



ANIMATION



AUDIO



DESIGN



FILM



GAMES



WEB & MOBILE

# GIT Workshop

Basic Knowledge

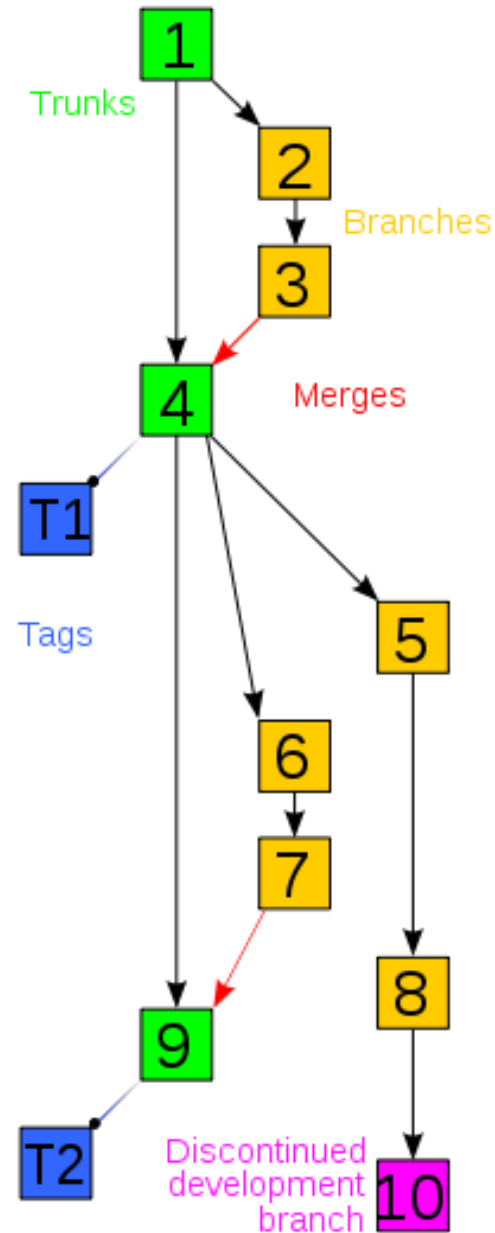
# Version Control Systems (VCS)

- Version Control: έλεγχος αλλαγών στο software → έλεγχος ονομασιών των εκδόσεων του software (πχ index\_1.html, index\_2.html, index\_3.html)
- Version Control System: ολοκληρωμένο σύστημα διαχείρισης ελέγχου των εκδόσεων
  - Stand Alone Application
  - Cloud Services

# Examples/Terms

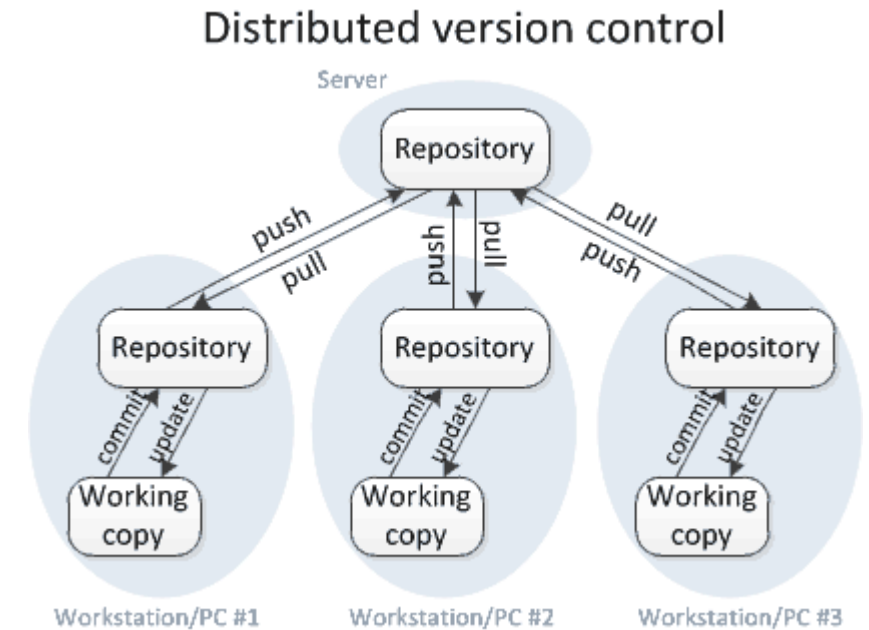
Ακολουθείται δομή δένδρου με κλαδιά:

- Βασική γραμμή development: baseline ή mainline ή trunk ή root
- Εκδόσεις που δεν θέλουμε να επηρεάσουν την root: branches
- Ένωση branch με την root: merge
- Ονομασία κάποιου branch/ subversion: Tags/Labels



# Χρήση Version Control

- Backup/Archive
- Versioning/History
- Comparing versions & Undo Changes
- Συνεργασία (ομάδες ανάπτυξης)
- Πειραματισμός χωρίς να επηρεαστεί το αρχικό project

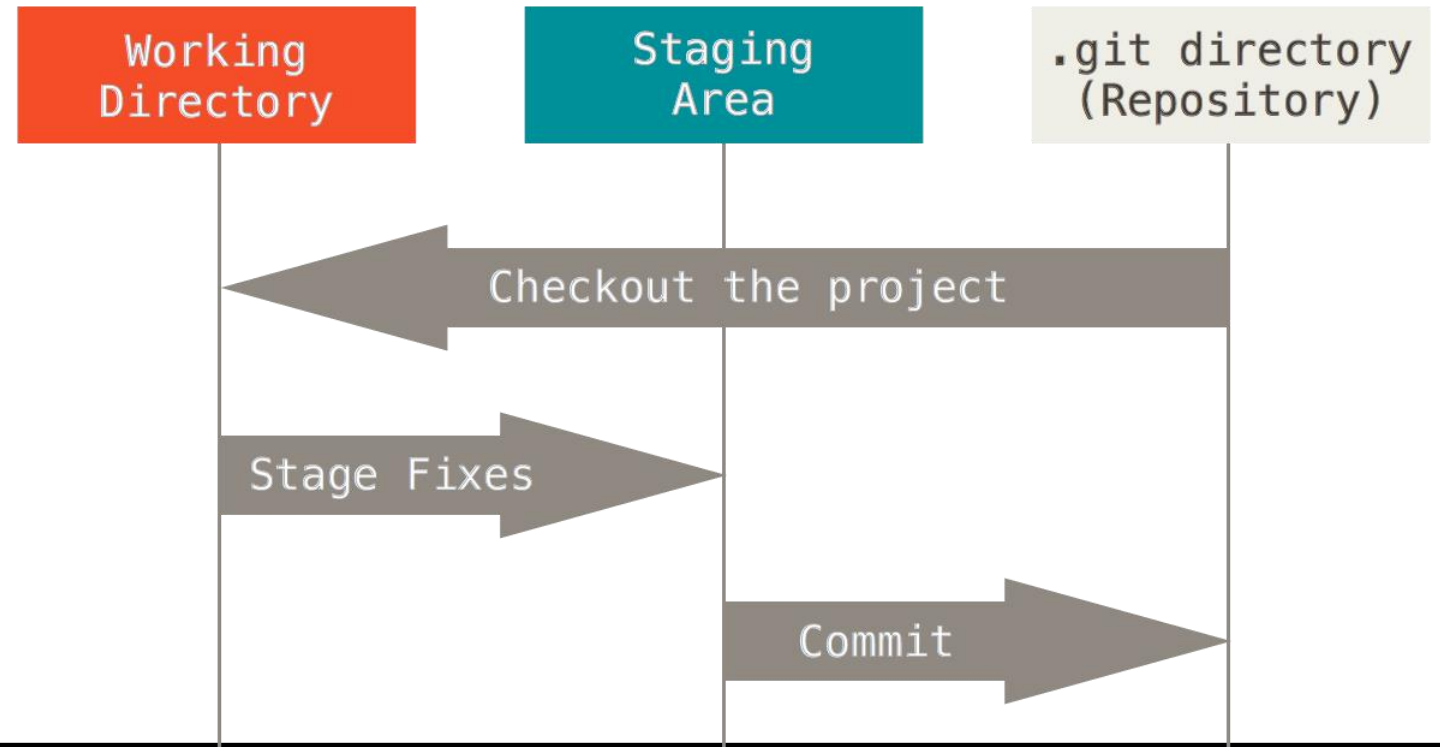


# GIT Terminology #1

Repository (Working Directory & GIT directory)

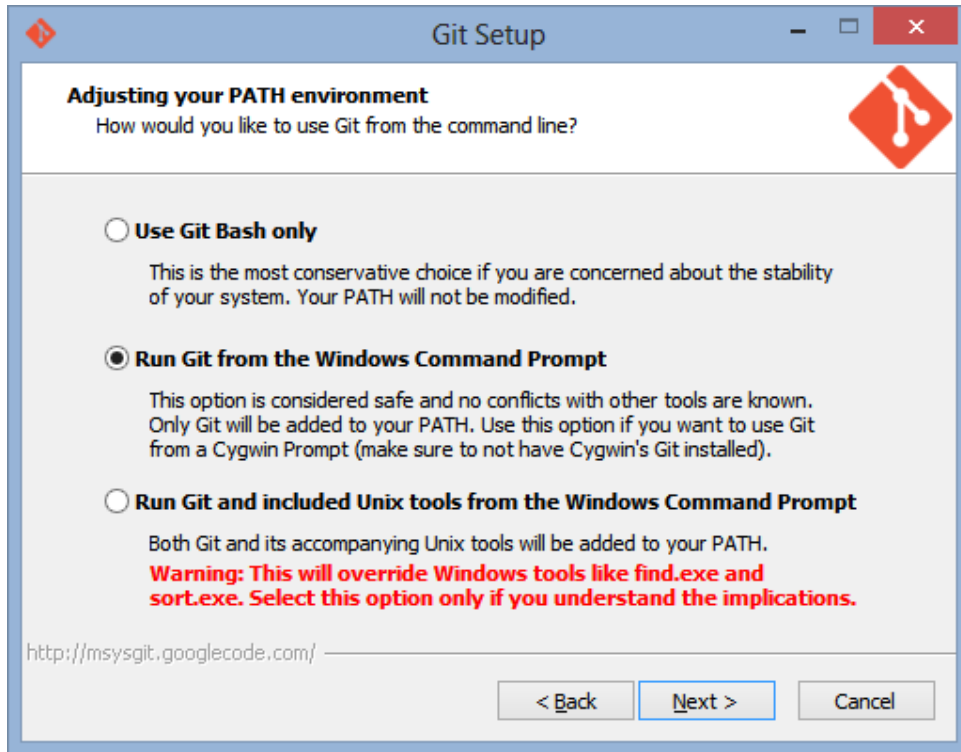
Καταστάσεις του Project μέσω GIT:

- Local Computer:
  - Working Directory
  - Staging Area
  - Repository (.git folder)
- Remote Repository



# Εγκατάσταση (Windows)

- <http://git-scm.com/>

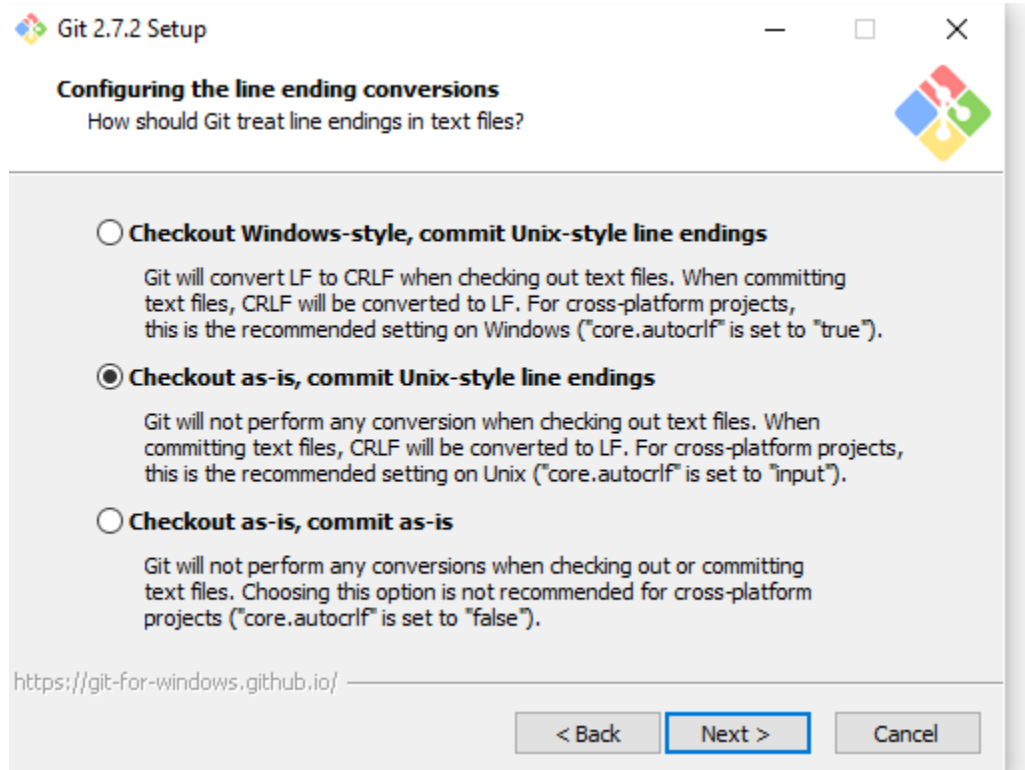


Επιλογή των πακέτων του GIT που χρειαζόμαστε.

Το GIT Bash είναι η γραμμή εντολών για το git (αντί command prompt)

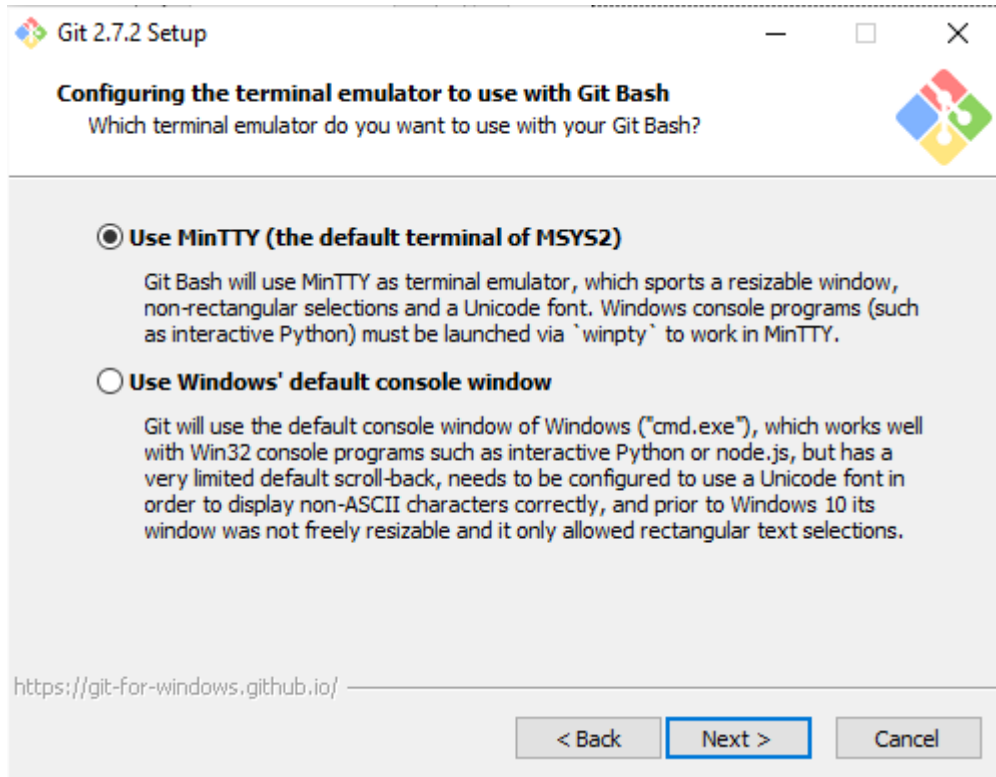
Η δεύτερη επιλογή (προτείνεται) μας δίνει δυνατότητα να τρέχουμε εντολές και στο git-bash και στο command prompt

# Εγκατάσταση (Windows)



- Line Endings: windows & unix έχουν διαφορετικό χαρακτήρα για την αλλαγή γραμμής
- Προτείνεται το Unix – style καθώς θα χρησιμοποιούμε terminal και πρέπει να φροντίσουμε την περίπτωση που κάποιος συνεργάτης έχει mac/linux

# Εγκατάσταση (Windows)



- Είδος Terminal που θα χρησιμοποιεί το git-bash
- Επιλέγουμε το MinTTY που μας δίνει χρωματικές επιλογές



# Εγκατάσταση (MacOS)




- MacOS: Terminal → git version (προαπαιτεί το Xcode)
- Ανοίγουμε Terminal και γράφουμε git version
- Αν έχουμε ήδη το Xcode θα μας βγάλει την έκδοση που έχουμε στο GIT
- Αλλιώς θα μας βγάλει την διαδικασία εγκατάστασης του
- Μπορούμε να επιλέξουμε ένα terminal alternative για να έχουμε χρωματικούς συνδυασμούς όπως το xterm

# GitHub

- Το GitHub είναι μια υπηρεσία που προσφέρει VCS
- Υπάρχουν πολλές Git υπηρεσίες
- Το GitHub είναι το πιο δημοφιλές

## Join GitHub

The best way to design, build, and ship software.

 <b>Step 1:</b> Set up a personal account	 <b>Step 2:</b> Choose your plan	 <b>Step 3:</b> Go to your dashboard
---	--	--

### Create your personal account

Username

This will be your username — you can enter your organization's username next.

Email Address

You will occasionally receive account related emails. We promise not to share your email with anyone.

Password

Use at least one lowercase letter, one numeral, and seven characters.

By clicking on "Create an account" below, you are agreeing to the [Terms of Service](#) and the [Privacy Policy](#).

Create an account

### You'll love GitHub

**Unlimited** collaborators

**Unlimited** public repositories

- ✓ Great communication
- ✓ Friction-less development
- ✓ Open source community


# Getting Started

- Δημιουργούμε ένα νέο repository στο οποίο θα τοποθετήσουμε τα αρχεία μας
- Προσοχή: όχι κενά, απλά ονόματα
- Ελέγχουμε την κατάσταση του repository (public/private)

## Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

 apostolossae ▾

Repository name

wddexample ✓

Great repository names are short and memorable. Need inspiration? How about **fuzzy-invention**.

Description (optional)

☒ Public

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

☐ Private

You choose who can see and commit to this repository.

☐ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾

Add a license: None ▾



# Main Repository Page

[Code](#) [Issues 0](#) [Pull requests 0](#) [Wiki](#) [Pulse](#) [Graphs](#) [Settings](#)

This is a example repository for SAE WDD Classes — [Edit](#)

[1 commit](#) [1 branch](#) [0 releases](#) [1 contributor](#)

[Branch: master](#) [New pull request](#) [New file](#) [Upload files](#) [Find file](#) [HTTPS](#) <https://github.com/aposto> [Download ZIP](#)

[apostolossae](#) Initial commit Latest commit 265a583 just now

[README.md](#) Initial commit just now

[README.md](#)

## wddexample

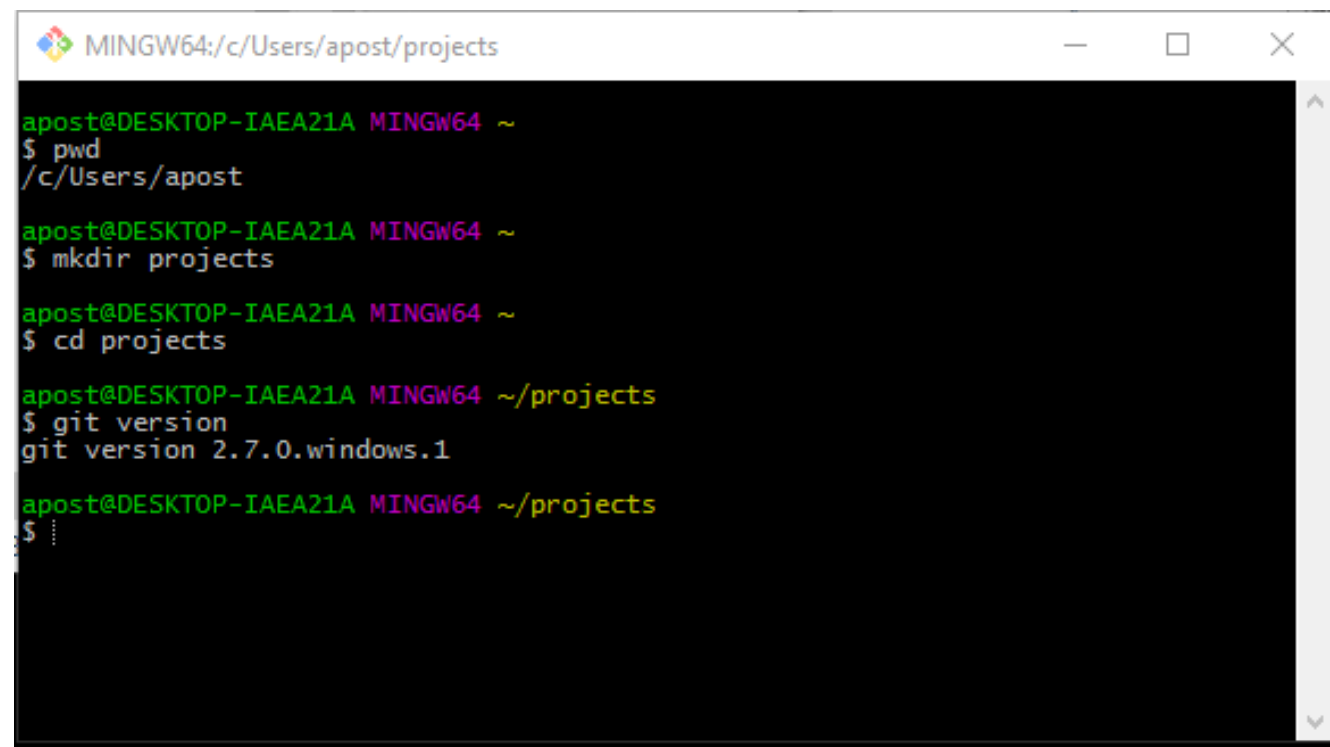
This is a example repository for SAE WDD Classes

# Connect with project

- Για να συνδεθούμε με το repository που έχουμε δημιουργήσει θα πρέπει να χρησιμοποιήσουμε είτε κάποιο software που μας επιτρέπει κατευθείαν σύνδεση με το GIT είτε θα συνδεθούμε μέσω terminal και θα επιλέξουμε με ποιον editor θα ελέγχουμε το repository (προτείνεται)
- Οι περισσότεροι editors προσφέρουν κάποιο plugin για σύνδεση με GIT

# Getting Started

- pwd: μας δείχνει τον φάκελο που βρισκόμαστε (Print Working Director)
- mkdir: δημιουργεί ένα νέο φάκελο
- cd: αλλάζει directory



```
MINGW64:/c/Users/apost/projects
apost@DESKTOP-IAEA21A MINGW64 ~
$ pwd
/c/Users/apost
apost@DESKTOP-IAEA21A MINGW64 ~
$ mkdir projects
apost@DESKTOP-IAEA21A MINGW64 ~
$ cd projects
apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ git version
git version 2.7.0.windows.1
apost@DESKTOP-IAEA21A MINGW64 ~/projects
$
```

<https://www.git-tower.com/learn/git/ebook/command-line/appendix/command-line-101>

# Global Configuration

- Για το γενικό configuration χρησιμοποιούμε την git config --global εντολή
- user.name: το όνομα μας
- user.email: το email μας
- --global --list: εμφανίζει τα δεδομένα που βάλαμε
- clear καθαρίζει το terminal

```
MINGW64:/c/Users/apost/projects

apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ git config --global user.name "Apostolos"

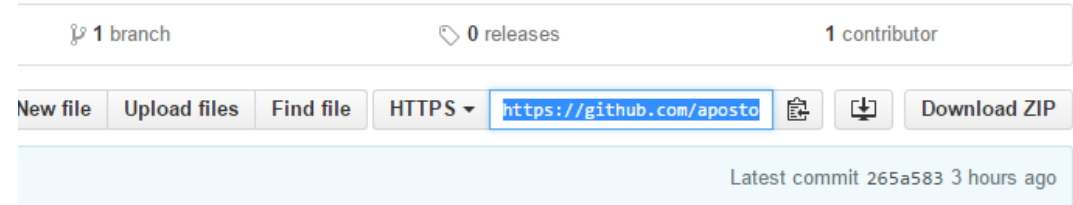
apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ git config --global user.email "a.ioannidis@sae.edu"

apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ git config --global --list
user.name=Apostolos
user.email=a.ioannidis@sae.edu

apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ |
```

# Download project

- Βρίσκουμε το url του project μας από το git web service και το αντιγράφουμε
- Μέσω του git-bash χρησιμοποιούμε την git clone εντολή για να δημιουργήσουμε ένα αντίγραφο του git project μας στον υπολογιστή μας (local area)



```
MINGW64:/c/Users/apost/projects
apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ git clone https://github.com/apostolossae/wddexample.git
Cloning into 'wddexample'...
remote: Counting objects: 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), done.
Checking connectivity... done.

apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ pwd
/c/Users/apost/projects

apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ ls
wddexample/

apost@DESKTOP-IAEA21A MINGW64 ~/projects
$ |
```



# Basic Control of files

- Με τον τελεστή > δημιουργούμε αρχείο
- Με το ls κάνουμε list τα αρχεία του φακέλου μας
- Με το git status ελέγχουμε σε ποιο stage βρισκόμαστε
- Με το git add <filename> μεταφέρει το αρχείο στο staging area
- Με το git commit <filename> μεταφέρουμε το αρχείο στο git local repository
- Με το git push origin master στέλνουμε όλες τις αλλαγές

```
MINGW64:/c:/Users/apost/projects/wddexample

apost@DESKTOP-IAEA21A MINGW64 ~/projects/wddexample (master)
$ >index.html

apost@DESKTOP-IAEA21A MINGW64 ~/projects/wddexample (master)
$ ls
index.html  README.md

apost@DESKTOP-IAEA21A MINGW64 ~/projects/wddexample (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        index.html

nothing added to commit but untracked files present (use "git add" to
apost@DESKTOP-IAEA21A MINGW64 ~/projects/wddexample (master)
$ |
```

# Text Editor

- Μπορούμε να «συνδέσουμε» οποιονδήποτε code editor με το git-bash και κατ'επέκταση το github
- Δημιουργούμε ένα αρχείο .bash\_profile στο user root
- Μέσα δημιουργούμε ένα alias για να φορτώνει τον text editor που επιθυμούμε:

```
alias npp="/c/Program\ Files\ \ (x86\)/Notepad++/notepad++.exe"
```

```
alias brk="/c/Program\ Files\ \ (x86\)/Brackets/Brackets.exe"
```

- Προσοχή στα slash & backslash

# Push/Pull

- Η διαδικασία προσθήκης κάποιου αρχείου στο git online repository ολοκληρώνεται με την push εντολή
- Προτείνεται πριν γίνει οποιοδήποτε push να γίνει ένα pull έτσι ώστε να ενημερωθούμε για τυχόν αλλαγές που έχουν γίνει από άλλους contributors
- Στις pull/push εντολές ορίζουμε το branch που θέλουμε να ελέγξουμε



# Fresh Project

- Μπορούμε να δημιουργήσουμε ένα νέο Project:
- Μέσω του online service (πχ github)
- Μέσω του terminal
- Μέσω του γραφικού περιβάλλοντος του OS
- Δεν ξεχνάμε να κάνουμε git init μέσα στον φάκελο του project για να ενεργοποιηθεί

```
C:\Users\apost\projects>mkdir test
C:\Users\apost\projects>cd test
C:\Users\apost\projects\test>git status
fatal: Not a git repository (or any of the parent directories): .git
C:\Users\apost\projects\test>git init
Initialized empty Git repository in C:/Users/apost/projects/test/.git
C:\Users\apost\projects\test>git status
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
C:\Users\apost\projects\test>
```

```
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git push dumorigin master
Counting objects: 20, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (19/19), done.
Writing objects: 100% (20/20), 67.21 KiB | 0 bytes/s, done.
Total 20 (delta 0), reused 0 (delta 0)
To https://github.com/apostolossae/dummy.git
 * [new branch]      master -> master
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ |
```

# Existing Project

Για ένα υπάρχων project ακολουθούμε αντίστοιχη διαδικασία:

- Μέσω του command prompt κάνουμε git init στον φάκελο του project και ακολουθούμε την ίδια διαδικασία (add→commit→pull&push)
- Προσοχή: θα πρέπει να υπάρχει κάποιο online repository ήδη στο account σας στο github
- Μπορούμε να ορίσουμε νέα repositories & branches είτε μέσω του github panel είτε μέσω του command prompt

```
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git add .

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   .htaccess
    new file:   apple-touch-icon.png
    new file:   browserconfig.xml
    new file:   css/main.css
    new file:   css/normalize.css
    new file:   css/normalize.min.css
    new file:   favicon.ico
    new file:   index.html
    new file:   js/main.js
    new file:   js/plugins.js
    new file:   js/vendor/jquery-1.11.2.min.js
    new file:   js/vendor/modernizr-2.8.3-respond-1.4.2.min.js
    new file:   robots.txt
    new file:   tile-wide.png
    new file:   tile.png

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git commit -m "All files in dumm"
[master (root-commit) 6bf8ce8] All files in dumm
15 files changed, 1882 insertions(+)
create mode 100644 .htaccess
create mode 100644 apple-touch-icon.png
create mode 100644 browserconfig.xml
create mode 100644 css/main.css
create mode 100644 css/normalize.css
create mode 100644 css/normalize.min.css
create mode 100644 favicon.ico
create mode 100644 index.html
create mode 100644 js/main.js
create mode 100644 js/plugins.js
create mode 100644 js/vendor/jquery-1.11.2.min.js
create mode 100644 js/vendor/modernizr-2.8.3-respond-1.4.2.min.js
create mode 100644 robots.txt
create mode 100644 tile-wide.png
create mode 100644 tile.png

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git remote add origin https://github.com/apostolossae/dummy.git

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git remote -v
origin https://github.com/apostolossae/dummy.git (fetch)
origin https://github.com/apostolossae/dummy.git (push)

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git pull origin master
From https://github.com/apostolossae/dummy
 * branch            master       -> FETCH_HEAD
Already up-to-date.

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git push origin master
Everything up-to-date

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$
```

# Fork Existing Project

- Για να συνδεθούμε με ένα υπάρχων project στο github θα πρέπει να έχουμε το url του
- Στην συνέχεια επιλέγουμε το Fork
- Δημιουργείται ένα copy του project στο account μας
- Για να επεξεργαστούμε το project από τον υπολογιστή μας θα πρέπει να το κάνουμε clone ή να χρησιμοποιήσουμε το αντίστοιχο desktop app



# Backing out changes & History

- Εφόσον δεν έχει γίνει commit/push το project μας μπορούμε εύκολα να γυρίσουμε πίσω σε αλλαγές μέσω της git reset & της git checkout --<file>
- Μέσω της git log μπορούμε να δούμε προηγούμενα commits και ανάλογα να μεταβούμε σε αυτά μέσω της checkout εντολής
- Στην συνέχεια μπορούμε να δημιουργήσουμε νέο branch και να δουλέψουμε σε αυτό ή απλά να συνεχίσουμε από εκείνο το commit και μετά

```
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git checkout 512b046
Note: checking out '512b046'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b <new-branch-name>

HEAD is now at 512b046... Change other Paragraph Titles
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm ((512b046...))
$ |
```

# Branches & Merge

- Τα branches μας βοηθάνε να κάνουμε αλλαγές στο project μας χωρίς να επηρεάζουμε το αρχικό μας
- Αφού γίνουν όλες οι αλλαγές μπορούμε να τα κάνουμε merge είτε κατευθείαν (fast forward merge) είτε κρατώντας τις αλλαγές (disable fast forward) γράφοντας `git merge <branch> --no-ff`

```
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (minor-upgrade)
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git diff master minor-upgrade
diff --git a/index.html b/index.html
index e790959..9745174 100644
--- a/index.html
+++ b/index.html
@@ -26,9 +26,9 @@
     <h1 class="title">h1.title</h1>
     <nav>
       <ul>
-        <li><a href="#">nav ul li a</a></li>
-        <li><a href="#">nav ul li a</a></li>
-        <li><a href="#">nav ul li a</a></li>
+        <li><a href="#">Menu 1</a></li>
+        <li><a href="#">Menu 2</a></li>
+        <li><a href="#">Menu 3</a></li>
       </ul>
     </nav>
   </header>

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git merge minor-upgrade
Updating 5494f29..1e14694
Fast-Forward
 index.html | 6 +++---
 1 file changed, 3 insertions(+), 3 deletions(-)

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ |
```

```
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git log --oneline --graph --decorate
* f66fef9 (HEAD -> master) Merging Branches
* 512b046 (minor-upgrade) Change other Paragraph Titles
* e080042 Changed heading on paragraph
* 1e14694 (origin/minor-upgrade) Changed Menu Text
* 5494f29 (origin/master) Changed Title
* 6bf8ce8 (dumorigin/master) All files in dumm

apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ |
```



# Check Differences

- Για να ελέγξουμε τις διαφορές ανάμεσα στις εκδόσεις του branch μας χρησιμοποιούμε την `diff <branch1> <branch2>`
- Μπορούμε να χρησιμοποιήσουμε και κάποιο εργαλείο για απεικόνιση όπως το github desktop, p4merge ή κάποιο αντίστοιχο

```
apost@DESKTOP-IAEA21A MINGW64 ~/projects/dumm (master)
$ git diff master minor-upgrade
diff --git a/index.html b/index.html
index e790959..9745174 100644
--- a/index.html
+++ b/index.html
@@ -26,9 +26,9 @@
     <h1 class="title">h1.title</h1>
     <nav>
       <ul>
-        <li><a href="#">nav ul li a</a></li>
-        <li><a href="#">nav ul li a</a></li>
-        <li><a href="#">nav ul li a</a></li>
+        <li><a href="#">Menu 1</a></li>
+        <li><a href="#">Menu 2</a></li>
+        <li><a href="#">Menu 3</a></li>
       </ul>
     </nav>
   </header>
```

<https://git-scm.com/downloads/guis>

# Tagging

- Η λειτουργία των tags μας επιτρέπει να ξεχωρίσουμε τα branches & των commits
- Βοηθάνε στο versioning και στον διαχωρισμό των λειτουργιών που εκτελούμε
- Μπορούμε να συγκρίνουμε διαφορές ανάμεσα στα tags
- Να ελέγχουμε μόνο συγκεκριμένα tags

# ΛΙΝΚΣ

- [https://en.wikipedia.org/wiki/Version\\_control](https://en.wikipedia.org/wiki/Version_control)
- <https://www.git-tower.com/learn/git/ebook/command-line/appendix/command-line-101>
- <https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell>
- <https://www.siteground.com/tutorials/git/commands.htm>
- <https://37s.backpackit.com/pub/1465067>
- <http://cheat.errtheblog.com/s/git>