not installed yet, you can download it from here: Git Downloads).
- A basic understanding of the command line is helpful but not mandatory. We'll guide you through.

Now that you're all set let's create your safe landing on GitHub.

- **Local Machine**: This is your computer. Everything you do on your computer is on your local machine.

## Tutorial

### Step 1: Setting Up Your GitHub Account

1. Start by visiting GitHub. If you already have an account, go ahead and sign in. If not, click the "Sign up" button to create your account.
2. Personalize your GitHub profile by adding a profile picture, a brief bio, and other relevant details about yourself. This will help others get to know you better in the open-source community.

### Step 2: Creating Your First Repository

A repository is like a digital home for your project. It's where your code lives and where you collaborate with others. Let's set up your first one:

1. Give your repository a name. This helps others understand your project better. In the example below, I chose 'my_first_repo.'
2. Choose the visibility of your repository. For a safe landing, let's make it public. You can always change this later if needed.
3. Click the "Create a new repository" green button, and voilà, you've got your first repository!

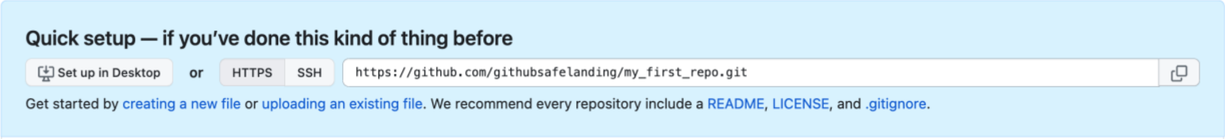
4. If all is done correctly, I can access my new repo via https://github.com/githubsafelanding/my_first_repo

   Where *githubsafelanding* is my username.

### Step 3: Clone Your Repository

Now that you have a repository on GitHub, it's time to bring it to your local machine:

1. Copy the provided URL.



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

```
https://github.com/githubsafelanding/my_first_repo.git
```

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

2. Open your terminal or command prompt and navigate to the directory to store your code.
3. Use the following command to clone your repository to your computer:

   ```
   git clone <repository-url>
   ```

   For example, in our case, we can c to some new directory via shell, and from the Terminal, we run the command:

   ```
   mkdir safe_landing
   cd safe_landing
   git clone <repository-url>
   ```

### Step 4: Making Changes

1. With your cloned repository, you can create or copy your code files into this local folder.
2. Use these commands to stage and commit your changes:

   ```
   git add .
   git commit -m 'Your commit message here'
   ```

- 'add' command add file contents (the dot means every file there is)
- 'commit' command lets you record changes to your repository

### Step 5: Pushing Changes to GitHub

Once you've committed your changes, it's time to send them back to GitHub:

```
git push
```

**Example for Steps 4,5:**

Let's Create a "HelloWorld" File and Push it to GitHub

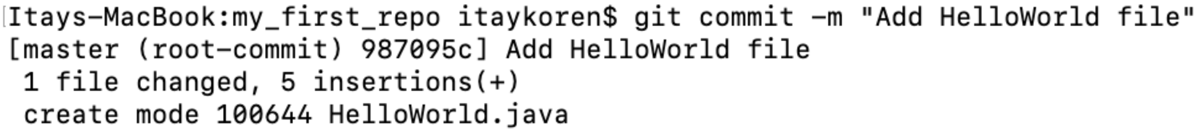
To help you understand how GitHub works in a practical sense, let's create a simple "HelloWorld" file and push it to GitHub.
We'll go through steps 4 and 5 of the tutorial.

Assuming we just did step 3. Let's create a file called HelloWorld.java and save it to our local clone

```
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

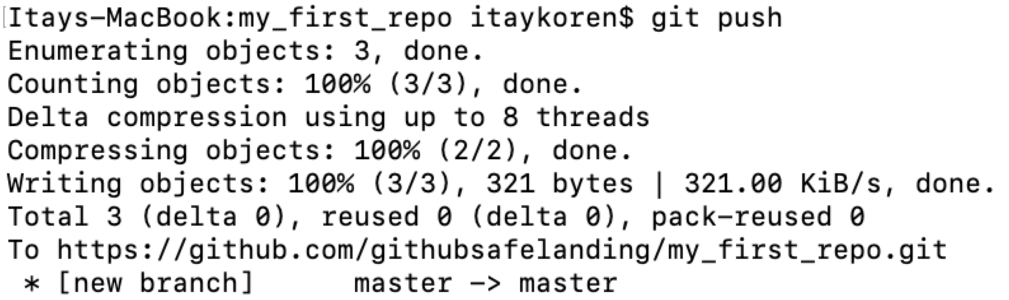
After running Git add, we can commit using an informative message.

You should see some message like that in your terminal:



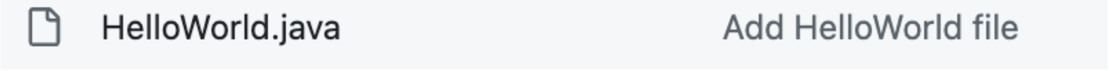
```
[Itays-MacBook:my_first_repo itaykoren$ git commit -m "Add HelloWorld file"
[master (root-commit) 987095c] Add HelloWorld file
 1 file changed, 5 insertions(+)
 create mode 100644 HelloWorld.java
```

Great! It means we are ready to push our file to GitHub



```
[Itays-MacBook:my_first_repo itaykoren$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 321 bytes | 321.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/githubsafelanding/my_first_repo.git
 * [new branch]      master -> master
```

Now, you can enter your GitHub repository URL in your browser and see your file there!



HelloWorld.java    Add HelloWorld file