# Manual de versionado de software TI

## Control de versiones

El control de versiones es cualquier tipo de práctica que rastrea y proporciona control sobre los cambios en el código fuente, en el desarrollo de software, **el control de versiones distribuido** es una forma de control de versiones en la que toda la base de código, incluyendo su historial completo, se refleja en el ordenador de cada desarrollador. Esto permite la gestión automática de bifurcación y fusión, acelera la mayoría de las operaciones, mejora la capacidad de trabajar sin conexión y no depende de una única ubicación para las copias de seguridad.

Para cualquier proyecto de software creado en CPE se utiliza **GIT** como control de versiones distribuido**. GIT** como con la mayoría de los otros sistemas distribuidos de control de versiones, y a diferencia de la mayoría de los sistemas cliente-servidor, cada directorio de Git en cada ordenador es un repositorio completo con un historial completo y capacidades completas de seguimiento de versiones, independientemente del acceso a la red o a un servidor central. **Git** es software libre y de código abierto distribuido bajo los términos de la Licencia Pública General GNU versión 2.

La gestión de los proyectos de software desarrollados en CPE se realiza a través de **Azure DevOps** de Microsoft. Los repositorios centrales de cada proyecto estarán alojados en **Azure Repos** que es parte de **Azure DevOps.**

### Repositorios centrales y ramificaciones (branch)

Cada proyecto debe tener un repositorio central que es considerado como el repositorio oficial, el cual es administrado por los mantenedores del proyecto. Los desarrolladores clonan este repositorio para crear copias locales idénticas del código base. Los cambios en el código fuente del repositorio central se sincronizan periódicamente con el repositorio local.

El desarrollador crea una nueva rama (branch) en su repositorio local y modifica el código fuente en esa rama. Una vez realizado el desarrollo, el cambio debe ser integrado en el repositorio central.

Las contribuciones a un repositorio se realizan mediante un pull request (solicitud de fusión), el colaborador solicita al responsable del proyecto que fusione el cambio de código fuente El encargado del mantenimiento tiene que fusionar el pull request si la contribución debe formar parte de la base de la fuente.

El desarrollador crea un pull request para notificar a los mantenedores de un nuevo cambio; un hilo de comentario está asociado a cada pull request. Esto permite una discusión enfocada en los cambios de código. Los pull request enviados son visibles para cualquiera que tenga acceso al repositorio. Un pull request puede ser aceptado o rechazado por los mantenedores.

Una vez que el pull request es revisado y aprobado, se fusiona con el repositorio.

## Metodología de numeración de versiones

En el proceso de asignar números de versión únicos a estados únicos al software desarrollado el método habitual de numeración de versiones es el **X.Y.Z**, que generalmente corresponde a los cambios: **Mayor.Menor.Parche (ej. 2.1.5)**. Estos números se asignan generalmente en orden ascendente y corresponden a los nuevos desarrollos en el software.

Cada estado se describe así:

* **Mayor:** Los números de las versiones principales o mayor cambian cada vez que se introduce algún cambio significativo. Por ejemplo, un cambio grande o potencialmente incompatible con una versión anterior de un paquete de software. Si el software no ha tenido ningún reléase previo o aún no está listo para el uso estable este número debe ser **0**.
* **Menor:** Los números de versión menores cambian cuando se introduce una característica nueva y menor o cuando se despliega un conjunto de características menores.
* **Parche:** Los números de parches cambian cuando se lanza una nueva versión del software a los clientes. Esto es normalmente para pequeñas correcciones de errores o similares.

# Guía de uso de GIT (Línea de comandos):

Descargar e instalr Git desde el sitio oficial <https://git-scm.com/>

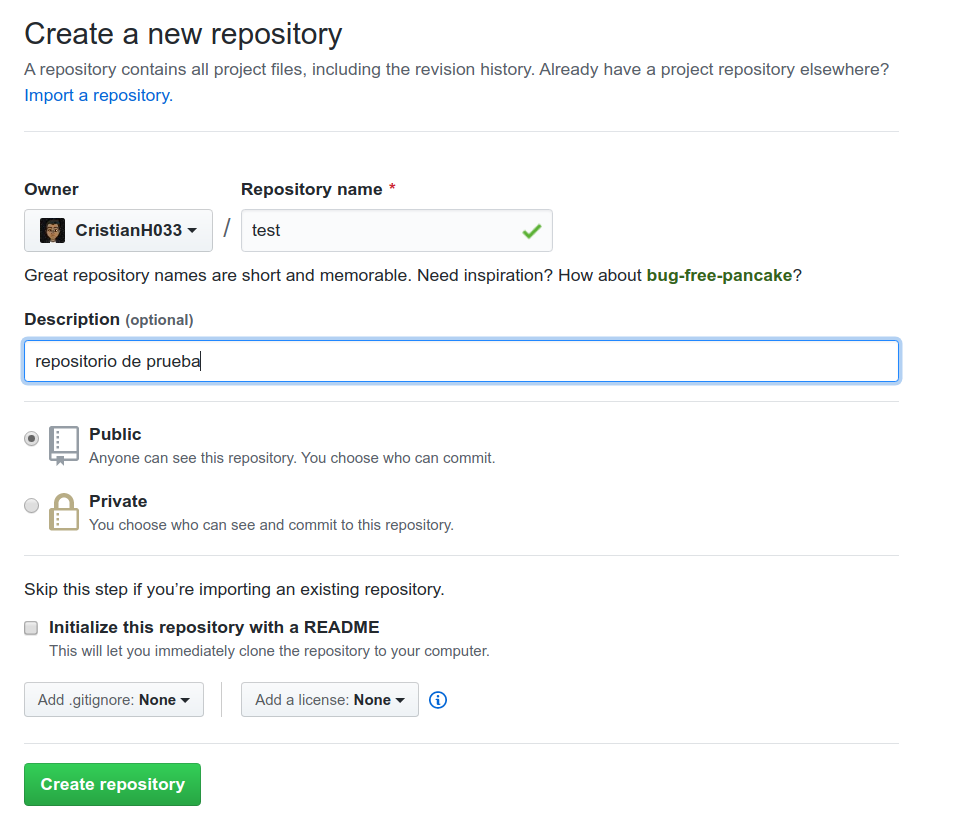
## Crear e iniciar repositorios:

Crea un directorio nuevo, ábrelo y ejecuta git init para crear un nuevo repositorio de git, Git init inicializara un repositorio con los archivos dentro del actual directorio.

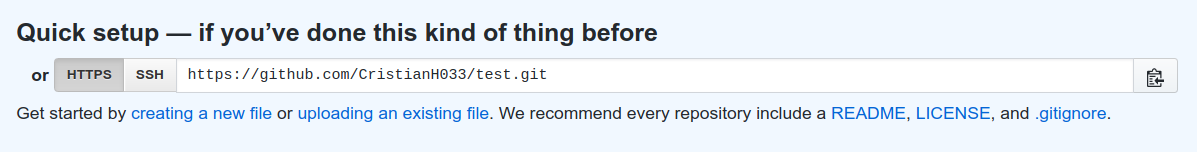
## Clonar repositorios:

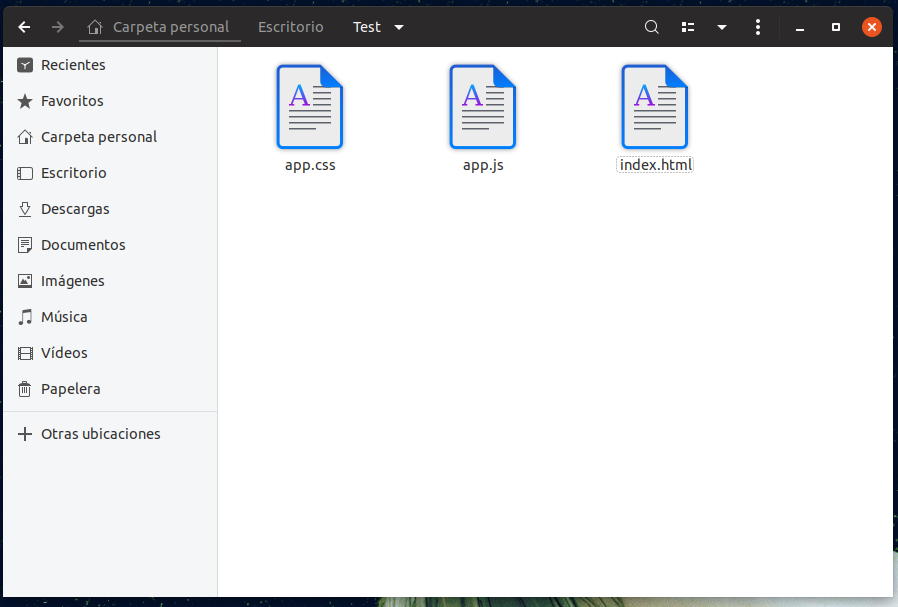
Crea una copia local de un repositorio ejecutando *git clone /ruta/al/repositorio.* Si utilizas un servidor remoto, ejecuta *git clone usuario@host: /ruta/al/repositorio.*

*Crear repositorio en GitHub*

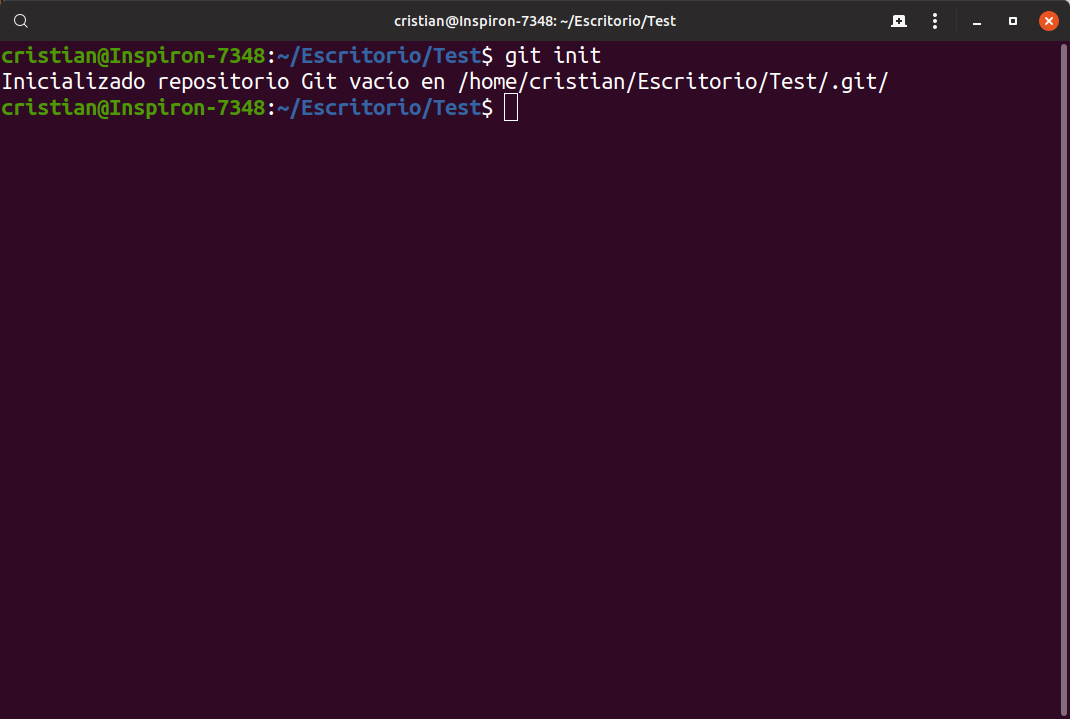


Ruta remota de repositorio



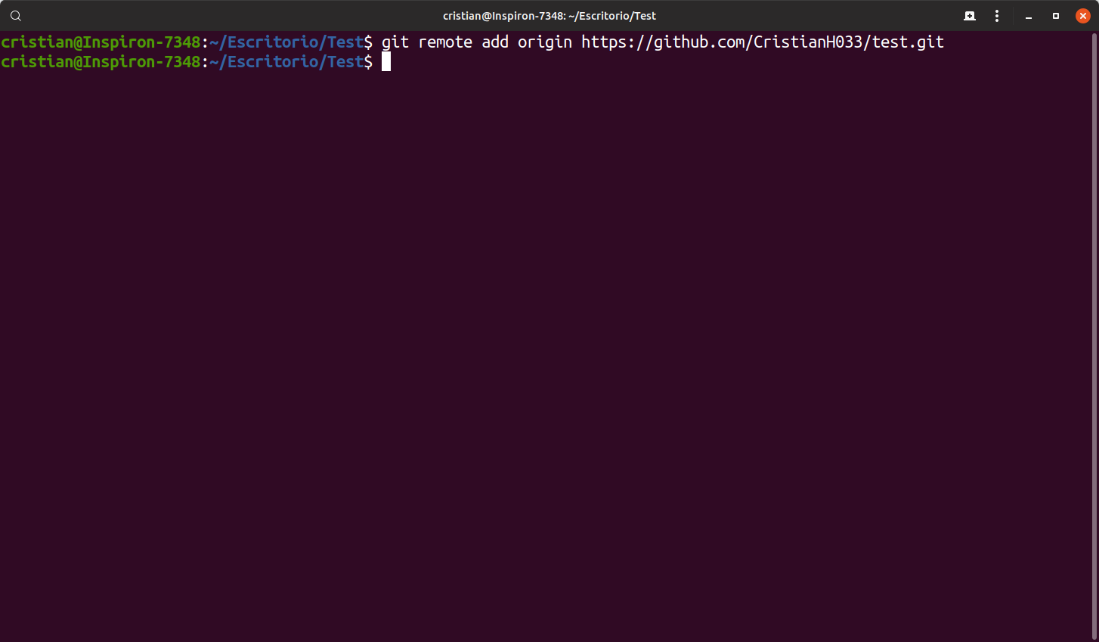


Inicializar directorio como repositorio de Git con el comando git init

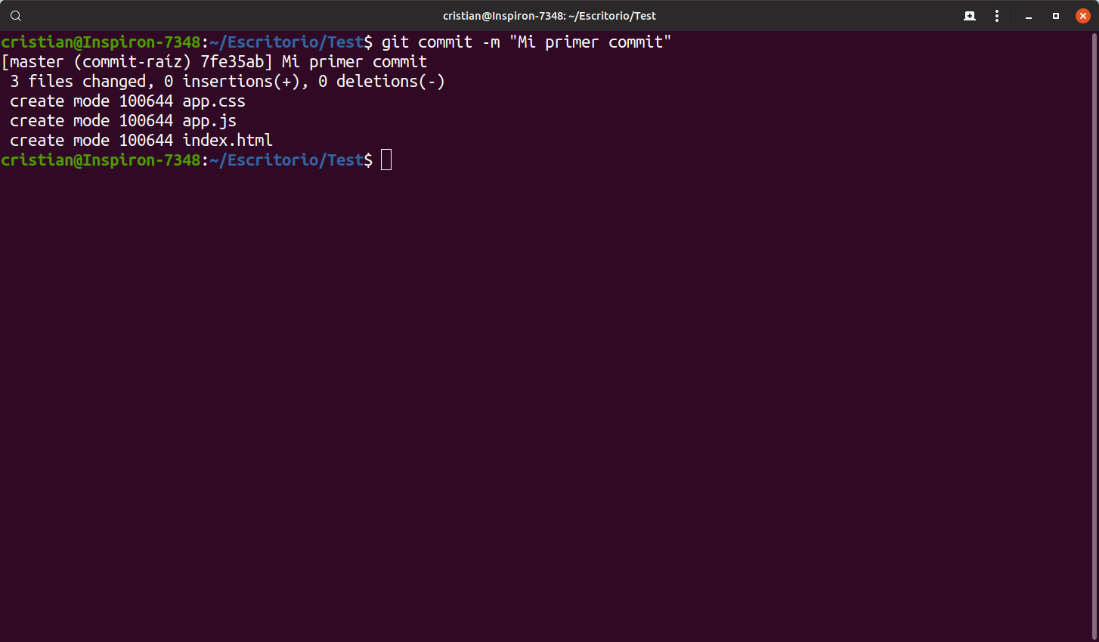


Añadir todos los archivos al repositorio con el comando *git add .*

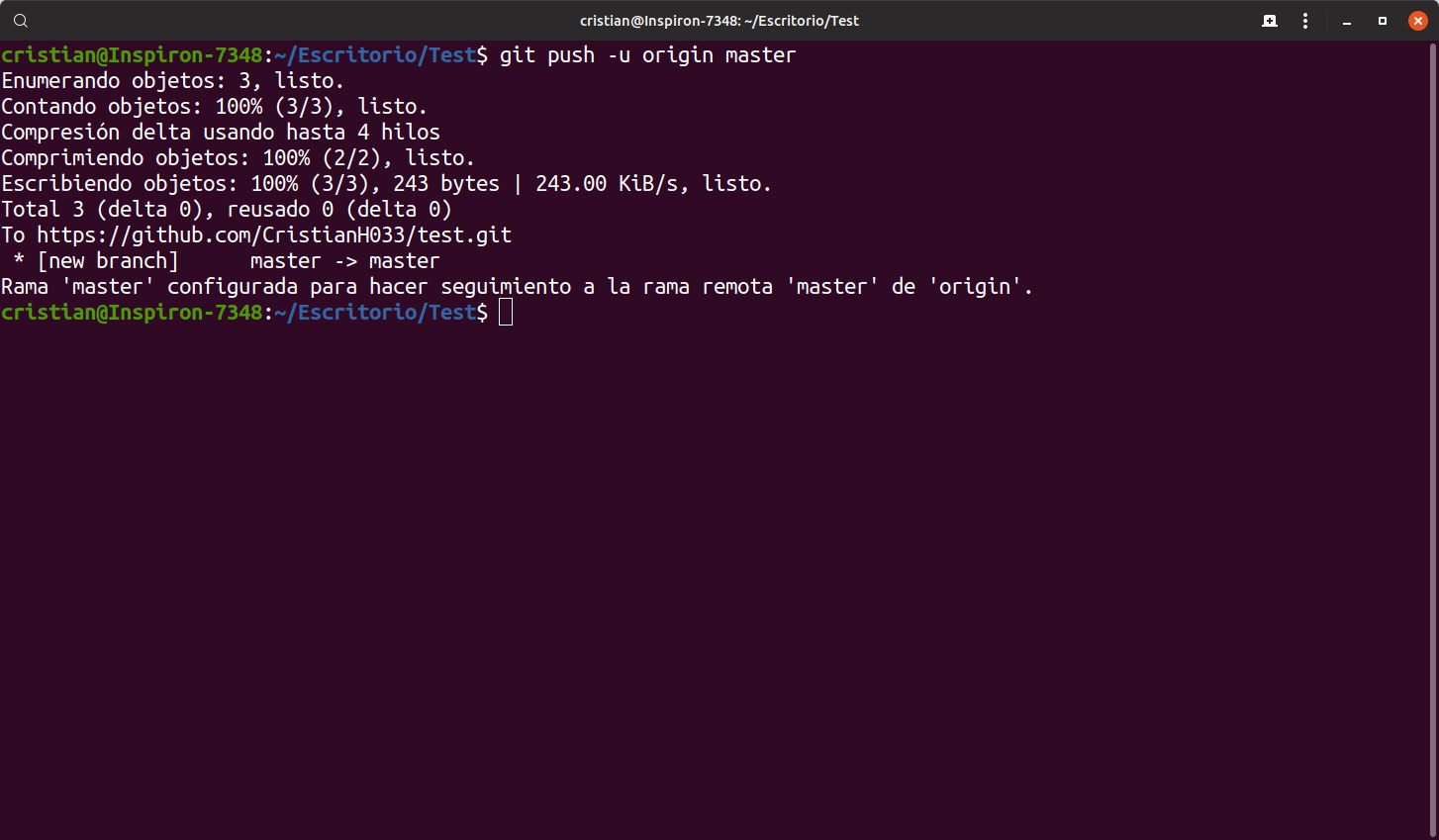
Agregar servidor remoto (github) al repositorio local con el comando *git remote add origin*



Agregar commit al master con el comando *git commit*



Empujar los cambios hacia el servidor (GitHub) con el comando git push



Comprobar que los cambios se reflejen en el Repositorio remoto (GitHub)

