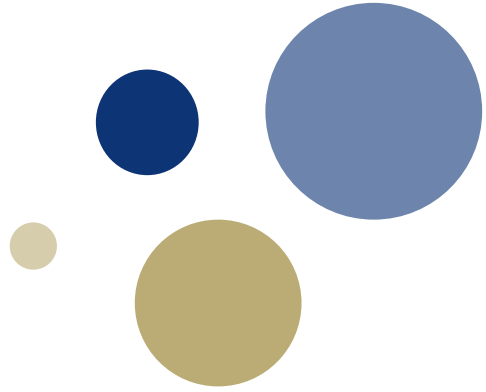


Version control for researchers

Sigurd Hofsmo Jakobsen

Department of electric power engineering

June 21, 2018

Outline

Taking over some old code and fixing it

Version control using git

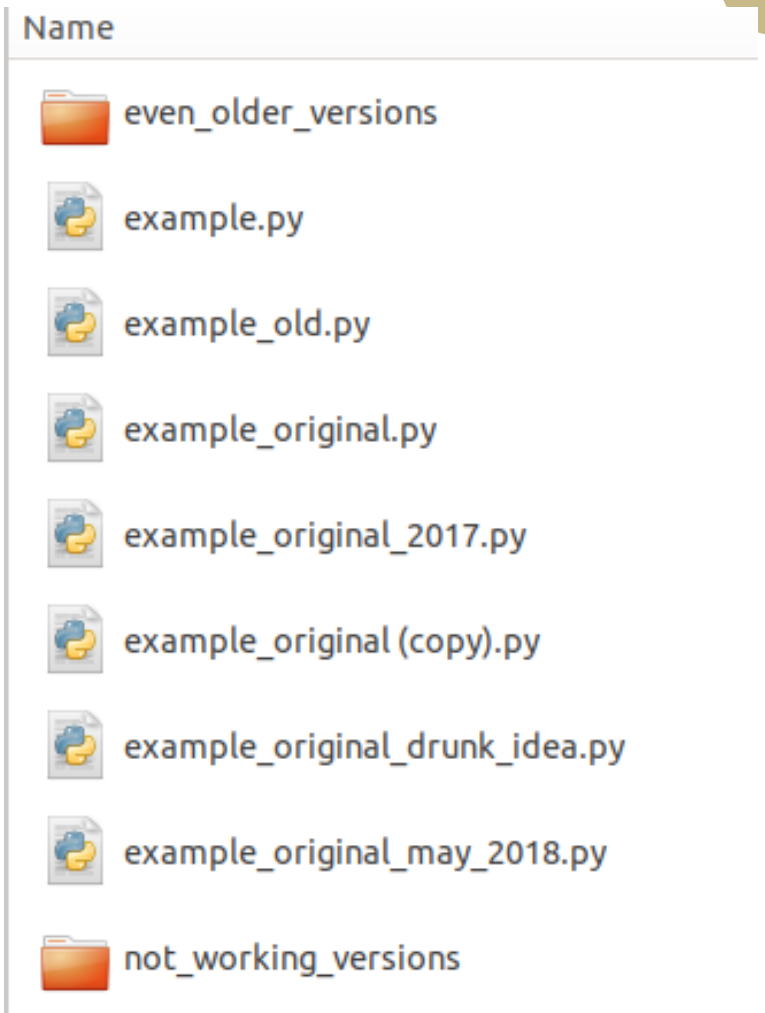
Making the code more readable

More topics to be considered

Discussion

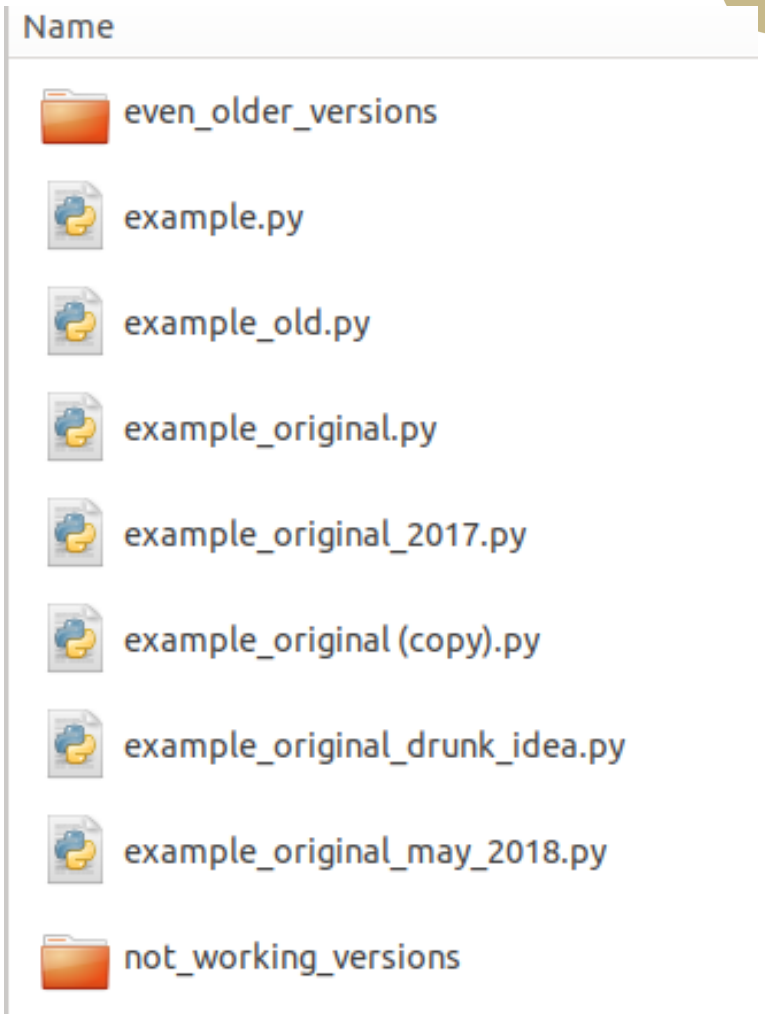
Presentation of the case

— Some master students are going to continue the work of a PhD



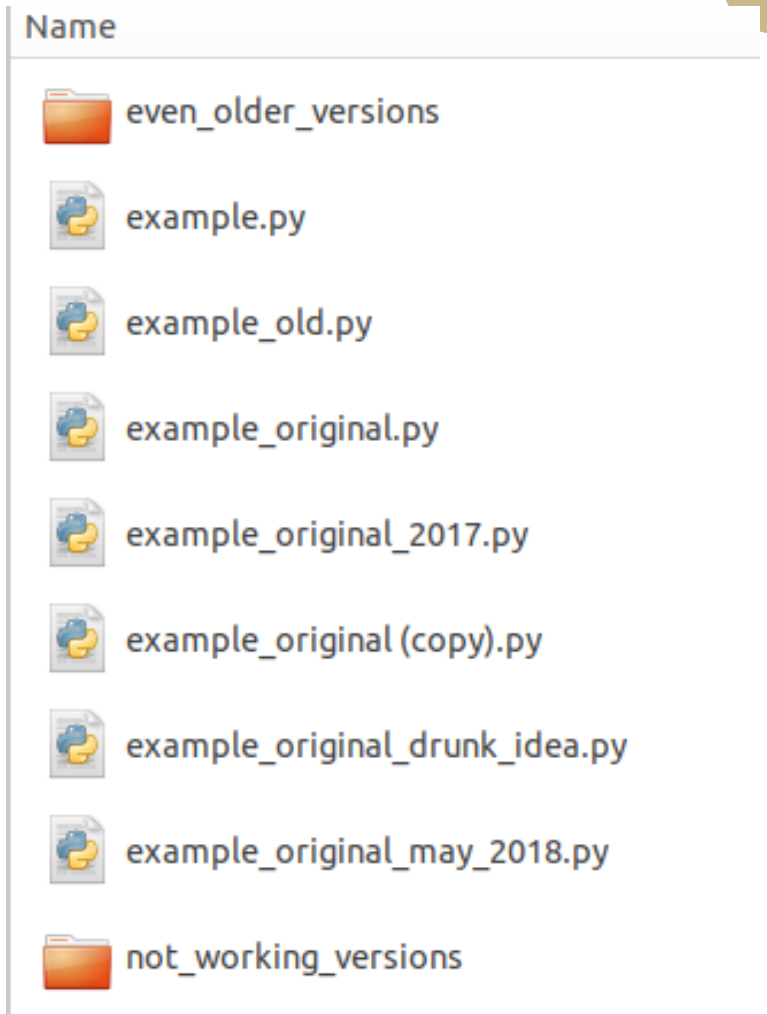
Presentation of the case

- Some master students are going to continue the work of a PhD
- They received several files on a USB stick (don't tell the it guy)



Presentation of the case

- Some master students are going to continue the work of a PhD
- They received several files on a USB stick (don't tell the it guy)
- There are no explanation on what the files do



Presentation of the case

- Some master students are going to continue the work of a PhD
- They received several files on a USB stick (don't tell the it guy)
- There are no explanation on what the files do
- Looking at one of the files doesn't help

```
import csv
import os
a = "../data"
b = "blood_thirsty_janitors.csv"
with open(os.path.join(a, b)) as cf:
    r = csv.DictReader(cf)
    temp_1 = 0
    temp_2 = 0
    temp_3 = 0
    temp_4 = 0
    for rr in r:
        temp_1 += int(rr["beard length"])
        temp_2 += int(rr["churches burned"])
        temp_3 += int(rr["age"])
        temp_4 += 1
a_b_l = temp_1/temp_4
cbl = temp_2/temp_1
aa = temp_3/temp_4
print("Calculations done")
b = "evil_vikings.csv"
with open(os.path.join(a, b)) as cf:
    r = csv.DictReader(cf)
    temp_1 = 0
    temp_2 = 0
    temp_3 = 0
    temp_4 = 0
    for rr in r:
        temp_1 += int(rr["beard length"])
        temp_2 += int(rr["churches burned"])
        temp_3 += int(rr["age"])
        temp_4 += 1
a_b_l = temp_1/temp_4
cbl = temp_2/temp_1
aa = temp_3/temp_4
print("Calculations done")
b = "hungry_crusaders.csv"
with open(os.path.join(a, b)) as cf:
    r = csv.DictReader(cf)
    temp_1 = 0
    temp_2 = 0
    temp_3 = 0
    temp_4 = 0
    for rr in r:
        temp_1 += int(rr["beard length"])
        temp_2 += int(rr["churches burned"])
        temp_3 += int(rr["age"])
        temp_4 += 1
a_b_l = temp_1/temp_4
cbl = temp_2/temp_1
aa = temp_3/temp_4
print("Calculations done")
```

Problems which have to be solved

— How do they share the code with each other?

Problems which have to be solved

— How do they share the code with each other?
  - With version control.

Problems which have to be solved

— How do they share the code with each other?
  - With version control.

— How do they keep track of the old versions?

Problems which have to be solved

— How do they share the code with each other?
  - With version control.

— How do they keep track of the old versions?
  - With version control.

Problems which have to be solved

— How do they share the code with each other?
  - With version control.

— How do they keep track of the old versions?
  - With version control.

— How do they make the code readable

Problems which have to be solved

- How do they share the code with each other?
  - With version control.
- How do they keep track of the old versions?
  - With version control.
- How do they make the code readable
  - Implement and follow code style guidelines

Problems which have to be solved

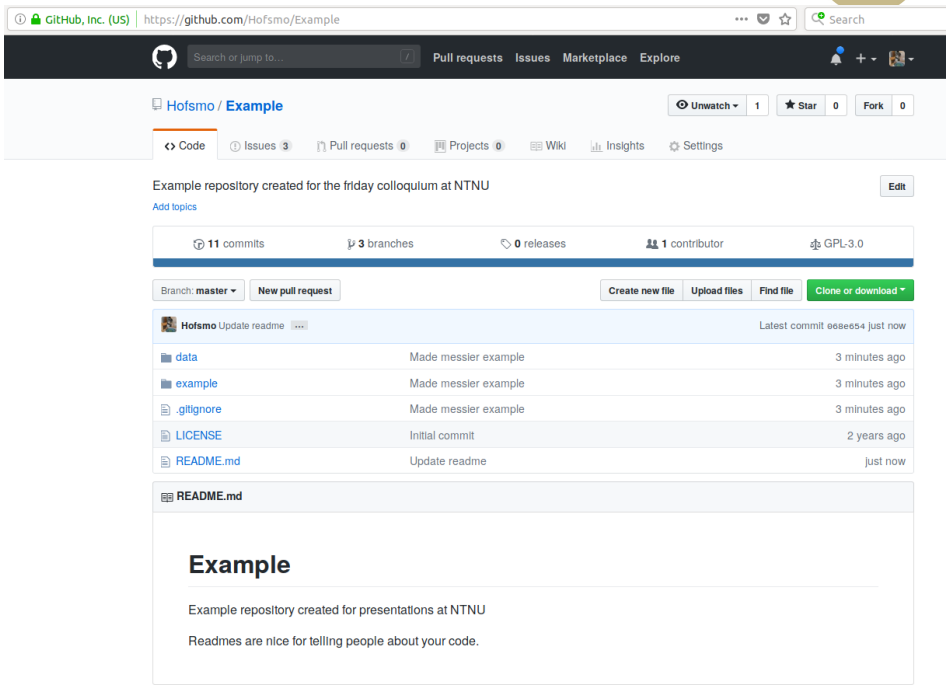
- How do they share the code with each other?
  - With version control.
- How do they keep track of the old versions?
  - With version control.
- How do they make the code readable
  - Implement and follow code style guidelines
- How do prevent copy paste?

Problems which have to be solved

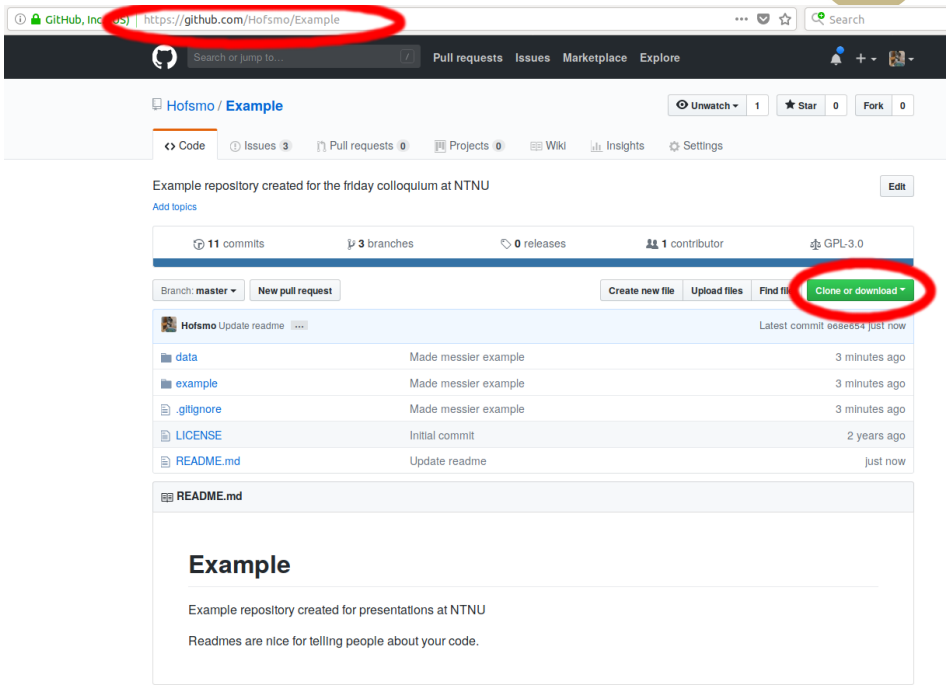
- How do they share the code with each other?
  - With version control.
- How do they keep track of the old versions?
  - With version control.
- How do they make the code readable
  - Implement and follow code style guidelines
- How do prevent copy paste?
  - Create functions.

Outline

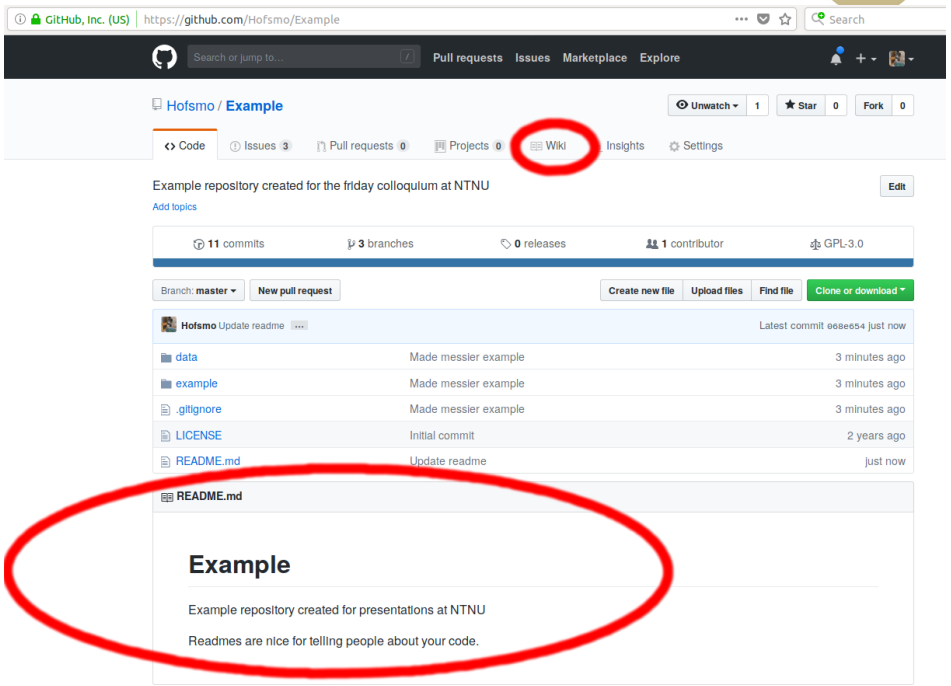
— The code can easily be shared online on GitHub



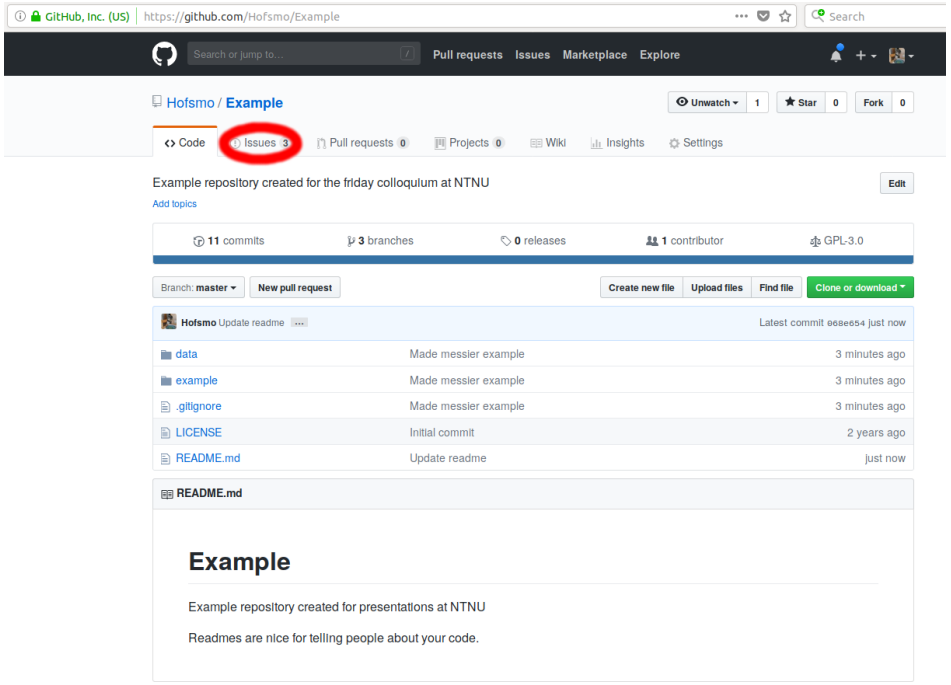
- The code can easily be shared online on GitHub
- Everyone can access and download the code



- The code can easily be shared online on GitHub
- Everyone can access and download the code
- Readme and wiki for documenting the code

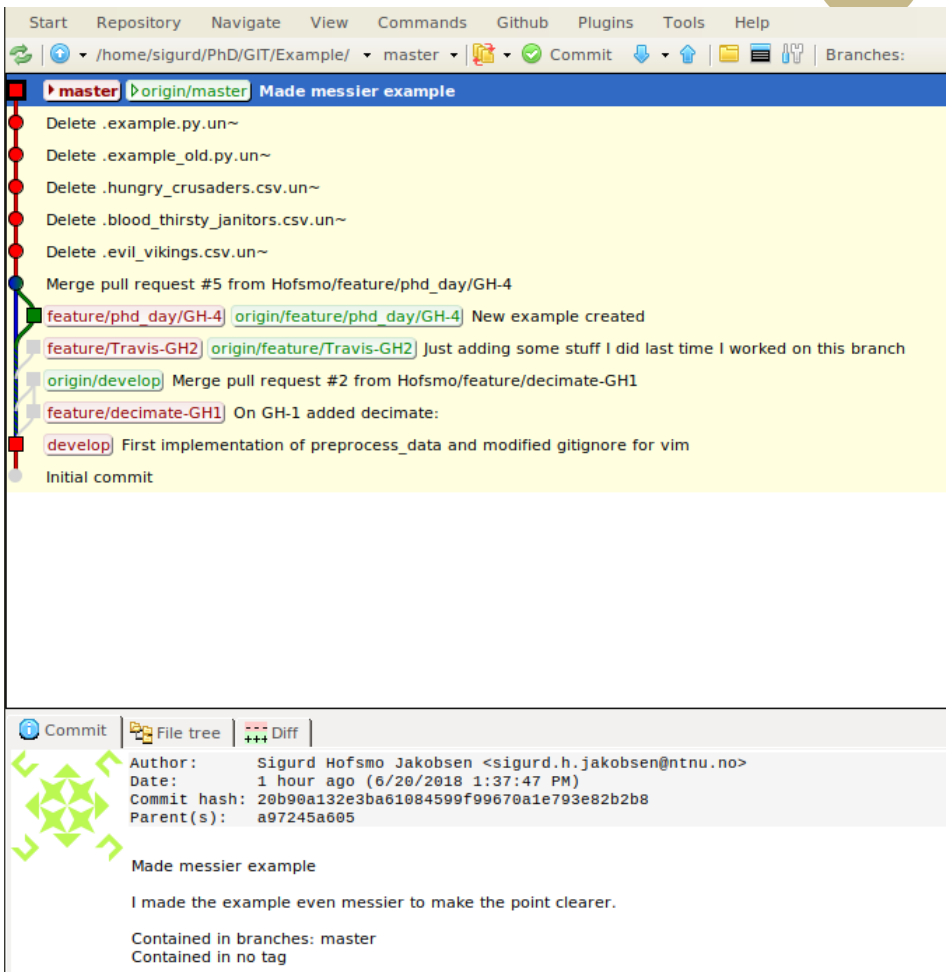


- The code can easily be shared online on GitHub
- Everyone can access and download the code
- Readme and wiki for documenting the code
- Issue tracker for reporting bugs and discussing the code



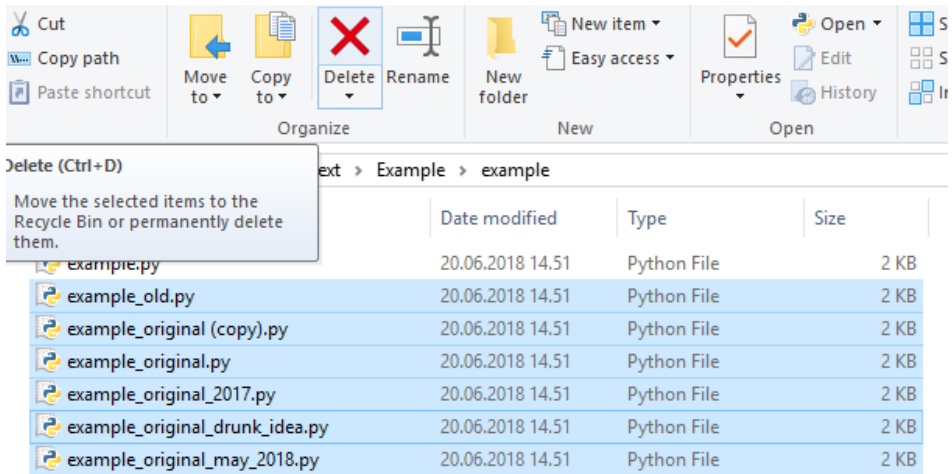
Getting rid of the old files using git

— Download the GitHub repository using a GUI or the terminal.



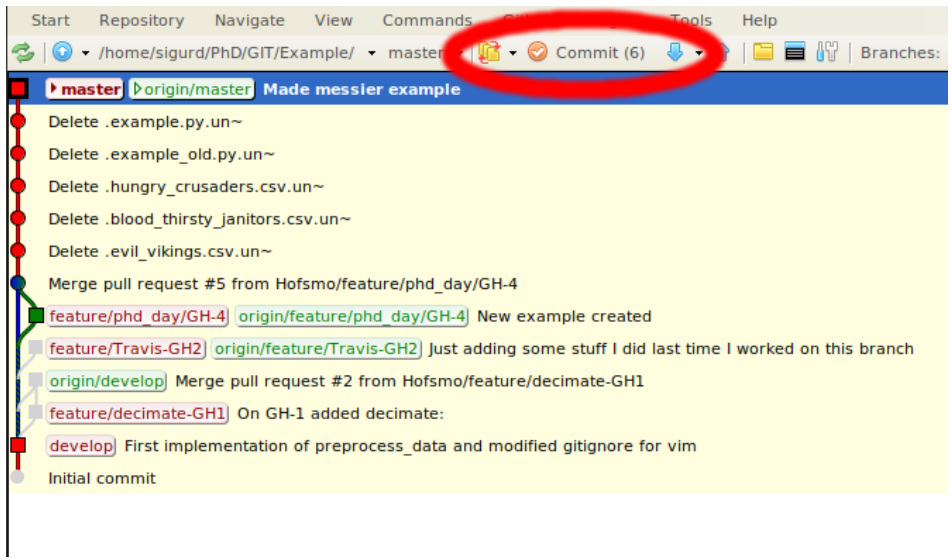
Getting rid of the old files using git

- Download the GitHub repository using a GUI or the terminal.
- Find the files and delete them.



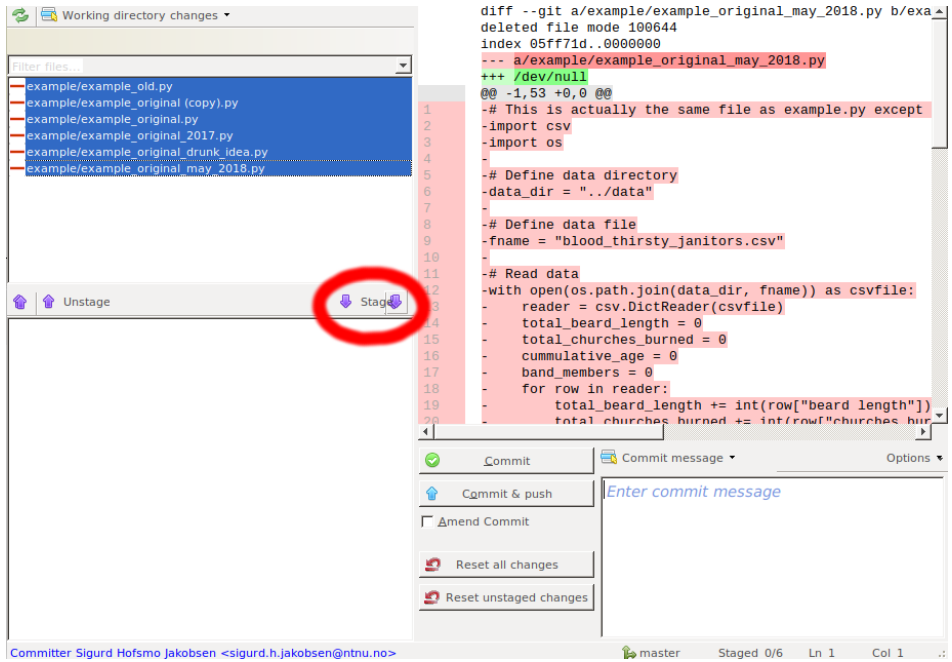
Getting rid of the old files using git

- Download the GitHub repository using a GUI or the terminal.
- Find the files and delete them.
- The GUI informs you that changes have been made.



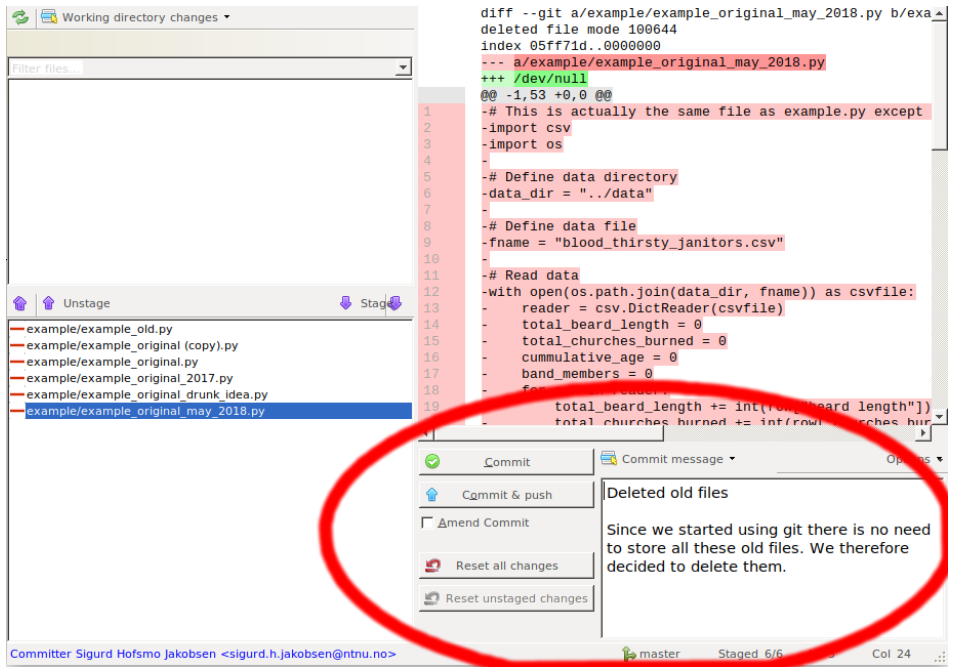
Getting rid of the old files using git

- Download the GitHub repository using a GUI or the terminal.
- Find the files and delete them.
- The GUI informs you that changes have been made.
- The changes have to be staged.



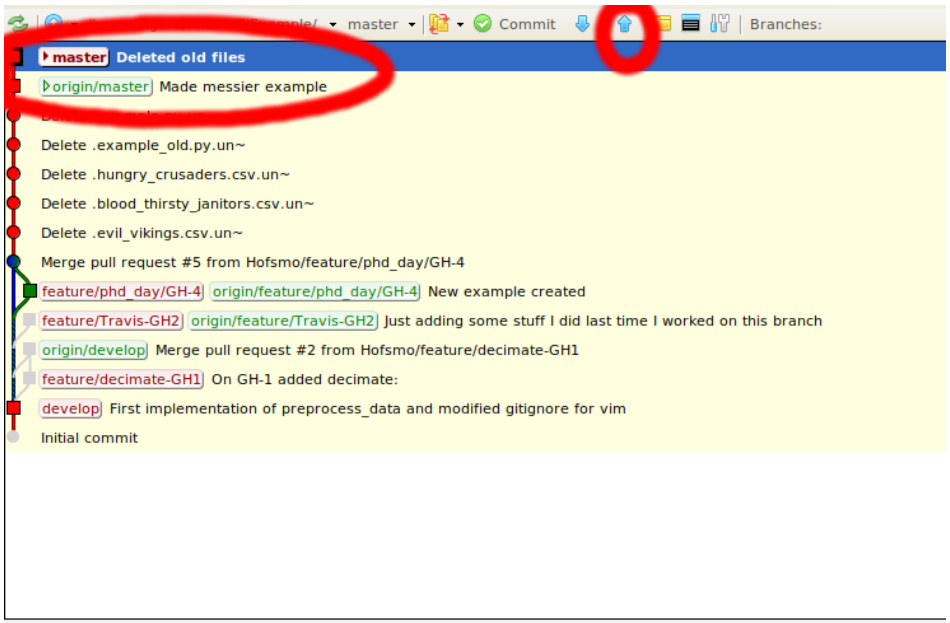
Getting rid of the old files using git

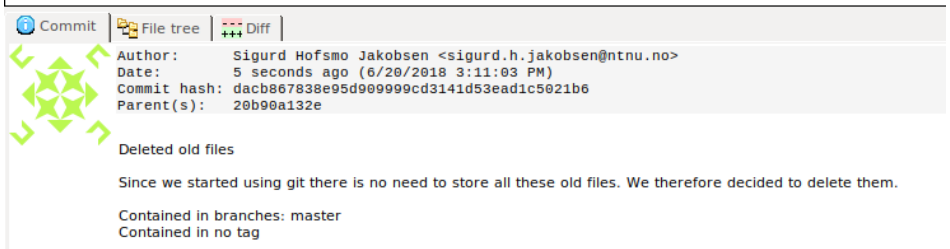
- Download the GitHub repository using a GUI or the terminal.
- Find the files and delete them.
- The GUI informs you that changes have been made.
- The changes have to be staged.
- Write a commit message and commit the change.



Getting rid of the old files using git

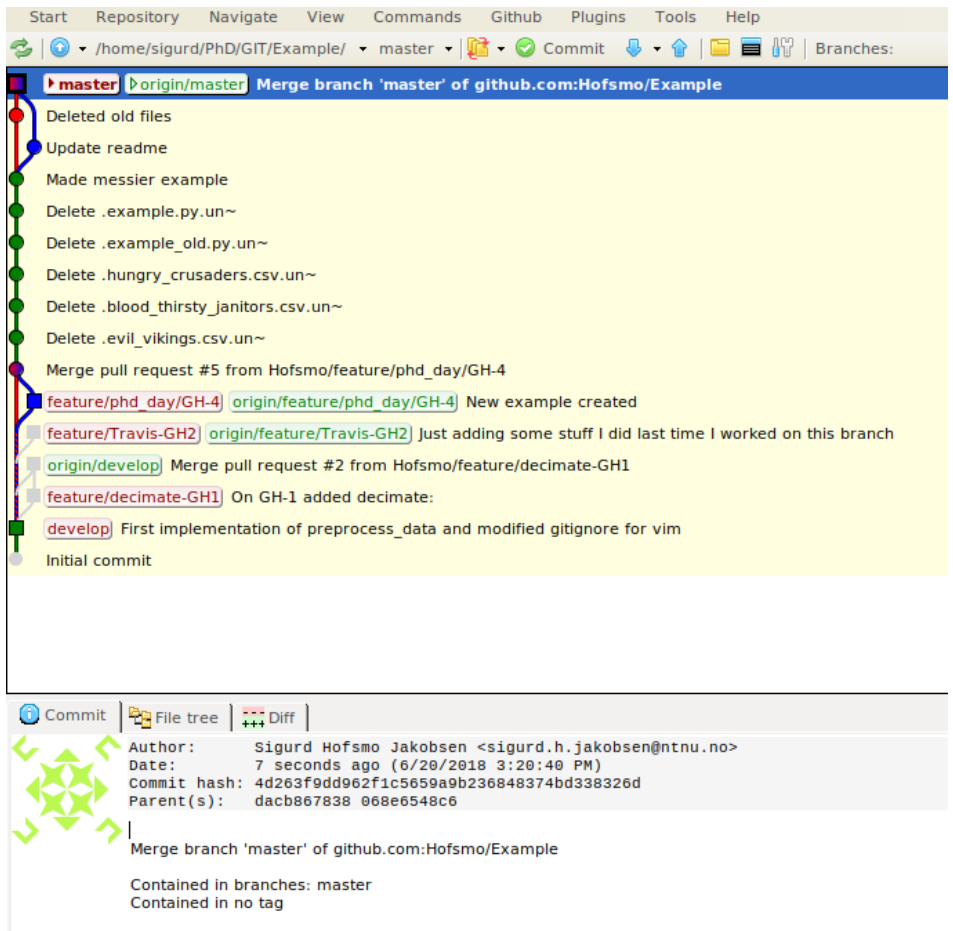
- Download the GitHub repository using a GUI or the terminal.
- Find the files and delete them.
- The GUI informs you that changes have been made.
- The changes have to be staged.
- Write a commit message and commit the change.
- Our local version is now ahead of the GitHub version and the changes can be pushed.





Getting rid of the old files using git

- Download the GitHub repository using a GUI or the terminal.
- Find the files and delete them.
- The GUI informs you that changes have been made.
- The changes have to be staged.
- Write a commit message and commit the change.
- Our local version is now ahead of the GitHub version and the changes can be pushed.
- The versions are now identical



Outline

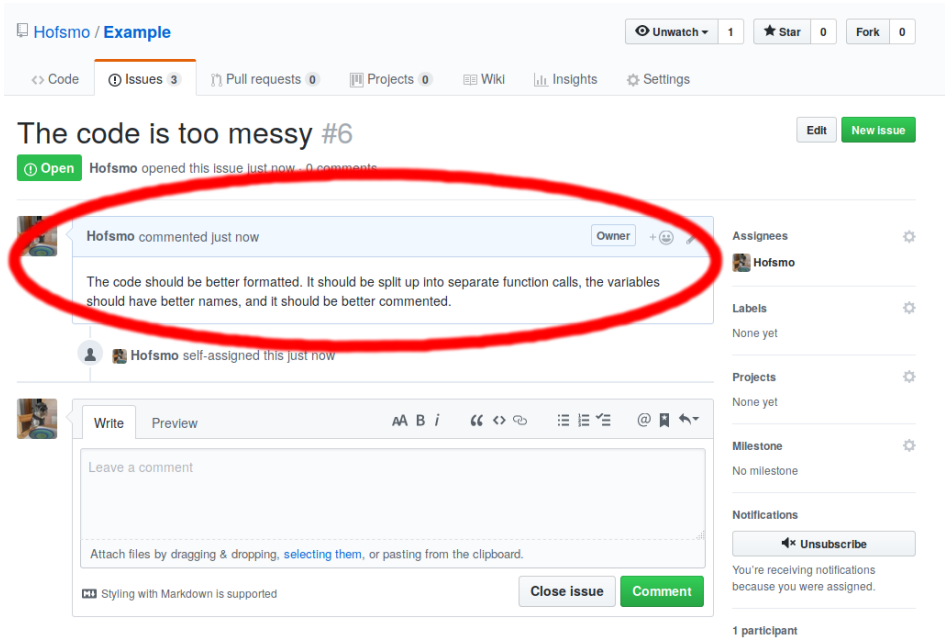
General tips on making code better

- Use descriptive variable names
- Have some space in between lines
- Write comments
- Write docstrings
- Don't copy and paste code

```
import csv
import os
a = "../data"
b = "blood_thirsty_janitors.csv"
with open(os.path.join(a, b)) as cf:
    r = csv.DictReader(cf)
    temp_1 = 0
    temp_2 = 0
    temp_3 = 0
    temp_4 = 0
    for rr in r:
        temp_1 += int(rr["beard length"])
        temp_2 += int(rr["churches burned"])
        temp_3 += int(rr["age"])
        temp_4 += 1
a_b_l = temp_1/temp_4
cbl = temp_2/temp_1
aa = temp_3/temp_4
print("Calculations done")
b = "evil_vikings.csv"
with open(os.path.join(a, b)) as cf:
    r = csv.DictReader(cf)
    temp_1 = 0
    temp_2 = 0
    temp_3 = 0
    temp_4 = 0
    for rr in r:
        temp_1 += int(rr["beard length"])
        temp_2 += int(rr["churches burned"])
        temp_3 += int(rr["age"])
        temp_4 += 1
a_b_l = temp_1/temp_4
cbl = temp_2/temp_1
aa = temp_3/temp_4
print("Calculations done")
b = "hungry_crusaders.csv"
with open(os.path.join(a, b)) as cf:
```

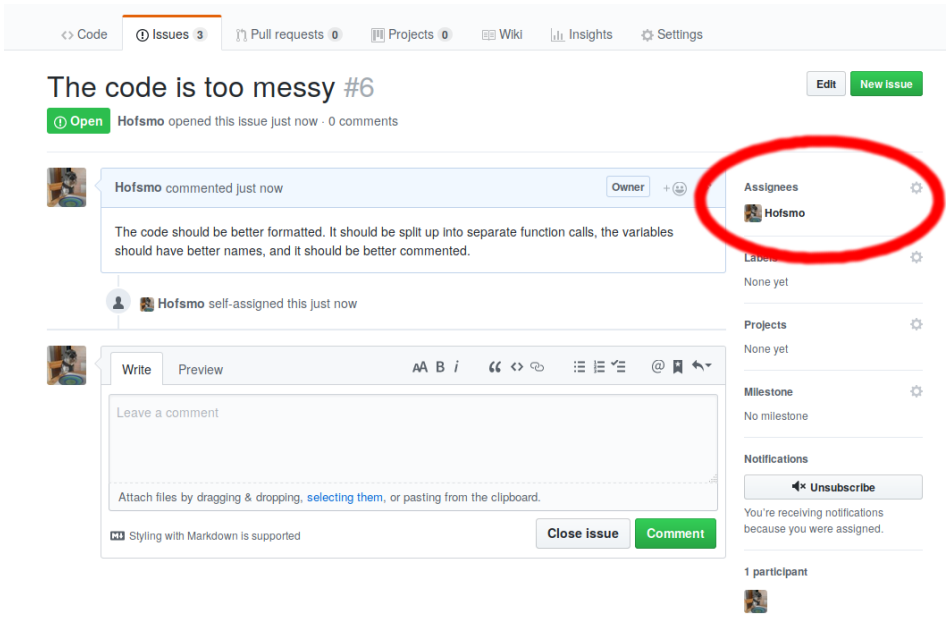
Using the GitHub issue tracker

— Write a descriptive issue



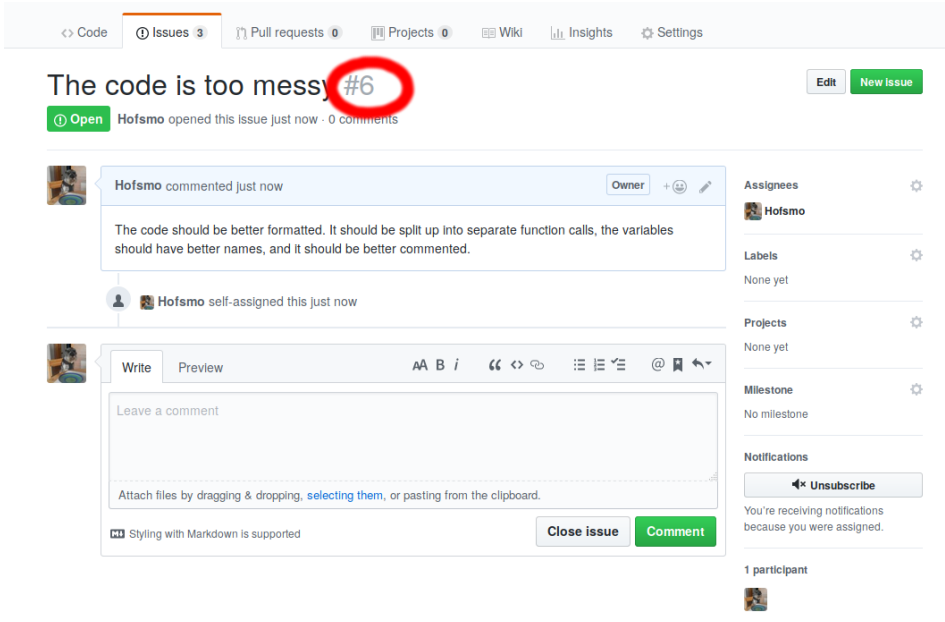
Using the GitHub issue tracker

- Write a descriptive issue
- You can make someone responsible for fixing the issue



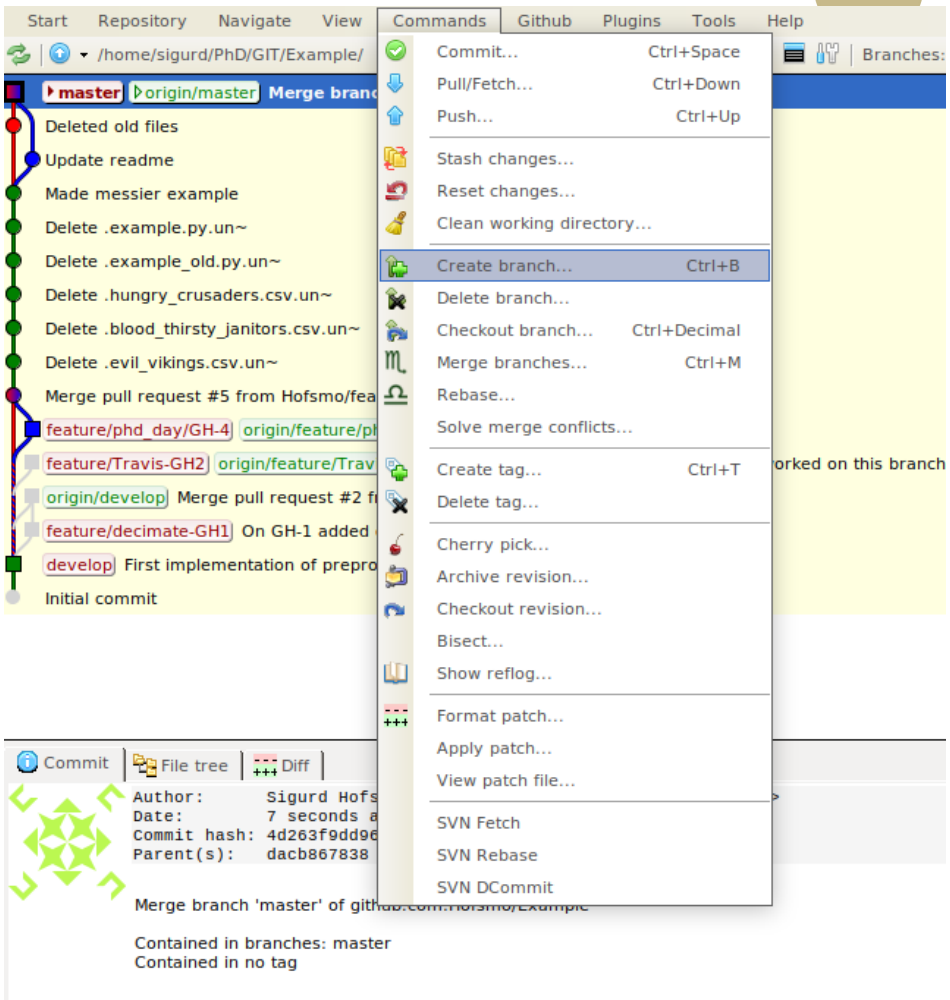
Using the GitHub issue tracker

- Write a descriptive issue
- You can make someone responsible for fixing the issue
- The issue get a number you can use for referencing it



Using git for fixing the code

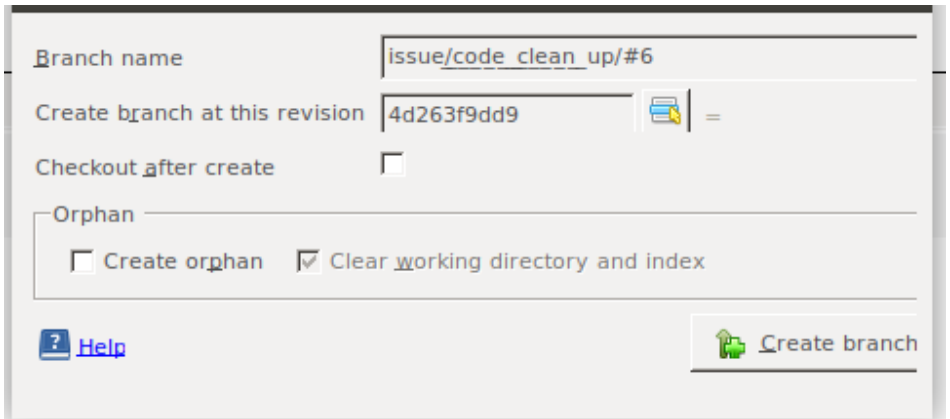
— Create a branch for the issue



Using git for fixing the code

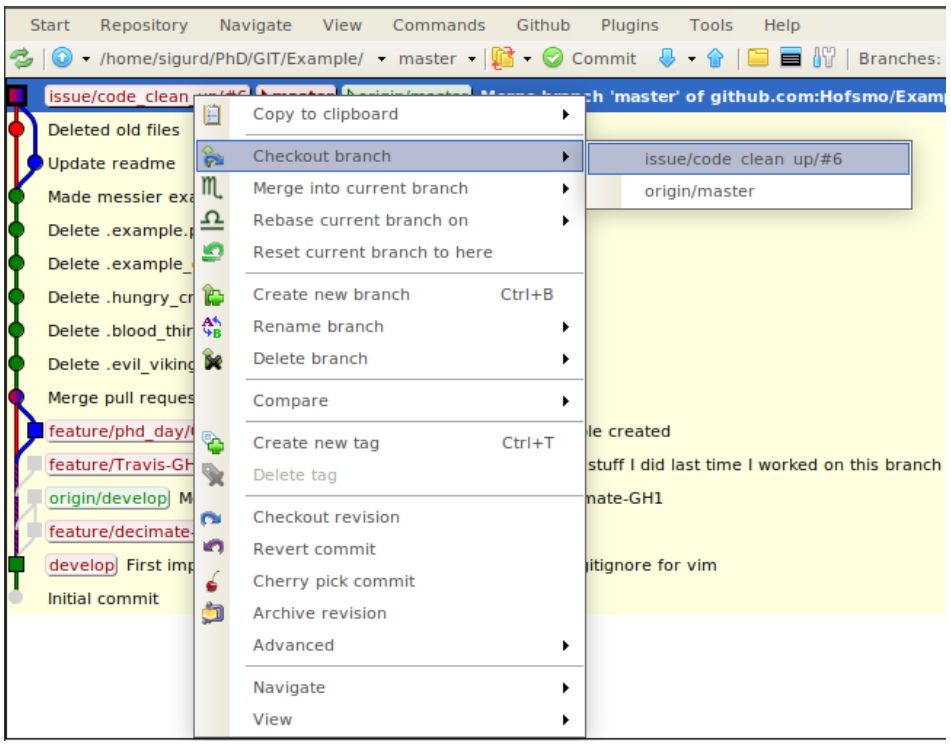
— Create a branch for the issue

— Give the branch a good name



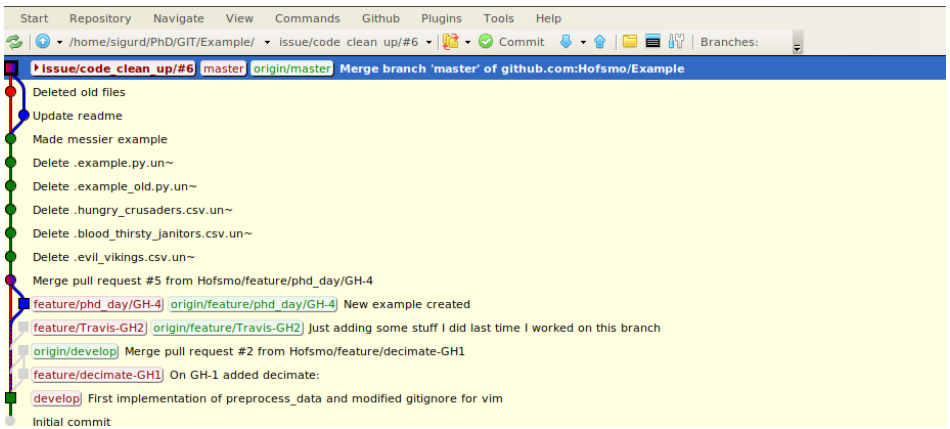
Using git for fixing the code

- Create a branch for the issue
- Give the branch a good name
- Checkout the branch



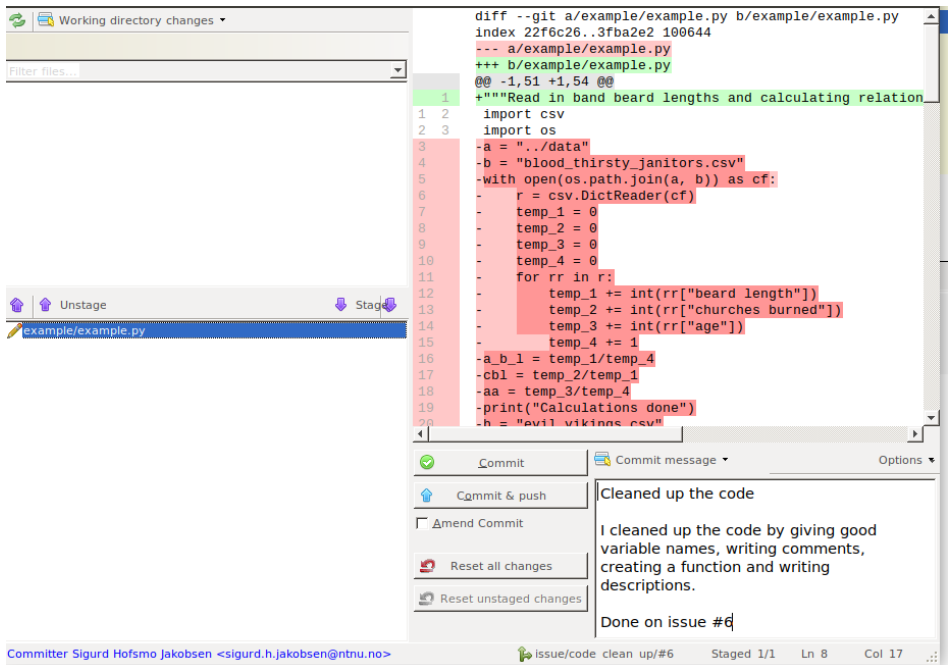
Using git for fixing the code

- Create a branch for the issue
- Give the branch a good name
- Checkout the branch
- We are now ready to clean up the code



Using git for fixing the code

- Create a branch for the issue
- Give the branch a good name
- Checkout the branch
- We are now ready to clean up the code
- After the code is clean reference the issue in the commit message



Using git for fixing the code

- Create a branch for the issue
- Give the branch a good name
- Checkout the branch
- We are now ready to clean up the code
- After the code is clean reference the issue in the commit message
- We can now see the commit message in GitHub

What clean up did I do on the code

— Added a docstring to the file

```
1  """Read in band beard lengths and calculate relation to church burnings."""
1
2  import os
3
4
5  def read_file_and_do_calculations(file_path):
6      """Function for reading band files and do the calculations.
7      INPUT:
8          file_path: The full path to the file
9      Output:
10         nothing
11     """
12
13     # Read the data
14     with open(os.path.join(data_dir, fname)) as csvfile:
15         reader = csv.DictReader(csvfile)
16         total_beard_length = 0
17         total_churches_burned = 0
18         cummulative_age = 0
19         band_members = 0
20         for row in reader:
21             total_beard_length += int(row["beard length"])
22             total_churches_burned += int(row["churches burned"])
23             cummulative_age += int(row["age"])
24             band_members += 1
25
26     # Do the calculations
27     average_beard_length = total_beard_length/band_members
28     churches_per_beard_length = total_churches_burned/total_beard_length
29     average_age = cummulative_age/band_members
30
31     print("Calculations done")
32
33
34 # Define data directory
35 data_dir = "../data"
36
37 # Define data file
38 fname = "blood_thirsty_janitors.csv"
39
40 # Do calculations
41 read_file_and_do_calculations(os.path.join(data_dir, fname))
42
43 # Define data file
44 fname = "evil_vikings.csv"
45
46 # Do calculations
47 read_file_and_do_calculations(os.path.join(data_dir, fname))
48
49 # Define data file
```

What clean up did I do on the code

- Added a docstring to the file
- Created a function

```
"""Read in band beard lengths and calculate relation to church burnings."""
import csv
import os


def read_file_and_do_calculations(file_path):
    """Function for reading band files and do the calculations.
    INPUT:
        file_path: The full path to the file
    Output:
        nothing
    """

    # Read the data
    with open(os.path.join(data_dir, fname)) as csvfile:
        reader = csv.DictReader(csvfile)
        total_beard_length = 0
        total_churches_burned = 0
        cummulative_age = 0
        band_members = 0
        for row in reader:
            total_beard_length += int(row["beard length"])
            total_churches_burned += int(row["churches burned"])
            cummulative_age += int(row["age"])
            band_members += 1

    # Do the calculations
    average_beard_length = total_beard_length/band_members
    churches_per_beard_length = total_churches_burned/total_beard_length
    average_age = cummulative_age/band_members

    print("Calculations done")


# Define data directory
data_dir = "../data"

# Define data file
fname = "blood_thirsty_janitors.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))

# Define data file
fname = "evil_vikings.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))
# Define data file
```

What clean up did I do on the code

- Added a docstring to the file
- Created a function
- Wrote a docstring for the function

```
"""Read in band beard lengths and calculate relation to church burnings."""
import csv
import os


def read_file_and_do_calculations(file_path):
    """Function for reading band files and do the calculations.
    INPUT:
        file_path: The full path to the file
    Output:
        nothing
    """

    # Read the data
    with open(os.path.join(data_dir, fname)) as csvfile:
        reader = csv.DictReader(csvfile)
        total_beard_length = 0
        total_churches_burned = 0
        cummulative_age = 0
        band_members = 0
        for row in reader:
            total_beard_length += int(row["beard length"])
            total_churches_burned += int(row["churches burned"])
            cummulative_age += int(row["age"])
            band_members += 1

    # Do the calculations
    average_beard_length = total_beard_length/band_members
    churches_per_beard_length = total_churches_burned/total_beard_length
    average_age = cummulative_age/band_members

    print("Calculations done")


# Define data directory
data_dir = "../data"

# Define data file
fname = "blood_thirsty_janitors.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))

# Define data file
fname = "evil_vikings.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))
# Define data file
```

What clean up did I do on the code

- Added a docstring to the file
- Created a function
- Wrote a docstring for the function
- Added proper spacing

```
"""Read in band beard lengths and calculate relation to church burnings."""
import csv
import os


def read_file_and_do_calculations(file_path):
    """Function for reading band files and do the calculations.
    INPUT:
        file_path: The full path to the file
    Output:
        nothing
    """

    # Read the data
    with open(os.path.join(data_dir, fname)) as csvfile:
        reader = csv.DictReader(csvfile)
        total_beard_length = 0
        total_churches_burned = 0
        cummulative_age = 0
        band_members = 0
        for row in reader:
            total_beard_length += int(row["beard length"])
            total_churches_burned += int(row["churches burned"])
            cummulative_age += int(row["age"])
            band_members += 1

    # Do the calculations
    average_beard_length = total_beard_length/band_members
    churches_per_beard_length = total_churches_burned/total_beard_length
    average_age = cummulative_age/band_members

    print("Calculations done")


# Define data directory
data_dir = "../data"

# Define data file
fname = "blood_thirsty_janitors.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))

# Define data file
fname = "evil_vikings.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))
# Define data file
```

What clean up did I do on the code

- Added a docstring to the file
- Created a function
- Wrote a docstring for the function
- Added proper spacing
- Added comments

```
"""Read in band beard lengths and calculate relation to church burnings."""
import csv
import os


def read_file_and_do_calculations(file_path):
    """Function for reading band files and do the calculations.
    INPUT:
        file_path: The full path to the file
    Output:
        nothing
    """

    # Read the data
    with open(os.path.join(data_dir, fname)) as csvfile:
        reader = csv.DictReader(csvfile)
        total_beard_length = 0
        total_churches_burned = 0
        cummulative_age = 0
        band_members = 0
        for row in reader:
            total_beard_length += int(row["beard length"])
            total_churches_burned += int(row["churches burned"])
            cummulative_age += int(row["age"])
            band_members += 1

    # Do the calculations
    average_beard_length = total_beard_length/band_members
    churches_per_beard_length = total_churches_burned/total_beard_length
    average_age = cummulative_age/band_members

    print("Calculations done")


# Define data directory
data_dir = "../data"

# Define data file
fname = "blood_thirsty_janitors.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))

# Define data file
fname = "evil_vikings.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))
# Define data file
```

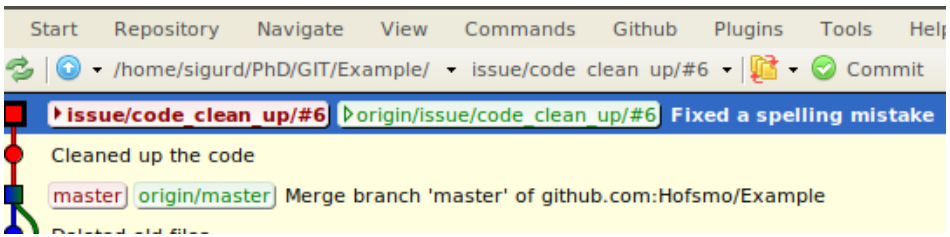
What clean up did I do on the code

- Added a docstring to the file
- Created a function
- Wrote a docstring for the function
- Added proper spacing
- Added comments
- Made reasonable variable names

```
"""Read in band beard lengths and calculate relation to church burnings."""
import csv
import os


def read_file_and_do_calculations(file_path):
    """Function for reading band files and do the calculations.
    INPUT:
        file_path: The full path to the file
    Output:
        nothing
    """

    # Read the data
    with open(os.path.join(data_dir, fname)) as csvfile:
        reader = csv.DictReader(csvfile)
        total_beard_length = 0
        total_churches_burned = 0
        cummulative_age = 0
        band_members = 0
        for row in reader:
            total_beard_length += int(row["beard length"])
            total_churches_burned += int(row["churches burned"])
            cummulative_age += int(row["age"])
            band_members += 1

    # Do the calculations
    average_beard_length = total_beard_length/band_members
    churches_per_beard_length = total_churches_burned/total_beard_length
    average_age = cummulative_age/band_members

    print("Calculations done")


# Define data directory
data_dir = "../data"

# Define data file
fname = "blood_thirsty_janitors.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))

# Define data file
fname = "evil_vikings.csv"

# Do calculations
read_file_and_do_calculations(os.path.join(data_dir, fname))
# Define data file
```

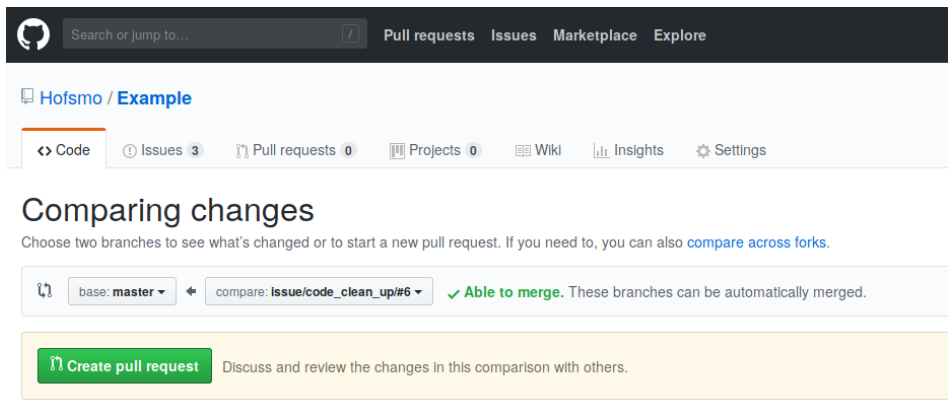
Resolving the issue

— The code is now clean, and we want the changes in our master branch.



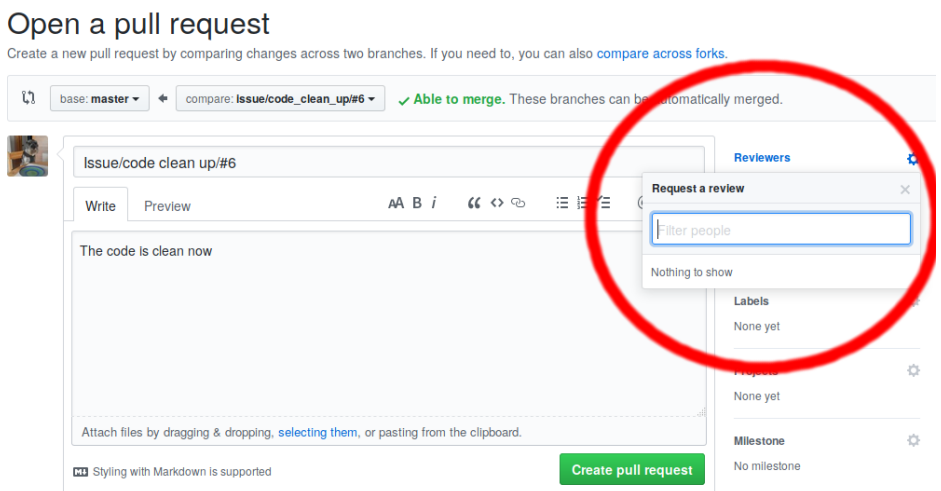
Resolving the issue

- The code is now clean, and we want the changes in our master branch.
- GitHub automatically checks if the changes are compatible.



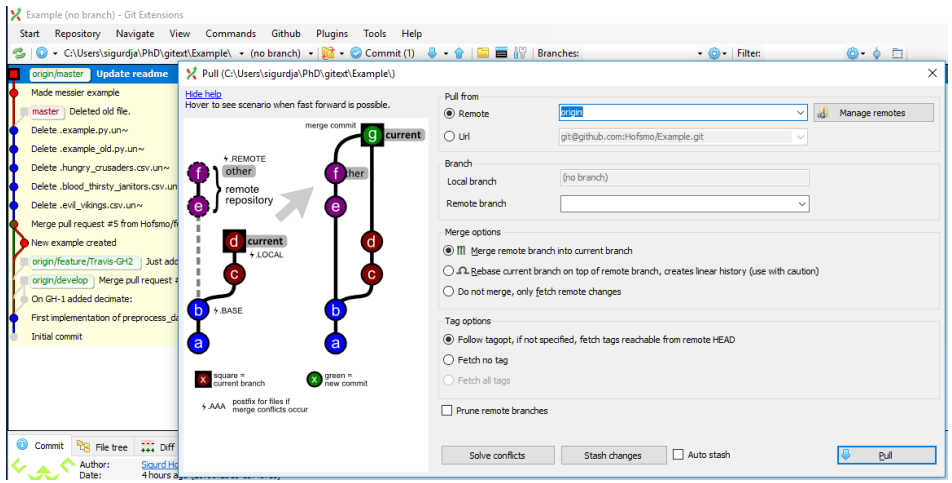
Resolving the issue

- The code is now clean, and we want the changes in our master branch.
- GitHub automatically checks if the changes are compatible.
- You can ask someone to review your code.



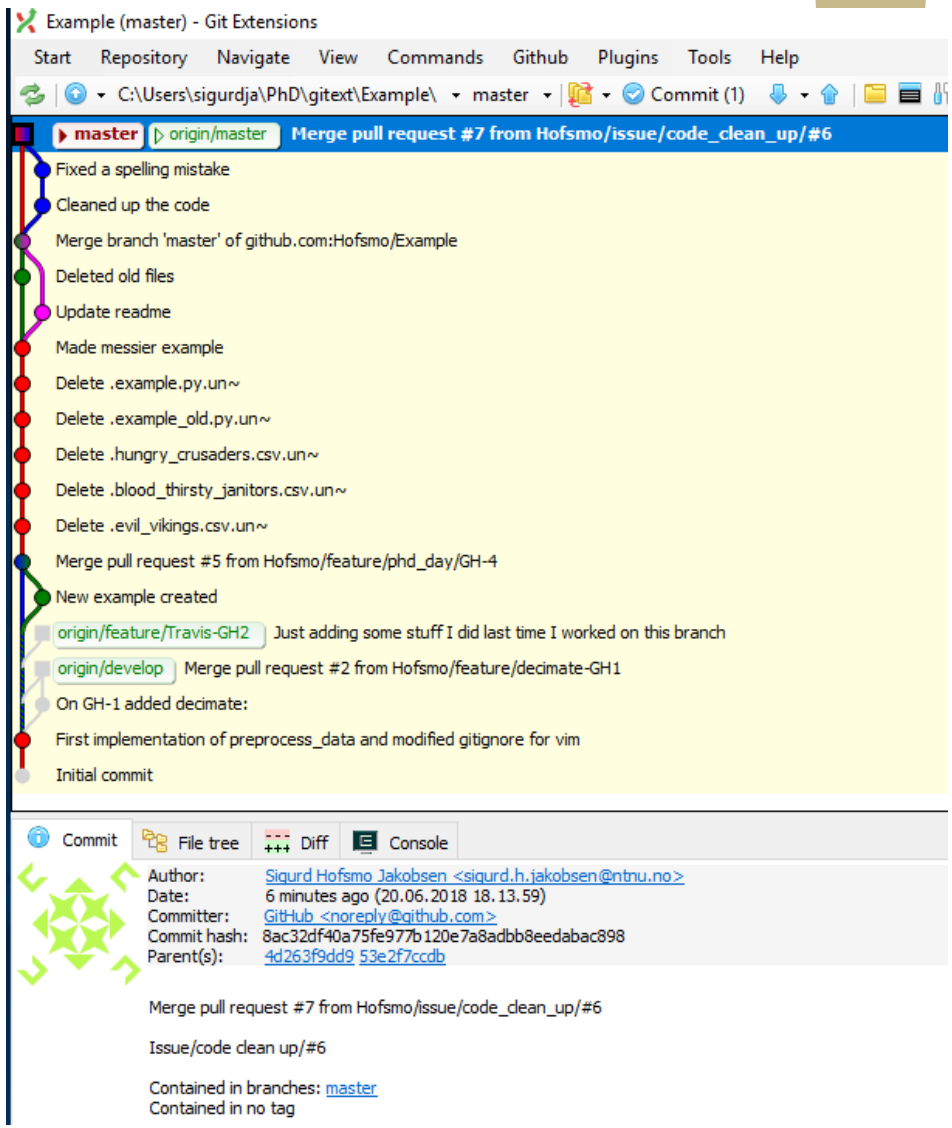
Another student downloading the changes

— Other students can now get the changes using pull



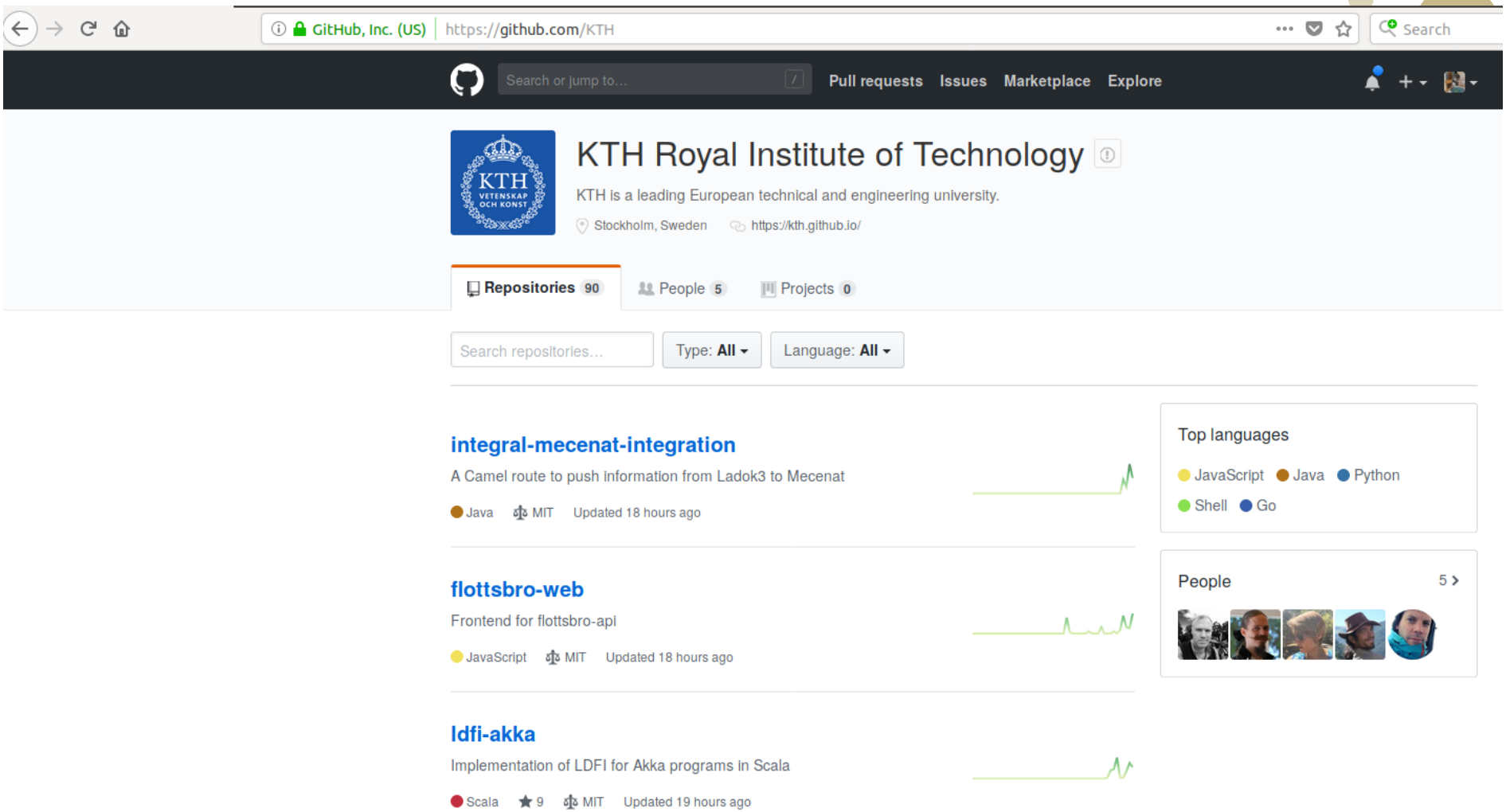
Another student downloading the changes

- Other students can now get the changes using pull
- Afterwards the changes will be merged and available

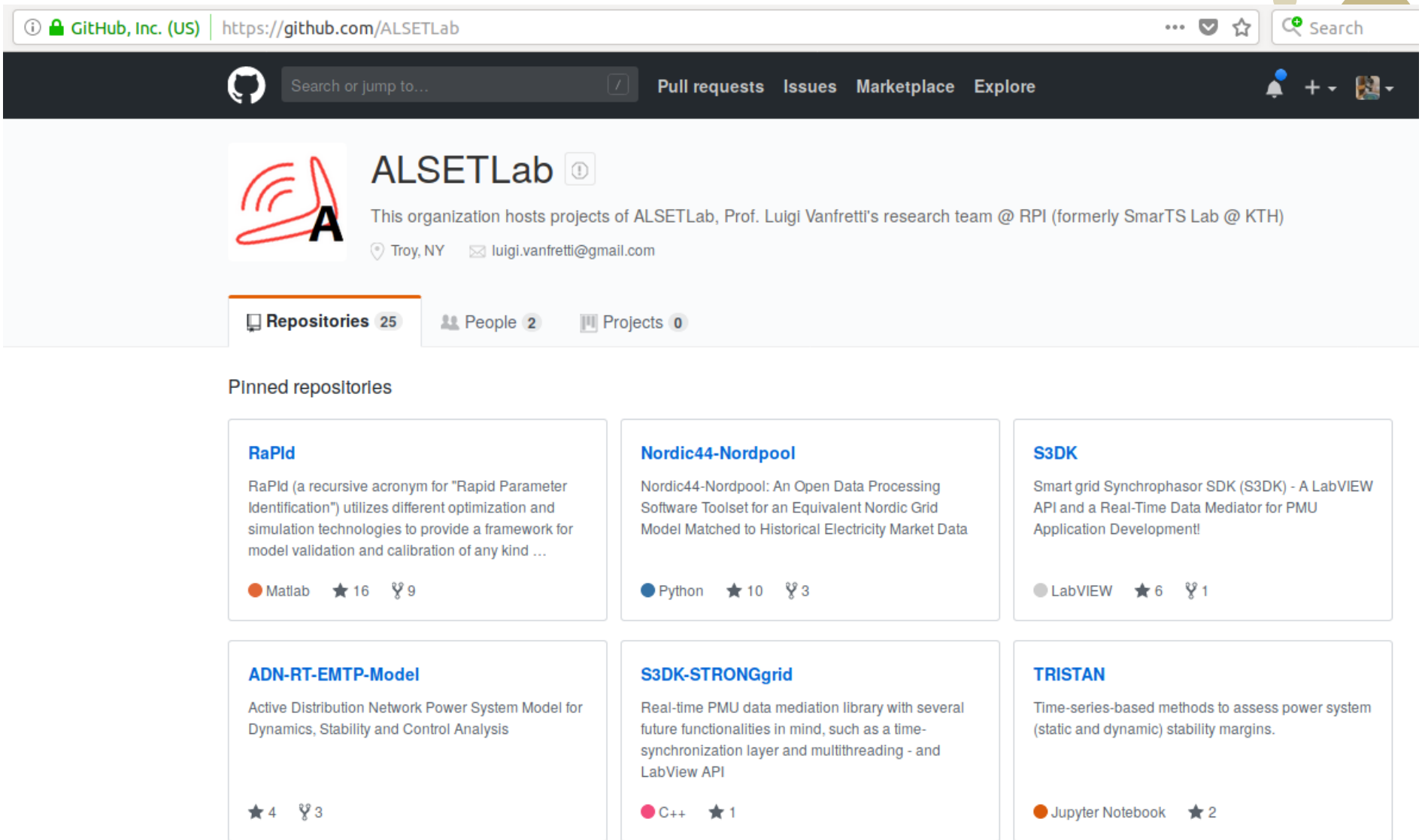


Outline

GitHub provides organizational pages

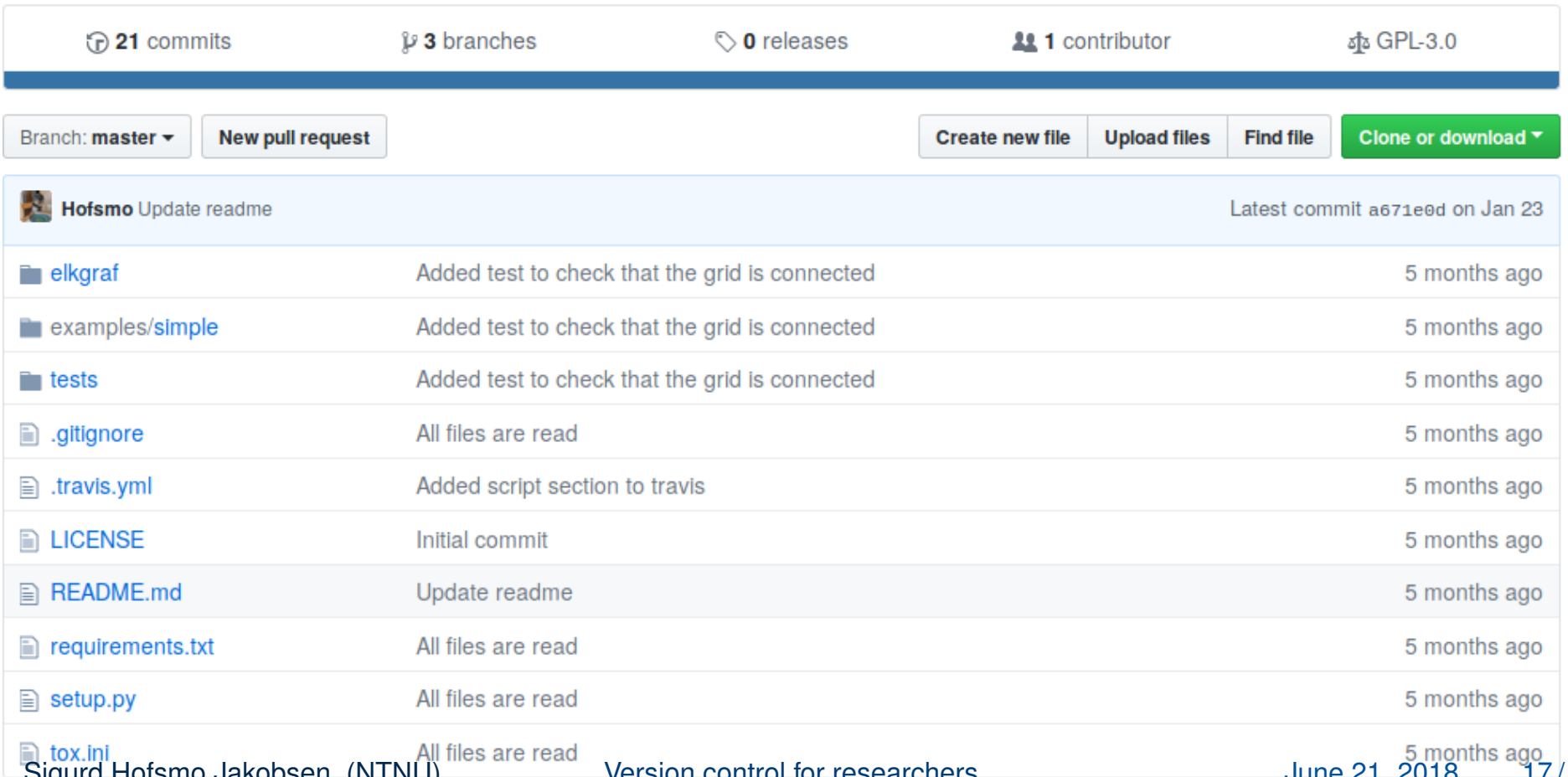


GitHub provides organizational pages



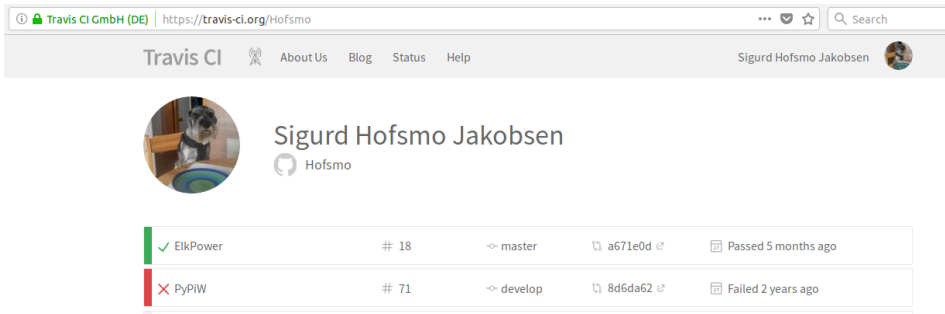
Structuring of packages and toolboxes

- Properly structured packages can easily be installed by others.
- For Python check out setuptools
- For Matlab check out their help pages



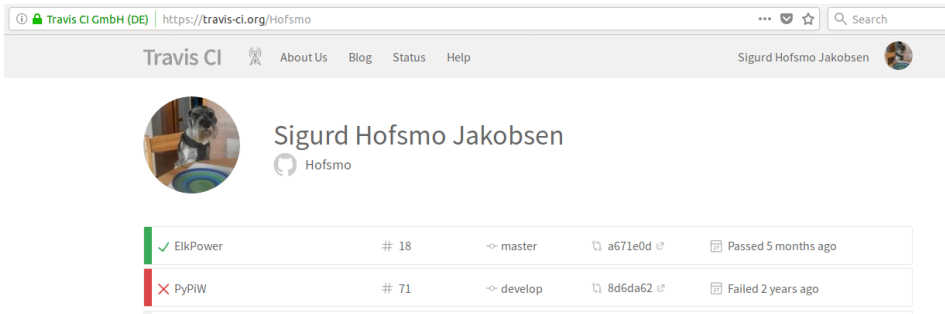
Automating code sanity tests

- Code can be tested for guidelines (PEP8)
- The code installation can be tested (tox)
- Unit tests can be done (pytest)
- Documentation can be generated and tested (sphinx)
- Many more options



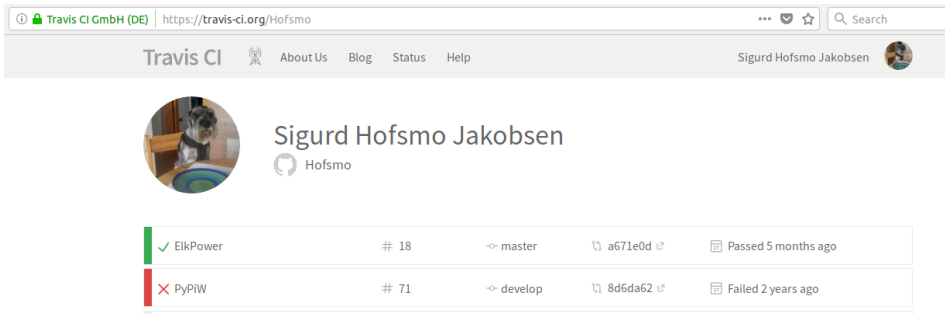
Automating code sanity tests

- Code can be tested for guidelines (PEP8)
- The code installation can be tested (tox)
- Unit tests can be done (pytest)
- Documentation can be generated and tested (sphinx)
- Many more options



Automating code sanity tests

- Code can be tested for guidelines (PEP8)
- The code installation can be tested (tox)
- Unit tests can be done (pytest)
- Documentation can be generated and tested (sphinx)
- Many more options



Automating code sanity tests

- Code can be tested for guidelines (PEP8)
- The code installation can be tested (tox)
- Unit tests can be done (pytest)
- Documentation can be generated and tested (sphinx)
- Many more options

Automating code sanity tests

- Code can be tested for guidelines (PEP8)
- The code installation can be tested (tox)
- Unit tests can be done (pytest)
- Documentation can be generated and tested (sphinx)
- Many more options

Outline

Questions for discussion

- How does my code look?
- How do we work with code today? Cooperation? Version control?
- Could my code be useful for others?
- Should the department have a more active approach to the topic?
  - Education?
  - Infrastructure?
  - Who has the responsibility?
- Actions?