### 1. ****Mi az a Git?****

* **Git** egy verziókezelő rendszer, amelyet szoftverfejlesztők használnak a kód változásainak nyomon követésére és a csapatmunka egyszerűsítésére.
* **Linus Torvalds** hozta létre 2005-ben, és azóta az egyik legnépszerűbb eszköz a verziókezelés terén.
* Git telepítése a hivatalos weboldalról: <https://git-scm.com/>
* Célja a csapatmunka támogatása és a szoftverfejlesztés hatékonyságának növelése.

### 2. ****Git Alapfogalmak****

* **Repository (repo)**: A projekt teljes története, beleértve a kódot, a verziókat és a kapcsolódó fájlokat.
* **Commit**: Egy állapot mentése a projektben. Minden commit tartalmaz egy üzenetet, amely leírja a változtatásokat.
* **Branch**: Egy fejlesztési ág, amely lehetővé teszi a párhuzamos munkát a fő kód alaptól elkülönítve.
* **Merge**: Az ágak összekapcsolása, hogy a változtatások visszakerüljenek a fő ágba.
* **Remote**: Egy távoli repository, ahol a projekt tárolása történik (például GitHub, GitLab).

### 3. ****Git Alap Parancsok****

* **git init**: Új Git repository létrehozása.
* **git clone [url]**: Egy meglévő repository letöltése.
* **git status**: Az aktuális repository állapotának megtekintése.
* **git add [file]**: Fájl(ok) hozzáadása a commit-hoz.
* **git commit -m "üzenet"**: A változtatások mentése egy commit-ban.
* **git push**: A helyi változtatások feltöltése a távoli repository-ba.
* **git pull**: A távoli repository frissítése helyi másolatra.

#### 2. ****Git Alapvető Szolgáltatásai és Parancsa****

**Repository kezelése**

* git init: Új repository létrehozása.
* git clone [url]: Egy meglévő repository letöltése.

**Fájlok követése és commit-ok létrehozása**

* git add [file]: Fájl hozzáadása a commit-hoz.
* git commit -m "Commit üzenet": A változások mentése egy commit-ban.
* git status: Az aktuális repository állapotának megtekintése.

**Ágak (branches) kezelése**

* git branch: A meglévő ágak listázása.
* git branch [branch-név]: Új ág létrehozása.
* git checkout [branch-név]: Átváltás egy másik ágra.
* git merge [branch-név]: Egy ág összekapcsolása a fő ággal.

#### 3. ****Távoli (remote) Repository-k használata a GitHub-on****

**Távoli repository-k parancsai**

* git push: A helyi változtatások feltöltése a távoli repository-ba.
* git pull: A távoli repository frissítése a helyi másolatra.
* git clone [url]: Távoli repository letöltése helyi gépre.

### 4. ****Git Használatának Előnyei****

* **Visszakereshetőség**: Könnyen vissza lehet térni egy korábbi verzióhoz.
* **Csapatmunka támogatása**: Több fejlesztő egyszerre dolgozhat a projekten.
* **Kísérletezés lehetősége**: Új ötleteket próbálhatunk ki ágak használatával, anélkül, hogy a fő kódot módosítanánk.

### 5. ****Git Remote Repository-k****

* **GitHub**, **GitLab**, és **Bitbucket** népszerű platformok, ahol a Git repository-k tárolhatók és megoszthatók.
* Lehetővé teszik a csapatok közötti együttműködést és a verziók követését.

### 6. ****Alapvető Git Workflow****

1. **Készíts egy repository-t:** git init vagy git clone.
2. **Dolgozz a projekteden:** módosíts fájlokat.
3. **Add a változtatásokat a commit-hoz:** git add [file].
4. **Készíts egy commit-ot:** git commit -m "üzenet".
5. **Push-olj a távoli repository-ba:** git push.
6. **Frissítsd a helyi repository-t:** git pull.

#### 7. ****Csapatmunka és Online Eszközök****

**Csoportmunka támogatása GitHub-on**

* A GitHub lehetővé teszi a több felhasználós együttműködést és a kód módosításainak nyomon követését.
* Pull request-ek használata a változtatások jóváhagyásához.

**Slack használata a csapatmunka során**

* A Slack egy kommunikációs platform, amely támogatja a csapatmunkát és a projektek kezelését.
* Célja, hogy gyors és hatékony kommunikációt biztosítson a fejlesztőcsapatok számára.