SAUL ALEXANDER LOPEZ HERRRA
3001259330101
999011806

24/02/2023

GIT

Que es GIT

Git es un sistema de control de versiones gratis y de código abierto. Es colocado como el sistema de control de versiones más popular por su gran comunidad de contribuidores.

Este sistema de control de versiones fue creado por Linus Torvalds para el desarrollo del kernel del sistema operativo Linux. Y luego por su gran versatilidad, rapidez y grandes características fue adoptado por prácticamente todos los proyectos de código abierto como estándar para volver el proyecto colaborativo y sea fácil de distribuir.

Git a diferencia de muchos sistemas control de versiones, es un sistema distribuido. La versión del proyecto distribuido en todas la maquinas/dispositivos tiene un clon local del proyecto original con todos los cambios históricos. Esto hace de Git un sistema aislado para el desarrollo privado como también tiene la facilidad de sincronizarse con sistemas remotos para distribuir los cambios a todos los demás repositorios.

Git además cuenta con una flexibilidad, rapidez y seguridad que muchos controles de versiones anteriores y posteriores no cuentan, haciendo su uso factible para múltiples proyectos desde los mas sencillos hasta proyectos grandes.

Control de versiones con GIT

Un control de versiones es usualmente utilizado en un proyecto de desarrollo de software, donde se lleva el control de versiones de los cambios hechos a el código fuente del proyecto. A pesar de que GIT fue creado para el control de versiones de software, últimamente también se ha adoptado para control de versiones de para diferentes tipos de desarrollos por su flexibilidad, rapidez, seguridad y gran comunidad.

El control de versiones en Git se basa en un sistema de versiones distribuidos, cada uno de los clientes cuenta con una copia exacta de todo el repositorio. Este repositorio cuenta con todos los históricos y cambios hechos a cada uno de los archivos que conforman el repositorio. Este sistema no centralizado ayuda a que los clientes no dependan de una conexión constante con el servidor para trabajar en un proyecto y que haya múltiples desarrollos realizándose en un mismo proyecto al ser destruido clones iguales en cada copia y luego los cambios son juntados al enviarse al servidor central.

Git como ya se mencionó maneja un histórico de todos los cambios en un repositorio. Git maneja estos históricos como una imagen instantánea de cada uno de los archivos y no se basa su funcionamiento con guardar únicamente los cambios efectuados encada versión.

SAUL ALEXANDER LOPEZ HERRRA 24/02/2023
3001259330101
999011806

Para que Git, cumpla con la flexibilidad y rapidez que son características de él. Únicamente guarda imágenes de los archivos si estos fueron modificados, si no únicamente se hace un enlace entre cada versión de archivos no modificados.

Estados de un archivo en GIT

GIT tiene 3 estados fundamentales en los que mantiene los archivos de un repositorio

1) Committed, confirmado.
2) Staged, preparado.
3) Modified, modificado.

Estos son los tres principales estados en los que puede estar tus archivos en un repositorio de Git, hay más estados, pero para el proceso de control versiones básico y regular no se usan y son usados en casos más particulares y que son dependientes de alguno de estos tres estados.

Commited

En español, confirmado, es un estado en el que los archivos están almacenados en la base de datos local correctamente. Este seria el ultimo estado de cualquier cambio realizado a un archivo, a este se le asigna un descripción de los cambios realizado y se un hash para comprobar la integridad de cada archivo guardado en la base de datos.

Staged

En español, preparado, es un estado en el que los archivos han sido modificados y que este cambio se guarde para la siguiente confirmación. Durante el estado de preparación del archivo aun no se ha guardado, es útil para llevar control de pequeños cambios en entornos de pruebas sin saturar la base datos local con varios cambios pequeños del mismo archivo.

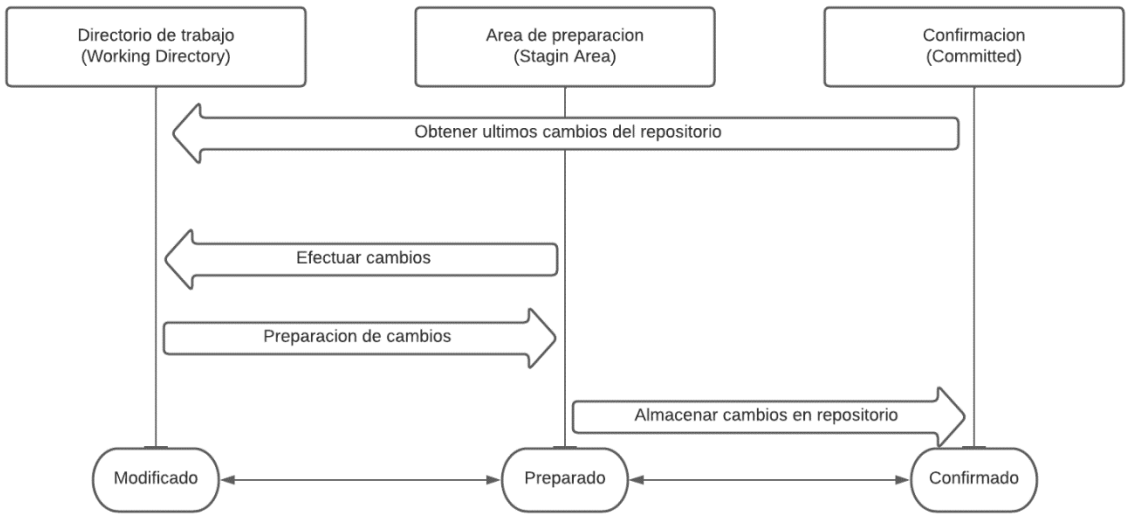
Modified

En español, modificado, es un estado en el que los archivos han sido modificados con respecto a la última versión del archivo almacenado en la base de datos local estos cambios son volátiles y se pueden perder si no son preparados o confirmados en el proceso de control de versiones.

Es el estado normal de cualquier archivo en el que se esté trabajando.

Todos estos datos del repositorio de Git, son almacenados en una carpeta oculta ene l directorio raíz (.git) aquí se guardan todos los archivos históricos comprimidos, con su respectivo hash para asegurar la integridad y seguridad de cada uno de ellos.

SAUL ALEXANDER LOPEZ HERRRA 24/02/2023
3001259330101
999011806



Como se configura un repositorio

Git trabaja en su versión oficial se trabaja con una línea de comandos, denominados Git.

Estos comandos, configuran e interactúan con el repositorio en el que estamos trabajando. Para configurar un repositorio es necesario instalar este CLI que es el medio por el que nos comunicamos usando la consola.

Una vez instalado podemos verificar la versión de git que tenemos instalada con **git –versión.**
Al tener instalada la versión deseada. Podemos navegar al directorio en nuestro equipo en el cual vamos a tener nuestro proyecto. En este directorio agregamos el siguiente comando **git init** este comando inicializa un repositorio Git en el directorio. Todos los archivos y directorios dentro de este directorio raíz se vuelven parte del repositorio, y git mantendrá un control de versiones de cada archivo en el.

Para agregar el repositorio a un servidor de Git, usamos el comando **git remote add [url]** donde URL es la url del servidor remoto.

Comandos en GIT

Los comandos de git es la forma de interactuar con nuestro repositorio para configurarlo e ir añadiendo cambios de nuestros archivos, etc. En este caso los voy a separar según la funcionalidad de los comandos.

- ***Crear***
  - ***git init,*** *se utiliza para inicializarun repositorio de manera local*
  - ***git clone,*** *se utiliza para inicializar un repositorio local desde uno remoto*
- ***Navegar***
  - ***git status,*** *muestra el estado del directorio actual y los archivos preparados*

- ***git log,*** *muestra el historial de commits*
- ***git blame,*** *muestra el contenido línea línea con las modificaciones con su respectivo autor*
- ***git show,*** *muestra información mas extendida de cada archivo y commit*
- ***git diff,*** *sirve para ver los cambios a un archivo.*

- ***Cambios***
  - ***git add,*** *agrega el archivo o archivos para preparar para el commit.*
- ***Revertir***
  - ***git reset,*** *desase los cambios realizados en el repositorio local.*
  - ***git revert,*** *restaura los cambios de un commit anterior pero haciendo un hijo de esos cambios obre el historial.*
- ***Actualizar***
  - ***git pull,*** *para actualizar desde un repositorio remoto.*
  - ***git fetch,*** *para actualizar desde un repositorio remoto de manera segura*
  - ***git merge,*** *mezcla diferentes ramas en una rama destino*
- ***Ramas***
  - ***git branch,*** *crea una nueva brancha con el nombre asignado*
  - ***git checkout,*** *para navegar entre ramas del repositorio*
- ***Commit***
  - ***git commit,*** *almacena un cambio de versión en la base de datos local con u comentario de versión.*
- ***Push***
  - ***git push,*** *sube los cambios de un repositorio local a un repositorio remoto de destino.*