Hector Roberto Merida Chan
2338701050901

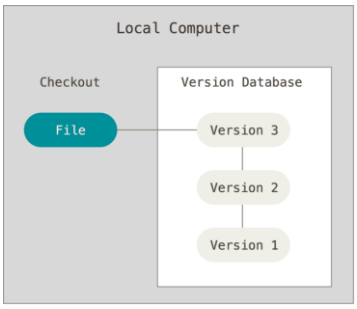
- **Que es GIT**

Git está optimizado para guardar cambios de forma incremental. Permite contar con un historial, regresar a una versión anterior y agregar funcionalidades. Lleva un registro de los cambios que otras personas realicen en los archivos.

- **Control de versiones con GIT**

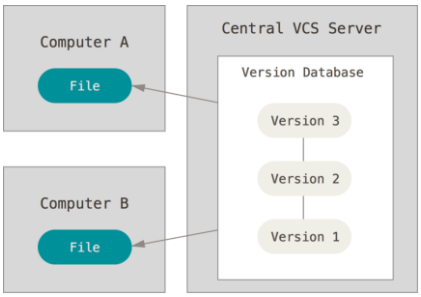
Registra los cambios realizados en un archivo o conjunto de datos a lo largo del tiempo, existen tres tipos principales de control de versiones:

  - **Sistemas de Control de Versiones Locales**



Una de las herramientas de control de versiones más popular fue un sistema llamado RCS, que todavía podemos encontrar en muchas de las computadoras actuales. Incluso el famoso sistema operativo Mac OS X incluye el comando RCS cuando se instalan herramientas de desarrollo.

  - **Sistemas de Control de Versiones Centralizados**

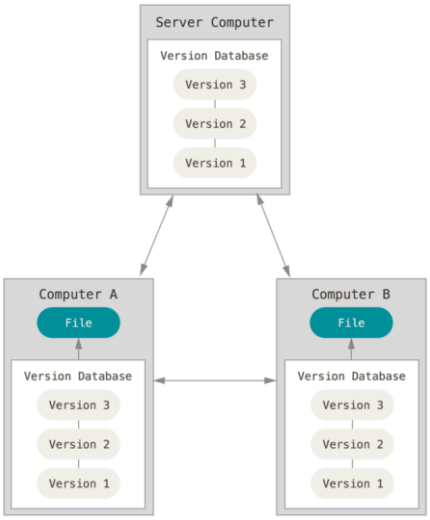


Tienen un único servidor que contiene todos los archivos versionados y varios clientes que descargan los archivos desde ese lugar central. Este ha sido el estándar para el control de versiones por muchos años.

Sin embargo, esta configuración también tiene serias desventajas. La más obvia es el punto único de fallo que representa el servidor centralizado. Si ese servidor se cae durante una hora, entonces durante esa hora nadie podrá colaborar o guardar cambios en archivos en los que hayan estado trabajando. Si el disco duro en el que se encuentra la base de datos central se corrompe, y no se han realizado copias de seguridad adecuadamente, se perderá toda la información del proyecto.

Hector Roberto Merida Chan
2338701050901

- **Sistemas de Control de Versiones Distribuidos**



Los sistemas de Control de Versiones Distribuidos (DVCS por sus siglas en inglés) ofrecen soluciones para los problemas que han sido mencionados. En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no solo descargan la última copia instantánea de los archivos, sino que se replica completamente el repositorio. De esta manera, si un servidor deja de funcionar y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios disponibles en los clientes puede ser copiado al servidor con el fin de restaurarlo. Cada clon es realmente una copia completa de todos los datos.

Hector Roberto Merida Chan
2338701050901

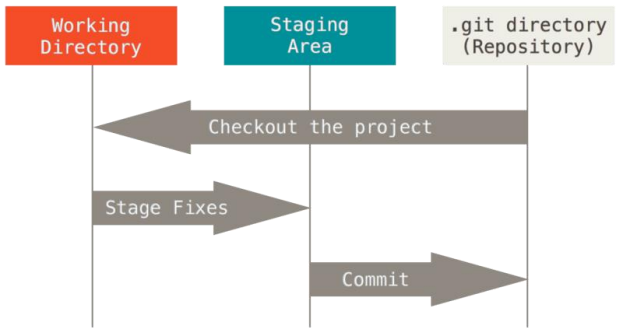
- **Estados de un archivo en GIT**

Confirmado (committed): significa que los datos están almacenados de manera segura en tu base de datos local.

Modificado (modified): significa que has modificado el archivo, pero todavía no lo has confirmado a tu base de datos.

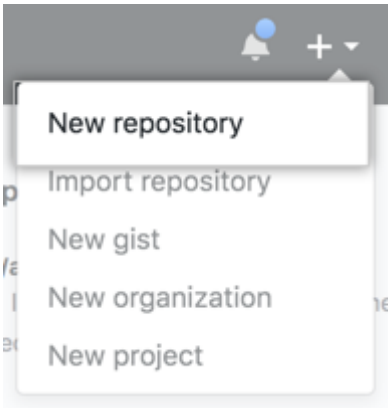
Preparado(staged): significa que has marcado un archivo modificado en su versión actual para que vaya en tu próxima confirmación.

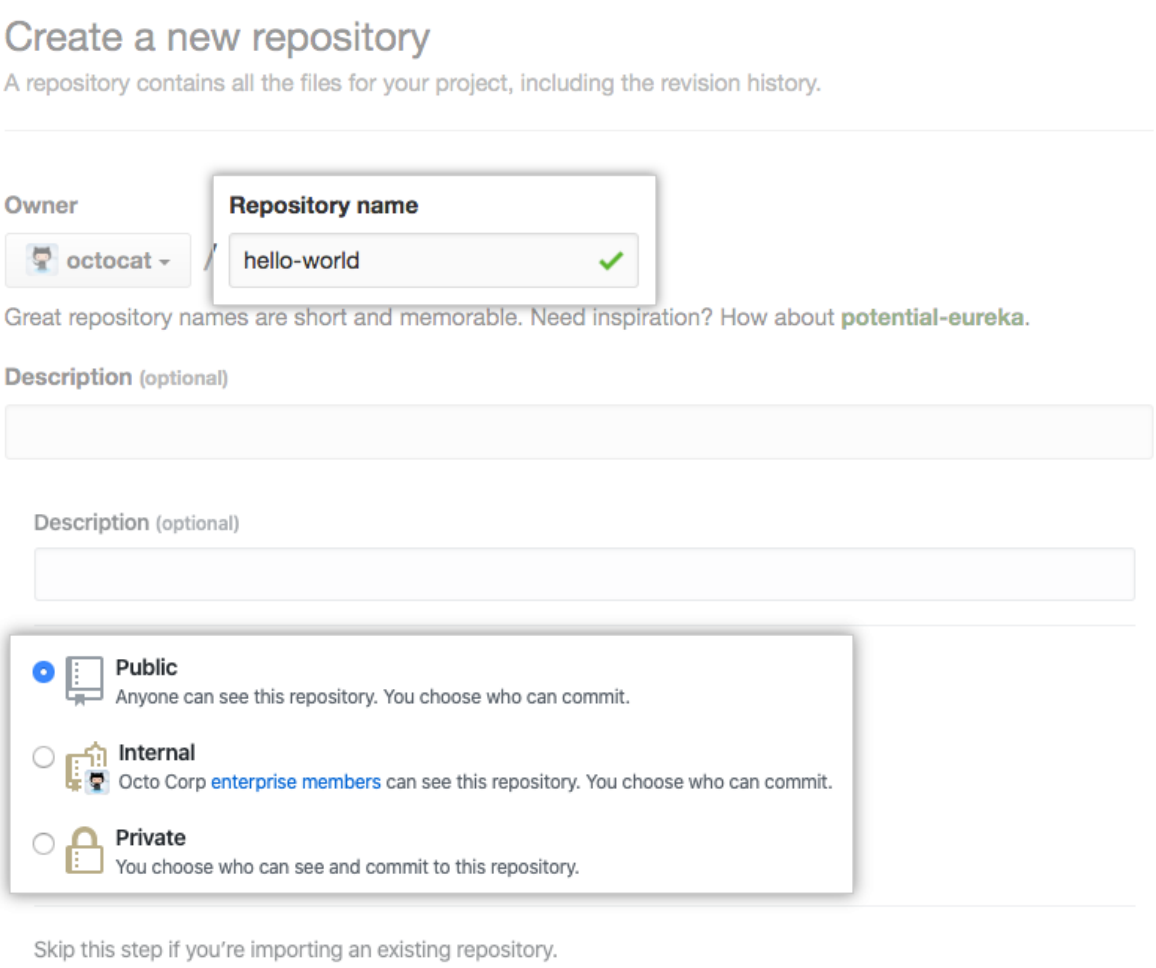
Esto nos lleva a las tres secciones principales de un proyecto de Git:
El directorio de Git (Git directory), el directorio de trabajo (working directory), y el área de preparación (staging area).



Hector Roberto Merida Chan
2338701050901

- **¿Como se configura un repositorio?**





Hector Roberto Merida Chan
2338701050901

- **Comandos en GIT (TOP 10)**

1. Git clone

Git clone es un comando para descargar el código fuente existente desde un repositorio remoto (como Github, por ejemplo.

2. Git branch

Las ramas son muy importantes en el mundo de git. Mediante el uso de ramas, varios desarrolladores pueden trabajar en paralelo en el mismo proyecto simultáneamente. Podemos usar el comando git branch para crear, enumerar y eliminar ramas.

3. Git checkout

Este es también uno de los comandos Git más utilizados. Para trabajar en una rama, primero debe cambiarse a ella. Usamos git checkout principalmente para cambiar de una rama a otra. También podemos usarlo para verificar archivos y confirmaciones.

4. Git status

El comando de estado de Git nos brinda toda la información necesaria sobre la rama actual.

5. Git add

Cuando creamos, modificamos o eliminamos un archivo, estos cambios ocurrirán en nuestro local y no se incluirán en la próxima confirmación (a menos que cambiemos las configuraciones).

6. Git commit

Este es quizás el comando más utilizado de Git. Una vez que llegamos a cierto punto en el desarrollo, queremos guardar nuestros cambios (tal vez después de una tarea o problema específico).

7. Git push

Después de confirmar los cambios (con git commit), lo siguiente que hay que hacer es enviar estos cambios al servidor remoto. Git push sube tus confirmaciones al repositorio remoto.

8. Git pull

El comando git pull se usa para obtener actualizaciones del repositorio remoto.

9. Git revert

A veces necesitamos deshacer los cambios que hemos hecho.

10. Git merge

Comando para fusionar con la rama principal (dev o master branch). Esto se hace con el comando git merge.