Taller de Git

¿Qué es esto del control de versiones?

Sirve para poder manejar de manera eficiente proyectos colaborativos de código en donde se comparten versiones de código de manera eficiente y sin conflictos entre todos los miembros del equipo.

Git: Conceptos

Repositorio: conjunto de archivos y directorios que hacen al proyecto en el cual estamos trabajando.

- Local: es el versionado de los archivos que tenemos en la PC
- Remoto: idem, pero en la nube, hosteado en uno de los servidores (GitHub, GitLab, Bitbucket). Puede ser público o privado.

¿Cómo empezar?

Instalar Git

Linux

```
sudo apt-get install git
```

Windows

https://git-scm.com/download/win

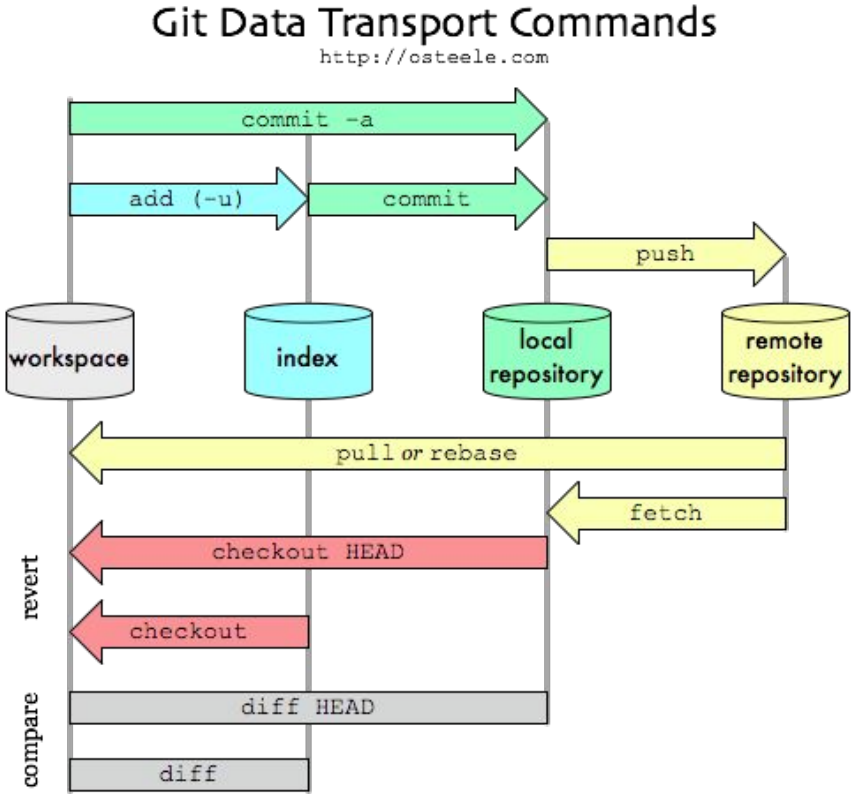
Mac OS X (con Homebrew)

```
brew install git
```

Configurar Credenciales

```
$ git config (--global) user.name “John Doe”

$ git config (--global) user.email “johndoe@email.com”
```

Workspace - Index - Local & Remote Repositories



Crear Repositorio - Primer Commit

1. Crear el repo en el sitio
2. Escribir los siguientes comandos en la consola

```
$ git init

$ echo "# taller-git" >> README.md

$ git add README.md

$ git commit -m “Initial Commit.”

$ git remote add origin https://github.com/fefee33/taller-git

$ git push -u origin
```

¿Qué pasó?

¿Qué pasó?

```
$ git log
```

¿Dónde estoy?

```
$ git status
```

```
$ gitk -all
```

Conceptos

Commit: snapshot de una versión del código en un momento dado. Tiene un id único entre todos los commits.

Branch: serie de commits que marcan una línea independiente de desarrollo.

Gitk está disponible solo en Linux/Mac. En Windows pueden usar la interfaz GUI de Git al descargarlo

Clonar Repositorio

```
$ git clone <repository-url> <directorio>
```

Hace una copia de todo el repositorio remoto en un directorio local. Trae todo, commits, branches, log, todo.

¿Y ahora qué hacemos? - I

¡Programamos!

Agregar cambios:

```
$ git add <archivo1> .. <archivoN>

$ git commit -m “Mensaje descriptivo de commit.”

$ git push origin HEAD
```

¿Y ahora qué hacemos? - II

¡Programamos aún más!

Traer cambios:

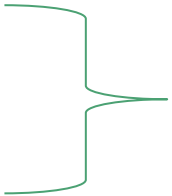
```
$ git fetch origin

$ git pull origin

$ git log

$ git diff <old_commit> <new_commit>
```

Resolución de Conflictos

```
$ git fetch origin

$ git pull origin

$ git add <files>

$ git commit -m “Mensaje descriptivo de commit”
```

Branches

```
$ git branch branchname
$ git checkout branchname
```



```
$ git checkout -b branchname
```

```
$ git status / $ git branch
```

Una vez terminado todo sobre la branch, para unir con la branch principal:

```
$ git merge <branchname>
```

Parado sobre la branch a la cual quiero añadir los commits.

¿Problemas? Don’t Panic!

Se pusheó algo que no sirve.

```
$ git revert <commit>
$ git reset --hard HEAD~N
```

Perdí un commit. Borré algo que no debía. Hice mal un merge.

```
$ git log

$ git reflog
```

Más info y tutoriales

Documentación Oficial de Git

https://git-scm.com/doc

GitHub Git Tutorial (básico)

https://try.github.io/levels/1/challenges/1

Otro tutorial (desde básico hasta avanzado)

http://learngitbranching.js.org/