Universidad de San Carlos de Guatemala
Facultad de ingeniería
Escuela de Estudios de Post-Grado
Maestría en ingeniería para la industria con especialización en ciencias de la computación

GIT

Que es GIT

Es un sistema de control de versiones que fue pensado en la eficiencia sobre el mantenimiento en las aplicaciones a la hora del mantenimiento, en el caso de un gran numero de archivos de código fuente.
Almacena la información en un conjunto de archivos, no permite cambios, corrupción o alteraciones sin que GIT lo sepa o lo registre. Lo malo es que todos los recursos deben de ser de forma loca.

Control de versiones con GIT

Registra los cambios realizados sobre un archivo en toda la línea del tiempo, así se puede registrar la evolución de un proyecto.

Estados de un archivo en GIT

En la actualidad cuenta con tres estados del archivo.

1. Staged
2. Modified
3. Committed

Como se configura un repositorio

- Repositorio GIT: es un almacenamiento virtual del proyecto permitiendo guardar versiones del código a las que pueden acceder cuando lo necesitas.
  Colocar el comando git init.

Comandos en GIT

- **git init:**
  Esto crea un subdirectorio nuevo llamado .git, el cual contiene todos los archivos necesarios del repositorio – un esqueleto de un repositorio de Git. Todavía no hay nada en tu proyecto que esté bajo seguimiento.

- git fetch:
  Descarga los cambios realizados en el repositorio remoto.

- git merge <nombre_rama>:
  Impacta en la rama en la que te encuentras parado, los cambios realizados en la rama “nombre_rama”.

- git pull: Unifica los comandos fetch y merge en un único comando.

- git commit -m "<mensaje>":
- Confirma los cambios realizados. El “mensaje” generalmente se usa para asociar al commit una breve descripción de los cambios realizados.
- git push origin <nombre_rama>:
  Sube la rama “nombre_rama” al servidor remoto.

- git status: Muestra el estado actual de la rama, como los cambios que hay sin commitear.

- git add <nombre_archivo>: Comienza a trackear el archivo “nombre_archivo”.

- git checkout -b <nombre_rama_nueva>: Crea una rama a partir de la que te encuentres parado con el nombre “nombre_rama_nueva”, y luego salta sobre la rama nueva, por lo que quedas parado en esta última.

- git checkout -t origin/<nombre_rama>: Si existe una rama remota de nombre “nombre_rama”, al ejecutar este comando se crea una rama local con el nombre “nombre_rama” para hacer un seguimiento de la rama remota con el mismo nombre.

- git branch: Lista todas las ramas locales.

- git push origin <nombre_rama>: Commitea los cambios desde el branch local origin al branch “nombre_rama”.

- git remote prune origin: Actualiza tu repositorio remoto en caso de que algún otro desarrollador haya eliminado alguna rama remota.

- git reset --hard HEAD: Elimina los cambios realizados que aún no se hayan hecho commit.