Manual de versionamiento con Herramienta GITHUB

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

GIT. Es un sistema Distribuido de Versiones. Puede trabajar en varios equipos para un mismo repositorio.

GITHUB. Es un plataforma de desarrollo colaborativo para alojar proyectos utilizando del sistema de control de versiones Git.

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.

. 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 asi:

•

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.

•

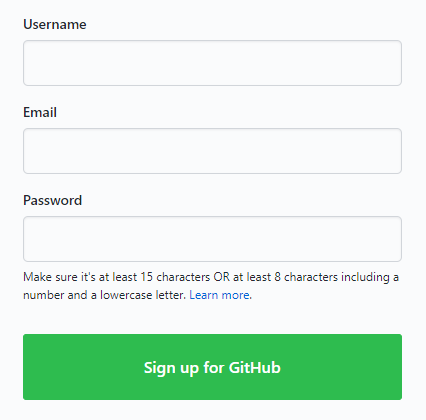
Parche:

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.

1. Instalación.

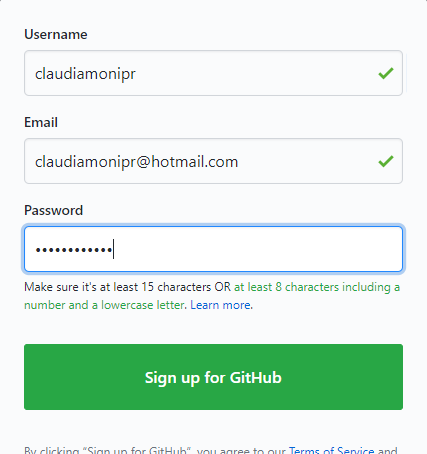
Crear una cuenta https://github.com/



Siempre que se necesite trabajar con Github debo instalar GIT. A continuación se escribe la página oficial.

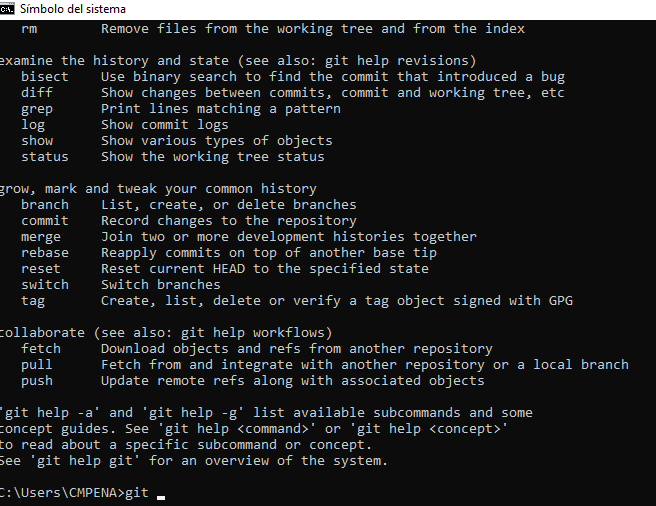
https://git-scm.com/

1. Digitar datos para crear usuario



1. Pantalla Inicial

GIT open source mutiplataforma. Git.-scm-com mirar como instalar en cada plataforma.



Programa de línea de comandos

Git lo que hace es mirar constantemente lor archios que están dentro del repostiroio, coo esta antes como estaba ahora

Github no tengo que instalarlo , es un servicios de repositorio de fuentes.

Github lo compro hace poco Microsoft y l comenzó a integrar en azuredevops. Azuredevops tiene mas herramienta como planecion de sprints, sericios detrás de esto para colaboración entre las empresas.

Devops . devops

Visual Studio Codeves el editor de código . más NO el entorno el desarrollo

Lo representa d

Git trabaja por debajo de cada editor. Independiente del software desarrollo. Los comandos son universales. GitFii

Instalación. Git no maneja respotiorio remotos. Pero se debe realizar la configuración al servidor. Tipo de autenticación Github esuna especie de Onedrive

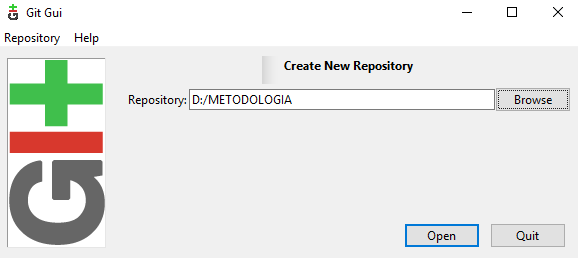
Gitdag es otro programa q utiliza otro programa . programa para gestión un repositroio

Una vez esta instalado GIT, aparecerá la siguiente pantalla. Con las siguientes Opciones

1. Creación nuevo Repositorio.
2. Clonar Repositorio Existente.
3. Abrir un Repositorio Existente.



1. Creación de un repositorio.

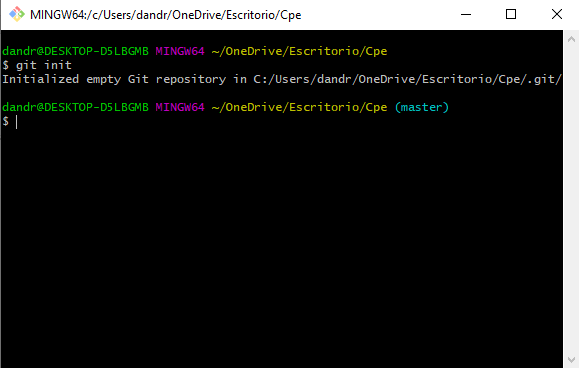


**Dispositivos de audio**

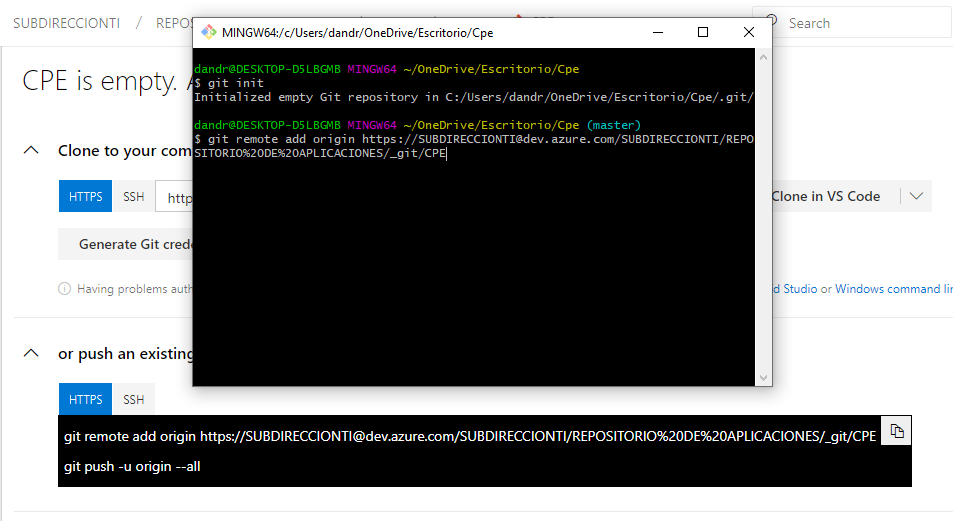
Versionar un Branch.

1. Creación repositorio desde ventana de comandos
2. Inicialización del repositorio GIT

Git init



1. Definir la url del repositorio remoto



1. Agregar los archivos al repositorio

Git add -A

1. Crear el primer commit

git commit -m “Initial commit”

1. Subir los cambios

Git push origin master