Git: The Basics

In-House Training Workshop

National Environmental Standards and Regulations Enforcement Agency (NESREA)

25 April 2017

What is git?

- Git is all about:
  - Version control
  - Collaboration
  - Storage
- It is an extremely useful tool being used by professionals who generate files on computers, particularly those that involve some form of coding.

Getting started

- First we have to install Git to our computers from this page.
- Though it's relatively simple, one may encounter some minor hiccups. Ask for help!

The 2 Key Aspects of Git

- Version Control
- Collaboration

How to use Git

- Command Line (recommended!!!)
- Integrative (with major applications like IDEs)
- GUI

Command Line Basic Basics

- Depending on your OS
  - Windows: Command prompt, Powershell
  - macOS: Terminal (bash)
  - Linux…
- To use git you'll like need to know how to
  - Change directory (cd)
  - Check directory contents (dir, ls)
  - Delete a file (del, rm)
  - Copy/move a file (cp)
  - Write to file (echo, cat)
  - Read a file (various commands)
- You can still use GUI-based folder management (at the beginning) but this is really cumbersome and time-consuming.

Git in action

- Open your command line interface of choice
- Run git.exe

- Type and run git
- Now try git help
- Note that EVERY command run in the utility is preceded by the keyword “git”.

Create a Git repository

- A git repository ("repo") is a folder that has been initialized to work with the utility.
- Let try something out:
  - Create and open a blank folder called GitLesson in your favourite GUI e.g. Windows Explorer. (Make sure it is configured to show hidden files/folders.)
  - Open the CLI and navigate to that folder/directory
  - Start Git and make sure its running
  - Run this command: git init
  - Look at your GUI. What did you notice?

A brief explanation…

- Git essentially takes snapshots of files in the repo.
- 2 steps to snapshotting:
  - Staging
  - Committing

- ***git add:*** Staging your changes
- ***git commit:*** Taking that snapshot
- ***git push:*** Sending your files' new state to remote
- ***git pull***: Collect your own copy of remotely stored changes

git add

- The command used for staging the changes made in the file for commit
- Prepares the stage for the commit
- Allows you to review which files to include

git commit

- With this command, a snapshot of the file in its current state is made
- Has a unique ID – the SHA key
- Must be accompanied with a message to describe your changes to help
  - Other people
  - Your future self

git push

- The committed changes are sent to a remote server where the repository resides
  - This is usually at GitHub, but now exclusively so.
- You will need to use password authentication to push the changes.
- This step is purely optional.

git pull

- Involves 2 operations:
  - Fetching
  - Merging
- git fetch enables you to ‘collect’ changes found in the remote repo
- git merge incorporates the changes into your repo (branch)

A note about .gitignore

- .gitignore is a text file that carries the list of stuff you do not want be committed.
- Important for utility files that change frequently & whose details add no real value
- Also files related to privacy or security

Live coding

(`Shell commands are in white`)

1. Open Git bash
2. Open a new folder and call it ***GitPractice***.
3. Initialise Git (`git init`) inside this folder.
4. Download this file and save in the folder as ***lorem-ipsum.txt*** (in some browsers you will have to copy the text and copy into notepad
5. Check the file contents from Bash with `notepad`
6. Give it a heading in English; save and close file
7. Check the `status` of Git, then stage `(add)` and `commit` the file.

Assignment

- Go to NESREA GitHub profile and fork the repository of the same name to your own account
- Push your *local* repository ***GitPractice*** to your own *remote* repository
- Create a pull request for NESREA Admin to incorporate your changes.
- If you're lost, click on '*Issues*' and register your complaint, question or suggestion there.

Conclusion

- There's actually a lot more to Git that is better to learn on the fly
  - branch, reset, rebase, diff, mv, log, remote, …
- But the ones discussed here are all you need to get started.

Resources

- git help
- http://git-scm.com/documentation
- http://gitref.org/basic/ - very easy to use site
- https://www.atlassian.com/git/ - powerful tutorials
- Of course, GitHub's help pages

- Bonus: Git Cheatsheet

GitHub
GIT CHEAT SHEET

V 1.1.1

Git is the open source distributed version control system that facilitates GitHub activities on your laptop or desktop. This cheat sheet summarizes commonly used Git command line instructions for quick reference.

INSTALL GIT

GitHub provides desktop clients that include a graphical user interface for the most common repository actions and an automatically updating command line edition of Git for advanced scenarios.

GitHub for Windows
https://windows.github.com

GitHub for Mac
https://mac.github.com

Git distributions for Linux and POSIX systems are available on the official Git SCM web site.

Git for All Platforms
http://git-scm.com

CONFIGURE TOOLING

Configure user information for all local repositories

MAKE CHANGES

Review edits and craft a commit transaction

```
$ git status
```

Lists all new or modified files to be committed

```
$ git diff
```

Shows file differences not yet staged

```
$ git add [file]
```

Snapshots the file in preparation for versioning

```
$ git diff --staged
```

Shows file differences between staging and the last file version

```
$ git reset [file]
```

Unstages the file, but preserve its contents

```
$ git commit -m "[descriptive message]"
```

Records file snapshots permanently in version history

Thanks again!