



ΚΕΝΤΡΟ ΕΠΙΜΟΡΦΩΣΗΣ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗΣ

Διαχείριση Εκδόσεων Λογισμικού με Git και GitHub

Αθ. Ανδρούτσος



Συστήματα Διαχείρισης Εκδόσεων Λογισμικού

Version Control με Git / GitHub

• VCS (Version Control Systems)

- Παρακολουθεί και διαχειρίζεται τις αλλαγές σε ένα αρχείο, συνήθως πηγαίου κώδικα (Source Control)
- Στόχος είναι να μπορούμε στο μέλλον να επαναφέρουμε τα αρχεία σε μία πρότερη κατάσταση, να συγκρίνουμε τις αλλαγές, να δούμε ποιος έκανε τις αλλαγές και πότε, να συνεργαζόμαστε με τα μέλη μίας ομάδας πάνω σε ένα κοινό codebase



Τοπικό Version Control

Version Control με Git / GitHub

- Ο εκφυλισμός της διαδικασίας του *Version Control* θα μπορούσε να είναι να κάνουμε copy/paste τα αρχεία που θέλουμε να πάρουμε backup (αντίγραφα ασφαλείας) σε ένα νέο φάκελο (με time-stamp στο όνομα του φακέλου, π.χ. Java-21-12-2022, που παίζει το ρόλο του version)
- Κάτι τέτοιο όμως δεν είναι ασφαλές ως στρατηγική διαχείρισης λαθών, δεδομένου ότι μπορεί να συμβούν λάθη που να μην μπορούμε να επαναφέρουμε (π.χ. να γράψουμε από πάνω - update)
- Θα ήταν χρήσιμο αν είχαμε μια τοπική απλή ΒΔ που να αποθηκεύει όλες τις αλλαγές στα αρχεία που θέλουμε να εποπτεύουμε



Αρχιτεκτονική Συστημάτων

Version Control με Git / GitHub

- Υπάρχουν δύο αρχιτεκτονικές διάρθρωσης συστημάτων διαχείρισης εκδόσεων λογισμικού
- **Centralized**, όπου υπάρχει ένα κεντρικός απομακρυσμένος υπολογιστής (κεντρικός server) που διαχειρίζεται όλα τα αρχεία που θέλουμε να διαχειριστούμε
- **Decentralized**, όπου εκτός από τον κεντρικό server υπάρχουν και άλλες ομότιμες οντότητες που έχουν όλα τα αρχεία που θέλουμε να διαχειριστούμε. Αν έχουμε η ομότιμες οντότητες, τότε υπάρχουν η αντίγραφα (copy ή clone) των αρχείων μας



Centralized Συστήματα

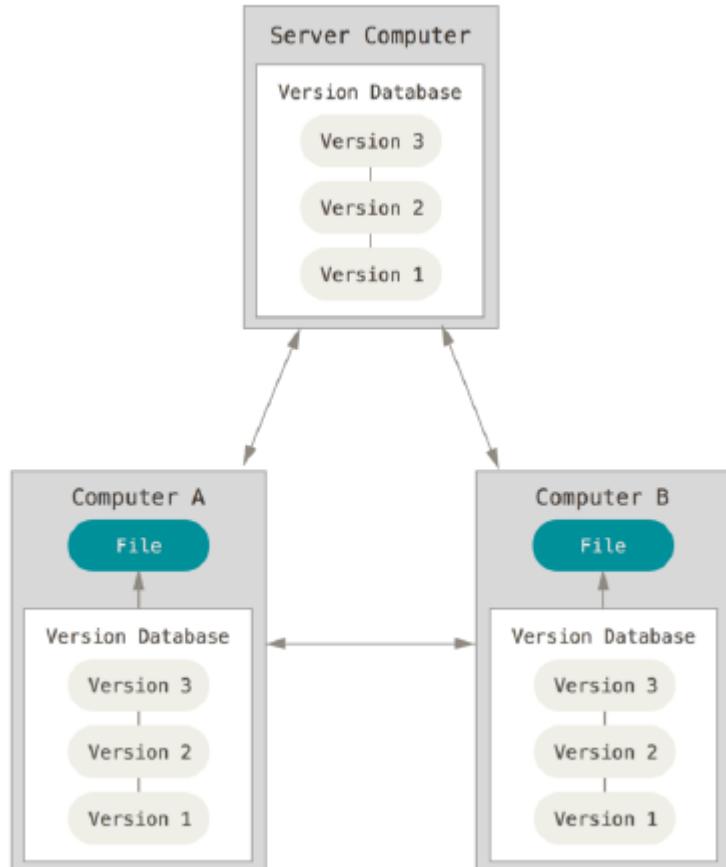
Version Control με Git / GitHub

- **Οφέλη**
 - Κεντρική διαχείριση δικαιωμάτων (ποιος κάνει τι)
 - Sharing
- **Μειονεκτήματα**
 - **Single Point of Failure** (Το δίκτυο μπορεί να μην είναι διαθέσιμο ή όταν έχουμε όλο το history του project μας σε ένα μέρος και δεν παίρνουμε backup κινδυνεύουμε να τα χάσουμε όλα)
 - **Όχι πάντα efficient** (Βασίζεται σε επικοινωνία με τον Server, το δίκτυο μπορεί να μην είναι διαθέσιμο ή να είναι αργό)



Κατανεμημένα (Distributed) Συστήματα

Version Control με Git / GitHub



- Τα τοπικά PCs είναι κλώνοι (κάνουν **mirror**). Έχουν όλο το repository και όλο το history των αρχείων του Server
- Αν ο κεντρικός server χάσει τα αρχεία του Project, κάθε μέλος της ομάδας μπορεί να τα επαναφέρει
- Άρα κάθε **τοπικό clone** είναι ένα πλήρες backup του Project



Git (1)

- Decentralized Σύστημα Διαχείρισης Εκδόσεων Λογισμικού (Version Management System)
- Μπορούμε να το εγκαταστήσουμε στον υπολογιστή μας και να το τρέχουμε τοπικά από command line. Υπάρχει για όλα τα ΛΣ (Windows, Linux, Mac)
- Μπορούμε να χρησιμοποιήσουμε ως κεντρικό server το GitHub (με δωρεάν account)
- Απαραίτητο εργαλείο οργάνωσης και διαχείρισης του κώδικα



Git (2)

Version Control με Git / GitHub

- Γιατί Git:
 - **Back-up** των project μας
 - **Portfolio-management**
 - **Version-Management**
 - **Συνεργασία ομάδων** (Collaboration)
 - **Ιστορικό** (History)
 - Δυνατότητα αναίρεσης αλλαγών, μετάβαση σε παλαιότερες εκδόσεις
 - Εύκολη αναίρεση σε τυχόν λάθη



Git (3)

- Το Git είναι ένα κατανεμημένο VCS και εν αντιθέσει με τα υπόλοιπα κατανεμημένα συστήματα που κάνουν delta-based version control, το Git είναι ένα mini filesystem το οποίο αποθηκεύει τα snapshots (περιεχόμενα) της κάθε έκδοσης του κάθε αρχείου
- Δηλαδή αποθηκεύει τα αρχεία και όχι τα delta changes



Git (4)

- Η βασική δομή των version control συστημάτων είναι το **Repository (mini file system)**. Τυπικά κάθε Project αντιστοιχεί σε ένα Repository
- Μέσα στο repository, κάθε άτομο της ομάδας μπορεί να δουλεύει σε διαφορετικό branch, δηλαδή σε διαφορετικό μονοπάτι
- Μπορούμε να ενώσουμε όλα τα branches σχεδόν αυτόματα σε ένα ενιαίο branch (main branch)
- Σε ανοιχτά (open) repositories ο καθένας μπορεί να βελτιώσει τον κώδικα (open source)



Git (5)

- Οι λειτουργίες του Git γίνονται σε ένα **τοπικό repository**, που περιέχει τα αρχεία που εποπτεύουμε
- Όλο το history των αλλαγών βρίσκεται τοπικά στο pc μας (στο **local repository**) και επομένως όλες οι λειτουργίες γίνονται εξαιρετικά γρήγορα χωρίς τις καθυστερήσεις δικτύου που υπάρχουν στα centralized συστήματα
- Σε μεταγενέστερο χρόνο κάνουμε upload (push) τις αλλαγές στον remote server (π.χ. GitHub)



Git (6)

- Ο remote server είναι γενικά ανεξάρτητος από τα local repos
- Τα local repos μπορούμε να τα συνδέουμε με οποιονδήποτε remote server
- Άλλοι γνωστοί remote servers εκτός από το **GitHub** είναι τα **GitLab** και **BitBucket**



Ασφάλεια Δεδομένων

Version Control με Git / GitHub

- Integrity - Secure Hash Algorithm 1 (**SHA-1**)
- Για κάθε αρχείο, με βάση τα περιεχόμενά του δημιουργείται ένα hash value 160-bit / 40 hex-digits-μοναδικό! (Κάθε δεκαεξαδικό ψηφίο είναι μισό byte, συνεπώς 40 δεκαεξαδικά ψηφία είναι 20 bytes = 160 bits)
Π.χ.
- 129fac190ea7e659129cd6190ebc1324feb66290
- .. αν και η google έδειξε πρόσφατα (2017) ότι ο SHA-1 δεν είναι 100% ασφαλής σε **collision attacks**, δηλ. διαφορετικά αρχεία με το ίδιο SHA-1 hash value
- Το Git αποθηκεύει αυτό το hash value με κάθε snapshot των αρχείων



Τρεις βασικές καταστάσεις

Version Control με Git / GitHub

- Τρεις καταστάσεις των αρχείων

1. *Committed*

- stored στο Git Directory / *.git directory*

2. *Modified*

- changed αλλά όχι committed.

3. *Staged*

- το αρχείο εισάγεται σε μία ενδιάμεση περιοχή (Staging Area) για να αποθηκευτεί με το επόμενο commit



Μέρη ενός Git Project

Version Control με Git / GitHub

- **Git Repository** (.git directory). Τεχνικά, αυτός ο φάκελος αντιπροσωπεύει όλο το git filesystem
- **Working directory** – Μία version του project στην οποία δουλεύουμε
- **Staging area** – Μία περιοχή που λειτουργεί ως ενδιάμεσος buffer των αρχείων που θα αποθηκευτούν στο επόμενο commit



Git Workflow

Version Control με Git / GitHub

- Κάνουμε **modify** (add new files, change ή delete) τα αρχεία μας στο working dir
- Κάνουμε **stage** εκείνα τα αρχεία που θέλουμε να πάνε με το επόμενο commit
- Κάνουμε **commit** – το git κοιτάει το staging area και παίρνει ένα snapshot των αρχείων και τα αποθηκεύει στο git directory



Ασφάλεια Git

Version Control με Git / GitHub

- Αν τα αρχεία μας δεν έχουν γίνει commit, τότε μπορεί να χαθούν ή να αλλοιωθούν
- Αν γίνουν commit, τότε δύσκολα χάνονται ιδιαίτερα αν έχουν γίνει push σε κάποιο εξωτερικό repository



Εγκατάσταση Git (1)

Version Control με Git / GitHub

- Το command line είναι **το μόνο μέρος που μπορούμε να τρέξουμε όλα τα git commands**
- Τα γραφικά περιβάλλοντα παρέχουν περιορισμένες δυνατότητες, δηλαδή παρέχουν ένα υποσύνολο των git commands
- Ωστόσο και τα γραφικά περιβάλλοντα μπορούν πιο περιορισμένα να χρησιμοποιηθούν



Εγκατάσταση Git (2)

Version Control με Git / GitHub

- Linux
 - sudo apt-get update
 - sudo apt-get install git
 - git --version
- Windows
 - <https://git-scm.com/download>
 - <https://gitforwindows.org/>



UNIX / Linux

Version Control με Git / GitHub

- Το UNIX και το Linux, που είναι η υλοποίησή του UNIX για προσωπικούς υπολογιστές, είναι ένα Λειτουργικό Σύστημα, δηλαδή ένα σύνολο προγραμμάτων που μας επιτρέπει την επικοινωνία με τον υπολογιστή
- Από το σύνολο των προγραμμάτων, ένα ειδικό πρόγραμμα που ονομάζεται φλοιός (shell) υλοποιεί την επικοινωνία του χρήστη με το Λειτουργικό Σύστημα, παρέχοντας ένα σύνολο εντολών (**CLI – Command Line Interface**) που μπορεί ο χρήστης να δίνει και να επικοινωνεί με το Linux
- Ο φλοιός είναι δηλαδή η Διεπαφή (interface) του χρήστη με τον υπολογιστή



Bash Shell

Version Control με Git / GitHub

- Υπάρχουν διάφορες υλοποιήσεις φλοιών του Linux με πιο γνωστές τις C Shell (csh), Korn Shell (ksh), Bourne Shell (sh) και GNU Bourne-Again Shell (Bash)
- Ο Bash shell είναι ο πιο δημοφιλής φλοιός σε συστήματα Linux και είναι ο φλοιός που χρησιμοποιείται στο Git for Windows, που είναι ένα emulation του φλοιού Bash Shell για Windows ενώ παρέχει επιπλέον και Git Commands
- To Git for Windows παρέχεται μέσω του MinGW64, ενός περιβάλλοντος που παρέχει το emulation του Linux σε Windows



Git for Windows

Version Control με Git / GitHub

- Το Git for Windows είναι επομένως ένα Git CLI (Command Line Interface) emulation για Windows
- Πρόκειται για Linux/UNIX CLI emulation, ένα πρόγραμμα δηλαδή, που παρέχει όλες τις Linux εντολές (CLI)
- Στα υπόλοιπα λειτουργικά συστήματα MacOS και Linux, το Git τρέχει με native τρόπο



Install Git for Windows (1)

Version Control με Git / GitHub

The screenshot shows the official Git website at git-scm.com/download. The main navigation bar includes links for About, Documentation, Downloads (highlighted), GUI Clients, Logos, and Community. A sidebar promotes the "Pro Git book". The central content area features a large "Downloads" section with links for macOS, Windows, and Linux/Unix. To the right, a monitor displays the latest source release version 2.43.0 and a "Download for Windows" button. Below this are sections for "GUI Clients" (describing built-in tools like git-gui and gitk) and "Logos" (offering various Git logo formats). A search bar at the top right allows users to search the entire site.

- Κατεβάζουμε την τελευταία έκδοση για το λειτουργικό σύστημα που έχουμε



Install Git for Windows (2)

Version Control με Git / GitHub

Download for Windows

[Click here to download](#) the latest **(2.43.0) 64-bit** version of **Git for Windows**. This is the most recent maintained build. It was released **13 days ago**, on 2023-11-20.

Other Git for Windows downloads

[Standalone Installer](#)

[32-bit Git for Windows Setup.](#)

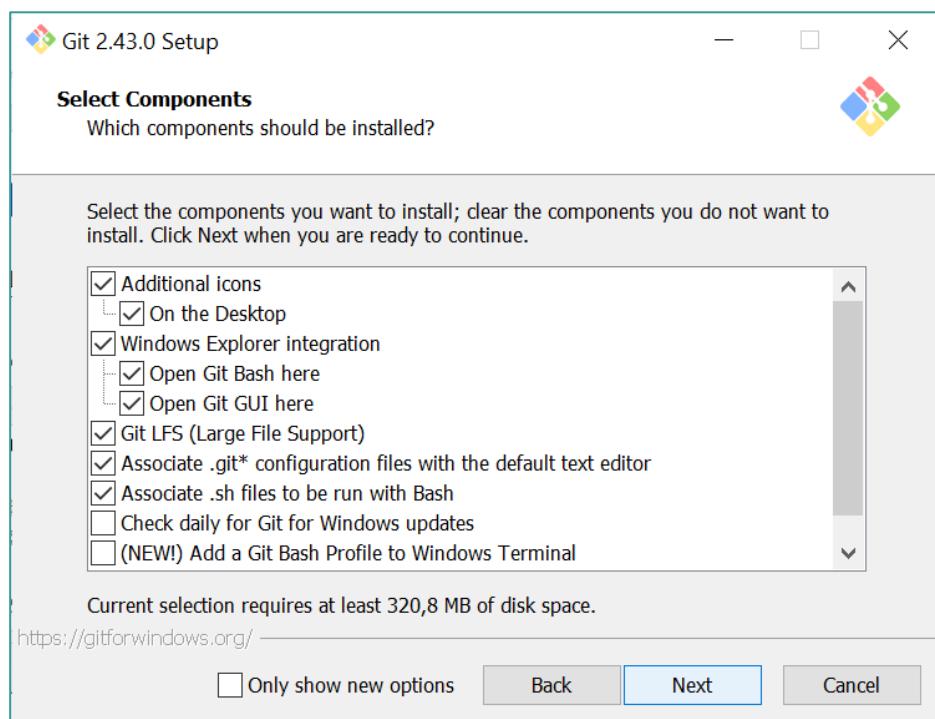
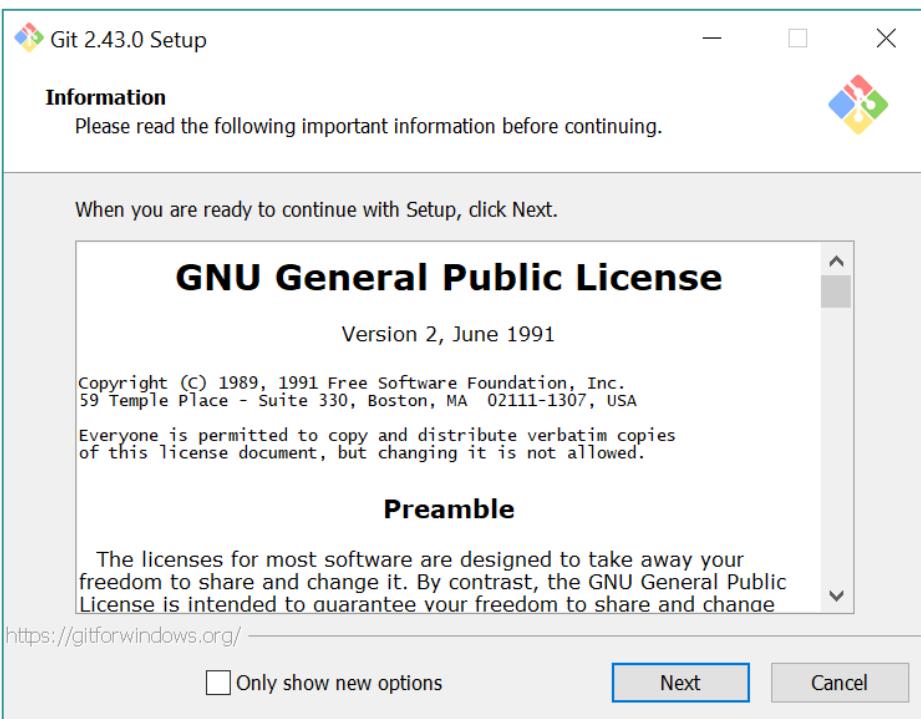
[64-bit Git for Windows Setup.](#)

- Κατεβάζουμε την 64-bit έκδοση



Git for Windows (1)

Version Control με Git / GitHub



- Next, Next



Git for Windows (2)

Version Control με Git / GitHub

Git 2.43.0 Setup

Choosing the default editor used by Git

Which editor would you like Git to use?

Use Vim (the ubiquitous text editor) as Git's default editor

The [Vim editor](#), while powerful, [can be hard to use](#). Its user interface is unintuitive and its key bindings are awkward.

Note: Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.

Note: This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

Git 2.43.0 Setup

Adjusting the name of the initial branch in new repositories

What would you like Git to name the initial branch after "git init"?

Let Git decide

Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project [intends](#) to change this default to a more inclusive name in the near future.

Override the default branch name for new repositories

NEW! Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:

main

This setting does not affect existing repositories.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

- Δεξιά επιλέγουμε το **main** ως το default branch, δεδομένου ότι το GitHub από τον Οκτώβριο 2020 προτείνει όλα τα νέα repositories να χρησιμοποιούν ως default branch name, το **main** αντί του **master branch** που ίσχυε τα προηγούμενα χρόνια



Git for Windows (3)

Version Control με Git / GitHub

Git 2.43.0 Setup

Adjusting your PATH environment

How would you like to use Git from the command line?

Use Git from Git Bash only

This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

Git from the command line and also from 3rd-party software

(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

Use Git and optional Unix tools from the Command Prompt

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

Git 2.43.0 Setup

Choosing the SSH executable

Which Secure Shell client program would you like Git to use?

Use bundled OpenSSH

This uses ssh.exe that comes with Git.

Use (Tortoise)Plink

To use PuTTY, specify the path to an existing copy of (Tortoise)Plink.exe:

C:\Program Files\PuTTY\plink.exe [...](#)

Set ssh.variant for Tortoise Plink

Use external OpenSSH

NEW! This uses an external ssh.exe. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

- Next



Git for Windows (4)

Version Control με Git / GitHub

Git 2.43.0 Setup

Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?

Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.

Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

Git 2.43.0 Setup

Configuring the line ending conversions
How should Git treat line endings in text files?

Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

- Next



Git for Windows (5)

Version Control με Git / GitHub

Git 2.43.0 Setup

Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?

Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.

Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

Git 2.43.0 Setup

Choose the default behavior of `git pull`
What should `git pull` do by default?

Fast-forward or merge
Fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

Rebase
Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.

Only ever fast-forward
Fast-forward to the fetched branch. Fail if that is not possible.
This is the standard behavior of `git pull`.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

- Next



Git for Windows (6)

Version Control με Git / GitHub

Git 2.43.0 Setup

Choose a credential helper

Which credential helper should be configured?

Git Credential Manager
Use the [cross-platform Git Credential Manager](#).
See more information about the future of Git Credential Manager [here](#).

None
Do not use a credential helper.

<https://gitforwindows.org/>

Only show new options [Back](#) [Next](#) [Cancel](#)

Git 2.43.0 Setup

Configuring extra options

Which features would you like to enable?

Enable file system caching
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

Enable symbolic links
Enable [symbolic links](#) (requires the SeCreateSymbolicLink permission).
Please note that existing repositories are unaffected by this setting.

<https://gitforwindows.org/>

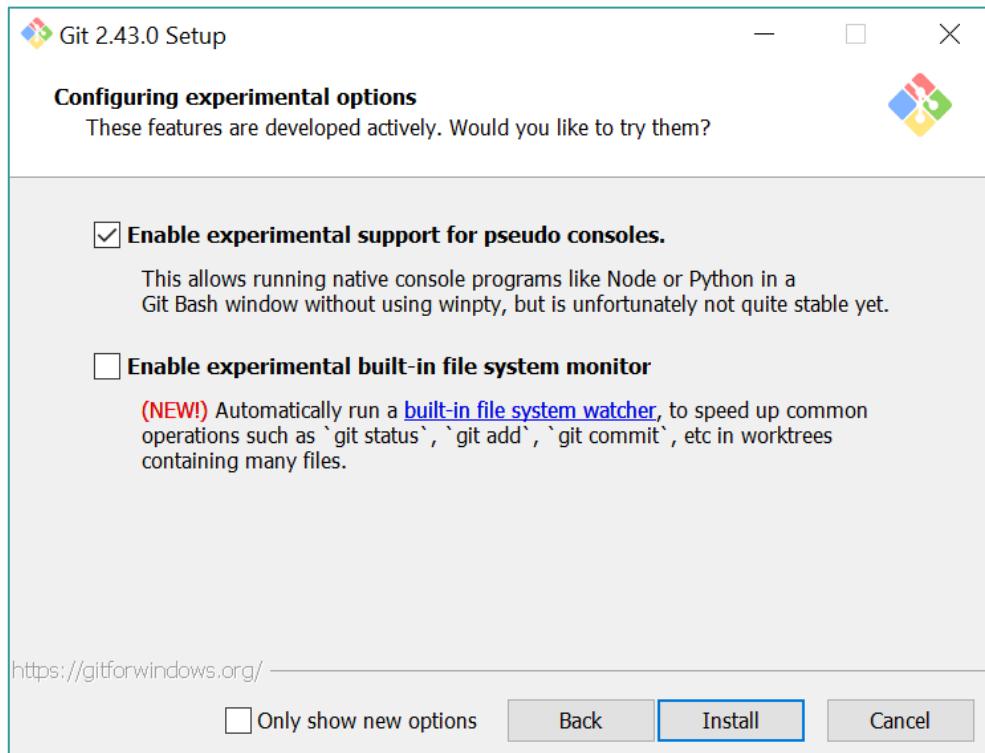
Only show new options [Back](#) [Next](#) [Cancel](#)

- Next



Git for Windows (7)

Version Control με Git / GitHub

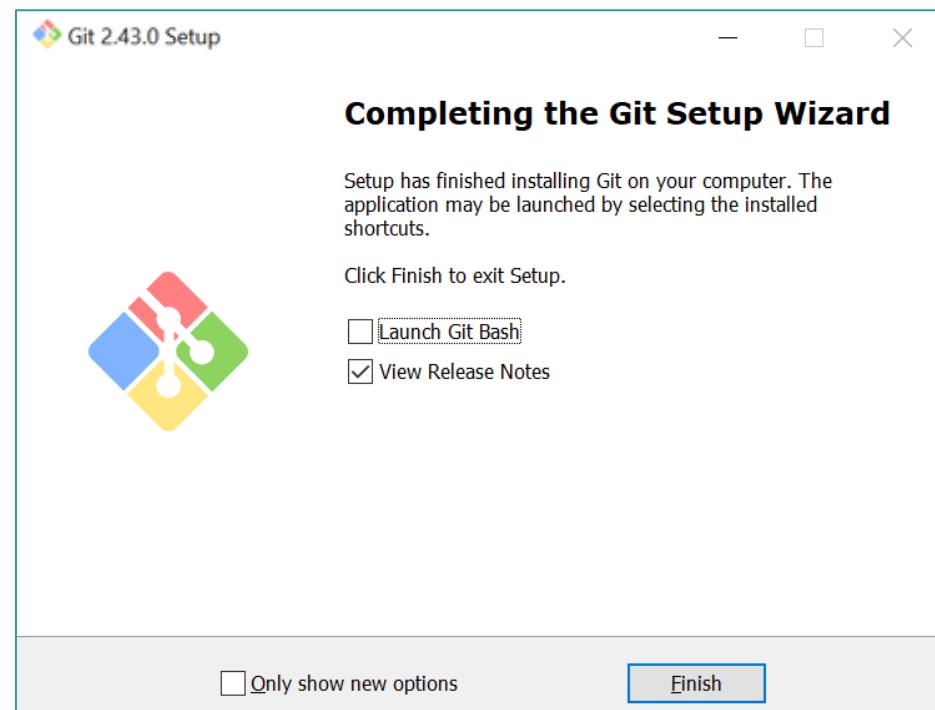
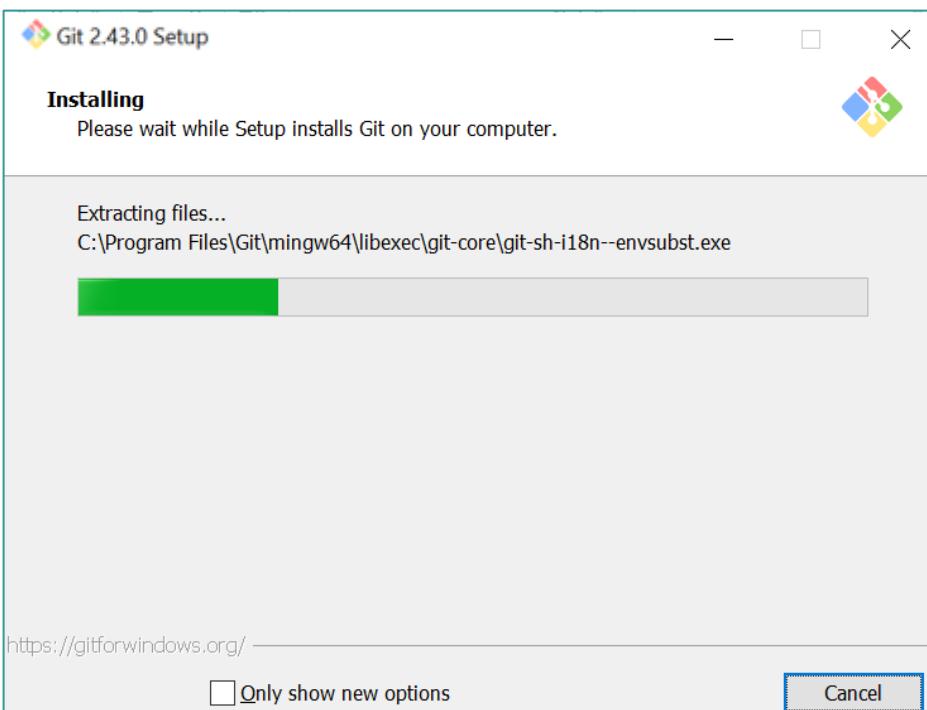


- **Install**



Git for Windows (7)

Version Control με Git / GitHub



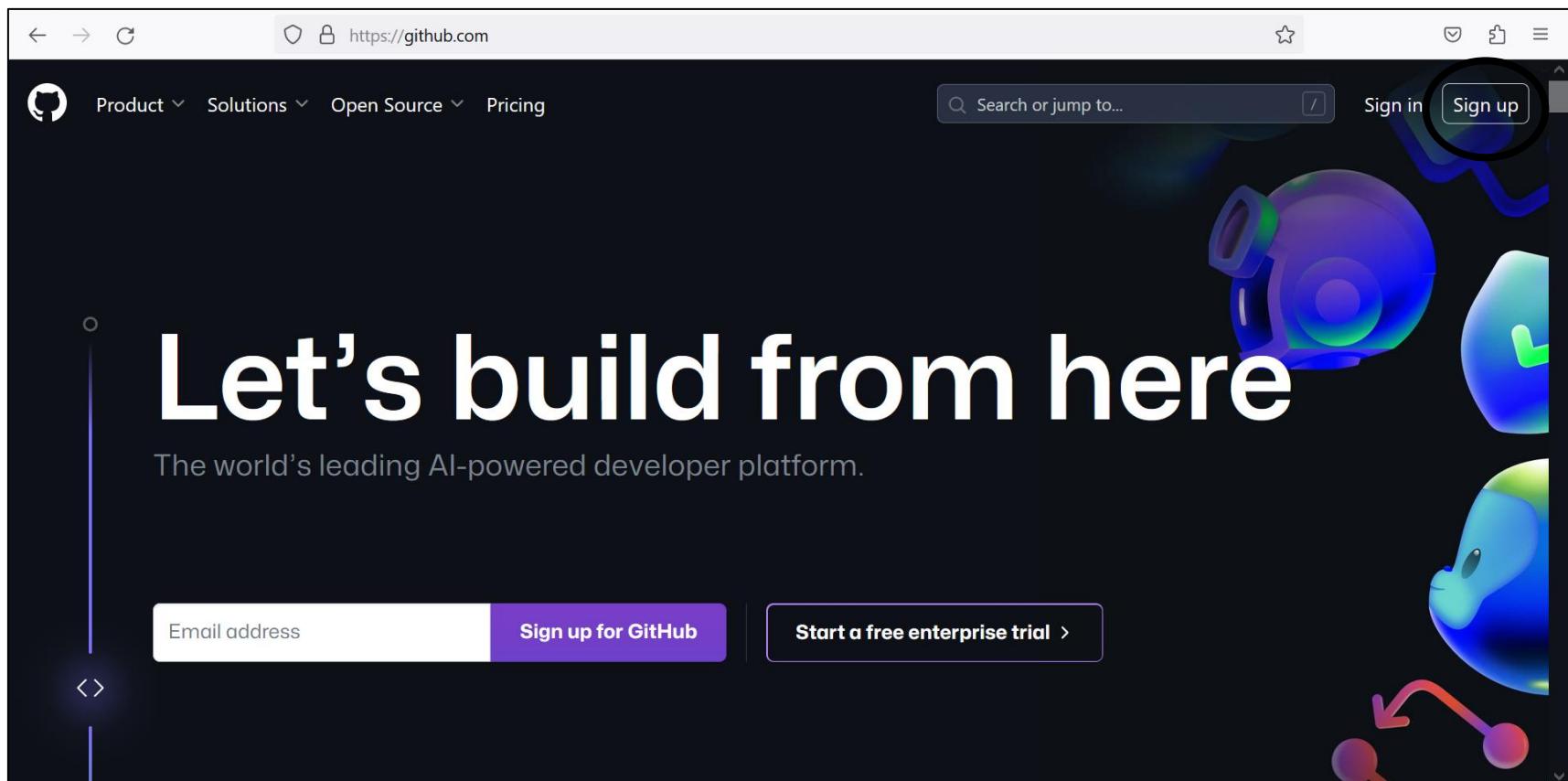
- Finish



GitHub

Version Control με Git / GitHub

- Τώρα ας πάμε στο GitHub (<https://github.com>) για να δημιουργήσουμε ένα λογαριασμό. Επιλέγουμε **Sign up**





Λογαριασμός χρήστη στο GitHub

Version Control με Git / GitHub

Join GitHub

Create your account

Username *

Email address *

Password *

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

Email preferences

Send me occasional product updates, announcements, and offers.

Verify your account

Please solve this puzzle so we know you are a real person

[Verify](#)

[Create account](#)

- Στο sign up δίνετε τα στοιχεία που χρειάζεται, μετά Verify και Create Account



Δημιουργία προφίλ χρήστη

Version Control με Git / GitHub

The screenshot shows the GitHub settings profile page for a user named Athanassios Androutsos. The left sidebar has a menu with options: Account settings, Profile (which is selected), Account, Appearance (with a 'New' badge), Account security, Billing & plans, Security log, Security & analysis, Emails, Notifications, and SSH and GPG keys. The main content area is titled 'Public profile'. It includes fields for 'Name' (Athanassios Androutsos) and 'Profile picture' (a circular photo of a man speaking into a microphone). Below these are sections for 'Public email' (a8anassis@gmail.com) and 'Bio' (PhD in Informatics. Research fields: Software Engineering, Educational Technologies, Networking and Economics.). There is also a note about mentioning other users and organizations.

- Πάνω δεξιά επιλέγετε το avatar με το βελάκι και στο μενού **Settings**
- Εισάγετε στο **προφίλ** σας μία φωτογραφία, το ονοματεπώνυμό σας και το *e-mail* σας και πιθανώς και άλλα στοιχεία που προσδιορίζουν την επαγγελματική σας ταυτότητα



Προβολή Προφίλ

Version Control με Git / GitHub



Athanassios
Androutsos
a8anassis

PhD in Informatics. Research fields:
Software Engineering, Educational
Technologies, Networking and
Economics.

- Athens University of Economics and Bu..
- Athens, Greece
- a8anassis@gmail.com
- <https://aueb.gr/>
- @a8anassis

- Πάλι από το avatar δεξιά επιλέγετε “Your profile”
 - Όνομα
 - Username
 - E-mail



Git username, email (1)

Version Control με Git / GitHub

- Αφού δημιουργήσουμε το λογαριασμό μας στο GitHub, ξαναπάμε στο Git Bash και εισάγουμε:
 - **username**, μπορεί να είναι το ίδιο ή διαφορετικό από αυτό που έχουμε στο GitHub, Μπορεί εδώ να είναι το όνομά μας, για παράδειγμα "Th. Androutsos"
 - **e-mail**, το ίδιο με αυτό που έχουμε στο GitHub, ώστε να μπορούμε να συνδέσουμε το GitHub με το τοπικό μας Git
- Για εισαγωγή username, δίνουμε:
 - **git config --global user.name "username"**
π.χ. git config --global user.name "Th. Androutsos"
- Για επιβεβαίωση, δίνουμε:
 - **git config --global user.name**



Git username, email (2)

Version Control με Git / GitHub

- Κάνουμε το ίδιο για το e-mail
 - **git config --global user.email "email@example.com"**
π.χ. `git config --global user.email "a8anassis@gmail.com"`
 - **git config --global user.email**
- Για επιβεβαίωση
- Το Git χρησιμοποιεί αυτές τις πληροφορίες για τα πεδία του 'Αποστολέα/Συγγραφέα' του commit object, οπότε δεν χρειάζεται να τα δίνουμε κάθε φορά



Git username, email (3)

Version Control με Git / GitHub

```
MINGW64:/c/Users/a8ana
a8ana@thanassis-pc MINGW64 ~
$ git config --global user.name "a8ana"

a8ana@thanassis-pc MINGW64 ~
$ git config --global user.name
a8ana

a8ana@thanassis-pc MINGW64 ~
$

a8ana@thanassis-pc MINGW64 ~
$ git config --global user.email "a8anassis@gmail.com"

a8ana@thanassis-pc MINGW64 ~
$ git config --global user.email
a8anassis@gmail.com

a8ana@thanassis-pc MINGW64 ~
$
```

- Ως **username** μπορούμε να εισάγουμε είτε το username του GitHub, είτε το όνομά μας ή ένα ψευδώνυμο. Αυτό θα φαίνεται στα commits που κάνουμε στο Git και στο GitHub. Ως **e-mail** πρέπει να δώσουμε το ίδιο που έχουμε δώσει στο GitHub



git config --global

- Το *--global* flag λέει στο git να αποθηκεύσει τα στοιχεία του χρήστη στο global configuration file του για όλα τα Git Repositories.
- Στα Windows, UNIX, Linux, OSX το αρχείο είναι το *.gitconfig* και βρίσκεται στο user home directory (~).
- Χωρίς το *--global* flag, το git απλά εφαρμόζει τις ρυθμίσεις στο επίπεδο του repository που δουλεύουμε



.gitconfig

Version Control με Git / GitHub

```
MINGW64:/c/Users/a8ana
a8ana@thanassis-pc MINGW64 ~
$ git config --global user.email
a8anassis@gmail.com

a8ana@thanassis-pc MINGW64 ~
$ pwd
/c/Users/a8ana

a8ana@thanassis-pc MINGW64 ~
$ cat .gitconfig
[user]
    name = a8ana
    email = a8anassis@gmail.com

a8ana@thanassis-pc MINGW64 ~
$ |
```

- To .gitconfig file στο user home directory



Άλλα global settings

Version Control με Git / GitHub

```
git config --global init.defaultBranch main
```

```
git config --global color.ui auto
```

- Σε περίπτωση που το default branch δεν είναι το main, μπορούμε να το δώσουμε όπως παραπάνω
- Επίσης, το default είναι το colored terminal output, ενώ σε περίπτωση που ήταν color.ui false θα μπορούσαμε να επαναφέρουμε το colored terminal όπως παραπάνω



Repository

Version Control με Git / GitHub

- Η βασική δομή στο Git/GitHub είναι το **repository**
- Ένα repository αντιστοιχεί λογικά σε ένα project
- Τεχνικά, το repository υλοποιείται με ένα **.git** folder που αποθηκεύει όλα τα data (files/folders) καθώς και τα metadata των αρχείων/φακέλων



Δημιουργία repository (1)

Version Control με Git / GitHub

- Με δύο (2) τρόπους:
 1. Αν θέλουμε ένα τοπικό μας φάκελο (local folder/directory) να τον μετατρέψουμε σε Git folder/directory δίνουμε την εντολή **git init** μέσα στον φάκελο που θέλουμε να μετατρέψουμε σε git folder (οπότε και δημιουργείται ένας .git folder μέσα στον φάκελο)
 2. Στη συνέχεια δημιουργούμε ένα repository στο GitHub και **συνδέουμε το local repository με το remote repository (στο GitHub)**, ώστε να μεταφέρουμε (ανεβάζουμε/κατεβάζουμε – push/pull) αρχεία και φακέλους



Δημιουργία repository (2)

Version Control με Git / GitHub

2. Ο δεύτερος τρόπος είναι να αντιγράψουμε (clone) ένα ήδη υπάρχον repository του GitHub
 - Αν θέλουμε να αντιγράψουμε ένα repository από το GitHub και να το κάνουμε cloning στον τοπικό μας δίσκο, τότε κάνουμε **git clone**



Δημιουργία νέου project directory (1)

Version Control με Git / GitHub

- Όπως έχουμε αναφέρει ένα project τεχνικά είναι ένας folder/directory του ΛΣ
- Στο ΛΣ Windows ο όρος είναι φάκελος (folder), στα Linux/MacOS ο όρος είναι κατάλογος (directory)
- Οι δύο όροι χρησιμοποιούνται εναλλακτικά



Δημιουργία νέου project directory (2)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects
$ mkdir git-testbed

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects
$ cd git-testbed/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed
$ git init
Initialized empty Git repository in C:/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git/
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ |
```

- Αφού πάμε με `cd` σε ένα φάκελο (στο παράδειγμα `webprojects`) μέσα στον οποίο θέλουμε να δημιουργήσουμε ένα νέο directory με όνομα `git-testbed`, κάνουμε `mkdir git-testbed`
- Στη συνέχεια μεταφερόμαστε στο νέο φάκελο με `cd git-testbed` και στη συνέχεια κάνουμε `git init`



ls -la

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ ls -la
total 8
drwxr-xr-x 1 a8ana 197609 0 Dec  4 13:32 .
drwxr-xr-x 1 a8ana 197609 0 Dec  4 13:32 ..
drwxr-xr-x 1 a8ana 197609 0 Dec  4 13:32 .git/
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ |
```

- Η εντολή git init δημιουργησε ένα sub-directory με όνομα .git. Το **.git directory** είναι τεχνικά το **Git repository**, όπου αποθηκεύονται όλα όσα το Git αποθηκεύει
- Το μοναδικό version του Project μας είναι αυτό που προσδιορίζεται από το **main branch**, που είναι το default branch του project μας



.git directory (1)

Version Control με Git / GitHub

```
MINGW64:/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ pwd
/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ cd .git/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git (GIT_DIR!)
$ ls -la
total 11
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:54 .
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 ../
-rw-r--r-- 1 a8ana 197609 21 Dec 4 13:32 HEAD
-rw-r--r-- 1 a8ana 197609 130 Dec 4 13:32 config
-rw-r--r-- 1 a8ana 197609 73 Dec 4 13:32 description
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 hooks/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 info/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 objects/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 refs/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git (GIT_DIR!)
$ |
```

- Κάνοντας `ls -la` εμφανίζονται τα default περιεχόμενα του `.git` directory. Το αρχείο `description` αφορά το πρόγραμμα GitWeb, το αρχείο `config` αφορά project-specific config εν αντιθέσει με το `~/.gitconfig` που περιέχει global settings για όλα τα repositories. Ο φάκελος `/hooks` περιλαμβάνει scripts που υλοποιούν τις βασικές λειτουργίες του git



.git directory (2)

Version Control με Git / GitHub

```
MINGW64:/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ pwd
/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ cd .git/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git (GIT_DIR!)
$ ls -la
total 11
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:54 .
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 ..
-rw-r--r-- 1 a8ana 197609 21 Dec 4 13:32 HEAD
-rw-r--r-- 1 a8ana 197609 130 Dec 4 13:32 config
-rw-r--r-- 1 a8ana 197609 73 Dec 4 13:32 description
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 hooks/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 info/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 objects/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 refs/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git (GIT_DIR!)
$ |
```

- Ο φάκελος info περιέχει ένα αρχείο με όνομα exclude όπου μπορούμε να ορίσουμε καταλήξεις αρχείων, φακέλους, ή αρχεία που δεν θέλουμε να αποθηκεύονται στο repository
- Ένα σημαντικό entry είναι το αρχείο HEAD που περιέχει το branch name που βρισκόμαστε (δείκτης στο current branch)



.git directory (3)

Version Control με Git / GitHub

```
MINGW64:/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ pwd
/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ cd .git/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git (GIT_DIR!)
$ ls -la
total 11
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:54 .
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 ..
-rw-r--r-- 1 a8ana 197609 21 Dec 4 13:32 HEAD
-rw-r--r-- 1 a8ana 197609 130 Dec 4 13:32 config
-rw-r--r-- 1 a8ana 197609 73 Dec 4 13:32 description
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 hooks/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 info/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 objects/
drwxr-xr-x 1 a8ana 197609 0 Dec 4 13:32 refs/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed/.git (GIT_DIR!)
$ |
```

- Ο φάκελος objects αποθηκεύει όλα τα πραγματικά περιεχόμενα (λειτουργεί σαν βάση δεδομένων). Ο φάκελος refs περιλαμβάνει references (δείκτες όπως branches, tags, remotes) στα αντικείμενα που έχουν γίνει commit (έχουν αποθηκευτεί στο repository, commit objects). Κάτι ακόμα που περιλαμβάνεται στο .git είναι το index file (που εδώ δεν υπάρχει ακόμα) που υλοποιεί το staging area



Git Store – Git Database

Version Control με Git / GitHub

- Το Git είναι ένα content-addressable σύστημα διαχείρισης περιεχομένου, δηλαδή ένα **key-value** data store
- Αυτό σημαίνει πως οτιδήποτε αποθηκεύουμε στο git repository, το Git μας επιστρέφει ένα μοναδικό κλειδί, ένα μοναδικό SHA-1 hash value, 40 χαρακτήρων
- Τα hash values μπορούμε να τα βλέπουμε ως signatures, ως μοναδικές τιμές που προκύπτουν από τα περιεχόμενα ενός αρχείου
- Κατ' αυτόν τον τρόπο δεν μπορεί κανείς να αλλοιώσει τα περιεχόμενα ενός αρχείου, γιατί το hash value που θα προκύψει θα είναι διαφορετικό από το αρχικό και θα γίνει αντιληπτό ότι το αρχείο έχει γίνει tamper (unauthorized changes)



Hash value (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/C  
$ echo 'test coding' | git hash-object -w --stdin  
4b92f9b413c0d955911653a9ff3a941a86595329  
  
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/C  
$ |
```

- Στο παράδειγμα η git hash-object παίρνει το αποτέλεσμα της echo 'test coding' και δημιουργεί ένα hash-value
- Ταυτόχρονα με το -w flag αποθηκεύουμε στην βάση δεδομένων. Με --stdin λέμε στο git να αποθηκεύσει, όχι ένα αρχείο, αλλά τα περιεχόμενα του stdin, δηλαδή το 'test coding'
- Παρατηρούμε επίσης το SHA-1 hash που δόθηκε.



Hash value (2)

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Coding
$ find .git/objects -type f
.git/objects/4b/92f9b413c0d955911653a9ff3a941a86595329

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFac
$ git cat-file -p 4b92f9b413c0d955911653a9ff3a941a86595329
test coding
```

- Με `find .git/objects -type f` λέμε στο `git` να μας επιστρέψει τα αρχεία του φακέλου `.git/objects`. Παρατηρούμε πως το SHA-1 hash που δόθηκε είναι στην πραγματικότητα ένα αρχείο όπου τα δύο πρώτα αριστερά γράμματα χρησιμοποιούνται ως directory name και οι υπόλοιποι 38 χαρακτήρες ως το filename
- Με `git cat-file` μπορούμε να εμφανίσουμε τα περιεχόμενα του αρχείου. Το `-p` σημαίνει ότι το `git` (αφού πρώτα να εντοπίσει κατάλληλα τον τύπο του αρχείου) θα εκτυπώσει τα περιεχόμενά του αρχείου



.git/objects files (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ echo 'version 1 of a file' > aFile.txt

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ git hash-object -w aFile.txt
warning: in the working copy of 'aFile.txt', LF will be replaced by CRLF the next time Git touches it
2ace886a86c179376f226319235d7541ce26f633

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ echo 'version 2 of a file' > aFile.txt

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ git hash-object -w aFile.txt
warning: in the working copy of 'aFile.txt', LF will be replaced by CRLF the next time Git touches it
28962e0758e3bbf80e9ff3352c13a4a518e62eae

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbed (main)
$ |
```

- Έστω μία έκδοση του αρχείου aFile.txt με περιεχόμενα 'version 1 of a file' που αποθηκεύουμε στην ΒΔ και μία νέα έκδοση με μία αλλαγή 'version 2 of a file' που αποθηκεύουμε επίσης στην ΒΔ



.git/objects files (2)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codin
$ find .git/objects -type f
.git/objects/28/962e0758e3bbf80e9ff3352c13a4a518e62eae
.git/objects/2a/ce886a86c179376f226319235d7541ce26f633
.git/objects/4b/92f9b413c0d955911653a9ff3a941a86595329
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codin
$ |
```

- Παρατηρούμε ότι στην ΒΔ του Git έχουν αποθηκευτεί οι δύο τελευταίες εκδόσεις του αρχείου aFile.txt



Περιεχόμενα αρχείων ΒΔ (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFact
$ git cat-file -p 2ace886a86c179376f226319235d7541ce26f633
version 1 of a file

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFact
$ git cat-file -p 28962e0758e3bbf80e9ff3352c13a4a518e62eae
version 2 of a file

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFact
$ |
```



Περιεχόμενα αρχείων ΒΔ (2)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT
$ git cat-file -p 2ace886a86c179376f226319235d7541ce26f633 > test.txt

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT
$ cat test.txt
version 1 of a file

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT
$ git cat-file -p 28962e0758e3bbf80e9ff3352c13a4a518e62eae > test.txt

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT
$ cat test.txt
version 2 of a file

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT
$
```

- Εναλλακτικά ανακατευθύνουμε με > τα περιεχόμενα των αρχείων στο αρχείο test.txt και εκτυπώνουμε με cat



Τύπος αντικειμένων blob (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFac
$ git cat-file -t 2ace886a86c179376f226319235d7541ce26f633
blob
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFac
$ git cat-file -t 28962e0758e3bbf80e9ff3352c13a4a518e62eae
blob
```

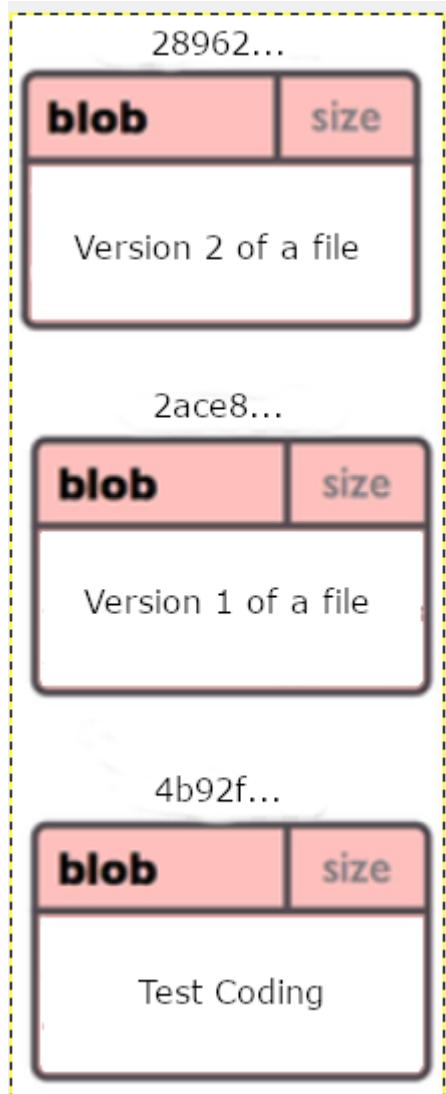
```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFac
$
```

- Με `git cat-file -t` παίρνουμε τον τύπο του αντικειμένου. Ο τύπος αυτός των αντικειμένων, τα αρχεία δηλαδή, όταν αποθηκεύονται στη ΒΔ, ονομάζονται **blob (binary large objects)**
- Δεν είναι ωστόσο πρακτικό να δίνουμε όλο το SHA-1 hash ενώ επίσης δεν έχει αποθηκευτεί το όνομα του αρχείου



Τύπος αντικειμένων blob (2)

Version Control με Git / GitHub



- Στο Git blob object παρατηρούμε ότι δεν αποθηκεύεται το όνομα του αρχείου παρά μόνο το SHA-1 hash και τα περιεχόμενα του αρχείου



Tree objects (1)

Version Control με Git / GitHub

- Ο επόμενος τύπος Git object είναι το **Tree object**, που μας επιτρέπει να αποθηκεύουμε το όνομα του αρχείου καθώς επίσης και να αποθηκεύουμε πολλά αρχεία
- Τα **trees** αντιστοιχούν σε **directories** και περιεχόμενα του **directory** ενώ τα **blobs** σε αρχεία και περιεχόμενα αρχείων
- Το Git δημιουργεί ένα tree object λαμβάνοντας το state του stage area ή index και στη συνέχεια δημιουργεί μία σειρά από tree objects (directory και sub-directories)



Tree objects (2)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codi
$ find .git/objects -type f
.git/objects/28/962e0758e3bbf80e9ff3352c13a4a518e62eae
.git/objects/2a/ce886a86c179376f226319235d7541ce26f633
.git/objects/4b/92f9b413c0d955911653a9ff3a941a86595329
```

MINGW64:/c/Users/a8ana

```
a8ana@thanassis-pc MINGW64 ~
$ git update-index --add --cacheinfo 100644 2ace886a86c179376f226319235d7541ce26f633 version1File.txt

$ git update-index --add --cacheinfo 100644 28962e0758e3bbf80e9ff3352c13a4a518e62eae version2File.txt
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBO
$ echo 'version 3 of a file' > newFile.txt
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana
$ git update-index --add newFile.txt
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/
$ git write-tree
58eb7214ba40e893a15068fc4865504e462560d4
```

- Με `git write-tree` αποθηκεύουμε ένα tree-object των blobs που έχουμε εισάγει στον Index (Staging area)



Tree objects (3)

Version Control με Git / GitHub

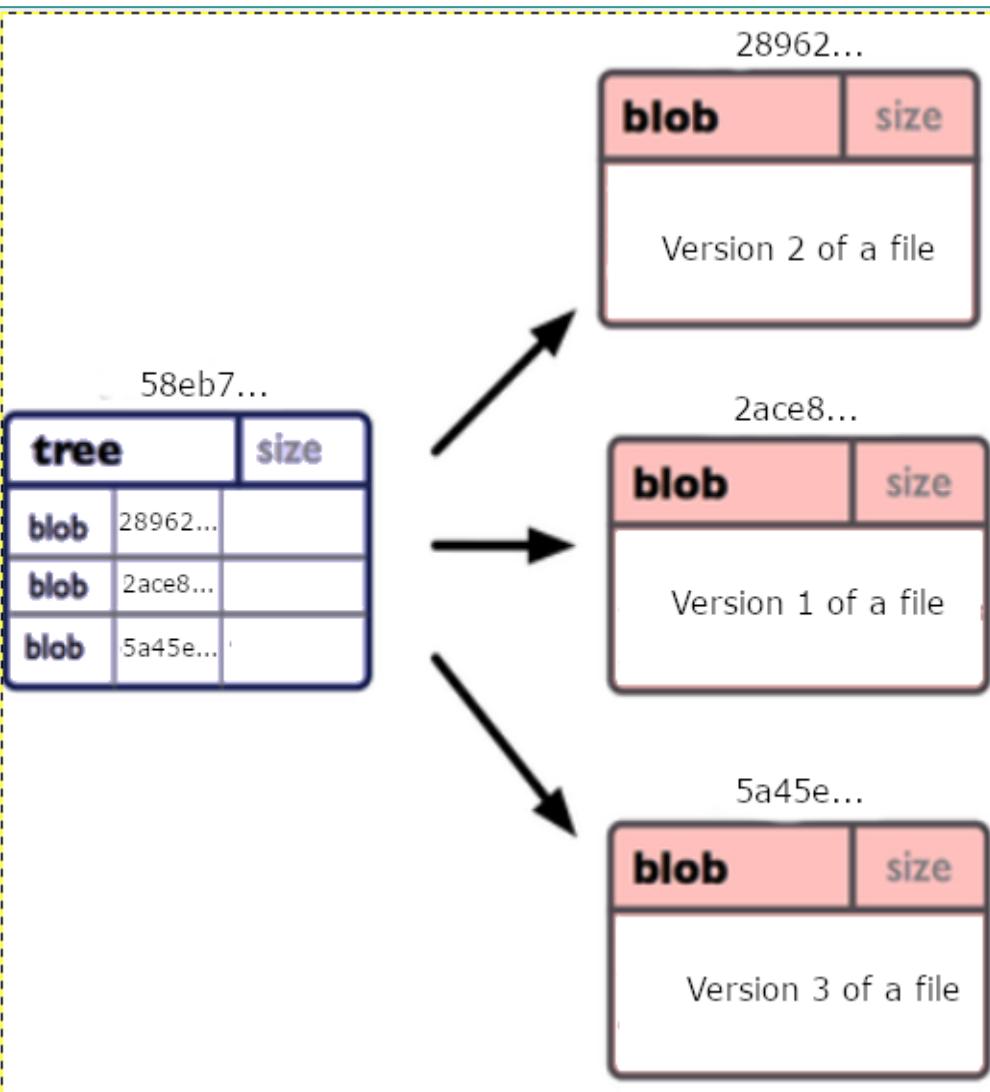
```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codi
$ find .git/objects/ -type f
.git/objects/28/962e0758e3bbf80e9ff3352c13a4a518e62eae
.git/objects/2a/ce886a86c179376f226319235d7541ce26f633
.git/objects/4b/92f9b413c0d955911653a9ff3a941a86595329
.git/objects/58/eb7214ba40e893a15068fc4865504e462560d4
.git/objects/5a/45ef9b88ebbebde48fe373a5c3d2afd46135db
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/wel
$ git cat-file -p 58eb7214ba40e893a15068fc4865504e462560d4
100644 blob 5a45ef9b88ebbebde48fe373a5c3d2afd46135db      newFile.txt
100644 blob 2ace886a86c179376f226319235d7541ce26f633      version1File.txt
100644 blob 28962e0758e3bbf80e9ff3352c13a4a518e62eae      version2File.txt
```

Παρατηρούμε ότι τα περιεχόμενα του tree object είναι τα blobs (τα hashes των blob). Επίσης, το SHA-1 hash του tree object έχει προέλθει από τα hashes των blobs που περιέχει



Tree objects (4)



- Το tree object υλοποιεί μία δομή δεδομένων που ονομάζεται Merkle Tree ή Hash Tree, όπου το κάθε leaf (τελικός κόμβος του δένδρου) γίνεται labeled με ένα cryptographic hash των περιεχομένων του, ενώ κάθε ενδιάμεσος κόμβος (inode) γίνεται labeled με το cryptographic hash των labels των απογόνων του (child nodes)



Commit Object (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Co  
$ echo 'first commit' | git commit-tree 58eb721  
15d3a60bd83e519fe65aed550b1d86759a96cc1f
```

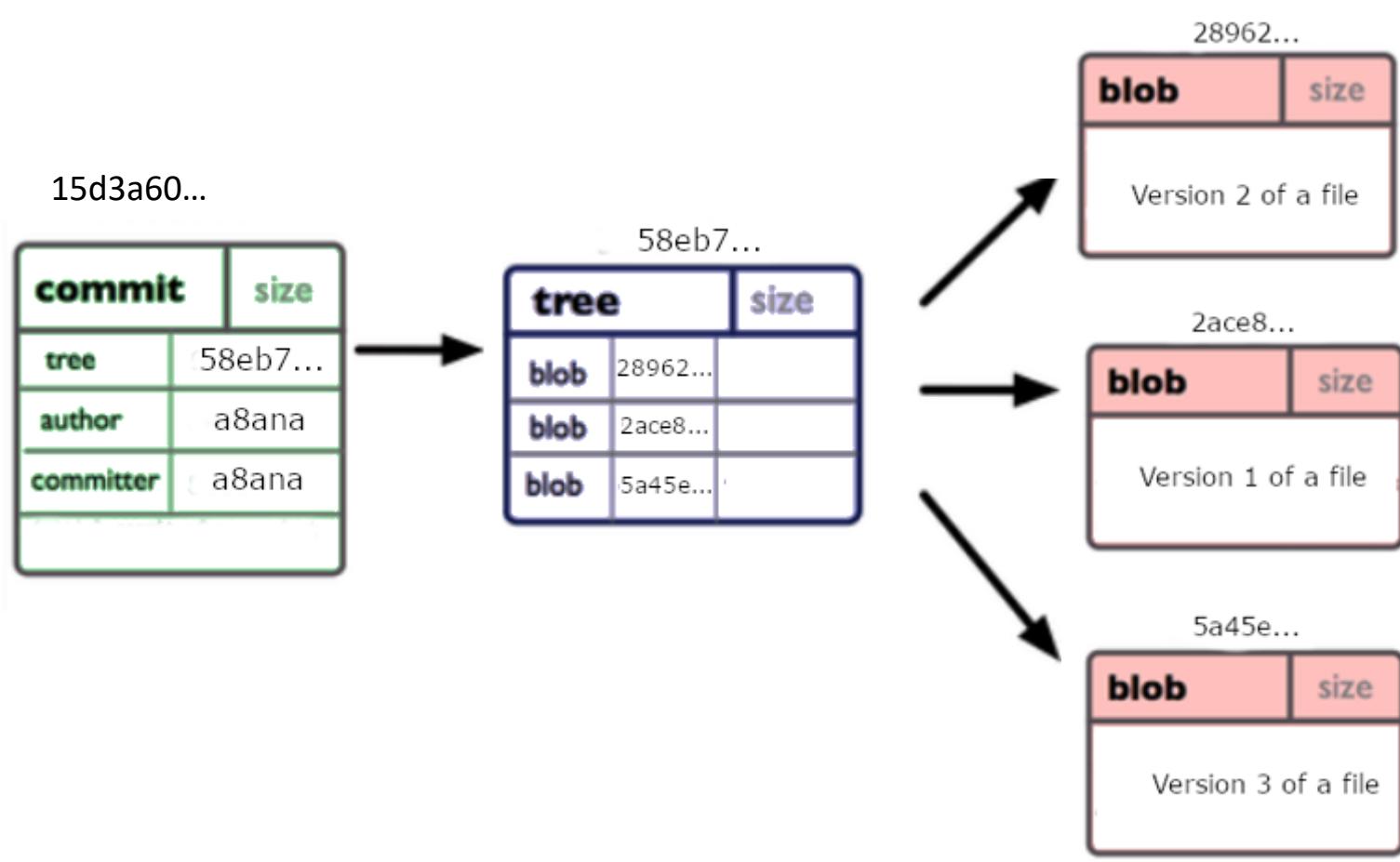
```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codir  
$ git cat-file -p 15d3a6  
tree 58eb7214ba40e893a15068fc4865504e462560d4  
author a8ana <a8anassis@gmail.com> 1670176006 +0200  
committer a8ana <a8anassis@gmail.com> 1670176006 +0200  
  
first commit
```

- Το 1^o commit δεν έχει parent commit
- Περιλαμβάνει τον author, τον committer



Commit Object (2)

Version Control με Git / GitHub





Nέο tree και commit object

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Cod
$ echo 'version 2 of a new file' > newFile.txt

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Cod
$ git update-index --add newFile.txt
warning: in the working copy of 'newFile.txt', LF will be replaced by CRLF

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Cod
$ git write-tree
01cca46637ac47ed23f825b4a5c9511fdfcccd1a8
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/git-testbe
$ echo 'Second Commit' | git commit-tree 01cca46 -p 15d3a60bd83e519fe65aed550b1d86759a96cc1f
470d6bfa02e9a12e29c91809abc4246b0490bfd3
```

- To 2nd commit έχει και το parent commit object με την παράμετρο -p



2nd Commit object

Version Control με Git / GitHub

```
a8ana@thanassis-PC MINGW64 /c/Users/a8ana/OneDrive/Coding
$ git cat-file -p 470d6bf
tree 01cca46637ac47ed23f825b4a5c9511fdfcccd1a8
parent 15d3a60bd83e519fe65aed550b1d86759a96cc1f
author a8ana <a8anassis@gmail.com> 1670176388 +0200
committer a8ana <a8anassis@gmail.com> 1670176388 +0200

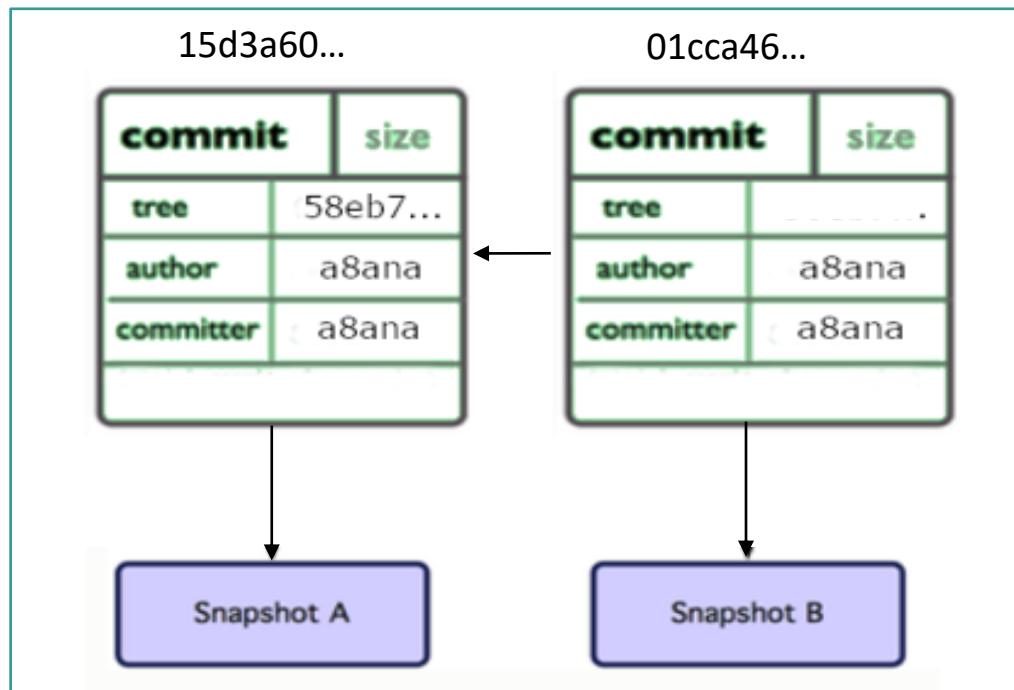
Second Commit
```

- Το 2o commit και κάθε άλλο commit περιλαμβάνει και ένα δείκτη (το SHA-1 hash) του parent commit object (του προηγούμενου στην ιεραρχία commit object) δημιουργώντας μία αντίστροφη λίστα (single-direction list in reverse)



Commit History

Version Control με Git / GitHub

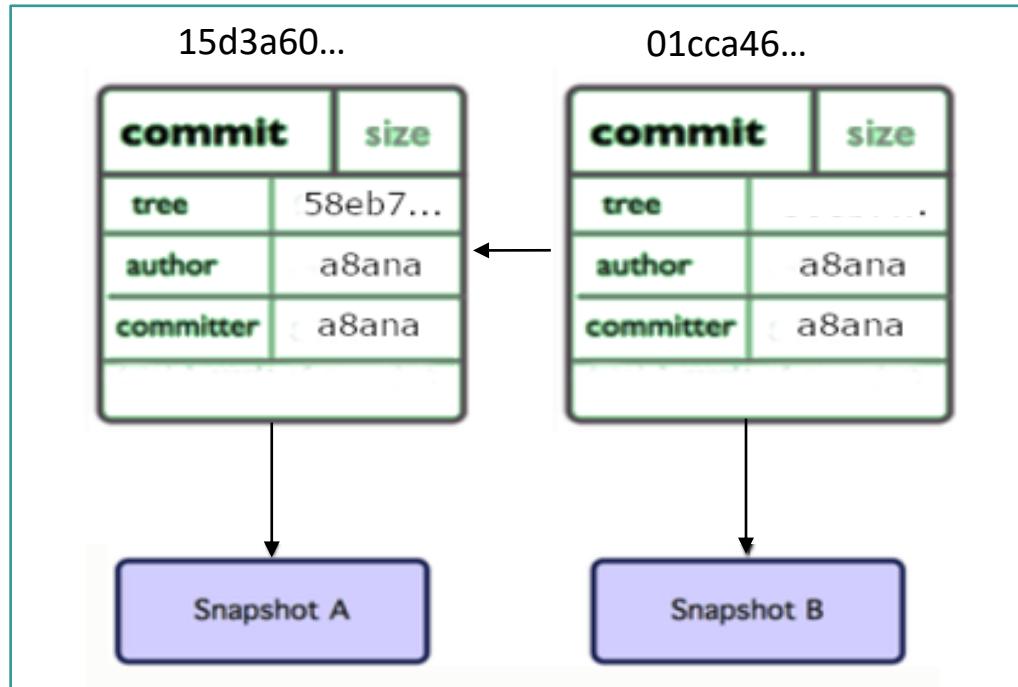


- Το Git σε κάθε commit υπολογίζει τα SHA-1 hashes με βάση το store, όπου $\text{store} = \text{Header} + \text{Content}$ ενώ επίσης κάνει compress το store με τη χρήση της Zlib και μετά αποθηκεύει



Merkle Trees

Version Control με Git / GitHub



- Αυτή η λογική των Merkle Trees και σύνδεσης με αντίστροφες λίστες μονής κατεύθυνσης είναι και ο μηχανισμός του Blockchain



- Όπως το git repository είναι ένα chain of blocks (blockchain) όπου κάθε block έχει ένα header και ένα payload (data) καθώς και μία αναφορά στο προηγούμενο block, και αυτό το repository είναι distributed έτσι και η τεχνολογία Distributed Ledger Technology (DLT) είναι ένα distributed ledger of changes
- Σε αυτή την τεχνολογία βασίζονται οι τεχνολογίες blockchain όπου στη θέση του Payload βρίσκονται NFTs (Non-Fungible Tokens) δηλαδή οτιδήποτε ψηφιακό αγαθό κατέχουμε



Git log --stat στο τελευταίο commit (1)

Version Control με Git / GitHub

```
a8ana@thanassis-PC MINGW64 /C/Users/a8ana/OneDrive/C  
$ git log --stat 470d6bf  
commit 470d6bfa02e9a12e29c91809abc4246b0490bfd3  
Author: a8ana <a8anassis@gmail.com>  
Date: Sun Dec 4 19:53:08 2022 +0200  
  
    Second Commit  
  
newFile.txt | 2 +-  
1 file changed, 1 insertion(+), 1 deletion(-)  
  
commit 15d3a60bd83e519fe65aed550b1d86759a96cc1f  
Author: a8ana <a8anassis@gmail.com>  
Date: Sun Dec 4 19:46:46 2022 +0200  
  
    first commit  
  
newFile.txt      | 1 +  
version1File.txt | 1 +  
version2File.txt | 1 +  
3 files changed, 3 insertions(+)  
|
```

- Εμφανίζουμε το ιστορικό (Git History) με git -log --stat στο τελευταίο commit
- Με --stat εμφανίζονται και στατιστικά των αλλαγών (insertions, deletions)



Git log στο τελευταίο commit

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive  
$ git log 470d6bf  
commit 470d6bfa02e9a12e29c91809abc4246b0490bfd3  
Author: a8ana <a8anassis@gmail.com>  
Date:   Sun Dec 4 19:53:08 2022 +0200  
  
        Second Commit  
  
commit 15d3a60bd83e519fe65aed550b1d86759a96cc1f  
Author: a8ana <a8anassis@gmail.com>  
Date:   Sun Dec 4 19:46:46 2022 +0200  
  
        first commit
```

- Και πιο απλά το commit history χωρίς στατιστικά για τις αλλαγές μεταξύ των commits, το παίρνουμε με git log στο τελευταίο commit



References

- Πως όμως είναι δυνατό να θυμόμαστε το SHA-1 hash του τελευταίου commit;
- Θα ήταν ευκολότερο αν αποθηκεύαμε το SHA-1 του τελευταίου commit σε ένα αρχείο με ένα απλό όνομα
- Αυτά τα απλά ονόματα που δείχνουν σε commits ονομάζονται references ή refs και βρίσκονται στο directory .git/refs



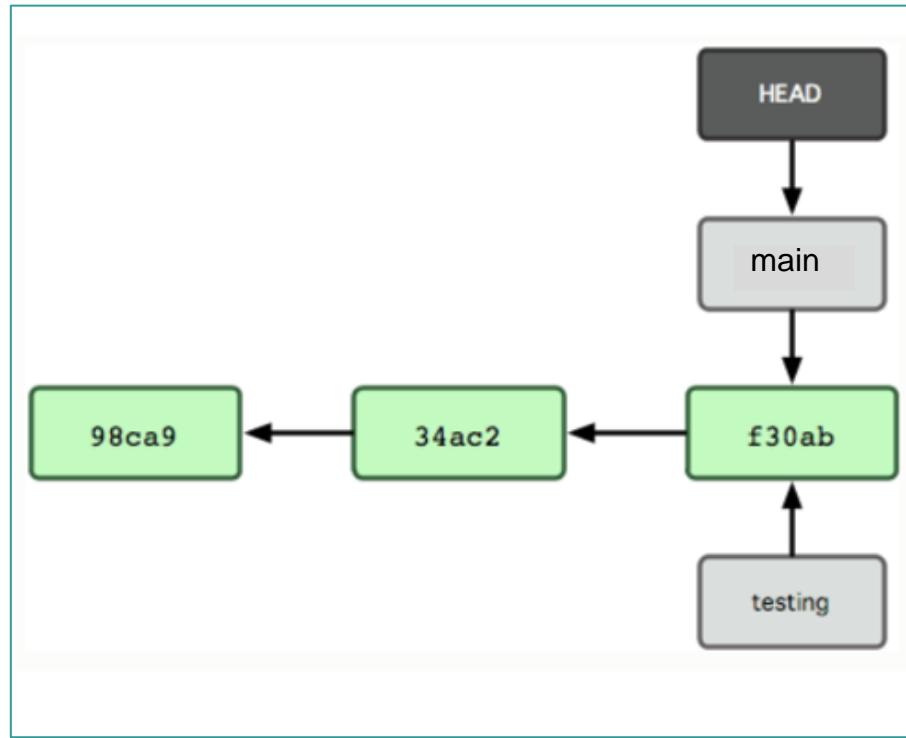
Branches

- Στο Git ένα branch είναι ένα version του Project. Τεχνικά ένα reference στο τελευταίο commit
- Αν όμως έχουμε πολλά branches, πως γνωρίζει το Git σε ποιο branch είμαστε, ποιο είναι δηλαδή το working directory;
- Η απάντηση είναι το HEAD file. Το HEAD file είναι ένα symbolic reference, δηλαδή ένας δείκτης σε ένα άλλο reference
- Επομένως το HEAD δείχνει στο τρέχον branch



HEAD και branch names

Version Control με Git / GitHub



- Αν έχουμε δύο branches, main και testing, και το τρέχον branch (checkout branch) είναι το main, τότε το HEAD δείχνει στο main



Tags

- Ένα tag είναι ένα reference που δείχνει σε ένα commit.
- Είναι σαν το branch αλλά δεν δείχνει μόνο στο τελευταίο commit και επίσης δεν μετακινείται, απλά 'σημαδεύει' και δίνει ένα πιο φιλικό όνομα σε ένα commit



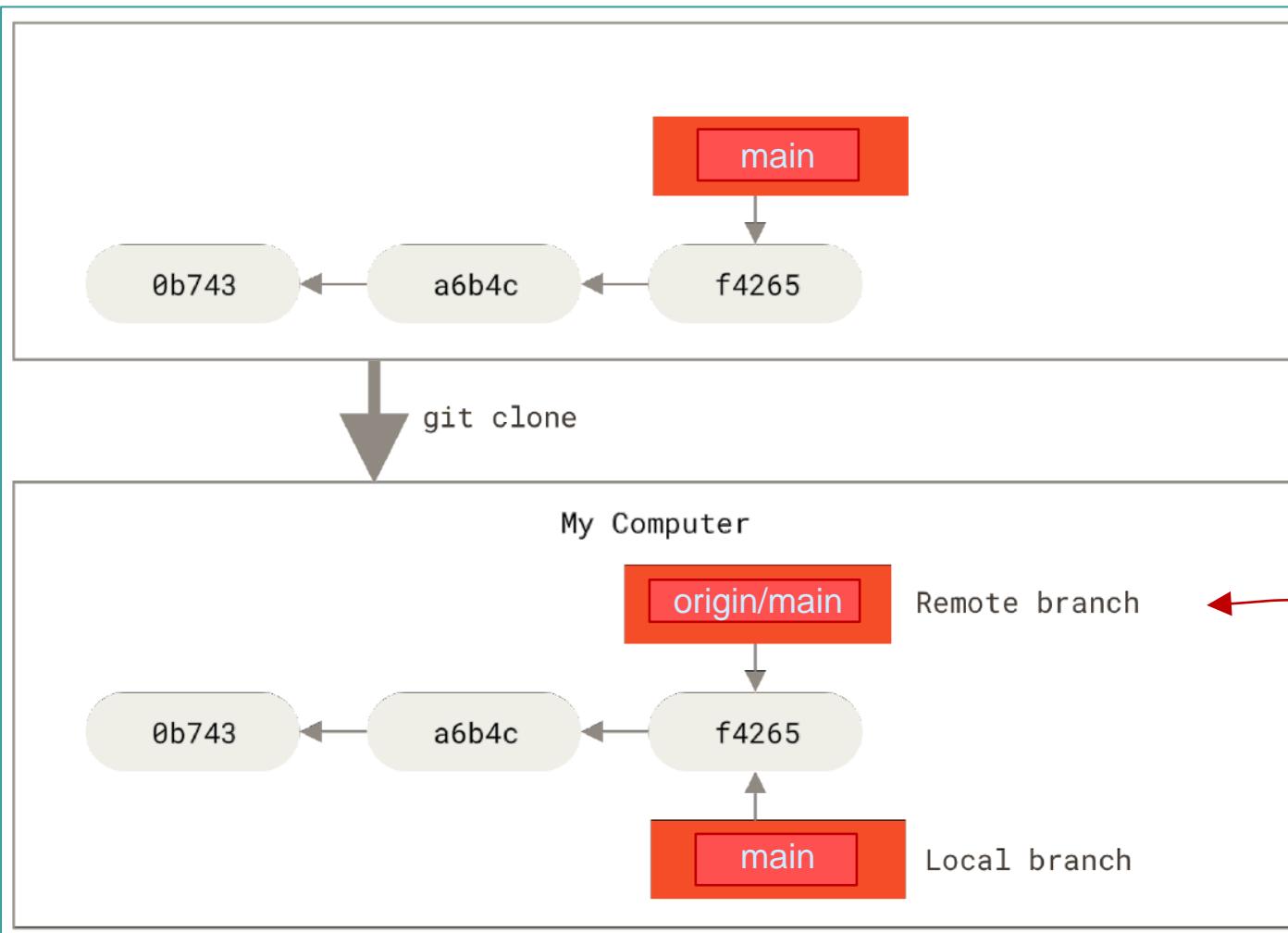
Remotes

- Στα remote references (π.χ. origin/main) (refs/remotes/origin/main) το git αποθηκεύει το SHA-1 hash του branch (π.χ. main) την τελευταία φορά που ανεβάσαμε (push) ή κατεβάσαμε (fetch, pull) ένα line of work (version) στον origin server (π.χ. origin main)
- Γενικά υπάρχουν τρεις τύποι branches: τα remote branches που είναι στον server, τα local branches που είναι στον τοπικό υπολογιστή και τα remote (tracking) branches, που είναι στον τοπικό υπολογιστή και 'δείχνουν' στον remote
- Ενημερώνονται κάθε φορά που κάνουμε push, fetch, pull (fetch + merge), clone



Remote (tracking) branch

Version Control με Git / GitHub



- Remote tracking branch



Plumbing vs Porcelain

Version Control με Git / GitHub

- Όλα τα προηγούμενα είναι οι λεγόμενες Git plumbing commands (που χρησιμοποιούνταν πριν την έκδοση 1.5), και δεν είναι user friendly
- Τις παρουσιάσαμε απλά για να δούμε τα Git internals και να εξοικειωθούμε με τα inner workings του Git
- To Git μας παρέχει πλέον ένα πολύ πιο απλό interface με user-friendly commands (porcelain commands), τις οποίες θα δούμε στα επόμενα



Μετατροπή φάκέλου σε repo -- git init

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /c/Users
$ cd /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects
$ cd codingfactorygit/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit
$ |
```

- Έστω ένα dir που περιέχει ένα Web Project, έστω με όνομα codingfactorygit
- Κάνουμε cd και μετακινούμαστε στον παραπάνω φάκελο



git init

```
a8ana@thanassis-pc MINGW64 /c/Users/a8an  
$ cd codingfactorygit/  
  
a8ana@thanassis-pc MINGW64 /c/Users/a8an  
$ git init  
Initialized empty Git repository in c:/u
```

```
a8ana@thanassis-pc MINGW64 /c/Users/a8ana/OneDrive/c  
$ ls -la  
total 48  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:23 ./  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 ../  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:23 .git/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter1/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter2/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter3/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter4/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter5/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter6/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 img/  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 testbed/
```

- Δίνοντας *git init* μέσα στον φάκελο μετατρέπουμε τον φάκελο σε *git folder*
- Παρατηρούμε τον φάκελο *.git* που δημιουργήθηκε μετά το *git init*



GitHub – Νέο Repository (1)

Version Control με Git / GitHub

The screenshot shows the GitHub user interface. At the top, there is a navigation bar with links for Pull requests, Issues, Codespaces, Marketplace, and Explore. Below the navigation bar, there are tabs for Overview, Repositories (26), Projects, Packages, and Stars. The Overview tab is selected. In the center, there is a section for Popular repositories, showing two repos: "hello-world" (Public, 1 star) and "HelloSev" (My First Repo, Java, 1 star). On the right side, there is a context menu triggered by a '+' icon, listing options: New repository, Import repository, New codespace, New gist, New organization, and New project. A link to 'your pins' is also visible.

- Στη συνέχεια, στο GitHub δημιουργούμε νέο repository με όνομα codingfactorygit (δεν είναι όμως απαραίτητο τα ονόματα των repositories να είναι τα ίδια στο GitHub και στο τοπικό μας repo)



GitHub – Νέο Repository (2)

Version Control με Git / GitHub

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 *

a8anassis / codingfactorygit

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

Description (optional)

Coding Factory Oct 2022 First Git Project

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.](#)

.gitignore template: None ▾

Choose a license

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

License: None ▾

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

You are creating a public repository in your personal account.

Create repository

- Δίνουμε το **όνομα** του repository, μία προαιρετική περιγραφή, αφήνουμε **public** repository εφόσον θέλουμε να έχουν πρόσβαση όλοι, αλλιώς **private** και επίσης επιλέγουμε Add a README file, ένα αρχείο κειμένου που συνηθίζουμε να περιλαμβάνουμε σε κάθε project με μία περιγραφή του project
- Στο τέλος αναφέρεται ως **default branch** το **main**.



GitHub Repository

Version Control με Git / GitHub

The screenshot shows a GitHub repository page for 'a8anassis/codingfactorygit'. The repository is public and contains one branch ('main') and one tag ('0 tags'). A single commit was made by 'a8anassis' titled 'Initial commit' with hash '4ad6d99' and timestamp '22 seconds ago'. The README.md file is present with the same content. The repository has 0 stars, 1 watching, and 0 forks. The sidebar includes sections for About, Releases, and a link to Create a new release.

- Έτσι δημιουργήσαμε το remote repository στο GitHub



Main vs master branch

Version Control με Git / GitHub

- Τα branches είναι ανεξάρτητα μονοπάτια, που περιέχουν τα αρχεία του repository και όπου μπορούμε να κάνουμε αλλαγές που δεν επηρεάζουν τα άλλα branches
- Για πολλά χρόνια πριν τον Οκτώβριο 2020 το βασικό default branch του GitHub ήταν το master. Από τον Οκτώβριο 2020 το όνομα άλλαξε σε main και κάθε νέο repository που δημιουργούμε στο GitHub έχει ως default branch το main



GitHub – codingfactory repo

Version Control με Git / GitHub

- Αφού δημιουργήσαμε στο GitHub ένα repository, τώρα πάμε πάλι στο τοπικό μας git repository (local repository) για να συνδέσουμε το local (τοπικό) με το remote (GitHub) repository
- Αντιγράφουμε πρώτα το URL του remote repository στο GitHub, είτε το HTTPS όταν έχουμε απλό username/password auth ή το SSH αν έχουμε SSH authentication

The screenshot shows two side-by-side views of a GitHub repository's clone options. Both views include the following elements:

- Top navigation bar: Go to file, Add file, and a green <> Code button.
- Clone section: Local, Codespaces (New).
- Clone methods: Clone (with a dropdown arrow), a question mark icon, and a copy icon.
- Protocol buttons: HTTPS, SSH, GitHub CLI.
- Clone URL: A text input field containing the URL for each protocol.
 - Left view: https://github.com/a8anassis/codingfactorygit
 - Right view: git@github.com:a8anassis/codingfactorygit.git
- Instructions below the URLs:
 - Left view: Use Git or checkout with SVN using the web URL.
 - Right view: Use a password-protected SSH key.



Σύνδεση τοπικού με remote

Version Control με Git / GitHub

```
MINGW64:/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit  
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT  
$ git remote add origin git@github.com:a8anassis/codingfactorygit.git  
  
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactor  
$ git remote -v  
origin  git@github.com:a8anassis/codingfactorygit.git (fetch)  
origin  git@github.com:a8anassis/codingfactorygit.git (push)  
  
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactor  
$ |
```

git remote add <name> <URL>

- Συνδέουμε το remote repository με το τοπικό με την git remote add δίνοντας ένα όνομα στη σύνδεση, συνήθως origin αλλά μπορεί να είναι και άλλο όνομα. Στη συνέχεια ακολουθεί το URL που αντιγράψαμε
- Για να κάνουμε verify ότι έγινε η σύνδεση δίνουμε git remote -v



git pull origin main

Version Control με Git / GitHub

- Σε αυτή τη χρονική στιγμή τα δύο branches, origin main και το τοπικό main δεν είναι sync γιατί το remote repo έχει κάνει commit ένα αρχείο, το README.md ενώ το local repo δεν έχει κάνει κανένα αρχείο commit
- Για να είναι sync τα δύο repos θα πρέπει να κατεβάσουμε το remote main branch και τα περιεχόμενά του τοπικά με την εντολή **git pull origin main**
- Με git pull μπορούμε επίσης να κατεβάσουμε τις επόμενες φορές εφόσον έχει δημιουργηθεί το remote tracking branch



git pull origin main (1)

Version Control με Git / GitHub

```
MINGW64:/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codin
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFacto
$ git pull origin main
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 642 bytes | 58.00 KiB/s, done.
From github.com:a8anassis/codingfactorygit
 * branch           main      -> FETCH_HEAD
 * [new branch]     main      -> origin/main

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFacto
$ |
```

- Η **git pull origin main** φέρνει τα περιεχόμενα του main branch από το GitHub. Εδώ τα περιεχόμενα είναι μόνο το README.md, το οποίο και αντιγράφεται από το origin main στο τρέχον branch που είναι το main



git pull origin main (2)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/C
$ ls -la
total 49
drwxr-xr-x 1 a8ana 197609 0 Dec  3 22:08 .
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 ../
drwxr-xr-x 1 a8ana 197609 0 Dec  3 22:08 .git/
-rw-r--r-- 1 a8ana 197609 64 Dec  3 22:08 README.md
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter1/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter2/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter3/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter4/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter5/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter6/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 img/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 testbed/
```

- Παρατηρούμε ότι έχει κατέβει το main branch από το remote και έχει ενσωματωθεί (merge) το README.md στο local main branch
- Ουσιαστικά δηλαδή η git pull κάνει δύο πράξεις: 1) git fetch, όπου κατεβάζει το remote main branch και 2) git merge, όπου συγχωνεύει τις 'διαφορές' στο main branch



git pull origin main (3)

Version Control με Git / GitHub

- Όπως αναφέραμε, η pull κάνει fetch το branch και στη συνέχεια merge
- Ωστόσο, επειδή το τοπικό main branch δεν έχει κάποια έκδοση του README.md το merge είναι trivial, απλά παραμένει η έκδοση που έγινε fetch και τα δύο branches, local και remote γίνονται sync



git pull origin main (4)

Version Control με Git / GitHub

```
MINGW64:/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codin
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFacto
$ git pull origin main
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 642 bytes | 58.00 KiB/s, done.
From github.com:a8anassis/codingfactorygit
 * branch           main      -> FETCH_HEAD
 * [new branch]     main      -> origin/main

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFacto
$ |
```

- Παρατηρούμε επίσης ότι έχει δημιουργηθεί ένα νέο branch το origin/main που είναι local branch και είναι στην πραγματικότητα remote tracking branch, ένα τοπικό αντίγραφο του origin main
- Επίσης, υπάρχει και το FETCH_HEAD, ένα reference στο tip του origin/main branch, δηλαδή ένας δείκτης στο τελευταίο commit του origin/main branch



FETCH_HEAD

Version Control με Git / GitHub

- Στην πραγματικότητα το **FETCH_HEAD** είναι το όνομα ενός αρχείου στο **.git folder** που περιέχει το hash value του τελευταίου commit στο origin/main branch

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codin
$ cd .git/

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codin
$ ls -la
total 21
drwxr-xr-x 1 a8ana 197609  0 Dec  3 22:08 .
drwxr-xr-x 1 a8ana 197609  0 Dec  3 22:08 ..
-rw-r--r-- 1 a8ana 197609  97 Dec  3 22:08 FETCH_HEAD
-rw-r--r-- 1 a8ana 197609  21 Dec  3 19:23 HEAD
-rw-r--r-- 1 a8ana 197609  246 Dec  3 21:53 config
-rw-r--r-- 1 a8ana 197609   73 Dec  3 19:23 description
drwxr-xr-x 1 a8ana 197609  0 Dec  3 19:23 hooks/
-rw-r--r-- 1 a8ana 197609  137 Dec  3 22:08 index
drwxr-xr-x 1 a8ana 197609  0 Dec  3 19:23 info/
drwxr-xr-x 1 a8ana 197609  0 Dec  3 22:08 logs/
drwxr-xr-x 1 a8ana 197609  0 Dec  3 22:08 objects/
drwxr-xr-x 1 a8ana 197609  0 Dec  3 22:08 refs/
```



cat FETCH_HEAD

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/.git (GIT_D
$ cat FETCH_HEAD
4ad6d9969537fb009d32c894d5cb422c00fa5670           branch 'main' of github.com:a8anassis/codingfactorygit

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/.git (GIT_D
$ |
```

- Με cat στο FEATCH_HEAD βλέπουμε ότι περιέχει το hash value του τελευταίου commit στο origin/main branch



cat <αρχείο>

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDri
$ cd ..

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDri
$ git branch -a
* main
  remotes/origin/main

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDri
$ cat README.md
# codingfactorygit
Coding Factory Oct 2022 First Git Project

a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDri
$
```

- Με git branch -a εμφανίζονται όλα τα τοπικά branches, που τώρα είναι το main αλλά και το remotes/origin/main
- Με cat εμφανίζουμε τα περιεχόμενα του αρχείου README.md, που είναι πλέον στο τοπικό main branch



Remotes/origin/main (1)

Version Control με Git / GitHub

MINGW64:/C/Users/a8ana/OneDrive/CodingFactory-REBOOT/webproject

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codi  
$ cd .git/
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codi  
$ ls -la  
total 21  
drwxr-xr-x 1 a8ana 197609 0 Dec 3 22:08 ./  
drwxr-xr-x 1 a8ana 197609 0 Dec 3 22:08 ../  
-rw-r--r-- 1 a8ana 197609 97 Dec 3 22:08 FETCH_HEAD  
-rw-r--r-- 1 a8ana 197609 21 Dec 3 19:23 HEAD  
-rw-r--r-- 1 a8ana 197609 246 Dec 3 21:53 config  
-rw-r--r-- 1 a8ana 197609 73 Dec 3 19:23 description  
drwxr-xr-x 1 a8ana 197609 0 Dec 3 19:23 hooks/  
-rw-r--r-- 1 a8ana 197609 137 Dec 3 22:08 index  
drwxr-xr-x 1 a8ana 197609 0 Dec 3 19:23 info/  
drwxr-xr-x 1 a8ana 197609 0 Dec 3 22:08 logs/  
drwxr-xr-x 1 a8ana 197609 0 Dec 3 22:08 objects/  
drwxr-xr-x 1 a8ana 197609 0 Dec 3 22:08 refs/
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codi  
$ cd refs/
```

```
a8ana@thanassis-pc MINGW64 /C/Users/a8ana/OneDrive/Codi  
$ ls  
heads/ remotes/ tags/
```

- Το main είναι τεχνικά ένα αρχείο με ένα reference στο τελευταίο commit
- Πάμε στο φάκελο refs/remotes/origin



Remotes/origin/main (2)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 /c/Users/a8ana/OneDrive  
DIR!)  
$ ls -la  
total 0  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 22:08 ./  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 22:08 ../  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 22:08 origin/  
  
a8ana@thanassis-pc MINGW64 /c/Users/a8ana/OneDrive  
DIR!)  
$ cd origin/  
  
a8ana@thanassis-pc MINGW64 /c/Users/a8ana/OneDrive  
n (GIT_DIR!)  
$ ls -la  
total 1  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 22:08 ./  
drwxr-xr-x 1 a8ana 197609 0 Dec  3 22:08 ../  
-rw-r--r-- 1 a8ana 197609 41 Dec  3 22:08 main  
  
a8ana@thanassis-pc MINGW64 /c/Users/a8ana/OneDrive  
n (GIT_DIR!)  
$ cat main  
4ad6d9969537fb009d32c894d5cb422c00fa5670
```

- Πάμε στον φάκελο *origin* και στο αρχείο *main* όπου με cat βλέπουμε το reference που είναι ίδιο με το `FETCH_HEAD` μιας και το `FETCH_HEAD` σε αυτή τη στιγμή έδειχνε στο τελευταίο commit του *main branch*



README.md (1)

Version Control με Git / GitHub

main ▾ 1 branch 0 tags Go to file Add file ▾ Code ▾

a8anassis Initial commit 4ad6d99 1 hour ago 1 commit

README.md Initial commit 1 hour ago

README.md

codingfactorygit

Coding Factory Oct 2022 First Git Project

A red circle highlights the edit icon (pencil) next to the README.md file entry.

- Το αρχείο README.md περιέχει μία περιγραφή του Project. Αν πατήσουμε το Edit icon μεταβαίνουμε στα περιεχόμενα του README



README.md (2)

Version Control με Git / GitHub

The screenshot shows a GitHub repository page for 'a8anassis / codingfactorygit'. The repository is public. The main navigation tabs are 'Code' (selected), 'Issues', 'Pull requests', 'Actions', and 'Projects'. Below the tabs, the repository name 'codingfactorygit' and the file path 'README.md' are shown, along with a 'main' branch indicator. A preview button is visible. The content of the README.md file is displayed below:

```
1 # codingfactorygit
2 Coding Factory Oct 2022 First Git Project
3
```

- Τα περιεχόμενα είναι authored με mark-up γλώσσα
- Παρατηρούμε το # που είναι Heading 1
- Βλ. την επόμενη διαφάνεια για το Markdown syntax



Markdown Syntax

Version Control με Git / GitHub

Element	Markdown Syntax
Heading	# H1 ## H2 ### H3
Bold	bold text
Italic	<i>italicized text</i>
Blockquote	>blockquote
Ordered List	1. First item 2. Second item 3. Third item
Unordered List	- First item - Second item - Third item
Code	`code`
Horizontal Rule	---
Link	[title](https://www.example.com)
Image	![alt text](image.jpg)

- Η Markdown είναι μία γλώσσα mark-up για να μορφοποιούμε απλό κείμενο (plain text) όπως το αρχείο README.md



git status (Staging area)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    chapter1/
    chapter2/
    chapter3/
    chapter4/
    chapter5/
    chapter6/
    img/
    testbed/
nothing added to commit but untracked files present (use "git add" to track)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ |
```

- Με git status βλέπουμε τα περιεχόμενα της staging area, που εδώ δεν έχει τίποτα (με κόκκινο είναι τα untracked αρχεία και φάκελοι) μιας και δεν έχουμε κάνει τίποτα add



Git Status

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    chapter1/
    chapter2/
    chapter3/
    chapter4/
    chapter5/
    chapter6/
    img/
    testbed/

nothing added to commit but untracked files present (use "git add" to track)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ |
```

- Παρατηρούμε ωστόσο πως το README δεν είναι μέσα στα untracked files
- Ο λόγος είναι πως η git pull έφερε το commit object με τις αλλαγές που είχε το README.md, επομένως και το τοπικό main branch έχει πλέον ένα commit



Git pull & Commit (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codin
$ git log main
commit 4ad6d9969537fb009d32c894d5cb422c00fa5670 (HEAD -> main, origin/main)
Author: Athanassios Androutsos <a8anassis@gmail.com>
Date:   Sat Dec 3 21:44:58 2022 +0200

    Initial commit

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codin
$ |
```

- Παρατηρούμε πως με git log main -δηλαδή βλέποντας το ιστορικό του τοπικού main branch- βλέπουμε ότι έχει έρθει το commit από το remote στο local (τοπικό main branch) με την git pull
- Είναι το commit που έχει έρθει με την git pull



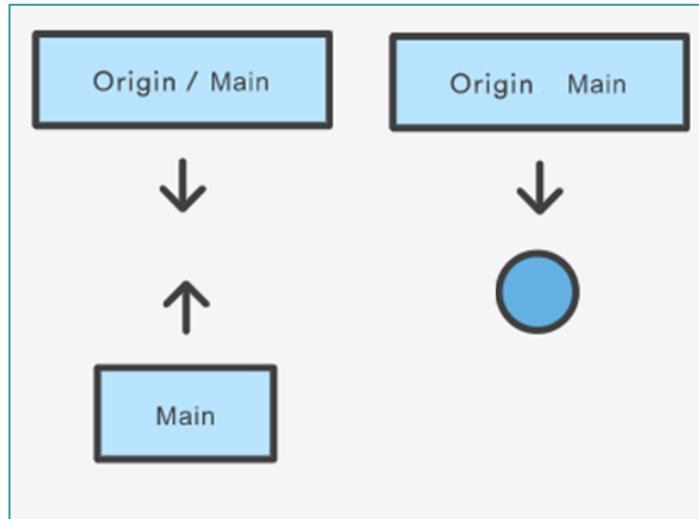
Git pull & Commit (2)

Version Control με Git / GitHub

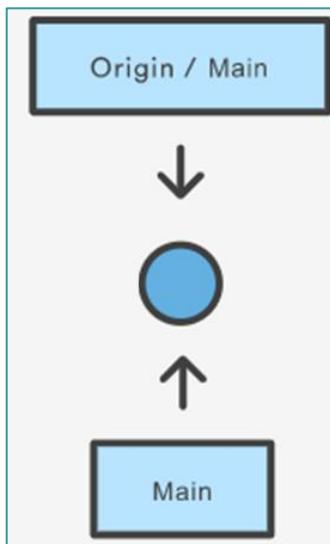
- Η git pull είναι στην πραγματικότητα συνδυασμός δύο εντολών: git fetch και git merge
- Η git fetch κάνει download το upstream content και αμέσως κάνει update το τοπικό repository
- Θα κατέβει όλο το commit history που δεν υπάρχει στο τοπικό branch (στο branch που δείχνει το HEAD (checkout branch)) και το HEAD θα δείξει στο τελευταίο commit object



Git pull



- Αρχικά μόνο το upstream (Origin main στο GitHub) είχε ένα initial commit (μπλε κύκλος), ενώ στο local, τα main και origin/main (remote tracking branch) δεν είχαν κανένα commit
- Μετά το git pull το main και το origin/main θα δείχνουν στο νέο commit object που θα περιέχει το README.md όπως το κατεβάσαμε από το remote ενώ επίσης θα έχει γίνει update και το origin/main (remote tracking branch) που θα δείχνει επίσης στο νέο commit object





GitHub νέο version 3

Version Control με Git / GitHub

a8anassis / codingfactorygit Public

Code Issues Pull requests Actions

codingfactorygit / README.md in main

Edit file Preview

```
1 # codingfactorygit
2 Coding Factory Oct 2022 First Git Project
3 *Version 3!*
4
```

- Πάμε στο GitHub και αλλάζουμε το README.md σε ένα νέο version 3



Commit changes

Version Control με Git / GitHub

The screenshot shows a 'Commit changes' dialog box. At the top left is a circular profile picture of a man. The main title is 'Commit changes'. Below it is a file list: 'README v0.3'. A large text input field says 'Add an optional extended description...'. At the bottom, there are two radio button options: one selected ('Commit directly to the main branch.') and one unselected ('Create a new branch for this commit and'). At the very bottom are two buttons: a green 'Commit changes' button and a white 'Cancel' button.

- Κάνουμε commit αλλαγές στο main branch



Git pull origin main

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory
$ git pull origin main
From github.com:a8anassis/codingfactorygit
 * branch           main            -> FETCH_HEAD
Updating 4ad6d99..42b8bc4
Fast-forward
 README.md | 1 +
 1 file changed, 1 insertion(+)
```

- Κάναμε pull τις αλλαγές με το version 3 του README.md



git log main

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/coding1
$ git log main
commit 42b8bc4baa4cdca141a8ed4e257cf6153d822519 (HEAD -> main, origin/main)
Author: Athanassios Androutsos <a8anassis@gmail.com>
Date:   Wed Dec 7 15:17:54 2022 +0200

    README v0.3

commit 4ad6d9969537fb009d32c894d5cb422c00fa5670
Author: Athanassios Androutsos <a8anassis@gmail.com>
Date:   Sat Dec 3 21:44:58 2022 +0200

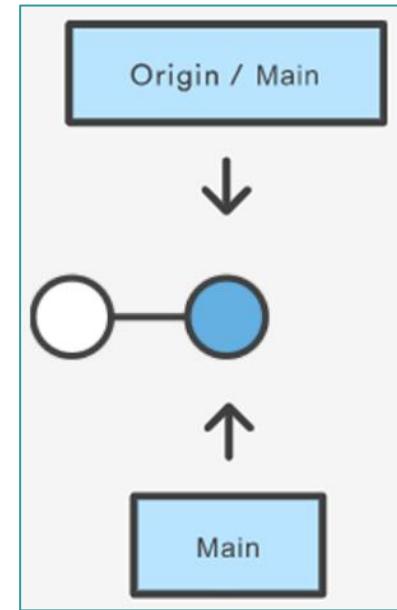
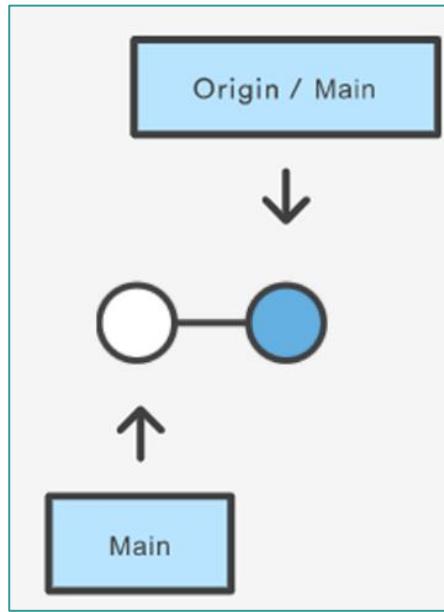
    Initial commit

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/coding1
```

- Βλέπουμε ότι έχει έρθει και το 2^o commit και τώρα το main δείχνει στο καινούργιο commit



Git pull

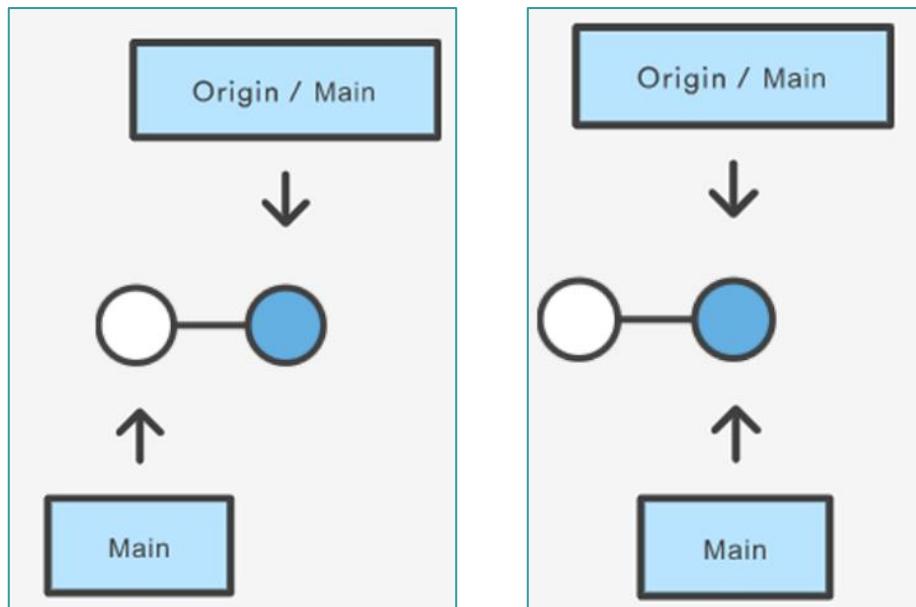


- Η git pull πρώτα κάνει git fetch και αφού κατεβάσει το νέο README.md και κάνει update το local repository μετακινεί τον origin / main να δείχνει στο νέο commit (Εικόνα 1)
- Στη συνέχεια γίνεται git merge στο checkout branch που είναι το main (από όπου κάναμε το git pull) και μετακινείται και το main στη νέα θέση (Εικόνα 2)



Fast Forward

Version Control με Git / GitHub



- Όταν κάνουμε merge δύο commits που το ένα είναι πίσω από το άλλο, το git απλά μετακινεί τον δείκτη του current branch μπροστά
- Αυτό ονομάζεται Fast-Forward (FF)
- Άν στο merge δώσουμε --no-ff τότε δημιουργείται commit object για το merge και δεν γίνεται απλά fast forward



Αλλαγές στο τοπικό README

Version Control με Git / GitHub

```
a8ana@thanassis-pc ~ % $ vi README.md |
```

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/wi
# codingfactorygit
Coding Factrory Oct 2022 First Git Project
*Version 3!*
**Version 4**|
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/Codin
$ cat README.md
# codingfactorygit
Coding Factrory Oct 2022 First Git Project
*Version 3!*
**Version 4**|
```

- Αλλάζουμε σε Version 4



Modified file

```
a8ana@thanassis-pc MINGW64 ~/One  
$ git status  
On branch main  
Changes not staged for commit:  
  (use "git add <file>..." to update what will be committed)  
    (use "git restore <file>..." to discard changes in working directory)  
           modified:   README.md  
  
Untracked files:  
  (use "git add <file>..." to include in what will be committed)  
    chapter1/  
    chapter2/  
    chapter3/  
    chapter4/  
    chapter5/  
    chapter6/  
    img/  
    testbed/
```

- Το Git εντοπίζει ότι το README που είναι ήδη tracked έχει γίνει modify



git add <filename>

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webpro
$ git add README.md

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webpro
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    chapter1/
    chapter2/
    chapter3/
    chapter4/
    chapter5/
    chapter6/
    img/
    testbed/
```

Με git –add
README.md
εισάγουμε το
αρχείο
README.md
στην Staging
area



git commit με message

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDri
$ git commit -m "README v0.4"
[main d9c72bf] README v0.4
 1 file changed, 1 insertion(+)
```

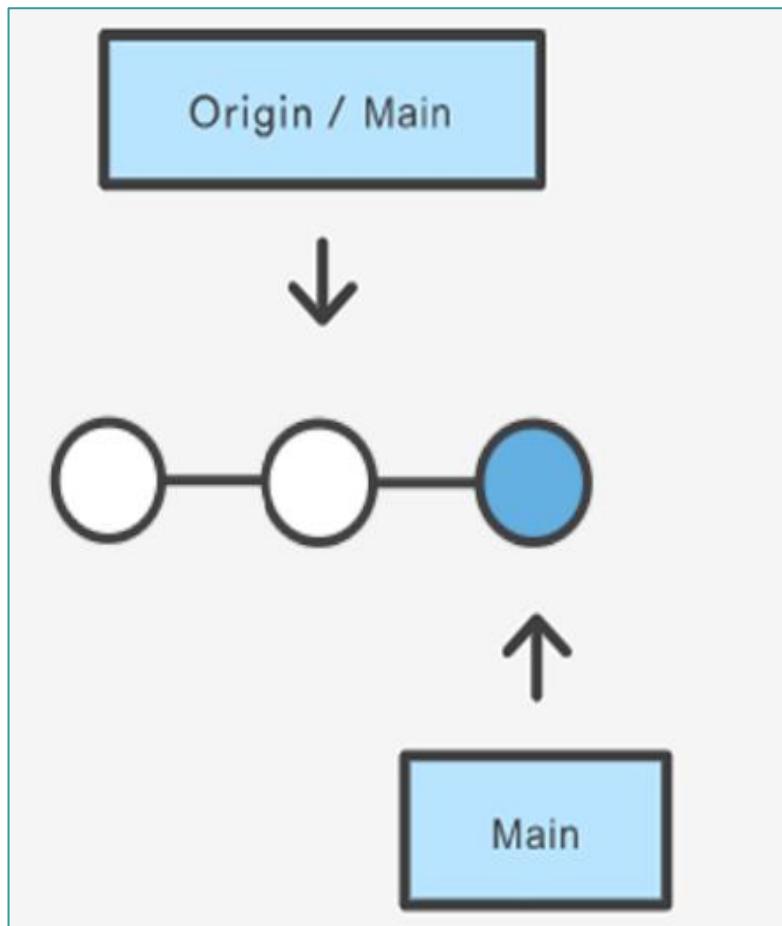
```
a8ana@thanassis-pc MINGW64 ~/OneDri
$
```

- Με `git commit -m "..."` αποστέλλουμε ότι υπάρχει στο staging area στην ΒΔ του Git (Git repository)



Commit στο local branch

Version Control με Git / GitHub



- Μετά το commit τα δύο histories διαφέρουν (unrelated histories). Το main branch έχει μετακινηθεί στο νέο commit, ενώ το remote tracking branch δεν έχει μετακινηθεί



git push origin main (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 283 bytes | 283.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:a8anassis/codingfactorygit.git
  42b8bc4..d9c72bf  main -> main
```

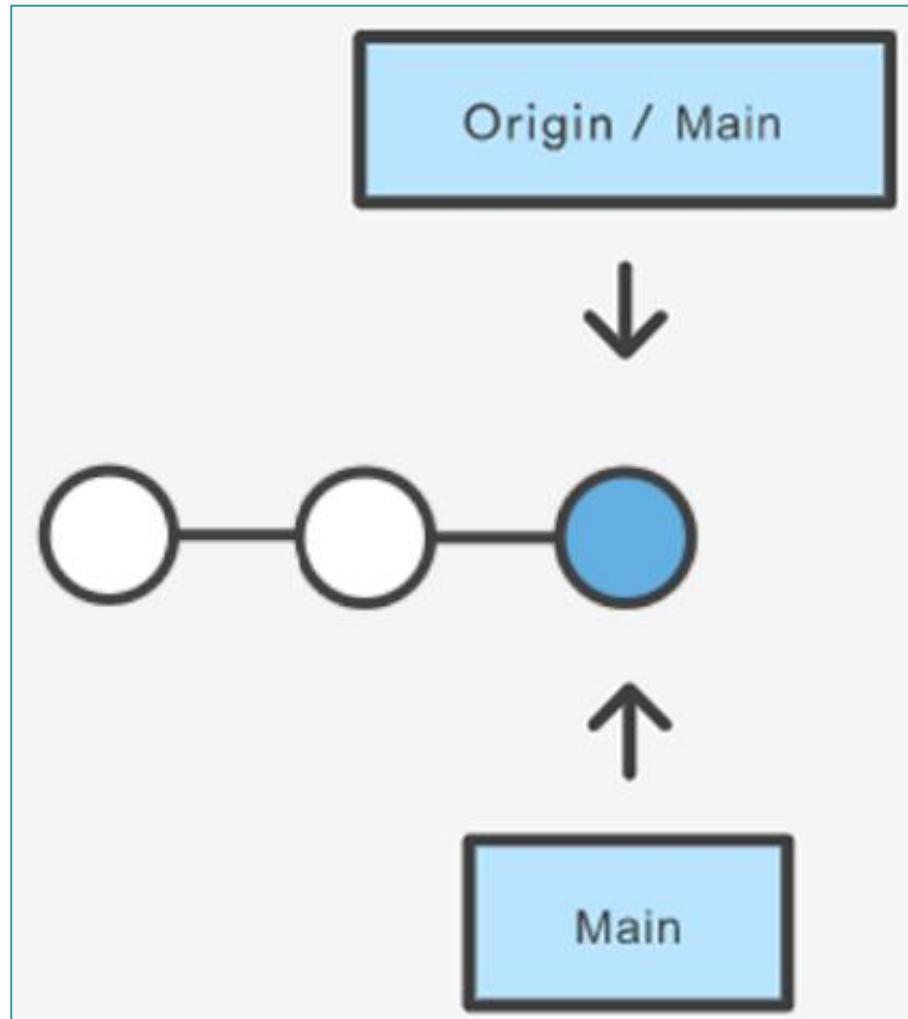
```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects
$ |
```

- Με git push ανεβάζουμε στο remote branch (εδώ origin main)



git push origin main (2)

Version Control με Git / GitHub



- Με git push ανεβάζουμε στο GitHub και ενημερώνεται και το remote tracking branch (origin / main) και το GitHub



GitHub

Version Control με Git / GitHub

github.com/a8anassis/codingfactorygit

Search or jump to... / Pull requests Issues Codespaces

a8anassis / codingfactorygit Public

Code Issues Pull requests Actions Projects Wiki

main 1 branch 0 tags

a8anassis README v0.4

README.md README v0.4

README.md

codingfactorygit

Coding Factory Oct 2022 First Git Project Version 3! Version 4

- Με refresh στο GitHub παρατηρούμε το commit και τις αλλαγές



Προσθήκη στην staging area

Version Control με Git / GitHub

Εντολή	New Files	Modified Files	Deleted Files	Περιγραφή
git add -A (git add --all)	✓	✓	✓	Stages όλα τα αρχεία του Working Dir, και τα deleted
git add	✓	✓	✓ ✗	Stages όλα τα νέα και updated αρχεία του τρέχοντος dir και subdirs, (για τα deleted εξαρτάται από το version του git)
Git add -u (git add --updated)	✗	✓	✓	Stages όλα τα modified & deleted αρχεία του Working Dir
Git add --ignore-removal	✓	✓	✗	Stages όλα, εκτός deleted του τρέχοντος dir και subdirs



Αφαίρεση από το Staging

Version Control με Git / GitHub

- `git rm --cached <filename>`
 - Αφαιρεί ένα αρχείο από το staging area. Αν το αρχείο είχε ήδη γίνει commit, τότε διαγράφεται και από το staging area και από το working dir.
- `git restore --staged <filename>`
 - Αφαιρεί ένα αρχείο με όνομα <filename> από το staging area. Δεν το διαγράφει από το working dir.
- `git reset`
 - Αφαιρεί όλα από το staging area



Σενάριο add, commit, push (1)

Version Control με Git / GitHub

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ cd chapter1

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ ls -la
total 32
drwxr-xr-x 1 a8ana 197609    0 Dec  3 19:11 .
drwxr-xr-x 1 a8ana 197609    0 Dec  7 16:29 ..
-rw-r--r-- 1 a8ana 197609  834 Oct 21 20:06 form1.html
-rw-r--r-- 1 a8ana 197609 1742 Oct 21 20:42 form2.html
-rw-r--r-- 1 a8ana 197609  876 Oct 27 18:45 index.html
-rw-r--r-- 1 a8ana 197609  735 Oct 21 19:03 lists.html
-rw-r--r-- 1 a8ana 197609 1742 Oct 21 18:41 tables.html
-rw-r--r-- 1 a8ana 197609  852 Oct 27 18:45 video.html

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ./chapter2/
    ./chapter3/
    ./chapter4/
    ./chapter5/
    ./chapter6/
    ./img/
    ./testbed/
nothing added to commit but untracked files present (use "git add" to track)
```

- cd chapter1



Σενάριο add, commit, push (2)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git add .

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   form1.html
    new file:   form2.html
    new file:   index.html
    new file:   lists.html
    new file:   tables.html
    new file:   video.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ./chapter2/
    ./chapter3/
    ./chapter4/
    ./chapter5/
    ./chapter6/
    ./img/
    ./testbed/
```

- Git add



Σενάριο add, commit, push (3)

Version Control με Git / GitHub

- Commit & push

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/Codir
$ git commit -m "chapter-1 v0.1"
[main da31c7b] chapter-1 v0.1
 6 files changed, 246 insertions(+)
 create mode 100644 chapter1/form1.html
 create mode 100644 chapter1/form2.html
 create mode 100644 chapter1/index.html
 create mode 100644 chapter1/lists.html
 create mode 100644 chapter1/tables.html
 create mode 100644 chapter1/video.html
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/Codir
$ |
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/
$ git push origin main
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 3.25 KiB | 1.62 MiB/s, done.
Total 9 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To github.com:a8anassis/codingfactorygit.git
  d9c72bf..da31c7b  main -> main
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/
$ |
```



GitHub (1)

Version Control με Git / GitHub

The screenshot shows a GitHub repository page for the user 'a8anassis' with the repository name 'codingfactorygit'. The repository is public. At the top, there is a search bar and a 'Pull requests' button. Below the header, the repository name 'a8anassis / codingfactorygit' is displayed along with a 'Public' badge. A navigation bar below the header includes 'Code' (which is underlined in red), 'Issues', 'Pull requests', 'Actions', and a grid icon. Underneath, a dropdown menu shows 'main' is selected. It also displays '1 branch' and '0 tags'. The main content area lists the repository's contents: 'a8anassis chapter-1 v0.1' (with a profile picture icon), 'chapter1' (with a folder icon), and 'README.md' (with a file icon). To the right of these files are their respective versions: 'chapter-1 v0.1' and 'README v0.4'.

- Τα αρχεία και ο folder έχουν ανέβει στο GitHub



GitHub (2)

Version Control με Git / GitHub

main ▾ codingfactorygit / chapter1 /

a8anassis chapter-1 v0.1

..

form1.html	chapter-1 v0.1
form2.html	chapter-1 v0.1
index.html	chapter-1 v0.1
lists.html	chapter-1 v0.1
tables.html	chapter-1 v0.1
video.html	chapter-1 v0.1

- Τα περιεχόμενα του folder έχουν ανέβει



Modified files - Add & commit ταυτόχρονα

Version Control με Git / GitHub

- Αν θέλουμε να κάνουμε add και commit τις αλλαγές που έχουμε κάνει (update/delete) σε ήδη υπάρχοντα files (ήδη tracked files) που είναι modified ή deleted στο git repository, τότε μπορούμε με μια εντολή:
- **git commit -a -m "..."** ή
- **git commit -am "..."**



Modified

Version Control με Git / GitHub

```
a8ana@thanassis-pc ~  
$ vi form1.html |
```

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1  
<!-- Version 2 --&gt;<br/><!DOCTYPE html>  
<html lang="en">  
<head>  
    <meta charset="UTF-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title>Register</title>  
</head>  
<body>  
  
    <form>  
        <div>  
            <label for="username">Username</label>  
            <input type="text" id="username" placeholder="Enter username">  
        </div>  
        <div>  
            <label for="email">E-mail</label>  
            <input type="email" id="email" name="email">  
        </div>  
        <div>  
            <label for="message">Message</label>  
            <textarea id="message" cols="60" rows="10"></textarea>  
        </div>  
        <div>  
            <input type="submit" value="Submit">  
        </div>  
  
    </form>  
</body>  
</html>
```

Κάναμε μία αλλαγή σε ένα ήδη tracked αρχείο, οπότε έγινε modified



Stage & Commit μαζί

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   form1.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ./chapter2/
    ./chapter3/
    ./chapter4/
    ./chapter5/
    ./chapter6/
    ./img/
    ./testbed/

no changes added to commit (use "git add" and/or "git commit -a")

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git commit -a -m "form1 v0.2"
```

- Stage & commit με `git commit -a -m "..."`



Push changes

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git commit -a -m "form1 v0.2"
[main 95fc326] form1 v0.2
 1 file changed, 2 insertions(+), 1 deletion(-)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git push origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 397 bytes | 397.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:a8anassis/codingfactorygit.git
  da31c7b..95fc326  main -> main

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$
```

- Push στον origin main



GitHub (1)

Version Control με Git / GitHub

a8anassis / **codingfactorygit** Public

<> Code Issues Pull requests Actions Projects Wiki

main ▾ codingfactorygit / chapter1 /

a8anassis form1 v0.2

..

form1.html	form1 v0.2
form2.html	chapter-1 v0.1

- Επιβεβαιώνουμε



GitHub (2)

Version Control με Git / GitHub

a8anassis / codingfactorygit Public

<> Code Issues Pull requests Actions

main codingfactorygit / chapter1 / form1.html

a8anassis form1 v0.2

1 contributor

31 lines (29 sloc) | 825 Bytes

```
1 <!-- Version 2 -->
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
```

- Βλέπουμε τον κώδικα του **form1.html**



Delete από stage area / Delete commit

Version Control με Git / GitHub

- Από staging area
 - `git restore --cached <file_name>`
- Delete commit
 - `git reset 00453ad` (όπου με `git log` έχουμε δει ότι τα 7 πρώτα digits του hash του προηγούμενου commit είναι 00453ad)
 - `git reset --soft HEAD~1` (ένα commit back στο history του current branch, αλλά οι αλλαγές διατηρούνται στο working dir)
 - `git reset --hard HEAD~1` (ένα commit back στο history του current branch, και οι αλλαγές δεν διατηρούνται στο working dir)



Clone

Version Control με Git / GitHub

- Ο δεύτερος τρόπος δημιουργίας repository είναι να αντιγράψουμε (Clone) ένα repository άλλου χρήστη από το GitHub
- Στο παράδειγμα που ακολουθεί έχουμε δημιουργήσει ένα φάκελο, έστω git και μέσα στον git θα κάνουμε τις αντιγραφές (git clone) όλων των repositories που θέλουμε



Δημιουργία φάκέλου εργασίας

Version Control με Git / GitHub

- Δημιουργούμε λοιπόν ένα γενικό φάκελο git μέσα στον οποίο θα κάνουμε clone repositories από το GitHub
 - `cd /users/a8ana` σε Win ή `cd /home/username` σε Linux
- Δημιουργία φακέλου git-clones για παράδειγμα μέσα στο home dir (`/users/username` σε Win ή `/home/username` σε Linux)
 - `mkdir git-clones`

```
a8ana@thanassis-PC MINGW64 ~
$ pwd
/c/Users/a8ana

a8ana@thanassis-PC MINGW64 ~
$ mkdir git-clones

a8ana@thanassis-PC MINGW64 ~
$ cd git-clones/

a8ana@thanassis-PC MINGW64 ~/git-clones
$ |
```



Αντιγραφή URL (1)

Version Control με Git / GitHub

The screenshot shows a GitHub repository page for 'a8anassis/hello-world'. The main navigation bar includes 'Pull requests', 'Issues', 'Codespaces', 'Marketplace', and 'Explore'. Below the navigation is a search bar and a 'Pin' button. The repository details show it's 'Public' and has '2 branches' and '0 tags'. A message states 'Your main branch isn't protected'. On the left, there are two README files: 'a8anassis v 1.0' and 'hello-world'. The 'Code' tab is active, and a modal window is overlaid on the page. The modal has tabs for 'Local' and 'Codespaces' (marked as 'New'). It contains a 'Clone' section with three options: 'HTTPS' (selected), 'SSH', and 'GitHub CLI'. The 'HTTPS' field contains the URL 'git@github.com:a8anassis/hello-world.git'. Below this, a note says 'Use a password-protected SSH key.' There are also buttons for 'Open with GitHub Desktop' and 'Open with Visual Studio'.

- Αντιγράφουμε το URL του hello-world project



Δημιουργία local repository μέσω αντιγραφής

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/git-clones
$ git clone git@github.com:a8anassis/hello-world.git
Cloning into 'hello-world'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 27 (delta 0), reused 21 (delta 0), pack-reused 0
Receiving objects: 100% (27/27), done.

a8ana@thanassis-pc MINGW64 ~/git-clones
$ pwd
/c/Users/a8ana/git-clones

a8ana@thanassis-pc MINGW64 ~/git-clones
$ ls -la
total 40
drwxr-xr-x 1 a8ana 197609 0 Dec  7 21:30 .
drwxr-xr-x 1 a8ana 197609 0 Dec  7 21:15 ../
drwxr-xr-x 1 a8ana 197609 0 Dec  7 21:30 hello-world/
```

- Παρατηρούμε ότι δημιουργείται νέος git folder με όνομα hello-world και περιεχόμενα τα περιεχόμενα του repo στο GitHub



Cloned Repository

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/git-clones/hello-world (main)
$ ls -la
total 5
drwxr-xr-x 1 a8ana 197609 0 Dec  7 21:30 .
drwxr-xr-x 1 a8ana 197609 0 Dec  7 21:30 ../
drwxr-xr-x 1 a8ana 197609 0 Dec  7 21:30 .git/
-rw-r--r-- 1 a8ana 197609 17 Dec  7 21:30 README.md
```

```
a8ana@thanassis-pc MINGW64 ~/git-clones/hello-world (main)
$ git remote -v
origin git@github.com:a8anassis/hello-world.git (fetch)
origin git@github.com:a8anassis/hello-world.git (push)
```

```
a8ana@thanassis-pc MINGW64 ~/git-clones/hello-world (main)
$ |
```

- Παρατηρούμε ότι με το clone έχει γίνει ταυτόχρονα add το remote repository οπότε μπορούμε να κάνουμε pull, push



Fork – Ανεξάρτητο copy

Version Control με Git / GitHub

The screenshot shows a GitHub repository page. At the top, there are links for 'Marketplace' and 'Explore'. On the right side of the header are icons for notifications, a plus sign, and a user profile. Below the header, there are buttons for 'Pin', 'Unwatch' (with 1 watch), 'Fork' (with 0 forks, circled in red), and 'Star' (with 1 star). Underneath these buttons are links for 'Security', 'Insights', and 'Settings'. In the center, there's a green 'Code' button with a dropdown arrow. To the right of the code button is an 'About' section containing the text 'No description, website, or topics provided.' Below this are links for 'Readme' and '1 star'. At the bottom left, there's a 'Protect this branch' button and an 'X' icon.

- Τα forks είναι αυτόνομα copies. Μπορούμε να κάνουμε fetch και pull αλλά δεν μπορούμε να κάνουμε push, παρά μόνο *pull requests* για να ανεβάσουμε στο upstream repository (GitHub instance του Repository).



Pull Request

Version Control με Git / GitHub

- Με ένα pull request ειδοποιούμε τους peers ότι έχουμε ανεβάσει ένα νέο version σε ένα branch (push) και να το κάνουν pull (δηλαδή merge, άρα κατά βάση πρόκειται για merge request)
- Επομένως όλη η ομάδα μαθαίνει με το pull request ότι θα πρέπει να κάνει review το request και να το κάνει merge
- Τα pull requests λειτουργούν και ως χώροι συζήτησης (review forums)



Clone vs Fork

Version Control με Git / GitHub

- Clone κάνουμε όταν συνεργαζόμαστε με ομάδες και πρέπει να κάνουμε pull και push
- Fork κάνουμε όταν θέλουμε να αναπτύξουμε ένα νέο isolated project βασισμένο στο αρχικό
- Για παράδειγμα:
 - το Ubuntu είναι fork του Debian
 - Η ΒΔ MariaDB είναι fork της MySQL



Branching

- Στο Git μπορούμε να δουλεύουμε σε διαφορετικά μονοπάτια ή branches. Σε κάθε ένα από αυτά μπορούμε να κάνουμε αλλαγές ανεξάρτητα χωρίς να επηρεάζεται το αρχικό branch. Έτσι έχουμε ένα κομμάτι κώδικα που δουλεύει και μένει απείραχτο, ενώ μπορούμε να δουλεύουμε σε ομάδες σε διαφορετικό πράγμα ο καθένας, σε διαφορετικές εκδόσεις του κώδικα
- Όταν οι αλλαγές από κάθε ένα κομμάτι ολοκληρωθούν τότε μπορούμε να τις ενώσουμε, στο κεντρικό κορμό του main branch



Check branches

```
a8ana@thanassis-pc MINGW64 ~
$ git branch
* main

a8ana@thanassis-pc MINGW64 ~
$ git branch -a
* main
  remotes/origin/main

a8ana@thanassis-pc MINGW64 ~
$
```

- Με **git branch** βλέπουμε τα branches ενώ με **-a** εμφανίζουμε όλα τα branches

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/Coding
$ git branch -a -vv
* main           95fc326 form1 v0.2
  remotes/origin/main 95fc326 form1 v0.2

a8ana@thanassis-pc MINGW64 ~/OneDrive/Coding
$ |
```

Με **-vv** εμφανίζουμε και το τελευταίο commit. Όπως βλέπουμε τα δύο branches, main και origin/main είναι sync



Workflow

Version Control με Git / GitHub

- Ένα Git Workflow είναι ένα μοντέλο εργασίας, ένα convention για το πως μπορούμε να χρησιμοποιούμε το Git με παραγωγικό τρόπο στο πλαίσιο του developing ή του DevOps
- Κατά τη φάση του Development η ομάδα της ανάλυσης συνεργάζεται με τους χρήστες και τον product Owner και θέτει τις απαιτήσεις του συστήματος, που στη συνέχεια εκχωρούνται σε μέλη της ομάδας για ανάπτυξη



DevOps

Version Control με Git / GitHub

- Κατά τη φάση της συντήρησης, με τον όρο DevOps εννοούμε μία διαδικασία συνεχούς ανάπτυξης μία εφαρμογής με βάση τις απαιτήσεις των χρηστών. Η ομάδα του Operations (π.χ. HelpDesk) συνεργάζεται με τους χρήστες και ανοίγει tickets (βλ. Jira, Ticket Management System) τα οποία εκχωρεί σε κάποιο μέλος της ομάδας του Development
- Υπάρχει δηλαδή ένα pipeline όπου το Operation Team στέλνει αιτήματα χρηστών στο Development Team



GitFlow / Trunk-based Development

Version Control με Git / GitHub

- Τα δύο πιο γνωστά Workflows είναι το GitFlow και το Trunk-based Development
- Το GitFlow περιλαμβάνει τη δημιουργία μεγάλων feature branches τα οποία ενσωματώνονται στο main branch όταν ολοκληρωθούν και χρησιμοποιείται πιθανά κατά την αρχική ανάπτυξη ενός project
- Το Trunk-based Development είναι πιο ευέλικτο με μικρότερα feature branches και χρησιμοποιείται ιδιαίτερα στο πλαίσιο του DevOps



GitFlow

- Το **GitFlow** χρησιμοποιεί εκτός από το main branch, το developer branch
- Το main branch έχει το official release (safe-code). Το developer branch χρησιμοποιείται για integration των features. Συνήθως τα main και developer branches πρόσβαση έχουν οι managers, ενώ τα μέλη των ομάδων έχει ο καθένας συνήθως ένα δικό του branch και κάνει push, ενώ ο manager κάνει το merge στο development branch και integrate στο main branch
- Το **GitFlow** χρησιμοποιείται σε συνδυασμό με το Feature Branch Workflow



Feature Branch Workflow

Version Control με Git / GitHub

- Ένα γνωστό Workflow είναι το Feature Branch Workflow, όπου η ανάπτυξη κάθε feature μία εφαρμογής θα πρέπει να λαμβάνει χώρα σε ένα ξεχωριστό branch και όχι στο main
- Ξεκινώντας κάποιο μέλος μιας ομάδας να αναπτύσσει ένα feature branch (π.χ. `feature-new_button`) μπορεί όταν τελειώσει να κάνει pull requests στα άλλα μέλη της ομάδας



Trunk-based Development

Version Control με Git / GitHub

- Στο Trunk-based development κάθε developer μπορεί να κάνει merge μικρά και συχνά updates στο core trunk ή main branch
- Επομένως το merging και το integration γίνονται σε ένα βήμα στο πλαίσιο της μεθοδολογίας CI/CD (Continuous Integration / Continuous Delivery ή Deployment)
- Το delivery αναφέρεται σε αυτοματοποιημένα tests στο main repository ή production environment
- Το Deployment πάει ένα βήμα παραπέρα και κάθε αλλαγή που περνάει όλες τις φάσεις του production pipeline (build, test) γίνεται release στους πελάτες



Git branch και git checkout

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git branch develop
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git checkout develop
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (develop)
$ |
```

- Αν θέλουμε να δημιουργήσουμε ένα νέο branch ώστε να δουλέψουμε χωρίς να αλλάξουμε το main branch, τότε μπορούμε να κάνουμε **git branch <branch-name>**, αν είμαστε στο main
- Με **git branch develop αν είμαστε στο main branch** δημιουργούμε ένα νέο branch develop από το main branch
- Με **git checkout develop** αλλάζουμε branch (αλλάζει ο HEAD και δείχνει στο develop)



Git checkout -b

Version Control με Git / GitHub

- Με `git checkout -b <branch-name>` δημιουργούμε ένα branch και κάνουμε checkout ταυτόχρονα

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git checkout -b hotfix
Switched to a new branch 'hotfix'

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (hotfix)
$ |
```



Set upstream branch

- Για κάθε branch που φτιάχνουμε τοπικά θα πρέπει να φτιάχνουμε και το αντίστοιχο upstream branch (με push στο GitHub). Επίσης, θα πρέπει να θέτουμε και την αντιστοίχιση μεταξύ του remote/<branchname> με το local branch ώστε να μην χρειάζεται σε κάθε branch να γράφουμε git pull origin develop ή git push origin develop, για παράδειγμα, αλλά να γράφουμε git pull και git push χωρίς παραμέτρους
- Αυτή η αντιστοίχιση remote tracking branch με το local branch γίνεται με το -u στο push ή μπορούμε ρητά να το κάνουμε, για παράδειγμα για το main branch με git branch --set-upstream=origin/main



Git push το branch και set default upstream (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (hotfix)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git push -u origin hotfix
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'hotfix' on GitHub by visiting:
remote:     https://github.com/a8anassis/codingfactorygit/pull/new/hotfix
remote:
To github.com:a8anassis/codingfactorygit.git
 * [new branch]      hotfix -> hotfix
branch 'hotfix' set up to track 'origin/hotfix'.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ |
```

- Κάνουμε **git push -u** οπότε: 1) ανεβαίνει το branch στο GitHub και ταυτόχρονα, αφού ανέβει, δημιουργείται και το remote tracking branch (origin/hotfix) και η σύνδεση μεταξύ του hotfix και origin/hotfix



Git push το branch και set default upstream (2)

Version Control με Git / GitHub

```
a8ana@thanassis - pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git push -u origin develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:     https://github.com/a8anassis/codingfactorygit/pull/new/develop
remote:
To github.com:a8anassis/codingfactorygit.git
 * [new branch]      develop -> develop
branch 'develop' set up to track 'origin/develop'.
```

- Κάνουμε το ίδιο, δηλαδή git push –u origin develop, και για το develop branch



Git branch – u

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git push
fatal: The current branch main has no upstream branch.
To push the current branch and set the remote as upstream, use

  git push --set-upstream origin main

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git branch -u origin/main
branch 'main' set up to track 'origin/main'.
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git push
Everything up-to-date
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ |
```

Εναλλακτικά, αν υπάρχει ήδη το branch, όπως το main branch, μπορούμε να κάνουμε set το upstream με **git branch – u origin/main**



GitHub

Version Control με Git / GitHub

- Παρατηρούμε ότι έχουν γίνει push στο GitHub τα hotfix και production branches



Νέο branch στο GitHub

Version Control με Git / GitHub

a8anassis / codingfactorygit Public

Code Issues Pull requests

develop ▾ 3 branches 0 tags

Switch branches/tags

fetaure-new_button

Branches Tags

Create branch: fetaure-new_button from 'develop'

View all branches

feature-new_bu... ▾ 4 branches

Switch branches/tags

Find or create a branch...

Branches Tags

main default

develop

✓ feature-new_button

hotfix

View all branches

- Δημιουργούμε νέο branch με όνομα `feature-new_button` στο GitHub



Git pull (το branch)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (hotfix)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git pull
From github.com:a8anassis/codingfactorygit
 * [new branch]      feature-new_button -> origin/feature-new_button
Already up to date.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (main)
$ git checkout feature-new_button
Switched to a new branch 'feature-new_button'
branch 'feature-new_button' set up to track 'origin/feature-new_button'.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (feature-new_button)
$
```

- git pull και κατεβάζουμε το feature-new_button branch και κάνουμε checkout



Συνηθισμένες πράξεις στο git

Version Control με Git / GitHub

- Κάνουμε κάποιες αλλαγές σε ένα αρχείο του *working dir*
- Ελέγχουμε τι έχει αλλάξει με: *git status*
- Κάνουμε Review τις αλλαγές με *git diff*
- Προσθέτουμε στο staging area με *git add*
- Κάνουμε το commit με: *git commit*



cd chapter1

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (feature-new_button)
$ ls -la
total 49
drwxr-xr-x 1 a8ana 197609 0 Dec  7 16:29 .
drwxr-xr-x 1 a8ana 197609 0 Dec  4 13:32 ..
drwxr-xr-x 1 a8ana 197609 0 Dec  8 13:56 .git/
-rw-r--r-- 1 a8ana 197609 93 Dec  7 16:29 README.md
drwxr-xr-x 1 a8ana 197609 0 Dec  7 20:55 chapter1/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter2/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter3/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter4/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter5/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 chapter6/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 img/
drwxr-xr-x 1 a8ana 197609 0 Dec  3 19:11 testbed/

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit (feature-new_button)
$ cd chapter1

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ ls -la
total 32
drwxr-xr-x 1 a8ana 197609 0 Dec  7 20:55 .
drwxr-xr-x 1 a8ana 197609 0 Dec  7 16:29 ..
-rw-r--r-- 1 a8ana 197609 856 Dec  7 20:54 form1.html
-rw-r--r-- 1 a8ana 197609 1742 Oct 21 20:42 form2.html
-rw-r--r-- 1 a8ana 197609 876 Oct 27 18:45 index.html
-rw-r--r-- 1 a8ana 197609 735 Oct 21 19:03 lists.html
-rw-r--r-- 1 a8ana 197609 1742 Oct 21 18:41 tables.html
-rw-r--r-- 1 a8ana 197609 852 Oct 27 18:45 video.html

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ vi form1.html

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ |
```

- Πάμε στο chapter1 και θα κάνουμε μία αλλαγή στο form1.html και θα προσθέσουμε ένα new button



Αλλαγή στο form1

Version Control με Git / GitHub

```
<!-- Version 2 -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
</head>
<body>

    <form>
        <div>
            <label for="username">Username</label>
            <input type="text" id="username" placeholder="Enter username">
        </div>
        <div>
            <label for="email">E-mail</label>
            <input type="email" id="email" name="email">
        </div>
        <div>
            <label for="message">Message</label>
            <textarea id="message" cols="60" rows="10"></textarea>
        </div>
        <div>
            <input type="submit" value="Submit">
        </div>

    </form>
    <button type="button">New Button</button>
</body>
</html>
~
```

- Προσθέτουμε
ένα
button



Git status

Version Control με Git / GitHub

- To form1.html είναι modified

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/c  
$ git status  
On branch feature-new_button  
Your branch is up to date with 'origin/feature-new_button'.  
  
Changes not staged for commit:  
(use "git add <file>..." to update what will be committed)  
(use "git restore <file>..." to discard changes in working directory)  
      modified:   form1.html  
  
Untracked files:  
(use "git add <file>..." to include in what will be committed)  
      .../chapter2/  
      .../chapter3/  
      .../chapter4/  
      .../chapter5/  
      .../chapter6/  
      .../img/  
      .../testbed/  
  
no changes added to commit (use "git add" and/or "git commit -a")  
  
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/c  
$
```



Git diff

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/we
$ git diff
diff --git a/chapter1/form1.html b/chapter1/form1.html
index 05a50d4..f37702c 100644
--- a/chapter1/form1.html
+++ b/chapter1/form1.html
@@ -27,5 +27,7 @@
      </div>

      </form>
+
+      <button type="button">New Button</button>
</body>
</html>

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/we
$ |
```

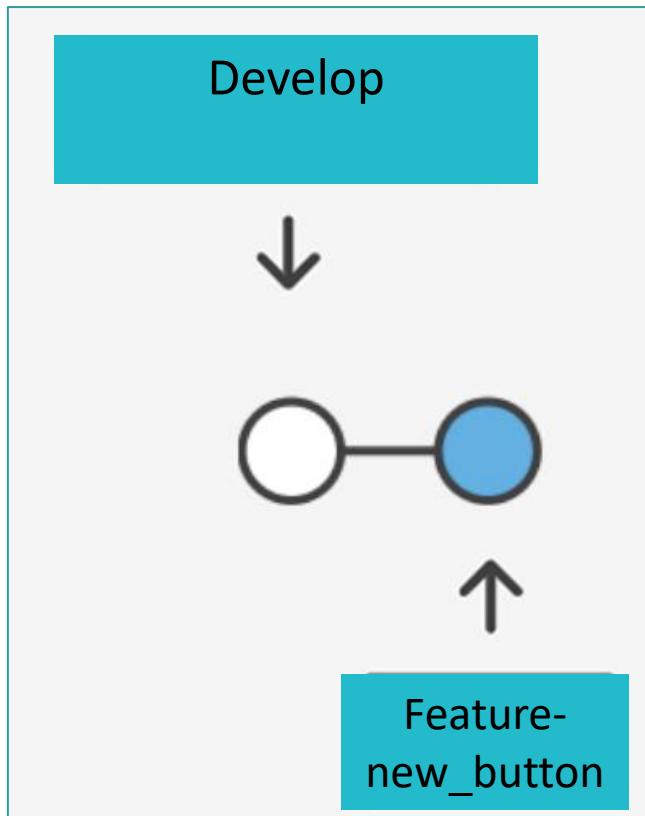
- Με git diff βλέπουμε τις διαφορές.
- Η 1^η γραμμή με @@ είναι ο header (summary of changes) και λέει ότι από το 1^ο αρχείο από τη γραμμή 27 και μετά έγιναν extract 5 γραμμές και στο 2^ο αρχείο από τη γραμμή 27, προστέθηκαν 7 γραμμές



Git commit

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git commit -a -m "form1 v0.2"
[feature-new_button 775efeb] form1 v0.2
 1 file changed, 2 insertions(+)
```



- Κάνουμε commit τις αλλαγές στο feature-new_button



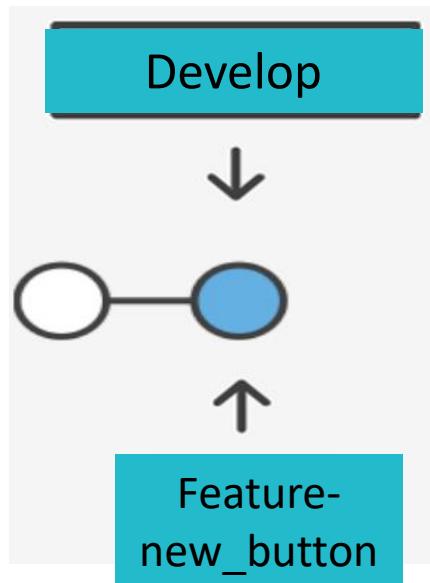
Git merge

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git checkout develop
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git merge feature-new_button
Updating 95fc326..775efeb
Fast-forward
 chapter1/form1.html | 2 ++
 1 file changed, 2 insertions(+)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ |
```



- Γίνεται FF (Fast Forward) του Develop στο νέο commit



Git push

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 422 bytes | 422.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:a8anassis/codingfactorygit.git
  95fc326..775efeb  develop -> develop

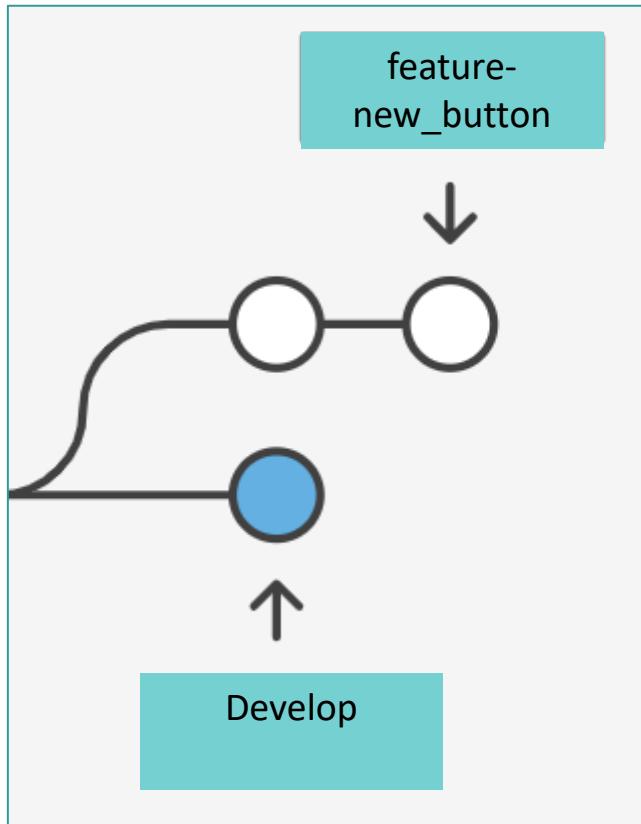
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ |
```

- Κάνουμε push τις συγχωνευμένες αλλαγές στο develop στο origin develop



3-way merge (1)

Version Control με Git / GitHub

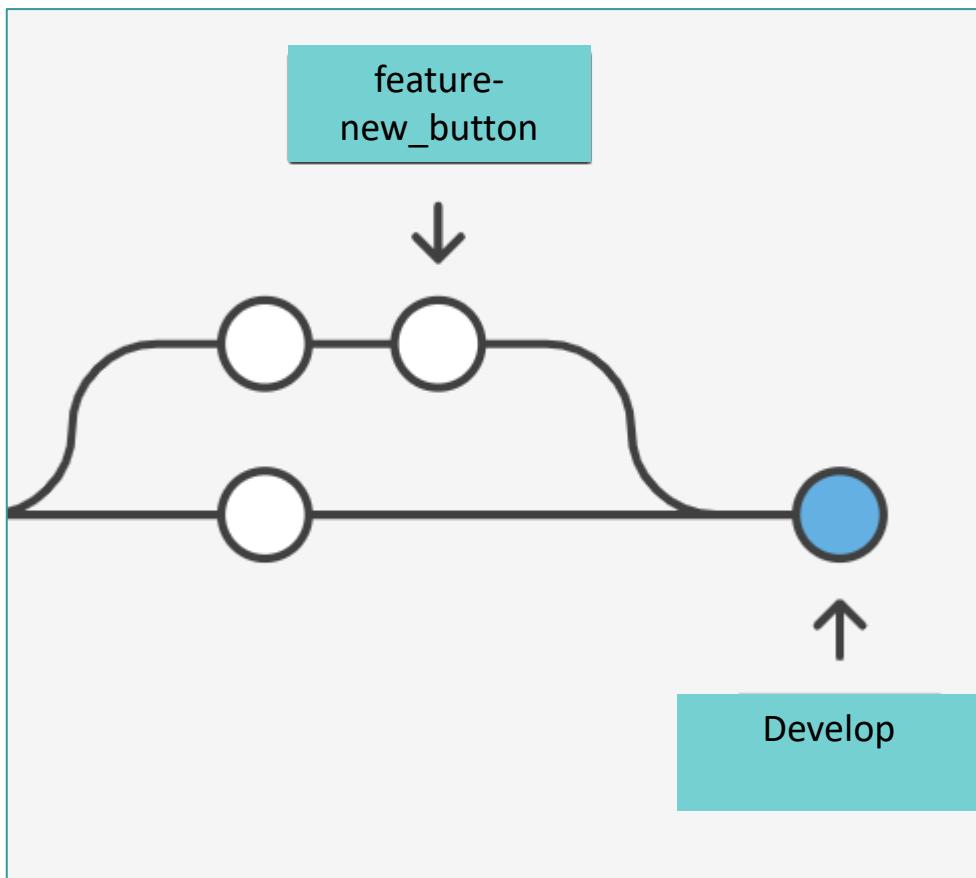


- Τα fast forward merges δεν μπορούν να γίνουν αν έχουν γίνει diverge (αποκλίνουν) τα δύο branches
- Αν για παράδειγμα γίνει μία αλλαγή στο file1.html στο develop και μία άλλη αλλαγή στο ίδιο αρχείο στο feature-new_button branch, τότε γίνεται 3-way merge και δημιουργείται ένα νέο merge commit (βλ. επόμενη διαφάνεια)



3-way merge (2)

Version Control με Git / GitHub



- Δημιουργείται ένα merge commit στο develop branch



Αλλαγή στο feature

Version Control με Git / GitHub

- Στο τέλος έχουμε προσθέσει ένα σχόλιο

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1
<!-- Version 2 -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
</head>
<body>

<form>
    <div>
        <label for="username">Username</label>
        <input type="text" id="username" placeholder="Enter username">
    </div>
    <div>
        <label for="email">E-mail</label>
        <input type="email" id="email" name="email">
    </div>
    <div>
        <label for="message">Message</label>
        <textarea id="message" cols="60" rows="10"></textarea>
    </div>
    <div>
        <input type="submit" value="Submit">
    </div>

</form>

<button type="button">New Button</button>
</body>
</html>
<!-- A comment in feature branch |-->
```



Git status & commit

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git status
On branch feature-new_button
Your branch is ahead of 'origin/feature-new_button' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   form1.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ./chapter2/
    ./chapter3/
    ./chapter4/
    ./chapter5/
    ./chapter6/
    ./img/
    ./testbed/

no changes added to commit (use "git add" and/or "git commit -a")

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git commit -am "form1 v.03"
[feature-new_button 477e431] form1 v.03
 1 file changed, 1 insertion(+)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$
```



Form1.html στο develop

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git checkout develop
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ cat form1.html
<!-- Version 2 -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
</head>
<body>

    <form>
        <div>
            <label for="username">Username</label>
            <input type="text" id="username" placeholder="Enter username">
        </div>
        <div>
            <label for="email">E-mail</label>
            <input type="email" id="email" name="email">
        </div>
        <div>
            <label for="message">Message</label>
            <textarea id="message" cols="60" rows="10"></textarea>
        </div>
        <div>
            <input type="submit" value="Submit">
        </div>
    </form>

    <button type="button">New Button</button>
</body>
</html>

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
```



Αλλαγή στο develop

Version Control με Git / GitHub

MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1

```
<!-- Version 2 -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
</head>
<body>
    <!-- A comment in develop branch -->
    <form>
        <div>
            <label for="username">Username</label>
            <input type="text" id="username" placeholder="Enter username">
        </div>
        <div>
            <label for="email">E-mail</label>
            <input type="email" id="email" name="email">
        </div>
        <div>
            <label for="message">Message</label>
            <textarea id="message" cols="60" rows="10"></textarea>
        </div>
        <div>
            <input type="submit" value="Submit">
        </div>
    </form>
    <button type="button">New Button</button>
</body>
</html>
~
```



Git status & commit

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git status
On branch develop
Your branch is up to date with 'origin/develop'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   form1.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ./chapter2/
    ./chapter3/
    ./chapter4/
    ./chapter5/
    ./chapter6/
    ./img/
    ./testbed/

no changes added to commit (use "git add" and/or "git commit -a")

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git commit -am "form1 v0.3 develop"
[develop 7d74cd1] form1 v0.3 develop
 1 file changed, 1 insertion(+), 1 deletion(-)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$
```

- Git status και commit στο develop



Git merge

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git merge feature-new_button
Auto-merging chapter1/form1.html
Merge made by the 'ort' strategy.
 chapter1/form1.html | 1 +
 1 file changed, 1 insertion(+)
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ |
```

- 3-way merge. Δεν υπήρχε conflict γιατί οι αλλαγές αφορούσαν διαφορετικά κομμάτια κώδικα



Conflicts

Version Control με Git / GitHub

- Αν ωστόσο γίνει μία αλλαγή στο file1.html στο develop branch και μία αλλαγή στο ίδιο κομμάτι κώδικα στο feature-new_button branch και πάμε να κάνουμε merge θα υπάρξει conflict
- Θα δούμε στις επόμενες διαφάνειες πως διαχειριζόμαστε τα conflicts



Αλλαγή στο feature

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git checkout feature-new_button
Switched to branch 'feature-new_button'
Your branch is ahead of 'origin/feature-new_button' by 2 commits.
  (use "git push" to publish your local commits)
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ vi form1.html
```

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1
<!-- Version 2 -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Student Register</title>
</head>
<body>
```

- Κάναμε μία αλλαγή στο 'Student Register'



Git status & commit

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git status
On branch feature-new_button
Your branch is ahead of 'origin/feature-new_button' by 2 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   form1.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ./chapter2/
    ./chapter3/
    ./chapter4/
    ./chapter5/
    ./chapter6/
    ./img/
    ./testbed/

no changes added to commit (use "git add" and/or "git commit -a")

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git commit -am "form1 v0.4 feature"
[feature-new_button 309f810] form1 v0.4 feature
 1 file changed, 1 insertion(+), 1 deletion(-)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ |
```

- Commit στο feature-new_button branch



Αλλαγές στο develop

Version Control με Git / GitHub

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter  
<!-- Version 2 -->  
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <meta charset="UTF-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title>Teacher Register</title>  
</head>  
<body>
```

- Η αλλαγή που κάναμε στο develop αφορά την ίδια γραμμή κώδικα
- Εδώ δώσαμε Teacher Register (ενώ στο feature ήταν Student Register)



Git commit και merge στο develop

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git commit -am "form1 v0.4 develop"
[develop 7f3e664] form1 v0.4 develop
 1 file changed, 1 insertion(+), 1 deletion(-)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git merge feature-new_button
Auto-merging chapter1/form1.html
CONFLICT (content): Merge conflict in chapter1/form1.html
Automatic merge failed; fix conflicts and then commit the result.

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop|MERGING)
$
```

- Αφού κάναμε commit τις αλλαγές στο develop branch, μετά κάνουμε merge
- Το merge αποτυγχάνει. Θα πρέπει να επεξεργαστούμε manually τις αλλαγές και μετά να κάνουμε add και commit



Επίλυση conflict manually (1)

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop|MERGING)
$ vi form1.html |
```

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1
<!-- Version 2 -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<<<<<< HEAD
    <title>Teacher Register</title>
=====
    <title>Student Register</title>
>>>>> feature-new_button
</head>
<body>
```

- Πάνω από το === marker (από το <<<<<) είναι το receiver branch (develop) και κάτω από το === μέχρι >>>>> είναι το merging branch
- Φτιάχνουμε το merge αρχείο όπως νομίζουμε



Επίλυση conflict manually (2)

Version Control με Git / GitHub

```
MINGW64:/c/Users/a8ana/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter  
<! -- Version 2 -->  
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <meta charset="UTF-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title>Student Register</title>  
</head>  
<body>
```

- Επιλύουμε manually και αφήνουμε το Student Register



Git add & commit

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop|MERGING)
$ git commit -am "form1 v0.4 resolved"
[develop 943b398] form1 v0.4 resolved
```

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ |
```

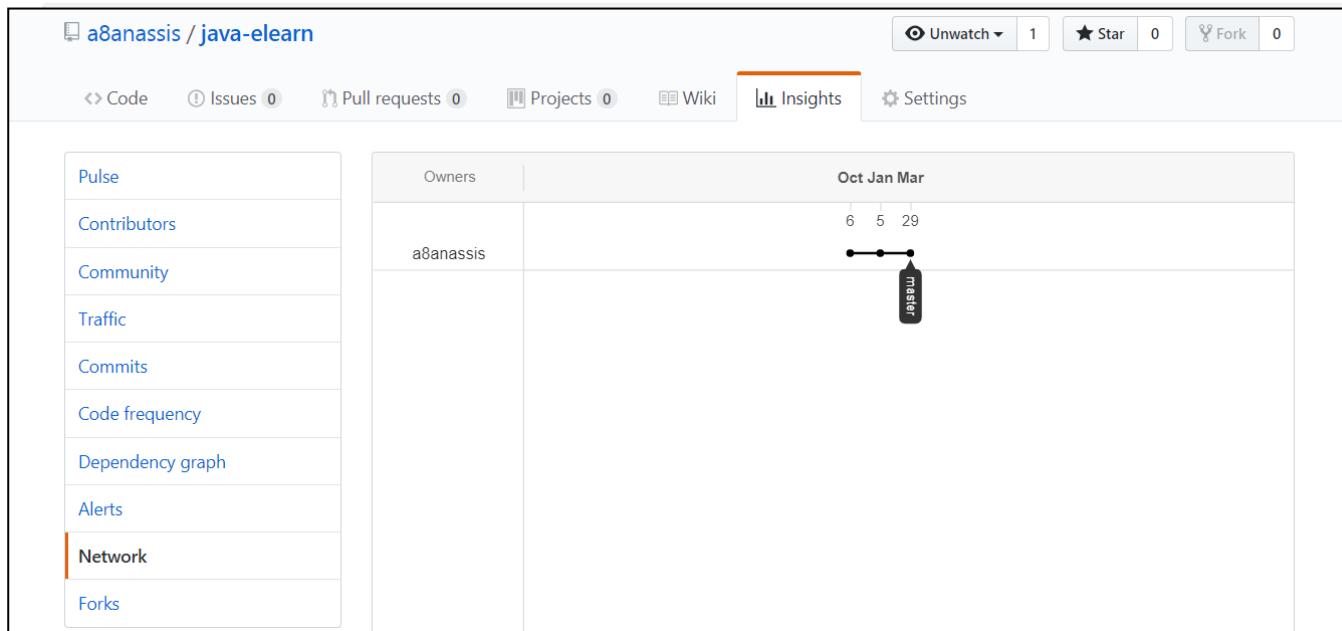
- Αφού επιλύσουμε το conflict μετά κάνουμε add και commit
- Δημιουργείται ένα νέο merge commit



Επισκόπηση των branches – Git Network

Version Control με Git / GitHub

Πηγαίνουμε στο GitHub και πατάμε insights -> network. Μας εμφανίζεται η εικόνα του δικτύου του repository μας:



Προφανώς αυτή η εικόνα μπορεί να γίνει **πολύ** πιο περίπλοκη.



Καλές πρακτικές

Version Control με Git / GitHub

Αυτή ήταν η βασική χρήση του Git. Μερικές καλές πρακτικές:

- Φτιάχνουμε branches που αντιπροσωπεύουν διαφορετικές ενότητες εργασίας
- Δεν σπρώχνουμε ημιτελή κώδικα στο master branch. Το master πρέπει **πάντα** να περιέχει κώδικα που δουλέψει.
- Αν δουλεύουμε με άλλους, ρωτάμε από ποιό branch να αρχίσουμε να δουλεύουμε και τραβάμε και σπρώχνουμε εκεί. Συνήθως ο κάθε χρήστης έχει δικό του branch και σπρώχνει εκεί, και ο επιβλέπων κάνει την ένωση με τα develop και master
- Αποφεύγουμε να δουλεύουμε σε branches που δουλεύουν και άλλοι παράλληλα. Απλά κάνουμε νέο branch και κάνουμε τις αλλαγές μας εκεί, και μετά κάνουμε τα αντίστοιχα merge.



Git tags

- Τα tags είναι references σε ένα commit στο history ενός branch,
- Μπορούμε για παράδειγμα να κάνουμε tag ένα release point στο history των commits.
- Υπάρχουν Lightweight tags και Annotated tags (με -a). Η διαφορά είναι στα μετά-δεδομένα που αποθηκεύονται. Τα annotated tags αποθηκεύουν περισσότερη πληροφορία, όπως tagger name, e-mail, date
- Τα lightweight tags είναι βασικά bookmarks



Lightweight tags

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (main)
$ git checkout feature-new_button
Switched to branch 'feature-new_button'
Your branch is ahead of 'origin/feature-new_button' by 3 commits.
  (use "git push" to publish your local commits)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git tag v0.4

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git tag
v0.4

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ |
```

- Και με git log βλέπουμε το tag

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git log --oneline
309f810 (HEAD -> feature-new_button, tag: v0.4) form1 v0.4 feature
477e431 form1 v.03
775efeb (origin/develop) form1 v0.2
95fc326 (origin/main, origin/hotfix, origin/feature-new_button, main, hotfix) form1 v0.2
da31c7b chapter-1 v0.1
d9c72bf README v0.4
42b8bc4 README v0.3
4ad6d99 Initial commit
```



Annotated Tags

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (feature-new_button)
$ git checkout develop
Switched to branch 'develop'
Your branch is ahead of 'origin/develop' by 6 commits.
  (use "git push" to publish your local commits)

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git tag -a v1.0 -m "version 1.0"

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
```

- Με -a εννοούμε annotated tags
- Τα annotated tags θέλουμε και ένα message με -m γιατί αποθηκεύονται ως κανονικά objects στην Git Database



Git show

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git show v1.0
tag v1.0
Tagger: a8ana <a8anassis@gmail.com>
Date:   Thu Dec 8 16:19:42 2022 +0200

version 1.0

commit 943b39808f2bad871b2abb91102aa9bc694ee39c (HEAD -> develop, tag: v1.0)
Merge: 7f3e664 309f810
Author: a8ana <a8anassis@gmail.com>
Date:   Thu Dec 8 15:29:55 2022 +0200

    form1 v0.4 resolved

diff --cc chapter1/form1.html
index 647fa3a,9f703ae..4e62085
--- a/chapter1/form1.html
+++ b/chapter1/form1.html
@@@ -5,10 -5,10 +5,10 @@@
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
-   <title>Teacher Register</title>
+   <title>Student Register</title>
</head>
<body>
-
+
<form>
    <div>
        <label for="username">Username</label>
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$
```

- Με git show v1.0 βλέπουμε πληροφορίες για το tag v1.0



Git log

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git log --oneline
943b398 (HEAD -> develop, tag: v1.0) form1 v0.4 resolved
7f3e664 form1 v0.4 develop
309f810 (tag: v0.4, feature-new_button) form1 v0.4 feature
97ea312 Merge branch 'feature-new_button' into develop
7d74cd1 form1 v0.3 develop
477e431 form1 v.03
775efeb (origin/develop) form1 v0.2
95fc326 (origin/main, origin/hotfix, origin/feature-new_button, main, hotfix) form1 v0.2
da31c7b chapter-1 v0.1
d9c72bf README v0.4
42b8bc4 README v0.3
4ad6d99 Initial commit

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ |
```

- Με git log βλέπουμε τα tags στα αντίστοιχα commits



Push tags to origin

Version Control με Git / GitHub

```
a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ git push origin v1.0
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 8 threads
Compressing objects: 100% (25/25), done.
Writing objects: 100% (25/25), 2.21 KiB | 1.11 MiB/s, done.
Total 25 (delta 13), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (13/13), completed with 2 local objects.
To github.com:a8anassis/codingfactorygit.git
 * [new tag]      v1.0 -> v1.0

a8ana@thanassis-pc MINGW64 ~/OneDrive/CodingFactory-REBOOT/webprojects/codingfactorygit/chapter1 (develop)
$ |
```

- Κάνουμε push το tag στο upstream



Αρχείο .gitignore (1)

Version Control με Git / GitHub

- Αρχεία που αγνοούμε όταν προσθέτουμε αρχεία στο staging area (π.χ. με git add –A)
- Το αρχείο .gitignore βρίκεται στο root φάκελο του repository
- Π.χ. με *.class αγνοούμε όλα τα class αρχεία της java που μπορεί να βρίσκονται μέσα στο φάκελο που εποπτεύουμε



Αρχείο .gitignore (2)

Version Control με Git / GitHub

- out/ Κάνουμε ignore τον φάκελο out/ που βρίσκεται στο root dir. Οι φάκελοι τελειώνουμε με /
- **/out/ Κάνει ignore όλους τους φακέλους out σε οποιοδήποτε βάθος iεραρχίας
- *.log όλα τα αρχεία .log σε οποιοδήποτε βάθος της iεραρχίας
- !than.log όλα τα .log όπως έχουμε πει παραπάνω, εκτός από το than.log όπως λέμε εδώ
- than.txt όλα τα than.txt σε οποιοδήποτε βάθος iεραρχίας



Παράδειγμα αρχείου .gitignore (2)

Version Control με Git / GitHub

```
.gitignore - Σημειωματάριο
Αρχείο Επεξεργασία Μορφή Προβολή Βοήθεια
# Compiled class file
*.class

# Log file
*.log

# BlueJ files
*.ctxt

# Mobile Tools for Java (J2ME)
.mtj.tmp/

# Package Files #
*.jar
*.war
*.nar
*.ear
*.zip
*.tar.gz
*.rar

# virtual machine crash logs, see http://www.java.com/en/download/help/error_hotspot.xml
hs_err_pid*
```



Gitignore.io

Version Control με Git / GitHub

The screenshot shows the homepage of gitignore.io. At the top, there's a navigation bar with icons for back, forward, search, and user profile. The URL in the address bar is `toptal.com/developers/gitignore`. Below the bar, the Toptal logo and the site name "gitignore.io" are displayed, along with a GitHub icon.

The main content area features a large blue header with the text ".gitignore.io" in white. Below it, a sub-header reads "Create useful .gitignore files for your project". A text input field contains the text "VisualStudioCode" followed by a close button "X". To the right of the input field is a green "Create" button. At the bottom of the page, there are links for "Source Code" and "Command Line Docs".

- Δημιουργία .gitignore με keywords



Github/gitignore

Version Control με Git / GitHub

The screenshot shows the GitHub repository page for 'github/gitignore'. The repository has 3.3k watchers, 80.4k forks, and 141k stars. It contains 346 pull requests, 8 branches, and 0 tags. The main branch is 'main'. The repository is public and was last updated on May 10, 2024, with 3,526 commits. The 'About' section describes it as a collection of useful .gitignore templates. The 'Code' tab is selected, showing a list of files:

File	Description	Last Commit
.github	Adds a relationship question	12 months ago
Global	Update Xcode.gitignore	12 months ago
community	Ignore SQLite files	7 months ago
AL.gitignore	Add .gitignore for Microsoft Business Central	13 months ago
Actionscript.gitignore	Fix comments on same line causing ignore to break	5 years ago
Ada.gitignore	ensure single trailing newline	9 years ago
Agda.gitignore	Add MAlonzo directory. (#2978)	4 years ago
Android.gitignore	Android Studio	10 months ago

On the right side, there are links to 'Readme', 'CC0-1.0 license', 'Code of conduct', 'Security policy', '141k stars', '3.3k watching', and '80.4k forks'. There is also a 'Releases' section.

- Συλλογή .gitignore templates



Τύποι .gitignore

Version Control με Git / GitHub

1. Single repository and share

- **touch .gitignore** στο root dir του repository
(δημιουργούμε κενό αρχείο .gitignore)

2. All repositories (global .gitignore)

- Στο `~/.gitignore_global`, δηλαδή στο user home dir, κάνουμε `touch .gitignore_global`
- Configure git με `git config --global core.excludesFile ~/.gitignore_global`

3. Single repo, exclude local files χωρίς .gitignore, not share π.χ. .git/info/exclude



Βιβλιογραφικές Αναφορές (1)

Version Control με Git / GitHub

1 <https://git-scm.com/>

- Η ιστοσελίδα του Git.

2 <https://github.com/>

- Η ιστοσελίδα του GitHub.

3 <https://www.atlassian.com/git/tutorials/what-is-version-control>

- Περισσότερα για το version control.

4 <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

- Αναλυτικές οδηγίες εγκατάστασης.

5 <https://git-scm.com/book/en/v2/Getting-Started-First-Time-Git-Setup>

- Αναλυτικές οδηγίες αρχικοποίησης.



Βιβλιογραφικές Αναφορές (2)

Version Control με Git / GitHub

S. Chacon and B. Straub, *Pro Git*, 2nd ed. Apress 2014. [E-book] Available.

- Το βασικό ebook που αναλύει τα πάντα για το git για όποιον επιθυμεί να εμβαθύνει πλήρως. Μπορεί να βρεθεί στη διεύθυνση:

<https://git-scm.com/book/en/v2>



Ειδικά Θέματα στο Git

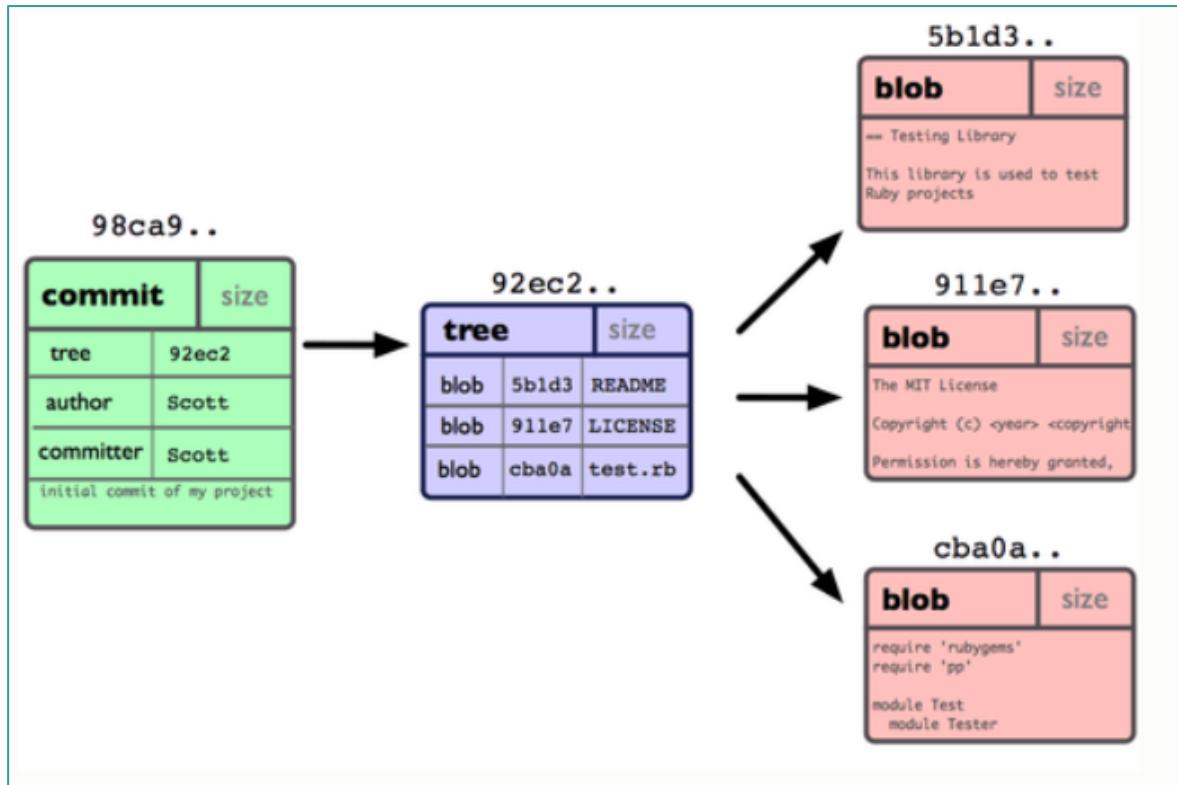
Version Control με Git / GitHub

- Στις επόμενες διαφάνειες θα δούμε πως το git χειρίζεται τα branch / checkout και merge



Commit object

Version Control με Git / GitHub

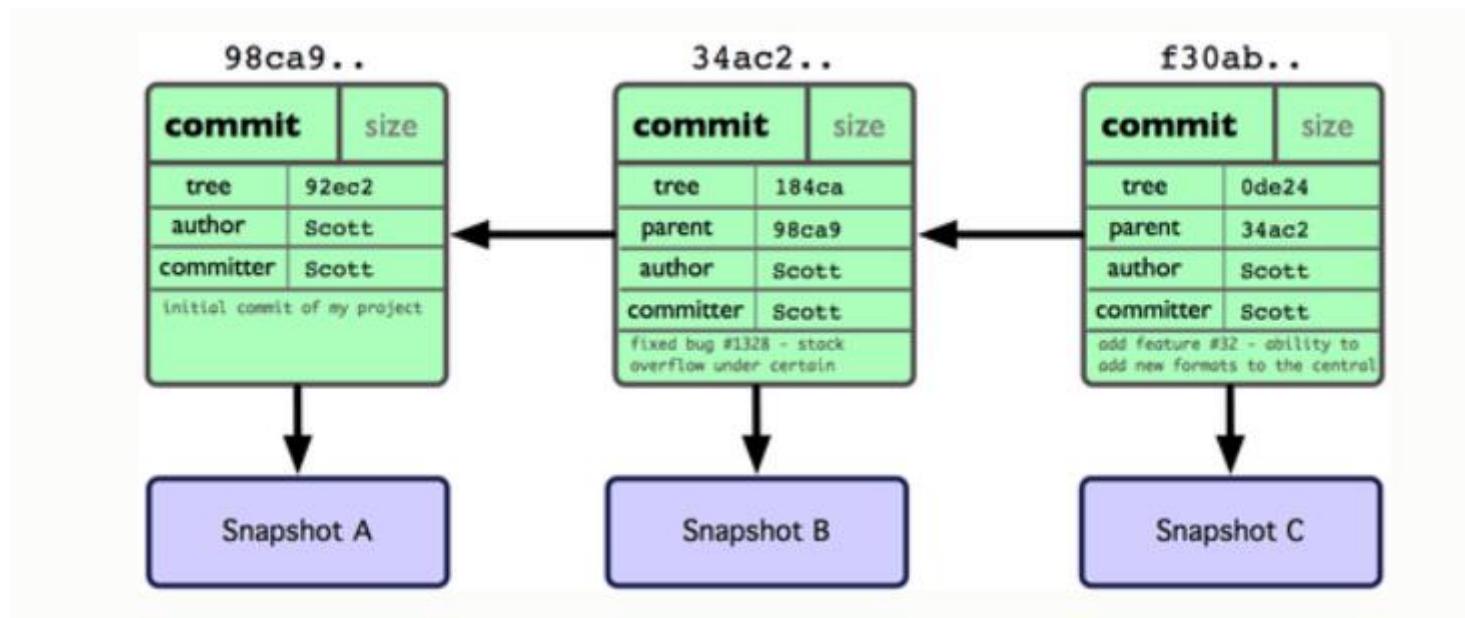


- Κάθε commit δημιουργεί tree objects και ένα blob για κάθε αρχείο που γίνεται commit. Όταν κάνουμε git add <aFile> το git δημιουργεί ένα blob (binary large object) που περιέχει τα περιεχόμενα του αρχείου.



Commit objects

Version Control με Git / GitHub



- Για κάθε καινούργιο commit δημιουργείται νέο commit object που δείχνει στο προηγούμενο



Τύποι Git Objects

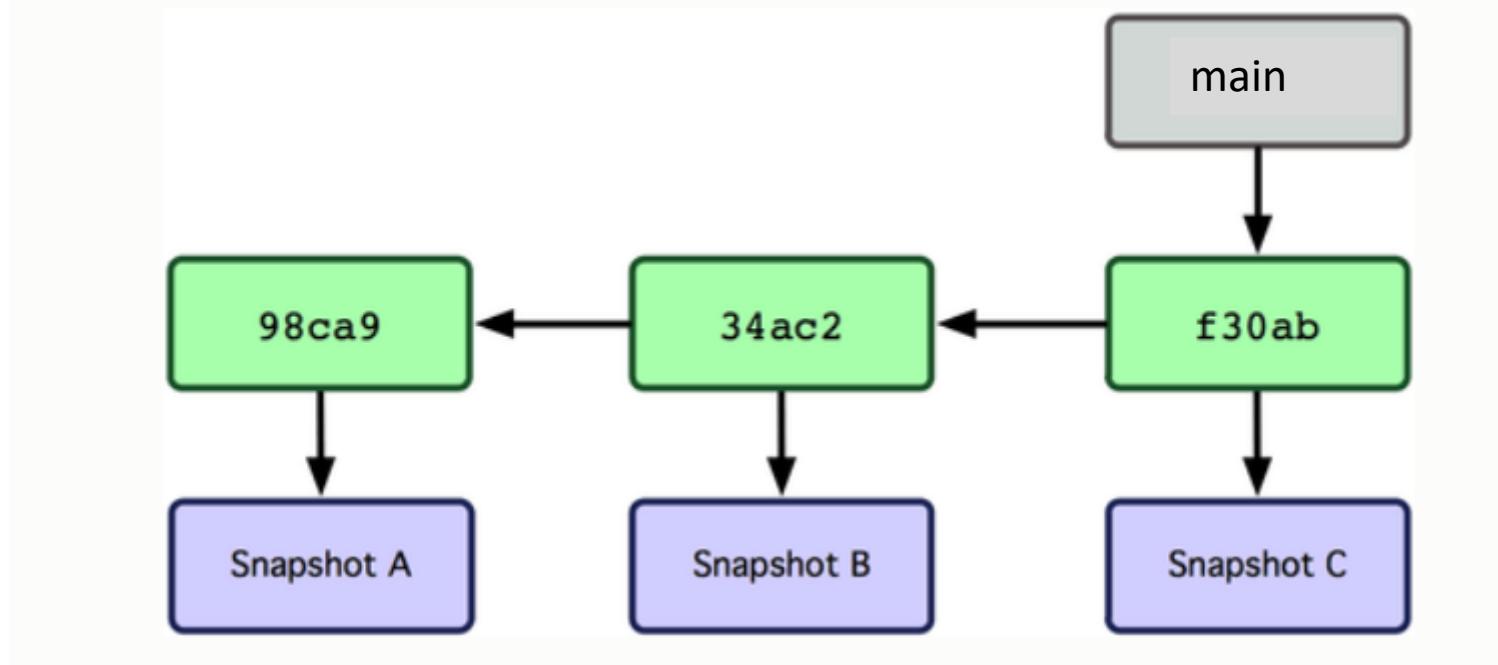
Version Control με Git / GitHub

- *commit*;
- *tree*;
- *blob*;
- *annotated tag*.



Branches – main branch

Version Control με Git / GitHub



Κάθε branch στο git είναι ένα δείκτης που μπορεί να μετακινείται μεταξύ των commits. Δείχνει στο τελευταίο commit και μετακινείται κάθε φορά που κάνουμε commit

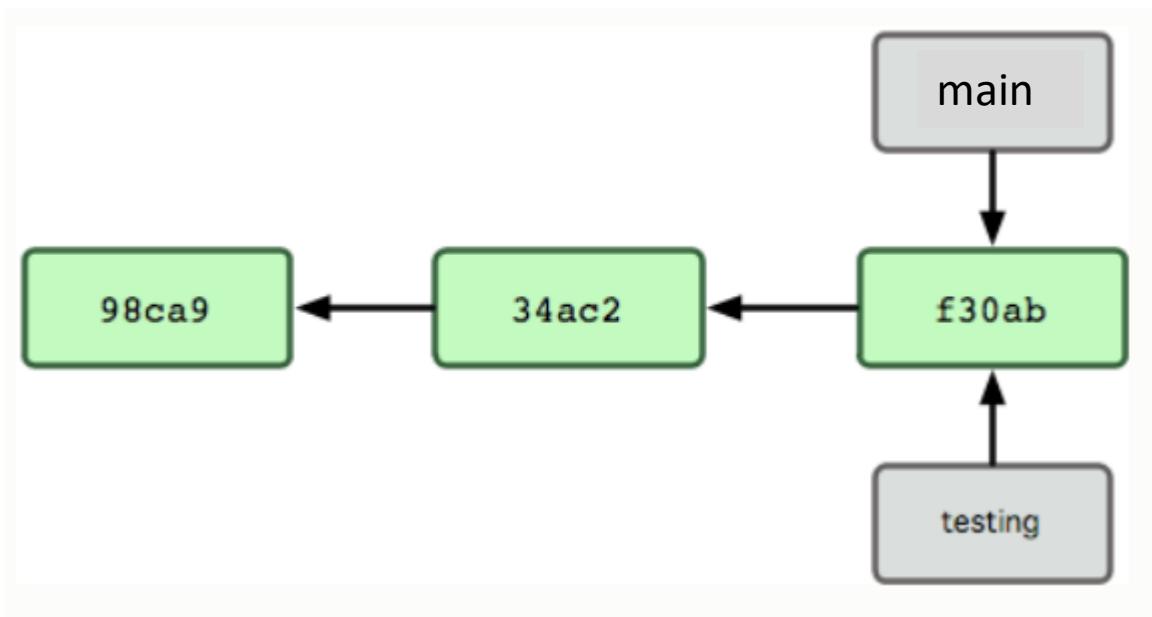


Δημιουργία νέου branch

Version Control με Git / GitHub

- **git branch testing**

- Δημιουργείται νέος δέικτης στο τελευταίο commit





HEAD (1)

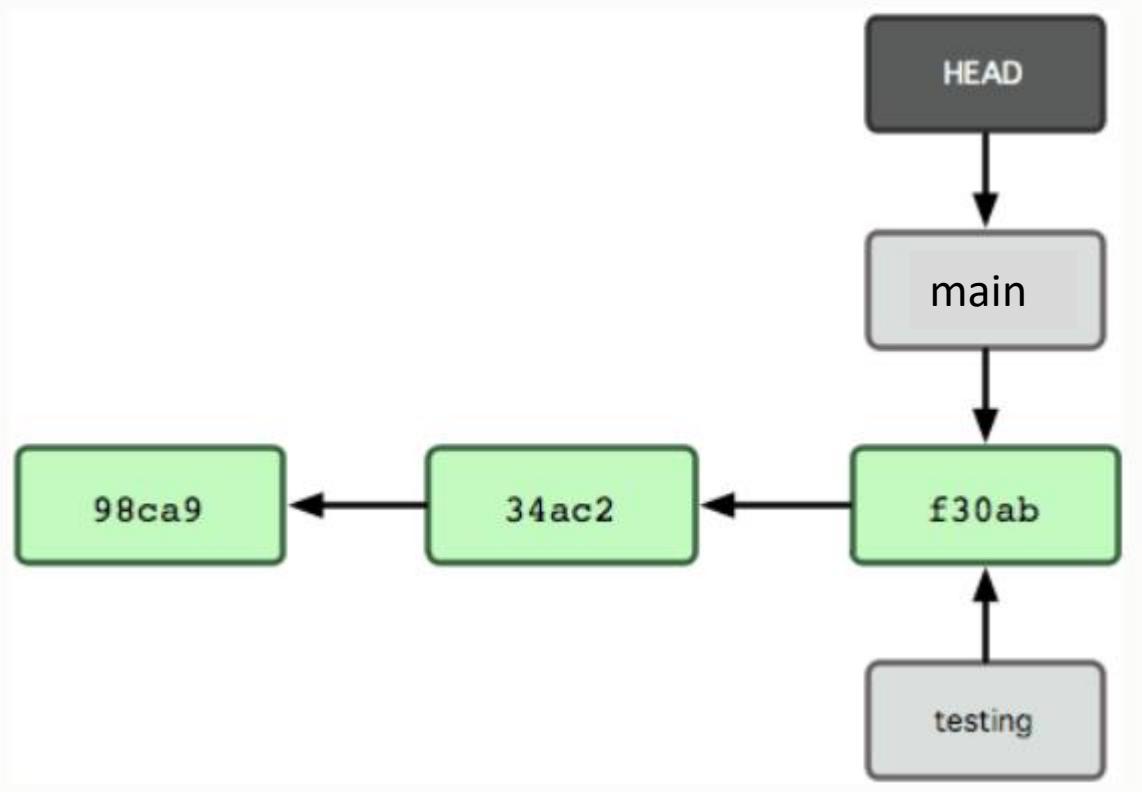
Version Control με Git / GitHub

- Πως το git γνωρίζει σε ποιο branch είμαστε;
- Χρησιμοποιεί ένα άλλο δείκτη (HEAD pointer) που δείχνει στο current branch
- Δείτε επόμενη διαφάνεια



HEAD(2)

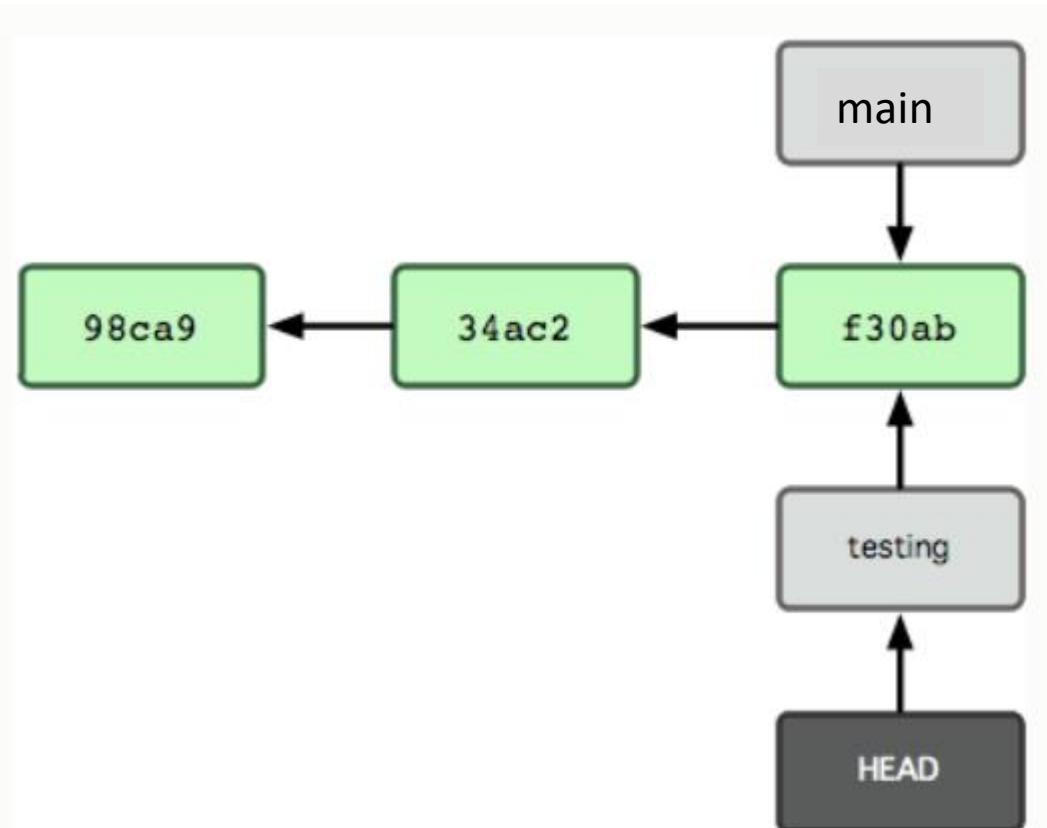
- Ο HEAD είναι ένας δείκτης που δείχνει στο current branch





HEAD (3)

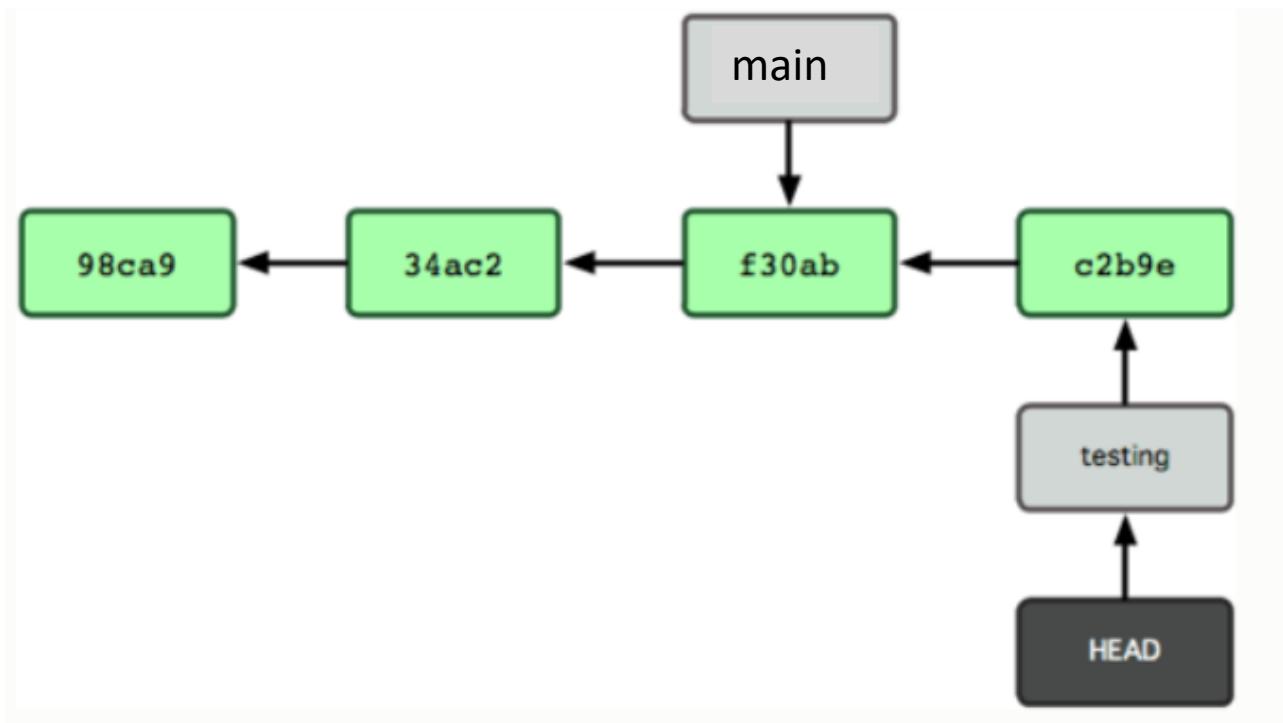
- **git checkout testing**





HEAD (4)

- **git commit –am “a change to a file”**



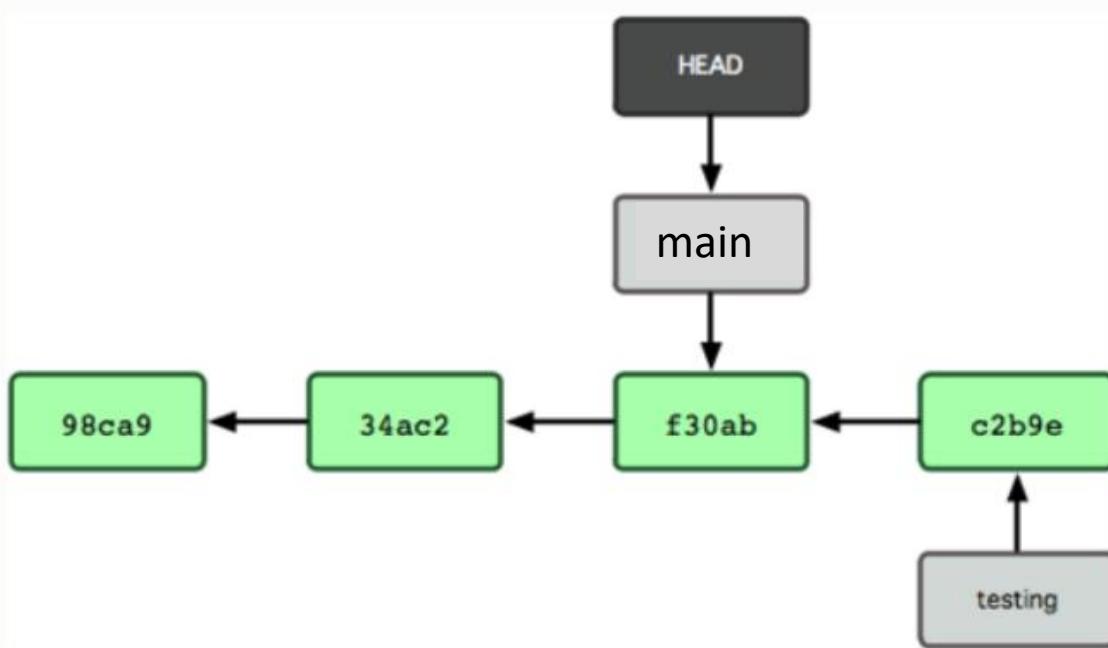
To HEAD προχωράει με κάθε commit

To ίδιο συνέβη και με το testing αφού το commit έγινε όταν είχαμε κάνει checkout στο branch testing



HEAD (5)

- **git checkout main**



To **HEAD** δείχνει στο *current branch*

Η εντολή αυτή έκανε δύο πράγματα

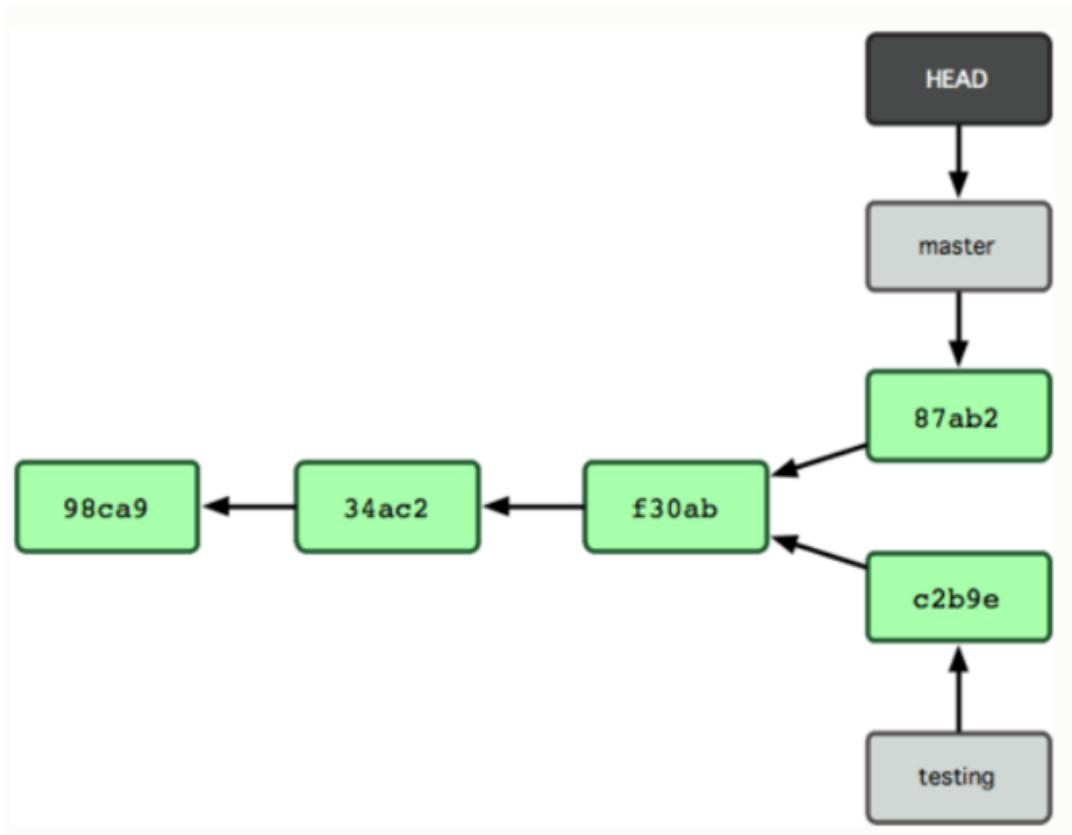
1. **Μετακίνησε τον HEAD πίσω** στο main branch, και
2. **Τοποθέτησε στο working directory τα αρχεία του snapshot που δείχνει ο main**

Ότι αλλαγές γίνουν εδώ θα δημιουργήσουν ένα άλλο μονοπάτι που θα διαφέρει από την παλιά version του project



HEAD (6)

- `git -a -m "new changes"`



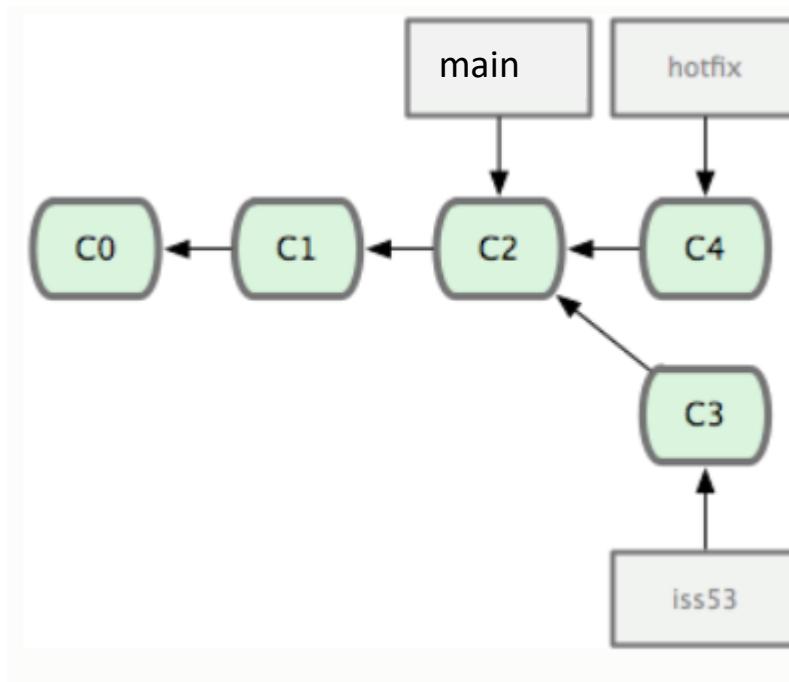
Οι αλλαγές μετά το 3^o commit βρίσκονται σε 2 διαφορετικά branches
Μπορούμε να δουλεύουμε στα 2 branches και όταν ολοκληρώσουμε να κάνουμε merge



Merging

Version Control με Git / GitHub

- `$ git checkout -b hotfix`
- `$ git -a -m "experiments"`

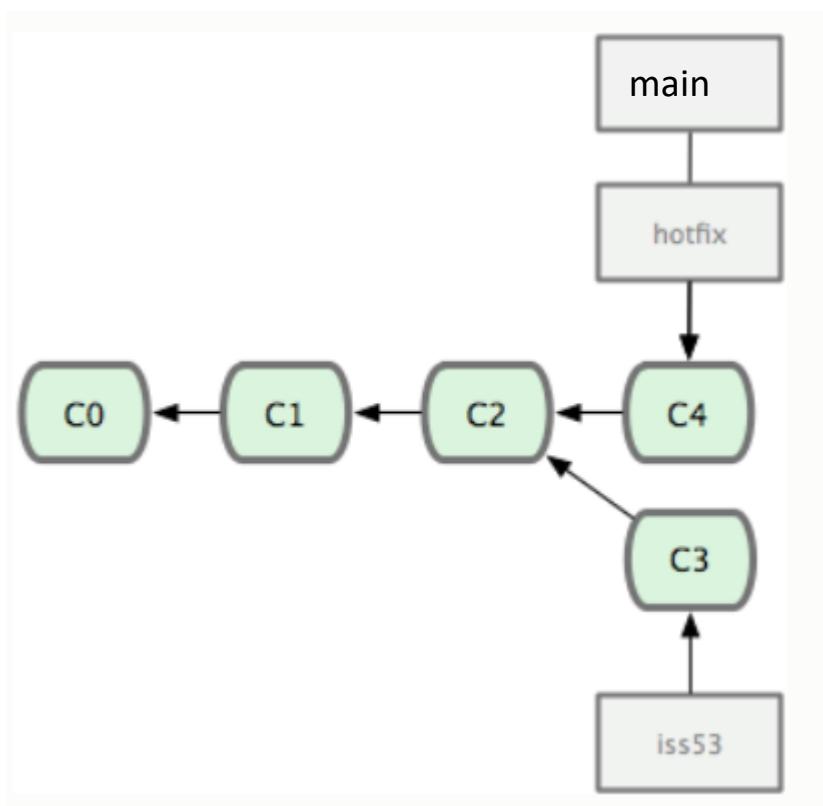


Κάνουμε πειράματα και αλλαγές στο hotfix και μετά κάνουμε merge στο main

`$ git checkout main`
`$ git merge hotfix`



Fast-forward



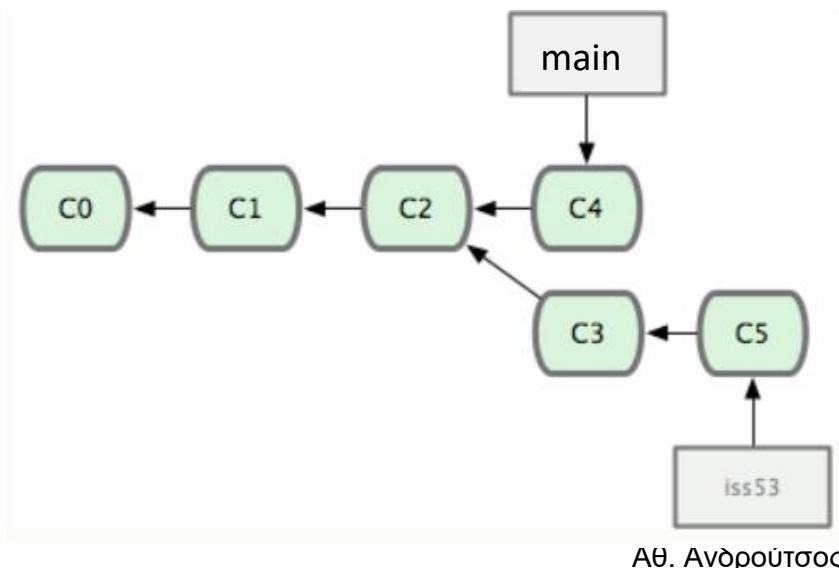
- Όταν κάνουμε merge δύο commits που το ένα είναι μπροστά από το άλλο, το git απλά μετακινεί τον δείκτη του current branch μπροστά
- Αυτό ονομάζεται fast-forward



Διαγραφή branch

Version Control με Git / GitHub

- Τώρα που κάναμε ένα merge με το hotfix μπορούμε να το διαγράψουμε και να συνεχίσουμε τη δουλειά μας
- `$ git branch -d hotfix`





Three-way merge (1)

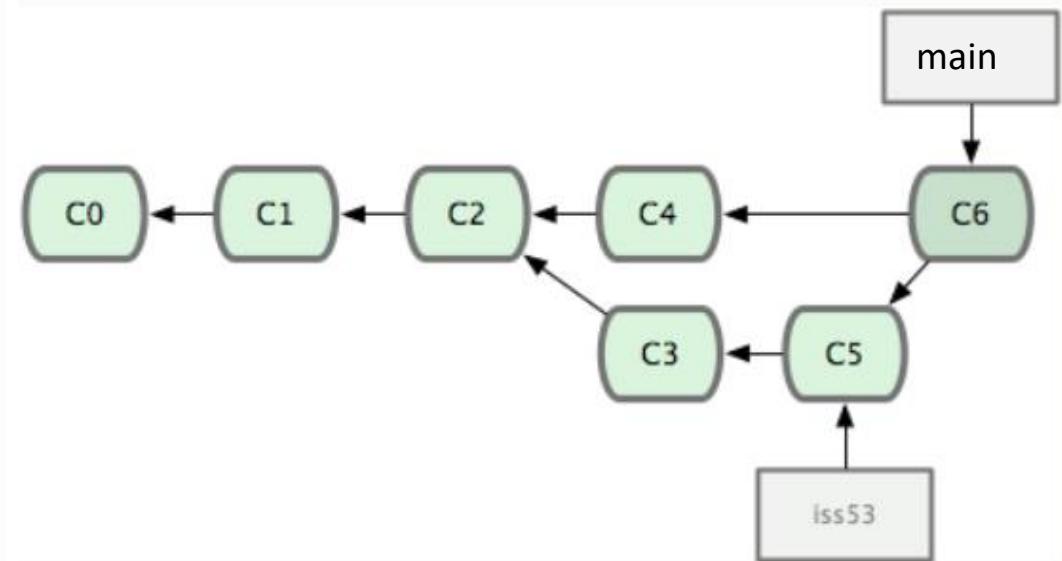
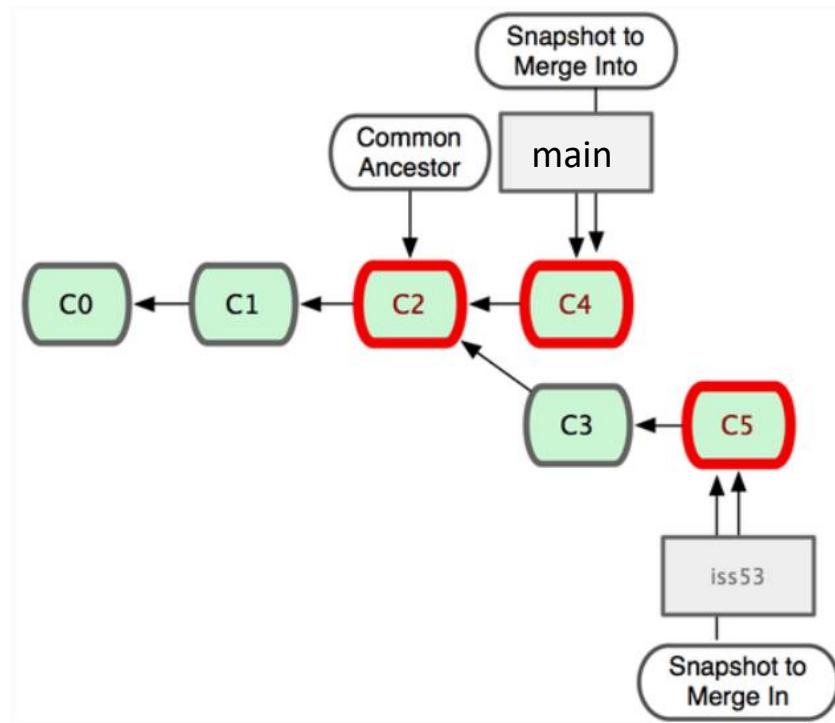
Version Control με Git / GitHub

- Αν θέλουμε να κάνουμε merge branches που είναι σε διαφορετικά μονοπάτια, το git δημιουργεί ένα merge commit, ένα νέο δηλαδή Commit βασισμένο στα δύο branches καθώς και στον κοινό τους πρόγονο



Three-way merge (2)

Version Control με Git / GitHub



- Τώρα μπορούμε να διαγράψουμε το iss53



SSH Keys (1)

Version Control με Git / GitHub

- Τα SSH Keys είναι κλειδιά κρυπτογράφησης. Επειδή η αποστολή username/passwords μέσω του δικτύου δεν είναι ασφαλής, θα θέλαμε να ακολουθήσουμε ένα διαφορετικό τρόπο ασφαλούς επικοινωνίας
- Καταρχάς η επικοινωνία με username/password δεν είναι ασφαλής αν αυτά φεύγουν μη-κρυπτογραφημένα αλλά και κρυπτογραφημένα πάλι 'ταξιδεύουν' στο δίκτυο και υπόκεινται σε απειλές



SSH Keys (2)

Version Control με Git / GitHub

- Γενικά, η λύση στη μεταφορά δεδομένων στο δίκτυο είναι η **κρυπτογράφηση**. Δεν υπάρχει άλλος τρόπος ασφάλειας στην επικοινωνίες.
- Υπάρχουν δύο ειδών μέθοδοι κρυπτογράφησης
 - **Symmetric**. Όπου η επικοινωνία γίνεται με την κρυπτογράφηση των δεδομένων με ένα κλειδί κρυπτογράφησης (shared secret) και αποκρυπτογράφησης στην άλλη πλευρά πάλι με το secret. Το πρόβλημα εδώ είναι η διανομή του κλειδιού (key distribution).
 - **Asymmetric**. Όπου δεν υπάρχει ένα μόνο κλειδί, αλλά ένα ζεύγος ιδιωτικού / δημόσιου κλειδιού (private / public key pair), όπου τα δύο αυτά σχετίζονται μεταξύ τους και μπορούμε να κρυπτογραφούμε με το ιδιωτικό μας κλειδί και η άλλη πλευρά να αποκρυπτογραφεί με το δημόσιο κλειδί μας. Δεν μας πειράζει η άλλη πλευρά να γνωρίζει το δημόσιο κλειδί μας, μιας και δεν μπορεί από το δημόσιο κλειδί να εξάγει το ιδιωτικό κλειδί. Το ιδιωτικό κλειδί είναι απολύτως προσωπικό και πρέπει να το κρατάμε ασφαλές



Private / Public Key Pair

Version Control με Git / GitHub

- Με το private/public key μπορούμε να λύσουμε το πρόβλημα της διανομής των κλειδιών της συμμετρικής κρυπτογράφησης μιας και μπορούμε να στείλουμε το κλειδί κρυπτογραφώντας το με το δημόσιο κλειδί του παραλήπτη και να είμαστε σίγουροι ότι μόνο παραλήπτης μπορεί να αποκρυπτογραφήσει με το δικό του ιδιωτικό κλειδί



GitHub Βήματα

Version Control με Git / GitHub

- 1. Δημιουργούμε ένα private / public key pair στον υπολογιστή μας
- 2. Πάμε στο GitHub και κάνουμε register το public key
- 3. Επικοινωνούμε με το GitHub χωρίς να χρειάζεται να δίνουμε username/password. Ουσιαστικά δεν κυκλοφορεί καν το password στο δίκτυο, μόνο η επικοινωνία μέσω SSH γίνεται
- Αν κατά τη δημιουργία του private/public key δώσουμε passphrase, τότε θα πρέπει να κάνουμε register το passphrase στον SSH Agent με

```
eval `ssh-agent`  
ssh-add
```



1ο Βήμα

Version Control με Git / GitHub

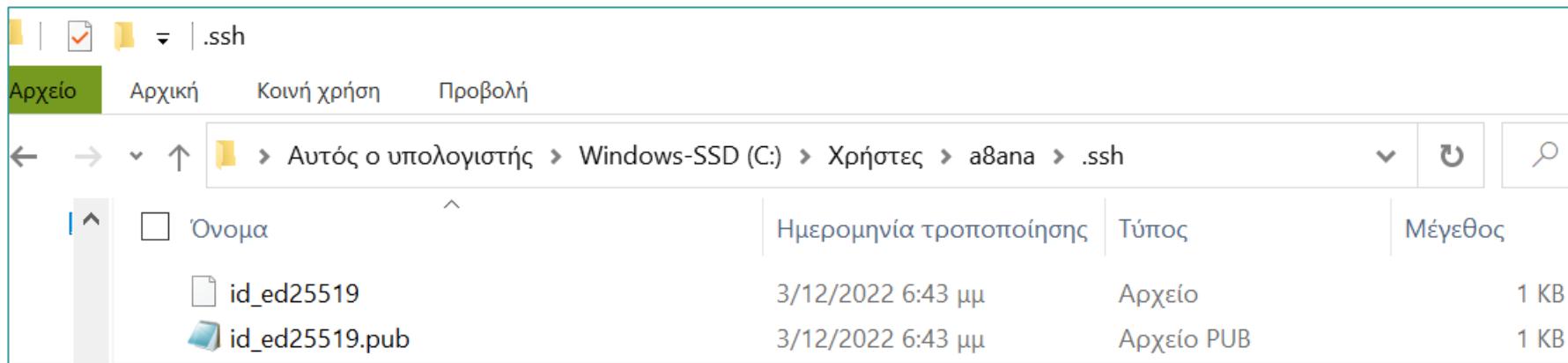
```
a8ana@thanassis-pc MINGW64 ~
$ ssh-keygen -t ed25519 -C a8anassis@gmail.com
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/a8ana/.ssh/id_ed25519):
Created directory '/c/Users/a8ana/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/a8ana/.ssh/id_ed25519
Your public key has been saved in /c/Users/a8ana/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:/8hpI8nZ+hmbOszjN5au2uN1HDDzd8mZNqMUBiZeEug a8anassis@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
 .+..
 .. =
 . = o
 E = . o +
 S o o X
 o + + o
 o * ..
 o*o*o
 .+%&%o.
+---[SHA256]---
```

- Δημιουργία public/private key με **ssh-keygen -t ed25519 -C <email>**
Δεν δίνουμε passphrase, αλλιώς αν δώσουμε θα πρέπει να κάνουμε τη διαδικασία με τον SSH Agent που αναφέρθηκε



Αρχείο private / public key

Version Control με Git / GitHub



- Δημιουργούνται δύο αρχεία στο user home dir (C:\Users\username σε windows ή /home/username σε Linux ή /Users/username σε macOS) στο οποίο αναφερόμαστε σε όλα τα Λειτουργικά Συστήματα ως ~
- Δημιουργείται το ~/.ssh/id_25519 που είναι το private key και το προσέχουμε μην το χάσουμε ή μη το αποκαλύψουμε καθώς και το ~/.ssh/id_25519.pub που είναι το δημόσιο κλειδί



GitHub Settings (1)

Version Control με Git / GitHub

The screenshot shows the GitHub Settings interface. On the left, there's a sidebar with links like Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and plans, Emails, Password and authentication, Sessions, and **SSH and GPG keys** (which is currently selected). The main content area has sections for SSH keys, GPG keys, and Vigilant mode. The SSH keys section shows a green "New SSH key" button. The GPG keys section shows a green "New GPG key" button. A red oval highlights the "New SSH key" button.

Athanassios Androultsos
Your personal account

SSH keys

New SSH key

GPG keys

New GPG key

Vigilant mode

Flag unsigned commits as unverified

- Πάμε στα settings του GitHub και στο μενού αριστερά επιλέγουμε **SSH and GPG keys** και μετά **New GPG Key**



GitHub Settings (2)

Version Control με Git / GitHub

SSH keys / Add new

Title
home-lenovo

Key type
Authentication Key

Key

```
ssh-ed25519 AAAAC3NzaC1IzDI1NTE5AAAAICn9OFdC/E1cJp5tOlJi2GxWuk2UsJk30RkRbTG+ZLRD
a8anassis@gmail.com|
```

Add SSH key

- Δίνουμε τίτλο καθώς και το key (το public key μας) που αρχίζει με ssh-ed25519 και τελειώνει με το e-mail



GitHub Settings (3)

Version Control με Git / GitHub

The screenshot shows the GitHub Settings page for the user 'Athanasios Androutso'. The left sidebar includes links for Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and plans, Emails, Password and authentication, Sessions, SSH and GPG keys (which is the active tab), Organizations, and Moderation. The main content area is titled 'SSH keys' and displays a list of associated keys. One key, 'home-lenovo', is listed with details: SHA256: /8hp18nZ+hmb0szjN5au2uN1HDDzd8mZNqMUBiZeEug, Added on Dec 3, 2022, and Never used — Read/write. A 'Delete' button is next to the key name. Below this, a note says 'Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#)'. The 'GPG keys' section below shows a message from GitHub: 'a8anassis@gmail.com received 1 new message' and '[GitHub] A new SSH authentication public key was added to your account'. It also provides instructions for removing the key if it was added in error.

Athanassios Androutso
Your personal account

SSH keys

New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication Keys

home-lenovo
SHA256: /8hp18nZ+hmb0szjN5au2uN1HDDzd8mZNqMUBiZeEug
Added on Dec 3, 2022
Never used — Read/write

Delete

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#).

GPG keys

a8anassis@gmail.com received 1 new message

[GitHub] A new SSH authentication public key was added to your account

The following SSH key was added to your account: home-lenovo
SHA256: /8hp18nZ+hmb0szjN5au2uN1HDDzd8mZNqMUBiZeEug If you believe this key was added in error, you can remove the key and disable access at the following location: <https://github.com/>

- Έρχεται μήνυμα επιβεβαίωσης



Test σύνδεση στο GitHub

Version Control με Git / GitHub

MINGW64:/c/Users/a8ana

```
a8ana@thanassis-pc MINGW64 ~
$ ssh -T git@github.com
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Hi a8anassis! You've successfully authenticated, but GitHub does not provide shell access.

a8ana@thanassis-pc MINGW64 ~
$
```

- Κάνουμε ένα test με ssh -T όπου με -T απενεργοποιούμε το interactivity. To git είναι ένας default user με τον οποίο συνδεόμαστε στο github.com
- Επιβεβαιώνουμε το public key του GitHub



Known hosts config file

Version Control με Git / GitHub

The screenshot shows a Windows File Explorer window with the path: Autός ο υπολογιστής > Windows-SSD (C:) > Χρήστες > a8ana > .ssh. The 'Αρχείο' tab is selected. The contents of the folder are listed in a table:

Όνομα	Ημερομηνία τροποποίησης	Τύπος	Μέγεθος
id_ed25519	3/12/2022 6:43 μμ	Αρχείο	1 KB
id_ed25519.pub	3/12/2022 6:43 μμ	Αρχείο PUB	1 KB
<input checked="" type="checkbox"/> known_hosts	22/5/2023 2:06 μμ	Αρχείο	1 KB
known_hosts.old	3/12/2022 7:03 μμ	Αρχείο OLD	1 KB

- Δημιουργείται το known_hosts με τα public keys του GitHub ώστε να είναι trusted
- Το αρχείο αυτό δημιουργείται κατά την 1^η σύνδεση (ή το δημιουργούμε εμείς manually) και χρειάζεται από τον SSH Client για να συνδέεται με το SSH Servers ελέγχοντας το authenticity του remote server
- Το .old είναι backup



GitHub Public Key Fingerprints

Version Control με Git / GitHub

You can add the following ssh key entries to your `~/.ssh/known_hosts` file to avoid manually verifying GitHub hosts:

```
github.com ssh-ed25519
AAAAAC3Nzac1lZDI1NTE5AAAAIOMqqnkVzrm0SdG6UOoqKLsabgH5C9okWi0dh2l9GKJ1
github.com ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAIBmlzdHAyNTYAAABBEmKSENjQEez0mxkZMy7opKgwFB9nkt5YRrY
MjNuG5N87uRgg6CLrbo5wAdT/y6v0mKV0U2w0WZ2YB/++Tpockg=
github.com ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQgQCj7ndNxQowgcQnjshcLrqPEiiphnt+VTTvDP6mHBL9j1aNUnkY4Ue1gvwnG
Lv1ohGeYrnZaMgRK6+PKCUXaDbC7qtbW8gIkhl7aGCS0r/C56SJMy/BCZfxd1nWzAOxSDPgVsmerOBYfNqltV9/h
WCqBywINIR+5dIg6JTJ72pcEpEjcYgXkE2YEFXV1JHnsKgbLWNlhScqb2UmyRkQyytRLtL+38TGxkxclfmo+5Z8C
SSNY7GidjMIZ7Q4zMjA2n1nGr1TDkzwDCsw+wqFPGQA179cnfGWOWRVruj16z6XyvxvjJwbz0wQZ75XK5tKsb7FN
yeIEs4TT4jk+s4dhPeAUC5y+bDYirYgM4GC7uEnztnZyaVWQ7B381AK4Qdrwt51ZqExKbQpTUNn+EjqoTwvqnj4k
qx5QUCI0Ths/Yk0xJCXmPUWzbhjpCg56i+2aB6CmK2JGhn57K5mj0MNdBXA4/WnwH6XoPWJzK5Nyuu2zB3nAZp+S5
hpQs+p1vN1/wsjk=
```

- Για manually προσθήκη στο `known_hosts` του GitHub fingerprint (Hash του public key του GitHub)