**II\_4**

**GIT:** es un software de control de versiones que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo. De esta forma, podemos recuperar y tener acceso a versiones específicas cuando queramos.

GIT te permite tener un backup en la nube de tu proyecto. También te permite revertir archivos y proyectos enteros a un estado anterior, comparar cambios a lo largo del tiempo, ver quién modificó por última vez algo que puede estar causando un problema, quién introdujo un error y cuándo, y mucho más.

 Al ser un sistema de control de versiones, GIT te permite evertir archivos y proyectos enteros a un estado anterior, comparar cambios a lo largo del tiempo, ver quién modificó por última vez algo que puede estar causando un problema, quién introdujo un error y cuándo, y mucho más.

BACKUP DE ARCHIVOS

COMPARTIR ARCHIVOS

**Software de control de versiones**

**Propósito llevar el registro del cambio de los archivos**

**Se puede hacer el seguimiento de los cambios**

**Git --version / para saber la versión que tenemos instalados**

git --version

**Corrección**  Para verificar que la instalación se haya realizado correctamente, abrí una terminal y correr el comando git --version.

**Palabras clave**

A modo de resumen, repasemos las palabras claves de este video:

* **Repositorio local:** es el que tiene todos los archivos (que hayas guardado en él) en la computadora.
* **Commits:** son los **paquetes** que nos van a permitir ir haciendo un seguimiento de los cambios que vamos realizando, dado que cada uno de ellos tiene una timestamp, o fecha de creación, y un autor. Los commits van a ser nuestro historial de cambios que se fueron haciendo en el proyecto.

Estos son los comandos que vimos en el video:

git init *// crea el repositorio local en la carpeta de los archivos*

git config user.name "nombre de usuario" *// agrega nuestra identidad*

git config user.email "nombreUsuario@email.com" *// agrega nuestro email*

para configurarlo siempre con mis datos

**git config --global user.name luciayanicelli**

CONECTARSE CON EL REPOSITORIO REMOTO DE GITHUB PREVIAMENTE COPIADO

git remote add origin https:*//github.com/DH/RepoRemoto // apunta al repositorio remoto*

*git remote -v /// para ver si se vinculó con el repositorio remoto*

*CLASE\_4\_GIT HUB*

## Palabras clave

A modo de resumen, repasamos las palabras y conceptos clave del video sobre GitHub:

* GitHub es un lugar en la nube.
* **Repositorio:** es el lugar en donde se irán almacenando los archivos de nuestro proyecto y a través del cual podremos hacer seguimiento de los mismos.
* **Repositorios remotos:** viven en la nube, es decir, en GitHub.
* **Repositorios locales:** viven en nuestra computadora.

git add . *// agrega todos los archivos o sólo uno git add funciones.js*

git commit -m "mensaje" *// comitea los cambios hechos*

git push origin master *// envia los cambios al repositorio remoto (cambiar master por main para repositorios creados a partir de octubre 2020)*

git status *// seguimiento del estado de los archivos*

git clone “url” *// descarga exactamente el repositorio remoto en mi compu, sincronizando este repositorio de mi compu con el de la nube*

git pull origin master *// actualiza el repositorio desde la nube si es que previamente ya hice un git clone en algún momento y por lo tanto tengo el proyecto en mi compu*

git log *// permite ver el historial de los commits realizados*

*GIT HUB*

*https://github.com/lmyanicelli/*

*Proceso: crear, add, commit, push*

**CONFLICTOS al querer subir**

<<<<HEAD

Aquí aparecerá todo lo que está en mi compu

====

Aquí aparecerán los cambios que están en la nube con los que estamos teniendo conflictos

>>>>>andnbasfbdhvbwdv

**Conectar repositorio local con repositorio remoto**

Para que nuestro repositorio local sepa a dónde queremos subir nuestros archivos, hace falta especificarlo.

⇒ Tener creado previamente un repositorio en GitHub.

⇒ Ir a la ubicación del mismo y copiar la URL.

⇒ Escribir el comando git remote add origin .

⇒ Pegar la URL después de la palabra origin (dejando un espacio de por medio) y

presionar Enter.

⇒ Para verificar que el paso anterior se ejecutó correctamente, correr el comando

git remote -v . Deberíamos ver en la terminal la palabra origin seguida de la

URL.

* Para sacar a un archivo para que deje de hacerle el seguimiento (sería como el inverso del add) **git rm -- cached 3.html**

**Otro comando:** git checkout dev (sirve para cambiar de rama, de branch)

git push -u origin dev 🡪 para subir una nueva branch … -u hace referencia a la rama

git diff 🡪 muestra las diferencias entre los archivos que hay entre el repositorio de la nube y nuestro git local

git merge nombreBranch 🡪 ubicado en un branch lo q hago con merge es unificar el branch en el q estoy con el nombreBranch

[1](https://playground.digitalhouse.com/)

**[Un commit tiene...](https://playground.digitalhouse.com/)**

fecha, autor y mensaje

**Corrección**  ¡Muy bien! Cada vez que subimos archivos (nuevos o modificados) a un repositorio remoto, se suben en forma de un pequeño paquete de modificaciones de archivos. Cada paquete tiene una fecha de creación (timestamp) y un autor.

[2](https://playground.digitalhouse.com/)

**[Con Git...](https://playground.digitalhouse.com/)**

podemos revertir cambios y volver a un estado anterior.

podemos tener un orden cronológico del proyecto.

[3](https://playground.digitalhouse.com/)

**[Seleccionar el orden correcto para persistir los cambios de productos.js.](https://playground.digitalhouse.com/)**

git add productos.js —> se modifica el archivo —> git add productos.js —>git commit -m "Carrito agregado"

**Corrección**  ¡Muy bien! Luego de modificar un archivo, si queremos guardar las modificaciones, tenemos que decirle a Git que las incluya, utilizando el comando git add.

[1](https://playground.digitalhouse.com/)

**[Git y GitHub:](https://playground.digitalhouse.com/)**

Git es un sistema de versionado y GitHub un sistema en la nube que provee de ese servicio.

**Corrección**   ¡Muy bien! Si bien tienen un nombre similar, Git es la tecnología que mantiene el sistema de versionado, mientras que GitHub es un proveedor del servicio. Hay muchos más, como GitLab o Bitbucket, los invitamos a visitar sus páginas webs y ver qué ofrecen.

[2](https://playground.digitalhouse.com/)

**[Un repositorio en GitHub es…](https://playground.digitalhouse.com/)**

un proyecto en la nube utilizando Git como sistema de versionado

**Corrección**  Podés almacenar archivos y proyectos ¡DE FORMA GRATUITA!

[3](https://playground.digitalhouse.com/)

**[¿Cuál es la diferencia entre un repositorio local o remoto?](https://playground.digitalhouse.com/)**

Un repositorio local es el que se encuentra en mi computadora y el remoto es el de GitHub.

**Corrección**   ¡Muy bien! Recordá que es necesario crear un vínculo entre ambos para poder mantener actualizados los archivos locales que están conectados a ese repositorio en la nube.

[1](https://playground.digitalhouse.com/)

**[¿Con qué comando envío los archivos de mi repositorio a GitHub?](https://playground.digitalhouse.com/)**

git push

[2](https://playground.digitalhouse.com/)

#### [En Git, ¿qué es una rama?](https://playground.digitalhouse.com/)

Es una línea paralela para agregar funcionalidades sin afectar a la rama principal.

**Corrección**  ¡Muy bien! Es una muy buena práctica crear una rama (branch) para trabajar en una nueva funcionalidad así no afectamos el proyecto hasta que esta funcionalidad esté finalizada.

[1](https://playground.digitalhouse.com/)

**[¿Cómo descargás por primera vez un repositorio?](https://playground.digitalhouse.com/)**

git clone "url del proyecto"

[2](https://playground.digitalhouse.com/)

**[¿Cómo actualizás el código en tu repositorio local?](https://playground.digitalhouse.com/)**

git pull origin master