Introduction to git and GitHub

Daniel Thaagaard Andreasen
daniel.andreasen@astro.up.pt

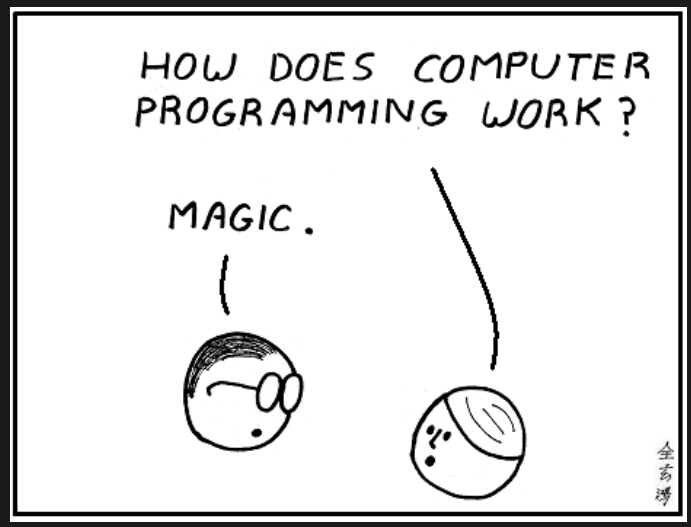
CAUP

29th of January, 2015

Table of contents

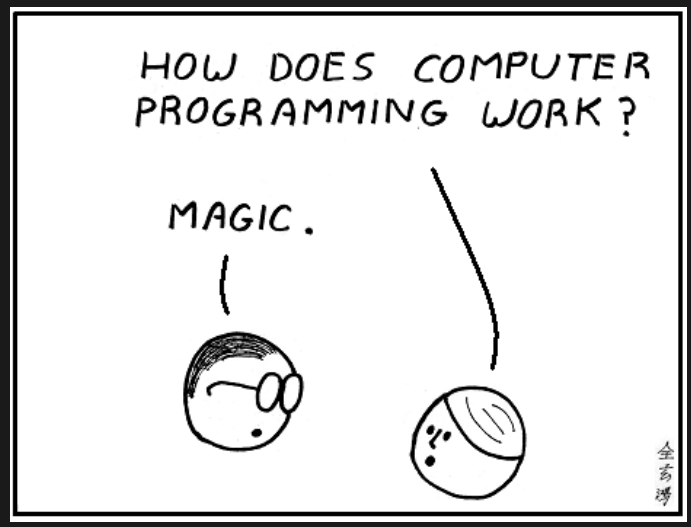
Your job

- **Create** things



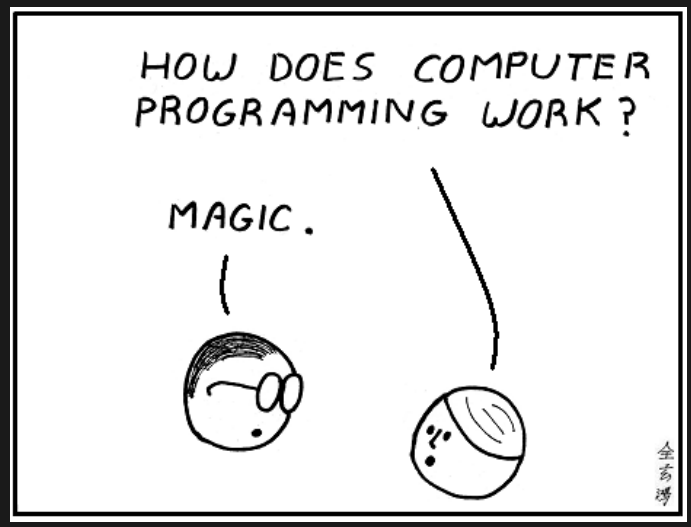
Your job

- **Create** things
- **Save** things



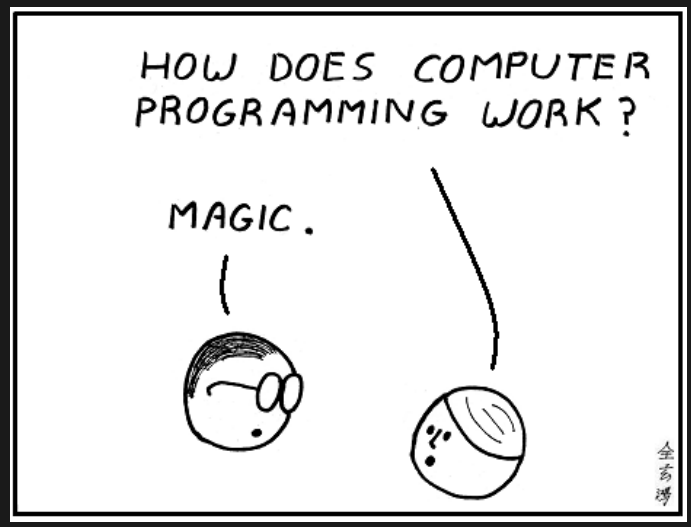
Your job

- **Create** things
- **Save** things
- **Edit** things



Your job

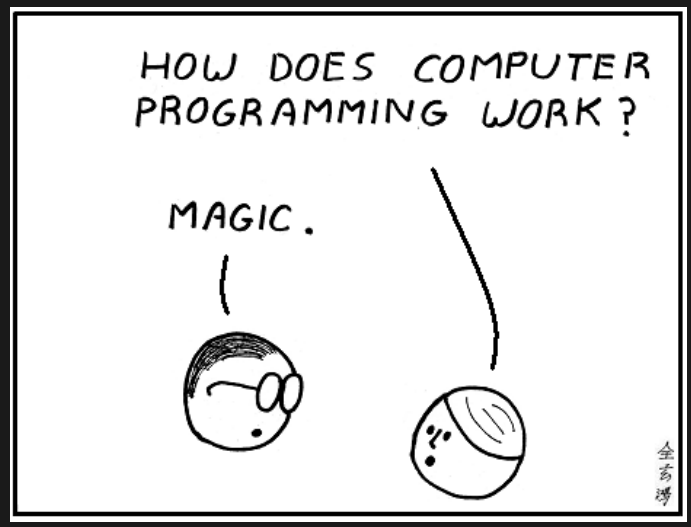
- **Create** things
- **Save** things
- **Edit** things
- **Save** things again (overwrite)



Your job

- **Create** things
- **Save** things
- **Edit** things
- **Save** things again (overwrite)

- The last point here is crucial!



Your job, improved

- Keep track of a file
  every time it is saved.

Your job, improved

- Keep track of a file every time it is saved.
- This is the step further for backups which just overwrite previous versions of a file.

Your job, improved

- Keep track of a file every time it is saved.
- This is the step further for backups which just overwrite previous versions of a file.
- Be sure to write helpful messages!

What is git?

- Git is a modern and fast version control (invented by Linus Torvalds).

What is git?

- Git is a modern and fast version control (invented by Linus Torvalds).
- A tool to keep track of changes in files with a time stamp and a user-written message ("What did I do and why?").

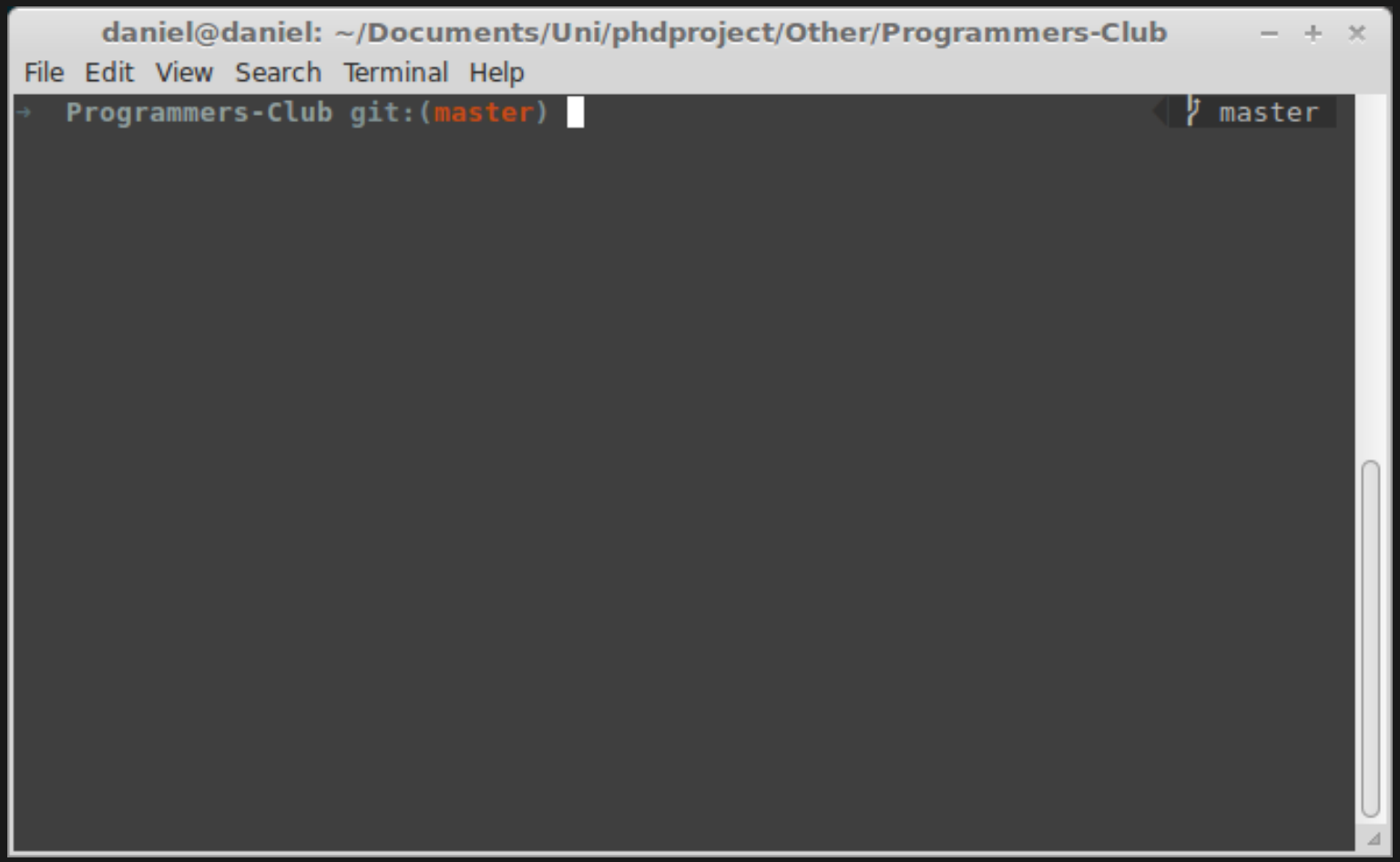
What is git?

- Git is a modern and fast version control (invented by Linus Torvalds).
- A tool to keep track of changes in files with a time stamp and a user-written message ("What did I do and why?").
- Allow the user to "push" files to a server: supernova, GitHub, etc.

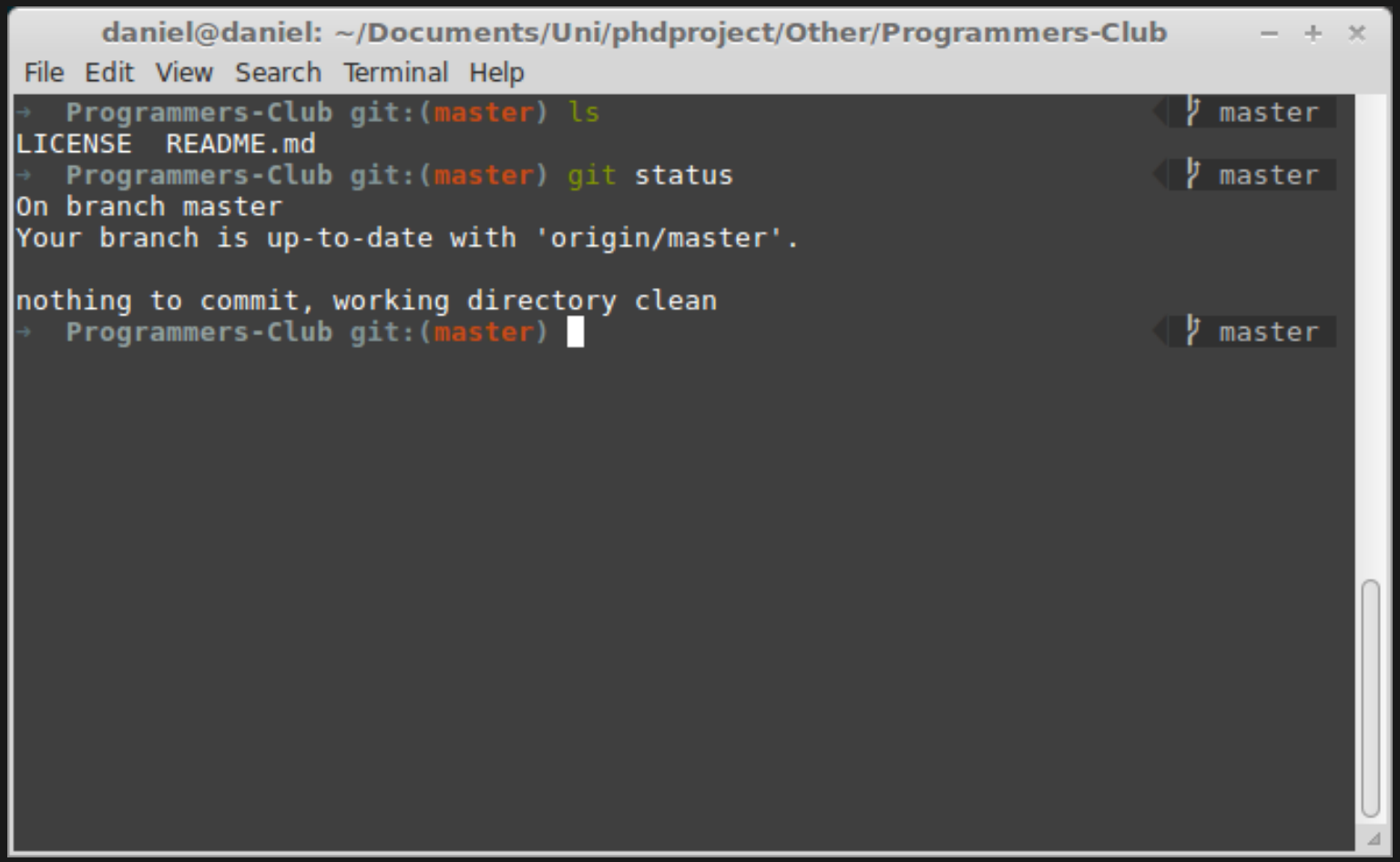
What is git?

- Git is a modern and fast version control (invented by Linus Torvalds).
- A tool to keep track of changes in files with a time stamp and a user-written message ("What did I do and why?").
- Allow the user to "push" files to a server: supernova, GitHub, etc.
- So in the end you have backup *and* all previous versions of files.

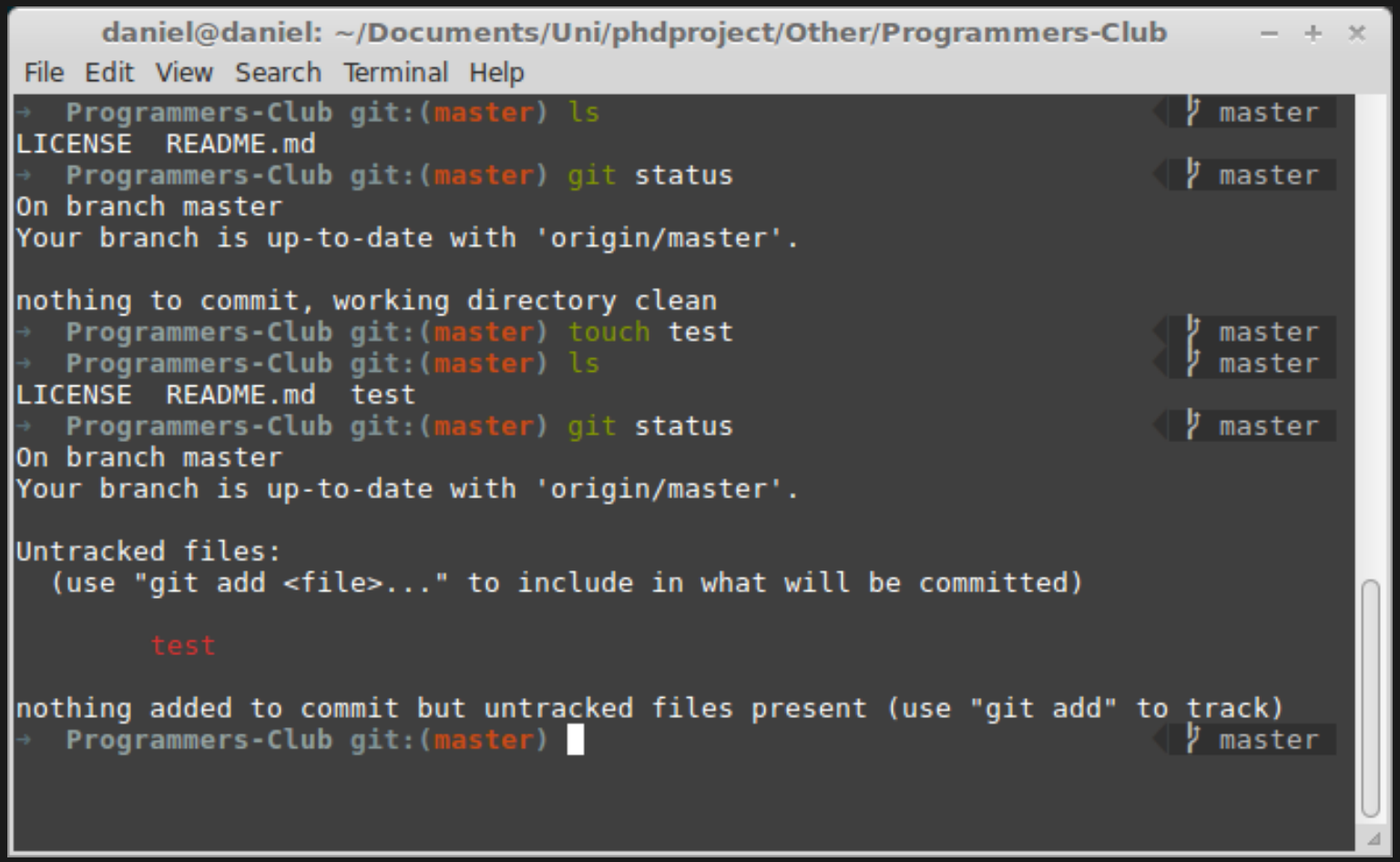
Workflow - forward



Workflow - forward



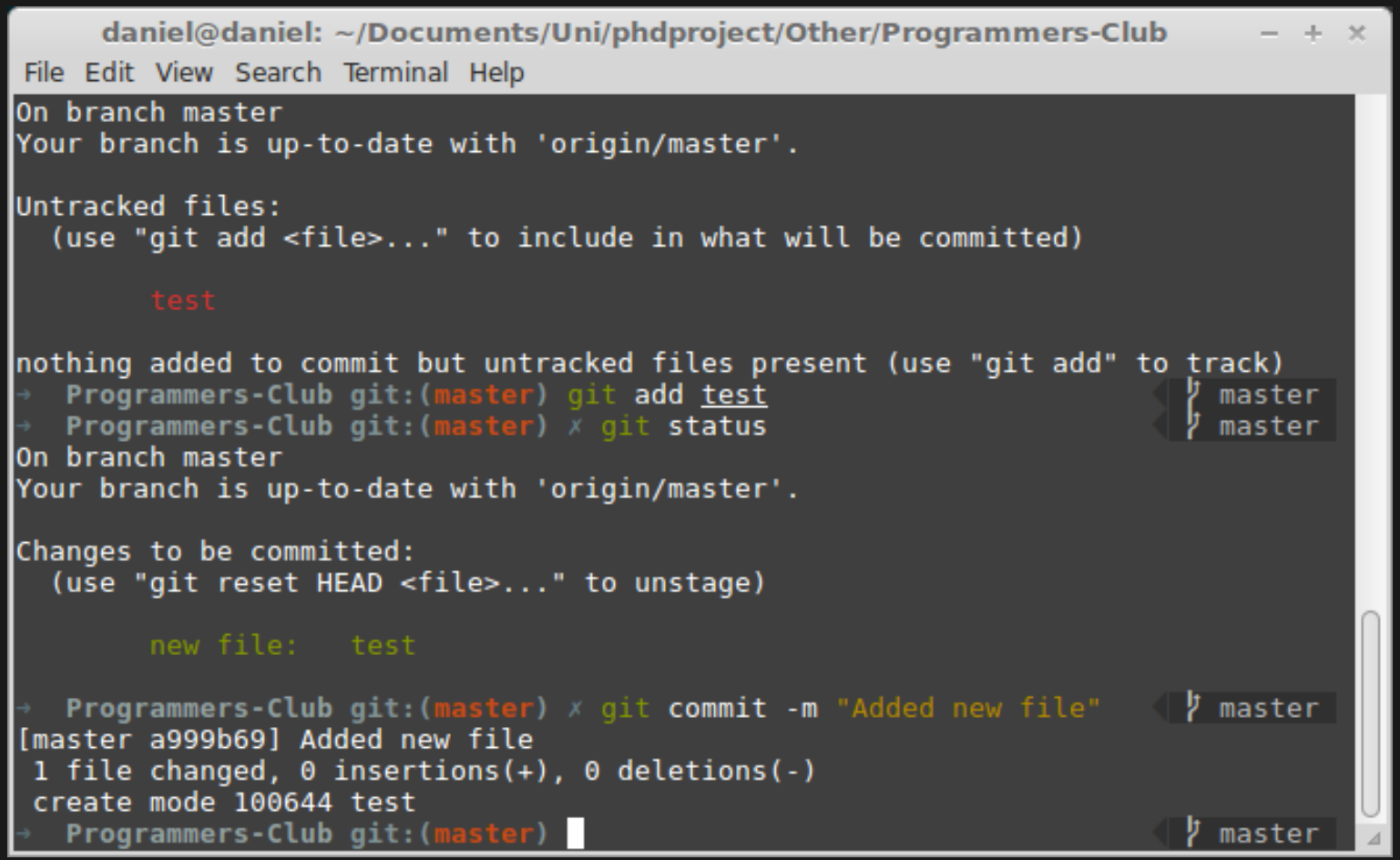
Workflow - forward



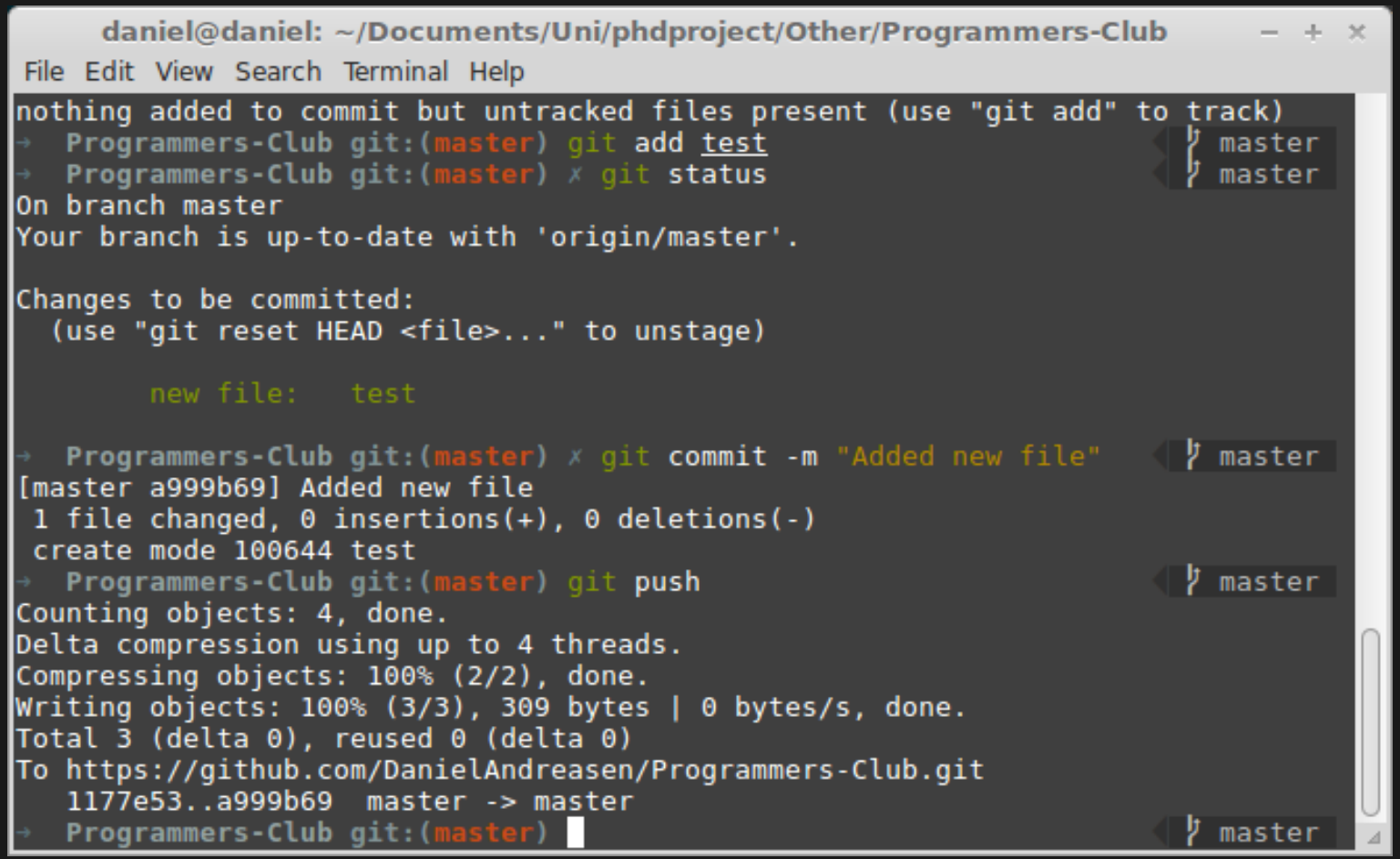
Workflow - forward

```
daniel@daniel: ~/Documents/Uni/phdproject/Other/Programmers-Club
File  Edit  View  Search  Terminal  Help
→  Programmers-Club git:(master) touch test                    master
→  Programmers-Club git:(master) ls                            master
LICENSE  README.md  test
→  Programmers-Club git:(master) git status                    master
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

nothing added to commit but untracked files present (use "git add" to track)
→  Programmers-Club git:(master) git add test                  master
→  Programmers-Club git:(master) ✗ git status                  master
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   test

→  Programmers-Club git:(master) ✗                             master
```

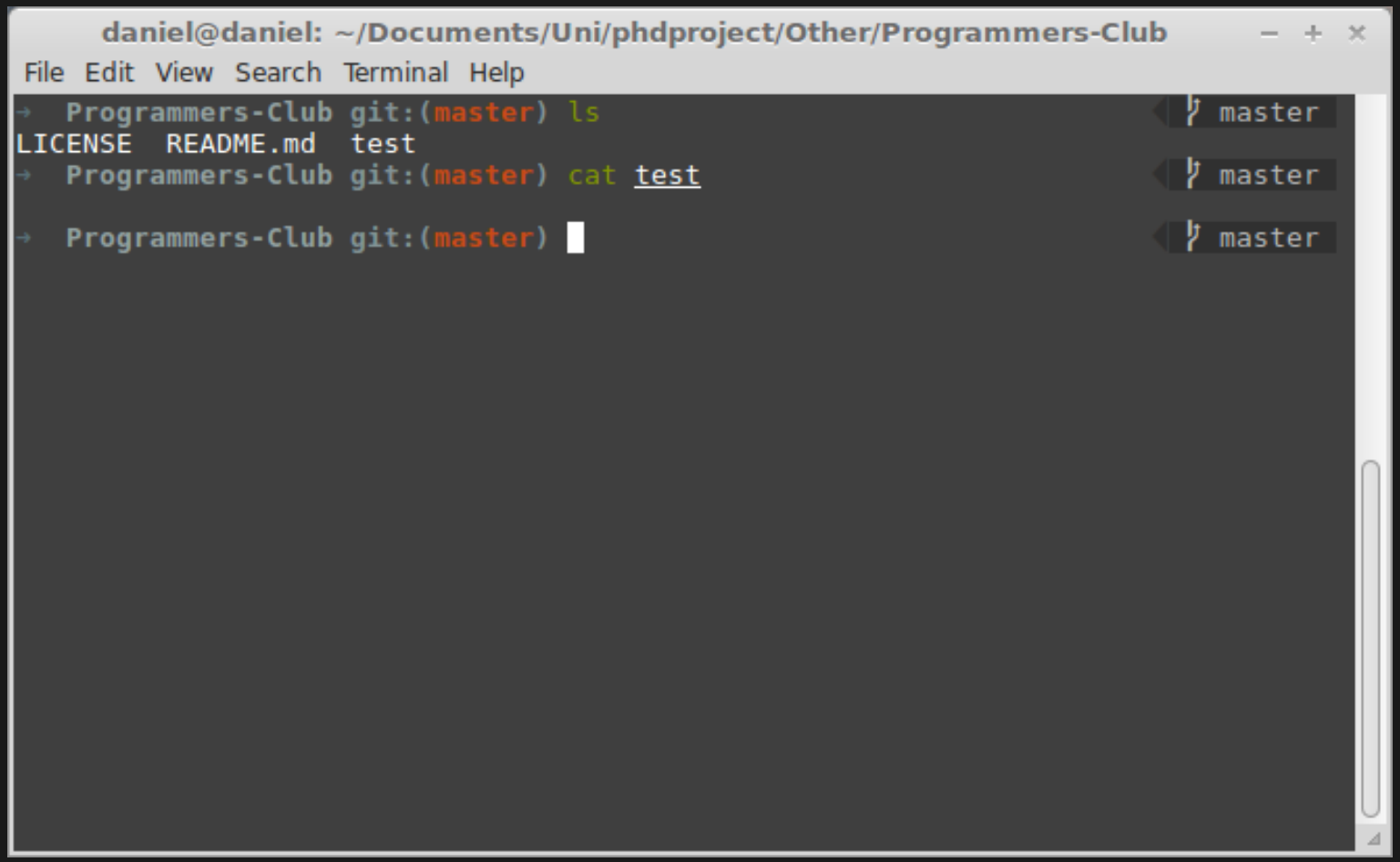
Workflow - forward



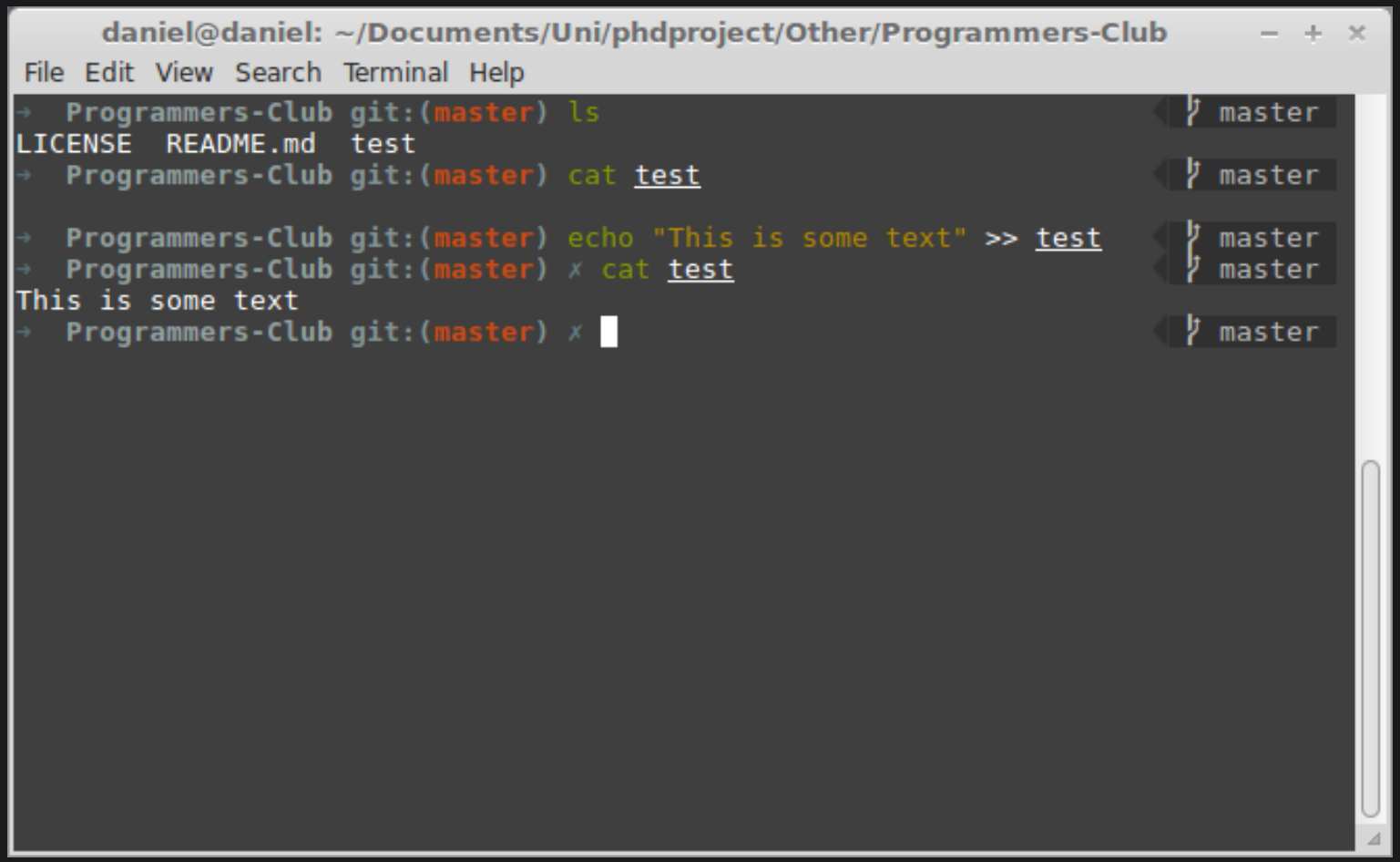
Workflow - forward



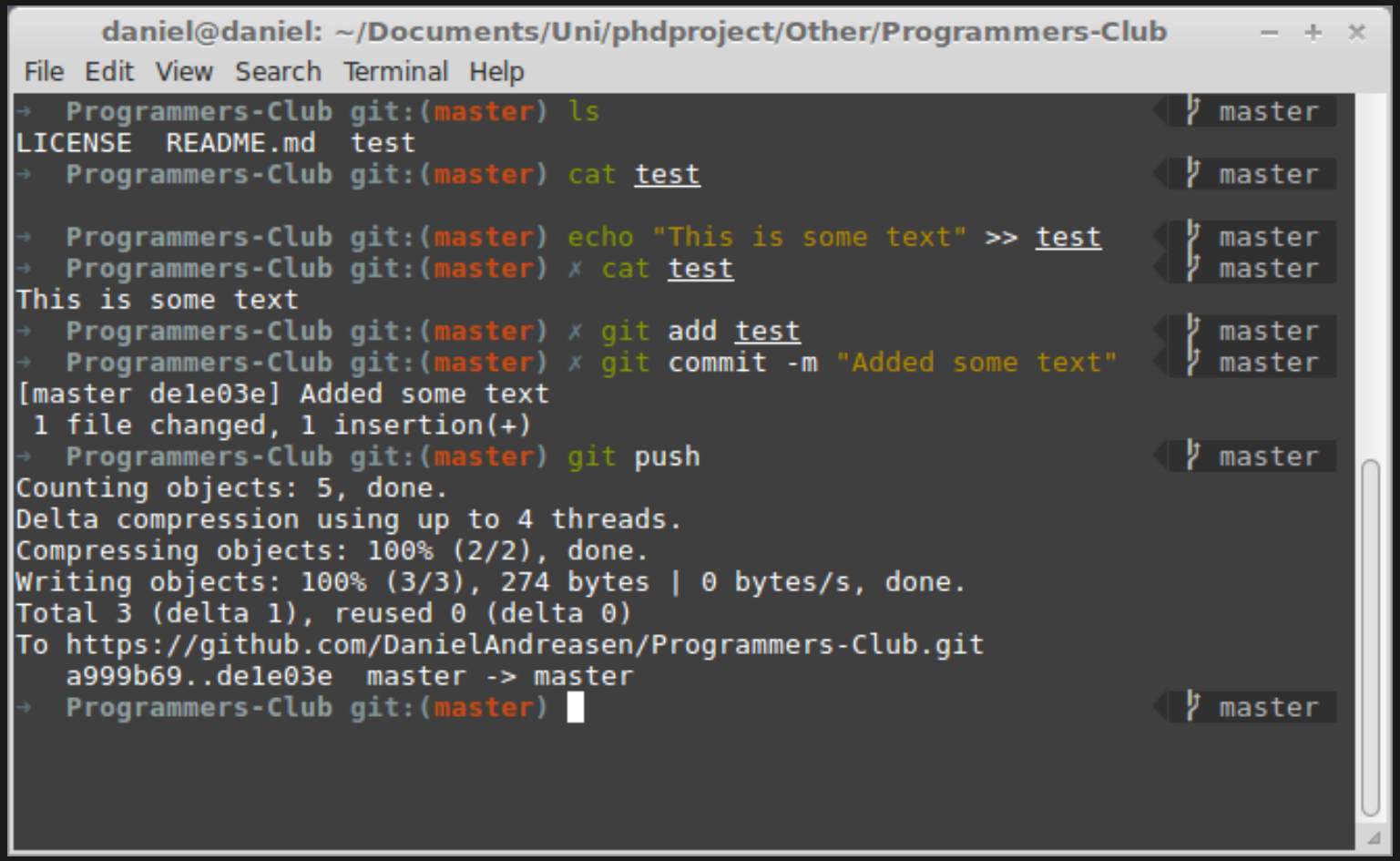
Workflow - backward



Workflow - backward



Workflow - backward



Workflow - backward

git log

File Edit View Search Terminal Help

```
commit de1e03eccb41b540a1023732271139f068b808608
Author: DanielAndreasen <daniel.andreasen@astro.up.pt>
Date:   Tue Jan 27 14:51:23 2015 +0000

    Added some text

commit a999b6957738d3d799b78c25d01e2f6fe964da34
Author: DanielAndreasen <daniel.andreasen@astro.up.pt>
Date:   Tue Jan 27 14:39:55 2015 +0000

    Added new file

commit 1177e532519b3ab918e82b8aadcfa4d1f283f49f
Author: DanielAndreasen <daniel.andreasen@astro.up.pt>
Date:   Mon Jan 26 15:07:51 2015 +0000

    Added wiki

commit a6de123bd044cea3f8b05ff3c3176875deacb538
Author: DanielAndreasen <daniel.andreasen@astro.up.pt>
Date:   Mon Jan 26 15:00:34 2015 +0000

    Added description
:
```

Workflow - backward

daniel@daniel: ~/Documents/Uni/phdproject/Other/Programmers-Club

File Edit View Search Terminal Help

```
 1 file changed, 1 insertion(+)
➜  Programmers-Club git:(master) git push                          master
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 274 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To https://github.com/DanielAndreasen/Programmers-Club.git
   a999b69..de1e03e  master -> master
➜  Programmers-Club git:(master) git log                           master
➜  Programmers-Club git:(master) git checkout a999b6957738         master
Note: checking out 'a999b6957738'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b new_branch_name

HEAD is now at a999b69... Added new file
➜  Programmers-Club git:(a999b69)                                  a999b69
```

Workflow - backward

daniel@daniel: ~/Documents/Uni/phdproject/Other/Programmers-Club

File Edit View Search Terminal Help

```
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 274 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To https://github.com/DanielAndreasen/Programmers-Club.git
   a999b69..de1e03e  master -> master
→  Programmers-Club git:(master) git log                    master
→  Programmers-Club git:(master) git checkout a999b6957738  master
Note: checking out 'a999b6957738'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b new_branch_name

HEAD is now at a999b69... Added new file
→  Programmers-Club git:(a999b69) ls                        a999b69
LICENSE  README.md  test
→  Programmers-Club git:(a999b69) cat test                  a999b69

→  Programmers-Club git:(a999b69)                           a999b69
```

Workflow - backward

daniel@daniel: ~/Documents/Uni/phdproject/Other/Programmers-Club

File Edit View Search Terminal Help

```
To https://github.com/DanielAndreasen/Programmers-Club.git
   a999b69..de1e03e  master -> master
→  Programmers-Club git:(master) git log                          master
→  Programmers-Club git:(master) git checkout a999b6957738        master
Note: checking out 'a999b6957738'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b new_branch_name

HEAD is now at a999b69... Added new file
→  Programmers-Club git:(a999b69) ls                              a999b69
LICENSE  README.md  test
→  Programmers-Club git:(a999b69) cat test                        a999b69

→  Programmers-Club git:(a999b69) git branch                      a999b69
* (detached from a999b69)
  master
→  Programmers-Club git:(a999b69)                                 a999b69
```

Workflow - backward

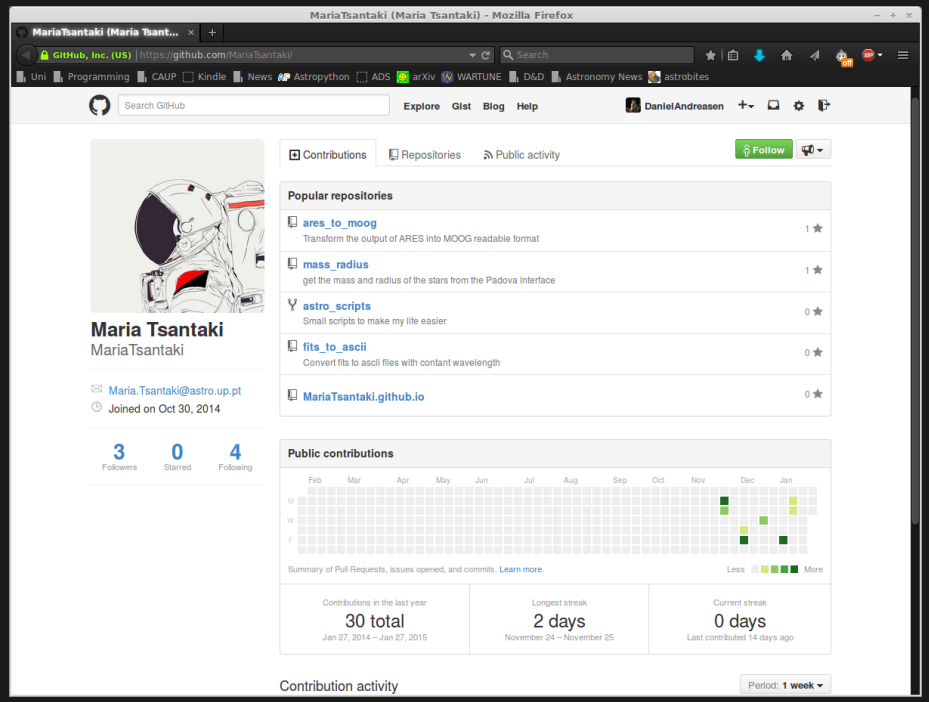
daniel@daniel: ~/Documents/Uni/phdproject/Other/Programmers-Club

File Edit View Search Terminal Help

```
You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b new_branch_name

HEAD is now at a999b69... Added new file
→  Programmers-Club git:(a999b69) ls                                  a999b69
LICENSE  README.md  test
→  Programmers-Club git:(a999b69) cat test                            a999b69

→  Programmers-Club git:(a999b69) git branch                          a999b69
* (detached from a999b69)
  master
→  Programmers-Club git:(a999b69) git checkout master                 a999b69
Previous HEAD position was a999b69... Added new file
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
→  Programmers-Club git:(master) cat test                              master
This is some text
→  Programmers-Club git:(master)                                      master
```

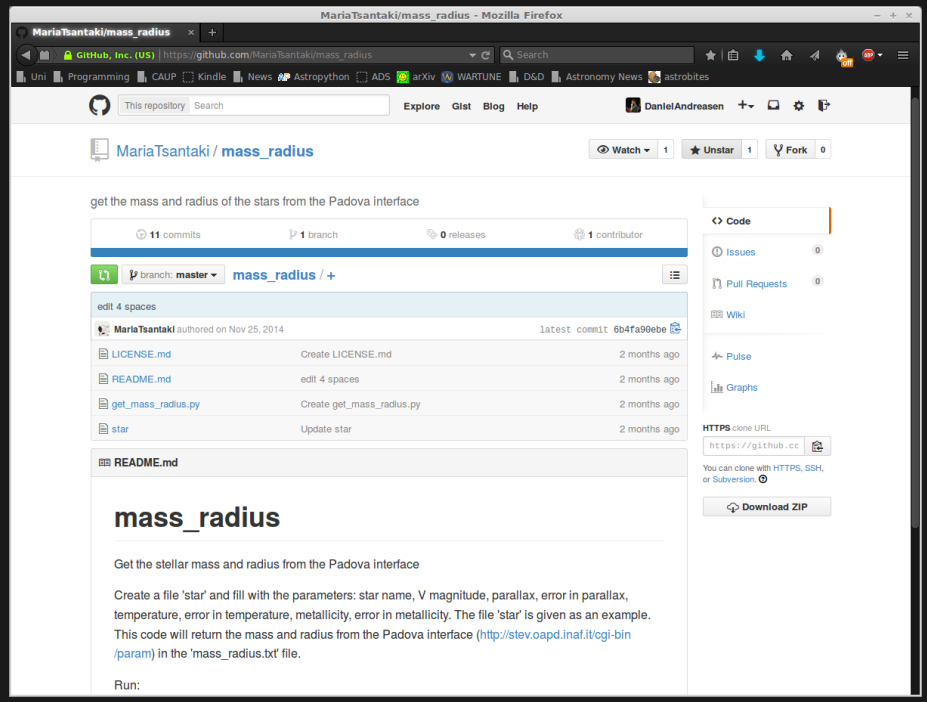
Getting other's code

- Follow people you like (social).



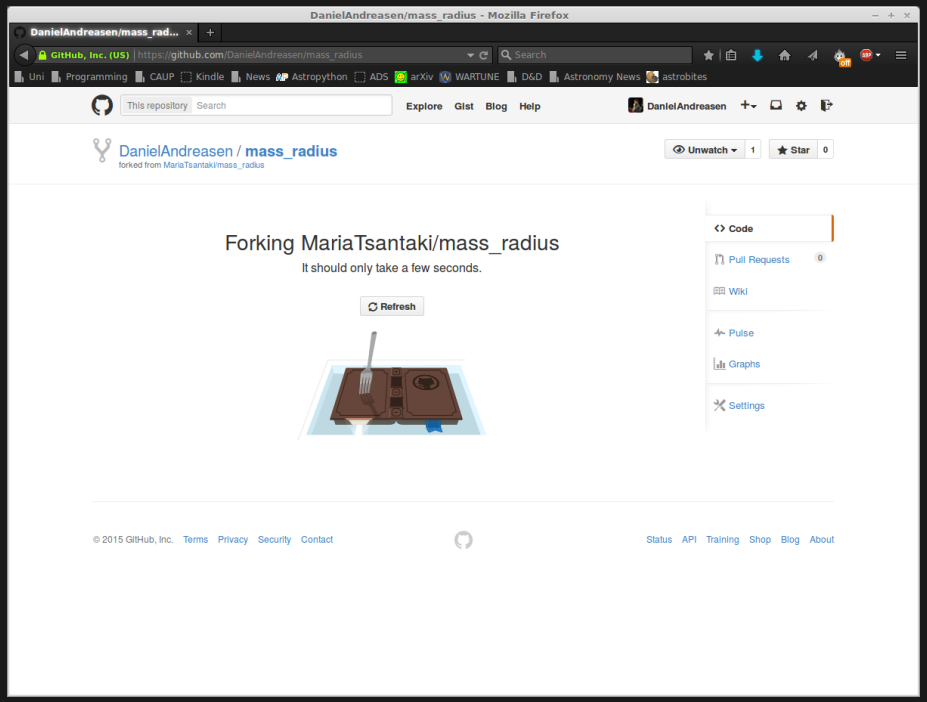
Getting other's code

- Follow people you like (social).
- Find the code you want to use.



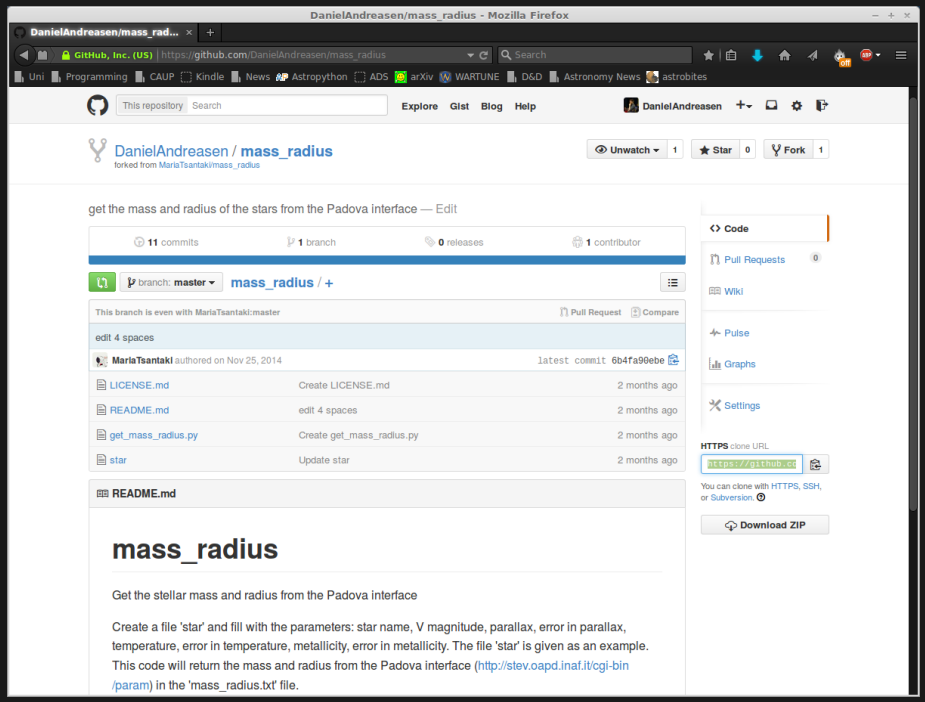
Getting other's code

- Follow people you like (social).
- Find the code you want to use.
- Fork the project.



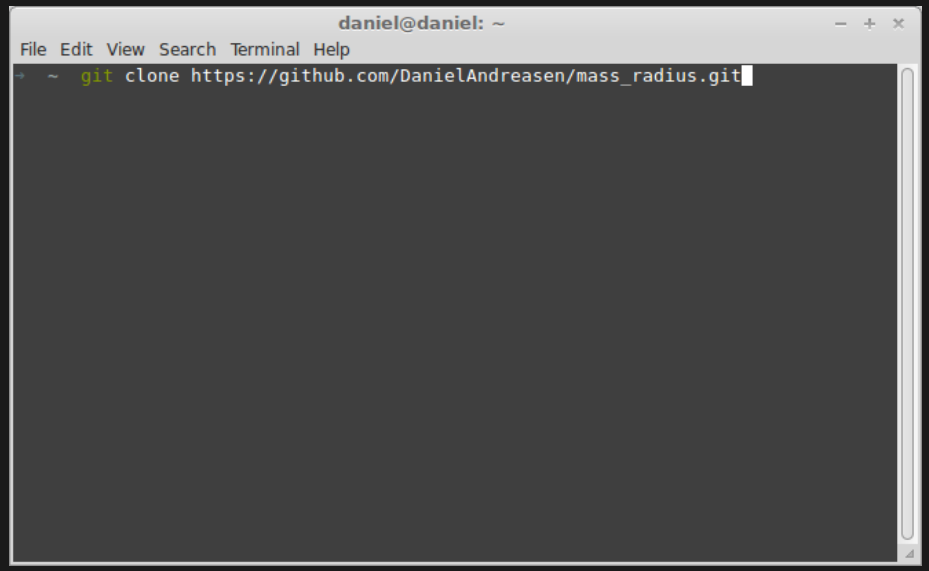
Getting other's code

- Follow people you like (social).
- Find the code you want to use.
- Fork the project.
- Clone to your local computer.



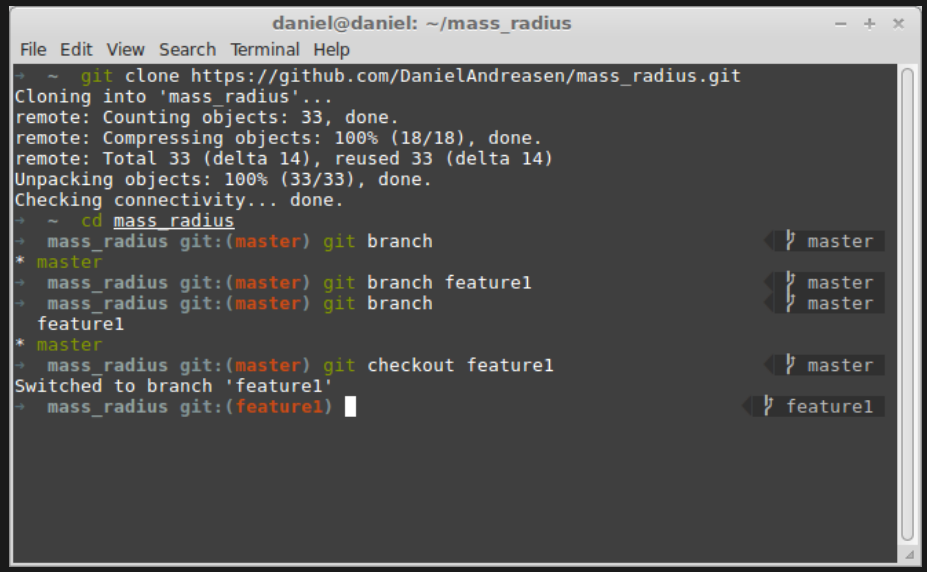
Getting other's code

- Follow people you like (social).
- Find the code you want to use.
- Fork the project.
- Clone to your local computer.



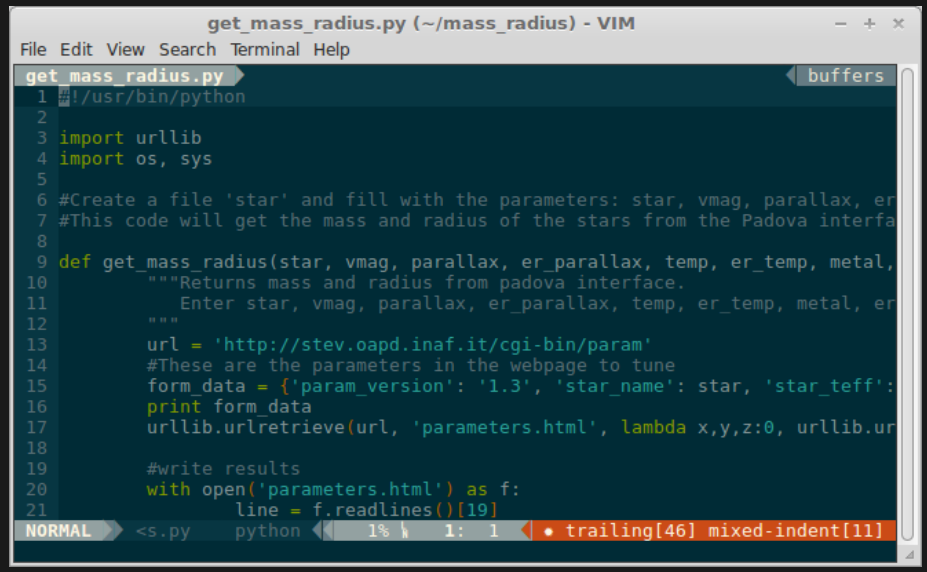
Branching

- New branch (feature1)



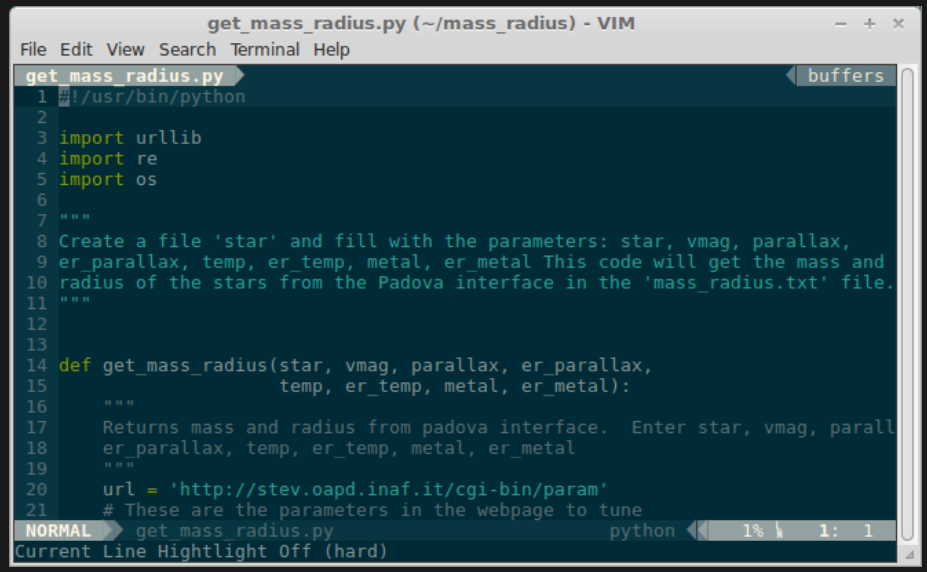
Branching

- New branch (feature1)
- Implement your changes



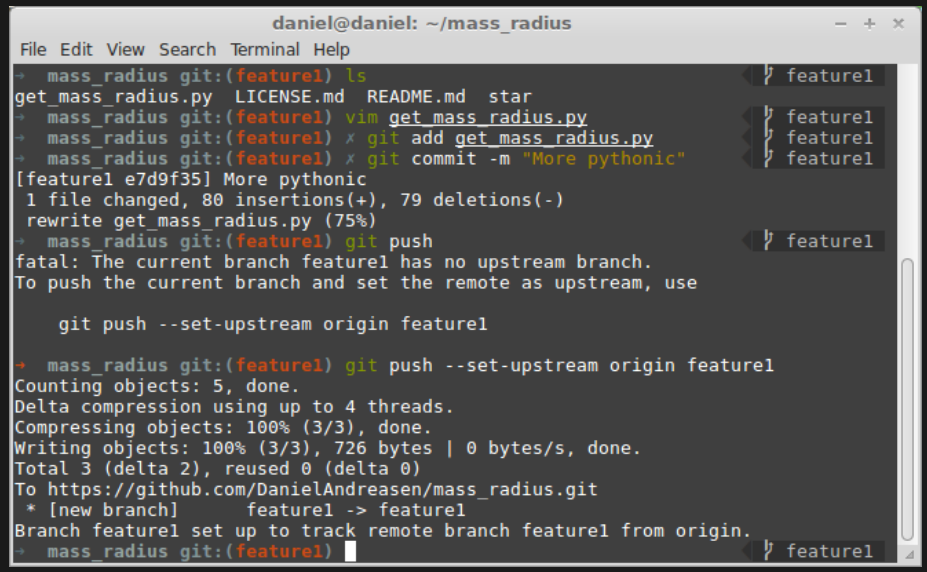
Branching

- New branch (feature1)
- Implement your changes



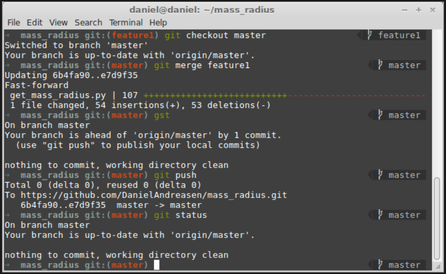
Branching

- New branch (feature1)
- Implement your changes
- Push your branch



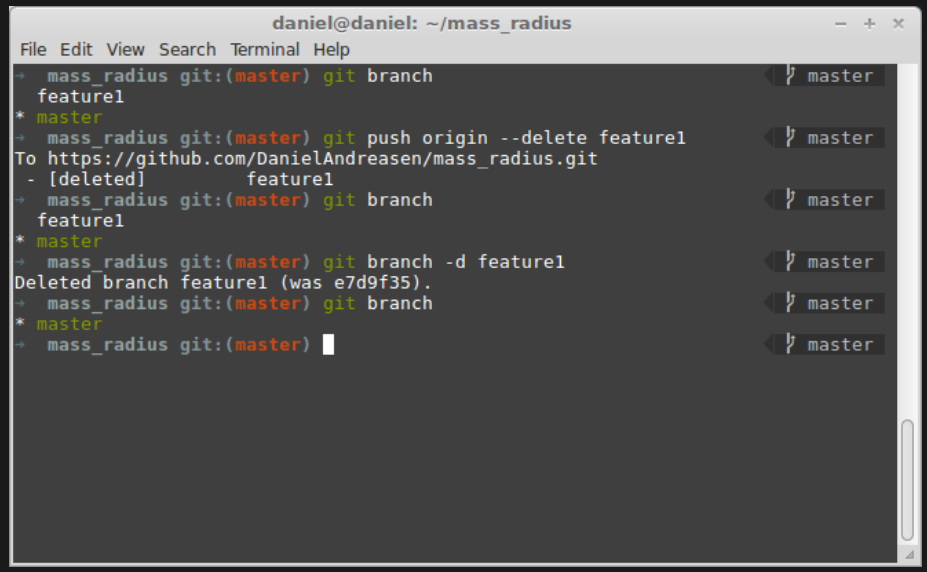
Branching

- New branch (feature1)
- Implement your changes
- Push your branch
- Merge your branch with master

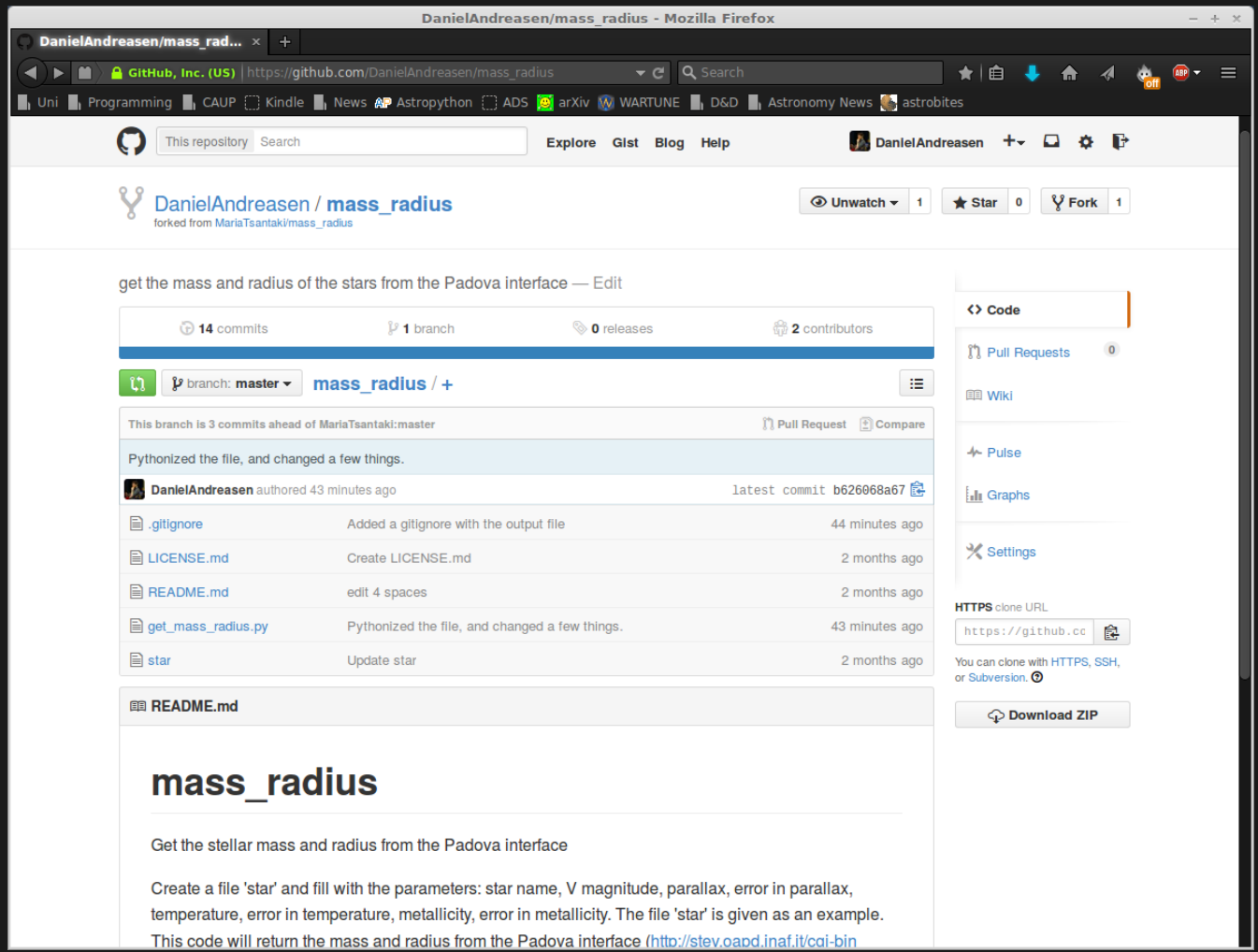


Branching

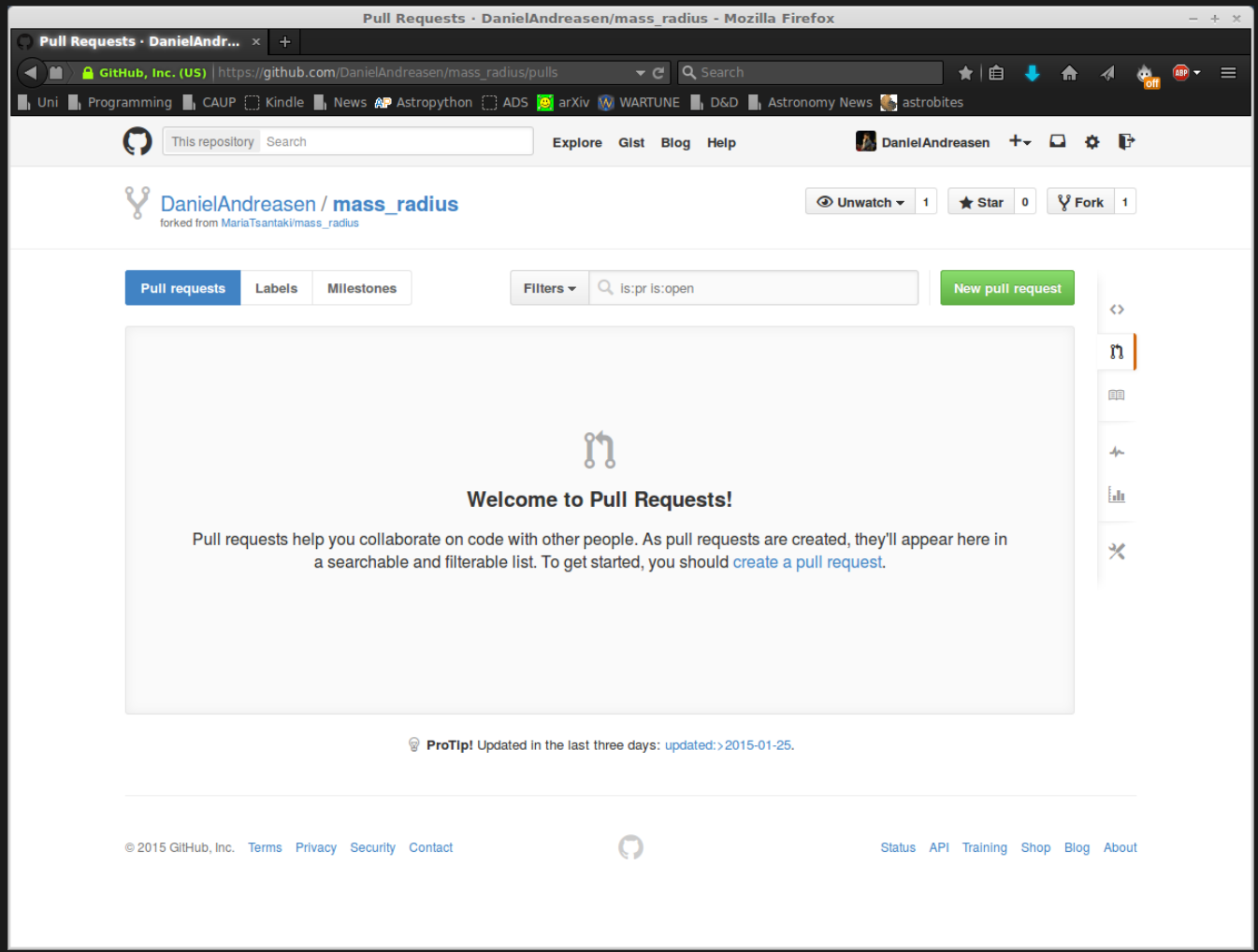
- New branch (feature1)
- Implement your changes
- Push your branch
- Merge your branch with master
- Clean up



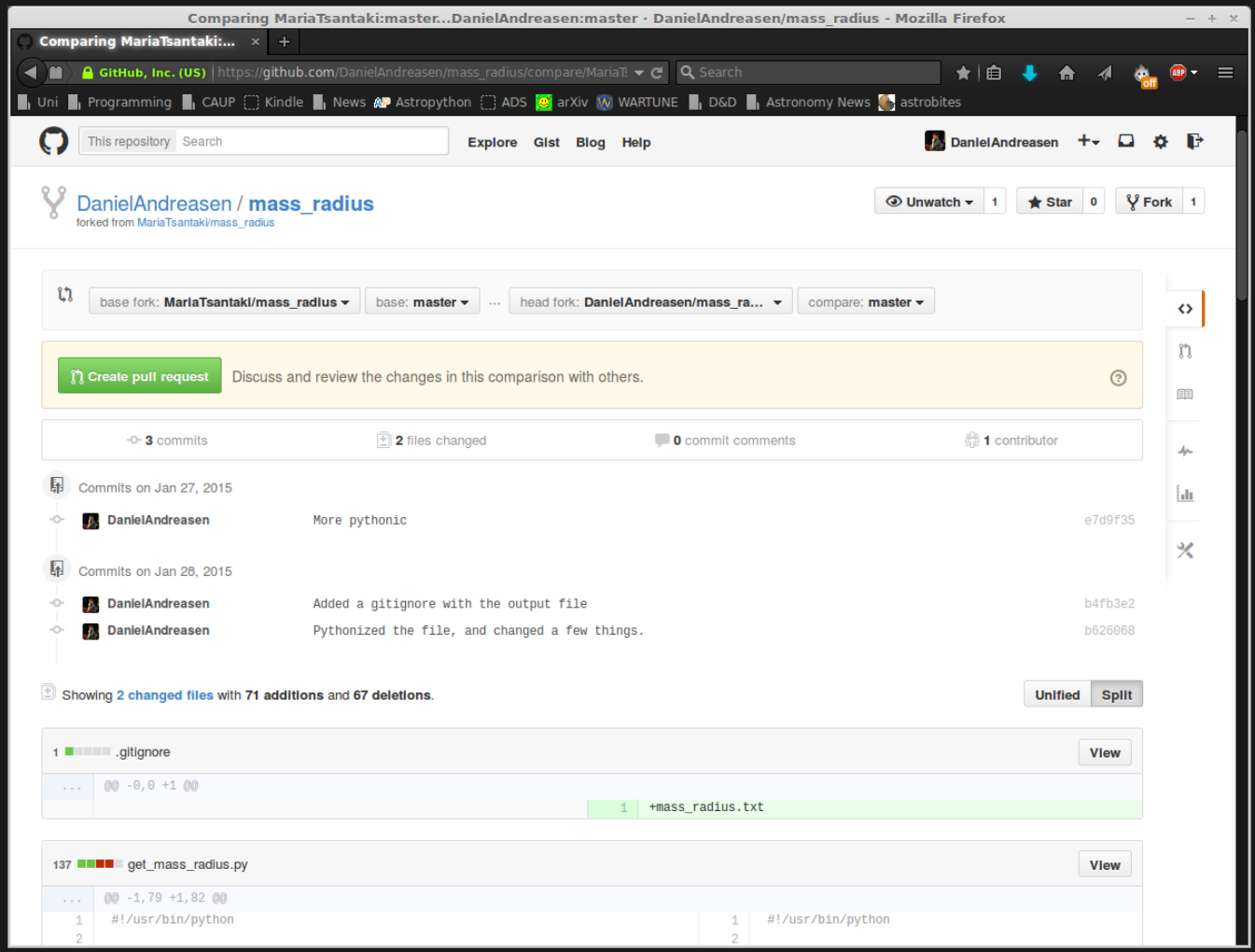
Pull request



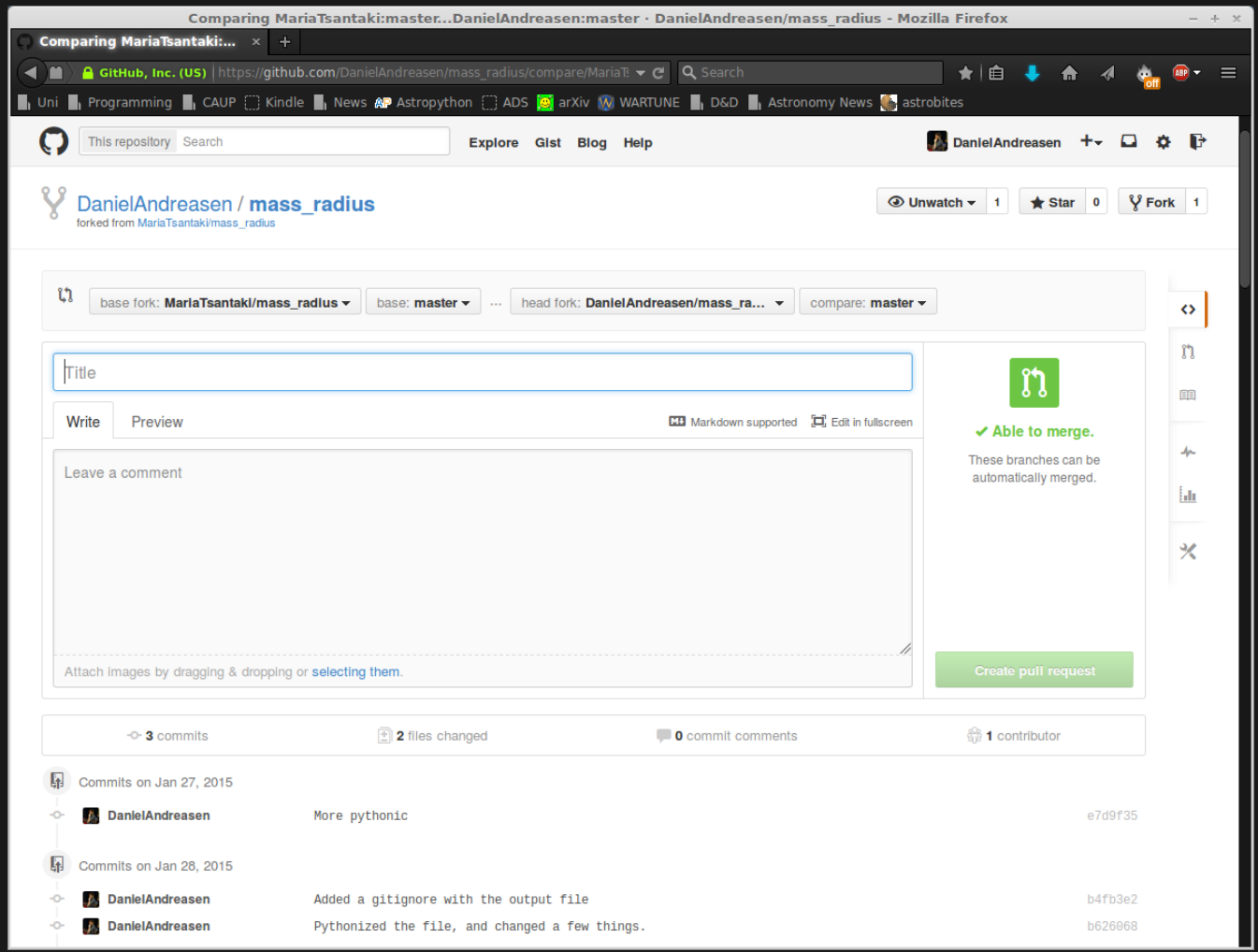
Pull request



Pull request



Pull request



Bonus slides: git cheat sheet

- **Most used commands (all starting with git)**

status:	Show status on files in directory
add:	Add <files> to be commited/updated
commit:	Attach a <message> (what did I do and why) to the added files
push:	Push the commited files to a server (GitHub/supernova/etc.)
log:	Show a commit log (author, time stamp, message, and a key)
branch:	Create a branch of the work with <name>. Good for adding new features.
checkout:	Switch to a different branch (<name>) or commit (<key>).
blame:	See who did what and when on a file.
clone:	Get a repository locally with the link provided on GitHub.
pull:	Get the latest updates from a code (if working on multiple computers or with collaborators).