Getting Started with Git and GitHub
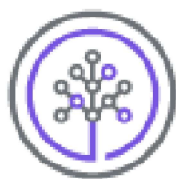
Module 2 Cheat Sheet: Git Commands and Managing GitHub Projects

Package/Method	Description	Code Example
git add	Used to move changes from the working directory to the staging area	`git add sample.md`
git add .	Allows to move the changed files into the staging area on GitHub repositories	`git add .`
git am	Used to apply patches emailed to the repository	`git am < patchfile.patch`
git branch	Allows to create an isolated environment within the repository to make changes	`git branch <new-branch>`
git checkout	Allows to see and change existing branches	`git checkout <existing-branch>`
git checkout main	Allows to switch to the main branch	`git checkout main`
git clone	Allows to create a copy of the remote repository	`git clone <repository-url>`
git commit	Allows you to take staged snapshots if changes and commit them to the project	`git commit -m "Your commit message here"`

git config --global user.email	Example 1: Sets a global email configuration for Git
	
	Example 2: Sets a global username configuration for Git
		Example 1:
		`git config --global user.email "your.email@example.com"`
		
		Example 2:
		`git config --global user.name "Your Name"`

git daemon	Used to allow anonymous download from the repository	`git daemon --reuseaddr --verbose`
git diff	Helps others to review your code to identify and compare the changes	`git diff example.txt`
git fetch	Used to transfer the changes from the remote repo to your local repo	`git fetch <options> <remote name> <branch name>`
git fetch upstream/master	Used to grab upstream branches	`git fetch upstream master:upstream-master`
git format-patch	Generates or prepares e-mail submission if you adopt Linux kernel-style public forum workflow	`git format-patch -n <number_of_commits>`

git http-backend	Provides a server-side implementation of Git-over-HTTP, allowing both fetch and push services	`git clone --bare /path/to/repos/myrepo.git`
		`cd myrepo.git`
		`git update-server-info`

git init	Used to clone an existing repository	`git init <directory>`
git instaweb	Allows to set up web front-end to Git repositories	`git instaweb -p 8080`
git log	Enables to browse previous changes to a project	`git log -p filename`
git merge	Used to merge changes in the active branch into another branch	`git merge feature_branch`
git merge upstream/master	Merges changes from the 'upstream/master' branch to the current branch	`git merge upstream/master`
git pull	Used to transfer the changes from the remote repo to your local repo, and merge them to a branch	`git pull origin main`
git pull downstream	Pulls changes from a downstream repository, specifically from the master branch of that repository	`git pull downstream main`
git pull upstream	Pulls changes from the "upstream" repository into the current branch	`git pull upstream main`

Package/Method	Description	Code Example
git push	Used to push all the committed changes into the repository	`git push origin your_branch_name`
git remote	A command to manage a set of tracked repositories	`git remote add upstream https://github.com/original/repo.git`
git remote add origin <URL>	Adds a remote repository named "origin" with the specified URL	`git remote add origin https://github.com/yourusername/your-repo.git`
git remote add upstream	Adds the original repository as a new remote repository labeled upstream	`git remote add upstream https://github.com/original/repo.git`
git remote rename	The git remote rename command is followed by the name of the remote repository(origin) you want to rename and the new name(upstream) you want to give it	`git remote rename origin new-origin`
git remote -v	Allows to view the remotes associated with the local repository	`git remote -v`

git request-pull	Example 1: Creates a summary of changes for your upstream to pull
	
	Example 2: Generates a summary of pending changes for an email request
		Example 1:
		
		`git request-pull origin/main your-branch`
		
		Example 2:
		
		`git request-pull <base> <head> <repository>`

git rerere	Reuses recorded resolution of previously resolved merge conflicts	`git rerere` `git rerere diff`
git reset	Undoes changes that were made to the files in your working directory	`git reset HEAD~1`
git revert	Used to undo botched commits	`git revert HEAD`

git send-email	Example 1: Sends your email submission without corruption by your MUA
	
	Example 2: Sends a collection of patches as emails
		Example 1:
		
		`git send-email --to=recipient@example.com path/to/patchfile.patch`
		
		Example 2:
		
		`git send-email --to recipient@example.com patches/*.patch`

git-shell	Used as a restricted login shell for shared central repository users	`sudo usermod -s /usr/bin/git-shell gituser`
git status	Allows to see the state of your working directory and the staged snapshot of the changes	`git status`
git version	Displays the current Git version installed on your system	`git --version`
git web	Provides a web front-end to Git repositories	`git instaweb --port=8080`