

# Εργαστηριακή άσκηση 3: Version Control

## ΑΣΚΗΣΗ:

### Github account

Ανοίξτε λογαριασμό στο github αν δεν έχετε.

### Εισαγωγή

Το GitHub είναι μια πλατφόρμα φιλοξενίας κώδικα για έλεγχο έκδοσης και συνεργασία. Σας επιτρέπει να εργάζεστε μαζί σε έργα από οπουδήποτε.

Αυτό το εργαστήριο σας διδάσκει βασικά στοιχεία του GitHub όπως αποθετήρια, διακλαδώσεις, δεσμεύσεις και αιτήματα έλξης. Θα δημιουργήσετε το δικό σας αποθετήριο του Hello World και θα μάθετε τη ροή εργασίας αιτημάτων έλξης του GitHub, έναν δημοφιλή τρόπο δημιουργίας και ελέγχου κώδικα.

Σε αυτόν τον οδηγό γρήγορης εκκίνησης, θα:

- Δημιουργήστε και χρησιμοποιήστε ένα αποθετήριο
- Ξεκινήστε και διαχειριστείτε ένα νέο branch
- Κάντε αλλαγές σε ένα αρχείο και ωθήστε τις στο GitHub ως commits
- Open and merge a pull request

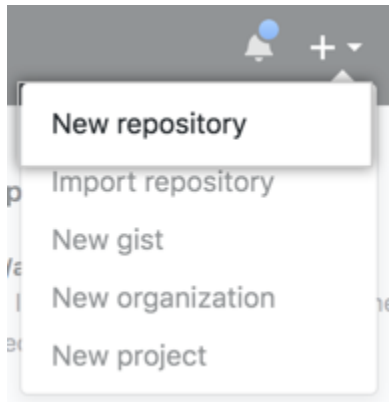
Για να ολοκληρώσετε αυτό το εργαστήριο, χρειάζεστε έναν λογαριασμό GitHub και πρόσβαση στο Διαδίκτυο. Δεν χρειάζεται να ξέρετε πώς να γραφете κωδικα, να χρησιμοποιείτε τη γραμμή εντολών ή να εγκαταστήσετε το Git (το λογισμικό ελέγχου έκδοσης στο οποίο είναι ενσωματωμένο το GitHub).

### Δημιουργία αποθετηρίου (repository)

Ένα αποθετήριο χρησιμοποιείται συνήθως για την οργάνωση ενός μεμονωμένου έργου. Τα αποθετήρια μπορούν να περιέχουν φακέλους και αρχεία, εικόνες, βίντεο, υπολογιστικά φύλλα και σύνολα δεδομένων -- οτιδήποτε χρειάζεται το έργο σας. Συχνά, τα αποθετήρια περιλαμβάνουν ένα αρχείο *README*, ένα αρχείο με πληροφορίες για το έργο σας. Τα αρχεία *README* είναι γραμμένα στη γλώσσα Markdown απλού κειμένου. Μπορείτε να χρησιμοποιήσετε αυτό (<https://www.markdownguide.org/cheat-sheet/>) για να ξεκινήσετε με τη σύνταξη Markdown. Το GitHub σας επιτρέπει να προσθέτετε ένα αρχείο *README* την ίδια στιγμή που δημιουργείτε το νέο σας αποθετήριο. Το GitHub προσφέρει επίσης άλλες κοινές επιλογές, όπως ένα αρχείο άδειας χρήσης, αλλά δεν χρειάζεται να επιλέξετε καμία από αυτές τώρα.

Το `hello-world` αποθετήριο σας μπορεί να είναι ένα μέρος όπου αποθηκεύετε ιδέες, πόρους ή ακόμα και μοιράζεστε και συζητάτε πράγματα με άλλους.

1. Στην επάνω δεξιά γωνία οποιασδήποτε σελίδας, χρησιμοποιήστε το αναπτυσσόμενο μενού και επιλέξτε **New repository** .




- αίσιο **Όνομα αποθετηρίου** , πληκτρολογήστε `hello-world`.
- Στο πλαίσιο **Περιγραφή** , γράψτε μια σύντομη περιγραφή.
- Επιλέξτε **Προσθήκη αρχείου README** .
- Επιλέξτε εάν το αποθετήριο σας θα είναι **Δημόσιο** ή **Ιδιωτικό** .
- Κάντε κλικ στην επιλογή **Δημιουργία αποθετηρίου** .

## Create a new repository

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

Owner \*

 ctranoris

Repository name \*

hello-world 

Great repository names are short and memorable. Need inspiration? How about [reimagined-octo-funicular?](#)

Description (optional)

☐



Public

Anyone on the internet can see this repository. You choose who can commit.

☒



Private

You choose who can see and commit to this repository.

Initialize this repository with:

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

☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

☐ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

This will set  `main` as the default branch. Change the default name in your [settings](#).



You are creating a private repository in your personal account.

Create repository

## Δημιουργία branch (διακλάδωση)

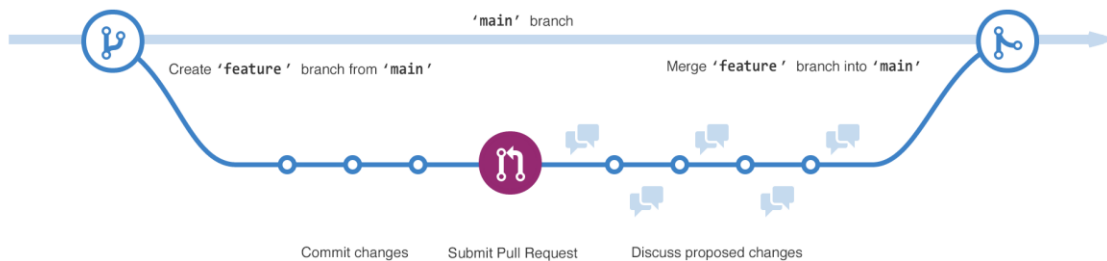
Η διακλάδωση σάς επιτρέπει να έχετε διαφορετικές εκδόσεις ενός αποθετηρίου ταυτόχρονα.

Από προεπιλογή, το αποθετήριο σας έχει έναν κλάδο με το όνομα `main` που θεωρείται ο οριστικός κλάδος. Μπορείτε να δημιουργήσετε επιπλέον διακλαδώσεις του `main` στο αποθετήριο σας. Μπορείτε να χρησιμοποιήσετε κλάδους για να έχετε διαφορετικές εκδόσεις ενός έργου ταυτόχρονα. Αυτό είναι χρήσιμο όταν θέλετε να προσθέσετε νέες δυνατότητες σε ένα έργο χωρίς να αλλάξετε τον κυριο κώδικα. Η εργασία που κανετε σε διαφορετικούς κλάδους δεν θα εμφανιστεί στον κύριο κλάδο `main` μέχρι να τον συγχωνεύσετε. Μπορείτε να χρησιμοποιήσετε κλάδους για να πειραματιστείτε και να κάνετε αλλαγές πριν τις δεσμεύσετε στο `main`.

Όταν δημιουργείτε διακλάδωση από τον `main` κλάδο, δημιουργείτε ένα αντίγραφο ή στιγμιότυπο, `main`όπως ήταν εκείνη τη στιγμή. Εάν κάποιος άλλος έκανε αλλαγές στο `main` κλάδο ενώ εργαζόσασταν στη διακλάδωση σας, μπορείτε να παρείτε αυτές τις ενημερώσεις.

Αυτό το διάγραμμα δείχνει:

- Το `main` κλάδο
- Ένα νέο κλάδο ονομάζεται `feature`
- Τη διαδρομή που έκανε `feature` χρειάζεται πριν συγχωνευθεί με το `main`



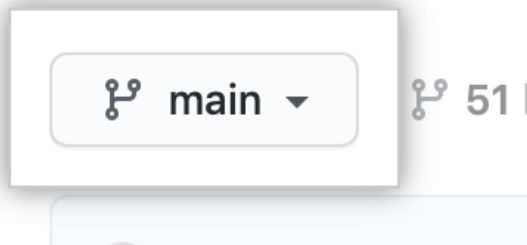
Έχετε αποθηκεύσει ποτέ διαφορετικές εκδόσεις ενός αρχείου; Κάτι όπως:

- `story.txt`
- `story-edit.txt`
- `story-edit-reviewed.txt`

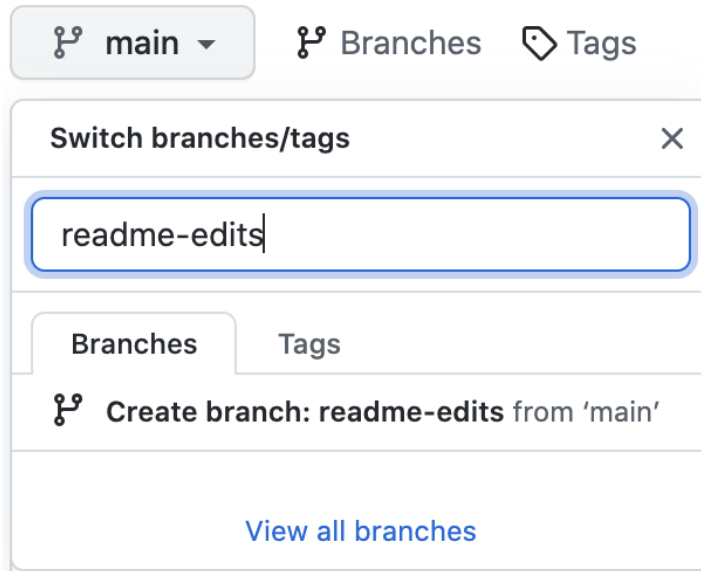
Οι διακλαδώσεις επιτυγχάνουν παρόμοιους στόχους στα αποθετήρια GitHub.

Δημιουργήστε ένα κλάδο

1. Κάντε κλικ στην καρτέλα **Code** του `hello-world` του αποθετηρίου σας.
2. Κάντε κλικ στο αναπτυσσόμενο μενού στην κορυφή της λίστας αρχείων που λέει **κύριο**.



3. Πληκτρολογήστε ένα όνομα κλάδου, `readme-edits`, στο πλαίσιο κειμένου.
4. Κάντε κλικ στο **Create branch: readme-edits from main**.



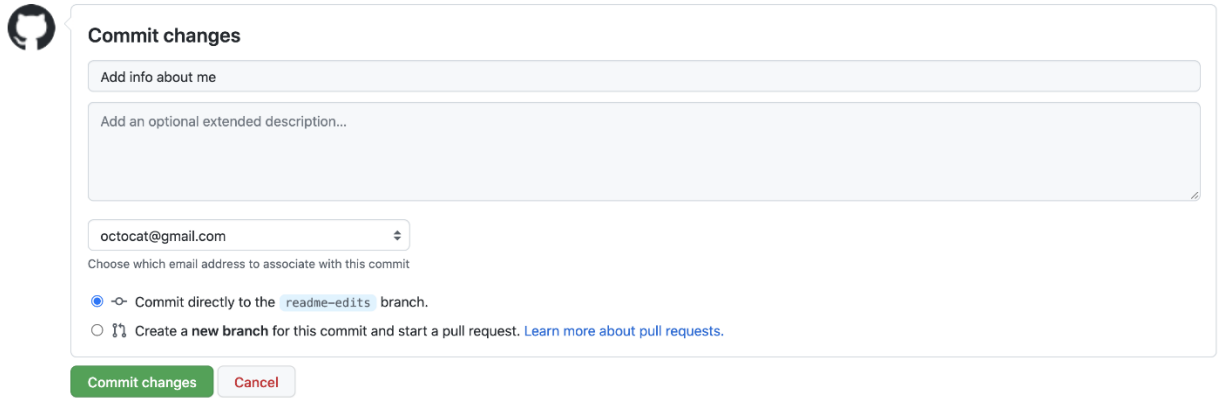
Τώρα έχετε δύο κλάδους `main` και `readme-edits`. Αυτή τη στιγμή, φαίνονται ακριβώς τα ίδια. Στη συνέχεια, θα προσθέσετε αλλαγές στο νέο κλάδο.

### Πραγματοποίηση και δέσμευση αλλαγών (Making and committing changes)

Όταν δημιουργήσατε ένα νέο κλαδο στο προηγούμενο βήμα, το GitHub σάς έφερε στην σελίδα του νέου σας κλάδου `readme-edits`, η οποία είναι αντίγραφο του `main`.

Μπορείτε να κάνετε και να αποθηκεύσετε αλλαγές στα αρχεία του αποθετηρίου σας. Στο GitHub, οι αποθηκευμένες αλλαγές ονομάζονται `commits` (δεσμεύσεις). Κάθε `commit` έχει ένα σχετικό μήνυμα δέσμευσης, το οποίο είναι μια περιγραφή που εξηγεί γιατί έγινε μια συγκεκριμένη αλλαγή. Τα μηνύματα `commit` καταγράφουν το ιστορικό των αλλαγών σας, έτσι ώστε οι άλλοι συνεισφέροντες να μπορούν να καταλάβουν τι κάνατε και γιατί.

1. Κάτω από τον `readme-edits` κλάδο που δημιουργήσατε, κάντε κλικ στο αρχείο `README.md`.
2. Κάντε κλικ για να επεξεργαστείτε το αρχείο.
2. Στον συντάκτη, γράψτε λίγο για τον εαυτό σας. Δοκιμάστε να χρησιμοποιήσετε διαφορετικά στοιχεία Markdown.
3. Στο πλαίσιο **Commit Changes**, γράψτε ένα μήνυμα δέσμευσης που περιγράφει τις αλλαγές σας.
4. Κάντε κλικ στην επιλογή **Commit Changes**.

The image shows the GitHub 'Commit changes' dialog box. At the top left is the GitHub Octocat logo. The title 'Commit changes' is in bold. Below it is a text input field labeled 'Add info about me'. Underneath is a larger text area labeled 'Add an optional extended description...'. Below that is a dropdown menu showing 'octocat@gmail.com' with a small arrow icon to its right. Below the dropdown is the text 'Choose which email address to associate with this commit'. There are two radio button options: the first is selected and labeled 'Commit directly to the `readme-edits` branch.', and the second is labeled 'Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)'. At the bottom are two buttons: a green 'Commit changes' button and a red 'Cancel' button.

Αυτές οι αλλαγές θα γίνουν μόνο στο αρχείο README στο `readme-edits` κλάδο σας, επομένως τώρα αυτός ο κλάδος περιέχει περιεχόμενο που διαφέρει από το `main`.

## Άνοιγμα αιτήματος έλξης (pull request)

Τώρα που έχετε αλλαγές σε έναν κλάδο του `main`, μπορείτε να ανοίξετε ένα pull request.

Τα pull request είναι η καρδιά της συνεργασίας στο GitHub. Όταν ανοίγετε ένα pull request, προτείνετε τις αλλαγές σας και ζητάτε από κάποιον να ελέγξει και να τραβήξει τη συνεισφορά σας και να τις συγχωνεύσει στον κλάδο του. Τα pull request εμφανίζουν διαφορές ή διαφορές του περιεχομένου και από τους δύο κλάδους. Οι αλλαγές, οι προσθήκες και οι αφαιρέσεις εμφανίζονται με διαφορετικά χρώματα.

Μόλις κάνετε commit, μπορείτε να ανοίξετε ένα αίτημα έλξης και να ξεκινήσετε μια συζήτηση, ακόμη και πριν ολοκληρωθεί ο κώδικας.

Χρησιμοποιώντας τη δυνατότητα @mention του GitHub στο μήνυμα αιτήματος έλξης, μπορείτε να ζητήσετε σχόλια από συγκεκριμένα άτομα ή ομάδες.

Μπορείτε ακόμη να ανοίξετε αιτήματα έλξης στο δικό σας αποθετήριο και να τα συγχωνεύσετε μόνοι σας. Είναι ένας πολύ καλός τρόπος για να μάθετε τη ροή του GitHub πριν εργαστείτε σε μεγαλύτερα έργα.

1. Κάντε κλικ στην καρτέλα **Pull requests** `hello-world` του αποθετηρίου σας.
2. Κάντε κλικ **στο New pull request**
3. Στο πλαίσιο **Example Comparisons**, επιλέξτε τον κλάδο που δημιουργήσατε, `readme-edits`, για σύγκριση `main` (το πρωτότυπο).
4. Εξετάστε τις αλλαγές σας στις διαφορές στη σελίδα Σύγκριση, βεβαιωθείτε ότι είναι αυτό που θέλετε να υποβάλετε.

Showing 1 changed file with 3 additions and 3 deletions.

Split Unified

```
6 README.md
@@ -1,3 +1,3 @@
1 - # test-area-2
2 - edit1
3 - edit2
1 + # About me
2 +
3 + My name is Mona Lisa.
```

5. Κάντε κλικ στην επιλογή **Create pull request**.
6. Δώστε στο αίτημα έλξης έναν τίτλο και γράψτε μια σύντομη περιγραφή των αλλαγών σας. Μπορείτε να συμπεριλάβετε emoji και να κάνετε μεταφορά και απόθεση εικόνων και gif.
7. Προαιρετικά, στα δεξιά του τίτλου και της περιγραφής σας, κάντε κλικ στο επόμενο στοιχείο **Reviewers, Assignees, Labels, Projects, or Milestone** για να προσθέσετε οποιαδήποτε από αυτές τις επιλογές στο αίτημα έλξης σας. Δεν χρειάζεται να προσθέσετε κανένα ακόμη, αλλά αυτές οι επιλογές προσφέρουν διαφορετικούς τρόπους συνεργασίας χρησιμοποιώντας αιτήματα έλξης.
8. Κάντε κλικ στην επιλογή **Create pull request**.

Οι συνεργάτες σας μπορούν πλέον να ελέγχουν τις αλλαγές σας και να κάνουν προτάσεις.

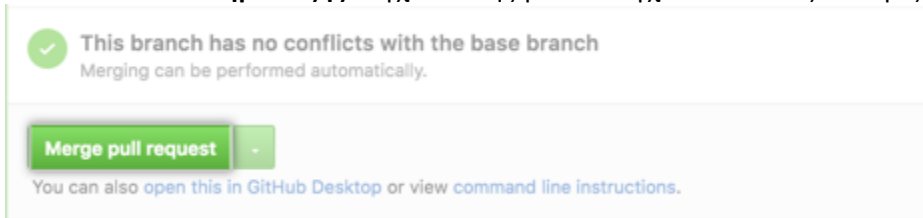
## Συγχώνευση του αιτήματος έλξης σας (Merging your pull request)

Σε αυτό το τελευταίο βήμα, θα συγχωνεύσετε τον `readme-edits` κλάδο σας στον `main` κλάδο. Αφού συγχωνεύσετε το αίτημα έλξης, οι αλλαγές στο `readme-edits` κλάδο σας θα ενσωματωθούν στο `main`.

Μερικές φορές, ένα pull request μπορεί να εισάγει αλλαγές στον κώδικα που έρχονται σε αντίθεση με τον υπάρχοντα κώδικα στο `main`. Εάν υπάρχουν διενέξεις (conflicts), το GitHub θα σας ειδοποιήσει για τον κώδικα που βρίσκεται σε διένεξη και θα αποτρέψει τη συγχώνευση μέχρι να επιλυθούν οι διενέξεις. Μπορείτε να κάνετε ένα commit που επιλύει τις διενέξεις ή να χρησιμοποιήσετε σχόλια στο αίτημα έλξης για να συζητήσετε τις διενέξεις με τα μέλη της ομάδας σας.

Στο παράδειγμα μας, δεν θα πρέπει να έχετε διενέξεις, επομένως είστε έτοιμοι να συγχωνεύσετε τον κλάδο σας στον κύριο κλάδο.

1. Κάντε κλικ στο **αίτημα έλξης** συγχώνευσης για να συγχωνεύσετε τις αλλαγές στο `main`.



2. Κάντε κλικ **στην Επιβεβαίωση συγχώνευσης Confirm merge**. Θα λάβετε ένα μήνυμα ότι το αίτημα συγχωνεύτηκε επιτυχώς και το αίτημα έκλεισε.
3. Κάντε κλικ στην επιλογή **Διαγραφή κλάδου Delete branch**. Τώρα που το αίτημα έλξης έχει συγχωνευθεί και οι αλλαγές σας είναι ενεργοποιημένες `main`, μπορείτε να διαγράψετε με ασφάλεια τον `readme-edits` κλάδο. Εάν θέλετε να κάνετε περισσότερες αλλαγές στο έργο σας, μπορείτε πάντα να δημιουργήσετε έναν νέο κλάδο και να επαναλάβετε αυτή τη διαδικασία.

**Password θα χρειαστεί ένα Github Personal Access Token!**

<https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>



- GitHub Apps
- OAuth Apps
- Personal access tokens

## New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

### Note

lab1\_token

Note has already been taken

30 days

The token will expire on Fri, May 6 2022

### Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input checked="" type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input checked="" type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys

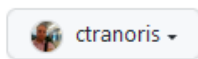
Φυλάξτε το **Personal Access Token!**

## Create a new repository

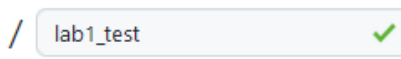
### Create a new repository

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

Owner \*



Repository name \*



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

Description (optional)

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

**Initialize this repository with:**

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

☐ **Add a README file**


This is where you can write a long description for your project. [Learn more](#).

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more](#).

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more](#).

 You are creating a public repository in your personal account.

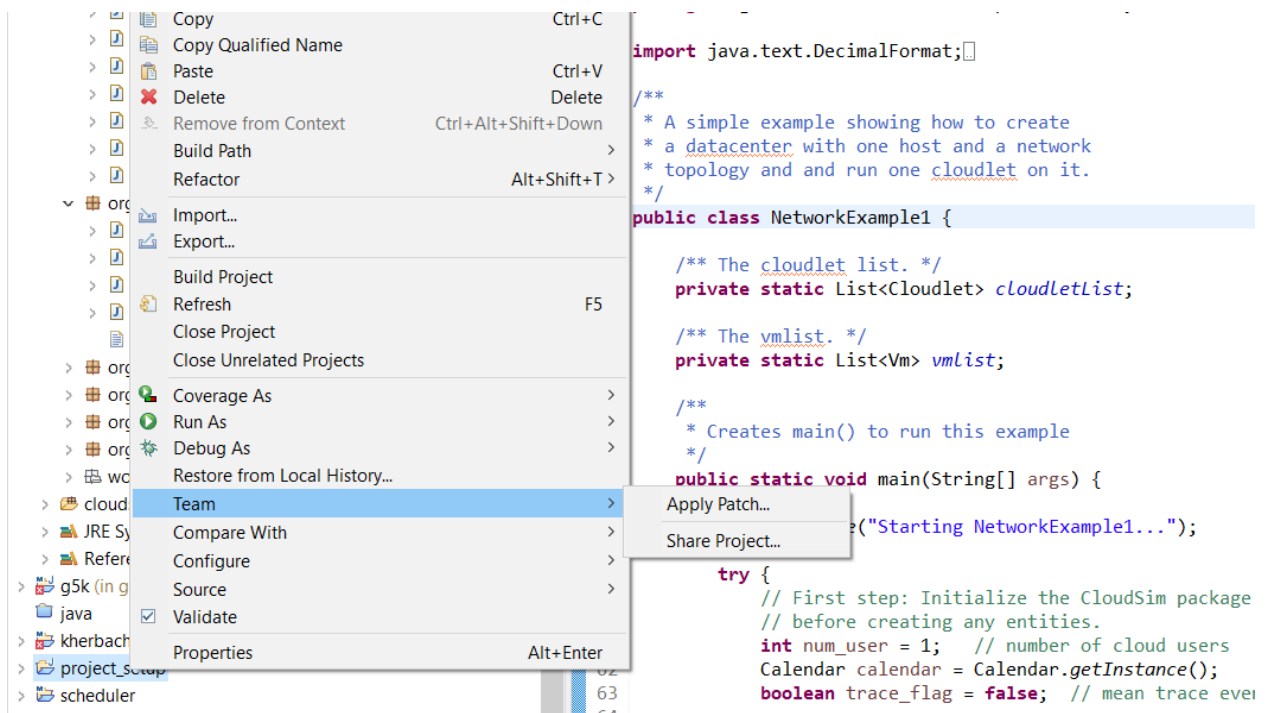
Create repository

## Export Eclipse projects to GitHub

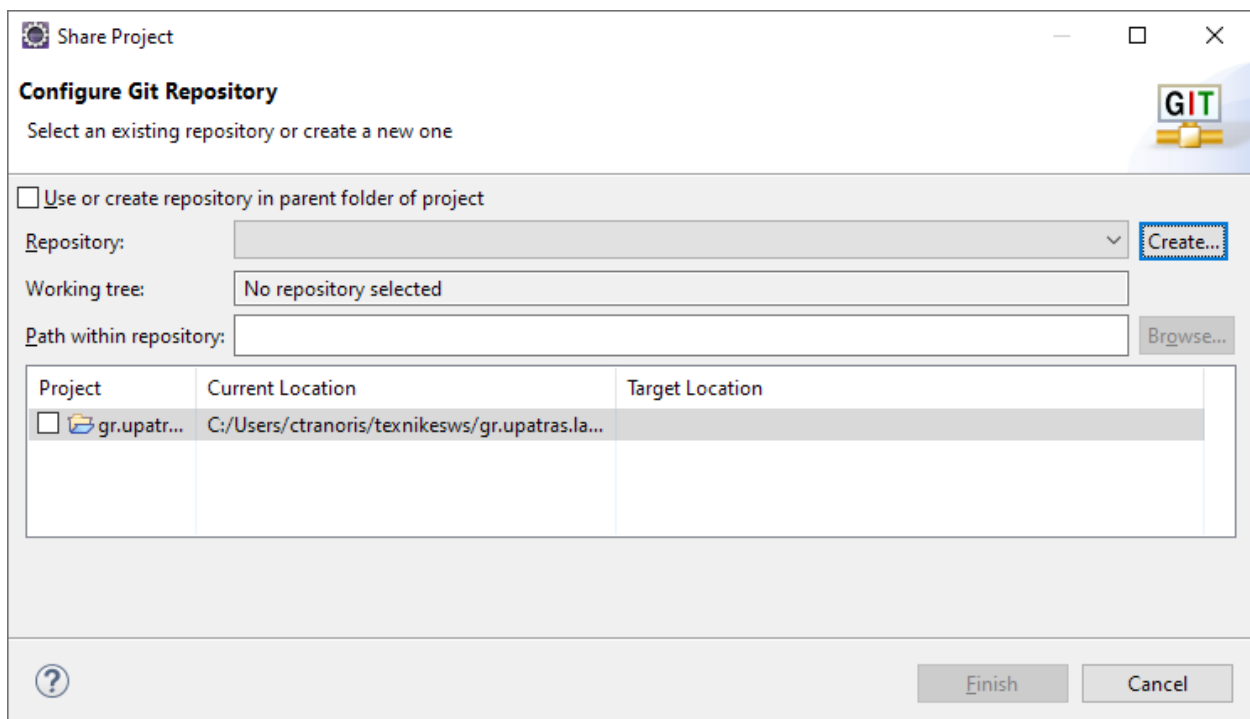
Σε αυτό το μέρος, θα δούμε πώς να προωθήσετε ένα υπάρχον έργο στο GitHub χρησιμοποιώντας το Eclipse IDE.

Βήμα-βήμα Υλοποίηση

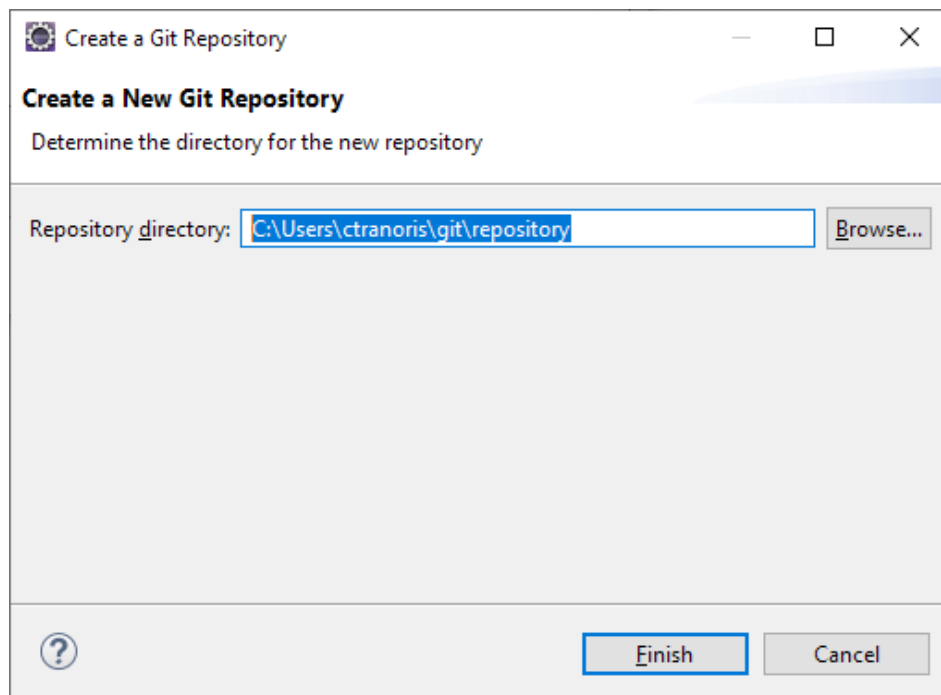
Βήμα 1: Ανοίξτε το Eclipse IDE και κάντε δεξί κλικ στο έργο που θέλετε να προωθήσετε και μεταβείτε στο έργο Team->share.



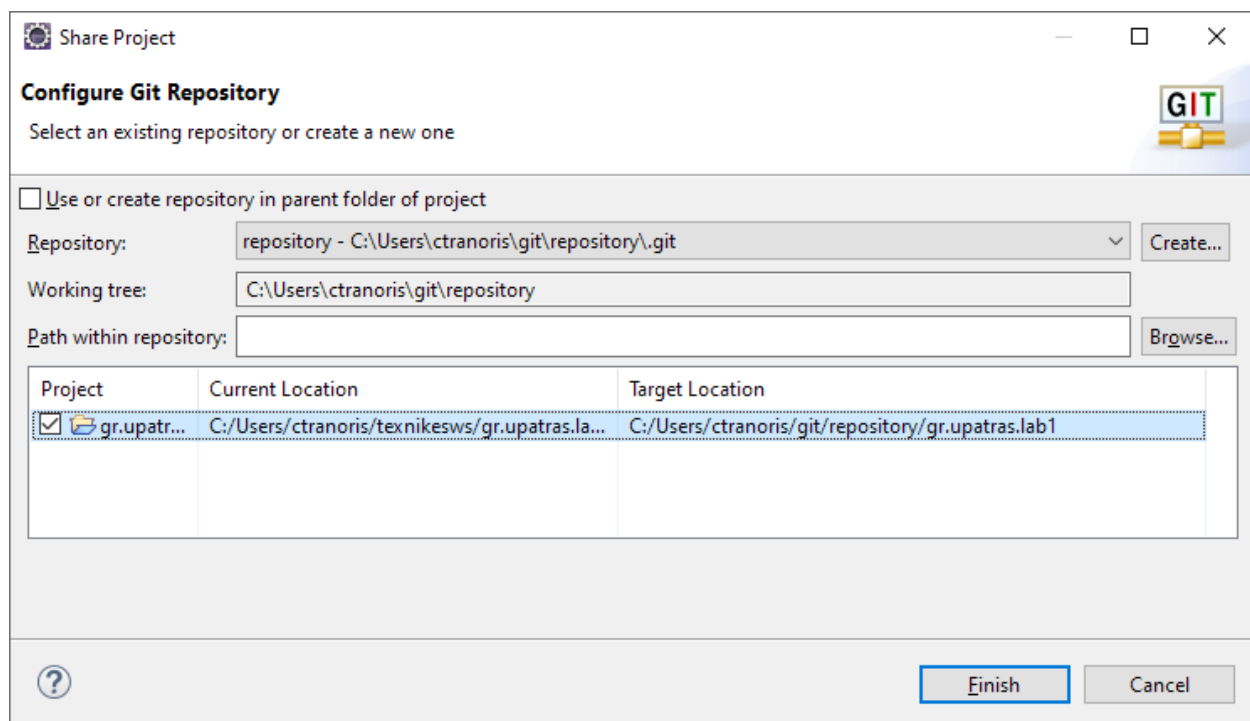
Βήμα 2: Θα προσθέσει το έργο στο δεδομένο αποθετήριο όπως φαίνεται παρακάτω:



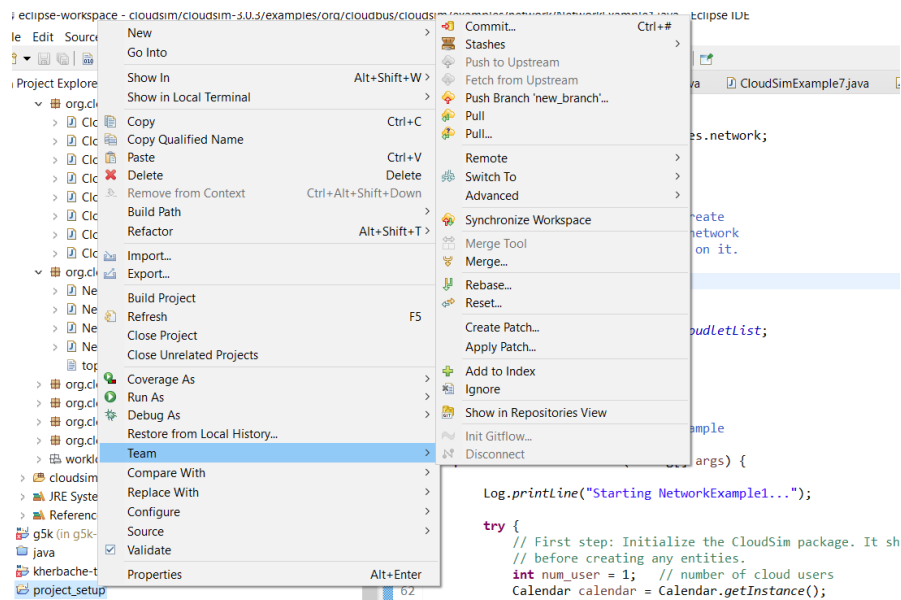
Πατήστε Create



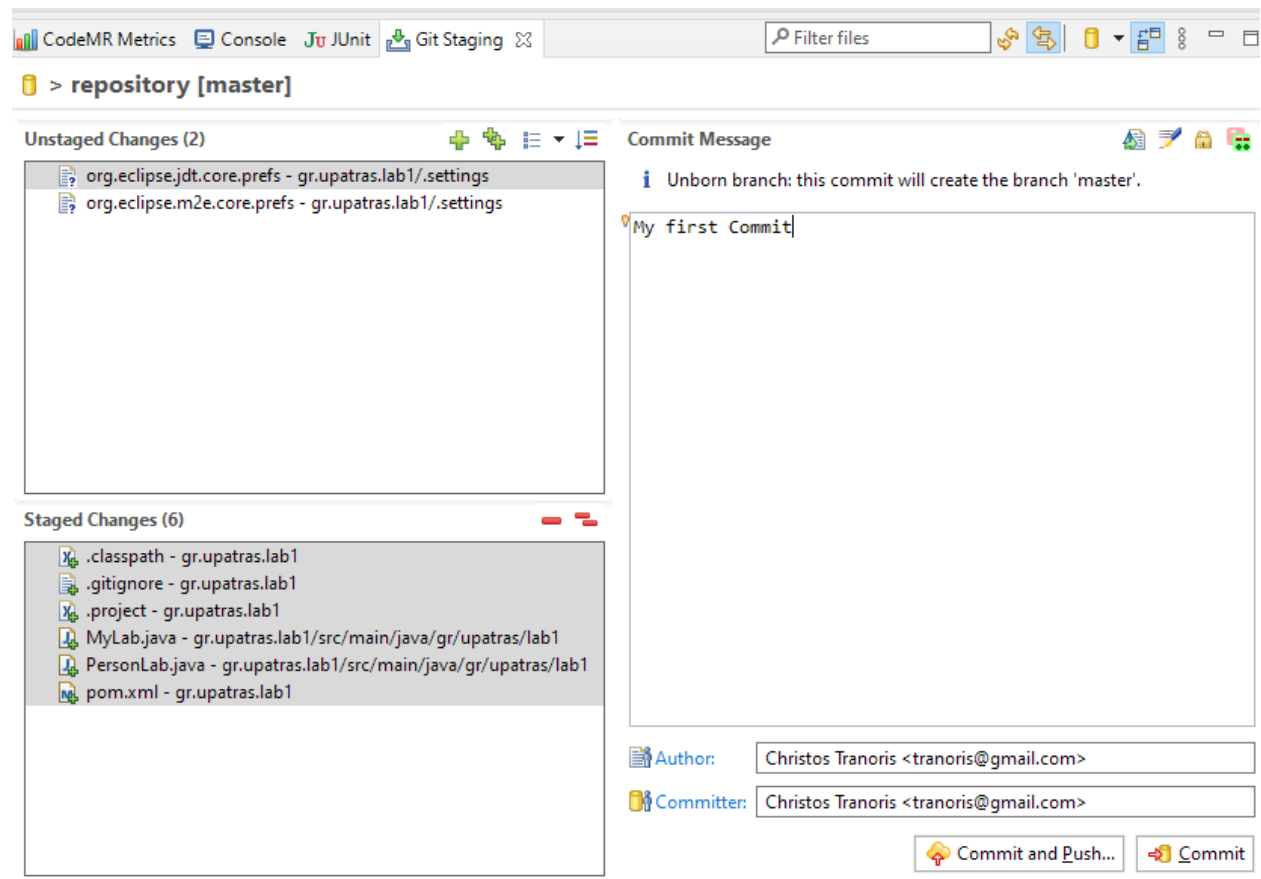
## Finish



**Βήμα 3:** Κάντε ξανά δεξί κλικ στο έργο και μεταβείτε στο Team->commit.

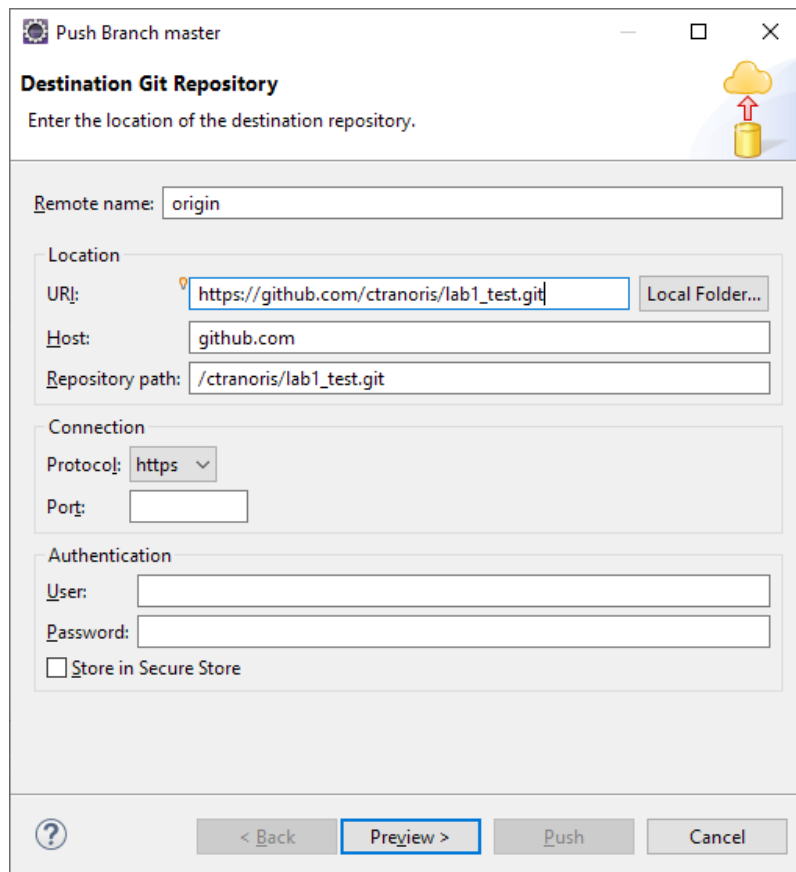


**Βήμα 4: Σύρετε και αποθέστε τα αρχεία που θέλετε να δεσμεύσετε από Αμετάβλητες αλλαγές σε Σταδιακές Αλλαγές. (Unchanged Changes to Staged Changes.)**



Βήμα 5: Γράψτε το μήνυμα στο “Commit Message” and click “Commit and Push”.

Στο παράθυρο:



Push Branch master

**Destination Git Repository**

Enter the location of the destination repository.

Remote name: origin

Location

URI:  Local Folder...

Host:

Repository path:

Connection

Protocol:

Port:

Authentication

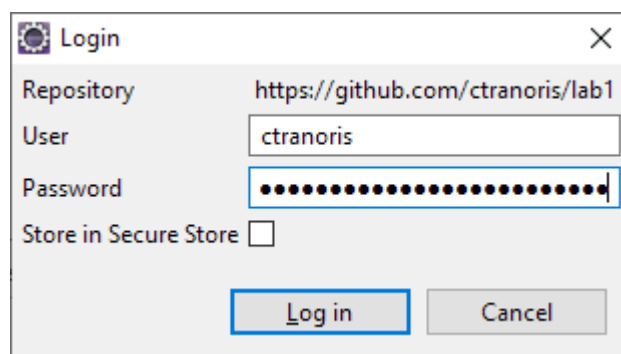
User:

Password:

☐ Store in Secure Store

? < Back Preview > Push Cancel

**Password θα χρειαστεί το Github Personal Access Token!**



Login

Repository

User

Password

Store in Secure Store ☐

Log in Cancel

Push Branch master

**Push to branch in remote**

Select a remote and the name the branch should have in the remote.

Source:

master

191eb35 My first Commit

Destination:

Remote: origin: https://github.com/ctranoris/lab1\_test.git

Branch: master

☒ Configure upstream for push and pull

When pulling: Merge

☐ Force overwrite branch in remote if it exists and has diverged

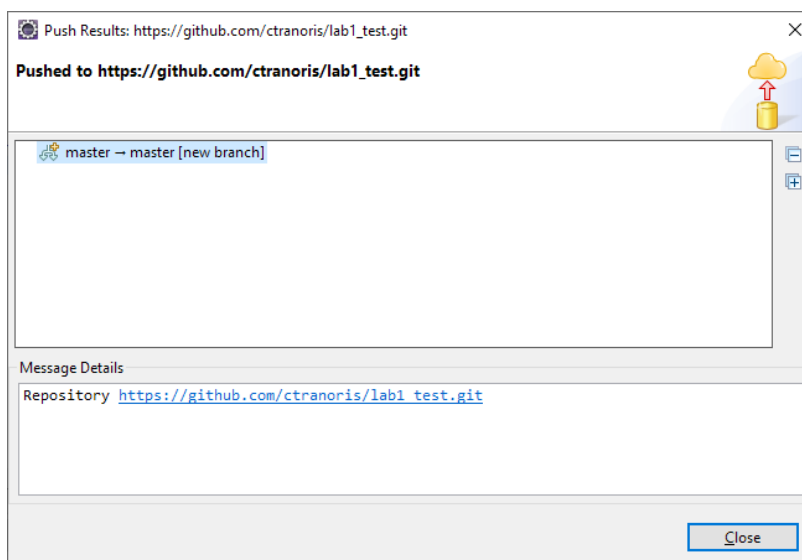
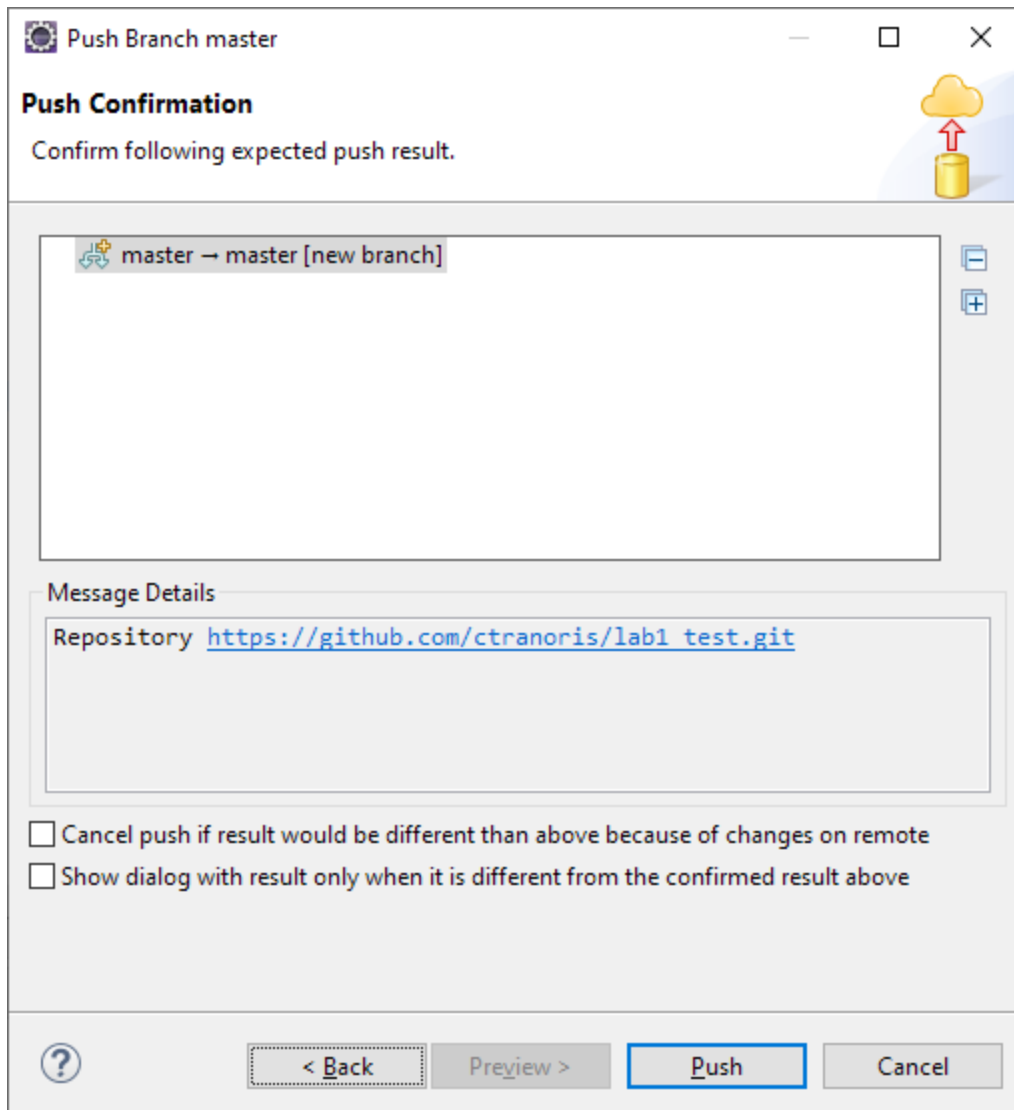
Show [advanced push](#) dialog

< Back

Preview >

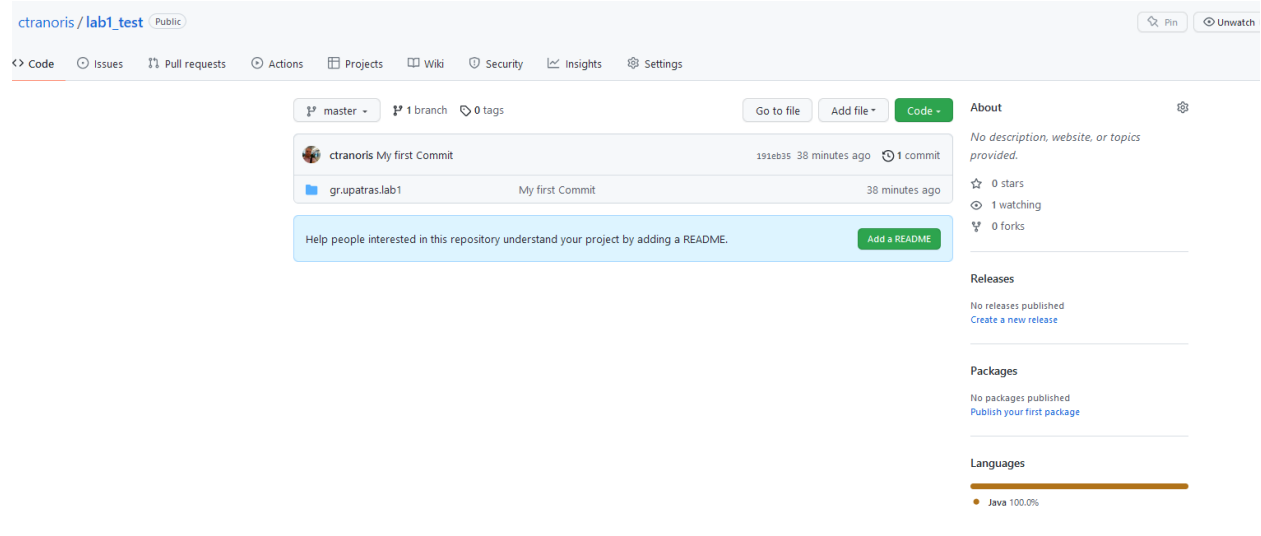
Push

Cancel





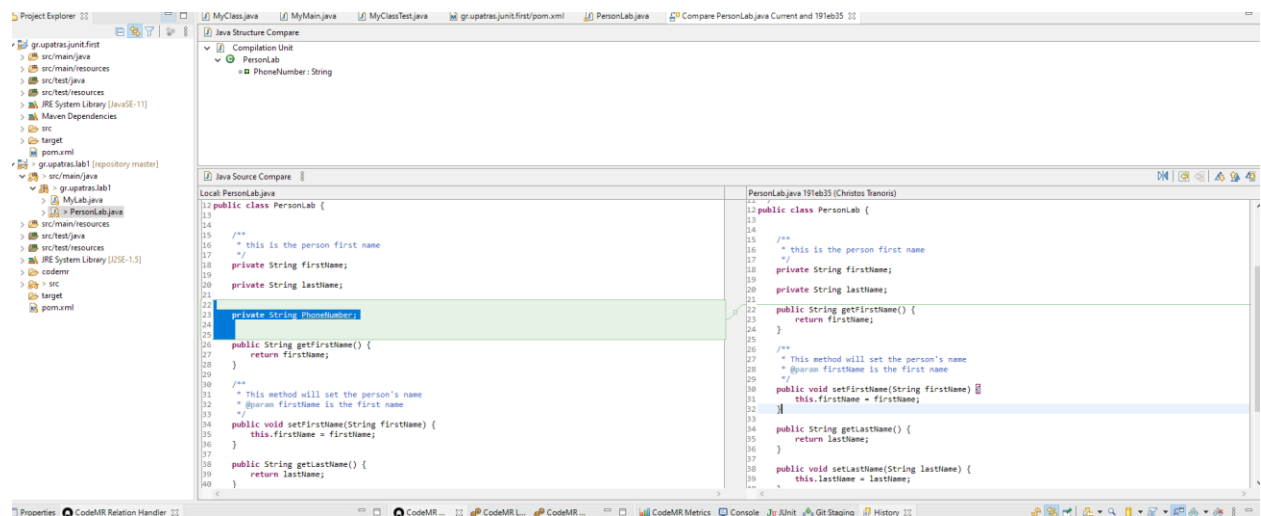
## The new branch is created on GitHub



## Κάνετε μικρές αλλαγές

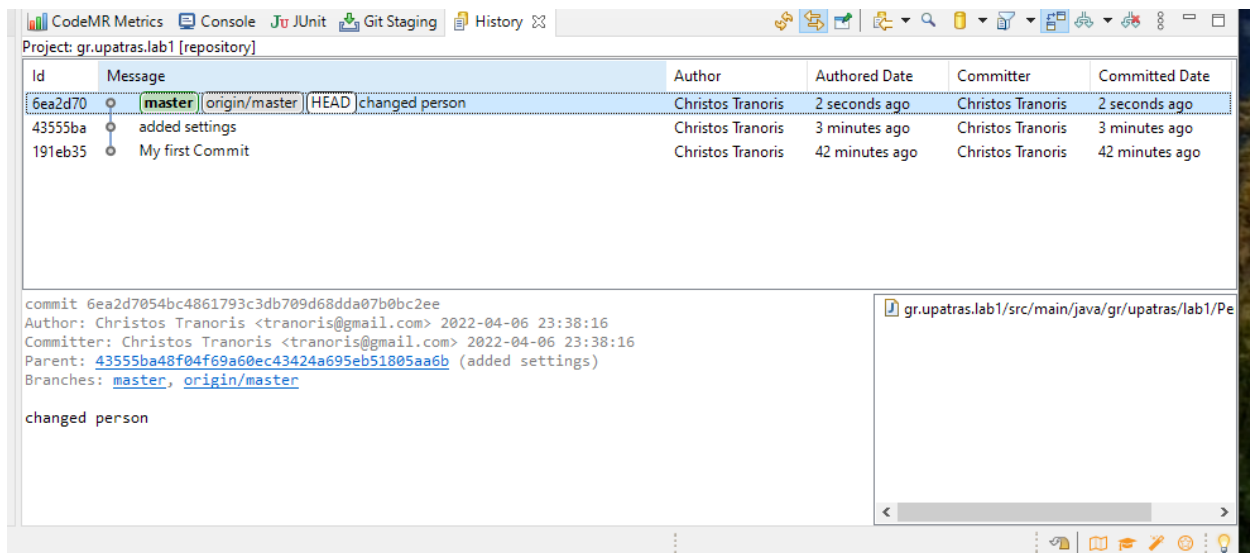
Κάνετε diff σε αρχείο. Δεξι κλικ στο project. Compare With->HEAD revision

Δειχνει αλλαγές που εχουμε κανει εμεις τοπικά



Κάνετε μικρές αλλαγές και μετά Commit/Push

Δεξι κλικ στο project. Team->Show in history για να δει το History View



Project: gr.upatras.lab1 [repository]

Id	Message	Author	Authored Date	Committer	Committed Date
6ea2d70	<a href="#">(master)</a>   <a href="#">origin/master</a>   <a href="#">HEAD</a> changed person	Christos Tranoris	2 seconds ago	Christos Tranoris	2 seconds ago
43555ba	added settings	Christos Tranoris	3 minutes ago	Christos Tranoris	3 minutes ago
191eb35	My first Commit	Christos Tranoris	42 minutes ago	Christos Tranoris	42 minutes ago

commit 6ea2d7054bc4861793c3db709d68dda07b0bc2ee  
Author: Christos Tranoris <tranoris@gmail.com> 2022-04-06 23:38:16  
Committer: Christos Tranoris <tranoris@gmail.com> 2022-04-06 23:38:16  
Parent: [43555ba48f04f69a60ec43424a695eb51805aa6b](#) (added settings)  
Branches: [master](#), [origin/master](#)

changed person

gr.upatras.lab1/src/main/java/gr/upatras/lab1/Pe