**Cómo funciona GIT**

La construcción de software es un proceso largo. Se hacen muchas versiones, se integran personas a trabajar en el proyecto, etc.

Para registrar la vida de un proyecto se lleva adelante un VCS o Sistema de Control de Versiones. GIT es el más popular de estos sistemas.

Terminología

**Repositorio:** es un proyecto seguido por GIT

**Commit:** cada uno de los cambios efectuados en el historial de GIT efectuado por cada uno de los desarrolladores.

**Ramas**: bifurcaciones o nuevos caminos que toma el proyecto. La rama Master es la rama principal, a partir de ahí se saca una rama separada en forma aislada (un ambiente separado) donde se trabaja y si todo va bien, luego se integra a la rama master

C**lon**: es una copia exacta del repositorio. Cuando un programador se integra al equipo de trabajo lo primero que tiene que hacer es clonar el proyecto a su equipo local.

**Fork:** es un proyecto completamente diferente que se crea a partir de otro.

Características

**Distribuido**: cada programador tiene una copia, tiene un backup del proyecto, no necesita conectarse a internet

**Ramas y Fusiones**: las ramas no comprometen a la rama principal. Luego se integran (fusión o merge)

**Integridad de Datos**: la seguridad es total.

Momentos Básicos

1. Se crea un repositorio. Git init (repositorio de cero) o git clone (si el proyecto ya existe).
2. Una vez que se tiene el repositorio local, se manda los cambios a través de los commits. Cada commit debería representar una funcionalidad específica.
3. Los cambios no van al repositorio sino a una Staging Area (con el comando git add).
4. Una vez en el Stagin Area, mando los cambios con el git commit.

Trabajo en Equipo

1. Lo normal es tener una rama Master (la que no se toca, la inmaculada) y una rama Dev.
2. Las ramas de Dev salen pero deben volver. Cuando vuelven el líder del proyecto revisan todos los conflictos (o con sistemas avanzados) y una vez chequeado, se hace la integración (fusión o merge) a Master. Es el momento de mandar a producción.

Herramientas

La herramienta más importante para trabajar con Git es la terminal o la línea de comandos.

git init: crear un proyecto

git add <file>: para pasar los archivos del working directory al staging area

git status: en qué estados están tus archivos

git commit: para pasar del staging area al repositorio (snapshot o foto)

git push: para subirlo a un repositorio remoto

git pull: para multiples desarrolladores y te traen los cambios de los otros

git clone: copia a tu local

Repositorios en la Nube

Son las ramas remotas llamadas origin: Github, Bitbucket, Gitlab. Hay repositorios públicos y también privados (pagos) para que nadie vea los proyectos.

IDEs y Editories

Git está integrado en VS Code por ejemplo. Git es la base de esto. Github es una de las empresas que dan repositorios