Introduction to GitHub

Python-do-ECARES
Fabrizio Leone

Introduction

Git and GitHub

- Git is a version control system, i.e. a way to keep track of the whole history of things you do on a file. It is useful to save, manage and edit all the different versions of your project.

- GitHub is a web service that allows to conveniently work with Git. It allows you to create your own directories, see projects of other people and collaborate with them.

- GitHub offers four different subscription plans. We will work with a free one, which means that *everybody* can see what we do. However, with an academic account, you can also create private repositories.

GitHub: What It Does And Does Not Do

- No matter which subscription plan you choose, GitHub offers very limited storage space (you cannot upload files $>$ 100MB). Therefore, it is **not** suitable for storing large files (e.g. datasets). **GitHub is not a substitute for a cloud**.
- GitHub is a platform where to upload mostly **source files** (e.g. .tex, .txt, .m, .R, .do, .py, .doc,...) and light pdf.
- You can read more about Git and GitHub here and here.

GitHub Desktop

- In this course, we interact with GitHub mostly through the GitHub Desktop application.
- GitHub Desktop provides a simple yet powerful desktop interface to GitHub.
- You can download GitHub desktop here.
- You can also interact with GitHub using the Terminal (not covered in this class).

Realistic Workflow

Suppose you are starting out your new project. If you use GitHub, you can

1. Create a local folder with your favourite sub-folders (e.g. code, slides, paper, literature, data,...).
2. Publish some sub-folders on GitHub (e.g. code, slides, paper).
3. Work on your code/paper/slides locally and then save different versions of them on GitHub. (No more: paper_v1, paper_v1.A, paper_v2.89.x7%,...).
4. Scroll through different versions of your files when you need. Collaborate with other people. (No more: paper_v1_myedition, paper_v1_youredition,...).

First steps

Essential Vocabulary

GitHub is built upon a few simple concepts

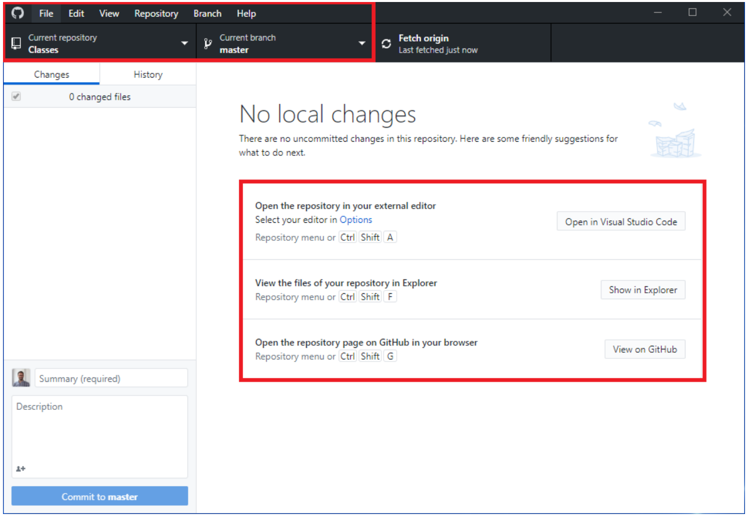
- **Repository** are the containers of your project. They can include folders, files, images, etc. Each repository is made of one or more branches.
- The default **branch** of your repository is called *master*. One can copy the master into other branches. In this way, you can work on multiple versions of the same file in parallel.
- **Commits** are changes you make to branches.

First Steps With GitHub Desktop

- You should already have (1) opened a GitHub account and (2) downloaded GitHub Desktop.
- If not, please do it now.

Desktop Interface

GitHub Desktop looks like this



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A

View the files of your repository in Explorer
Repository menu or Ctrl Shift F

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G

Desktop Interface

Take a few moments to familiarise with GitHub Desktop. It lets you

- Manage existing repositories, create versions, see history of changes, revert changes (this class).
- Create and manage new repositories (homework).
- Collaborate with other people (try it yourself).

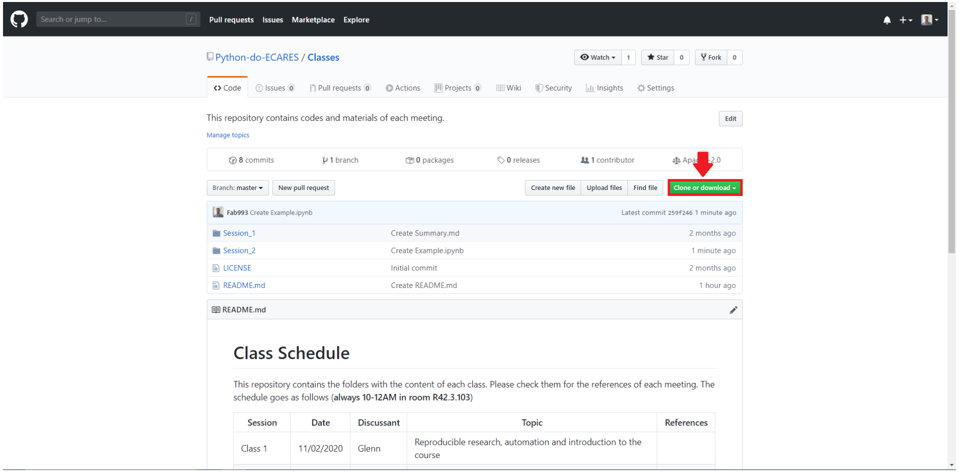
This Course

First Steps With GitHub Desktop

- This section shows how to clone a repository, create your own branch and how to commit and push changes to it.
- Make sure to be familiar with each step. You will do this many times in the future.

Step 1. Clone Repository

Go to the Classes repository on GitHub and click on the "Clone or download" button.



Python-do-ECARES / Classes

This repository contains codes and materials of each meeting.

Fab993 Create Example.ipynb — Latest commit 259f246 1 minute ago

Session_1	Create Summary.md	2 months ago
Session_2	Create Example.ipynb	1 minute ago
LICENSE	Initial commit	2 months ago
README.md	Create README.md	1 hour ago

README.md

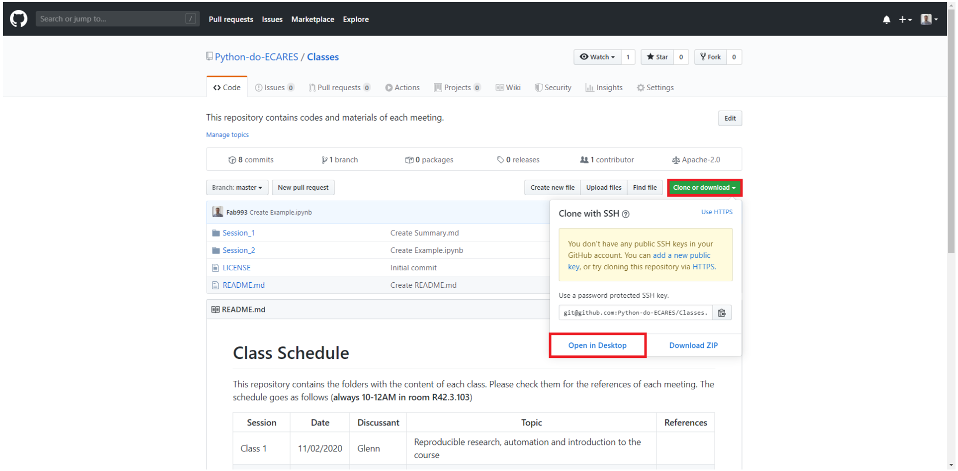
Class Schedule

This repository contains the folders with the content of each class. Please check them for the references of each meeting. The schedule goes as follows (**always 10-12AM in room R42.3.103**)

Session	Date	Discussant	Topic	References
Class 1	11/02/2020	Glenn	Reproducible research, automation and introduction to the course	

Step 1.A. Open Repository

Choose ”Open in Desktop”.



Python-do-ECARES / Classes

This repository contains codes and materials of each meeting.

Fab993 Create Example.ipynb

Session_1	Create Summary.md
Session_2	Create Example.ipynb
LICENSE	Initial commit
README.md	Create README.md

You don't have any public SSH keys in your GitHub account. You can add a new public key, or try cloning this repository via HTTPS.

Use a password protected SSH key.

```
git@github.com:Python-do-ECARES/Classes.
```

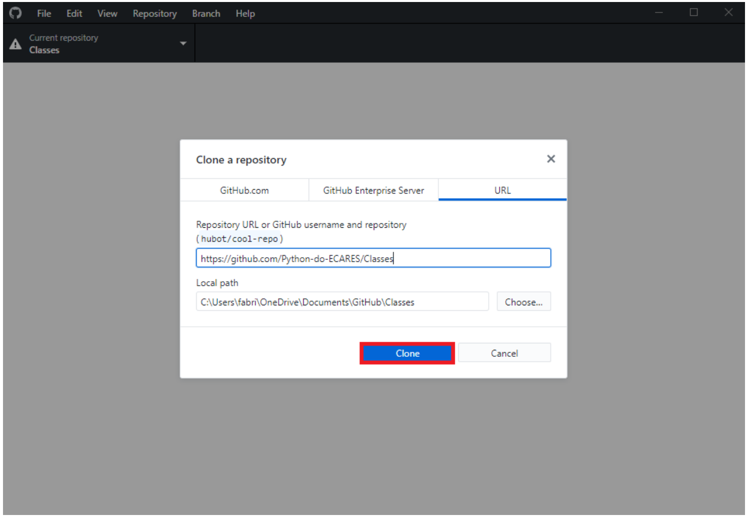
Class Schedule

This repository contains the folders with the content of each class. Please check them for the references of each meeting. The schedule goes as follows (**always 10-12AM in room R42.3.103**)

Session	Date	Discussant	Topic	References
Class 1	11/02/2020	Glenn	Reproducible research, automation and introduction to the course	

Step 1.B. Clone Repository

Check the local path where to clone the repository and hit ”Clone”.



Clone a repository

Repository URL or GitHub username and repository
(`hubot/cool-repo`)

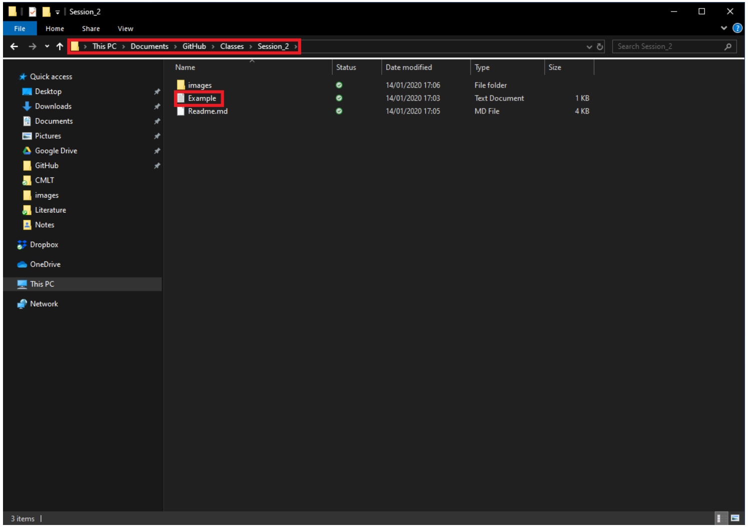
https://github.com/Python-do-ECARES/Classes

Local path

C:\Users\fabri\OneDrive\Documents\GitHub\Classes

Step 1.C. Open Local Directory

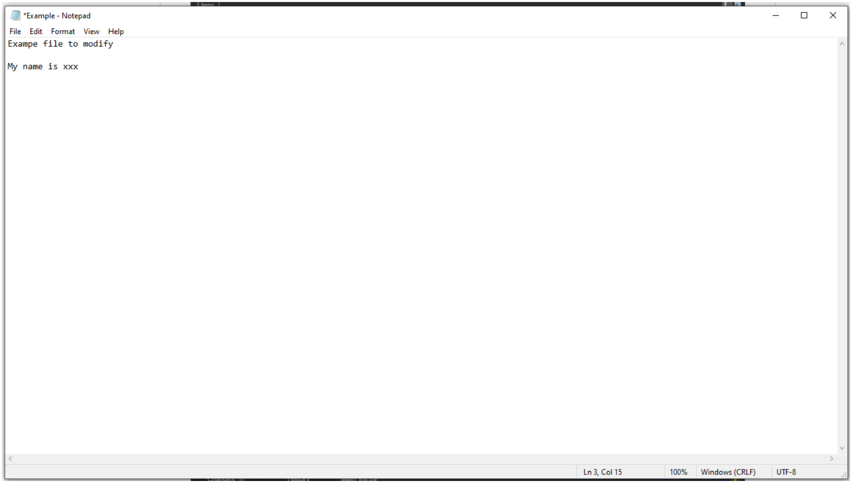
Under the chosen directory, you will see the following files.



Name	Status	Date modified	Type	Size
images	✓	14/01/2020 17:06	File folder	
Example	✓	14/01/2020 17:03	Text Document	1 KB
Readme.md	✓	14/01/2020 17:05	MD File	4 KB

Step 2. Make Changes

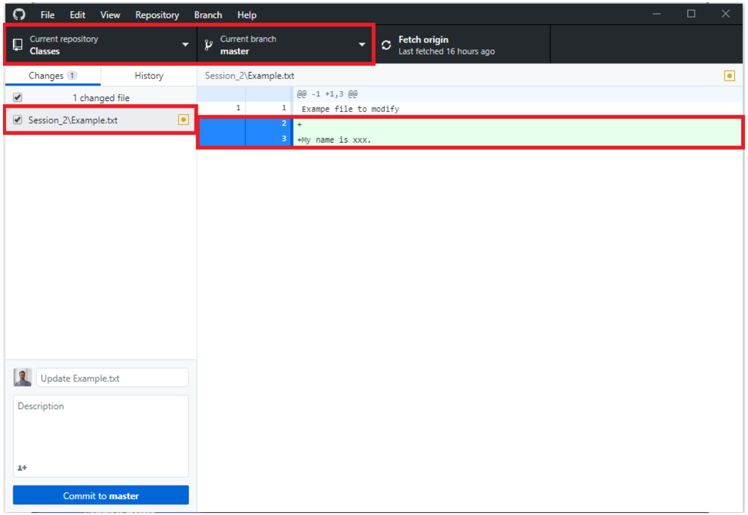
Open *Example.txt*. Add a comment line with your name.



```
Exampe file to modify

My name is xxx
```

Step 2.A. Save Changes

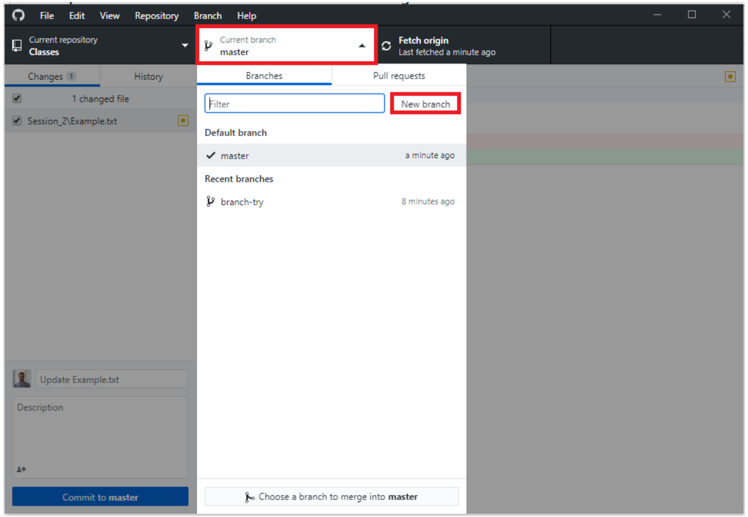
Save the file. Changes will be displayed in GitHub Desktop as follows.



```
@@ -1 +1,3 @@
 Exampe file to modify
+
+My name is xxx.
```

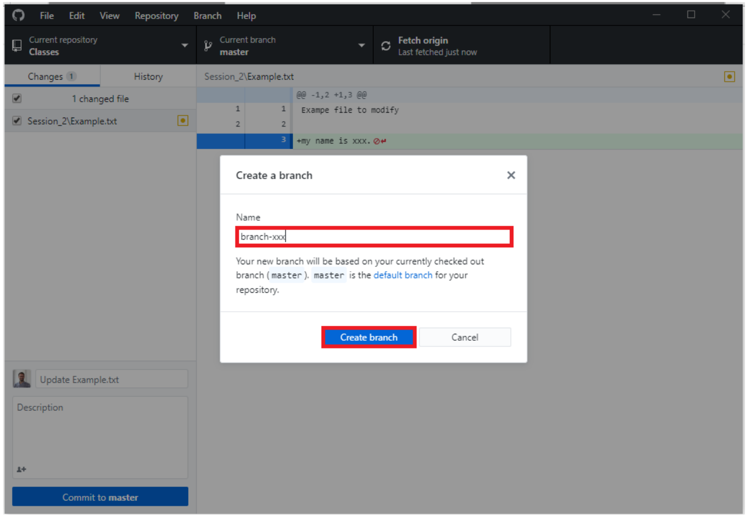
Step 2.B. Create Your Own Branch And Commit Changes

Select ”Master” and hit ”New Branch”.



Step 2.C. Name Branch

Give the new branch your name. Then click ”Create Branch”.



```
@@ -1,2 +1,3 @@
Exampe file to modify

+my name is xxx.
```

Create a branch

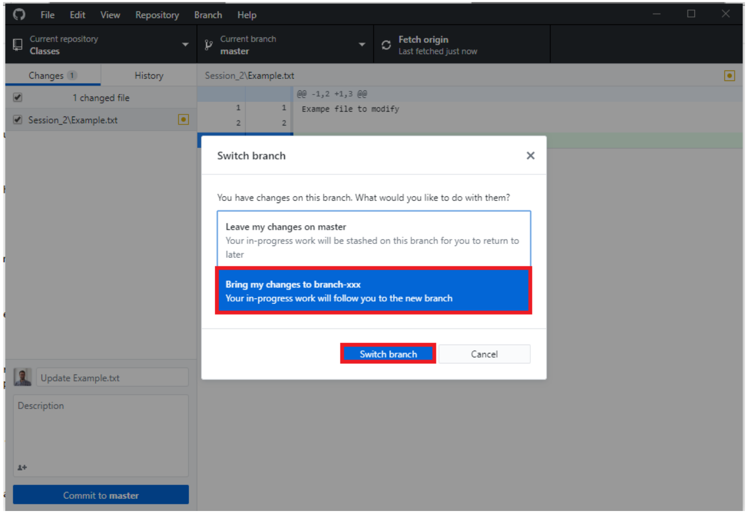
Name

branch-xxx

Your new branch will be based on your currently checked out branch (`master`). `master` is the default branch for your repository.

Step 2.D. Switch Branch

Switch changes to your own branch.



```
@@ -1,2 +1,3 @@
Exampe file to modify
```

Switch branch

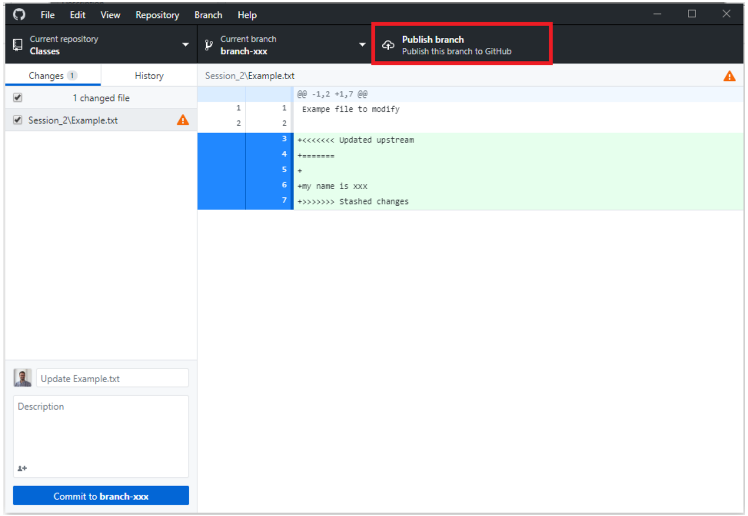
You have changes on this branch. What would you like to do with them?

Leave my changes on master
Your in-progress work will be stashed on this branch for you to return to later

Bring my changes to branch-xxx
Your in-progress work will follow you to the new branch

Step 2.E. Publish Branch

Publish your branch online.

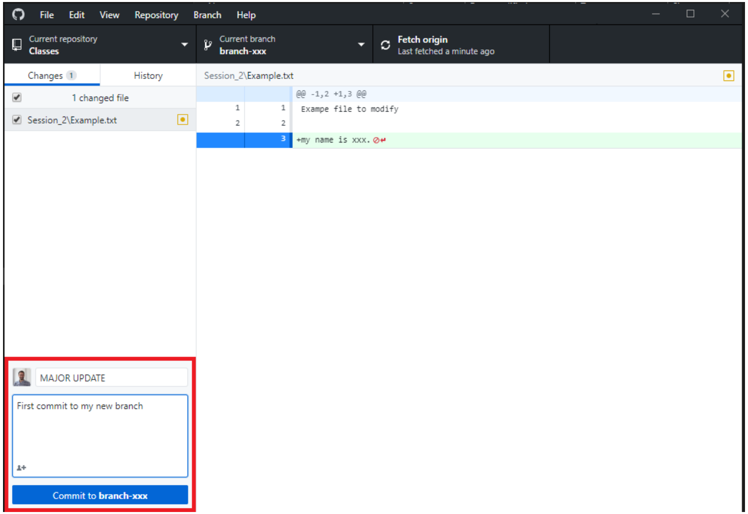


Session_2\Example.txt

```
@@ -1,2 +1,7 @@
 Exampe file to modify

+<<<<<<< Updated upstream
+=======
+
+my name is xxx
+>>>>>>> Stashed changes
```

Step 2.F. Commit to Branch

Give Description and summary. Then commit.



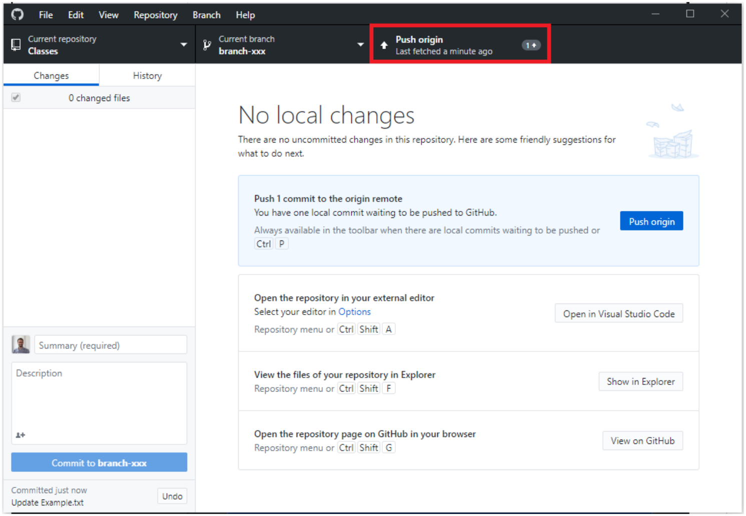
```
@@ -1,2 +1,3 @@
1 1  Exampe file to modify
2 2
  3 +my name is xxx.
```

MAJOR UPDATE

First commit to my new branch

Step 2.G. Push to Branch

Push changes to your own branch.



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Push 1 commit to the origin remote
You have one local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or Ctrl P

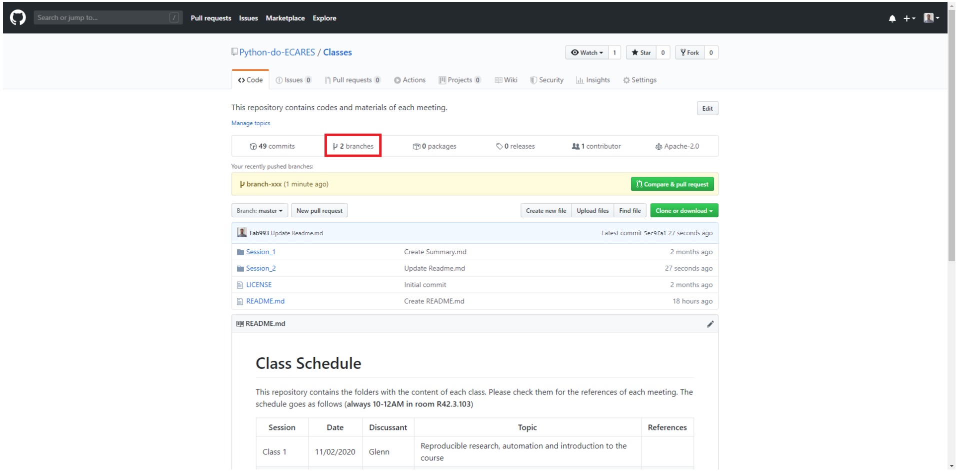
Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A

View the files of your repository in Explorer
Repository menu or Ctrl Shift F

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G

Step 3. Next Steps

Go to "Classes" page. Notice that now there are 2 branches. Click on "branches".



This repository contains codes and materials of each meeting.

Your recently pushed branches:

branch-xxx (1 minute ago)

Fab993 Update Readme.md — Latest commit 5ec9fa1 27 seconds ago

Session_1	Create Summary.md	2 months ago
Session_2	Update Readme.md	27 seconds ago
LICENSE	Initial commit	2 months ago
README.md	Create README.md	18 hours ago

README.md

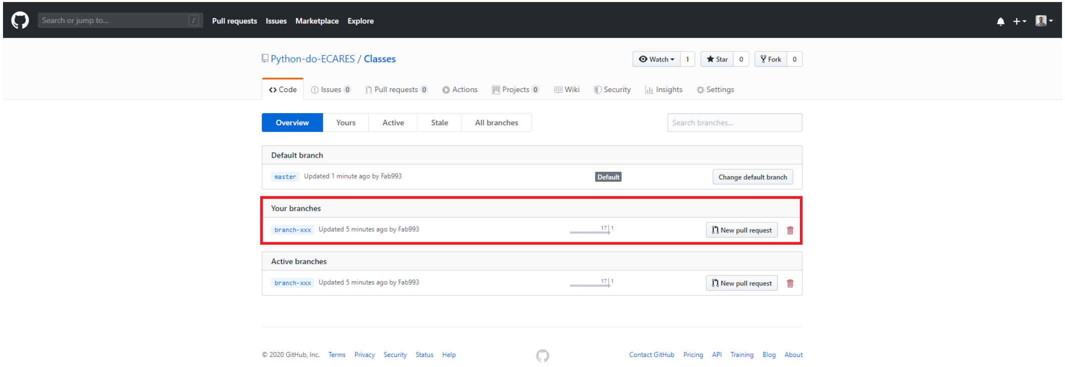
Class Schedule

This repository contains the folders with the content of each class. Please check them for the references of each meeting. The schedule goes as follows (**always 10-12AM in room R42.3.103**)

Session	Date	Discussant	Topic	References
Class 1	11/02/2020	Glenn	Reproducible research, automation and introduction to the course	

Step 3.A. Done!

Your directory will now appear below "branches" as follows.



Default branch

master Updated 1 minute ago by Fab993 Default Change default branch

Your branches

branch-xxx Updated 5 minutes ago by Fab993 17 | 1 New pull request

Active branches

branch-xxx Updated 5 minutes ago by Fab993 17 | 1 New pull request

© 2020 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

Taking Stocks

- You are now able to create and maintain **your own branch** for each session of this course.
- **You are not allowed to push to the master. If you try, you will get an error.** This is a useful feature if you manage a project and do not want collaborators to modify the master directly.
- Please make sure to **only push changes to your own branch** during this course. You are encouraged to collaborate and see what other people is up to, but **never** commit changes directly to other people's branches.
- Good news is that you can always revert changes back if you do so by mistake. This is why GitHub is so useful!

Moving Forward

- You do not need to clone this repository again in the future.
- If you are to present, please send your material to Glenn, Federico or me. We will make sure to upload it to the master.

Standards

Commit and Push

- **Committing** and **pushing** are the main two words you have to familiarize with.
- *Committing changes* to a branch, means that you are creating a new version of your file.
- *Pushing changes* means, instead, that you are publishing them online on GitHub. Think of the pushing action as a way of creating different stable releases of your code.
- Only push changes online if you have made a stable change.

Commit and Local Saving

- Notice that **committing to GitHub** and **saving your file locally** are very different things.
- If you save locally, you only change the file on your device. If you commit, you add a “node” to the chain of your versions.
- With this respect, we recommend to commit changes regularly, but also to use **standards**.

Commit Standards

- We encourage you to adopt the following standards to commit tidily.
- “Summary” should be either **Minor Change**, **Major Change** or **Bug Fixes**. The first should indicate small changes in syntaxis or general improvements. The second to major modifications (e.g. add new section or function), while the third is to notify that you have fixed some bug.
- **Description** should briefly explain what the summary refers to.

Commit Standards

- Suppose you **create a new function for data cleaning in your code**. When pushing this change to GitHub, you want to give **Major Change** as summary and ”added function for data cleaning” as description.
- A tidy commit activities will create a full history of changes in GitHub that you can scroll through to check different versions of your code.
- Finally, it will also help other people to understand your work.

Browse History

Browse Through History

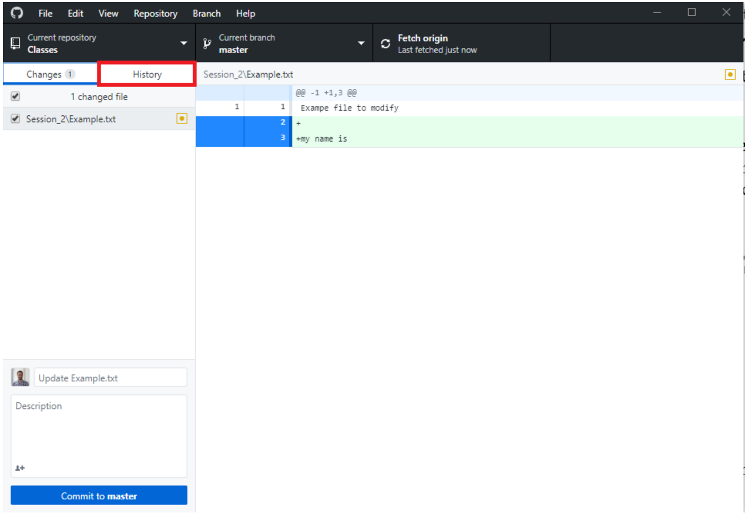
With GitHub Desktop, you can also browse through the entire history of your commits. This is very helpful to

- **Revert back changes**. Imagine you update a code but, at some point, you realise that one of the previous releases was better.
- **Compare different versions**. From time to time, you may want to look back at previous versions of your work (slide, paper, code,...).

A tidy committing activity will help you to easily browse through meaningful versions of your work. If you commit to often, you have to search a lot (versions only differ marginally from one another). If you commit too infrequently, you may lose information.

Step1. Revert Back Changes

Open GitHub Desktop at the current repository. Then hit "History".

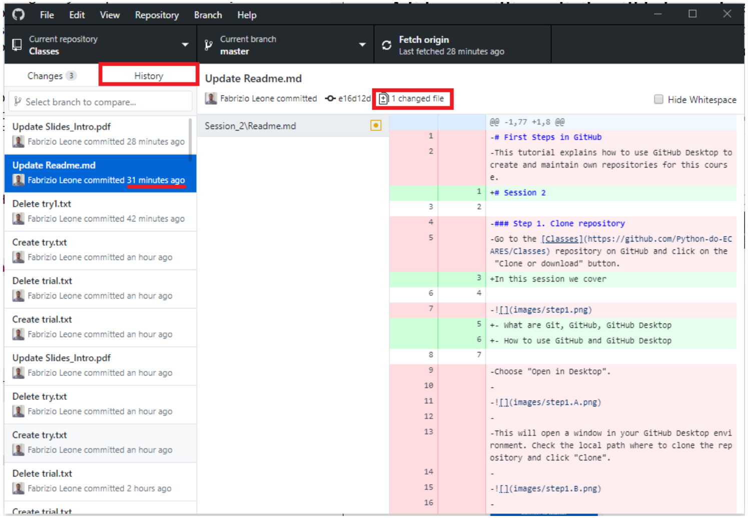


Session_2\Example.txt

```
@@ -1 +1,3 @@
 Exampe file to modify
+
+my name is
```

Step2. Revert Back Changes

You can scroll through the whole history of changes you have made up to that moment.



File Edit View Repository Branch Help
Current repository Classes
Current branch master
Fetch origin Last fetched 28 minutes ago
Changes 3
History
Select branch to compare...
Hide Whitespace

Update Slides_Intro.pdf — Fabrizio Leone committed 28 minutes ago
Update Readme.md — Fabrizio Leone committed 31 minutes ago
Delete try1.txt — Fabrizio Leone committed 42 minutes ago
Create try.txt — Fabrizio Leone committed an hour ago
Delete trial.txt — Fabrizio Leone committed an hour ago
Create trial.txt — Fabrizio Leone committed an hour ago
Update Slides_Intro.pdf — Fabrizio Leone committed an hour ago
Delete try.txt — Fabrizio Leone committed an hour ago
Create try.txt — Fabrizio Leone committed an hour ago
Delete trial.txt — Fabrizio Leone committed 2 hours ago
Create trial.txt

Update Readme.md

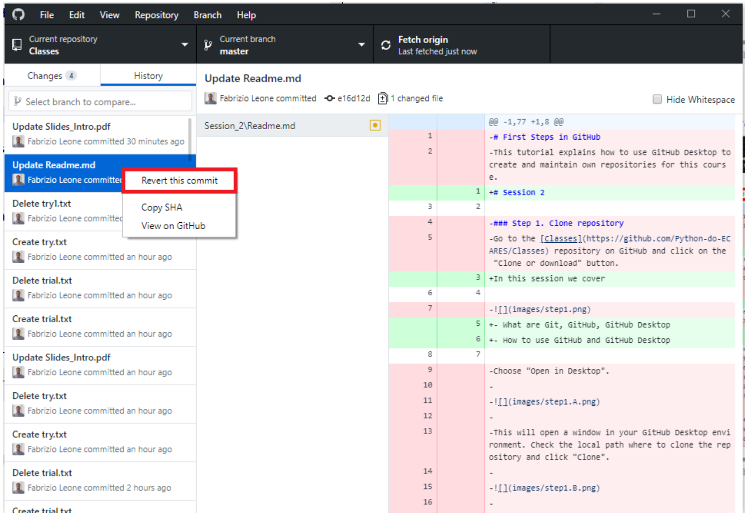
Fabrizio Leone committed e16d12d 1 changed file

Session_2\Readme.md

```
@@ -1,77 +1,8 @@
-# First Steps in GitHub
-This tutorial explains how to use GitHub Desktop to create and maintain own repositories for this course.
+# Session 2

-### Step 1. Clone repository
-Go to the [Classes](https://github.com/Python-do-ECARES/Classes) repository on GitHub and click on the "Clone or download" button.
+In this session we cover

-![](images/step1.png)
+- What are Git, GitHub, GitHub Desktop
+- How to use GitHub and GitHub Desktop

-Choose "Open in Desktop".
-
-![](images/step1.A.png)
-
-This will open a window in your GitHub Desktop environment. Check the local path where to clone the repository and click "Clone".
-
-![](images/step1.B.png)
-
```

Step3. Revert Back Changes

Click on ”Revert this commit” to go back to that version.



Update Slides_Intro.pdf
Fabrizio Leone committed 30 minutes ago

Update Readme.md
Fabrizio Leone committed

Delete try1.txt
Fabrizio Leone committed

Create try.txt
Fabrizio Leone committed an hour ago

Delete trial.txt
Fabrizio Leone committed an hour ago

Create trial.txt
Fabrizio Leone committed an hour ago

Update Slides_Intro.pdf
Fabrizio Leone committed an hour ago

Delete try.txt
Fabrizio Leone committed an hour ago

Create try.txt
Fabrizio Leone committed an hour ago

Delete trial.txt
Fabrizio Leone committed 2 hours ago

Create trial.txt

Update Readme.md

Fabrizio Leone committed e16d12d 1 changed file

Session_2\Readme.md

```
@@ -1,77 +1,8 @@
-# First Steps in GitHub
-This tutorial explains how to use GitHub Desktop to
create and maintain own repositories for this cours
e.
+# Session 2

-### Step 1. Clone repository
-Go to the [Classes](https://github.com/Python-do-EC
ARES/Classes) repository on GitHub and click on the
 "Clone or download" button.
+In this session we cover

-![](images/step1.png)
+- What are Git, GitHub, GitHub Desktop
+- How to use GitHub and GitHub Desktop

-Choose "Open in Desktop".
-
-![](images/step1.A.png)
-
-This will open a window in your GitHub Desktop envi
ronment. Check the local path where to clone the rep
ository and click "Clone".
-
-![](images/step1.B.png)
-
```

More Details

- You can switch back and forth your versions as many times as you want. **Your local directories will change accordingly**.
- If instead of ”Revert this commit” you hit ”View on GitHub”, you can see all the differences between two versions of your file online.

More Functionalities

More Functionalities

- Communicate with other people (your coauthors, other developers, etc.) by creating **issues**. Look at here for how to do it.
- Merge and pull.
- Automatic workflow.
- GitHub pages / wiki.