Git and GitHub Setup Guide for Absolute Beginners

Introduction

This guide will take you through the basics of Git and GitHub, from installation to your first commit. By the end, you'll understand how to track changes in your projects and collaborate with others.

Step 1: Install Git

1. **Download Git**:
   - Go to the official Git website.
   - Download the installer for your operating system (Windows, macOS, Linux).
2. **Install Git**:
   - **Windows**: Run the downloaded `.exe` file and follow the installation instructions.
   - **macOS**: Use Homebrew by running `brew install git` in your terminal, or download the installer.
   - **Linux**: Use the package manager for your distribution. For example, on Ubuntu, run `sudo apt-get update && sudo apt-get install git -y`.
3. **Verify Installation**:
   - Open your terminal or command prompt.
   - Run the command `git --version`. You should see the installed version of Git.

Step 2: Configure Git

1. **Set Your Username**:

   ```
   git config --global user.name "Your Name"
   ```

2. **Set Your Email**:

   ```
   git config --global user.email "your.email@example.com"
   ```

3. **Check Your Configuration**:

```
git config --list
```

Step 3: Create a GitHub Account

1. **Sign Up**:
   - Go to GitHub.
   - Click "Sign up" and fill out the registration form.
2. **Verify Your Email**:
   - Check your email for a verification link from GitHub.
   - Click the link to verify your account.

Step 4: Create a Repository on GitHub

1. **Create a New Repository**:
   - Go to your GitHub homepage.
   - Click the "+" icon in the top-right corner and select "New repository".
2. **Fill in Repository Details**:
   - Enter a repository name.
   - Optionally, add a description.
   - Choose the repository visibility (public or private).
   - Click "Create repository".
3. **Fill in Repository Details**:
   - On your repository page, click the green "Code" button
   - Copy the HTTPS URL
   - Open Terminal (or Git Bash on Windows)
   - Navigate to where you want to store your project:

     ```
     cd path/to/your/project
     ```

   - Clone the repository:

     ```
     git clone https://github.com/yourusername/your-repo-name.git
     ```

   - git clone https://github.com/yourusername/your-repo-name.git

Step 5: Initialize a Local Repository

1. **Open Your Terminal**:
   - Navigate to the directory where you want to create your project.
2. **Initialize Git**:

   ```
   git init
   ```

3. **Create a README File**:

   ```
   echo "# MyProject" >> README.md
   ```

4. **Add the README File**:

   ```
   git add README.md
   ```

5. **Commit the README File**:

   ```
   git commit -m "first commit"
   ```

Step 6: Connect Local Repository to GitHub

1. **Copy the Repository URL**:
   - Go to your newly created repository on GitHub.
   - Click the "Code" button and copy the repository URL.
2. **Add the Remote Repository**:

   ```
   git remote add origin https://github.com/yourusername/your-repository.git
   ```

3. **Push Your Code to GitHub**:

   ```
   git push -u origin master
   ```

Step 7: Basic Git Commands

1. **Check the Status of Your Repository**:

   ```
   git status
   ```

2. **Add Changes to Staging Area**:

```
git add .
```

3. **Commit Changes**:

```
git commit -m "your commit message"
```

4. **Push Changes to GitHub**:

```
git push
```

5. **Pull Changes from GitHub**:

```
git pull
```

Step 8: Basic Git Commands

1. Navigate into your project folder:

```
cd your-repo-name
```

2. Create or modify files
3. Check the status of your changes:

```
git status
```

4. Add changes to staging:

```
git add .
```

5. Commit changes:

```
git commit -m "Your commit message"
```

Step 9: Push Changes to GitHub

1. Push your changes:

```
git push origin main
```

2. Enter your GitHub username and password if prompted

Step 10: Pull Changes from GitHub

1. To get the latest changes from GitHub:

   ```
   git pull origin main
   ```

Conclusion

Congratulations! You've set up Git and GitHub, created a repository, and learned the basic commands. With these skills, you're ready to start tracking changes in your projects and collaborating with others. Happy coding!

Git and GitHub Setup Guide for Absolute Beginners

Introduction

This guide will walk you through the basics of Git and GitHub, from installation to your first collaboration. By the end, you'll understand how to track changes in your projects and work with others using these powerful tools.

Installing Git

1. **Download Git**:
   - Visit the official Git website.
   - Download the installer for your operating system (Windows, macOS, or Linux).
2. **Install Git**:
   - **Windows**: Run the downloaded `.exe` file and follow the installation wizard.
   - **macOS**:
     - Option 1: Use Homebrew by running `brew install git` in your terminal.
     - Option 2: Download and run the macOS installer from the Git website.
   - **Linux**: Use your distribution's package manager. For Ubuntu, run:

     ```
     sudo apt-get update && sudo apt-get install git -y
     ```

3. **Verify Installation**:
   - Open your terminal or command prompt.
   - Run `git --version` to see the installed Git version.

Configuring Git

1. **Set Your Username**:

   ```
   git config --global user.name "Your Name"
   ```

2. **Set Your Email**:

```
git config --global user.email "your.email@example.com"
```

3. **Verify Configuration**:

```
git config --list
```

Creating a GitHub Account

1. Go to GitHub.
2. Click "Sign up" and complete the registration form.
3. Verify your email address by clicking the link sent to you.

Creating a Repository

On GitHub:

1. Log in to GitHub.
2. Click the "+" icon in the top-right corner and select "New repository".
3. Fill in the repository name and optional description.
4. Choose the repository visibility (public or private).
5. Click "Create repository".

Locally:

1. Open your terminal.
2. Navigate to your project directory:

```
cd path/to/your/project
```

3. Initialize a new Git repository:

```
git init
```

Basic Git Workflow

1. **Check Status**:

```
git status
```

2. **Stage Changes**:

```
git add filename    # Stage a specific file
git add .           # Stage all changes
```

3. **Commit Changes**:

```
git commit -m "Your descriptive commit message"
```

4. **Push to GitHub**:

```
git push origin main
```

5. **Pull from GitHub**:

```
git pull origin main
```

Collaborating with GitHub

1. **Clone a Repository**:

```
git clone https://github.com/username/repository.git
```

2. **Create a Branch**:

```
git checkout -b new-feature
```

3. **Create a Pull Request**:
   - Push your branch to GitHub
   - Go to the repository on GitHub
   - Click "Pull requests" > "New pull request"
   - Select your branch and provide details
   - Click "Create pull request"
4. **Review and Merge**:
   - Reviewers will check your code
   - Once approved, merge the pull request on GitHub

Advanced Topics

1. **Using .gitignore**:
   Create a `.gitignore` file in your project root to specify files and directories Git should ignore.
2. **SSH Key Setup**:
   For secure, password-less authentication with GitHub.
3. **Git Branching Strategies**:
   Learn about different branching models for efficient collaboration.
4. **Git Hooks**:
   Automate tasks with scripts that run at certain points in Git's execution.

This reorganized guide provides a more structured approach to learning Git and GitHub, starting with the essentials and progressing to more advanced topics. It includes all the key information from the original content while improving the flow and organization.

GitHub SSH Key and CLI Setup Guide - Advance

Adding Your Credentials to Your Local Machine

To add your credentials to your Linux, Mac, or Windows machine and make it trusted from your GitHub account using Git and SSH, follow these step-by-step instructions:

1. **Open a Terminal/Command Prompt on Your PC**
2. **Set Git Username and Email**

   ```
   git config --global user.name "your_git_username"
   git config --global user.email "your_email@example.com"
   ```

3. **Generate a New SSH Key Pair**

   ```
   ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
   ```

   Replace `your_email@example.com` with the email address associated with your GitHub account.
4. **Store the Key Pair**
   Press Enter to accept the default location for storing the key pair (`~/.ssh/id_rsa`). You can provide a custom path if desired.
5. **Enter a Secure Passphrase**
   Enter a secure passphrase when prompted, or leave it blank if you don't want to use a passphrase (not recommended).
6. **Add the New SSH Key to the ssh-agent**

   ```
   eval "$(ssh-agent -s)"
   ssh-add ~/.ssh/id_rsa
   ```

   Enter your passphrase (if you set one) when prompted.
7. **Copy the Public Key to Your Clipboard**

   ```
   cat ~/.ssh/id_rsa.pub
   ```

   Select and copy the entire output (which starts with `ssh-rsa` and ends with your email address).
8. **Add the SSH Key to Your GitHub Account**

- Open your GitHub account in a web browser and navigate to the "Settings" page.
- Click on "SSH and GPG keys" in the left sidebar.
- Click on "New SSH key" and provide a descriptive title (e.g., "Linux PC").
- Paste the copied public key into the "Key" text area.
- Click "Add SSH key" to save the new key.

9. **Verify the SSH Connection**

   ```
   ssh -T git@github.com
   ```

   If you see a message like
   `Hi username! You've successfully authenticated, but GitHub does not provide shell access.` ,
   your SSH key has been added successfully.

Your Linux machine is now trusted by GitHub using the installed Git and SSH. You can use Git over SSH for securely communicating with GitHub repositories without needing to enter your GitHub credentials every time.

Using GitHub CLI (gh)

Installing GitHub CLI

1. **Install Git**: If you haven't already, download and install Git from git-scm.com/downloads.
2. **Install GitHub CLI (gh)**: You can install the GitHub CLI using a package manager or by downloading a binary from the GitHub repository.
   Using a Package Manager:
   - **macOS** (with Homebrew):

     ```
     brew install gh
     ```

   - **Windows** (with Chocolatey):

     ```
     choco install gh
     ```

   - **Linux** (with APT):

     ```
     sudo apt-get install gh
     ```

   Downloading a Binary:
   - Visit the GitHub CLI releases page.
   - Download the binary for your operating system and architecture.

- Unzip the downloaded file.
- Move the `gh` binary to a directory included in your system's `PATH`.

3. **Verify the Installation**

   ```
   gh --version
   ```

   This should print the version of the GitHub CLI that you have installed.

Authenticating with GitHub CLI

1. **Start Interactive Setup**

   ```
   gh auth login
   ```

2. **Authenticate Using a Token**

   ```
   gh auth login --with-token < mytoken.txt
   ```

   Replace `< mytoken.txt` with the path to your token file.

3. **Authenticate with a Specific Host**

   ```
   gh auth login --hostname enterprise.internal
   ```

4. **Use SSH Protocol**

   ```
   gh auth login --git-protocol ssh
   ```

5. **Verify Authentication Status**

   ```
   gh auth status
   ```

Cloning and Pushing Repositories

Cloning a Repository

1. **Clone the Repository**

   ```
   git clone https://github.com/username/repo.git
   ```

2. **Navigate to the Cloned Repository**

```
cd repo
```

Creating and Pushing a New Branch

1. **Create a New Branch**

   ```
   git checkout -b new_branch_name
   ```

2. **Make Changes and Commit**

   ```
   git add .
   git commit -m "Your commit message"
   ```

3. **Push the New Branch**

   ```
   git push origin new_branch_name
   ```

Creating a Pull Request

1. **Create a Pull Request on GitHub**
   Navigate to your repository on GitHub, and you should see a prompt to create a new pull request. Click on the "Compare & pull request" button.
2. **Select Reviewers**
   In the right sidebar of the pull request page, click on "Reviewers" and select the appropriate reviewers from the list of collaborators or teams.
3. **Submit the Pull Request**
   Provide a title and description for your pull request, then click on the "Create pull request" button.

Updating Local Repository and Pushing Changes

1. **Check Current Branch**

   ```
   git branch
   ```

2. **Switch to the Branch**

   ```
   git checkout new_branch_name
   ```

3. **Add and Commit Changes**

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

4. **Push Changes to GitHub**

```
git push origin new_branch_name
```

5. **Verify Changes on GitHub**
   Go to your GitHub repository and navigate to the branch to verify that the changes have been pushed successfully.

Merging via Command Line

1. **Update Local Repository**

```
git pull origin main
```

2. **Switch to the Base Branch**

```
git checkout main
```

3. **Merge the Head Branch**

```
git merge new_branch_name
```

4. **Push the Changes**

```
git push -u origin main
```

Undoing a Merge

Reverting a Merge Commit

1. **Checkout the Main Branch**

```
git checkout main
```

2. **Revert the Merge Commit**

```
git revert <commit-id>
```

Replace `<commit-id>` with the commit ID of the merge commit.

3. **Push Changes to GitHub**

```
git push origin main
```

Resetting the Main Branch

1. **Identify the Commit Before the Merges**
   Use `git log` to view the commit history and identify the commit you want to revert to.
2. **Checkout the Main Branch**

```
git checkout main
```

3. **Reset the Main Branch**

```
git reset --hard <commit-id>
```

   Replace `<commit-id>` with the commit ID of the main branch before any merges were made.
4. **Force Push Changes to GitHub**

```
git push origin main --force
```

   Note: Be cautious when using `--force` as it overwrites the history of the main branch on the remote repository.
5. **Confirm Reset on GitHub**
   Verify that the main branch has been reset to its state before any merges were made.

This guide should help you get started with setting up Git and GitHub CLI, as well as managing repositories and branches efficiently.