INTRODUCCIÓN A GIT Y GITHUB

¿Qué es GitHub?

* Es un servicio de alojamiento que ofrece a los desarrolladores repositorios de software utilizando el sistema de control de versiones, **Git**.
* CARACTERÍSTICAS:
* GitHub permite alojar proyectos en repositorios de forma gratuita
* Tiene repositorios privados con la versión de pago.
* Permite compartir los proyectos de una forma super fácil.
* Te permite colaborar para mejorar los proyectos de otros y que otros colaboren con los tuyos.
* Ayuda a reducir los errores humanos, a tener mejor mantenimiento de distintos entornos y a detectar fallos de una forma más rápida y eficiente.
* Opción perfecta para trabajar en equipo sobre un mismo proyecto.
* Todas las ventajas de Git + otras herramientas que ayudan a tener un mejor control de nuestros proyectos.
* HERRAMIENTAS DE GITHUB:
* Wiki: Ayuda en el mantenimiento

SISTEMAS DE CONTROL DE VERSIONES

**¿Qué es un sistema control de versiones (VCS)?**

Es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo.

Tipos de sistema control de versiones:

* Local (Local Computer): Alojado en nuestro equipo. Si ocurre algún desastre vamos a perder nuestro trabajo, además de que Ctrl + Z tiene cierta limitación. Si nuestro se destruye ya no hay forma de hacer ningún Ctrl + Z (no podemos recuperar el trabajo)
* Centralizado: No depende del equipo desde donde se trabaja sino del del super servidor donde se almacena la información. Podemos acceder a esa información desde cualquier equipo que se conecte al servidor, pero solo guarda los cambios en un solo lugar(server). Si se destruye el servidor perdemos todos los datos y todos los cambios.
* Distribuido: FAVORITO - Cada uno que participa en el repositorio, tiene su propio repositorio en su equipo local y otro en el servidor, para poder trabajar en tu propio repositorio sin afectar al de los otros y luego mezclarlo en el del servidor. Si se nos daña el ordenador o el servidor siempre vamos a poder hacer una copia del repositorio tanto del equipo local o del servidor del repositorio.

GIT es un VCS distribuido.

**Los beneficios de utilizar Git**

Git nace de la necesidad de que linux estuviera en un sistema de control de versiones, y para ello, en 2002, se asociaron con Beatkeeper para utilizar su VCS distribuido para almacenar el núcleo de Linux. En el 2005 las relaciones se deterioraron y les quitaron la licencia gratuita que tenían para almacenar Linux y les empezaron a cobrar, pero Linux no tenía dinero, así que Linus Torvalds creó su propio VCS distribuido (GIT).

Beneficios:

* Velocidad: Puedes trabajar fluidamente desde tu ordenador.
* Diseño sencillo: El código es robusto con las herramientas necesarias.
* Fuerte apoyo en el desarrollo no lineal: No trabaja de manera lineal. La linea del tiempo tiene bifurcaciones de forma independiente al proyecto principal.
* Completamente distribuido: Cada quien puede tener una copia del proyecto.
* Capaz de manejar grandes proyectos: Proyectos tan grandes como Linux (Django, Laravel, etc)

FUNDAMENTOS DE GIT

* Instantáneas no diferencias: Git almacena referencias a los archivos que no se han cambiado.
* Además cualquier operación en Git es local, puedes trabajar offline.
* Git tiene integridad. No puedes perder información durante su transmisión o sufrir corrupción de archivos sin que Git lo detecte. No importa lo que hagas, Git lo almacena todo, sea tirar para atrás o para adelante quedando todo registrado. Almacena estos cambios en una clave **SHA-1**, un conjunto de 40 carácteres hexadecimales que van a ser una referencia a uno de estas instantáneas(cambios). Ej: da39a3ee5e6b4b0d3255bfef95601890afd80709

**Los tres estados de Git**

Los tres estados de Git son la forma de como introducir cambios y representan el ciclo de vida de nuestros archivos en Git. Estos son:

* Working Directory: Es lugar donde se trabajan los archivos.
* Staging Area: Área de preparación de los archivos que serán registrados en el repositorio.
* Git Repository (repository): Es el lugar donde se almacenan los archivos de forma segura. Para llegar a tener los archivos almacenados en el repositorio tendremos que hacer un commit. Cuando el commit esté completado ya habremos completado el ciclo de vida de nuestros archivos en Git y volvemos al working directory donde empezará de nuevo cuando hagamos algún cambio.
* 

**Diferencias entre Git y Github**

Git no es lo mismo que GitHub. Pueden trabajar en conjunto pero también pueden trabajar totalmente separados.

Git es nuestro sistema control de versiones (VCS), el software que nos va a ayudar a versionar nuestro código en instantáneas.

GitHub es una plataforma que sirve como hosting para almacenar nuestro repositorio. Github te permite utilizar ciertas herramientas que extienden la funcionalidad base de Git, como por ejemplo compartir y redistribuir nuestro repositorio con más gente. Github también funciona como una red social para programadores.

**Instalación y configuración de Git**

Para descargar el instalador de Git en Mac hay que dirigirse a <https://git-scm.com> y darle al botón de descargar. (Git viene instalado por defecto en Mac OS)

Para descargar e instalar Git en Linux hay que instalarlo desde el correspondiente administrador de paquetes de cada distribución:

Debian y derivados (Ubuntu, Linux Mint, etc):

* sudo apt-get install git

Fedora 24 y superiores:

* sudo dnf install git

CentOS, Fedora 23 y menores:

* sudo yum install git

Arch Linux y derivados:

* sudo pacman -S git

Comandos básicos de Git

* **git** : lista de comandos disponibles en git
* **git --version**: Muestra en pantalla la versión de git instalada en nuestro equipo
* **git config --help** : Información del comando config y como configurar Git
* **git config --list** : Para ver las configuraciones de Git hechas anteriormente
* **git config --global (variable) (configuración asignada)**: Para añadir una configuración global dentro de una variable.

*Ej:* ***git config --global user.name*** *“Daniel Cueto”*

* + **git config --global user.email** (email)
  + **git config --global user.name** (nombre)
  + **git config --global color.ui** true
* **git config --global --list**: Para ver las configuraciones globales de Git hechas anteriormente

Si quieres tener una configuración local para un proyecto en concreto puedes utilizar:

* **git config** (sin el flag --global).

Esto te permitirá crear una configuración local para el proyecto en concreto que estés trabajando.

Para aprender los comandos básicos y el funcionamento básico de Git, Github nos proporciona un “emulador” de shell para ir probando los comandos desde el navegador: <https://try.github.io/>

La documentación oficial de Git: <https://git-scm.com/doc>

**Comandos útiles de la terminal**

Movernos entre carpetas y archivos

* **cd** (Change Directory) -> Nos sirve para movernos entre directorios. Ej: “cd proyectos/cursos/git”.
* **mkdir** (Make Directory) -> Nos sirve para crear directorios. Ej: “mkdir repositorio”.
* **ls** () -> Lista los archivos y directorios del directorio.
* ls -a -> Lista todos los archivos y directorios del directorio, incluido los ocultos.
* **clear** -> Limpia la terminal. No borra el historial del terminal sino que nos tira para abajo.
* **touch** -> Comando para crear archivos. Ej: “touch archivo.txt”.
* **rm** -> Comando para borrar archivos. Ej: “rm archivo.txt”.
* **rm -rf** -> Comando para borrar directorios. Ej: “rm -rf repositorio”.

Editor de archivos en terminal: VIM

* Entrar dentro del archivo: “vim archivo.txt”.
* Comandos dentro de VIM:
  + **i** -> poder editar el archivo.
  + Presionar “esc” para poder introducir un comando.
  + **:wq** -> guardar y salir de VIM.
  + **:q!** -> salir de VIM sin guardar el archivo.

Para instalar Oh My Zsh! y tener una terminal mejorada que te especifica ramas de Git y muchas cosas más, copiar el siguiente texto en la terminal:

sh -c "$(curl -fsSL <https://raw.github.com/robbyrussell/oh-my-zsh/master/tools/install.sh>)"