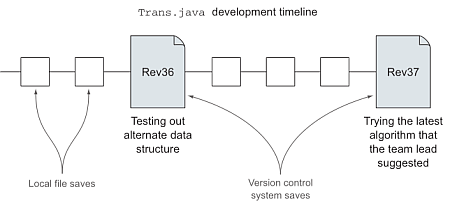
# Bevezetés

## Alapfogalmak

**Version control** (Verziókövetés): Változások követése, előző változatok megőrzése, szükség esetén visszatérés egy korábbi változathoz.

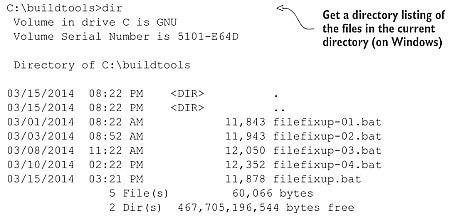
Egy program írása közben a lemezre gyakran mentünk, de csak azokat a változatokat kell megőrizni, amelyek egy-egy komolyabb módosítást tartalmaznak.



A szoftverek általában sok fájlból állnak, és mindegyiket követni kell.

**Repository** (tároló): A projekthez tartozó fájlok tárolására szolgál. Általában egy mappa, amelyben almappák is lehetnek.

Ha kézzel végezzük egy fájl verziókövetését, a mappánk így nézhet ki:



Több fájl esetén az egész mappából készíthetünk több változatot.

**Commit**: Egy pillanatfelvétel a fájlokról, amelyhez később vissza lehet térni.  
(RevXX, filefixup-XX)

Igeként *to commit*: egy pillanatfelvételt készíteni és elmenteni.

**Branch**: Ág, amelynek segítségével párhuzamosan fejleszthetünk.

Például:



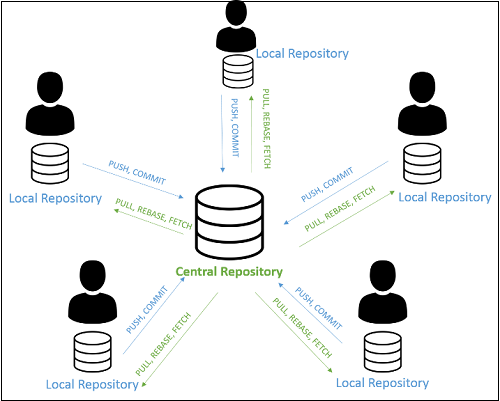
## git

A legelterjedtebb verziókezelő rendszer (version control system). Linus Torvalds vezetésével készítették a Linux kernel fejlesztéséhez.



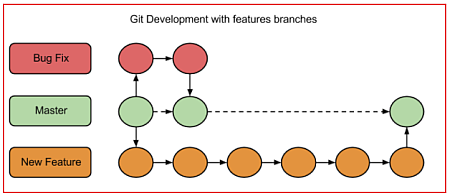
Legfontosabb jellemzői:

**Distributed repository**: Mindenkinek saját, helyi másolata van a repository-ról, és abban dolgozik. (így internet nélkül is tud dolgozni.) A módosításokat időnként fel kell tölteni a központi repository-ba (push), és onnan a többiek letöltik a saját példányaikba a változásokat (pull).



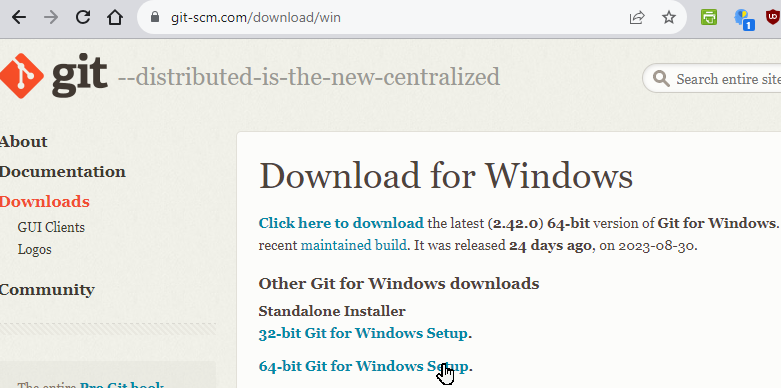
**Branching**: A fejlesztők külön ágakon dolgoznak, amelyeket később összefésülnek (merging).

Általában van egy fő ág, a master/main. Ebből készítenek elágazásokat különböző fejlesztésekhez, hibajavításokhoz. Amikor ezek elkészülnek, akkor összefésülik a fő ággal, majd törlik az elkészült ágat. (Vannak más, összetettebb stratégiák is.)



## Git telepítése

A Git-et Windowshoz a <https://git-scm.com/> címről töltheted le.



Ha nincs a gépedre telepítve a Git, töltsd le és telepítsd!

A telepítésnél nem kell semmit módosítani, csak a Next megnyomásával végig kell menni a beállításokon.

Kivétel ez alól az alapértelmezett szerkesztőprogram (editor) kiválasztása. Itt célszerű a VIM helyett egy egyszerűbb programot választani (például Notepad++ ).

## Alapvető parancsok

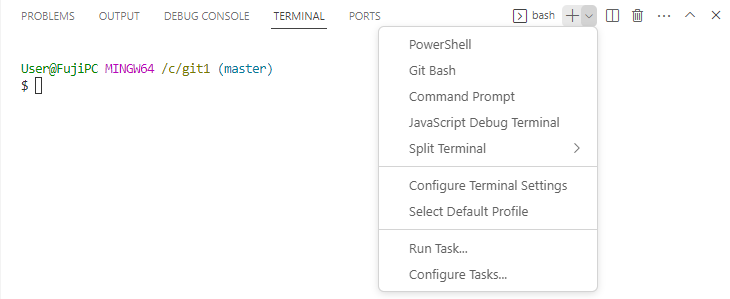
Ezután készíts egy mappát a C: meghajtóra, és nyisd meg a Visual Studio Code-ban!

A git parancssorból vagy grafikus felületről (pl. a Visual Studio Code-ból is) használható.

Először a parancssort tanuljuk, később pedig majd a Visual Studio Code lehetőségeit is megnézzük. Nyithatnák egy külön Parancssor vagy Terminal ablakot, de kényelmesebb a Visual Studio Code beépített terminálját használni.

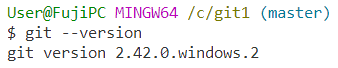
Nyiss egy terminált a Visual Studio Code-ban a Terminal/New terminal paranccsal, vagy a CTRL+Ö megnyomásával!

A Code-ban többféle terminal használható, most mindegy melyiket nyitod meg. Ha nem tetszik az alapértelmezett terminál, akkor a + melletti nyílra kattintva nyithatsz egy másikat, ezt pedig bezárhatod a kuka ikonra kattintva.



Én a Git Bash-t fogom használni, amely a Gittel együtt települt a gépre.

Először ellenőrizd a telepítést a Git verziójának kiíratásával:



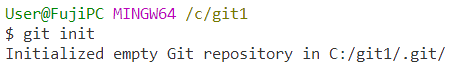
A Git első használatánál meg kell adni a felhasználónevet és az e-mail címet a következő parancsokkal:

**git config --global user.name** *felhasználóneved*

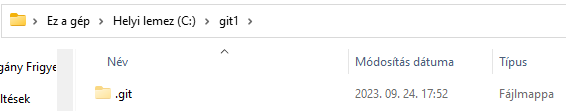
**git config --global user.email** *e-mail címed*

Végezd el ezt a saját adataiddal!

A Git használatának első lépése a repository inicializálása a **git init** paranccsal:



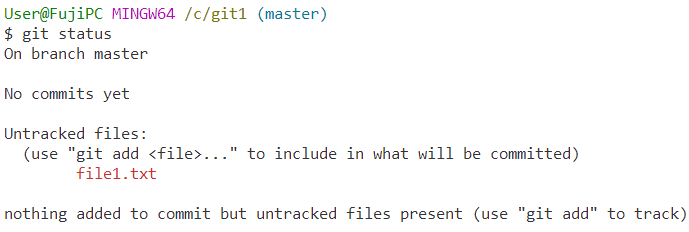
Ennek hatására egy rejtett .git nevű mappa kerül a git1 mappába:



Készíts egy üres szöveges fájlt file1.txt néven!



A fájlok és a repository állapotát a git status paranccsal jelenítheted meg:

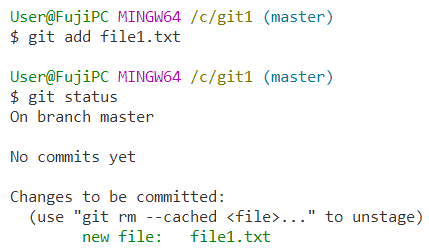


A repository-ban induláskor egy ág van, amelynek alapértelmezett neve master, és ezen vagyunk. Ezt olvashatjuk le az első sorból.

A következő sor azt jelzi, hogy még nem készítettünk egy commit-ot sem.

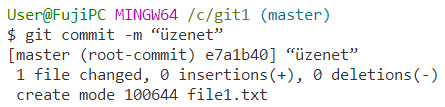
Egyelőre egy fájl, a file1.txt van a mappában, amelyet nem követünk (untracked).

Ha azt szeretnéd, hogy a fájlról pillanatfelvétel készüljön és ez bekerüljön a következő commitba, akkor add ki a **git add file1.txt** parancsot, majd nézd meg újra a státuszt:

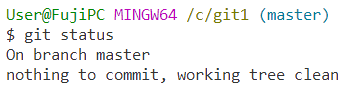


Ezzel a file1.txt bekerült a Staging Area-ba. Ez azt jelenti, hogy a következő commitban elmentjük az állapotát.

A Staging Area-ban lévő fájlok mentéséhez add ki a   
**git commit -m “*üzenet*”** parancsot!

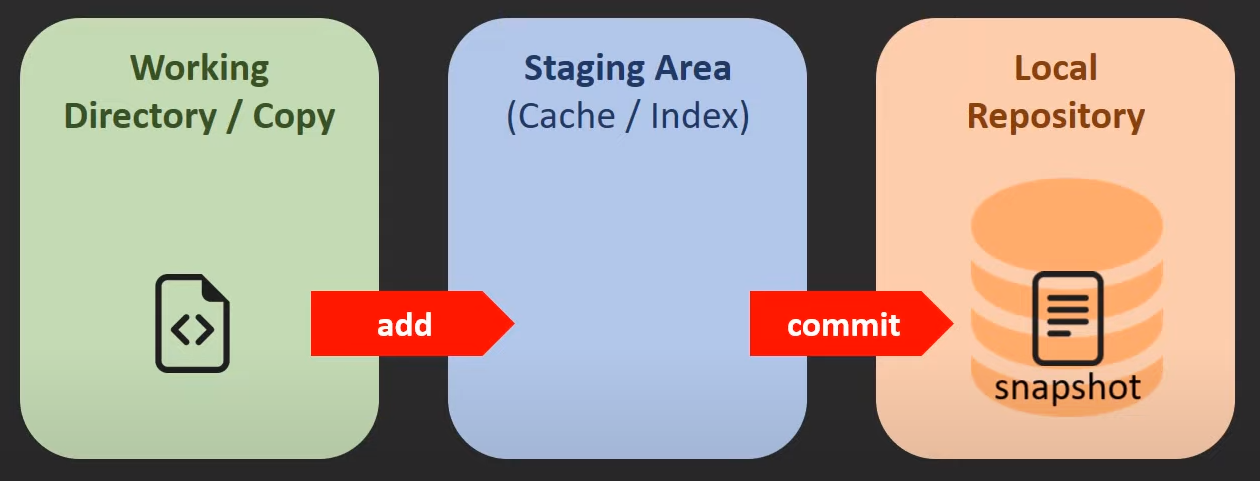


Ellenőrizd a státuszt (a korábbi parancsokat a felfelé nyíllal lehet visszahívni):



Itt azt látjuk, hogy továbbra is a master ágon vagyunk, nincs mit commitolni, és nincs semmi változás a mappában.

A következő ábra az előbbi műveleteket szemlélteti:



A Working Directory az a mappa, ahol dolgozunk (git1). Itt hoztunk létre egy új fájlt.

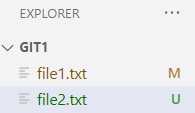
A fájlt az **add** paranccsal hozzáadtuk a Staging Area-hoz. Itt adjuk meg, hogy melyik fájlokat szeretnénk menteni.

A **commit** parancs kiadásakor a Staging Area-ban lévő fájlokat menti a Local Repository-ba (helyi tárolóba) egy commitba, egy üzenettel együtt.

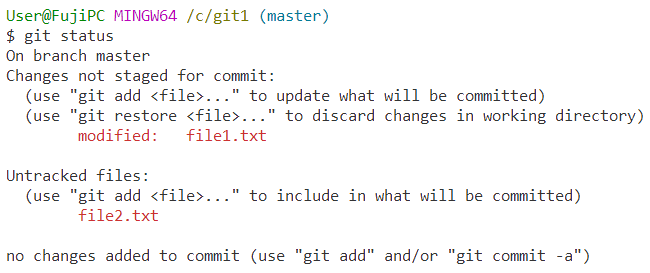
Utána a commitban szereplő fájlok állapota unchanged-re változik.

Most írj valamit a file1.txt fájlba és hozz létre egy üres file2.txt fájlt is!

Nézd meg, hogyan jelöli a Visual Studio Code a bal oldali sávban a módosított és az új fájlt!

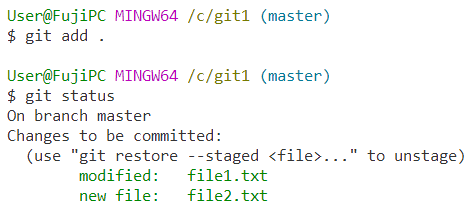


Ellenőrizd a státuszt!



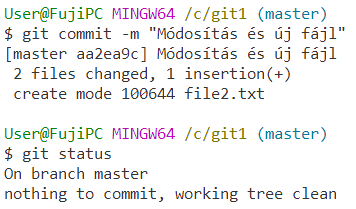
A piros sorok azt mutatják, hogy a file1.txt megváltozott és van egy új file2.txt fájl is.

Most add hozzá mindkét fájlt a Staging Area-hoz:

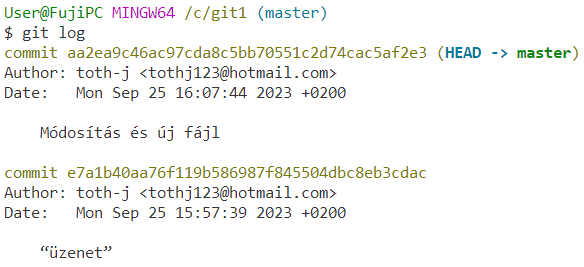


A . az aktuális mappa minden fájlját jelenti.

Commitold a változásokat a repository-ba:



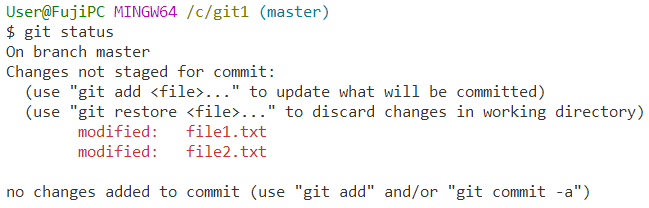
Listáztasd ki az eddigi commitokat a **git log** paranccsal:



Figyeld meg, hogy minden commitnak van egy hosszú azonosítója! A HEAD jelzi a jelenlegi állapotot.

## További parancsok

Módosítsd mindkét szöveges fájlt, majd nézd meg a státuszt:



Ha szeretnénk látni, hogy mi változott egy fájlban az utolsó commit óta, akkor a git diff parancsot használhatjuk:

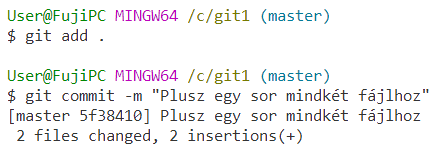


Itt az látszik, hogy a file1.txt-ben az utolsó commitban az alma sor szerepelt, és azóta hozzáírtam a körte sort is.

Ha minden fájl változásaira kíváncsiak vagyunk, akkor a HEAD-del kell összehasonlítani:



Add hozzá a fájlokat a Staging Area-hoz, majd commitold őket!

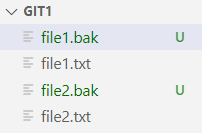


Figyeld meg, hogy az Explorer sávban eltűntek az M betűk a fájlok nevei mellől!

## Fájlok kizárása

Ha ki szeretnénk zárni fájlokat és mappákat a követésből, akkor egy .gitignore nevű fájlban kell megadnunk őket.

Készíts először másolatot mindkét szöveges fájlról bak kiterjesztéssel!

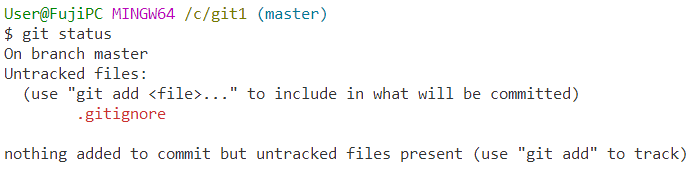


Ezután készíts egy .gitignore nevű fájlt, és add meg benne a .bak kiterjesztésű fájlokat:



A csillag azt jelzi, hogy a fájl neve bármi lehet.

Nézd meg ezután a státuszt:



Figyeld meg, hogy a .bak kiterjesztésű fájlok nem szerepelnek az untracked fájlok között sem!