Git Branch téma

A branch -ek (ágak) nagyon fontos szerepet töltenek be a git verziókezelőben. Segítségükkel párhuzamosan többen dolgozhatnak akár azonos fájlokkal is.

A branch

A branch a fejlesztés egy ágát jelenti, melyet névvel szoktak ellátni. A branch-ek lehetővé teszik, hogy a fejlesztés ne csak egy szálon történjen. Az egyidejű, párhuzamos fejlesztések egymástól elszeparálva működnek. Egy új projekt létrehozásakor automatikusan létrejön egy alapértelmezett branch, a master branch.

A projekt history commit-okból épül fel, melyek egymásra hivatkoznak. Minden egyes commit-nak legalább egy, legfeljebb kettő szülője van (merge-ölés után). Branch létrehozása során ki kell jelölni egy bázis commit-ot, mely az ág kezdetét fogja jelölni, s a branch származtatása innen történik. A branch-ek egymással párhuzamosan léteznek (természetesen más névvel is ellátható, de ez az elterjedt).

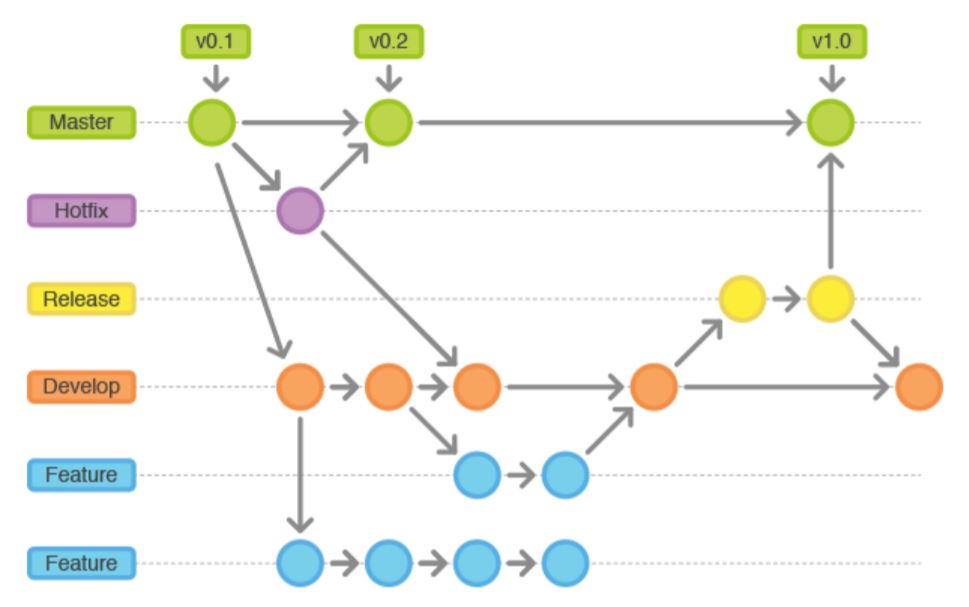
A branch tulajdonképpen arra szolgál, hogy a fejlesztést különböző részekre (pl. fejlesztési fázisokra és modulokra) lehessen osztani. Ez azért fontos, mert a fejlesztők egymástól függetlenül dolgozhatnak az egyes ágakon ugyanazon repo-n. Lehetőség van branch-ek közötti váltásra és branch-ek egybefűzésére is. Az előbbi azért fontos, mert átjárhatóságot biztosít a fejlesztés különböző részei között, az utóbbi pedig a fejlesztés párhuzamosan folyó meneteit képes egybeolvasztani, ezáltal kettő vagy több eredményből egy közös keletkezik.

Az egész Git repository-nak a gyökerét az első commit jelenti. Az összes commit és branch ebből a commit-ból, vagy ennek leszármazottjaiból ered.

Branch létrehozása

Minden tároló rendelkezik legalalább 1 branch-csel. A fejlesztés az ágakban történik.

Branch-ek közötti váltás során a Git visszakeresi a két branch legutolsó közös commit hivatkozását (bázis vagy ős branch), s attól a commit-tól kezdve a cél branch összes módosítását leírja a fájlrendszerre (útbejárás).

Az alábbi ábrán a brach típusok és elágazások is jól követhetőek:  


Az ágak létrehozásakor a master branch legtöbbször a production fázist jelöli, s ebből származtatják a release, develop és hotfixes ágazatokat.

A fő fejlesztés a develop ágon megy végbe, melyből modulonként 1-1 feature branch-et származtatnak. Az egyes modulok elkészülte után a develop ágba fűzik a kész feature branch-eket. A release branch-be a tesztelés alatt álló develop verziók kerülnek.

Ha az adott release verzió stabil, akkor a végleges master ágba kerül befűzésre. A master branch-ben felmerült hibákat a hotfix nevezetű branch-ekkel javítják, melyeket közvetlenül a master-ből származtatnak.

branch létrehozása

git branch <branch\_neve>

Új könyvtár és új filé létrehozása után.

git branch develop

git checkout develop

Ekkor még nem látszik az új branch a github-on

**git commit –all menti el a branch-ot!!!**

git add .

git commit --all -m "branch pelda updated"

git push --set-upstream origin develop

checkout

A checkout parancs a branch-ek közötti **váltásra** szolgál. Ilyenkor a Git az aktuális branch commit-jain visszafelé haladva megkeresi a legelső, cél branch-csel ugyanarra hivatkozó commit-ot, s attól fogva a cél branch commit-jainak változtatásait leírja a fájlrendszerre.

A következő példában a develop branch-ra történik váltás:

git ckeckout develop

Dokumentáció:

<https://git-scm.com/docs/git-branch>

Önálló feladatok

* Készíts egy develop branch -et, majd állj rá a VSCode -ban.
* Hajts végre módosításokat a fájlokon és commitold, pushold őket.