

GitHub Manual for EGAP NRG Metaketa III

Eugenia Nazrullaeva, enazrullaeva@ucla.edu

Aug 10, 2017

Contents

1	Introduction (adpated from GitHub Guides)	2
2	Sign up for a GitHub account (and upgrade to a free educational account)	3
3	Request access to the shared private repository Metaketa III	5
4	Collaborate on the project via the GitHub Desktop application	7
4.1	Install GitHub Desktop	7
4.2	Add the Metaketa-III repository to your computer	9
4.3	Sync the repo on your computer to reflect someone else's changes: "Fetch origin" and "Pull origin"	11
4.4	Sync the repo on your computer to reflect your changes: "Commit" and "Fetch origin"	12
5	Collaborate on the project via the Terminal / Command prompt	14
5.1	Open the Terminal on Mac	14
5.2	Open the Command Prompt on Windows	14
5.3	Copy the remote repo to your computer	15
5.4	Sync the repo on your computer to reflect someone else's changes: "Pull"	16
5.5	Sync the repo on your computer to reflect your changes: "Push"	17

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 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 “[Metaketa-III](#)”. 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!

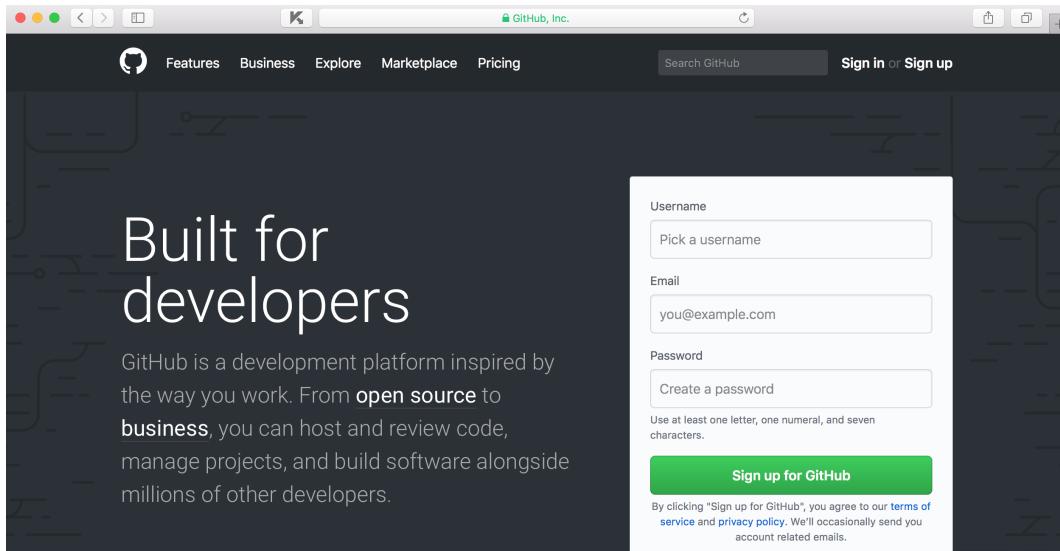
GitHub offers the best version control system currently available. This not only allows you to see and revert to different versions of the file, knowing who changed it and how, but **traces changes within a file made by multiple collaborators using `git blame`** (more on this [here](#)). Also, every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

Once you have cloned (i.e. copied) the repo (i.e. all the project 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 distracted — you don’t want your work to get lost!

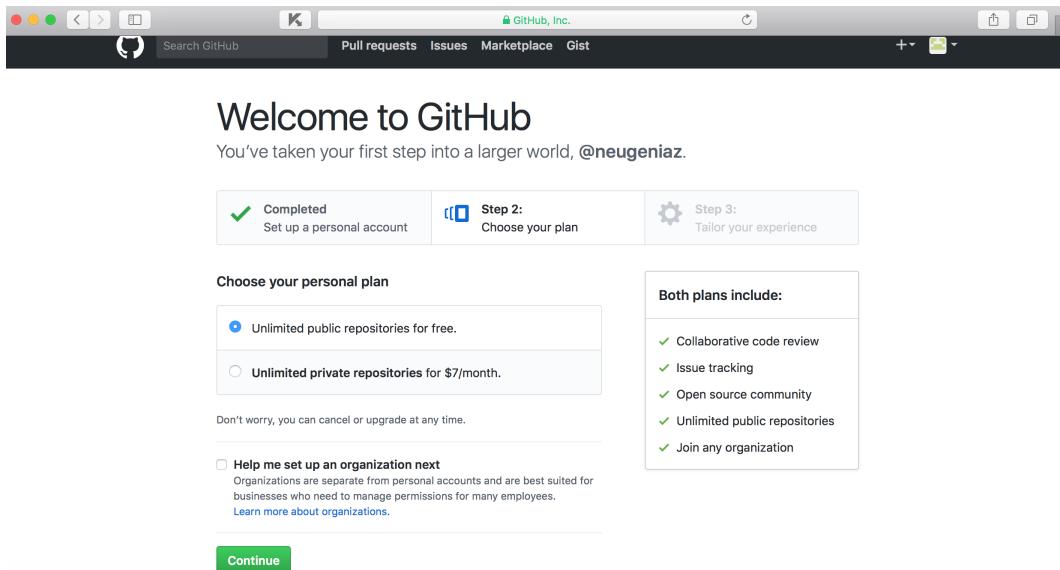
Finally, at the top of the Metaketa repo on GitHub, you will see the **Issues** tab. Click on it to see any tasks assigned to you, and use it to discuss progress with or ask questions of others assigned to the same task. Use of the Issues tab will automatically generate email records of the discussion.

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 Metaketa III repository on GitHub.



At the next step, you see options for public/private repositories. Choose the first option, “Unlimited public repositories for free,” because we are going to upgrade it later to “Unlimited private repositories” for free by setting up a free educational account with private repositories.



At the third step, you are asked to provide some details about your programming background and occupation, etc. Finally, GitHub is going to ask you to verify your email address. That should be it. This is what you should see at the final stage:

The screenshot shows the GitHub homepage. At the top, there's a search bar and navigation links for Pull requests, Issues, Marketplace, and Gist. Below the header, a main section titled "Learn Git and GitHub without any code!" is displayed, featuring a "Read the guide" button and a "Start a project" button. A promotional box for "GitHub Universe" (October 10-12 in San Francisco) is also visible. At the bottom of the page, there's a footer with links to GitHub's Terms, Privacy, Security, Status, Help, and various contact and API links.

Now let's upgrade your GitHub account to make it a free educational account with private repositories. Go to <https://education.github.com/> and click “Request a discount”:

The screenshot shows the GitHub Education website at https://education.github.com/. The page features a green chalkboard background with the text "TEACH AND LEARN BETTER, TOGETHER". On the left, there's an illustration of laboratory glassware. On the right, a cartoon GitHub cat character wearing a lab coat and safety goggles holds a flask. A green button labeled "Request a discount" is positioned between them. Below the chalkboard, there's a section for the "STUDENT DEVELOPER PACK" featuring a yellow backpack icon and a blue button labeled "Get the pack". The page also includes a brief description of the pack and a "Get the pack" button.

Fill in the necessary details (Researcher/Individual account).¹

¹You may notice that I already have an approved discount.

Request a discount

Discounted and free plans are available for educational use

The screenshot shows the GitHub Education 'Request a discount' form. At the top, there are two tabs: 'Step 1 Tell us what you need' and 'Step 2 Tell us about you'. Below the tabs, a message says 'You have submitted 1 request:' followed by a list item: '• Feb 15, 2016 for @neugenia – Approved'. The main section asks 'Which best describes you?' with five options: 'Student', 'Teacher', 'Researcher', 'Administrator/staff', and 'Other'. Below that, it asks 'What are you looking to get a discount for?' with two options: 'Individual account' and 'Organization account'. A green 'Next' button is at the bottom.

At the next step, you are going to be asked to fill in more details about yourself and how you are going to use GitHub, as well as provide your University email address. That should be it!

3 Request access to the shared private repository Metaketa III

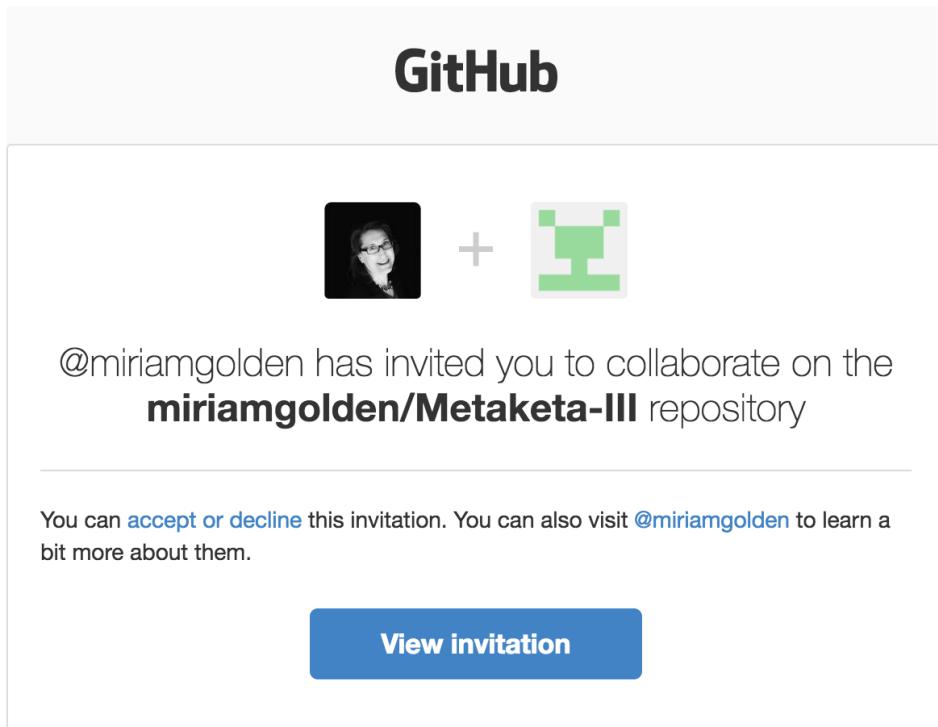
When you have your GitHub account set up, send an email to Miriam Golden (golden@ucla.edu) and request access to the private repository Metaketa III. To be added to this repository, you are going to need your username (as you are going to see in the examples below, my username is “neugenia”). Knowing your username, MG can give you access. You are going to get an email which looks like this:

miriamgolden invited you to miriamgolden/Metaketa-III

Inbox x

 miriamgolden <noreply@github.com> 2:10 pm (1 day ago) 

to Eugenia 



Click on “View invitation” and accept it. When you accept the invitation, you should be able to see all the materials from the repository “Metaketa III” online here <https://github.com/miriamgolden/Metaketa-III>.

The screenshot shows a GitHub repository page for 'miriamgolden / Metaketa-III'. At the top, there's a header with a search bar and navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Gist'. Below the header, the repository name 'miriamgolden / Metaketa-III' is displayed, along with a 'Private' status, an 'Unwatch' button (with 5 notifications), a 'Star' button (with 1 star), and a 'Fork' button (with 0 forks). A navigation bar below the repository name includes tabs for 'Code', 'Issues 1', 'Pull requests 0', 'Projects 0', 'Wiki', and 'Insights'. A note below the navigation bar states 'No description, website, or topics provided.' Below this, a summary bar shows '8 commits', '1 branch', '0 releases', and '1 contributor'. A dropdown menu for the 'master' branch is open, showing options like 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. The main content area lists 8 commits by 'miriamgolden' with the following details:

File	Description	Date
baseline_survey	MG creates preliminary subcommittee list	2 days ago
metapap	MG adds Paul's diagram of theory of change to metapap directory	3 days ago
training_manual	MG commits translated copies of China training manual to use as basis...	3 days ago
.Rhistory	MG updates todo materials	2 days ago
master_todo.Rmd	MG creates preliminary subcommittee list	2 days ago
master_todo.pdf	MG creates preliminary subcommittee list	2 days ago
subcommittees.Rmd	MG creates preliminary subcommittee list	2 days ago
subcommittees.pdf	MG creates preliminary subcommittee list	2 days ago
timelines.xlsx	MG adds Jaclyn's update of timelines of projects	2 days ago
~\$timelines.xlsx	MG adds Jaclyn's update of timelines of projects	2 days ago

At the bottom of the commit list, a note says 'Help people interested in this repository understand your project by adding a README.' with a 'Add a README' button.

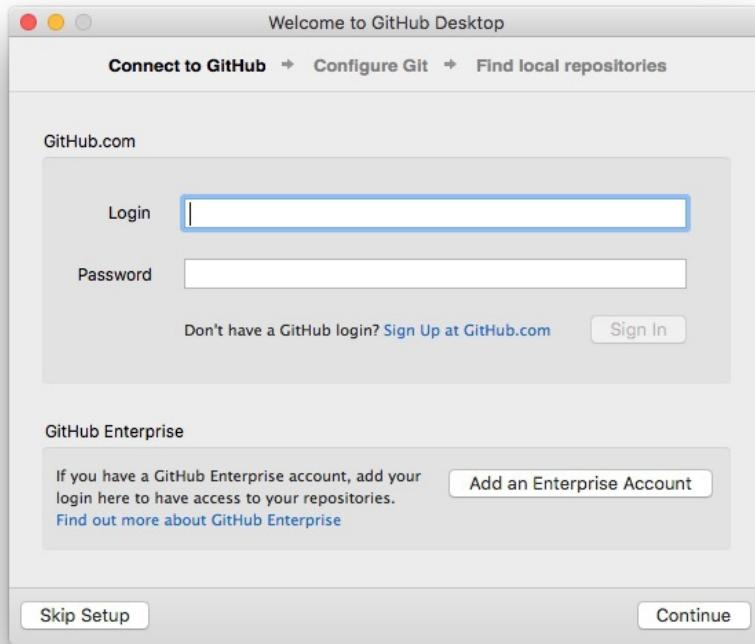
4 Collaborate on the project via the GitHub Desktop application

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.

The screenshot shows the GitHub Desktop Beta landing page. At the top, there's a header with a search bar and navigation links for 'Overview', 'Release Notes', and 'Help'. The main title 'The new native' is prominently displayed. Below the title, a subtitle reads 'Extend your GitHub workflow beyond your browser with our Desktop Beta, completely redesigned with Electron. Get a unified cross-platform experience that's completely open source and ready to customize.' A large 'Download for macOS' button is centered, with smaller buttons for 'Download for Windows' and a note about accepting 'Terms and Conditions'. To the right, there's a stylized globe icon with a heart shape.

When you open GitHub Desktop you see:

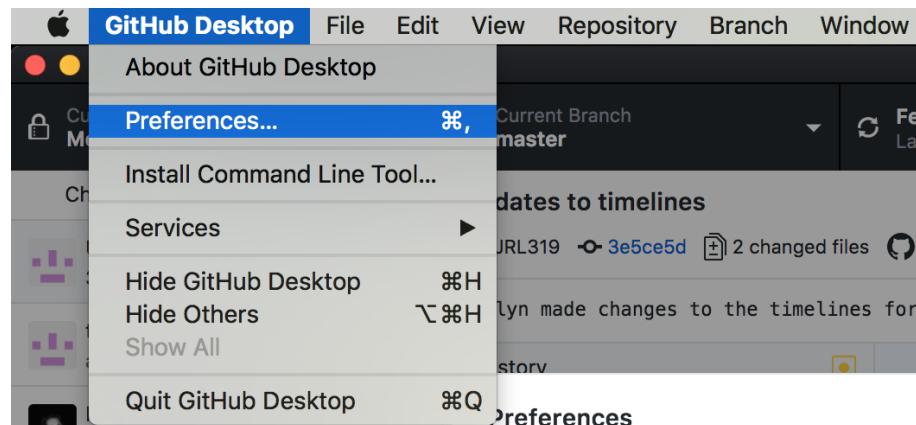


In order to work with the application you need to sign in. You need your username and password for the GitHub account.

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 → Preferences → Accounts

On Windows: Go to the upper-right corner and click your profile photo, select: Settings → Account



The screenshot shows the GitHub.com Preferences page with the 'Accounts' tab selected. There are two main sections: 'GitHub.com' and 'Enterprise'. Each section has a brief description and a blue 'Sign In' button.

GitHub.com
Sign in to your GitHub.com account to access your repositories. **Sign In**

Enterprise
If you have a GitHub Enterprise account at work, sign in to it to get access to your repositories. **Sign In**

4.2 Add the Metaketa-III repository to your computer

Now we need to add the Metaketa-III repository (repo) to your computer.

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

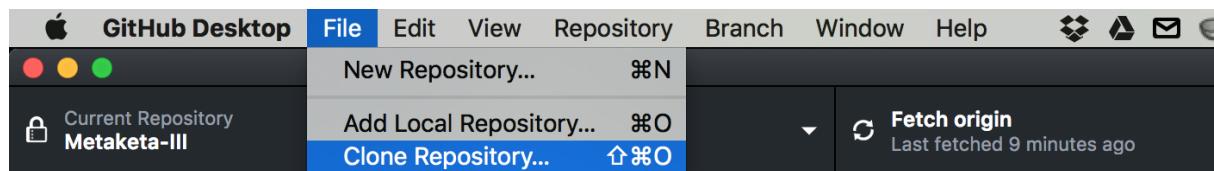
On Mac:

`~/GitHub/metaketa-iii_egap`

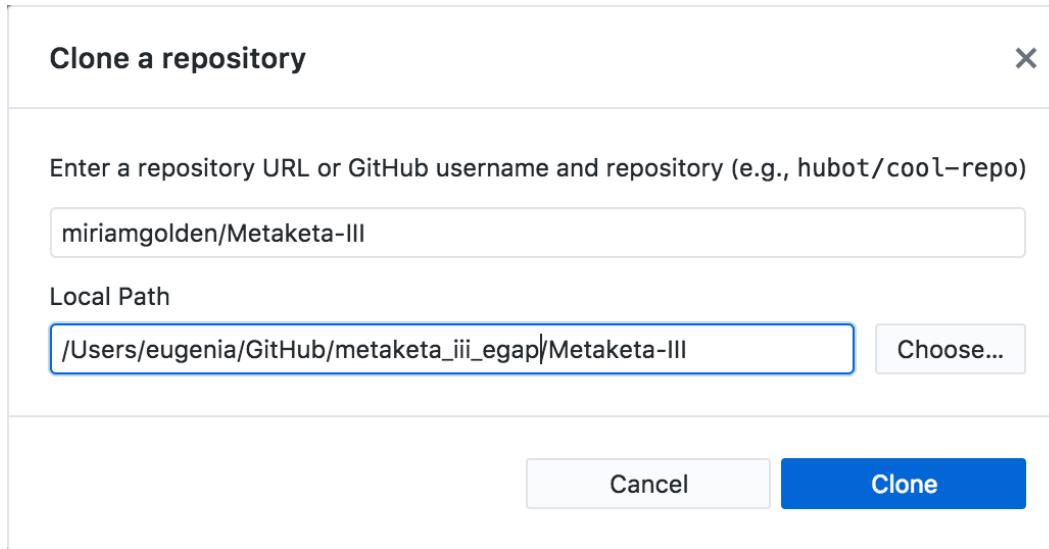
On Windows:

`C:\Users\Eugenia\GitHub\metaketa-iii_egap`

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

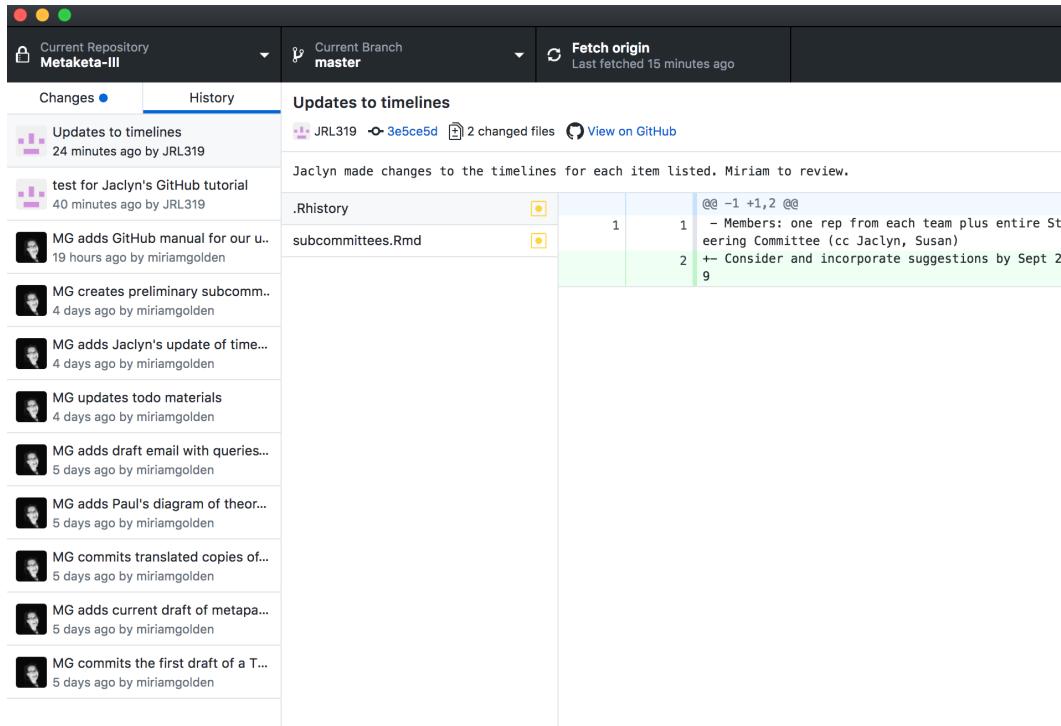


If you successfully accepted the invitation to join the private repo Metaketa III, in the menu below type in **miriamgolden/Metaketa-III**:

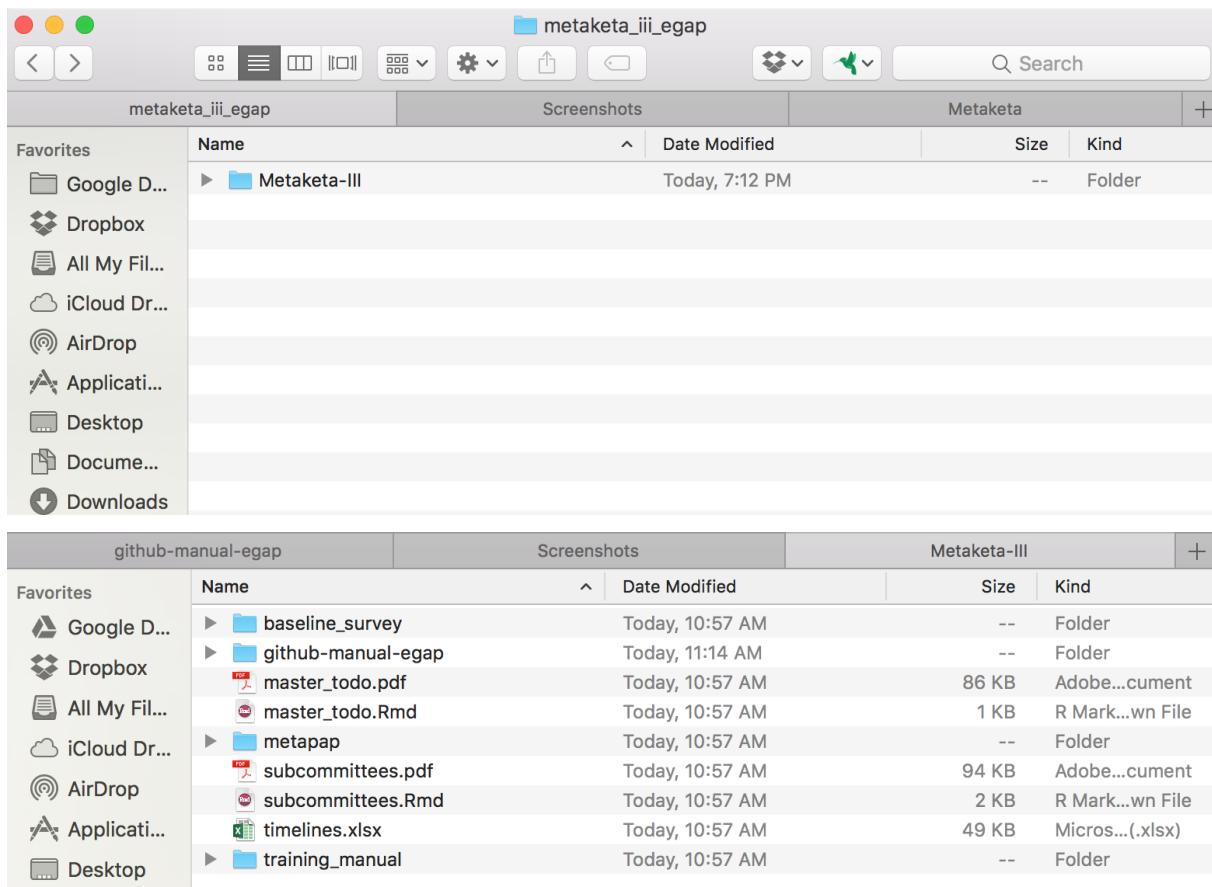


Choose a folder (“Local Path”) on your computer where you want to keep a copy of the repo (it is going to be linked to the remote repo). You could choose any folder/path. Now click “Clone”.

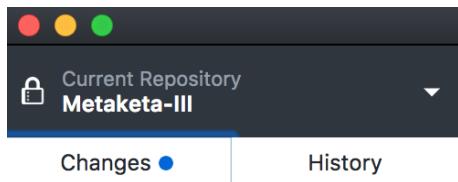
If you successfully “cloned” the folder, you should be able to see its contents via the GitHub Desktop.



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

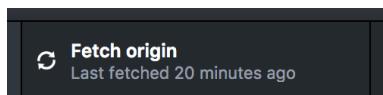


To the left you can see two panels: “Changes” and “History”. “Changes” reflects the changes to the repo that you made on your computer, and “History” keeps track of the changes that your collaborators made to the remote repo.



4.3 Sync the repo on your computer to reflect someone else's changes: “Fetch origin” and “Pull origin”

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



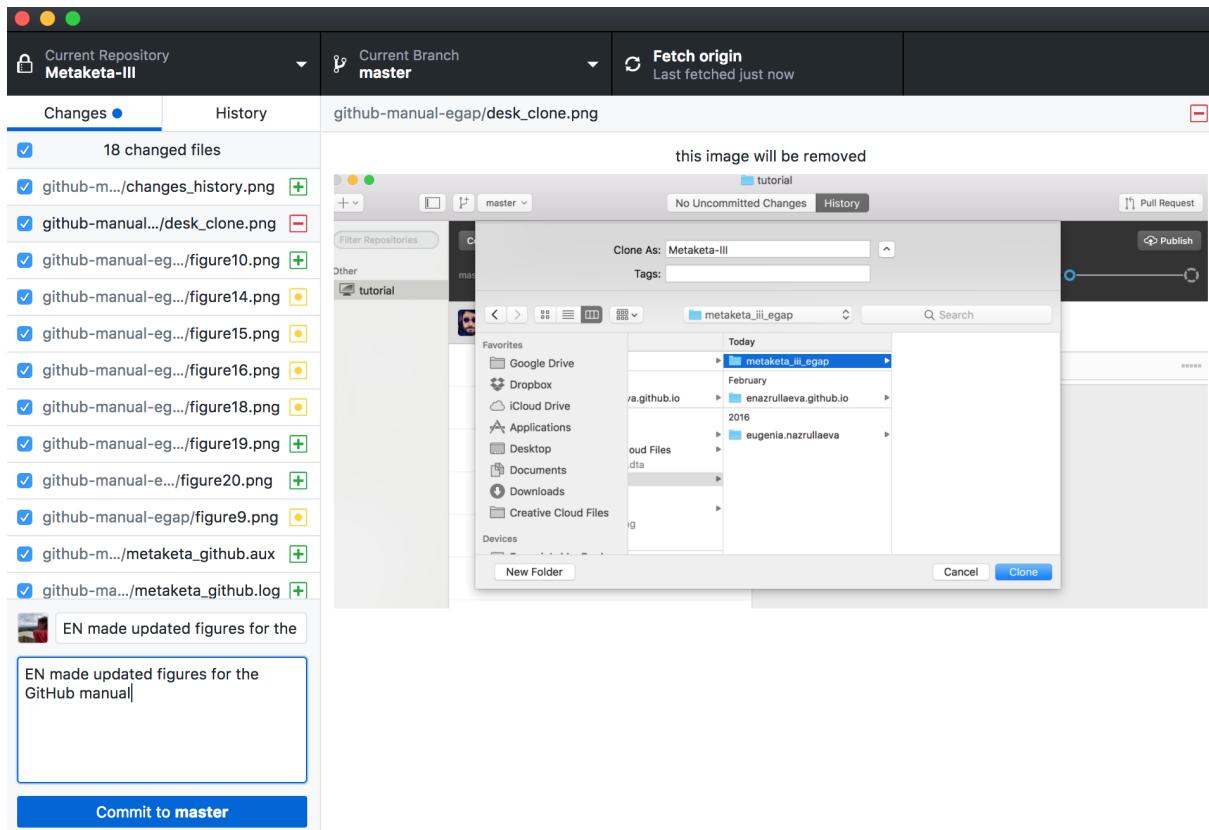
“Fetch origin” changes keeps track of the changes that you made in your local repository (on your

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”

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 changes have tick marks, 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.

The screenshot shows the GitHub desktop application interface. At the top, there's a header with 'Current Repository' set to 'Metaketa-III', 'Current Branch' set to 'master', and a 'Push origin' button with a tooltip 'Last fetched 4 minutes ago'. Below the header, there are two tabs: 'Changes' (selected) and 'History'. The 'Changes' tab shows a list of commits with file names like 'github-manual-egap/metaketa_github.tex' and 'github-ma.../metaketa_github.tex'. The 'History' tab shows a list of commits with file names like 'github-manual-egap/metaketa_github.tex' and 'github-ma.../metaketa_github.tex'. A large blue box highlights the commit message for commit 255, which contains instructions for committing changes to the master branch. Below the commit list, there's a summary dialog with fields for 'Summary' and 'Description', a 'Commit to master' button, and a status message 'Committed a minute ago'.

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 via it.

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



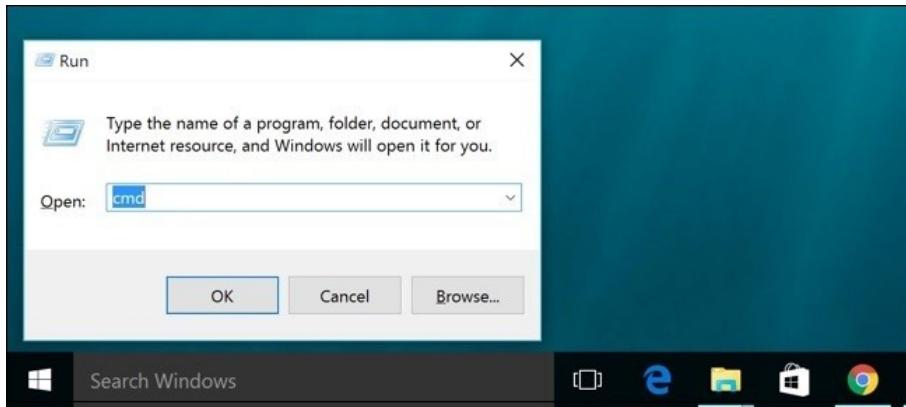
Here is the terminal:

A screenshot of a Mac terminal window titled "eugenia — -bash — 80x24". The window shows the user's last login information: "Last login: Wed Aug 9 09:27:54 on ttys000" and the command prompt "Eugenias-MacBook-Pro-2:~ eugenia\$". The window has standard OS X window controls (red, yellow, green) and a plus sign icon in the top right corner.

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 Metaketa III repository (repo) on your computer.

First, let's set the default folder where we are going to clone (copy) all files from the Metaketa-III 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/egap_metaketa_iii
```

On Windows:

```
cd C:\Users\Eugenia\GitHub\egap_metaketa_iii
```

```
Last login: Wed Aug  9 17:43:21 on ttys000
Eugenias-MacBook-Pro-2:~ eugenias$ cd ~/GitHub/egap_metaketa_iii
Eugenias-MacBook-Pro-2:egap_metaketa_iii eugenias$
```

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

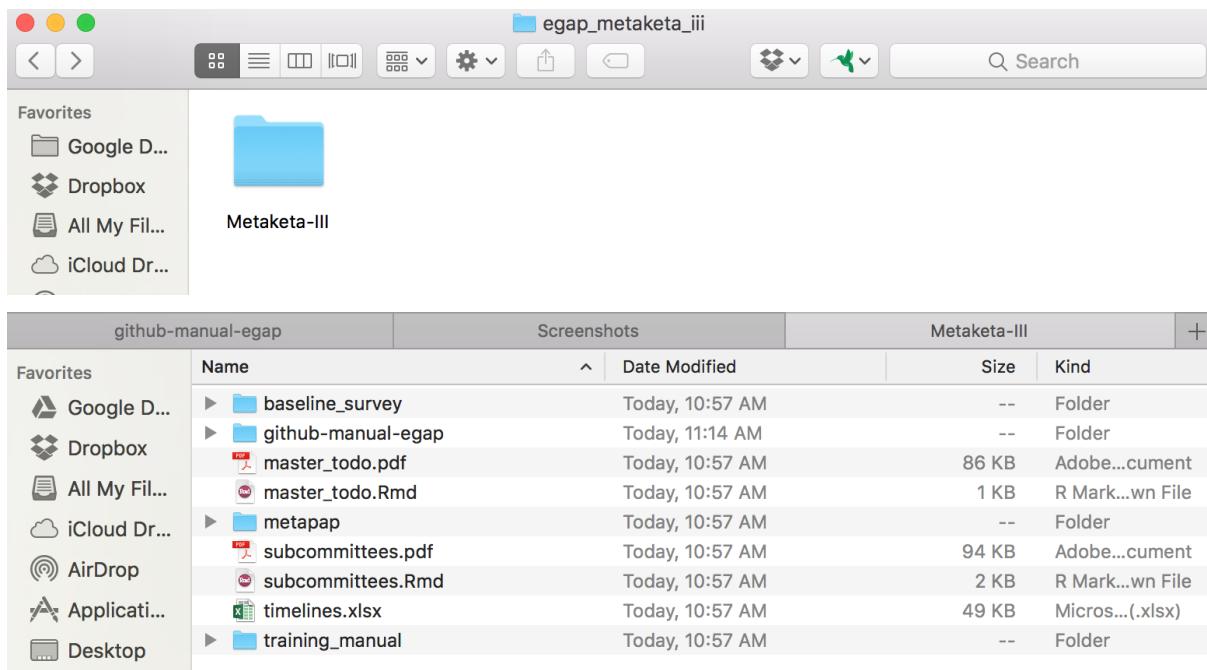
```
git clone https://username:password@github.com/miriampgolden/Metaketa-III.git
```

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

If cloning was successful you should see a report:

```
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 Metaketa-III with the files which 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/egap_metaketa_iii/Metaketa-III
git pull
```

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

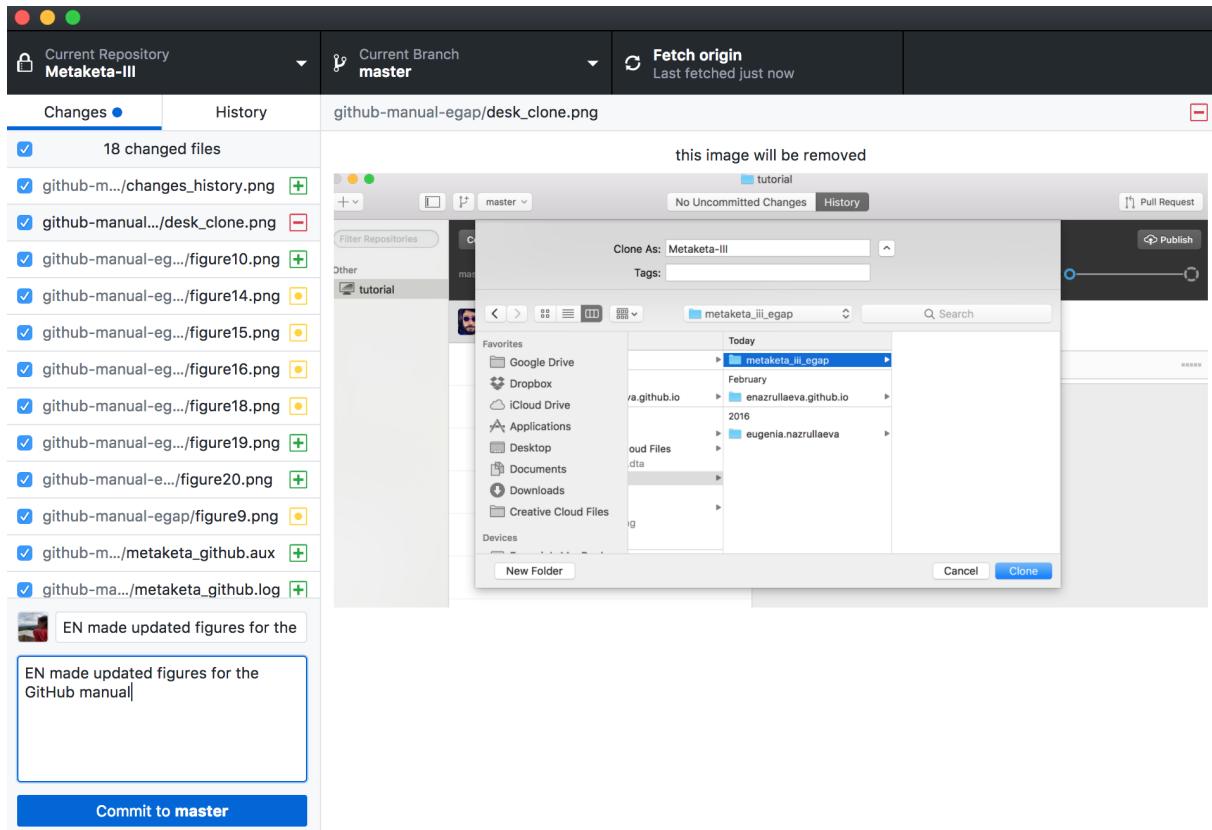
In my example, all the files are up to date, so you are going to see the following:

```
From https://github.com/miriamgolden/Metaketa-III
 * branch            HEAD      -> FETCH_HEAD
 Already up-to-date.
 Eugenia-MacBook-Pro-2:Metaketa-III eugenias$
```

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”.



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/egap_metaketa_iii/Metaketa-III
git add eugenia_test.txt
git commit -m "Eugenia's test txt file for GitHub cheat sheet"
```

```
git push
```

On Windows:

```
cd C:\Users\Eugenia\GitHub\egap_metaketa_iii\Metaketa-III  
git add eugenia_test.txt  
git commit -m "Eugenia's test txt file for GitHub cheat sheet"  
git push
```

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.