UCF Physics PHZ 3150: Introduction to Numerical Computing

Learning Git

Running Git Commands

In a shell, just type Git commands, like the ones below. On Windows outside of the BASH on Ubuntu on Windows environment, you can launch either Git BASH or the Git Unix-like Windows Command Prompt. In particular:

```
cd <directory>
```
As in the shell, change your working directory (your location) to the specified directory. The default is the home directory.

```
ls <directory>
```
As in the shell, list the contents of the specified directory. The default is the current directory.

Learning Git

The basic flow of a project is: making the original repository ("repo") with `git init` in an empty directory (or cloning one from somewhere else), creating/editing files there with an editor like Emacs, adding them to the staging list for committing to the repo, committing them, making more changes, adding the files to the staging list, committing them, change, add, commit, change, add, commit, etc. To get fine granularity in your commit log, **"Commit early, commit often!"**

A more sophisticated and faster use would be cycles of: change, test, add, commit. The tests swat bugs early. You can copy a repo (called "branching"), work in it to test something out, and merge the changes into the master branch when you're happy with them (or delete the branch if it's fruitless).

There are excellent online Git learning materials in a variety of formats. For this class, we will mainly use these commands: `init, status, log, add, rm, mv, diff, commit, branch,`
`checkout, merge, clone, push,` with less emphasis on the last two or five.
Try these short tutorials:
https://try.github.io/ - the Learn Git Branching exercise (hands-on and fun!)
https://git-scm.com/doc - the About section and the videos in the Documentation section

Some Useful Git Commands

```
git init
```
Transform the current directory into a Git repository.

```
git config --global user.name <name>
```
Set the user's name for all future log entries from this account. Used once ever per account.

```
git config --global user.email <email address?
```
Set the user's email address for all future log entries from this account. Used once ever per account.

```
git status
```
List any changes to your Git repo that have not been committed.

`git log [<filename>]`
Print the committer, timestamp, and message of the repo's `commits`.

`git add <file(s)>`
Stage files for the next `commit`.

`git rm <file(s)>`
Removes files from the staging list.

`git mv <file1> <file2>`
Rename a file.

`git diff <file1> <commit>`
Compares a file to the same file in a different commit, default the last one..

`git commit -m <message>`
Commit any staged changes (added files, removed files, modified files that you have added, etc.) to the repo.

`git branch <branch name>`
Create a new branch with the specified name. Branches allow you to make changes safely and test them, while still having access to the original branch. It is good practice to keep a working "master" branch and a separate development branch where you are free to break things.

`git checkout [-b] <branch name>`
Change to the named branch. If `-b` appears, create it first.

`git merge <branch name>`
Merge the named branch into the current branch. It refuses if changes were committed to the named branch after we last checked it out or pulled changes from it.

`git clone <git repository> <destination>`
Copy a (possibly remote) Git repository to the destination directory.

`git push <location> <branch>`
Send all changes that have been committed to the repo (which is usually a clone) to the specified location and branch. For example, `git push origin master` will push all changes to the master branch at the location from which the repo was cloned (the origin).

`git pull <repositiory>`
Incorporate differences between a repository and your current branch. This is useful if multiple people are working on the same project and you want to get their changes.

These are only the simplest commands in the Git suite. Look at the man pages, but don't be overwhelmed! Most of the Git commands issued in practice are the simple ones here.