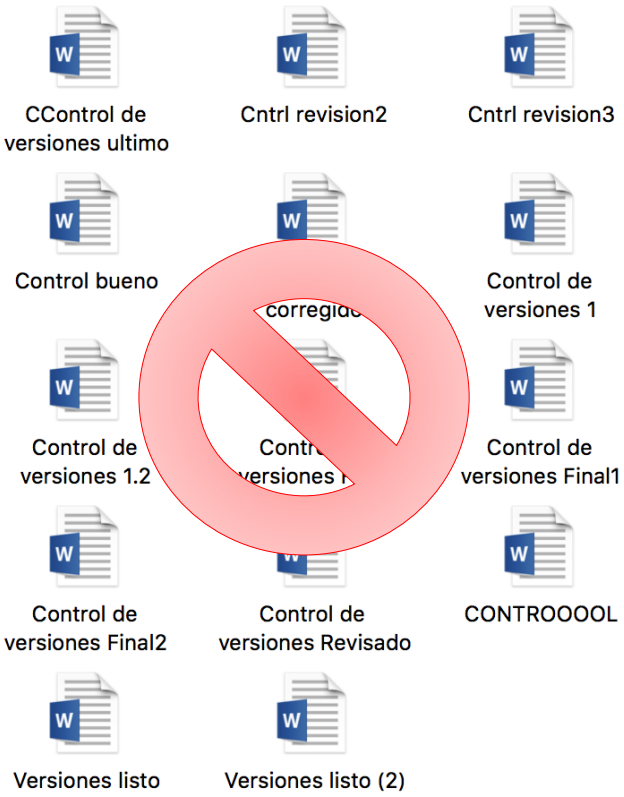
El control de versiones

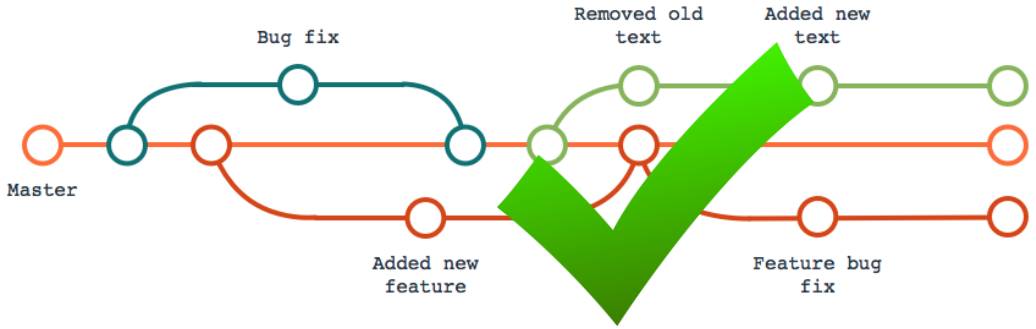
GitHub

git

Índice

El control de versiones

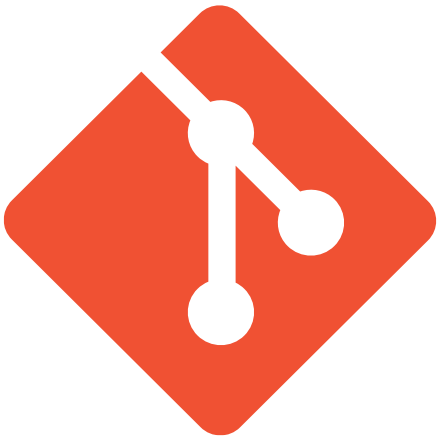




El control de versiones como su nombre indica es el control sobre los cambios de un proyecto, como sus actualizaciones, mejoras, etc… Con ello se crea un histórico del código el cual si en un momento se da un error se puede volver a la versión anterior. Con esto evitamos errores o engendrar múltiples proyectos iguales y tener el proyecto inicial limpio y ordenado.

Nos centraremos en el control de la codificación en el ámbito de la programación, pero esto también sirve y se utiliza en cualquier ámbito, como trabajo y proyecto.

Que es GIT y quien lo invento



git

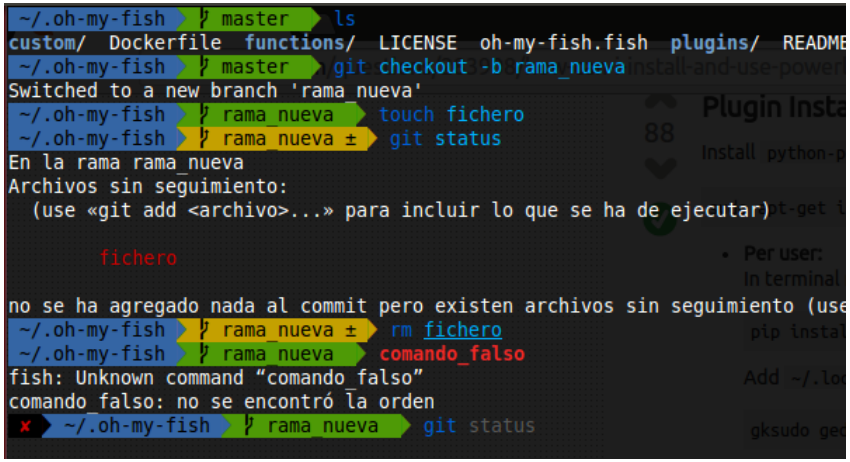
GIT como tal es el software de control de versiones que nos ayuda a manejar esta mecánica. Fue diseñado por Linus Torvalds, quien creo esto como una mejora personal en la organización de proyectos, y hoy en día se utiliza incluso de forma empresarial. Lo conoceréis como el creador de Linux. GIT se diseño con un algoritmo seguro llamado SHA1, con ello se salvaguarda el código y el historial de cambios frente modificaciones y que el historial sea totalmente trazable.

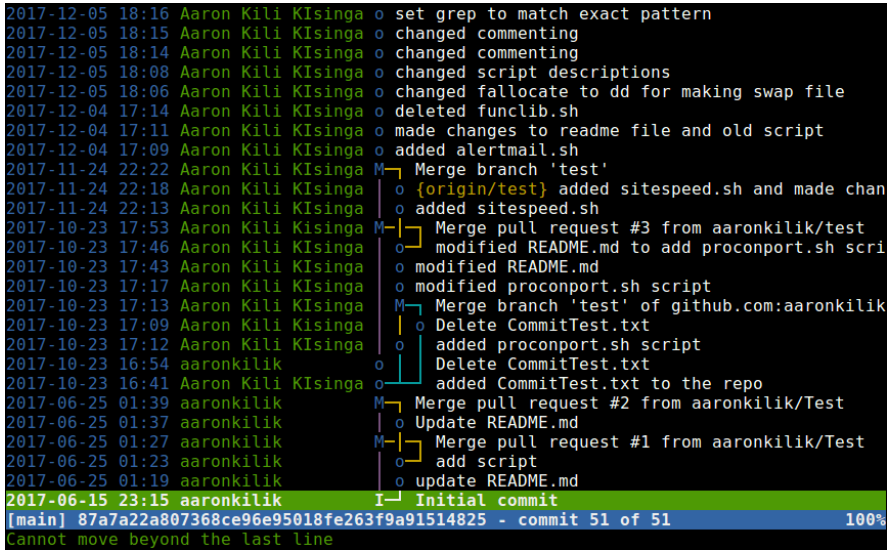
Que es GitHub o GitPot

git	GitHub
Herramienta de control de versiones distribuida	Plataforma basada en la nube
Herramienta de código abierto que los desarrolladores instalan localmente para gestionar el código fuente	Servicio en línea al que los desarrolladores que utilizan Git pueden conectarse y cargar o descargar recursos

GitHub y GitPot son dos web parecidas de servicio en la nube que ayudan a desarrolladores y aficionados a controlar y administrar de forma histórica sus proyectos. GitHub ofrece sin cobrarte nada (en caso de ser particular) un servicio de hosting donde podrás subir de forma publica o privada tus proyectos, incluso te permite crear tu propia pagina web. Básicamente, es un sitio de redes sociales para desarrolladores, y de proyectos personales o comunes.

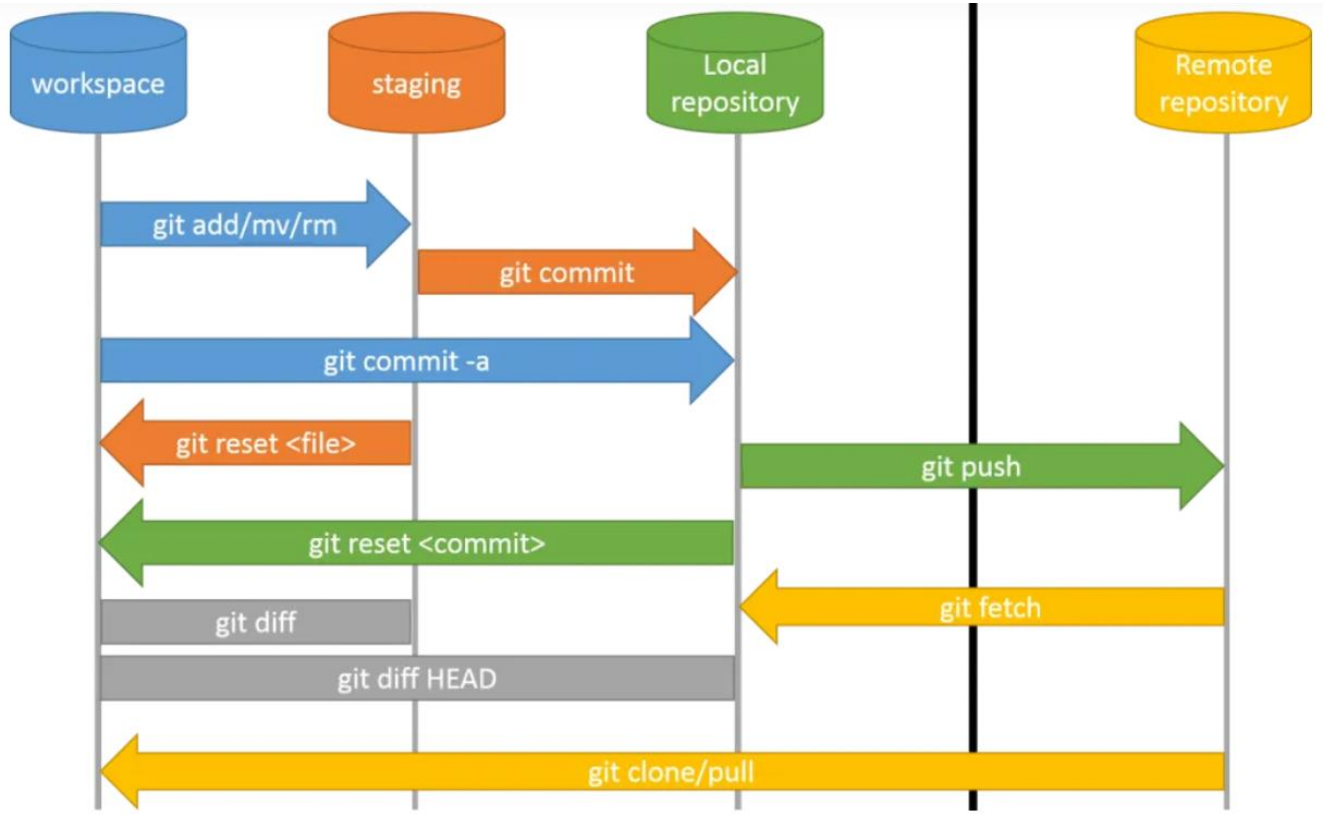
No todo es tan bonito





Git es muy útil a la hora de gestionar tu proyecto, pero su manejo se basa principalmente en la terminal y tiene una curva de aprendizaje alto. Existen opciones graficas no oficiales, como una oficial que ayudan a manejar git de forma mas amigable. Una de ellas es GitHub Desktop: link
Otra forma que se maneja muy bien, es directamente por Visual Studio Code (VSCode), que es un software de codificación potente que permite editar en multitud de lenguajes e IDE’s.

Manejo de GIT en el equipo local



En esta imagen se representa gráficamente como actúa GIT cuando lo estamos usando. De los diferentes comandos, veremos 7 de los cuales son mas utilizados.

Manejo de GIT en el equipo Local

A la hora de crear nuestro proyecto, en una terminal direccionada en la carpeta principal de este, utilizaremos el comando **git init** que solo lo usaremos una vez para iniciar un repositorio local en el equipo. Se creara una carpeta **.git** donde se guardara la configuración del GIT.

Otro comando muy utilizado y que se suele usar también solo una vez es el **git clone <url>** con el clonaremos un repositorio existente a nuestra carpeta local. Si el proyecto es publico o en grupo, entre el administrador del proyecto y tu podéis discutir cambios en el proyecto.

Una vez creado parte del proyecto o haber modificado una repo, podéis actualizar el repositorio local primero añadiendo la modificación a staging con el comando **git add** después tendremos que subir al local repositorio con **git commit -m <mensaje>**

Para que se entienda mejor, **git add** lo que hace es subir la actualización en un área intermedia donde puedes tirar atrás si falta algo a añadir, o quieres añadir un archivo que faltaba. El **git commit** crea una actualización con una etiqueta numérica y una descripción para redactar el cambiado que fue trabajado. El comentario no es necesario, pero de esta forma sabrás que habrás echo en ese commit (actualización de versión).

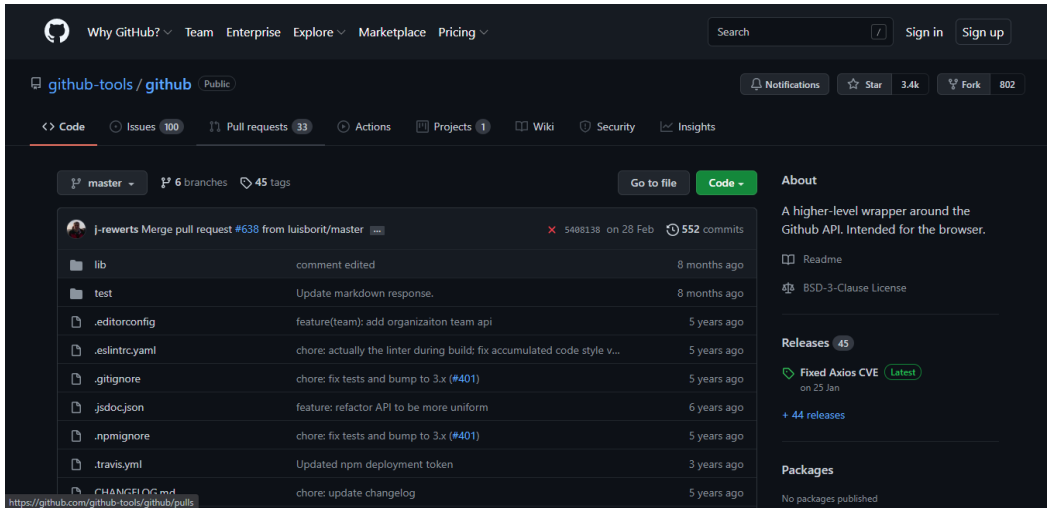
Manejo de GIT en el equipo Remoto

Veremos que con **git reset <file>** podemos tirar atrás el archivo en el área intermedia. O si nos arrepentimos de hacer commit **git reset <commit>** en este caso lo que se pone seria el código generado como commit.

Cuando todo esté listo en el repositorio local, lo ideal es tenerlo en el repositorio remoto, y aquí es donde entra GitHub para esto. Cuando queramos subir la actualización o crear una repo en nuestra cuenta de GitHub lo aremos con **git push** a la vez siempre es recomendable hacer un **git fetch**.

GitHub

[GitHub]



Webgrafía

Página web de GitHub
Página de descarga de GIT
Página de VScode diferencia de control de versiones link
Página web con información más profunda link
Página web de la Wikipedia sobre Git y Linus link
Paginas donde se sacó información e imágenes de aprendeIA link
Pagina donde se sacó información e imágenes de That CS guy link