Myscape

TECH

GitHub Version Control Quick Start Guide

Overview

When working on a computational project it can be very helpful to keep track of changes that are made by using a developer platform like **GitHub (https://github.com/)**. Project collaboration often entails several iterations and changes to the code being developed.

Objectives

After completing this tutorial you will know how to:

- download a GitHub repository in the terminal
- modify Git files for a repository in the terminal

- create your own repository on the GitHub website
- make changes to your repository in the terminal and online

About GitHub & Git

Git is a version control system that is used by developers to keep track of code changes. The GitHub web-based hosting service allows users to perform version control of code changes using Git repositories.

GitHub is a great place to store and share code for project collaboration. It is also a good way to make code publicly available.

Additionally, GitHub is the biggest and most used development platform. GitHub gives users the ability to conduct bug tracking, access control, software feature requests, task management, continuous integration, and create wikis for every project.

Getting GitHub & Git

First, make sure that you have created a **GitHub account (https://github.com/)**.

Next, check that Git it installed by opening the terminal and entering the following command.

```
git version
```

The output will either tell us which version of Git is installed, or it will alert us that git is an unknown command. If it's an unknown command, **read further and find out how to install Git (https://github.com/git-guides/install-git)**.

When we use Git on a new computer for the first time, it is good to configure settings like our user name and email for GitHub :

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

The user name and email specified in the git config command will now be associated with all Git activity, such as any changes we make and add to a code repository.

Enter the following command to check your git config settings.

```
git config --list
```

The output of the above command should list your user name and email, if you set your user name and email using the git config command.

Downloading GitHub Repositories

The GitHub repository with the code we will need is located **HERE (https://github.com/ElizabethBrooks/wpExampleRepository)**. To download the code onto a local computer or server space, click the green code button near the top of the page, and copy the link. Then, in the terminal:

```
git clone https://github.com/ElizabethBrooks/wpExampleRepository.git
```

Modifying Git Repositories

The wpExampleRepository repository is not owned by your GitHub account. If we want to make changes to this repository, we need to create our own GitHub repository and add the downloaded code or **create a fork (https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/about-forks)** of the original repository.

It can be simpler to add the downloaded code to our own GitHub repository. First, use the cd command to navigate into the folder for the repository that we downloaded using the git clone command.

```
cd wpExampleRepository
```

We need to remove the hidden .git folder that stores all of the git information for the repository we downloaded, since we want to make our own repository with the code that was downloaded.

It is possible to see the hidden folder using the ls command with a flag like so.

```
ls -a
```

We can remove the hidden folder with the rm command by using a couple of flags.

```
rm -rf .git
```

The **r** flag allows us to recursively delete the contents of the folder. The **f** flag forces the removal of hidden folders and files.

Now we can initialize a new repository using the git init command in the terminal, which will prepare the necessary hidden Git files for our new repository.

```
git init
```

This should output a message stating that there was an “initialized empty Git repository” and the file path of the repository.

Now we can use the git status command to check what files we need to add to our own GitHub repository.

```
git status
```

This outputs information about the repository branch, commits, and untracked files.

Any files that we downloaded from the original repository using the git clone command need to be added to the newly initialized repository. In this case, only one file needs to be added.

```
git add README.md
```

Next, we need to add a message to the commit that notes the type of changes that were made.

```
git commit –m "Adding initial files"
```

This should output the repository branch name, the commit message, the number of files changed, and the number and types of changes.

The wpExampleRepository directory is now setup with Git files that track the changes that we made and include a message for those changes.

Adding a SSH Key

In order to easily add changes to a GitHub repository from your terminal, you will need to **generate and add a SSH key (https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent)** associated with your computer to your GitHub account.

First, generate an SSH key using the following command in the terminal using your GitHub email address.

```
ssh-keygen -t ed25519 -C "your@email.com"
```

This should state that a public/private key pair is being generated and request a file to save the key in followed by a passphrase. In this example we'll press enter to save the file in a default location, press enter again to not set a passphrase, and press enter again to confirm the empty passphrase.

Next, start the ssh-agent in the background.

```
eval "$(ssh-agent -s)"
```

This will show the agent pid number.

Now we can add the SSH private key to the ssh-agent.

```
ssh-add ~/.ssh/id_ed25519
```

This will output the identity the was added, including the file path to the SSH ID and the email that was attached.

We can view the SSH key as follows.

```
cat ~/.ssh/id_ed25519.pub
```

Copy the output text, which should begin with “ssh-ed25519” and end with your GitHub email.

Finally, we can add the key to our GitHub account.

1. Navigate to the **GitHub (https://github.com/)**website
2. Click on your profile picture in the upper left corner
3. Click “Settings”
4. Click “SSH and GPG keys” in the left side bar
5. Click the green “New SSH key” button in the upper right corner
6. Enter a name for the key in the “Title” box
7. Paste the key that we copied from the terminal into the “Key” box
8. Click the green “Add SSH key” button

The newly added SSH key should appear on the list of “Authentication keys” in the “SSH keys” section of the page.

Creating GitHub Repositories

We need to create a new repository on the GitHub website before we can connect the folder that we downloaded and modified to our personal repository.

1. Navigate to the **GitHub (https://github.com/)**website
2. Click on your profile picture in the upper left corner
3. Click “Your repositories”
4. Click the green “New” button near the top right side of the page
5. Enter a name for the repository
6. Click “Create repository”
7. Click “SSH” in the blue “Quick setup — if you’ve done this kind of thing before” section of the page

We will now be following the steps specified in the "…or push an existing repository from the command line" section of the Github repository page.

First, we need to specify the remote location for the repository using the URL we just copied for the new GitHub repository. The command to specify the remote location should look similar to the following.

```
git remote add origin git@github.com/YourUserName/YourRepository.git
```

Note that if the HTTPS version of the repository URL was used, the upcoming git push command will report an error with a message stating that "Support for password authentication was removed on August 13, 2021." It is necessary to **generate and add a SSH key (https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent)** to your GitHub account in order to make changes to a GitHub repository using the terminal.

Next, we need to specify the **default branch (https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-branches# about-the-default-branch)** to which we are adding changes.

```
git branch –M main
```

Finally, the changes need to be pushed to the master branch of the remote repository.

```
git push –u origin main
```

This will output a message detailing the number of file objects that were processed and to which repository and branch.

Now we will be able to see the changes in a browser on the GitHub website *after refreshing* the page for the repository.

Adding Further Changes

We can use a few of the previous commands that we just learned to add further changes to our repository. Let's start by making a small change to the README.md file.

We should still be in the wpExampleRepository directory of the repository.

Now open the README.md file using nano, vim, or your favorite text editor. Change the README, for example by adding the following **markdown (https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax)**(MD) text to line 3.

```
## Hello
```

Make sure to save the changes that were made to the README, then use Git to add the changes to our repository.

```
git add --all
```

Like before, we need to add a message to our commit.

```
git commit -m "Added the hello heading"
```

Finally, we need to push the changes to our remote GitHub repository.

```
git push
```

It is possible to double check that the changes we made were added to our remote repository by navigating to, or refreshing the GitHub webpage for the repository.

Making Changes Online

It can be convenient to make changes to code in a repository using a browser.

1. Navigate to the GitHub webpage for your repository
2. Click the README.md file name
3. Click the pencil icon on the right side of the screen to edit the file online

Now change the file. For example, add the following text to line 4.

```
Hello!
```

Click the “Commit changes…” button, then click “Commit changes”. It is possible to add further notes to the commit message in the “Commit changes” window, which can be very helpful for future reference.

Receiving Changes

We can download and receive the changes that we made on the webpage for the repository using the git pull command in the terminal. While in a directory of the repository enter the following command.

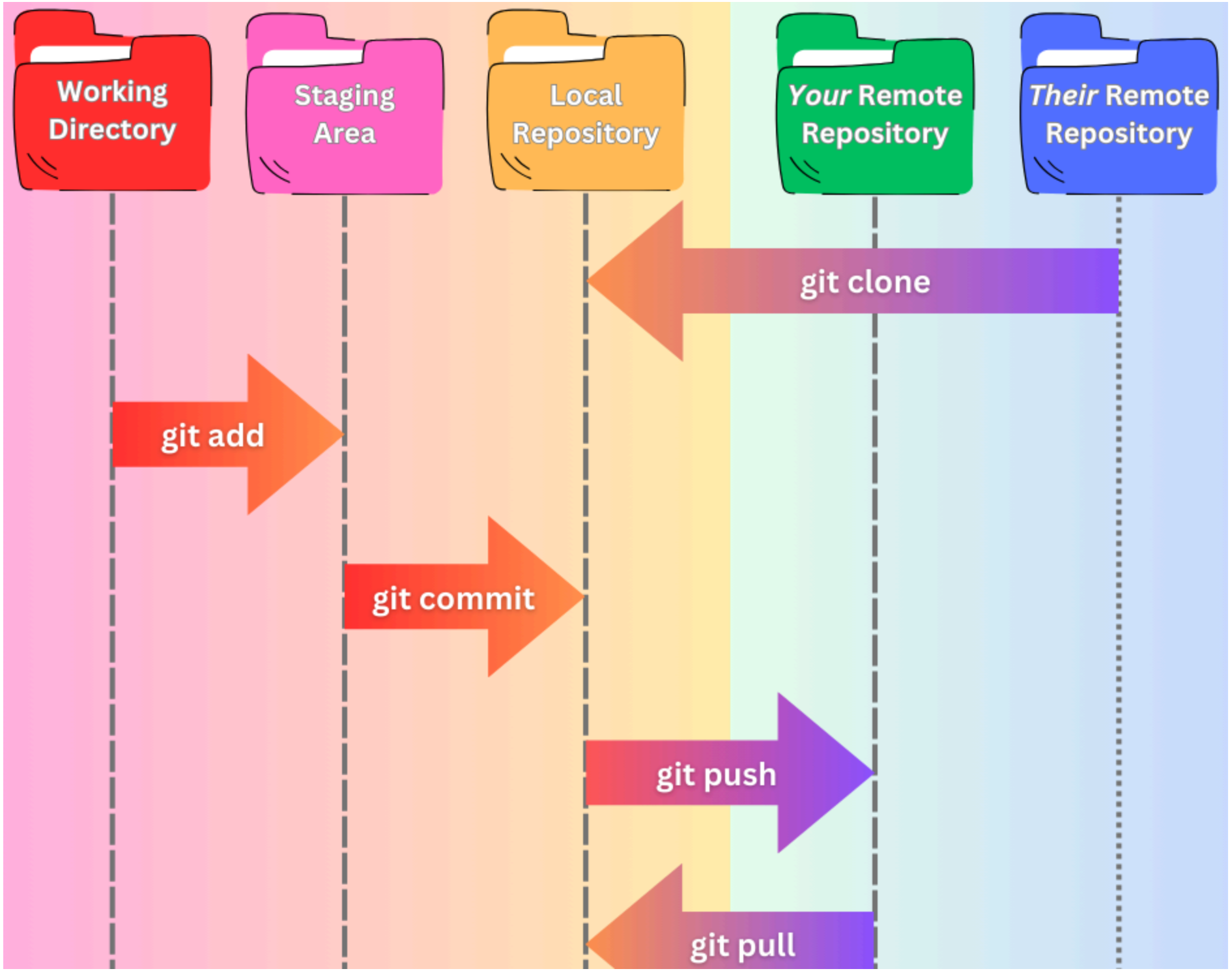
```
git pull
```

We should see a message saying which file objects were changed. In this case, README.md was changed with 1 insertion. Now if we open the README file on our computer, we will see the change we made to the file using the browser.

Key Tips

- The **git status** command can be used to check the state of the local repository. The git status command will let you know if there are changes that have not yet been committed or pushed to the remote repository.
- It is generally a good idea to perform a **git pull** command *before* making any changes and adding them using the **git push** command. This is particularly true if you make changes to your repository on multiple computers or through the repository webpage.
- Check out this resource on **GitHub for Git (https://github.com/git-tips/tips)** tips and tricks!

The following diagram depicts the general flow of commands for downloading files from a public remote repository that you don't own, in addition to adding and receiving changes to your own local or remote repository.


Working Directory
Staging Area
Local Repository
Your Remote Repository
Their Remote Repository
git clone
git add
git commit
git push
git pull

FEBRUARY 28, 2024FEBRUARY 29, 2024# BASH, # CODING, # PROGRAMMING, # TUTORIAL