title: Digging into versioning with GitHub Desktop

BACK TO MAIN PAGE

Digging into versioning with GitHub Desktop

1. Install git and GitHub Desktop software

Install git version control software

- Navigate to the official git website and follow the instructions to download and install for your operating system.
    - For Windows, use this download link and install using all of the default (recommended) options.

Create a GitHub account (if you don't have one)

- Navigate to Github and sign up for an account if you don't already have one.
- After registering, sign in to your account.
- (optional) Watch the short introductory videoWhat is GitHub?.

Install and configure GitHub Desktop

NOTE: If you are a Linux user, GitHub Desktop will not work for you. Instead, you'll need to learn how to use git and git Bash. Separate instructions are provided below.

If using a Windows or Mac, download and installGitHub Desktop - More guidance can be found in guides from GitHub and TechRepublic. - Sign into GitHub Desktop using your GitHub credentials - In the*Configure Git* page, enter the name and email address that you want associated with your changes - In File > Options > Advanced, set the Shell to Git Bash - (optional) Once configured, the main GitHub Desktop page will show any repositories that exist in your GitHub account. For those who use git already, you can add any local repositories that already exist on your machine.

2. Introduction to git and GitHub Desktop

Follow along with the slides describing git and GitHub Desktop

3A. Cloning, editing, pushing changes (For Windows and Mac users)

Clone your existing GitHub repository to your local computer

In this next step, you will 'clone' your existing GitHub repository to your local computer so that you can work on files locally. - **In GitHub Desktop**, your newly forked repository should appear in the**Your repositories** list (You may need to refresh the list for it to show). - **NOTE:** If it doesn't appear, double-check that the repository exists in your GitHub profile. If it does exist, you can copy the repository URL from GitHub and paste it into the appropriate place using the "Clone a repository from the Internet" button on GitHub Desktop. You can also use this approach to clone someone else's repository! - Highlight your repository in the list and click "Clone <*repository name*>". Select the local path where you would like to download the repository--a new folder will be created with the name of the repository.

Make some local edits | add some files!

- In GitHub Desktop, click the **Show in Explorer** button to open up your file explorer to your repository's contents.
- Open one of your Markdown (.md) files in a text editor. Make some changes and save them.
- Add a few files (images, maybe?) to the `docs` folder.

Commit new changes

- In GitHub Desktop, you should be shown the files that have been changed, and be able to view the specific changes.
- If you are comfortable with the changes, you are ready to commit them.
    - Provide a summary of the changes (or used the suggested text), and lengthier description, if desired.

- Click **Commit to master**. This commits your changes to your local repository.
- If you continue to work on your local files, you will need to again commit changes.

Push changes to GitHub (remote) repository

In this step, you'll 'push' your local changes back up to your GitHub repository, so that both are synced. - In GitHub Desktop, click the **Push origin** button to send your changes to your GitHub repository. - Verify your changes in your GitHub repository.

Make changes in the GitHub (remote) repository

- Make and commit a change to a file in the GitHub (remote) repository using the web editor.

Pull changes to the local repository

- In GitHub Desktop, click the **Fetch origin** button. This will check the GitHub repository to see if any changes have been made remotely.
- In GitHub Desktop, click the **Pull origin** button to sync remote changes to your local files.

Advanced: Fork a partner's repository | Request a pull

One of the use cases for version control is that it allows multiple individuals to work on a project at the same time. Collaborators can work on a single 'branch', or (often) they choose to 'fork' it and work on separate branches (with the idea that these branches could be 'merged' later).

In this exercise, your task is to fork a partner's repository, make some changes locally, 'push' it to your forked repository, and then make a 'pull request' to merge the changes. - Navigate to a partner's repository and click the **Fork** button. Follow the prompts to complete the task. - If successful, you now have a new repository in your GitHub account that is a 'forked' version of your partner's. - Make a (minor) edit/change to a Markdown (.md) file using the GitHub editor. Commit the changes - Click on the **Pull requests** tab for your repository. Click **New Pull Request** and then **Create pull request** - Your partner will now be notified of a pull request in their repository. They can choose to merge your change (automatically or manually), or reject it.

3B. Cloning, editing, pushing changes (For Linux users)

Configuring your git account

Open up Git Bash and navigate to the desired directory for your repository - Set your name `git config --global user.name "John Doe"` - Set your email address: `git config --global user.email johndoe@example.com` - Check your settings `git config --list` See git documentation for more information.

Clone your existing GitHub repository to your local computer

In this next step, you will 'clone' your GitHub repository to your local computer so that you can work on files locally. - In the top-level page of your GitHub repository, click on the **Clone or Download** button. Copy the URL

that is provided. - Open Git Bash in the desired directory for your repository. Enter the command: `git clone <copied url>` - Git should now download the contents of your GitHub repository to a new folder in your current directory

Make some local edits | add some files!

In your local repository folder, open one of your Markdown (.md) files in a text editor. Make some changes and save them. - Add a few files (images, maybe?) to the `docs` folder.

Add and commit new changes

- Check the status of your repository (i.e. what's been modified): `git status`
    - This will provide a list of items that are not yet being tracked (i.e. have not been added to the index), and those that are being tracked and have been modified.
- Add new items to the list of tracked files (individually): `git add <filename>`
    - **OR** Add all items to this list of tracked files: `git add --all`
- Commit changes to git (i.e. record changes): `git commit -m '<enter a note on what has changed>'`
    - **OR** add and commit all at once: `git commit -a -m '<enter a note on what has changed>'`

Push changes to GitHub (remote) repository

- To check if there are connected remote repositories use the command `git remote -v`
- Push changes to the target Github repository using the command: `git push origin master`
    - In this example -- which is the default case --**origin** specifies the remote (i.e. Github) repository that is the target of your 'push'. **master** specifies the branch of the git repository that you're working on as the source data.
- Verify your changes in your GitHub repository.

Make changes in the GitHub (remote) repository

- Make and commit a change to a file in the GitHub (remote) repository using the web editor.

Pull changes to the local repository

- You can check changes (before merging them) with: `git fetch` `git diff origin master`
- Pull (fetch and merge) changes: `git pull origin master`
    - **Note:** `git pull` actually runs two processes: `fetch` (get changes) and `merge` (place in your directory)
- Inspect the changes to your local repository.

Advanced: Fork a partner's repository | Request a pull

One of the use cases for version control is that it allows multiple individuals to work on a project at the same time. Collaborators can work on a single 'branch', or (often) they choose to 'fork' it and work on separate

branches (with the idea that these branches could be 'merged' later).

In this exercise, your task is to fork a partner's repository, make some changes locally, 'push' it to your forked repository, and then make a 'pull request' to merge the changes. - Navigate to a partner's repository and click the **Fork** button. Follow the prompts to complete the task. - If successful, you now have a new repository in your GitHub account that is a 'forked' version of your partner's. - Make a (minor) edit/change to a Markdown (.md) file using the GitHub editor. Commit the changes - Click on the **Pull requests** tab for your repository. Click **New Pull Request** and then **Create pull request** - Your partner will now be notified of a pull request in their repository. They can choose to merge your change (automatically or manually), or reject it.

4. More information

git

- Official git documentation page
- The Smart Ways to Correct Mistakes in Git