GitHub Manual for Golden's lab

Eugenia Nazrullaeva, enazrullaeva@ucla.edu

Miriam Golden, miriam.golden@eui.eu

March 19, 2018; rev. May 6, 2020

Contents

1	Introduction (adapted from GitHub Guides)					
2	Sign up for a GitHub account (and upgrade to a free educational account)					
3 Request access to the shared private repository golden-lab						
4	Collaborate on the project via the GitHub Desktop application					
	4.1	Install GitHub Desktop	6			
	4.2	Add the golden-lab repository to your computer	8			
	4.3	Sync the repo on your computer to reflect someone else's changes: "Fetch origin" / "Pull origin"	10			
	4.4	Sync the repo on your computer to reflect your changes: "Commit" and "Fetch origin" / "Push origin"	11			
5	Coll	aborate on the project via the Terminal / Command prompt	13			
	5.1	Open the Terminal on Mac	13			
	5.2	Open the Command Prompt on Windows	13			
	5.3	Copy the remote repo to your computer	14			
	5.4	Sync the repo on your computer to reflect someone else's changes: "Pull"	15			
	5.5	Sync the repo on your computer to reflect your changes: "Push"	16			

1 Introduction (adapted from GitHub Guides)

"GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere." The GitHub workflow includes working with files in repositories stored on GitHub, committing changes to them, and pushing (or syncing) your changes back to GitHub.

A **repository** (repo, for short) organizes a single project. "Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs." Our repository is called "golden-lab". It is private, meaning only collaborators invited by Miriam can view and change files.

"On GitHub, saved changes [to any file] are called **commits**. Each commit has an associated commit message, which is a description... [of each change]. Commit messages capture the history of your changes, so other contributors can understand what you have done and why." Good work practices include making modular commits — one commit for each set of changes — and writing clear, concise commit messages.

The workflow at the heart of collaboration on GitHub includes: (1) establishing (**cloning**) an initial copy of a remote repo on your computer, (2) **pulling** (**syncing**) the updated version of the repo from GitHub, which reflects changes made by your collaborators since you last pulled, and (3) **committing and pushing** (**committing and syncing**) the changes you make on your local machine to any document back to the remote repo on GitHub for your collaborators to see.

With GitHub, multiple users can work on the same document simultaneously. GitHub will automatically integrate changes made by multiple users. If two users simultaneously edit the same line in a file, GitHub will inform you that there is a conflict and ask you to resolve it manually. If this happens, please feel free to ask Miriam for real-time help if you are not sure what to do. Stack overflow also provides lots of Git help.

GitHub offers the best version control system currently available. This not only allows you to see and revert to different versions of the file, knowing who changed it and how, but **traces changes within a file made by multiple collaborators** using *git blame* (more on this here). Also, every commit saves the entire repository, so it is easy to go back in time to earlier versions of your project if you want.

Once you have cloned (i.e. copied) the repo (i.e. all the lab files) to your local computer(-s), your workflow will always be: (1) pull (sync), so you have the most recent version of all files; (2) edit any file(-s) you wish, using your normal editor; (3) commit your changes and record what you did with a commit message; (4) push (sync) your changes back to GitHub so everyone else has them. A good practice is to commit and push (sync) whenever you walk away from your computer, in case you get

¹GitHub Guides.

²Ibid.

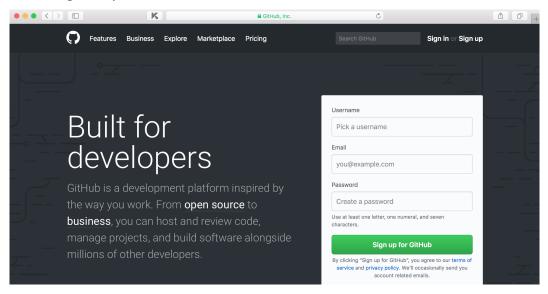
³Ibid.

distracted — you don't want your work to get lost! And don't forget to pull when you sit down so you have everyone else's updates.

Finally, at the top of the golden-lab repo on GitHub, you will see the **Issues** tab. Click on it to see any tasks assigned to you, and use it to discuss progress with or ask questions of others assigned to the same task. Use of the Issues tab will automatically generate email records of the discussion.

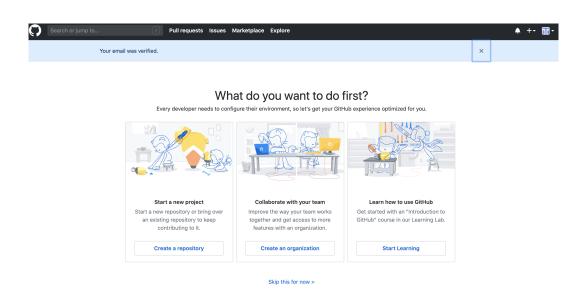
2 Sign up for a GitHub account (and upgrade to a free educational account)

Let's create a GitHub Account first if you do not have one. Sign up for GitHub here: https://github.com. Make sure to remember your username, because you are going to need it later in order to be added to the golden-lab repository on GitHub.



At the next step, you will verify that you are a real person, and answer some questions about your occupation, programming background, etc. Finally, GitHub asks you to verify your email address. That should be it.

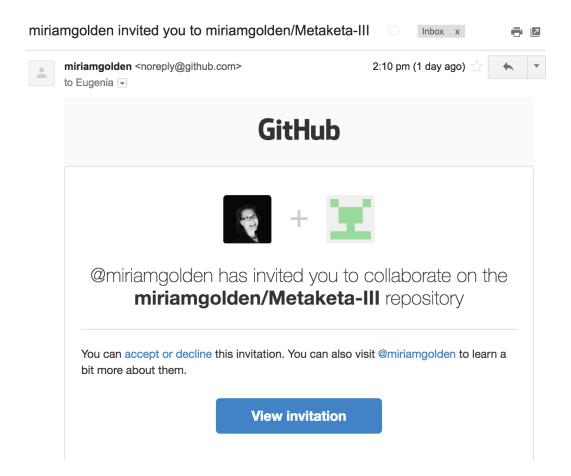
This is what you should see at the next stage:



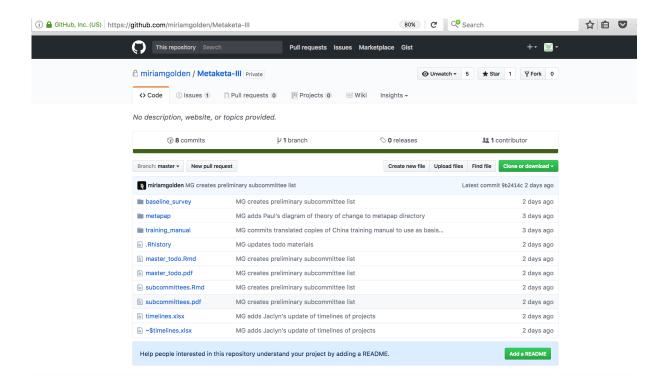
You can click "Skip this for now" at the bottom. You don't need to do anything else for the moment.

3 Request access to the shared private repository golden-lab

When you have your GitHub account set up, send an email to Miriam Golden (miriam.golden@eui.eu) and request access to the private repository golden-lab. To be added to this repository, you are going to need your username (as you are going to see in the examples below, my username is "neugenia"). Once she has your username, MG can give you access. You are going to get an email that looks like this (but with the name of the repository "golden-lab" in place of "Metaketa-III").



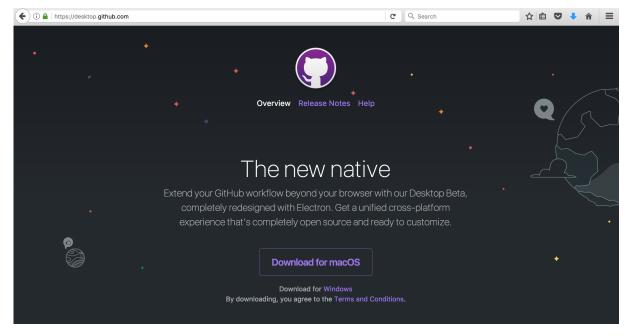
Click on "View invitation" and accept it. When you accept the invitation, you should be able to see all the materials that are in the repository "golden-lab" online here: https://github.com/miriamgolden/golden-lab.



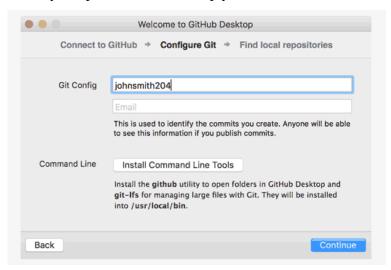
4 Collaborate on the project via the GitHub Desktop application

4.1 Install GitHub Desktop

Install the "GitHub Desktop" application/program. Go here: https://desktop.github.com/. It is available both for Mac and Windows.



When you open GitHub Desktop you see:

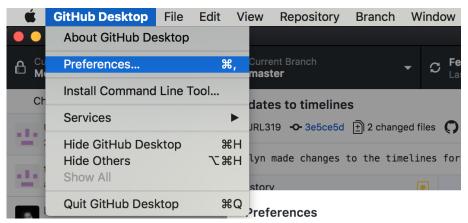


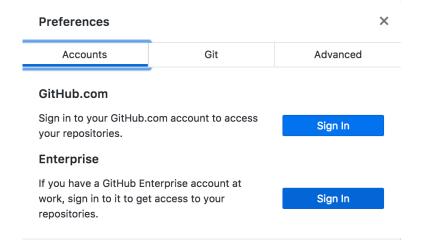
In order to work with the application you need to sign in. You need your username and password for the GitHub account.

If you skip the welcome step, you can always sign in (and sign out) using the menu bar at the top.

On Mac: Go to the upper-left corner and select: GitHub Desktop \rightarrow Preferences \rightarrow Accounts

On Windows: Go to the upper-right corner and click the Settings icon (log in), select: Settings \rightarrow Account





4.2 Add the golden-lab repository to your computer

Now we need to add the golden-lab repository (repo) to your computer.

For example, I am going to copy the remote repo golden-lab with all its files to the following directory on my computer:

On Mac:

~/GitHub/golden-lab

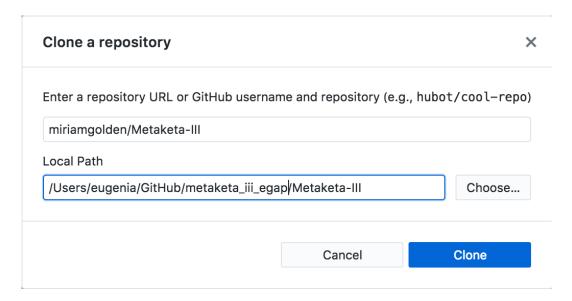
On Windows:

C:\Users\Eugenia\GitHub\golden-lab

Let's go to the menu bar at the top-left and choose File \rightarrow Clone Repository (both on Mac and Windows):

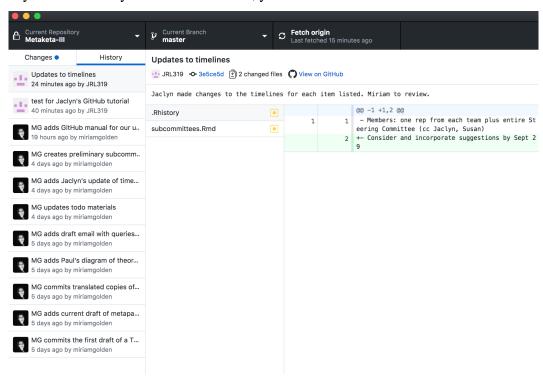


If you successfully accepted the invitation to join the private repo golden-lab, in the menu below type in **miriamgolden/golden-lab**:

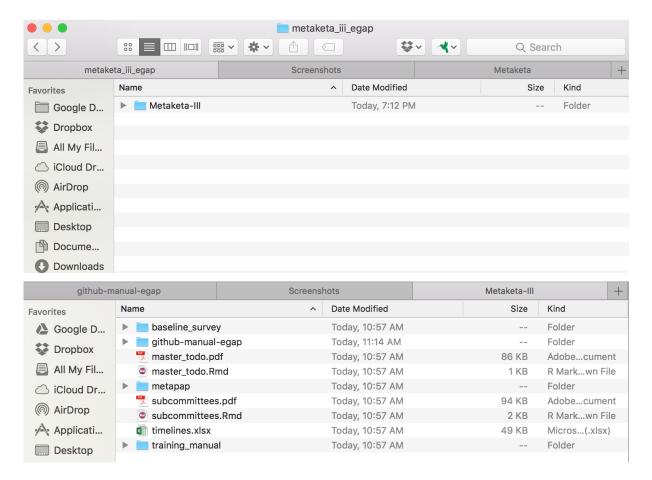


Choose a folder ("Local Path") on your computer where you want to keep a copy of the repo (it is going to be linked to the remote repo). You could choose any folder/path. Now click "Clone".

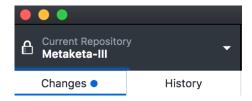
If you successfully "cloned" the folder, you should be able to see its contents via the GitHub Desktop.



as well as in the folder which you created on your computer:

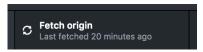


To the left you can see two panels: "Changes" and "History". "Changes" reflects the changes to the repo that you made on your computer, and "History" keeps track of the changes that your collaborators made to the remote repo.



4.3 Sync the repo on your computer to reflect someone else's changes: "Fetch origin" / "Pull origin"

Important: Whenever you want to update your local copy of the joint/remote repo, click the button "Fetch origin" to sync changes.



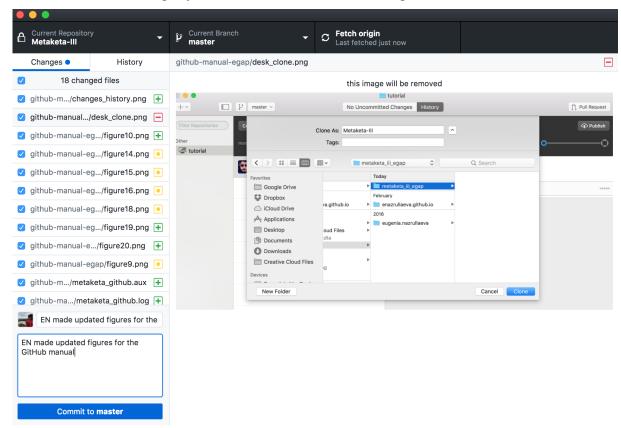
"Fetch origin" changes keeps track of the changes that you made in your local repository (on your

computer) and the changes someone else made and committed to the remote repo.

If someone else made changes to the remote repo which you do not yet have on your local computer, when you click on the "Fetch origin" button it becomes the "Pull origin" button. To update the files on your computer click the "Pull origin" button. This is how you pull changes from the remote repo to your computer.

4.4 Sync the repo on your computer to reflect your changes: "Commit" and "Fetch origin" / "Push origin"

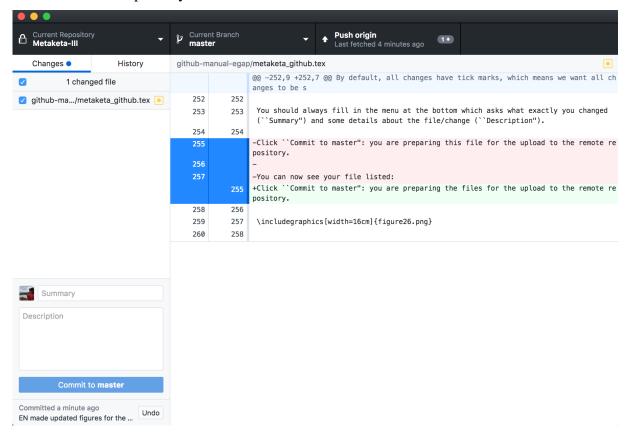
Suppose you have a new file to contribute, or you edited an existing file and want to commit changes. For example, I updated the figures for this manual on my local computer (in the folder which we synced with GitHub), and GitHub Desktop automatically reflects the changes under the "Changes" panel. If it does not reflect all the changes you made, click on the "Fetch origin" button.



By default, all updates to the files are selected, which means we want all changes to be synced with the remote repo.

You should always fill in the menu at the bottom which asks what exactly you changed ("Summary") and some details about the file/change ("Description").

Click the "Commit to master" button at the bottom-left corner: you are preparing the files for the upload to the remote repository.



You can now see at the top-right corner that the "Fetch origin" button turned into "Push origin". Click "Push origin". This is how you push changes from your computer to the remote repo for your collaborators to see.

That's it! Remember to always **commit** and **push** all the changes you make.

5 Collaborate on the project via the Terminal / Command prompt

5.1 Open the Terminal on Mac

If you use the terminal, you might prefer to work in GitHub this way. Use of the terminal is more efficient than working with a GUI, and offers a clearer grasp of what you are doing. But you should go with whichever route you prefer (and you can always switch from one to the other).

Open the command shell — the terminal (on Mac you can simply search for "terminal" or find the Terminal in "Other" applications).



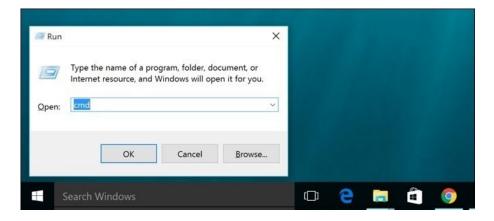
Here is the terminal:



5.2 Open the Command Prompt on Windows

Press Windows+R to open the "Run" box.

Type "cmd" and then click "OK" to open a regular Command Prompt. Type "cmd" and then press Ctrl+Shift+Enter to open an administrator Command Prompt.



Other ways to open the command prompt are described here.

5.3 Copy the remote repo to your computer

After you set up the GitHub account (your username and password), you need to establish your copy of the golden-lab repository (repo) on your computer.

First, let's set the default folder where we are going to clone (copy) all files from the golden-lab repo. To do this, we need to create an empty folder on your computer. Any working directory is fine, except for Dropbox (GitHub can be in conflict with Dropbox).

I set this folder as my working directory:

On Mac:

cd ~/GitHub/golden-lab

On Windows:

cd C:\Users\Eugenia\GitHub\golden-lab

```
egap_metaketa_iii — -bash — 80×24

~/GitHub/egap_metaketa_iii — -bash

Last login: Wed Aug 9 17:43:21 on ttys000

Eugenias-MacBook-Pro-2:~ eugenia$ cd ~/GitHub/egap_metaketa_iii

Eugenias-MacBook-Pro-2:egap_metaketa_iii eugenia$
```

Now let's establish your copy of the golden-lab private repo on your computer, that is let's "clone" this repo. In order to do this, run the following line:

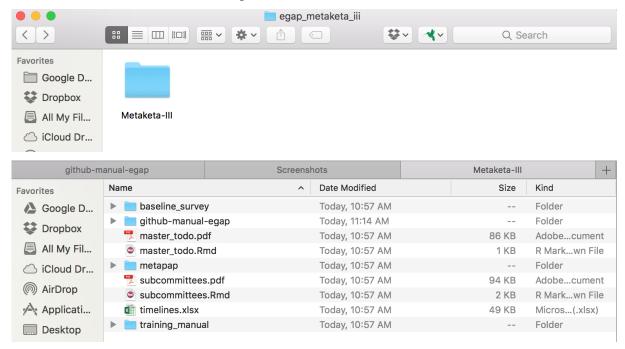
git clone https://username:password@github.com/miriamgolden/golden-lab.git

where you change "username" to your username and "password" to your password.

If cloning was successful you should see a report (substitute "golden-lab" in the report for "Metaketa-III"):

```
Cloning into 'Metaketa-III'...
remote: Counting objects: 40, done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 40 (delta 8), reused 37 (delta 5), pack-reused 0
Unpacking objects: 100% (40/40), done.
Checking connectivity... done.
Eugenias-MacBook-Pro-2:egap_metaketa_iii eugenia$ cd ~/GitHub/egap metaketa iii
```

Now if you check your working directory, you should be able to see the folder golden-lab with the files that were added from the remote repo.



5.4 Sync the repo on your computer to reflect someone else's changes: "Pull"

Important: Whenever you want to update your local copy of the repository to reflect changes made by others you do a "pull" from the repository as follows:

```
cd ~/GitHub/golden-lab
git pull https://username:password@github.com/miriamgolden/golden-lab.git
```

where you change "username" to your username and "password" to your password.

In the cd line write down the full path to the golden-lab folder which is linked to the remote repo.

In my example, all the files are up to date, so you are going to see the following:

```
From https://github.com/miriamgolden/Metaketa-III

* branch HEAD -> FETCH_HEAD

Already up-to-date.

Eugenias-MacBook-Pro-2:Metaketa-III eugenia$
```

5.5 Sync the repo on your computer to reflect your changes: "Push"

Now suppose you have a new file to contribute, or you edited an existing file. To commit changes to the remote repo for everyone to see you need to "push" your changes.

As an example, I added a new empty txt file called "eugenia_test.txt".

Name	Date Modified	~	Size	Kind
eugenia_test.txt	Today, 4:31 PM		72 bytes	Plain Text
github-manual-egap	Today, 4:28 PM			Folder
master_todo.pdf	Today, 10:57 AM		86 KB	Adobecument
master_todo.Rmd	Today, 10:57 AM		1 KB	R Markwn File
metapap	Today, 10:57 AM			Folder
📆 subcommittees.pdf	Today, 10:57 AM		94 KB	Adobecument
subcommittees.Rmd	Today, 10:57 AM		2 KB	R Markwn File
timelines.xlsx	Today, 10:57 AM		49 KB	Micros(.xlsx)
training_manual	Today, 10:57 AM			Folder
baseline_survey	Today, 10:57 AM			Folder

Let's add this text file to the joint repository.

Every time you want to update or add a file to the remote repo you should run the following sequence:

On Mac:

```
cd ~/GitHub/golden-lab
git add eugenia_test.txt
git commit -m "Eugenia's test txt file for GitHub cheat sheet"
git push https://username:password@github.com/miriamgolden/golden-lab.git
```

where you change "username" to your username and "password" to your password.

On Windows:

```
cd C:\Users\Eugenia\GitHub\golden-lab
git add eugenia_test.txt
```

git commit -m "Eugenia's test txt file for GitHub cheat sheet" git push https://username:password@github.com/miriamgolden/golden-lab.git

First, you should make sure that you have the cd set up, cd. Then you add the file you want to sync with the remote repo and "commit" the change, git add.

git commit—m means that you want to add a message which briefly describes what the new file or the change to the existing file is about. Finally, you run git push to sync the remote repo with your the folder on your computer.