Clase 3

Alumna: Laura Loreiro

22/03/2021

Playground

GIT

¿Qué vamos a ver en esta clase?

"Es muy útil para compartir proyectos con distintas personas, tener varias versiones de un mismo proyecto y poder guardar proyectos en la nube. Es por esto que es una de las más requeridas por el mercado a la hora de trabajar con control de versiones y de forma colaborativa"

Introducción a GIT

Antes se usaban Pendrive, para compartir archivos.

En el mundo web o mobile se necesita compartir y guardar (backup) archivos de trabajo para poder trabajar, antes se creaban copias versión uno, versión dos, etc,

Para realizar cambios constantes y varias veces al día, por eso existe la tecnología GIT, este hace un backup de los proyectos, y a la vez compartir con colaboradores.

Git es un software de control de versiones

Mantiene la eficientemente las actualizaciones sobre el código fuente, lleva registro de los cambios de los archivos, y coordinar el trabajo que varias personas realizan sobre los mismos

Se puede tener historial de versiones de un mismo archivo, pudiendo acceder a las distintas versiones, también se puede hacer seguimiento de quien realizó qué cambio

Instalación de GIT

Creando nuestro primer repositorio

Git init genera un repositorio local en la carpeta que tenemos los archivos.

Un repositorio es un almacen de archivos, que se van guardando en pequeños paquetes, "**commits**" (facilitan el seguimiento de los cambios).

Al repositorio hay que avisar quien es el usuario que lo está usando, para que cree el historial y seguimiento, para ello, **git config user.name "mi-usuario"**, para verificar el usuario **git config user.name**; también el correo electrónico de github **git config user.email "mail"**

Para hacer esto global **git config --global user.name "mi-usuario"** y **git config --global user.email "mail"**

Agregando archivos al repositorio

Cuando tenemos un repositorio local ya inicializado, el mismo es un almacén de archivos que está vacío, estos archivos que agregamos se agregan dentro de commits, que tienen una marca de tiempo, y firma del autor

Git add

Git status (te dirá el estado de los archivos respecto al repositorio)

Si hay archivos en la carpeta y no fueron agregados, git avisará que hay archivos sin seguimiento

Git add . (agrega todos los archivos al repositorio)

Cuando un archivo ya existente sufre un cambio, git lo toma como si fuera un nuevo archivo sin seguimiento, por el control de versiones.

Confirmando Archivos

Confirmar que los archivos agregados al repostirio, será un punto en la línea del tiempo al que podemos volver si así lo necesitamos.

Commits, generan un punto en la línea del tiempo, con datos del autor, fecha, y mensaje de la modificación y a su vez volver si es necesario

Git commit -m "mensaje"

Dentro del commit ingresan los archivos en los que hayas hecho previamente add

Git log

Genera un historial de los **commits**

Clase sincrónica

Nombre de fantasía de la clase "Fragmentados"

¿Qué es GIT?

Es una herramienta para ordenar en forma cronológica los archivos.

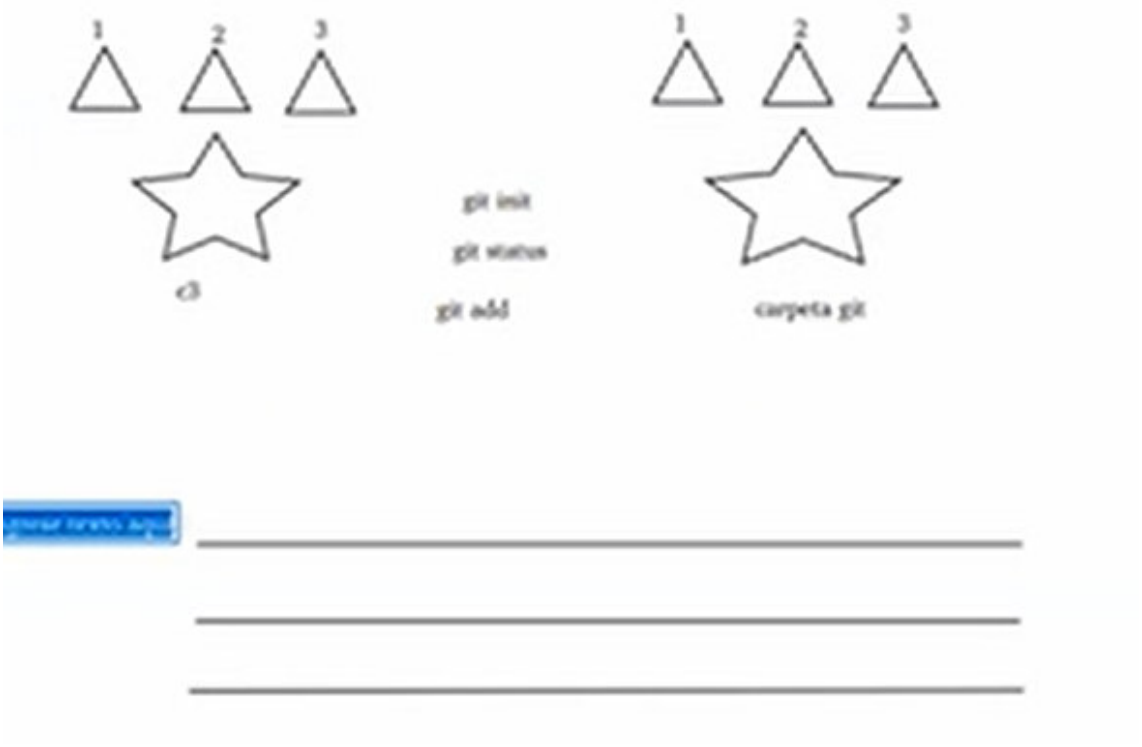
Git init, crea una carpeta git que es la que hace seguimiento a los archivos de esa carpeta

Configuramos nuestro nombre y mail **git config user.name** y **git config user.email .** Para hacer esto global **git config --global user.name "mi-usuario"** y **git config --global user.email "mail"**

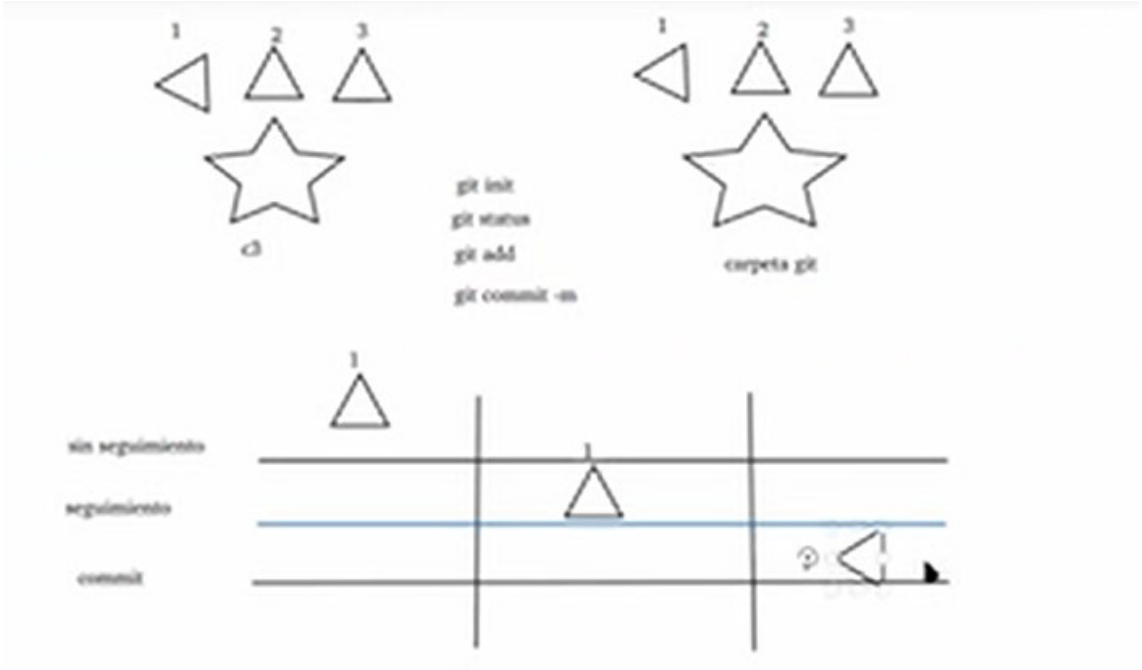
Si quiero generar otro usuario, tengo que poner **git config user.name** y **git config user.email,** con el nuevo usuario, lo pisa.

Git status, nos dice los archivos que tienen y no tienen seguimiento

Git add . esto hace que todos los archivos que no tienen seguimiento, tengan seguimiento

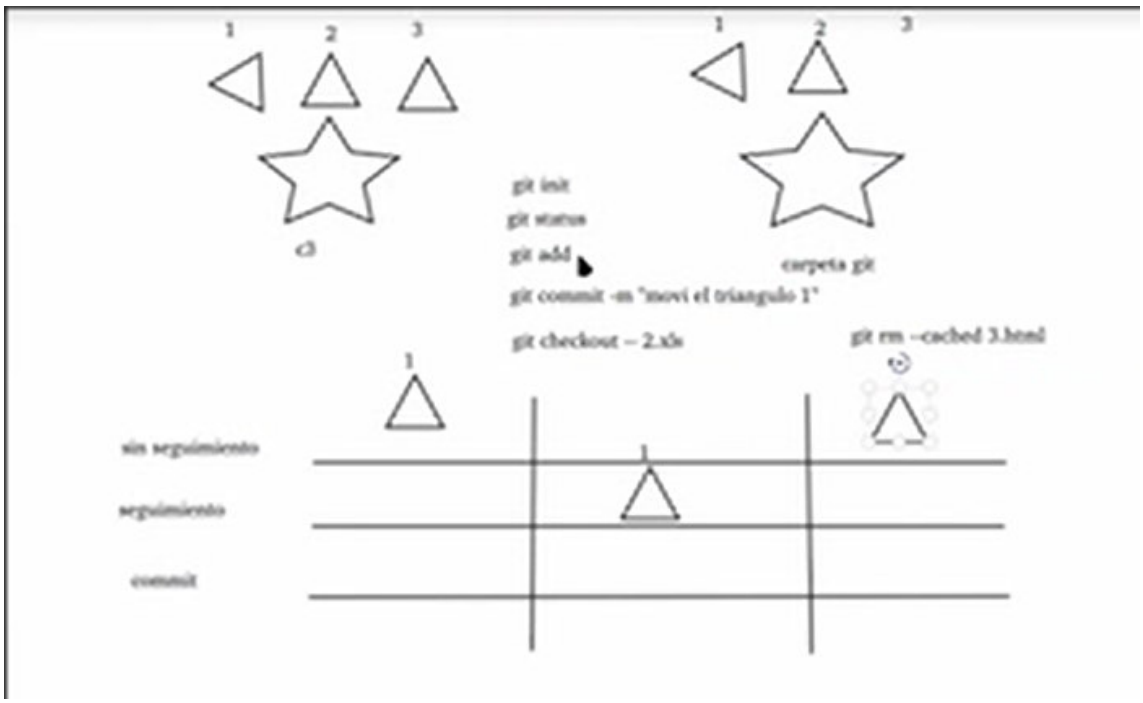


Cuando hago **git commit -m "mensaje"** se sincronizan los cambios que efectuamos

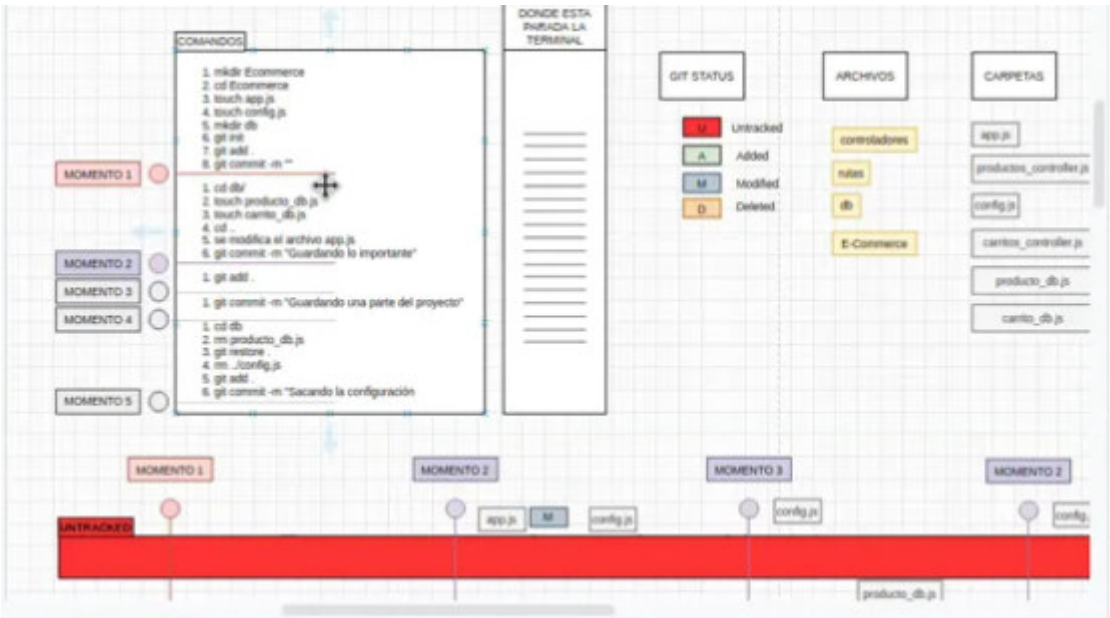


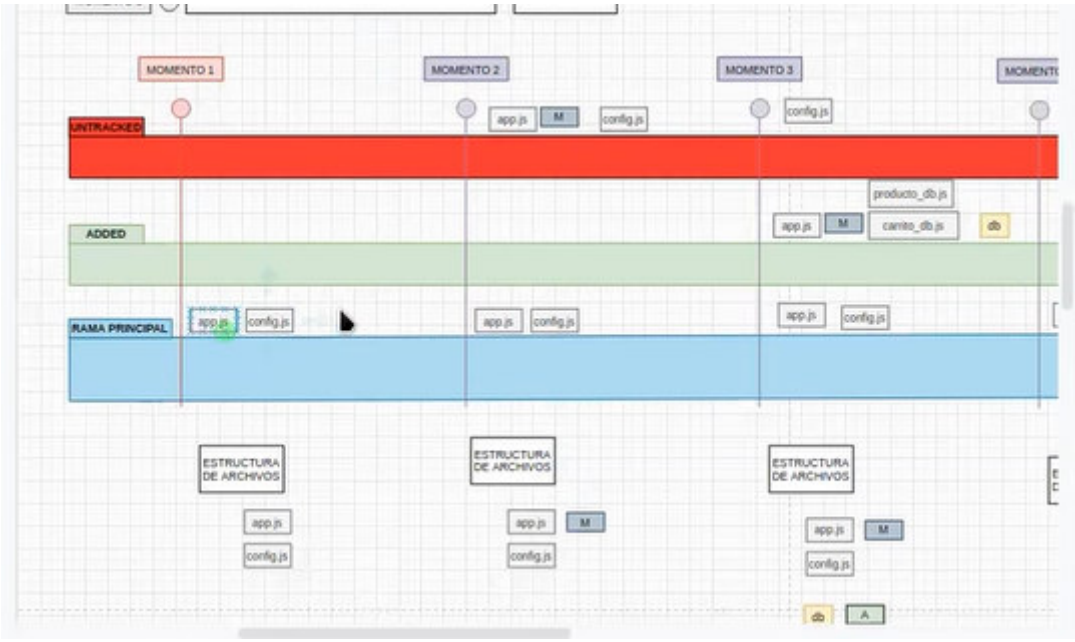
Hay dos formas de crear un repositorio, con **git init** o con **git clone**, cuándo se hace un clone se genera un nuevo Branch, en el repositorio local

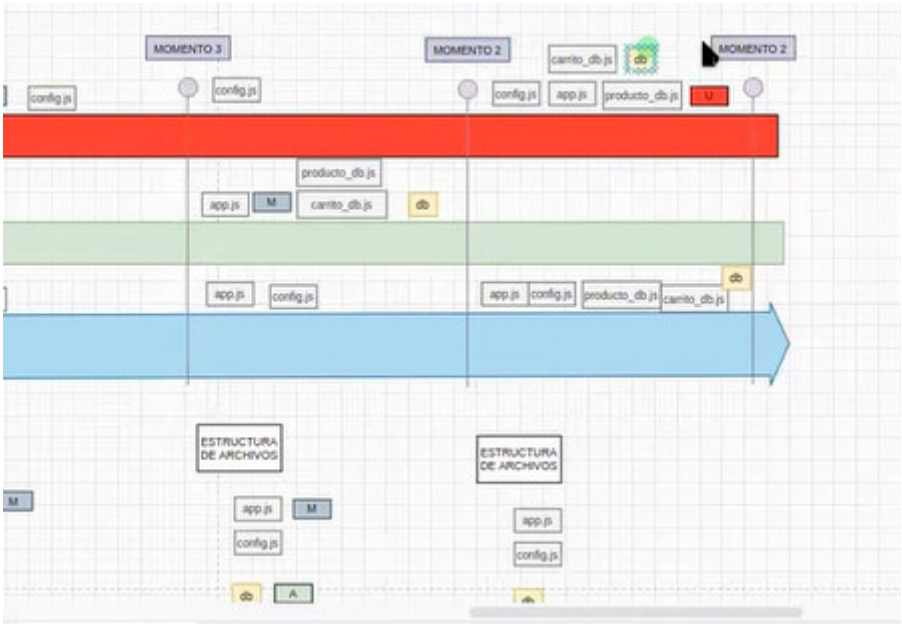
Git rm --cached nombre_archivo, deja sin seguimiento al archivo, es la inversa del **add**



Actividad




MOMENTO 1
MOMENTO 2
MOMENTO 3
UNTRACKED
app.js
M
config.js
producto_db.js
carrito_db.js
db
ADDED
RAMA PRINCIPAL
ESTRUCTURA DE ARCHIVOS
A


MOMENTO 3
MOMENTO 2
MOMENTO 2
carrito_db.js
db
config.js
app.js
producto_db.js
U
M
A
ESTRUCTURA DE ARCHIVOS