

## Git & GitHub Commands

<https://usm.maine.edu/departement-computer-science/>

## Department of Computer Science Programs

Bachelor of Science

Master of Science

### Setup

```
git config --global user.name "[username]"  
Set the Username Git will use to authenticate
```

```
git config --global user.email "[email]"  
Set the Email Git will use to authenticate
```

```
git config --global color.ui auto  
Set automatic command line coloring for Git
```

```
git init  
Initialize the current directory as a git repository
```

```
git clone [url]  
Download a local copy of a repository to the machine
```

### Staging and Snapshots

```
git status  
Show modified files that are staged for commit
```

```
git add [file/directory]  
Stage a file for the next commit
```

```
git reset [file]  
unstage a file for commit while keeping the changes
```

```
git diff  
Changes in files that are not staged for commit
```

```
git diff --staged  
Changes in files that are staged for commit but not yet committed
```

```
git commit -m "[commit message]"  
Commit your staged content as a new commit snapshot
```

```
git commit --amend  
Add staged content to a commit snapshot that already exists
```

### Initializing a Repository

```
git init  
Initialize the current directory as a git repository
```

```
git clone [url]  
Download a local copy of a repository to the machine
```

### File Path Changes

```
git rm [file]  
Delete the file from the project and stage the removal for commit
```

```
git mv [existing-path] [new-path]  
Change an existing file path and stage the move
```

```
git log -stat -M  
Show all commit logs with an indication of any paths that changed
```

### Branches and Merging

```
git branch  
List all branches. The active branch is marked with a '*'
```

```
git branch [branch-name]  
Create a new branch at the current commit
```

```
git branch -d [branch-name]  
Delete the specified branch at the current commit
```

```
git remote add upstream URL.git  
Add 'upstream' repository to sync changes made in a forked repo
```

```
git checkout [branch-name]  
Switch branches and check it out into the working directory
```

```
git merge [branch]  
Merge the selected branch's history into the current branch
```

### Inspect and Compare

```
git log  
Show all commits in the current branch's history
```

```
git log [branchB]..[branchA]  
Show the commits on branchA that are not on branchB
```

```
git log --follow [file]  
Show the commits that changed 'file', even across renames
```

```
git diff [branchB]..[branchA]  
Show the difference of what is in branchA but not in branchB
```

```
git show [SHA]  
Show any object in Git in human-readable format
```

### Share and Update

```
git remote add [alias] [URL]  
Add a Git URL as an alias
```

```
git fetch [alias]  
Fetch down all the branches from that Git remote
```

```
git merge [alias]/[branch]  
Merge remote branch into your current branch
```

```
git push [alias] [branch]  
Transmit local branch commits to the remote repository branch
```

```
git pull  
Fetch and merge any commits from the tracking remote branch
```

### Version Control

```
git rebase [branch]  
Apply any commits of current branch ahead of the specified one
```

```
git reset --hard [commit]  
Clear staging area, rewrite working tree from the specified commit
```

```
git checkout -- [file]  
Discard the changes to 'file' in the working directory
```

```
git checkout [commit id]  
Revert to a previous commit
```

```
git revert [commit id]  
Undo the specified commit
```

### Tokens

```
git remote set-url origin  
https://<NEW_TOKEN>@github.com/username/repo.git  
Set the remote repository using a Personal Access Token
```

Tokens can also be used to authenticate in the command line, for example:

```
$ git clone https://github.com/USERNAME/REPO.git
```

```
Username: YOUR-USERNAME
```

```
Password: YOUR-PERSONAL-ACCESS-TOKEN
```