**Creación y Modificación de un Repositorio**

**Objetivos:**

**Principal:**

* Profundizar y Dominar los comandos de Git bash aplicado a un repositorio.

**Secundarios:**

* Conocer la función que cumple cada uno de los comandos aplicados en el repositorio
* Aplicar función para poder controlar que puede ser visible y que no.

**Marco Teórico:**

Es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos.

Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o front end como Cogito o StGIT. 3​ Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena. 4​ Hay algunos proyectos de mucha relevancia que ya usan Git, en particular, el grupo de programación del núcleo Linux. El mantenimiento del software Git está actualmente (2009) supervisado por Junio Hamano, quien recibe contribuciones al código de alrededor de 280 programadores.

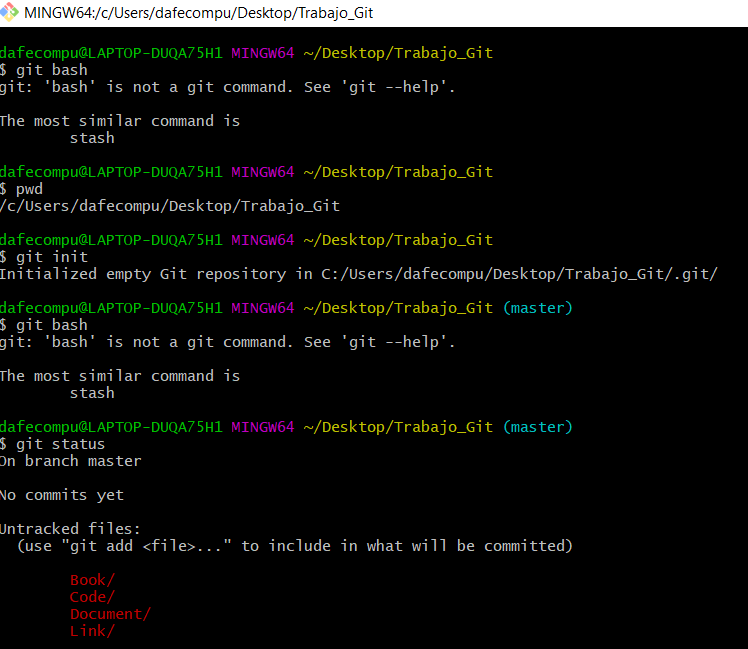
**Características:**

El diseño de Git se basó en BitKeeper y en Monotone. ​ Originalmente fue diseñado como un motor de sistema de control de versiones de bajo nivel sobre el cual otros podrían codificar interfaces frontales, tales como Cogito o StGIT. Desde ese entonces hasta ahora el núcleo del proyecto Git se ha vuelto un sistema de control de versiones completo, utilizable en forma directa.8​

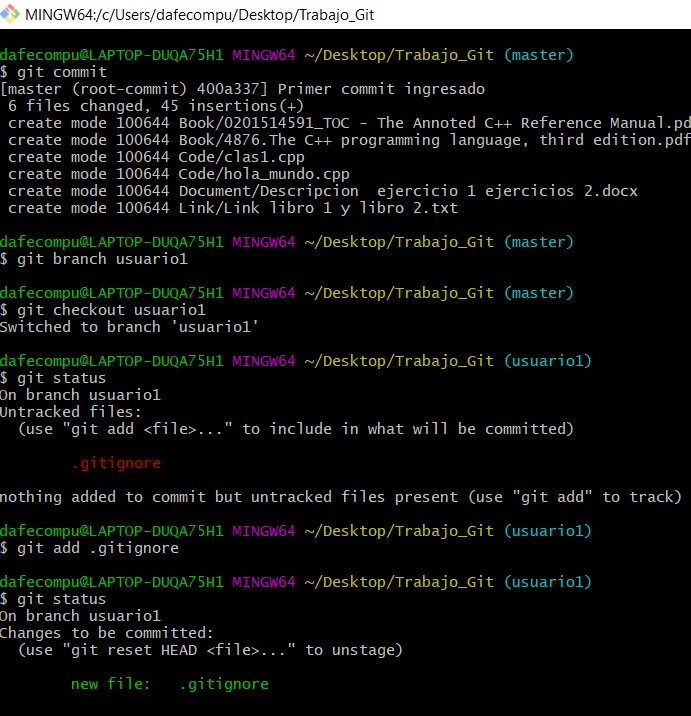
Linus Torvalds buscaba un sistema distribuido que pudiera usar en forma semejante a BitKeeper, pero ninguno de los sistemas bajo software libre disponibles cumplía con sus requerimientos, especialmente en cuanto a desempeño. El diseño de Git mantiene una enorme cantidad de código distribuida y gestionada por mucha gente, que incide en numerosos detalles de rendimiento, y de la necesidad de rapidez en una primera implementación.

**Procedimiento:**

1. Primero seleccionamos un repositorio para abrir con el Git bash
2. Seguido de esto aplicamos el comando “pwd”, que nos servirá para reconocer en que repositorio nos encontramos.
3. Aplicamos el comando “ls” para reconocer que archivos contiene nuestro repositorio
4. Para inicializar ingresamos el comando “” git init”, lo que nos ayuda a comenzar la modificación del repositorio.
5. Seguido de esto ponemos el comando “git status”, para ver si ha existido algunos cambios en el repositorio
6. Si hay cambios dentro del repositorio, debemos agregarlos para esto aplicamos el comando “git add ” más el archivo que se quiere agregar.



1. Una vez ya agregados los archivos, insertamos “git commit”, que nos servirá para guardar las modificaciones realizadas.
2. Después deberemos crear una copia del repositorio para lo cual insertamos “git branch” y después “git checkout”.
3. Una vez en la copia del repositorio podremos agregar nuevos archivos sin que influya en el principal
4. Agregamos un “.gitignore” en nuestro repositorio que contendrá los nombres del os documentos que no queramos que aparezcan.
5. Una vez hecho esto agregamos el .gitignore a nuestro repositorio colocaremos el comando “git add .gitignore”, para asi incluir completamente a nuestro repositorio



**Conclusiones:**

* Logramos conocer y manejar los comandos para modificación de un repositorio en Git Bash
* Hemos aprendido como hacer que un documento pueda ser invisible

Recomendaciones:

* Es recomendable digitar bien cada comando, caso contrario el programa no reconocerá y no realizara ningún cambio
* Además se recomienda saber la función de cada comando para que no exista complicaciones.