**Kocsis Márton 13.IRAÜ1**

**Github/Git:**

***1. Bevezetés, alapok:***

**- Repository**: Raktár/ Virtuális tárhely, rendszerez és hozzáférést biztosít.

**- Verziókezelő rendszer**: Megtartja egy fájl régi és új verzióját is, megnézhető a

módosítás ideje.

-A Github nem(!!) ugyanaz, mint a Git!

***2. Verziókezelők:***

-**Központosított verziókezelő** (*Centralized Version Constrol System*, CVCS):

Egy központi raktár/könyvtár (repository) van, amin mindenki feltölti a fájl új verzióit, frissítéseit. Folyamatosan frissíteni kell a fájlt a repo-ba. Hozzáférés (internet kapcsolat) kell a repo eléréséhez, különben nem lehet rá csatlakozni. Csak commit-olással működik.

-**Megosztott verziókezelő** (*Distributed Version Control System*, DVCS):

Van local (helyi) repository és remote (távoli, központi) repository. A helyi repo a saját számítógépen van, bármikor hozzá lehet férni, mindenkinek van egy saját helyi repo-ja. Nem feltétlenül kell a központi repo-hoz minden egyes verzió-t egyből feltölteni, csak a helyi repo-hoz. A központi repo-hoz ugyanúgy kell hozzáférés, de itt push-olással lehet feltölteni a verziókat a helyi repoból.

-**commit**: Végleges/új verzió, módosítás elküldése, feltöltése a repo-ba.

***3. Git Init:***

-Git kipróbálására/tanulására alkalmas weboldal: try.github.io

-Git eszköze: **terminál vagy Git BASH**

-Jó, ha a repo csak a **helyi gépen** létezik.

**3.1 parancsok:**

-**cd**: átvisz más könyvtárakba

-**dir** (*ls Linuxban*): listázás fájlok között

-**cls** (*clear Linuxban*): képernyő törlése

-**cd desktop**: felvisz a fő könyvtárba

-**mkdir**: mappa vagy könyvtár létrehozása

-**git init**: létrehozza a helyi könyvtárat (repo)

-**git status**: visszaad infókat a helyi repo(k)ról (milyen fájlok vagy adatok találhatóak)

-**touch**: létrehoz a gépen egy fájlt

***4. A színpad:***

-**színpad**: Fájlokat felküldünk vagy kitörlünk ide, még *mielőtt* felkerülnének a helyi repo-ba.

- **git add**: hozzáad fájlokat a *"színpadhoz"*

-**git commit -m "***valamilyen leírás***"**: Leírást/módosítást add a repo-ban lévő fájlokhoz (-m = message)

-**git commit**: fájlokat feltesz a helyi/központi repo-hoz

***5. Távoli repository:***

-**git add \***: minden fájlt felrakhatunk a színpadhoz

-**git add "\*"*.txt***: minden ***txt*** fájlt felrak a színpadhoz

-**git add \*.\***: mindent fel fog rakni a színpadhoz, fájltípustól függetlenül

-**git log**: Kilistázza, naplózza az eddig megtörtént commit-okat. Megjeleníti, hogy ki, mikor commit-olt. Commit mellett: színes, hosszú sor: azonosító száma a commit-nak.

-**git remote**: Rákapcsolódik a távoli elérésre, módosítani tudjuk azt.

-**git remote add**: Hozzáadjuk a távoli repo-t a saját reponk-hoz

-**git remote add origin** ***"url cím"***: add egy alias-t a fájlhoz, ahogy mi fogjuk azt „hívni”

-**git push origin**: hova push-oljon a helyi repo

-**git push origin master**: felpush-ol a helyi fő repo-ba (master = eredeti repo)

- **-u**: megjegyzi az előző origin-t (szobát)

-**Github**: Ingyenes, publikus account-tal lehet rendelkezni.

***6. Kapcsolódási módok:***

-**git config**: a beállításokban módosítunk valamit

-**git config** **--golbal** ***user.email***: e-mail cím módosítása, vagy beállítása

-**git config --global** ***user.name***: nickname beállítása a commit-olásoknál

-***Előbb jön létre a helyi repo mint a központi repo.***

-**git pull**: lehúzzuk a módosításokat a szerverről

-*git pull origin master --allow-unrelated-histories*: ha nem működik a pull a Git-en

-**master branch**: eredeti (fő) branch

-**development branch**: újonnan létrehozott branch

-**git clone "url cím"**: létrehozz egy lokális repo-t és egy mappát is hozzá

+ **lemásolja és összeköti** a távoli repo-val

***7. Fájl kezelés:***

-Csak akkor lehet push-olni ha minden az új verzión van a helyi repo-n/gépen.

-***Előbb pull-olunk, aztán push-olunk***.

-**git diff HEAD**: az előző, végleges változat megmutatása

-**git diff *"fájl"***: Egy adott fájl előző, végleges változatát mutatja meg.

-Ha valami a színpadon van, akkor nem működik a diff parancs!

-**git diff --staged**: megmutatja azokat a fájlokat, amik a színpadon vannak

-**git diff –cached *„fájl”***: Megmutatja a különbségeket egy adott fájlnál.

-**git reset *"fájl"***: leveszi a színpadról a fájlt

-**git checkout --*"fájl"****:* visszaviszi a fájlt az eredeti állapotába, kitöröl

minden módosítást

***8. Branch-ek:***

-A master és development branch-ek lehetnek a helyi repo-ban is

-**git branch *"név"***: új branch létrehozása

-**git branch**: kiírja a branch-eket

-**git checkout *"név"***: átváltás a másik branch-re

-**git rm "\*"**: kitöröl minden fájlt, kiírja a színpadon azt is, hogy mi hiányzik (törölt fájlok)

-**git rm *"\*.txt"***: kitöröl minden txt fájlt

-**git checkout branch**: vissza váltás a fő branch-re

-**git merge "név" branch**: áthúz mindent az éppen megjelenített branch-ből másik branch-re

-**git branch -d *"név"***: kitörli a megadott branch-et (-d = delete)