Basic Interaction with GitHub Cheat-Sheet

There are various remote repository hosting sites:

- GitHub
- BitBucket
- Gitlab.

Follow the workflow at https://github.com/join to set up a free account, username, and password. After that, these steps will help you create a brand new repository on GitHub.

Some useful commands for getting started:

Command	Explanation & Link
git clone URL	Git clone is used to clone a remote repository into a local workspace
git push	Git push is used to push commits from your local repo to a remote repo
git pull	Git pull is used to fetch the newest updates from a remote repository

This can be useful for keeping your local workspace up to date.

- https://help.github.com/en/articles/caching-your-github-password-in-git
- https://help.github.com/en/articles/generating-an-ssh-key

Git Remotes Cheat-Sheet

Command	Explanation & Links
git remote	Lists remote repos
git remote -v	List remote repos verbosely
git remote show <name>	Describes a single remote repo
git remote update	Fetches the most up-to-date objects
git fetch	Downloads specific objects
git branch -r	Lists remote branches; can be combined with other branch arguments to manage remote branches

You can also see more in the video Cryptography in Action from the course IT Security: Defense against the digital dark arts.

Difference between git fetch and git pull

Fetch

```
$ git fetch origin
```

git fetch really only downloads new data from a remote repository - but it doesn't integrate any of this new data into your working files. Fetch is great for getting a fresh view on all the things that happened in a remote repository.
Due to it's "harmless" nature, you can rest assured: fetch will never manipulate, destroy, or screw up anything. This means you can never fetch often enough.

Pull

```
$ git pull origin master
```

git pull, in contrast, is used with a different goal in mind: to update your current HEAD branch with the latest changes from the remote server. This means that pull not only downloads new data; it also directly **integrates** it into your current working copy files. This has a couple of consequences:

- Since "git pull" tries to merge remote changes with your local ones, a so-called "merge conflict" can occur. Check out our in-depth tutorial on How to deal with merge conflicts for more information.
- Like for many other actions, it's highly recommended to start a "git pull" only with a clean working copy. This means that you should *not* have any uncommitted local changes before you pull. Use Git's Stash feature to save your local changes temporarily.

Open this link to get more: https://www.git-tower.com/learn/git/faq/difference-between-git-fetch-git-pull/

Conflict Resolution Cheat Sheet

Merge conflicts are not uncommon when working in a team of developers, or on Open Source Software. Fortunately, GitHub has some good documentation on how to handle them when they happen:

- https://help.github.com/en/github/collaborating-with-issues-and-pull-requests/about-merge-conflicts
- https://help.github.com/en/github/collaborating-with-issues-and-pull-requests/resolving-a-merge-conflict-using-the-command-line

You can also use git rebase branchname to change the base of the current branch to be branchname

The git rebase command is a lot more powerful. Check out this link for more information.