Introducción a Git

Git es un sistema de control de versiones distribuido que permite a los desarrolladores rastrear los cambios en el código fuente durante el desarrollo de software. Fue creado por Linus Torvalds en 2005 para el desarrollo del kernel de Linux. Git es conocido por su velocidad, integridad de los datos y soporte para flujos de trabajo no lineales. A continuación, se presentan algunos conceptos clave de Git:

- **Repositorio**: Un directorio donde Git almacena todos los cambios y el historial del proyecto.
- **Commit**: Una instantánea del estado del proyecto en un momento específico.
- **Branch (Rama)**: Una versión separada del proyecto, que permite desarrollar nuevas características sin afectar la versión principal.

Comandos Básicos de Git

1. **Inicializar un Repositorio**

   git init

   Este comando inicializa un nuevo repositorio de Git en el directorio actual.

2. **Clonar un Repositorio**

   git clone <url-del-repositorio>

   Este comando crea una copia local de un repositorio remoto.

3. **Agregar Cambios al Área de Staging**

   git add <archivo>

   git add .

   Estos comandos agregan cambios al área de staging, preparándolos para un commit.

   git add <archivo> agrega el archivo mencionado, mientras que git add . agrega todos los archivos que hayan sido modificados.

4. **Hacer un Commit**

   git commit -m "Mensaje del commit"

   Este comando guarda los cambios del área de staging en el repositorio con un mensaje descriptivo.

5. **Ver el Estado del Repositorio**

   git status

Este comando muestra el estado de los archivos en el repositorio (modificados, no rastreados, etc.).

6. **Ver el Historial de Commits**

   git log

   Este comando muestra el historial de commits en el repositorio.

Trabajando con Repositorios en GitHub

GitHub es una plataforma de alojamiento de repositorios Git que proporciona herramientas de colaboración y gestión de proyectos. A continuación, se explica cómo trabajar con repositorios en GitHub:

1. **Crear un Repositorio en GitHub**
   - Inicia sesión en GitHub.
   - Haz clic en "New repository".
   - Completa los detalles del repositorio y haz clic en "Create repository".
2. **Conectar un Repositorio Local a GitHub**

   git remote add origin <url-del-repositorio-en-github>

   Este comando conecta el repositorio local con el repositorio remoto en GitHub.
3. **Enviar Cambios al Repositorio Remoto**

   git push origin master

   Este comando envía los commits locales al repositorio remoto en la rama master.
4. **Obtener Cambios del Repositorio Remoto**

   git pull origin master

   Este comando obtiene los últimos cambios del repositorio remoto y los fusiona con la rama local.

Branches (Ramas)

Las ramas en Git permiten desarrollar características, corregir errores o experimentar de manera aislada del resto del proyecto.

1. **Crear una Nueva Rama**

   git branch <nombre-de-la-rama>

Este comando crea una nueva rama.

2. **Cambiar de Rama**

   git checkout <nombre-de-la-rama>

   Este comando cambia a la rama especificada.

3. **Crear y Cambiar a una Nueva Rama**

   git checkout -b <nombre-de-la-rama>

   Este comando crea una nueva rama y cambia a ella de inmediato.

4. **Ver Todas las Ramas**

   git branch

   Este comando muestra todas las ramas en el repositorio.

Merge (Fusión) y Conflicts (Conflictos)

La fusión (merge) en Git combina cambios de diferentes ramas. Los conflictos ocurren cuando hay cambios contradictorios en las ramas que se están fusionando.

1. **Fusionar una Rama en la Rama Actual**

   git merge <nombre-de-la-rama>

Este comando fusiona la rama especificada en la rama actual.

2. **Resolver Conflictos**

   - **Paso 1**: Durante una fusión, Git intentará fusionar automáticamente los cambios. Si no puede, marcará los conflictos en los archivos afectados.
   - **Paso 2**: Abre los archivos con conflictos y resuelve los conflictos manualmente. Los conflictos estarán marcados con <<<<<<, ======, y >>>>>>.
   - **Paso 3**: Una vez resueltos los conflictos, agrega los archivos al área de staging:

     git add <archivo>

   - **Paso 4**: Completa la fusión con un commit:

     git commit -m "Resuelto conflicto de fusión"