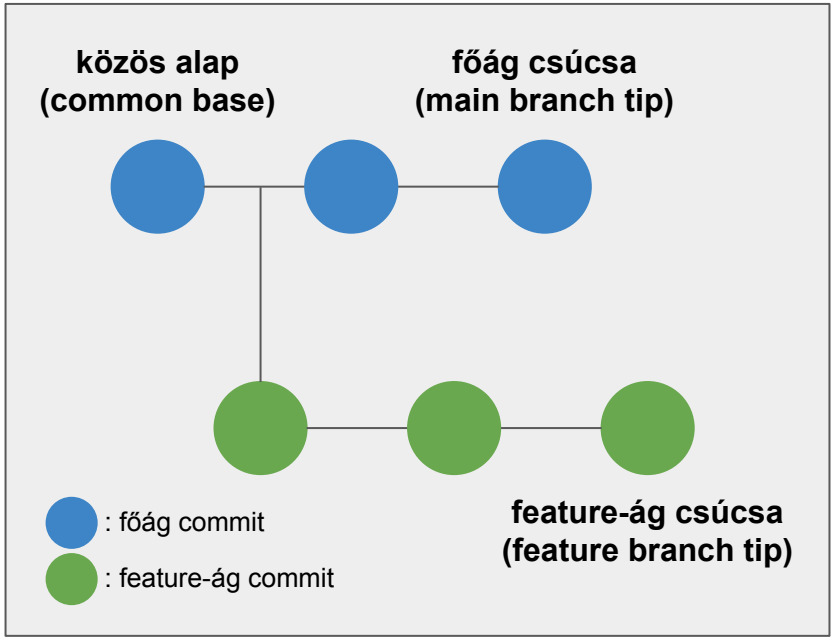
1. Merge vs. Rebase

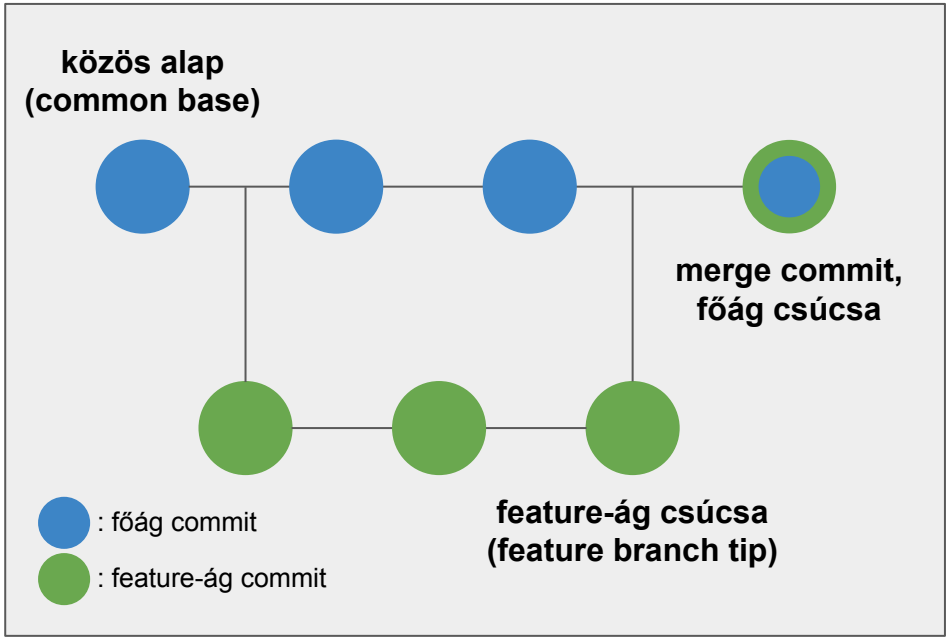
Merge (beolvasztás)

Ha két ág (branch) eltér egymástól, a git merge paranccsal olvaszthatjuk bele a másik ág változásait a munkakönyvtár állapotába. Ilyenkor tényleg összeolvasztásról van szó, hiszen mindkét ág változtatásait igyekszünk megtartani.

1. Merge előtt



2. Merge után



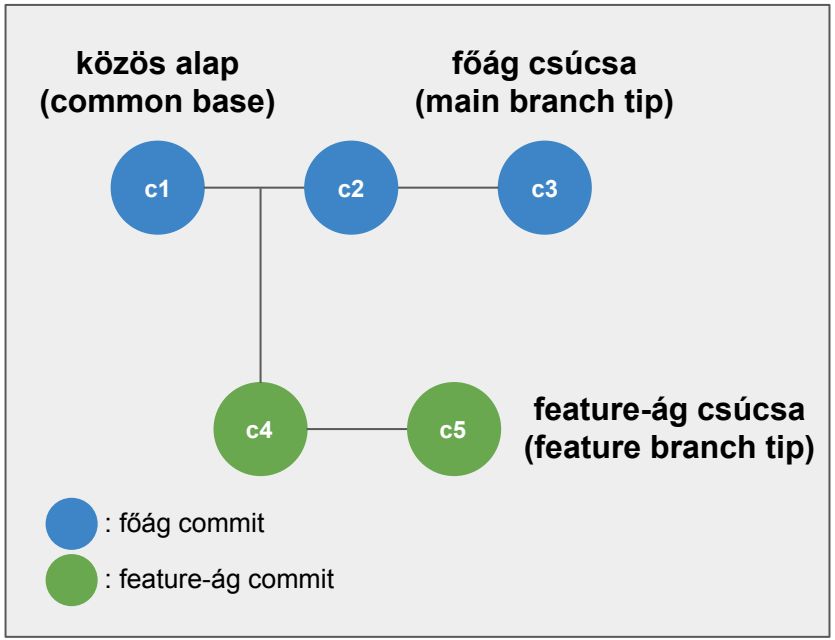
alap (base): Első commit adott ágon
csúcs (tip): Legfrissebb commit adott ágon

aliz

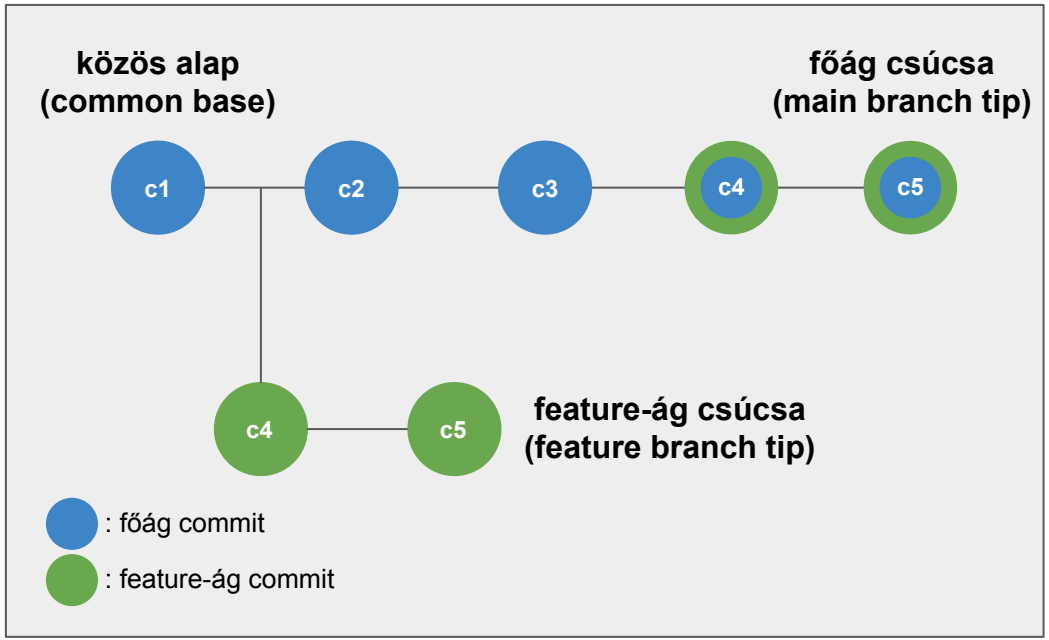
Rebase

A rebase parancs használatával egy adott ág változásait másolhatjuk át egy másik ágra, mintha azokat a változtatásokat eredetileg is azon az ágon vittük volna végbe.

1. Rebase előtt



2. Rebase után

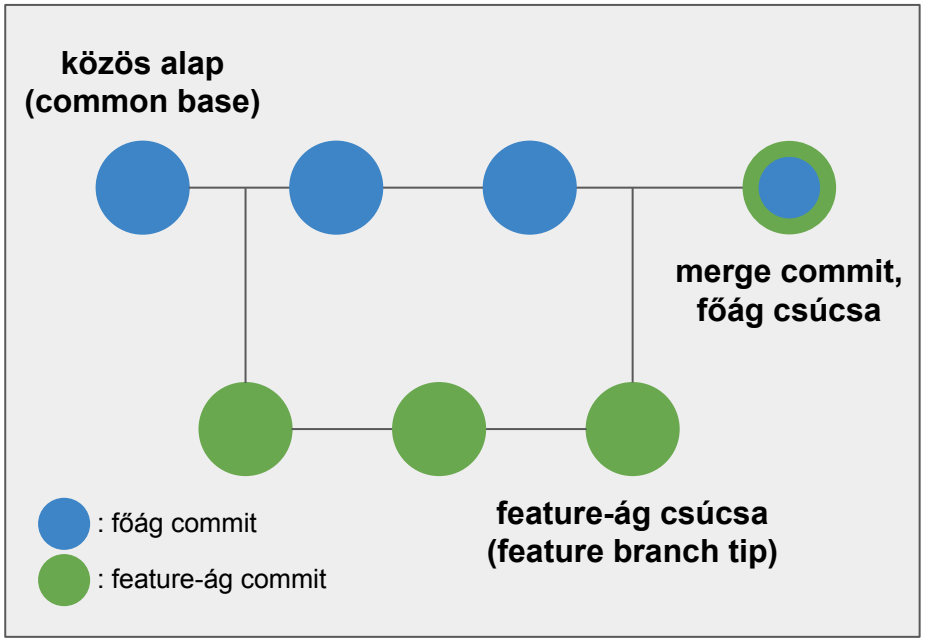


alap (base): Első commit adott ágon
csúcs (tip): Legfrissebb commit adott ágon

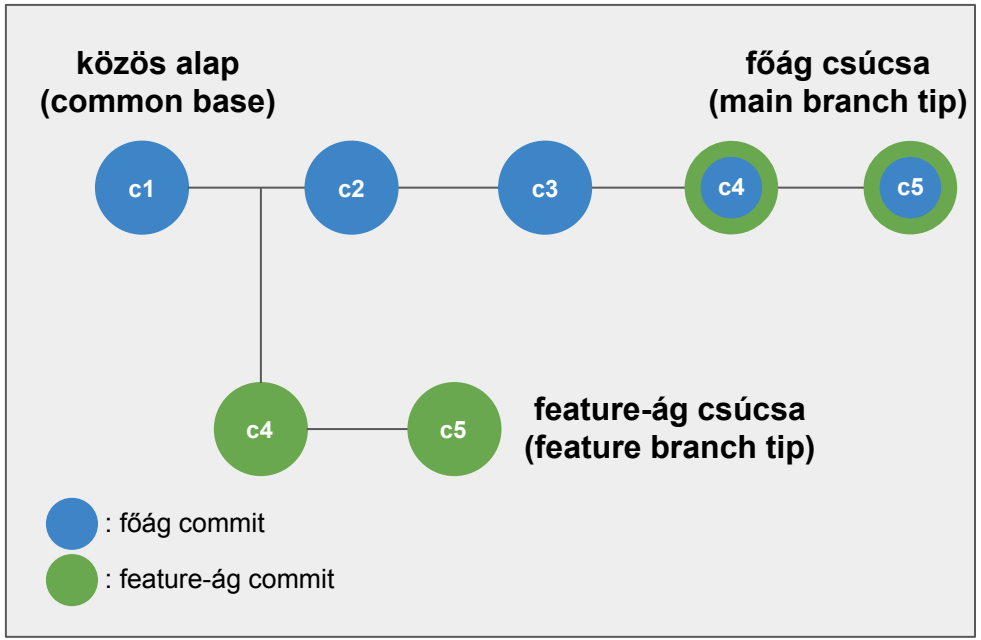
Merge vs. Rebase

A merge és rebase parancsok gyakorlatban ugyanazt a célt szolgálják, mindkét utasítás eltérő ágak változásainak kombinálását eredményezik. A fő különbség a commit-történet vizuális ábrázolásában rejlik.

2. Merge



2. Rebase

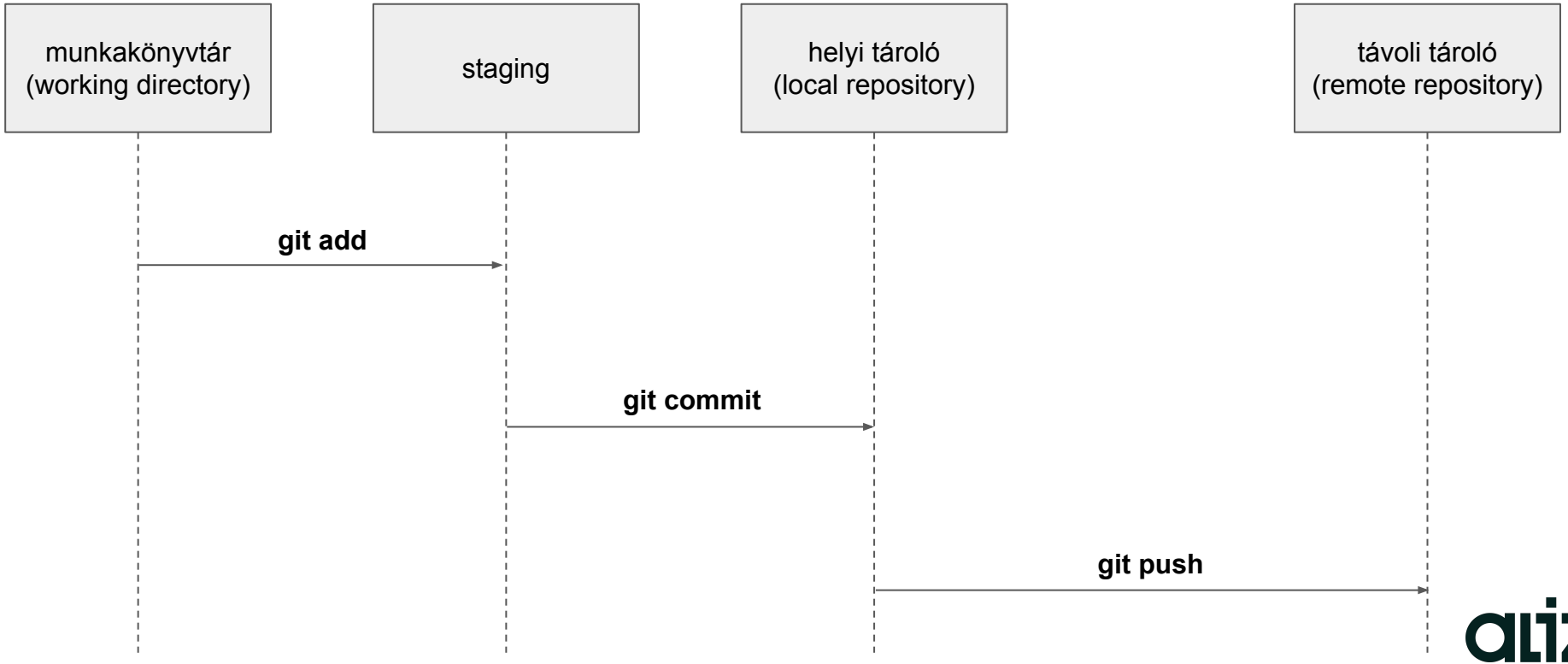


2. Add & Commit vs. Push

aliz

Hozzáadás (Add) & Commit vs. Tolás (push)

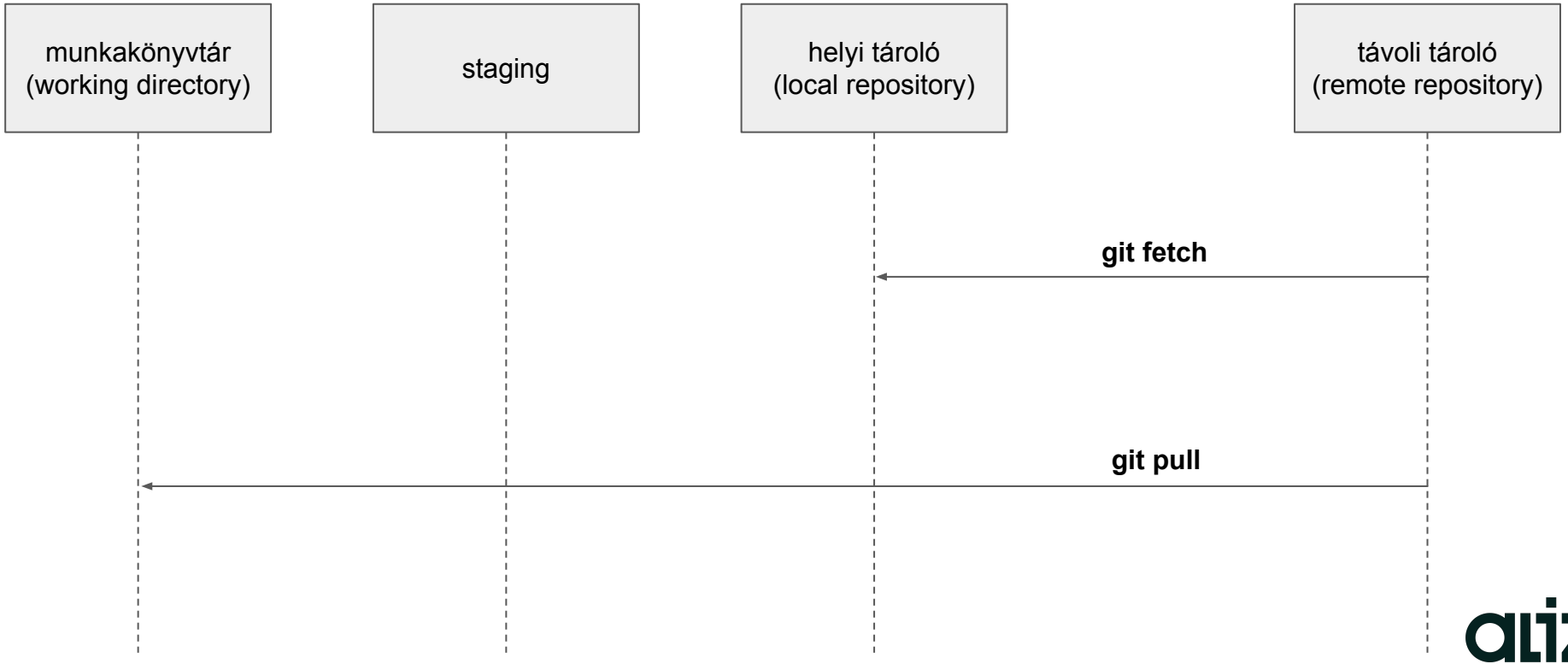
A hozzáadás és commit-olás a helyi tároló szintjén használt utasítások, mellyel a helyi változásokat elmentjük adott ágon egy commit-ba. A push paranccsal e helyi változásokat adjuk hozzá a távoli tárolóhoz.



3. Fetch vs. Pull

Elhozás (fetch) vs. Húzás (pull)

A fetch utasítással letölthetjük a távoli tároló változásait a helyi tárolónkba, azonban az utasítás ezeket nem adja hozzá automatikusan a helyi ághoz. A pull-utasítás ezzel szemben letölti és be is olvasztja a változásokat a helyi ágba.



4. Clone vs. Fork

Klónozás (clone) vs. Elágazás (fork)

Klónozással egy távoli tároló tartalmát másolhatjuk saját gépünkre helyi tárolóként, mely tárolók ezt követően egymással kapcsolatban állnak, a tárolók közötti változtatásokat szinkronizálhatjuk. Ezzel szemben az elágazással egy új, különálló tárolót hoz létre, mely kezdeti tartalma megegyezik az eredeti tárolóval.