2021-2022

ACM STUDENT CHAPTER - HITK

GIT BASICS

ADVANCING COMPUTING AS A SCIENCE & PROFESSION

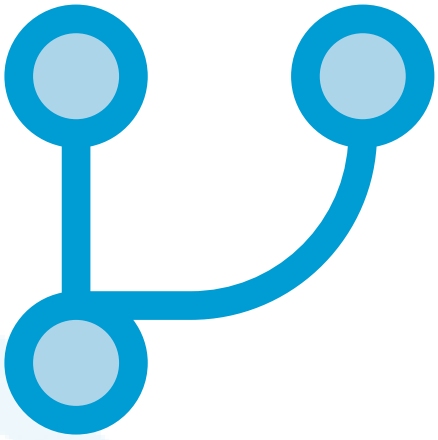
GIT BASICS & GIT CHEAT SHEET

GIT

Git is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows.

ACM Student Chapter at HITK

GIT BASICS



Contents

1. Installation
2. Basic Shell Commands
3. Basic Git Commands
4. Git Cheat Sheet
5. Explore More

Installation

For you to be able to use Git on your local machine, you would need to install it. Depending on the operating system that you are using, you can follow the steps here.

Install Git on **Linux**

Note: With most Linux distributions, the Git command-line tool comes installed out of the box. If this is not the case for you, you can install Git with the following command:

a. On Debian-based distributions including Ubuntu:

```
sudo apt install git-all
```

b. On CentOS/ Fedora distributions:

```
sudo dnf install git-all
```

Install Git on **Mac**

If you are using Mac, Git should be available out of the box as well. However, if this is not the case

```
brew install git
```

Install Git on **Windows**

Link:  Git

During the installation, make sure to choose the Git Bash option, as this would provide you with a Git Bash terminal that you will use while following along.

Basic Shell Commands

We will be using mainly Git via the command-line interface (CLI). It is important to know basic shell commands so that you could find your way around the terminal.

The **ls** command

It allows you to list the contents of a folder/directory.

```
ls
```

The **cd** command

It stands for Change Directory and allows you to navigate through the filesystem of your computer or server

```
cd <directory>
```

The **pwd** command

The pwd command stands for Print Working Directory which essentially means that when you run the command, it will show you the current directory that you are in.

```
pwd
```

Output: It will vary according to your system

```
/home/user/introduction-to-git
```

The **rm** command

It stands for remove and allows you to delete files

```
rm <filename>
```

To delete a directory/folder, use the -r flag

```
rm -r <directory>
```

It is recommended to stay away from **rm** command.
Read more here

Basic Git Commands

git config

Configuring user information used across all local repositories

```
git config --global user.name "[firstname lastname]"
```

set a name that is identifiable for credit when reviewing version history

```
git config --global user.email "[valid-email]"
```

set an email address that will be associated with each history marker

git init

Make sure you make a new directory/folder where you want to initialize git. If you are starting a new project or if you have an existing project which you would like to add to Git and then push to GitHub, you need to initialize a new Git project with the git init command.

```
git init
```

git status

Whenever you make changes to your Git project, you would want to verify what has changed before committing or before pushing your changes to GitHub.

```
git status
```

git add <directory/file>

Stage all changes in <directory> for the next commit. Replace <directory> with a <file> to change a specific file.

```
git add <directory/file>
```

git commit -m “Your msg”

Commit the staged snapshot.

```
git commit -m “My commit msg”
```

git push <remote> <branch>

Push the branch to <remote>, along with necessary commits and objects. Creates named branch in the remote repo if it doesn’t exist.

```
git push <remote> <branch>
```

git pull <remote>

Fetch the specified remote’s copy of current branch and immediately merge it into the local copy.

```
git pull <remote>
```

Git Cheat Sheet

SETUP

Configuring user information used across all local repositories

```
git config --global user.name "[firstname lastname]"
```

```
git config --global user.email "[valid-email]"
```

SETUP & INIT

initialize an existing directory as a Git repository

```
git init
```

retrieve an entire repository from a hosted location via URL

```
git clone [url]
```

STAGE & SNAPSHOT

Show status of files in working directory

```
git status
```

Add a file as it looks now to your next commit (stage)

```
git add [file]
```

Unstage a file in working directory

```
git reset [file]
```

Diff of what is changed but not staged

```
git diff
```

Commit your staged content as a new commit snapshot

```
git commit -m “[descriptive message]”
```

SHARE & UPDATE

Fetch and merge any commits from the tracking remote branch

```
git pull
```

Push local branch commits to the remote repository branch

```
git push [alias] [branch]
```

BRANCH & MERGE

List your branches.*will appear next to the currently active branch

```
git branch
```

Create a new branch at the current commit

```
git branch [branch-name]
```

Switch to another branch and check it out into your working directory

```
git checkout
```

Merge the specified branch’s history into the current one

```
git merge [branch]
```

INSPECT & COMPARE

Show the commit history for the currently active branch

```
git log
```

show the commits on branchA that are not on branchB

```
git log branchB..branchA
```

show the diff of what is in branchA that is not in branchB

```
git diff branchB...branchA
```

Explore More

Check these awesome articles:

How to Contribute to Open Source

What open source project should I contribute to?

Hacktoberfest Guide

Open Source contributions:

GSOC

Hacktoberfest Contribution Repository List

Awesome for Beginners

Free Git & GitHub Course:

Coding train GitHub tute

. . .

> WE HOPE TO DEVELOP A BETTER COMMUNITY RELATIONSHIP WITH OUR FELLOW ACM MEMBERS AT HITK