

Introduction to Git

PHYS 512

What is git/Github?

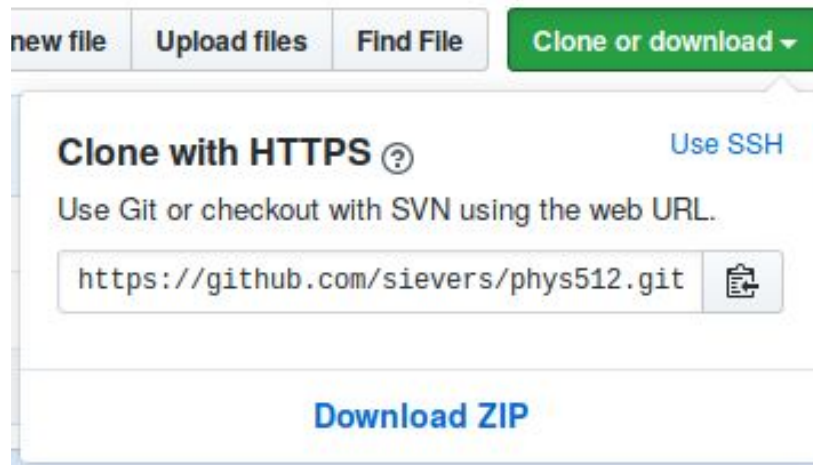
- Git is a Version Control System (VCS) used to keep a running history of code across potentially numerous branches. It is prolific in many areas of science, and indispensable for software engineers and computer/data scientists.
- See [here](#) for a more comprehensive introduction
- Installing git is simple, and it may already be on your computer. Try running **git --version** to see if your computer already has git installed
 - If not, follow the relevant instructions [here](#)

A screenshot of a terminal window with a dark background and light green text. The terminal shows the command 'git --version' being executed, resulting in the output 'git version 2.17.1'. The prompt 'marcus@latitude:~\$' is visible. The terminal window has a menu bar at the top with options: File, Edit, View, Search, Terminal, and Help. The name 'marcus' is visible in the top right corner of the terminal window.

```
marcus@latitude:~$ git --version
git version 2.17.1
marcus@latitude:~$
```

Getting Started

- The first things you'll want to do with git for this class are create a local directory that is linked to Sievers' [course repository](#). This is where the lecture/tutorial resources will go, and it will be useful to have a local copy.
- To make a local copy of the repository, you're going to want to **git clone** `<url>` the relevant repository. This will create a directory with the same name as the repository on your local machine in the directory from which you ran the command in on your terminal

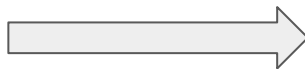


Git Pull

- Once you have a local copy of the repository, you can **git pull** whenever the repository is updated with new content to make the changes on your machine!
- Note: you won't be making changes to this repo -- you won't have permissions anyways -- so try not to make/edit files in this directory!

```
marcus@latitude:~/McGill/PHYSS12/phys512$ ls
lecture_1  tutorial_1
marcus@latitude:~/McGill/PHYSS12/phys512$
```

git pull




```
marcus@latitude:~/McGill/PHYSS12/phys512$ git pull
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 14 (delta 1), reused 14 (delta 1), pack-reused 0
Unpacking objects: 100% (14/14), done.
From https://github.com/sievers/phys512
  4c0094c..538acc8  master    -> origin/master
Updating 4c0094c..538acc8
Fast-forward
 lecture_2/bad_parabolas.py | 17 ++++++++
 lecture_2/cubic_interpolation.py | 25 ++++++++
 lecture_2/delta_polys.py | 24 ++++++++
 lecture_2/integrate_linear.py | 23 ++++++++
 lecture_2/integrate_things.py | 53 ++++++++
 lecture_2/legendre.py | 25 ++++++++
 lecture_2/phys512_2.pdf | Bin 0 -> 1923444 bytes
 lecture_2/simpsons.py | 25 ++++++++
 lecture_2/spline_example.py | 18 ++++++++
 9 files changed, 210 insertions(+)
 create mode 100644 lecture_2/bad_parabolas.py
 create mode 100644 lecture_2/cubic_interpolation.py
 create mode 100644 lecture_2/delta_polys.py
 create mode 100644 lecture_2/integrate_linear.py
```

Making Your Own Course Repository

- On Github (make an account with your **@mail.mcgill.ca** email if you haven't yet!), navigate to your repositories and select `new`

Repositories **3** Projects **0** Stars **8** Followers **0** Following **0**

itory... Type: **All** Language: **All** 

- Then just follow the instructions to make a public repository!

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner

Repository name *

 **marcusmerryfield** /

Great repository names are short and memorable. Need inspiration? How about **cuddly-octo-enigma**?

- Now you can **git clone** your personal PHYS512 repo (just like we did with Sievers' repo before!) wherever you'd like on your machine!

The screenshot shows the GitHub interface for the repository 'marcusmerryfield / phys512_mm'. At the top, there are buttons for 'Unwatch' (1), 'Star' (0), and 'Fork' (0). Below this is a navigation bar with links for 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Security', 'Insights', and 'Settings'. The main content area shows 'No description, website, or topics provided.' with an 'Edit' button. Below this is a section with statistics: '1 commit', '1 branch', '0 packages', '0 releases', and '1 contributor'. There are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find File', and a green 'Clone or download' button. A dropdown menu is open from the 'Clone or download' button, showing 'Clone with HTTPS' (with a help icon), 'Use SSH', and 'Download ZIP'. The 'Clone with HTTPS' option is selected, showing the URL 'https://github.com/marcusmerryfield/ph' and a copy icon. The repository content shows an 'Initial commit' by 'marcusmerryfield' with a file named 'README.md'.

marcusmerryfield / **phys512_mm**

Unwatch 1 Star 0 Fork 0

<> Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. Edit

Manage topics

1 commit 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find File Clone or download

marcusmerryfield Initial commit

README.md Initial commit

README.md

Clone with HTTPS ? Use SSH

Use Git or checkout with SVN using the web URL.

https://github.com/marcusmerryfield/ph

Download ZIP

git status/add/commit/push

- Once you've got a local version of your repository, you can add contents to it. For example, I can make a file called `test.py` which contains some code I want to put on Github. Then, if I **git status** in the directory:

```
marcus@latitude:~/McGill/PHY5512/phys512_mm$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    test.py

nothing added to commit but untracked files present (use "git add" to track)
marcus@latitude:~/McGill/PHY5512/phys512_mm$
```

- My changes not yet online show up! Let's get them online.

- The steps to getting our changes online are adding the new/edited files (**git add <filename>**), committing them to a local commit with a message of what you changed (-m), which is basically just a version stamp of the code at that moment (**git commit -m “test commit”**), and then pushing those commits to the online (‘remote’) repository (**git push**)

```
marcus@latitude:~/McGill/PHYSS512/phys512_mm$ git add test.py
marcus@latitude:~/McGill/PHYSS512/phys512_mm$ git commit -m "test commit"
[master bcf7591] test commit
 1 file changed, 1 insertion(+)
 create mode 100644 test.py
marcus@latitude:~/McGill/PHYSS512/phys512_mm$ git push
Username for 'https://github.com': marcusmerryfield
Password for 'https://marcusmerryfield@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 293 bytes | 293.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/marcusmerryfield/phys512_mm.git
   f74b4e7..bcf7591  master -> master
marcus@latitude:~/McGill/PHYSS512/phys512_mm$
```


Now you know the basics of git!

- These are just the barebones basics, but should be able to carry you through most of what you need in the course. For a more comprehensive list of commands, try **git --help** which will show many of the more advanced options.
- I highly recommend playing around and getting used to git and Github. If you have any interest in working for a developer in the future, or even in a scientific collaboration, git is an extremely important skill, and good knowledge of it will make your life much easier. :)
- If you have any more questions, just send us an email!