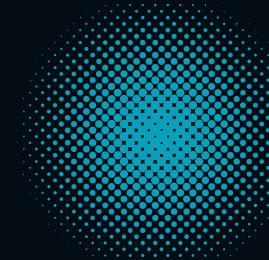


 GIT

QUE ES ?



git



TERMINOLOGIA

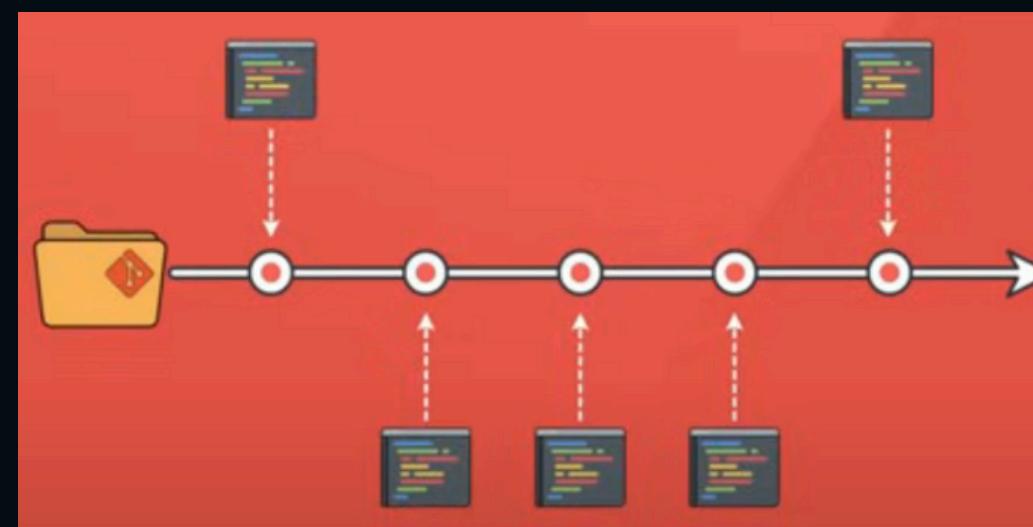
Repository

Es todo proyecto que es seguido por git



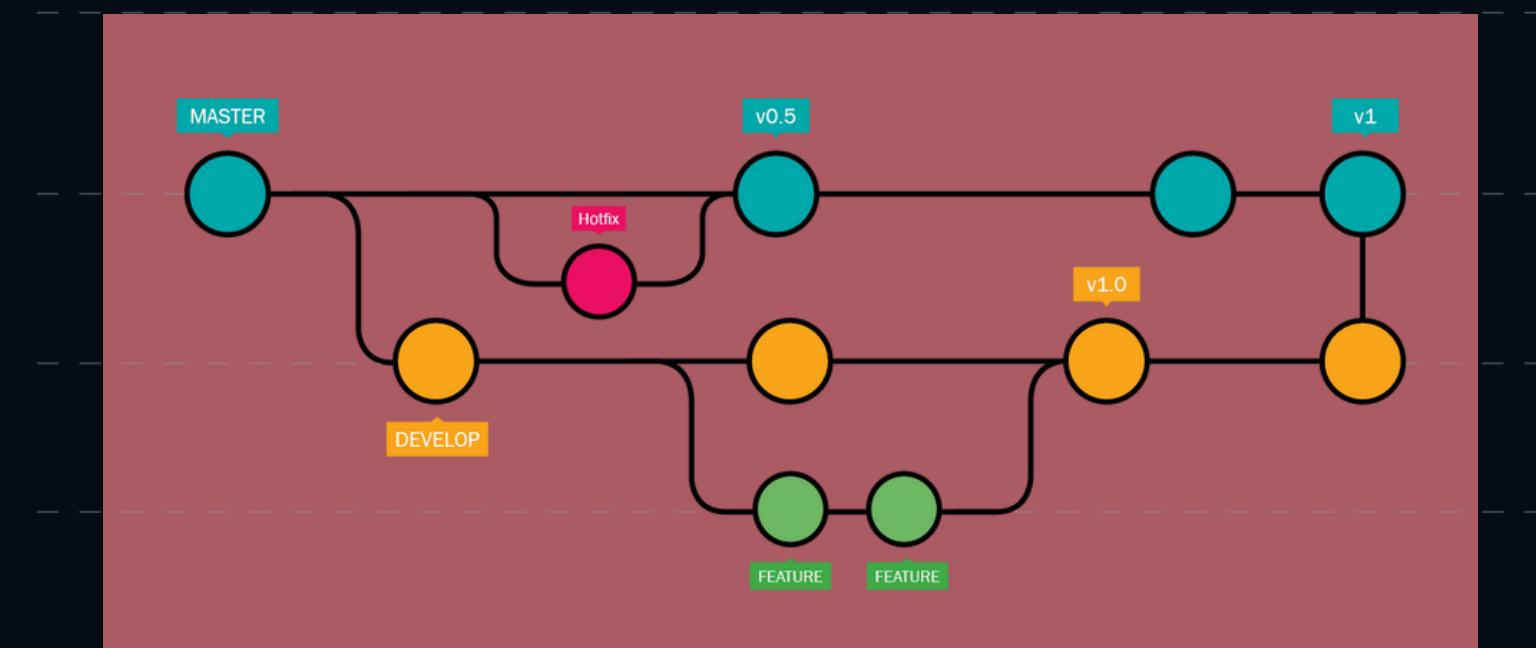
Commit

Cada uno de los cambios registrados en el historial de git, ejecutado por el desarrollador



Rama

Son ramificaciones, bifurcaciones, nuevos caminos que toma el proyecto



TERMINOLOGIA

Clon

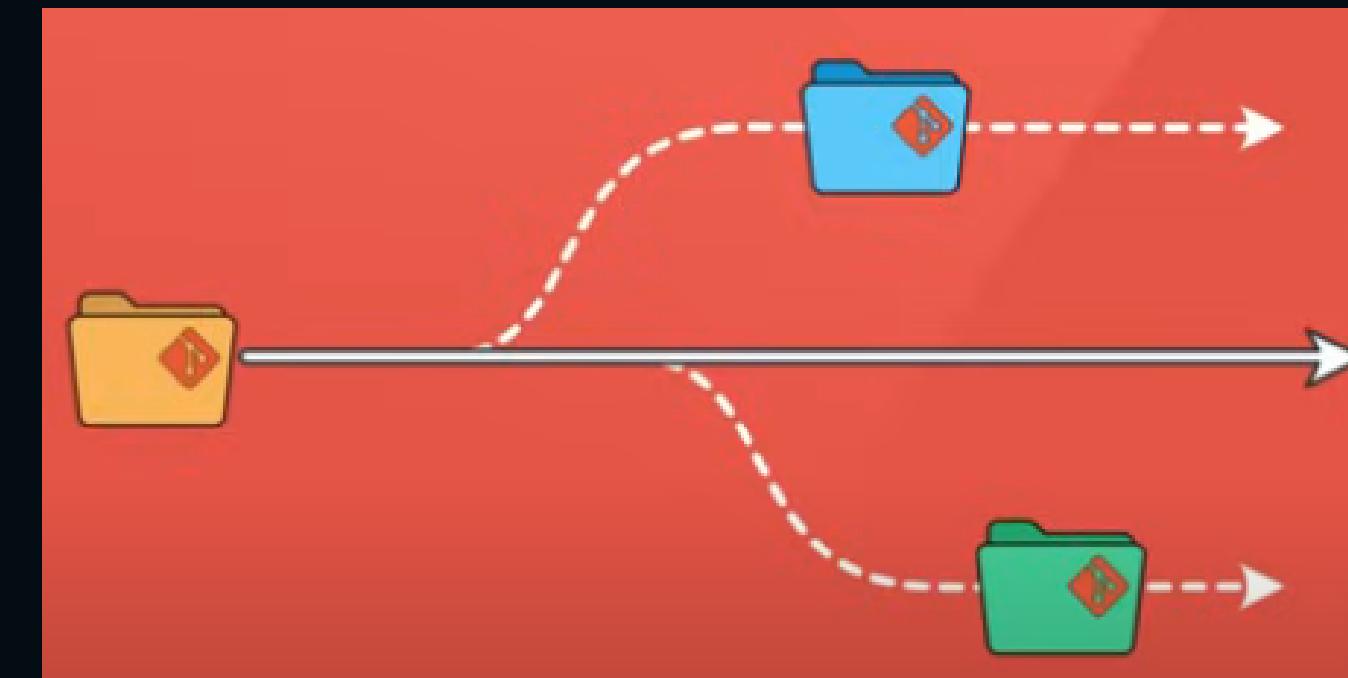
Copia exacta exacta del repositorio



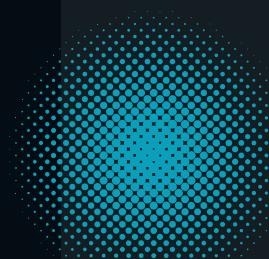
Fork

A diferencia de un clon y una rama, es un proyecto totalmente diferente en base a un proyecto

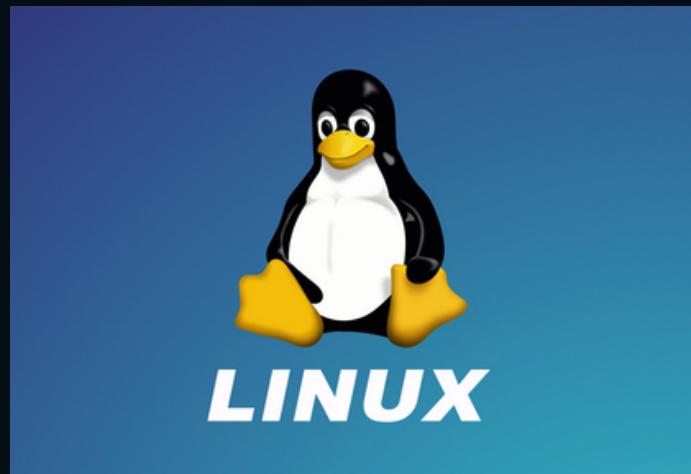
Toma su propio camino



HISTORIA



Linus Benedict Torvalds





Linus Torvalds sobre GIT

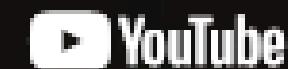


Share



que fue creado para poder
mantener mi primer gran proyecto.

Watch on



CARACTERISTICAS

01

Distribuido

- Repositorios locales completos: Cada desarrollador tiene una copia completa del proyecto en su máquina, incluyendo el historial y las ramas.
- No depende de una conexión constante: Las operaciones como commits, diffs y reversiones pueden hacerse localmente, sin conexión a internet.
- Múltiples flujos de trabajo: Dado que cada copia es independiente, permite flujos de trabajo más flexibles, como la contribución a proyectos desde distintas fuentes.
- Respaldo automático: Tener un repositorio distribuido significa que hay múltiples copias del historial del proyecto, lo que actúa como un respaldo.

CARACTERISTICAS

02

Ramas y Fusión (Branching and Merging)

- Ramas ligeras: Crear y borrar ramas en Git es rápido y consume pocos recursos, lo que fomenta su uso regular.
- Flujo de trabajo paralelo: Varios desarrolladores pueden trabajar en diferentes características sin interferir entre sí.
- Fusionado sencillo: Git facilita la integración de cambios en ramas independientes mediante comandos como merge y rebase, resolviendo conflictos de manera automática cuando es posible.
- Ramas a largo y corto plazo: Se pueden usar ramas a largo plazo como main o develop, y ramas temporales para características o correcciones puntuales.



when you merge your code with master
and everything breaks



CARACTERISTICAS

03

Integridad

- SHA-1 para verificación: Git asigna un hash SHA-1 único a cada commit, lo que asegura que los datos no se alteren accidentalmente.
- Confianza en los cambios: Cualquier cambio en los datos del proyecto es detectable debido a la verificación del hash.
- Seguridad en el historial: El historial de un proyecto no puede alterarse sin que Git detecte las modificaciones, lo que protege contra la manipulación de datos.

¿Qué es SHA-1?

Algoritmo de Hash Seguro o Secure Hash Algorithm (encriptacion sha1)

CARACTERISTICAS

04

Historial Completo

- Snapshots en lugar de diferencias: Git guarda un snapshot completo del proyecto en cada commit, no solo diferencias entre versiones.
- Acceso rápido a versiones previas: Con comandos como git log o git checkout, se puede navegar fácilmente por el historial de cambios.
- Reversión de cambios: Es fácil revertir un cambio problemático o un commit erróneo sin afectar otros aspectos del proyecto.
- Documentación implícita: Cada commit incluye un mensaje descriptivo, permitiendo rastrear el propósito de los cambios.

CARACTERISTICAS

05

Alto Rendimiento

- Optimizado para velocidad: Operaciones como el commit, clonación o creación de ramas son extremadamente rápidas debido a su arquitectura distribuida y su eficiente manejo de datos.
- Comparación rápida: Comandos como git diff y git log permiten comparar el código y ver cambios entre versiones de forma muy eficiente.
- Optimización de espacio: A pesar de guardar snapshots completos, Git utiliza técnicas de compresión y almacenamiento eficiente para mantener el uso de espacio bajo.

CARACTERISTICAS

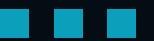
06

Compatibilidad con Herramientas y Plataformas

- Integración con servicios online: Git se integra con plataformas como GitHub, GitLab y Bitbucket para la colaboración, gestión de proyectos y control de versiones en la nube.
- Extensión por plugins: Git es compatible con una gran cantidad de herramientas de terceros que permiten mejorar la funcionalidad (como hooks de pre-commit o integración continua).
- Interfaces gráficas: Existen múltiples interfaces gráficas (como GitKraken, Sourcetree) para quienes prefieren evitar el uso de la línea de comandos.
- Edición en IDEs: Muchos entornos de desarrollo integrados (IDEs) como Visual Studio Code, IntelliJ o Eclipse tienen soporte nativo para Git, permitiendo trabajar con versiones sin salir del editor.

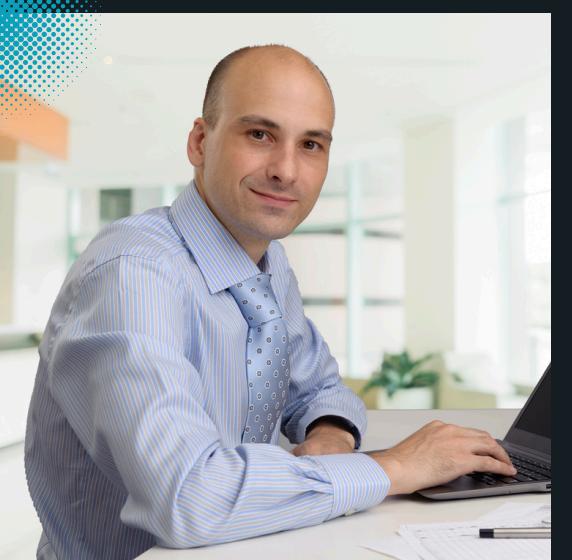
GIT O GITHUB?





GITHUB

RED SOCIAL DEL CODIGO



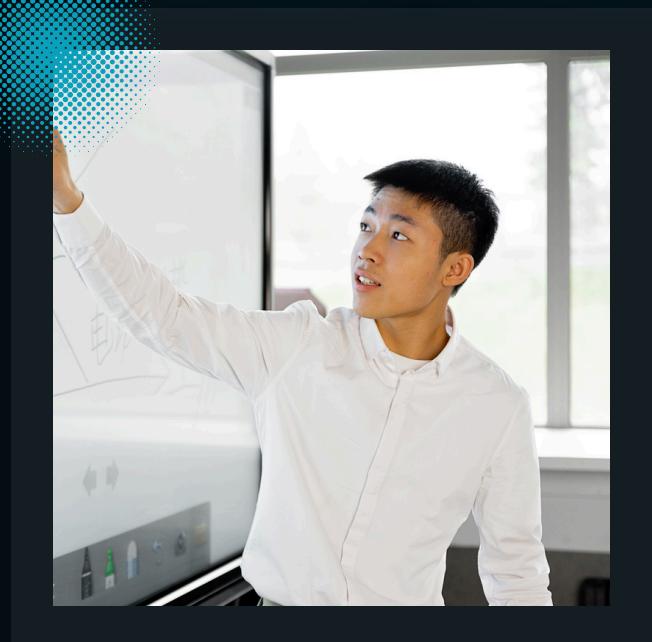
Francois Mercer

CEO en Facebook



Cod-007

Espia



Daniel Gallego

Desarrollador en Valve



QR

Paga tus salteñas! >:c

GIT

- Es un sistema de control de versiones distribuido:
- Funciona localmente:
- Independiente de la red:
- Es una herramienta de línea de comandos

GITHUB

- Es una plataforma basada en la nube: GitHub es un servicio web que proporciona alojamiento para repositorios Git. Es decir, permite almacenar y compartir repositorios de Git en línea para que otros puedan colaborar.
- Facilita la colaboración: GitHub añade funcionalidades adicionales a Git, como herramientas de gestión de proyectos, issues (problemas), pull requests, y revisiones de código, que están diseñadas para facilitar la colaboración entre equipos.
- Red social para desarrolladores: GitHub también funciona como una red social para desarrolladores, donde pueden compartir código públicamente, seguir a otros desarrolladores, y contribuir a proyectos de código abierto.
- Almacenamiento en la nube: GitHub almacena repositorios en servidores remotos, lo que permite a los desarrolladores sincronizar sus cambios desde sus repositorios locales (en Git) hacia GitHub y viceversa.
- Características extra: Ofrece integración con sistemas de integración continua (CI/CD), GitHub Actions (automatización de flujos de trabajo), páginas de documentación (GitHub Pages), y más.

THANK YOU

WWW.REALLYGREATSITE.COM