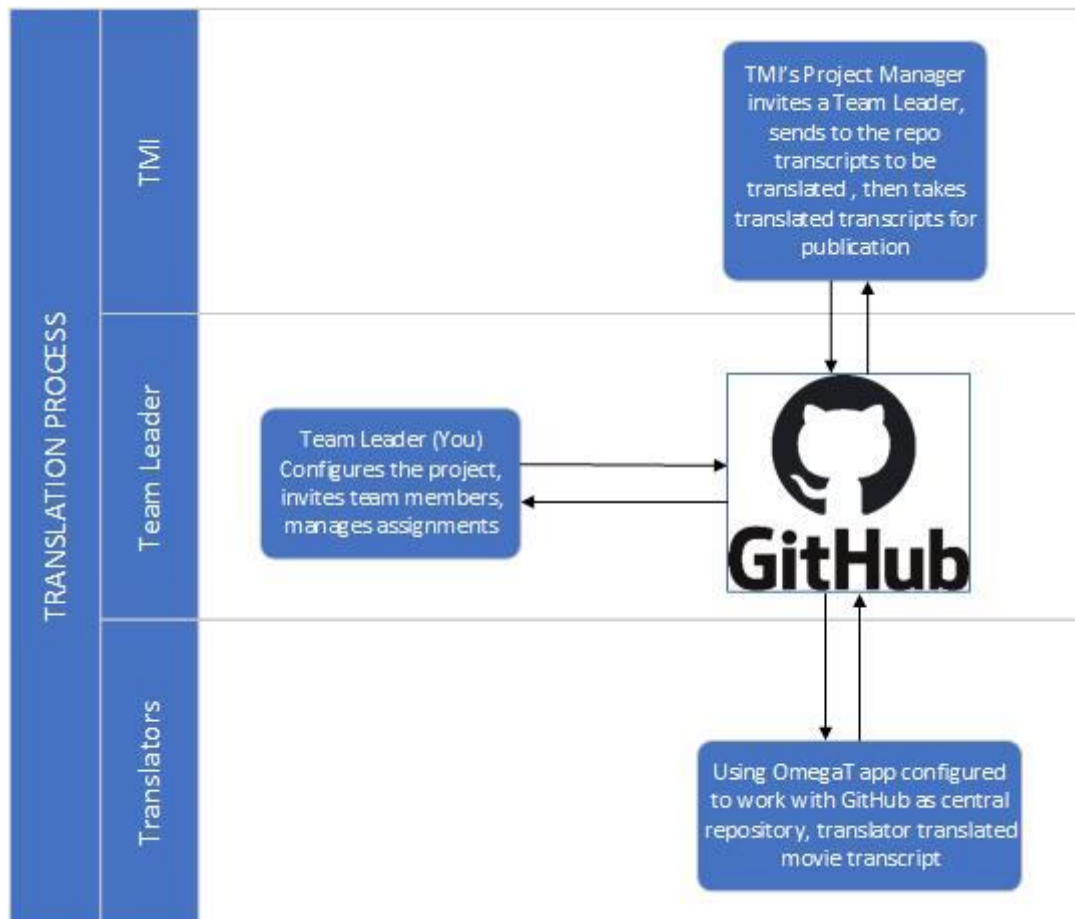


Team Leader's Manual

In this project you will be managing a subtitles translation. This project will deliver translated subtitle text files to be submitted to TMI's YouTube channel.

As a Team Leader you will have to know the whole process, so to manage translation tasks smoothly and deliver translations of high quality. Here is a schema of translation process:



The original subtitle files in the project are generated automatically by advanced speech to text conversion software. This software is able to learn from processed videos and deliver the more accurate transcripts as much material is processed. But, it does not analyze the semantics and therefore, depending on speaker's pronunciation and expression, sometimes unexpected punctuation or wrongly recognized words may occur, which may change the meaning of sentences. These are rare situations, but you have to be aware of that and be ready to advise translators to watch first the video and correct such questionable elements.

The translators in your Team will be using specialized CAT (*Computer Aided Translation*) software, which will facilitate the translation process and preserve the structure of translated files. This software also has the ability of learning from already translated phrases, so the more translations have been done the more the software will be able to provide reliable automated translations. You will be responsible for merging translation memory as to make this process going fast and smooth.

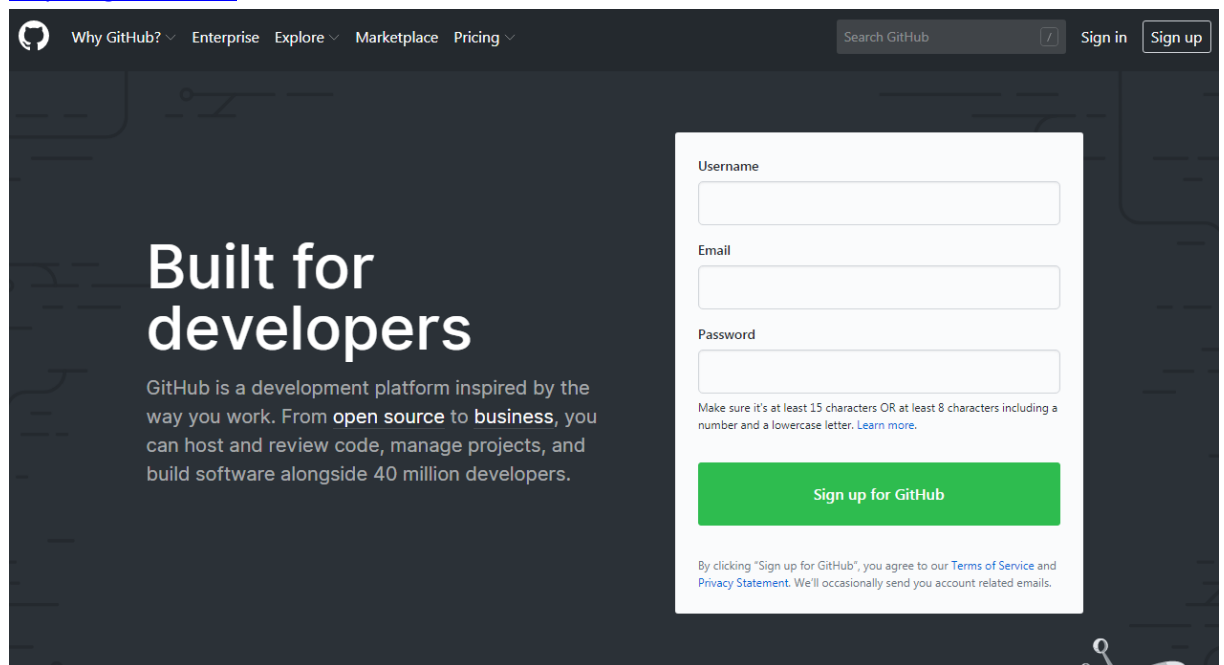
The subtitle files the translators will work on cannot be modified manually since this can damage their structure and so make them unusable for YouTube. This is another reason why the CAT software is used. It deals with these specific files preserving their format and presents only the text to be translated.

The vital role in the project plays GitHub repository. This is version control software that allows to keep track of changes in files, merge changes done at the same time by team members and store all necessary project files and documents. It also allows to take control over the access to the project files. Additionally it provides project managing tools like planning your project, sorting tasks, tracking progress, automating the workflow, wrapping project up and includes team discussion board.

Prepare your working environment

1. First of all sign up for GitHub – the repository we will be working with at:

<https://github.com>



Why GitHub? Enterprise Explore Marketplace Pricing

Search GitHub

Sign in Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside 40 million developers.

Username

Email

Password

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

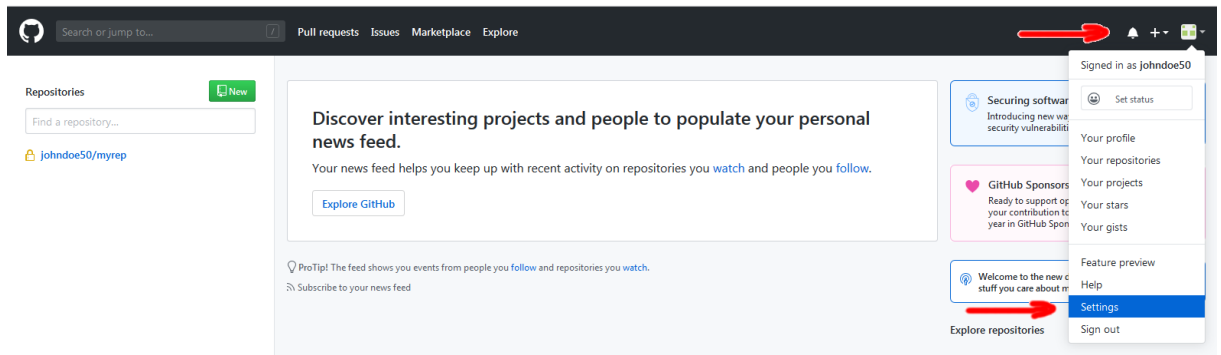
Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account related emails.

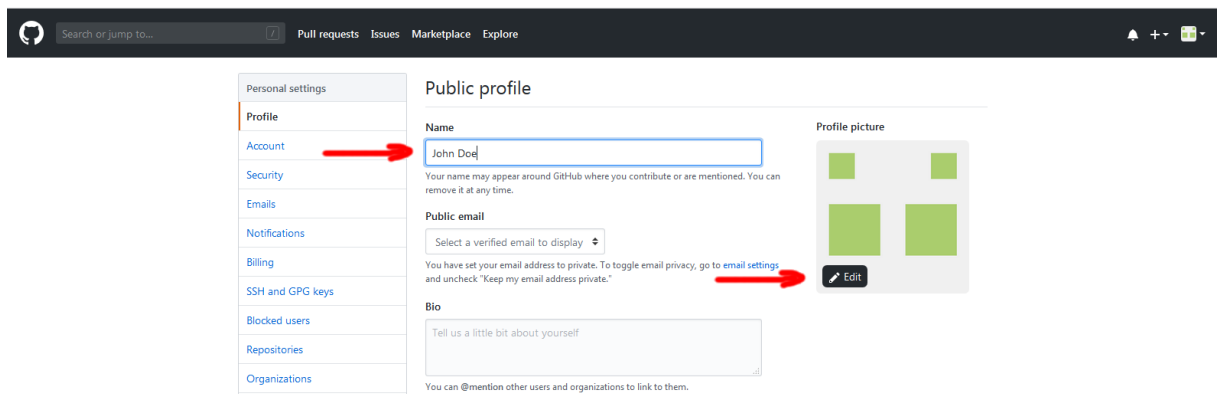
Go through registration process – it is very easy and quick. Don't forget to **click the confirmation link**, that you will obtain in an e-mail from GitHub. Then contact Project Manager from TMI and provide your username or e-mail address you used for signing up for GitHub. Having this information the TMI will add you to the *monroe-institute* organization and grant you necessary privileges to set up and run your translation project.

2. Spend a while on exploring your GitHub account making initial settings like your name and avatar. It will be easier to communicate within the team then, when team members will use real names instead of GitHub usernames.

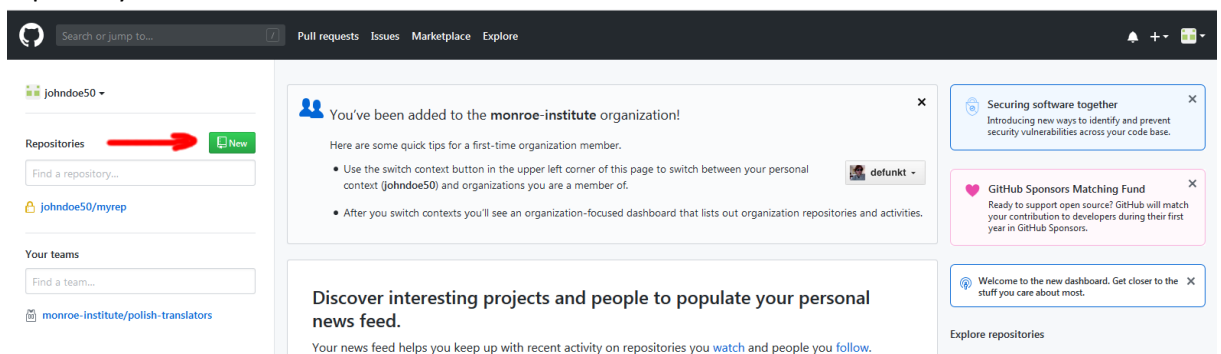
In order to do so, locate your avatar icon in the right top corner and click on that. Then, click on the *Settings* item from the dropdown menu.



You will be moved to your account settings. Fill up *Name* field. If you want, upload your photo and provide other information like your bio, URL, etc., do it as well. Having entered all necessary information scroll down the page to green button *Update profile* and confirm your changes.



3. A good idea is to **elevate the security of your account**. So, click the *Security* link on the left pane and then locate a green button labeled *Enable two factor authentication*. Click on it and follow the instructions.
4. When you are approved as a member of *monroe-institute* organization, create a new repository.



Select the *monroe-institute* organization, set the *Repository name*, *Description* (do not treat this as optional) and check the *Initialize this repository with a README*. Then click *Create repository* button. The pattern for naming the repositories is: *yourlanguage-subtitles* (e.g. english-subtitles)

Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner: monroe-institute / Repository name: yourlanguage-subtitles ✓

Great repository names are short and memorable. Need inspiration? How about [glowing-system?](#)

Description (optional): A translation project for TMI videos

☒ Public
Anyone can see this repository. You choose who can commit.

☐ Private
Your current plan does not support private repositories. Your organization's owners will need to upgrade to Team.

Skip this step if you're importing an existing repository.

☒ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None Add a license: None ⓘ

Create repository

- When you have the repository created click the green button *Clone or download*. The URL of your repository will be presented. This URL will be necessary on the next steps.

monroe-institute / yourlanguage-subtitles

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

A translation project for TMI videos

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request

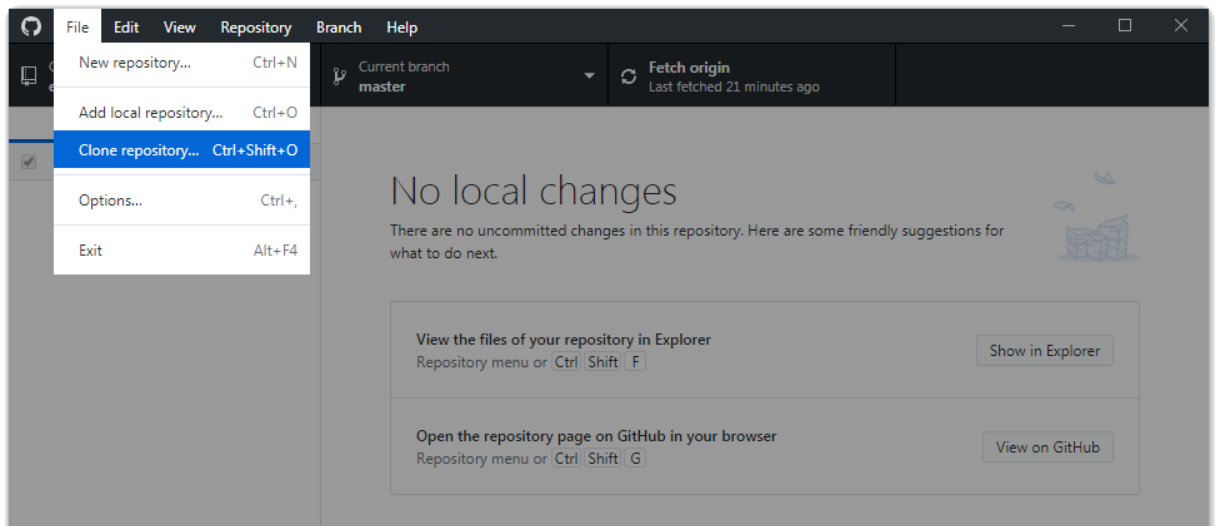
Create new file Upload files Find file Clone or download

Clone with HTTPS ⓘ
Use Git or checkout with SVN using the web URL.
<https://github.com/monroe-institute/yourlanguage-subtitles>

Open in Desktop Download ZIP

This is the end of the first stage of GitHub repository creation procedure.

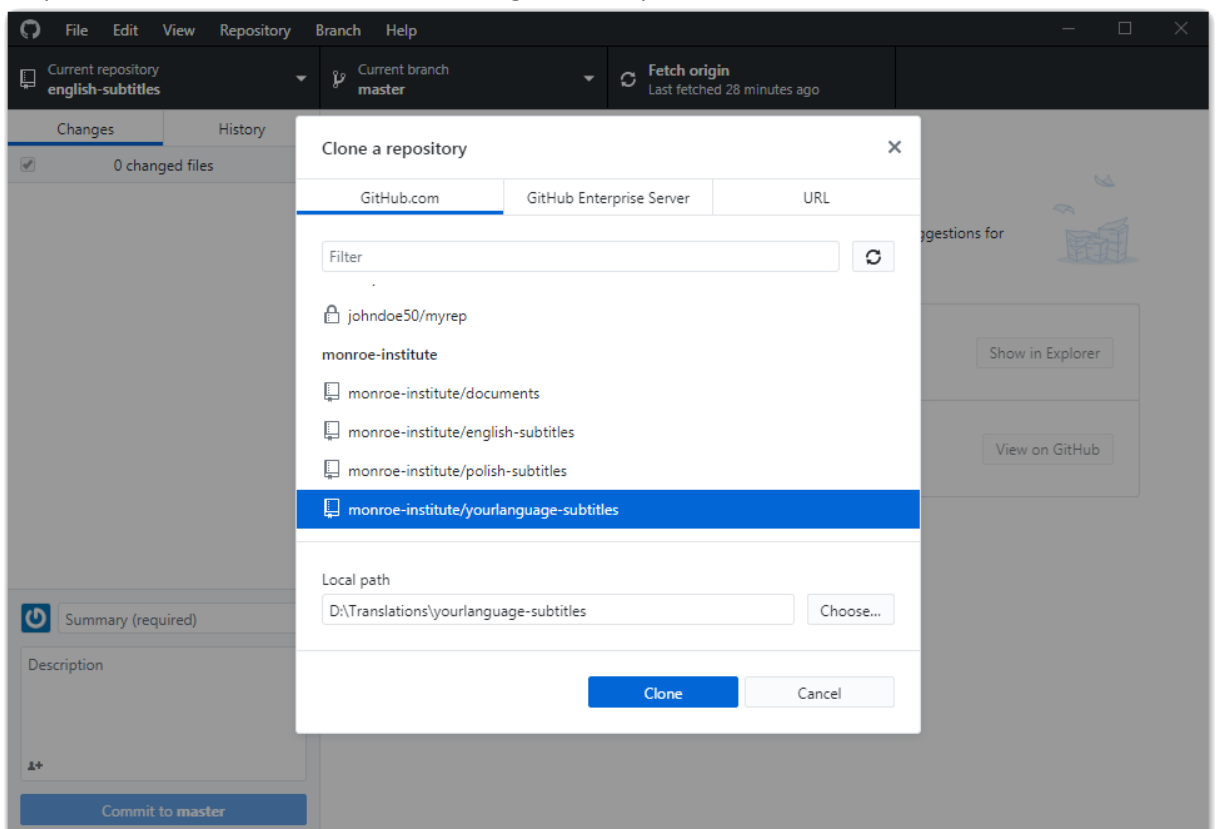
- Download **Github Desktop** application from:
<https://desktop.github.com>
This software works only on Windows and MacOS. If you use Linux choose another software like <https://github.com/shiftkey/desktop/releases>.
- Install *Github Desktop* on your machine.
- Run the *Github Desktop* and clone the GitHub repository you have just created to your machine. Go to menu *File* and then click on *Clone repository...*



You will be prompted to sign in.

9. After signing in go again to menu *File* and then click on *Clone repository...* A dialog box will appear. Select the repository you have just created on GitHub, then select the local path for the project files and click *Clone* button.

The files from the GitHub will be cloned to your local drive. You can add/change/delete files on your local drive and then all these changes will be pushed to GitHub.



10. Now, clone *english-subtitles* repository to your local drive in the same way. This repository contains files with English subtitles to be translated as well as some additional files helpful when setting up the translation project file structure.

11. Now, you will setup the project. This will include making necessary presets so the team members can start with completely prepared environment.

Download **OmegaT** software from:

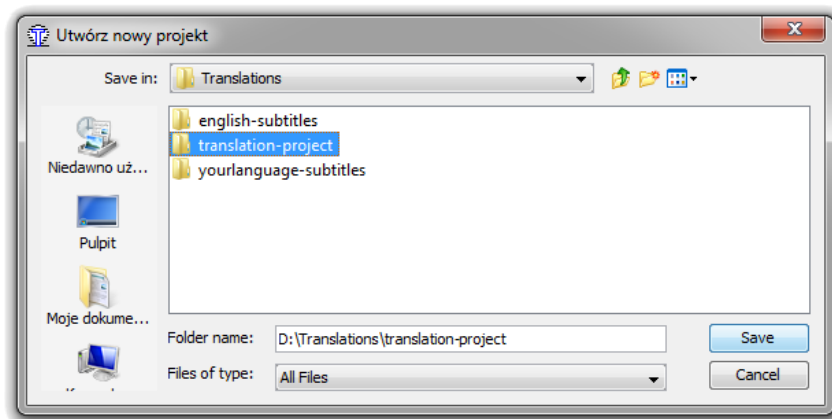
<https://omegat.org/download>

This is CAT software the translators will be working with. You need this software installed to setup the repository correctly for translators.

Choose the version you need depending on the operating system installed on your computer. The *OmegaT* software is a Java application so *Java Runtime Environment (JRE)* should be installed on your computer as well. If you are not sure if you have JRE installed on your machine just download the *OmegaT* installer with JRE. If you still have a problem deciding which version you should download use the *Download selector* link available on the top of the download page.

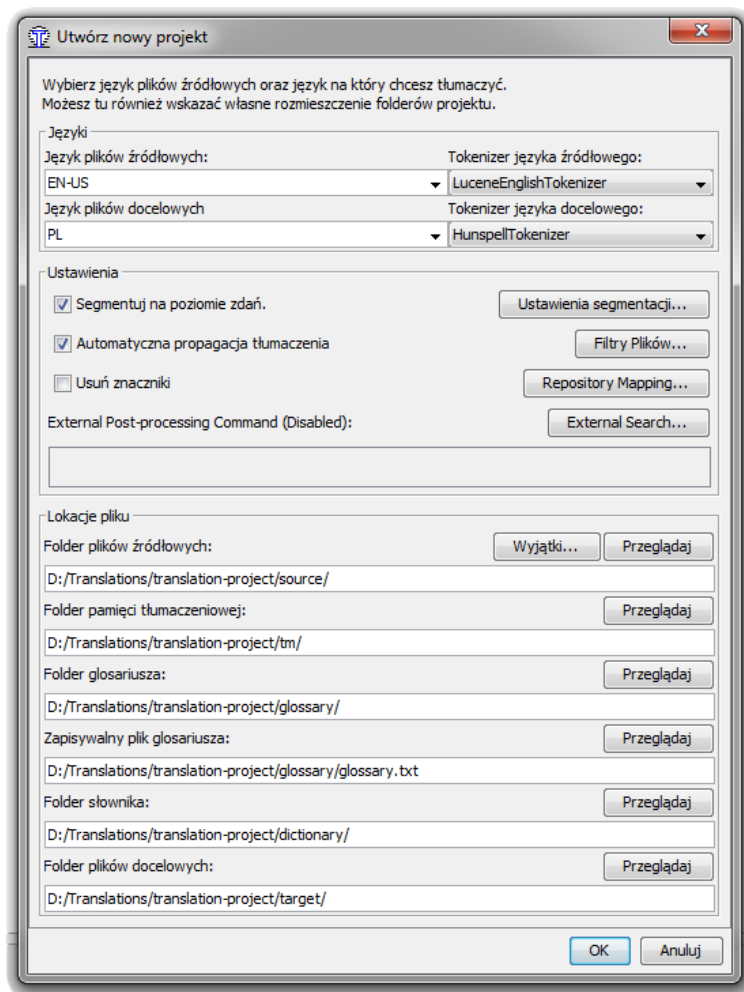
Caution: If you do not use English version of Operating System (especially Windows), you will see mixed content (English + your Operating System language) in some of dialog boxes to be displayed.

12. Install *OmegaT* on your machine.
13. Run *OmegaT*. The program will open with the interface in your operating system default language.
14. Go to menu *Project* and click New.... A standard Windows Explorer window will appear (in your language).

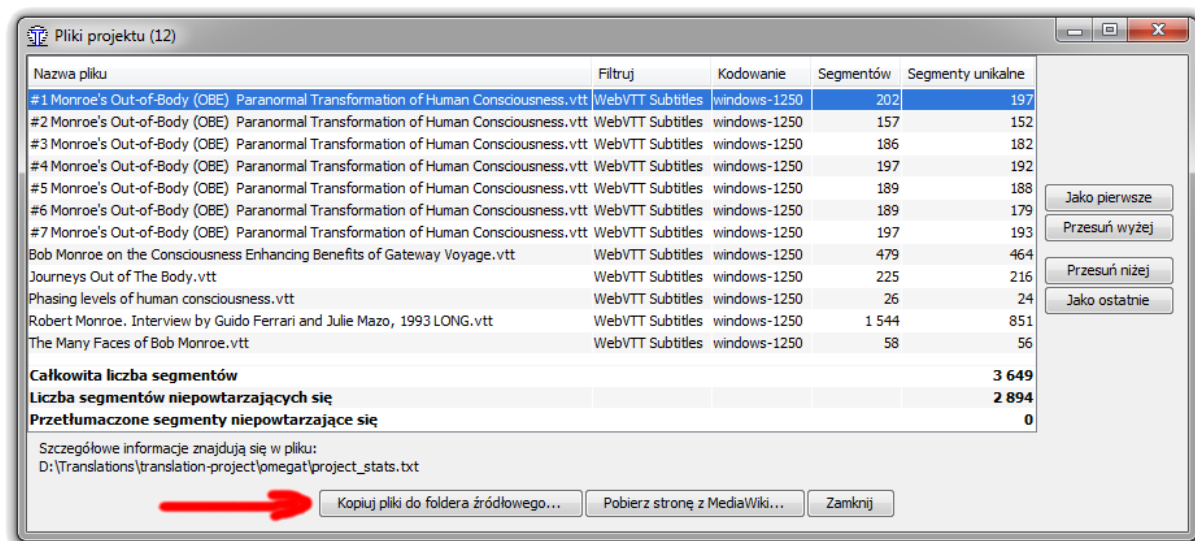


15. Navigate to a location where you want to create the translation project and select (or create) applicable folder. Do not point the same folder as your cloned repositories. Choose name for the project. This will be temporary project, so the folder name is not so important. Click *Save* button.
16. Project properties window will be presented to you. Note the paths to *Dictionaries*, *Glossary* and *Translation Memory*. They will be necessary later. Make any changes if necessary and

click *OK* button (you will see this window in your language).

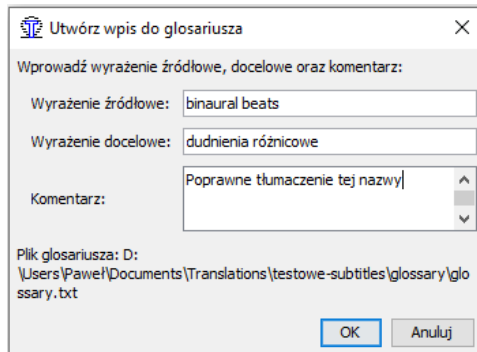


17. Another window will be presented to you. It allows you to manage project files. You can copy the files to be translated to the project (the *Copy files to the source folder* button) These files you will find in previously cloned *english-subtitles* folder inside *source* subfolder. (you will see this window in your language). After copying files close the window like this (with yourfiles).

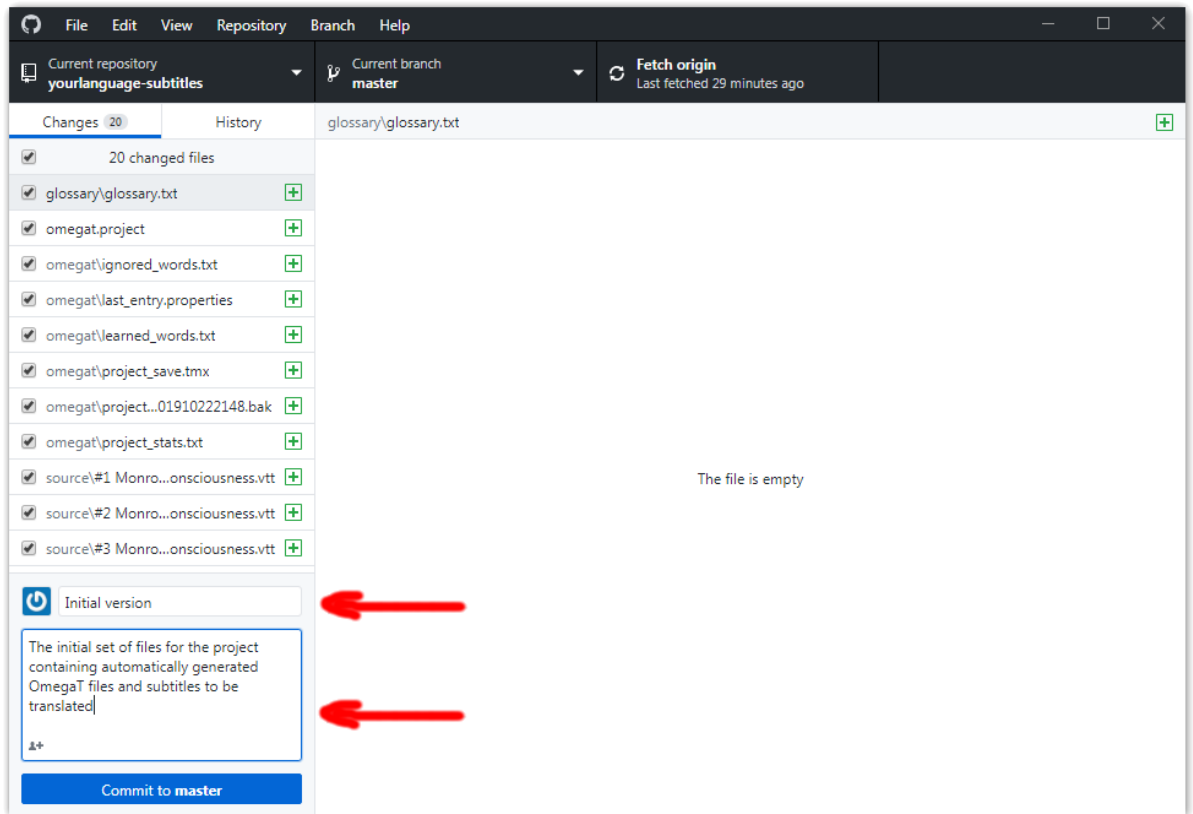


18. The *OmegaT* will create all necessary project files for you, now.

19. Create a glossary entry. You can do this by clicking right mouse button on the glossary area (next to *Editor* window). Click on *New glossary entry* in the context menu and populate all the text fields in a small pop-up window. Click *OK* button then. This will create a properly formatted glossary file. This is a good moment to create initial glossary with entries you consider should be translated in a particular way. This will allow to keep standards of translation of specific phrases or words.

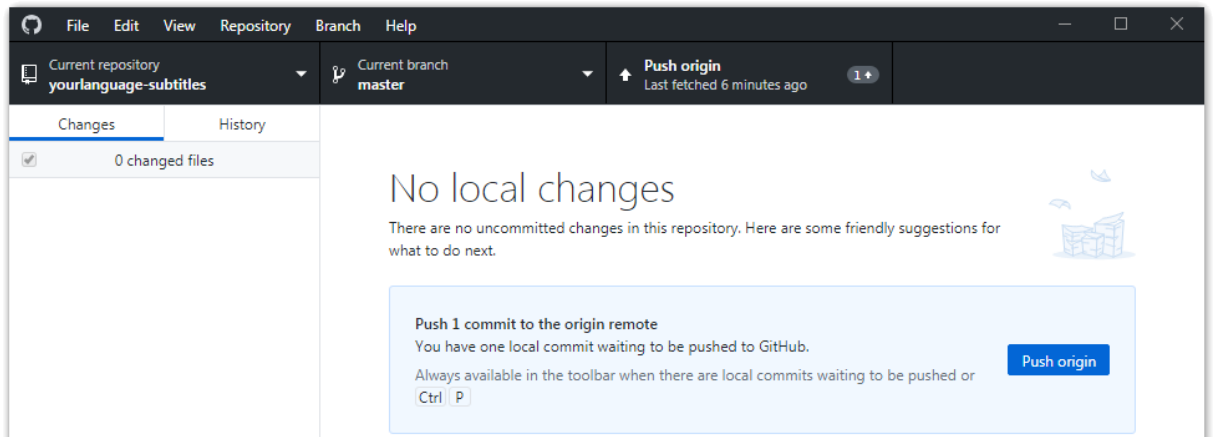


20. Close *OmegaT*.
21. Add Dictionaries. Download a dictionaries you need from: <https://sites.google.com/site/gtonguedict/home/stardict-dictionaries> or <https://tuxor1337.frama.io/firedict/dictionaries.html>. You can also try to find any other locations with StarDict type dictionaries. Unpack them and copy to the dictionary folder, which you will find inside your OmegaT project folder (the one mentioned before).
22. Locate the *project_save.tmx* file (Translation Memory) within *omegat* folder inside your project folder. Copy it to the *tm* folder and rename to **tm1.tmx**. The project contains all necessary files, now.
23. Move all files and folders from your *OmegaT* project into your local version of repository you have created previously on GitHub.
24. Go back to GitHub Desktop application. You will notice that a list of files appeared in left pane. Populate *Summary* and *Description* fields as to reflect the changes have been made.



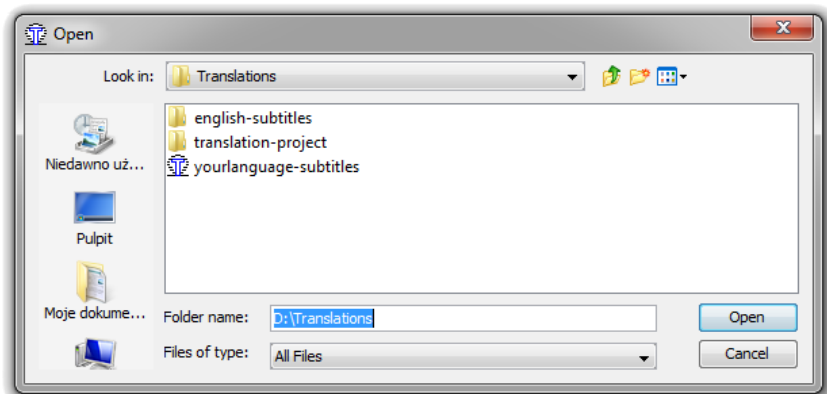
Then click *Commit to master* button.

25. Now, click button labeled *Push origin*. All the changes will be transferred to the GitHub.

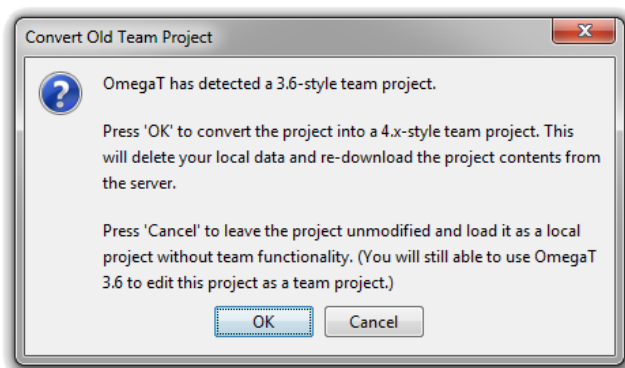


26. Run *OmegaT* again. Go to menu *Project* and click *Open*. A standard Windows Explorer window will appear showing location of the last project. Navigate to your repository which you cloned using *GitHub Desktop* application. You will notice that now your project is named the same as the repository name on GitHub you created. Just click on the project with

OmegaT icon and then click *Open* button.



27. Following information will be displayed. Just confirm it.



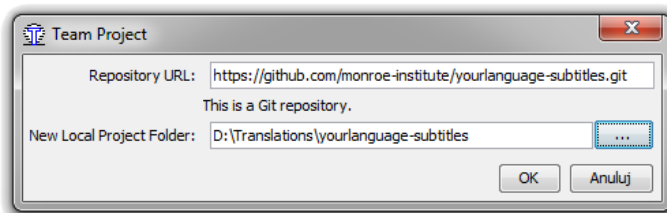
28. Project creation procedure is completed. From this moment on the OmegaT application controls communication with GitHub.

29. Go to menu *Project* and click *Save* and then *Close*.

30. Switch to *GitHub Desktop*. You will following message: *Can't find 'yourlanguage-subtitles'*. That is correct. Click the *Remove* button and clone the repository again to your computer. The access to the repository via GitHub Desktop application will be helpful later when managing project files, especially *Translation Memory*.

31. Now test the project settings:

- Delete your *OmegaT* project from your hard drive.
- Run *OmegaT*.
- From the project menu choose: *Download team file...*
- In a dialog box that appear put appropriate information



The URL you will find as described in point 5.

Caution: While entering *Local Project Folder* path navigate to the parent folder of your project (*Translations* as in the example above) and type the name of the

subfolder for your project manually instead of creating it via *Create New Folder*, for example *yourlanguage-subtitles*.

- e. You will be asked for Authentication details (username and password) then click OK. Enter your GitHub username and password.

That was the last step. The project will be downloaded then. *OmegaT* should open as previously, showing a file for translation. You may check if dictionaries do work and try to translate a segment and commit it then.

Create a team

GitHub allows creating teams to work on projects.

Learn OmegaT

You need to know how *OmegaT* works since you will manage the translation process and deal with new files to be translated, the translation memory and glossaries. You have to know where they reside within the project folder structure and how they are used by *OmegaT*. It is necessary to build up translation memory and glossaries which finally bring faster translations of higher quality.

With the application running and no project open a *Quick Start* tutorial is displayed in the left panel. Read it carefully. Also refer to *Translator's Manual* to the sections *Learn OmegaT* and *Start translating with OmegaT*.

Build up a translation memory

At the beginning the translation memory is empty because no previous translations do exist. Having translated at least one file the translation memory will be built. You have to control the translation memory files (*.tmx* files) occurring in *omegat* folder on GitHub. You will need to merge and copy them to *tm* folder to provide the most recent and extensive translation memory for the team. The helpful tools for this can be: *TMXMerger* and *TMXCleaner* from <https://omegat.org/resources>