**Git és Github alapfogalmak**

A Git használata az alapfogalmak megértésével lehetséges. A probléma ezzel "csak" annyi, hogy a magyar szakirodalom nem adaptálta még magyarul a szükséges kifejezéseket. Ezért az alapfogalmak ismertetésénél az angol megfelelőket használom, mivel nem szándékozok nyelvújító lenni.

**Repository**

Az adatbázis neve, amely a verziókezelt fájlokat tárolja. Lényegében egy mappa a fájlokkal, amiben van egy speciális rejtett .git mappa, ami a kezeléshez szükséges adatbázist tárolja.

**Server / Origin**

A szerver, amely a repository-t tárolja és kiszolgálja a fejlesztőknek. A fejlesztők a saját gépükön rendelkezhetnek privát repository-val is, azonban a többi fejlesztőnek csak az lesz elérhető, ami a szerveren van. Mivel a Git elosztott rendszerű, ezért nem szükséges a repository-t publikálni egy másik gépre, vagyis szerverre. Ezek nélkül is használható, viszont biztonsági mentés szempontjából ajánlott.

**Client**

A kliens gép, amely a szerverhez csatlakozik.

**Working Set / Working Copy**

A fejlesztő helyi gépén tárolt változata a repository-nak.

**Revision number**

A repository-ban tárolt változatokra ennek segítségével tudunk hivatkozni. Git esetén ez egy hasító függvénnyel1 képzett érték, ami egyedileg és egyértelműen beazonosítja a változatokat.

**Head**

A legutolsó revision number hivatkozó kifejezése.

**Main / branch**

A repository-t érdemes úgy felfogni, mint egy fa ágát. A main a fő ág. Ebből az ágból bármelyik ponton készíthetünk al ágat, amit branch-nek nevezünk. A branch létrehozása után külön változatként él tovább. A létrehozott branch-ek bármikor beintegrálhatóak a fő ágba. Legnagyobb előnyük, hogy kísérletezésre, új funkciók fejlesztésére biztosítanak lehetőséget anélkül, hogy új repository-t kellene létrehoznunk.

**Merge**

Két ág, branch összeolvasztásának folyamata. Az esetek többségében nem igényel manuális beavatkozást.

**Push**

A helyileg tárolt verziók áttöltése a távoli verziókezelő szerverre.

**Conflict**

Konfliktus akkor keletkezik, amikor a verziókezelő rendszer nem tud dönteni a változtatások sorsáról. Ez tipikusan akkor tud bekövetkezni, ha többen dolgoznak ugyanauon a kódon és egynél több ember azonos változtatást eszközöl. Például kitöröl egy sort. A leggyorsabb ember push művelete sikeres lesz, de a többieké nem, mivel nem lehet 2x törölni ugyan azt. Ebben az esetben konfliktus keletkezik, amit manuálisan kell nekünk megoldanunk.

Egy megoldás lehet az, hogy újra letöltjük a szerver repository-t és ismét elvégezzük a módosításainkat és utána zárunk push művelettel, vagy felülírjuk a repositoryban tárolt fájlt a saját verziónkkal.

Ennél azonban jobb megoldás, ha a conflict állapotba került fájlokat összefésüljük manuálisan. Az ütközéseket a Git megjelöli soronként. Az összefésülés végezhető bármilyen szövegszerkesztővel, de célszerű valami erre a célra kifejlesztett eszközt használni. A Visual Studio beépítetten természetesen tartalmaz ilyen eszközt.

**Tag**

Git esetén a revision number-rel való hivatkozás a fejlesztés egy bizonyos pontjára nem éppen a legkényelmesebb. Ezért a Tag-ek emberileg is olvasható hivatkozási pontokként szolgálnak. Például a szofver egy bizonyos verziójának kiadásakor létrehozunk egy tag-et, amivel aztán hivatkozhatunk az adott revision-re anélkül, hogy egy csomó hexadecimális karaktert írnánk be.

**Fork**

Egy szerveren megosztott repository helyi másolata, amin mi dolgozunk. Az általunk létrehozott változtatásokat nem feltétlen kell visszatölteni az eredeti repository-ba.

**Pull request**

A korábban fork-olt repository változásait úgy döntünk, hogy visszatöltjük az eredeti forrásba. Abban az esetben, ha nem mi vagyunk az eredeti repository tulajdonosai, vagy nincs ahhoz írási jogunk, akkor egy pull request keletkezik, amit csak az eredeti szerző tud beolvasztani a kódba. A pull request során a változtatási napló is megy a kóddal együtt, így a visszagörgethetőség megmarad.

**Patch**

A létrehozott változásokat egy fájlba írjuk ki, amely beolvasztható automatikusan a repository-ba abban az esetben, ha a repository revision number-je nem haladt tovább a patch-ben tárolt értékhez képest. Ellenkező esetben manuális beolvasztást igényel. A patch fájlba írt változtatások, majd a patch alkalmazása csak egy bejegyzéssel görgeti tovább a revision numbert, így vagy a patch csak teljes egészében távolítható el, részenként nem.