Investigación de Git & GitHub

Introducción a Git

¿Qué es Git?

Git es un sistema de control de versiones distribuido diseñado para seguir y registrar los cambios en archivos. Es útil para equipos de trabajo que colaboran simultáneamente en los mismos archivos. Lo creó Linus Torvalds en 2005 para gestionar el desarrollo del núcleo de Linux.

A continuación, se explica de manera simple los conceptos clave sobre cómo funciona Git:

1. Repositorio: Almacena el código y su historial de versiones, ya sea en una computadora o en un servidor online.
2. Commits: Guardan el estado del proyecto en momentos específicos y cada uno tiene un identificador único.
3. Ramas (Branches): Permiten trabajar en diferentes versiones del proyecto simultáneamente y desarrollar nuevas características sin afectar la rama principal.
4. Fusión (Merge): Une los cambios de diferentes ramas y resuelve cualquier conflicto que surja.
5. Staging Area (Área de Preparación): Permite revisar y seleccionar qué cambios se incluirán en el próximo commit.
6. Sincronización: Los cambios locales se sincronizan con repositorios remotos usando comandos como `push` (para enviar cambios) y `pull` (para recibir cambios).

¿Qué es GitHub?

GitHub es una plataforma de alojamiento, propiedad de Microsoft, que ofrece a los desarrolladores la posibilidad de crear repositorios de código y guardarlos en la nube de forma segura, usando el sistema de control de versiones llamado Git.

Ventajas de usar GitHub:

- Control de versiones: Historial completo de cambios con recuperación fácil.
- Trabajo en equipo: Colaboración fluida con otros desarrolladores.
- Seguridad: Protección contra errores y pérdidas con copias de seguridad.
- Flexibilidad: Se adapta a diversos proyectos y flujos de trabajo.
- Descubrimiento de código: Reutilización de código de otros desarrolladores.

Diferencias entre Git & GitHub

Git es una herramienta que permite establecer un control de versiones, permitiendo rastrear y gestionar cambios en el código fuente de un proyecto, mientras que GitHub es una plataforma en línea con funciones que facilitan el uso de Git, la colaboración en tiempo real y el almacenamiento en la nube.

Comandos básicos

Comando	Descripción
git config	Configura opciones globales o locales para Git, como el nombre de usuario y el correo electrónico.
git init	Crea un nuevo repositorio Git
git clone [url]	Clona un proyecto/repositorio remoto en una máquina local usando la URL proporcionada.
git add	Agrega archivos al área de preparación para el siguiente commit
git commit -m "mensaje"	Guarda los cambios en el historial de versiones con un mensaje descriptivo
git branch	Enumera las existentes
git branch [nombre-rama]	Crea nuevas ramas
git checkout [rama]	Cambia a una rama diferente
git merge [rama]	Combina los cambios de diferentes ramas
git push [alias] [rama]	Envía los cambios al repositorio remoto
git pull	Descarga los cambios del repositorio remoto

Trabajando con repositorios en GitHub (branches, merge y conflicts)

1. Ramas (Branches): Las ramas permiten trabajar en diferentes versiones de un proyecto simultáneamente sin afectar la versión principal. Incluso se puede cambiar entre ramas y trabajar en diferentes proyectos sin que interfieran entre sí.

- Creación de una rama:
  Para crear una rama en GitHub, puedes usar el siguiente comando:

  `git branch <nombre-de-la-rama>`

  Por ejemplo, si se quiere crear una rama llamada “versionDos”:

  git branch versionDos

- Cambiar a una rama:
  Para cambiar a una rama, puedes usar el siguiente comando:

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

  Por ejemplo, si se quiere cambiar a la rama “versionDos”:

  git checkout versionDos

2. Fusionar ramas (Merge): Cuando se haya terminado de trabajar en una rama, dicha rama puede fusionarse con la rama principal. Esto integrará los cambios de la rama secundaria en la rama principal.

Para fusionar una rama en GitHub, puedes usar el siguiente comando:

```
git merge <nombre-de-la-rama>
```

Por ejemplo, para fusionar la rama "versionDos" con la rama principal, se puede usar el siguiente comando:

```
git merge versionDos
```

3. Conflictos de fusión (Conflicts): Si dos ramas realizan cambios en las mismas líneas de código, se producirá un conflicto en el momento de fusionar las ramas. Se tendrá que resolver el conflicto manualmente antes de poder fusionar las ramas.

Para resolver un conflicto de fusión, GitHub mostrará las líneas de código que están en conflicto, y posteriormente se tendrá que editar manualmente las líneas de código para resolver el conflicto. En algunos IDEs como Visual Studio Code es un poco más fácil realizar esta tarea. Cuando se resuelva el conflicto, se puede volver a intentar fusionar las ramas.