TIC

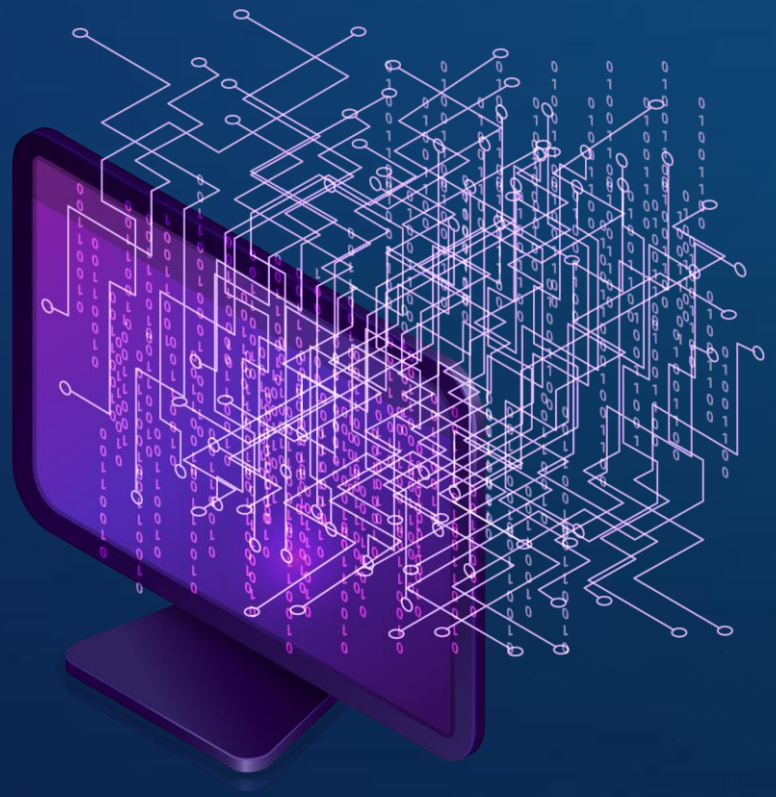
TALENTO TECH

Sesión 6

Programación Nivel Básico

UNIVERSIDAD LIBRE

etraining



Sesión 6:

Trabajando con Git

Configuración inicial de Git, creación de un repositorio, control de versiones.

Objetivos de la sesión

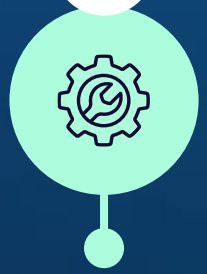
Al finalizar esta sesión estarás en capacidad de:

01

Comprender el funcionamiento básico de Git

Entender los conceptos fundamentales de Git, como el repositorio, los commits, las ramas, entre otros.

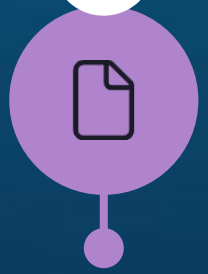
02



Configurar Git en tu sistema

Instalar y configurar Git en tu computadora para comenzar a utilizarlo de manera efectiva.

03



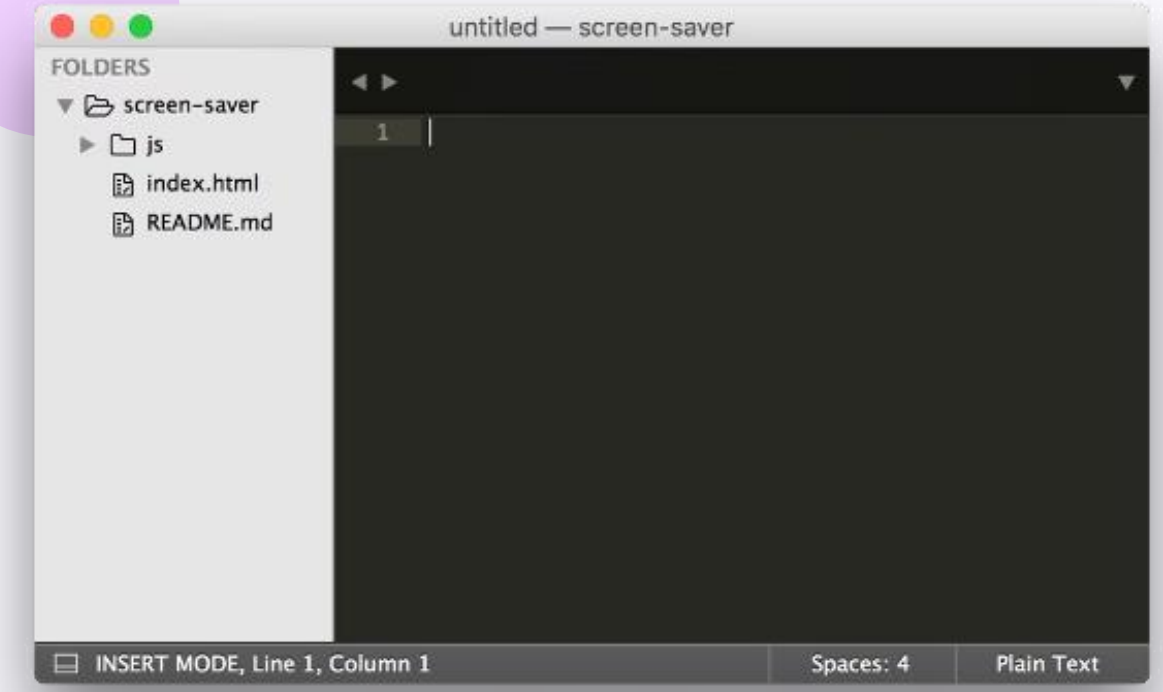
Crear un repositorio Git local

Aprender a crear un repositorio Git para un nuevo proyecto y comenzar a realizar un seguimiento de los cambios en los archivos.

04

Realizar los primeros commits

Aprender a realizar commits, que son esencialmente "instantáneas" de los cambios realizados en el proyecto.



Introducción a Git

Git es un sistema de control de versiones distribuido, diseñado para el seguimiento de cambios en archivos de código fuente durante el desarrollo de software. En resumen, Git te permite guardar las diferentes versiones de tu proyecto, trabajar en colaboración con otros desarrolladores y revertir los cambios en cualquier momento, entre otras funcionalidades.

Definición de Git

Control de versiones

Git es un sistema de control de versiones que permite realizar un seguimiento de los cambios en los archivos de un proyecto a lo largo del tiempo. Esto permite revertir a versiones anteriores, comparar cambios, colaborar con otros desarrolladores y más.

Distribuido

Git es un sistema de control de versiones distribuido, lo que significa que cada desarrollador tiene una copia completa del historial del proyecto en su computadora. Esto permite trabajar sin conexión y realizar cambios sin necesidad de una conexión centralizada.

Definición de Git

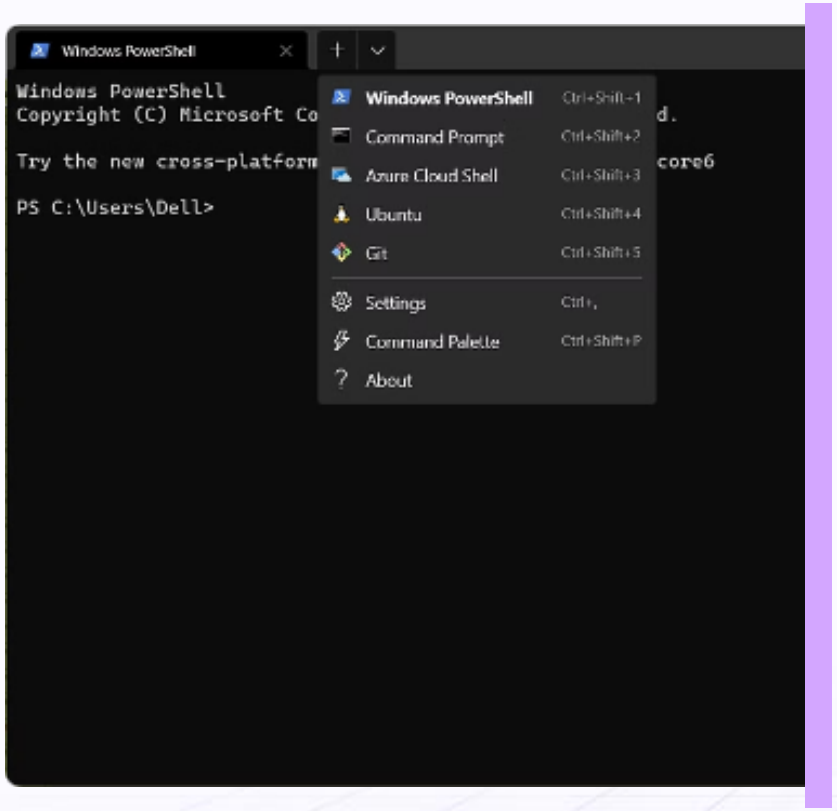
Repositorio

Un repositorio Git es un lugar donde se almacenan todos los archivos del proyecto, junto con su historial de cambios. Un repositorio puede ser local (en tu computadora) o remoto (en un servidor).

Commit

Un commit es una instantánea de los cambios realizados en los archivos de un proyecto. Cada commit tiene un identificador único y un mensaje que describe los cambios realizados.

Instalación y configuración básica de Git



1

Descarga e instalación

Descargue el instalador de Git desde el sitio web oficial de Git (https://git-scm.com/) y ejecute el instalador, siguiendo las instrucciones en pantalla. Es importante tener en cuenta que la configuración predeterminada de Git es aceptable para la mayoría de los usuarios.

2

Configuración inicial

Abra un terminal o línea de comandos y configure su nombre y correo electrónico de usuario en Git. Esto permitirá que se registre correctamente su autoría de los commits que realice.

3

Verificar la instalación

Verifique que Git esté correctamente instalado ejecutando el comando `git --version` en la terminal. Si la instalación es exitosa, se mostrará la versión de Git instalada en su sistema.

Comandos Git fundamentales

Comando	Descripción
git init	Inicializa un repositorio Git en un directorio.
git add	Agrega archivos al área de preparación para un commit.
git commit	Crea un nuevo commit con los archivos preparados.
git status	Muestra el estado actual del repositorio.
git log	Muestra el historial de commits.
git branch	Crea, lista o elimina ramas.
git checkout	Cambia de rama.
git merge	Combina ramas.
git pull	Descargue cambios de un repositorio remoto y los combina con la rama actual.
git push	Sube los cambios locales al repositorio remoto.

Repositorios locales y remotos

Repositorio local

Es una copia del repositorio Git en su computadora. Es donde realiza los cambios en los archivos y crea los commits.

Repositorio remoto

Es una copia del repositorio Git en un servidor. Es donde se almacena la copia principal del proyecto y donde los demás desarrolladores pueden acceder a los cambios.

Servicios de alojamiento

Existen servicios como GitHub, GitLab y Bitbucket que brindan alojamiento de repositorios remotos para proyectos. Estos servicios ofrecen funcionalidades adicionales como la gestión de problemas, la colaboración y la integración continua.

Flujo de trabajo con Git

1

Clone del repositorio

Crear una copia local del repositorio remoto.

2

Realizar cambios

Realizar cambios en los archivos del proyecto.

3

Agregar cambios al área de preparación

Preparar los cambios para ser incluidos en el próximo commit.

4

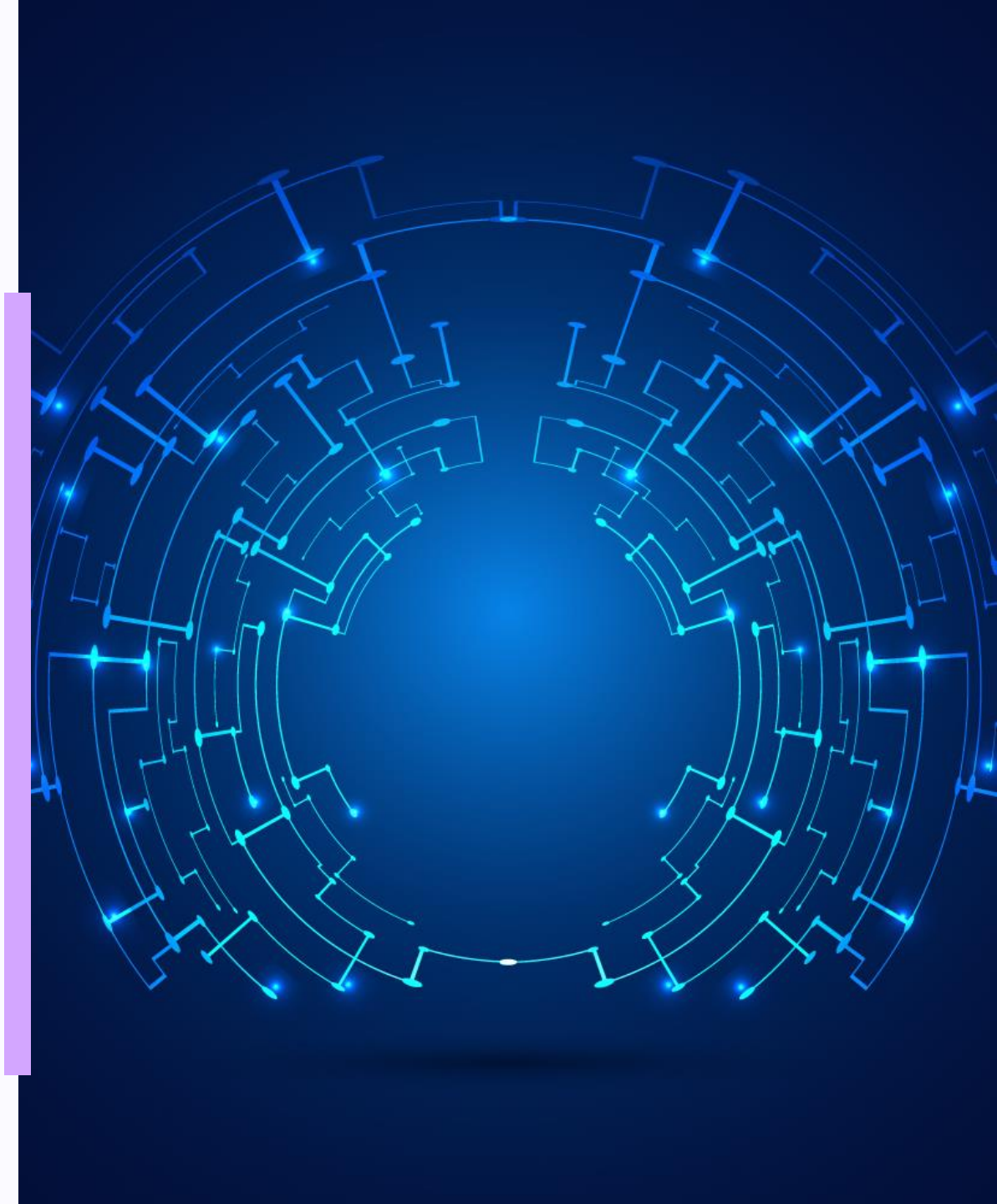
Crear un commit

Realizar un commit para guardar los cambios en el repositorio local.

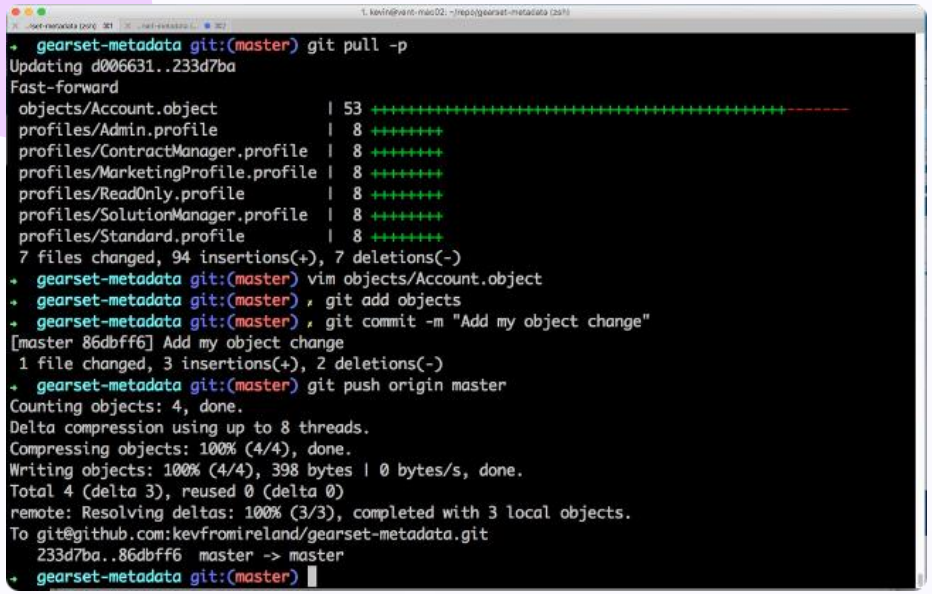
5

Subir cambios al repositorio remoto

Enviar los cambios locales al repositorio remoto.



Ejemplo de uso de Git



Crear un nuevo archivo

En el terminal, cree un nuevo archivo llamado "my_file.txt".

Agregar el archivo al área de preparación

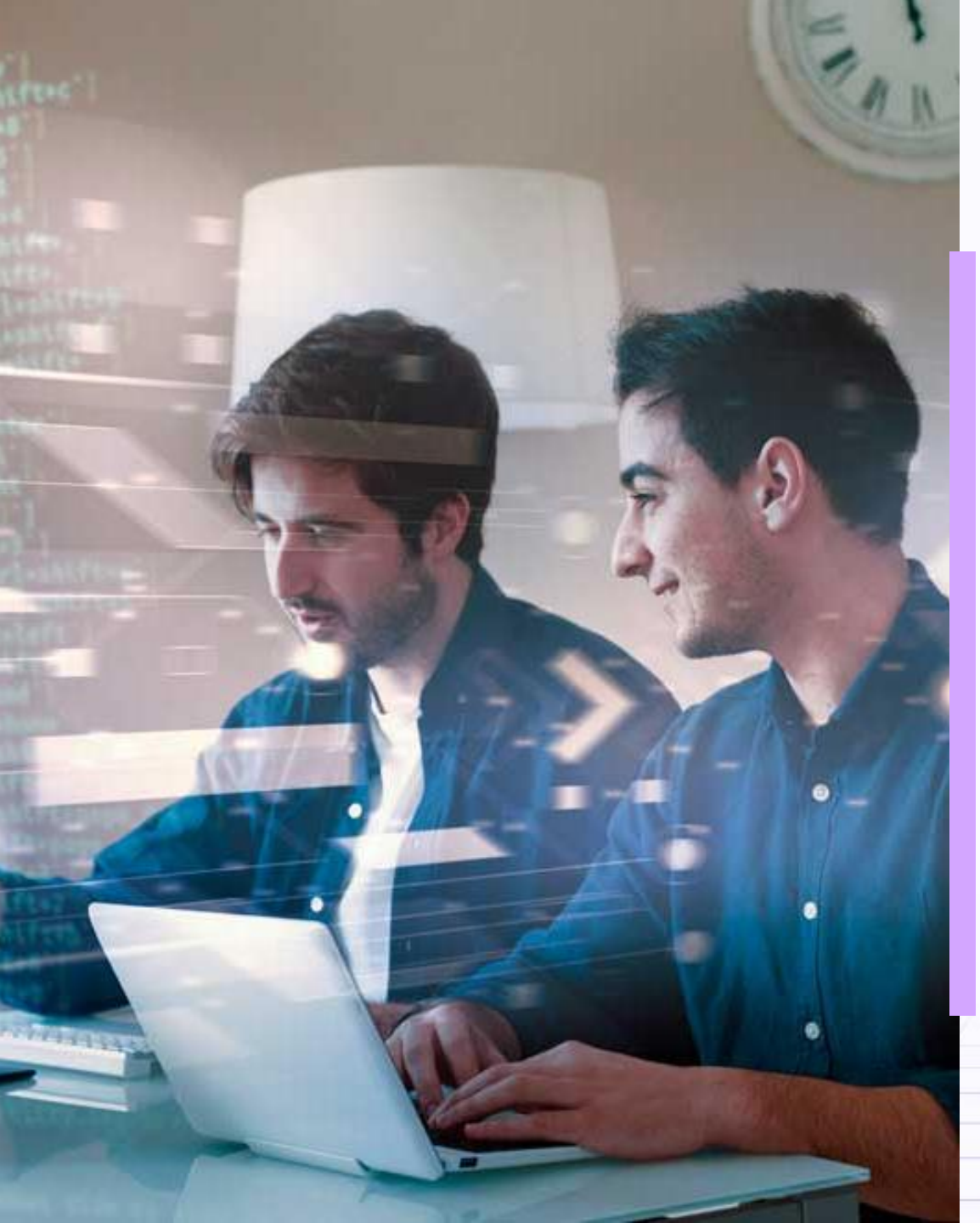
Ejecute el comando "git add my_file.txt" para agregar el archivo al área de preparación.

Realizar un commit

Ejecute el comando "git commit -m "Mensaje del commit"" para realizar un commit con un mensaje descriptivo.

Subir cambios al repositorio remoto

Ejecute el comando "git push origin master" para subir los cambios locales al repositorio remoto.



Introducción al control de versiones con Git

Git es un sistema de control de versiones distribuido que se ha convertido en una herramienta esencial para desarrolladores de software de todo el mundo. Git permite a los equipos trabajar de manera colaborativa y eficiente en proyectos, manteniendo un historial completo de los cambios realizados en el código fuente.

Definiciones básicas de Git

Antes de comenzar a trabajar con Git, es importante comprender algunos conceptos clave. Un repositorio Git es una carpeta que contiene todos los archivos de un proyecto y el historial de los cambios realizados. Un commit es una instantánea del repositorio en un momento determinado. Una rama (branch) es una copia independiente del repositorio, que permite a los desarrolladores trabajar en nuevas funcionalidades sin afectar el código principal.

Definiciones básicas de Git

Commit

Un commit representa un punto específico en la historia del proyecto. Cada commit está asociado a un mensaje que describe los cambios realizados.

Branch

Las ramas son como líneas paralelas del repositorio, que se pueden fusionar en el futuro. Esto permite trabajar en diferentes funcionalidades de forma independiente.

Merge

La fusión (merge) combina los cambios realizados en una rama con la rama principal. El proceso de merge puede generar conflictos si hay cambios que se solapan.

Creación de un repositorio Git

Para crear un repositorio Git, primero debes iniciar Git en tu sistema. Luego, puedes navegar a la carpeta del proyecto en la terminal y ejecutar el comando `git init`. Esto creará una nueva carpeta `.git` que contiene toda la información necesaria para gestionar el historial del proyecto.

1

Paso 1

Inicia Git en tu sistema.

2

Paso 2

Navega a la carpeta del proyecto en la terminal.

3

Paso 3

Ejecuta el comando `git init`.

4

Paso 4

Git crea una nueva carpeta `.git` para gestionar el historial del proyecto.

Comandos Git fundamentales

Git ofrece una variedad de comandos para gestionar el historial de versiones, realizar cambios y colaborar con otros desarrolladores. Los comandos más comunes incluyen:

Comando	Descripción
`git add`	Agrega archivos al área de preparación (staging area).
`git commit`	Crea un nuevo commit con los cambios preparados.
`git push`	Envía los commits locales al repositorio remoto.
`git pull`	Descargue los cambios del repositorio remoto e intégrelos en el repositorio local.
`git status`	Muestra el estado actual del repositorio.
`git log`	Muestra el historial de commits.

Trabajando con ramas

Las ramas son una de las características más poderosas de Git. Permiten a los desarrolladores trabajar en nuevas funcionalidades de forma independiente del código principal. Para crear una nueva rama, se utiliza el comando `git branch nombre_de_la_rama`. Luego, puedes cambiar a la nueva rama con el comando `git checkout nombre_de_la_rama`.

1

Paso 1

Crea una nueva rama.

2

Paso 2

Cambia a la nueva rama.

3

Paso 3

Realiza los cambios necesarios en la nueva rama.

4

Paso 4

Fusiona la rama con la rama principal.

Fusionando cambios

La fusión (merge) combina los cambios de una rama con la rama principal. Para fusionar una rama, primero debes cambiar a la rama principal con el comando `git checkout main` (o el nombre de la rama principal). Luego, puedes ejecutar el comando `git merge nombre_de_la_rama` para combinar los cambios de la rama con la rama principal.

Paso 1

Cambia a la rama principal.

Paso 2

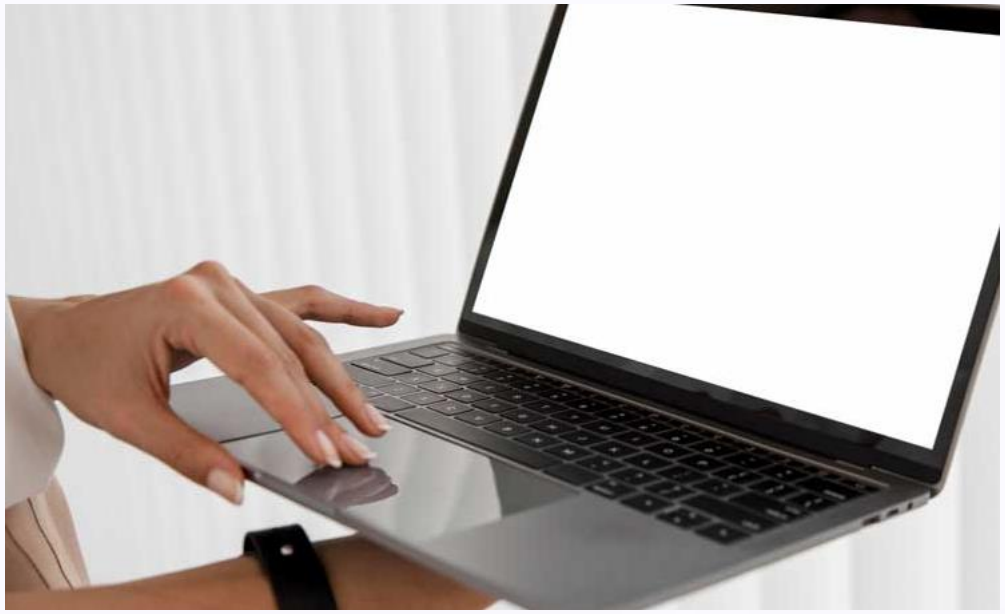
Fusiona la rama con la rama principal.

Paso 3

Resuelve cualquier conflicto.

Paso 4

Realiza un nuevo commit para integrar los cambios fusionados.



Resolviendo conflictos durante el merge

Si hay cambios que se solapan en diferentes ramas, se puede producir un conflicto durante la fusión. Git te mostrará los archivos que tienen conflictos y te pedirá que los resuelvas. Puedes editar los archivos manualmente para elegir las versiones correctas y luego utilizar el comando `git add` para añadir los archivos modificados.

Conflictos

Los conflictos surgen cuando los cambios en diferentes ramas se solapan.

Resolución

Git te ayuda a identificar y resolver los conflictos mediante un proceso sencillo.

Soluciones

Puedes elegir las versiones correctas de los archivos modificados o combinar los cambios de diferentes ramas.

¿Qué es GitHub?

GitHub se describe a sí mismo como “donde el mundo construye software”, y es bastante acertado.
GitHub es una plataforma de desarrollo avanzada donde las personas pueden crear y mantener software.
Proporciona hospedaje de Internet utilizando Git, un software que rastrea los cambios dentro de los archivos.

Además de proporcionar alojamiento, GitHub también ofrece muchas otras funciones para los usuarios. Puedes usar GitHub para colaborar en proyectos, resolver problemas, administrar el código fuente y comunicarte con personas de todo el mundo.

Con más de 83 millones de desarrolladores, GitHub es un recurso excelente y una plataforma para compartir funcional y comunicativa.

¿Para qué se usa GitHub?

Los usos de GitHub encajan en seis categorías: programación, colaboración, desarrollo, automatización, seguridad y comunidad.

Como servicio de alojamiento de Internet, GitHub se puede utilizar para alojar todos tus proyectos de forma gratuita con repositorios públicos y privados ilimitados. Un repositorio es una estructura de datos que almacena metadatos para un conjunto de archivos o estructuras de directorios.

Buenas prácticas y flujos de trabajo con Git

Hay varias buenas prácticas que se pueden seguir al trabajar con Git, que ayudan a evitar problemas y mejorar la colaboración. Algunas prácticas comunes incluyen:

1 **Commits pequeños**

Cada commit debe tener un propósito específico y claro.
Los commits grandes dificultan el seguimiento de los cambios.

2 **Mensajes descriptivos**

Los mensajes de commit deben ser concisos y descriptivos, explicando claramente los cambios realizados.

3 **Flujo de trabajo de ramificación**

Utiliza un flujo de trabajo de ramificación (branching workflow) para gestionar las nuevas funcionalidades y las correcciones de errores.

4 **Revisiones de código**

Las revisiones de código ayudan a identificar errores y mejorar la calidad del código.

Recursos y material de apoyo

Para obtener más información sobre Git, existen recursos valiosos disponibles en línea:

- **Documentación oficial de Git:** https://git-scm.com/doc
- **Tutoriales interactivos de Git:** https://learngitbranching.js.org/
- **Libros y cursos sobre Git:** Hay numerosos libros y cursos disponibles para aprender Git en profundidad.

Conclusiones y recomendaciones

Git es una herramienta fundamental para el desarrollo de software, que facilita la colaboración, la gestión de versiones y el seguimiento de los cambios. Se recomienda dedicarle tiempo a comprender los conceptos básicos de Git y practicar su uso para aprovechar al máximo sus funcionalidades. Hay una gran comunidad online que puede ser de ayuda en caso de necesitar ayuda o tener preguntas.

Ejercicios de práctica

¡Gracias

por ser parte de esta experiencia de aprendizaje!