Universidad Autónoma de Baja California

Facultad de Ingeniería, Arquitectura y Diseño

PARADIGMAS DE PROGRAMACIÓN

Taller 1. Github y Git

Ingeniero en Software y Tecnologías Emergentes

Brayan Ivan Perez Ventura

372781

16 de Febrero del 2024 en Ensenada, Baja California.

INTRODUCCIÓN

Cuando se trata de edición de video, fotografías, guardar archivos, etc. Es necesario siempre mantener un backup y un control de versiones. Para el presente reporte, se abordará Git, un sistema de control de versiones que, como su nombre lo indica, lleva un control de versiones de archivos y de carpetas, guardándose con ejecución de comandos mediante CMD o Terminal según tu sistema operativo.

Así mismo, también existe la página Github, donde puedes subir dicho control de versiones a la nube como una forma de backup o para trabajar en equipo con diferentes personas.

DESARROLLO

Para comenzar, es necesario instalar **Git**, que lo podrás encontrar en su página oficial (https://git-scm.com). Para el caso de Windows, podrás un instalador que te incluye una nueva terminal llamada **Git Bash** en la cual, se integran diferentes nuevos comandos para el control de versiones.

En el caso de Linux, es necesario conocer qué versión estás y realizar la instalación mediante un comando en la terminal (lo ideal). He de agregar que la terminal **Git Bash** integra diferentes comandos de Linux, por lo tanto, los comandos se pueden aplicar para ambos sistemas operativos.

Para poder comenzar, colocaré los diferentes comandos que existen y puedes utilizar para modificar o moverte entre carpetas desde git bash:

- **pwd:** Conocer la dirección en la cual estoy actualmente.
- **cd:** Para poder moverse entre carpetas.
- **ls:** Desplegar los archivos que contiene dicha carpeta.
- **mkdir:** Crear una carpeta en el directorio donde estás.
- **rm -r:** Para eliminar la carpeta o archivo creado.
- **nano:** Editor de texto en terminal.
- **cat:** Ver el contenido del archivo.

Una vez conocido estos comandos, ahora puedes comenzar con **Git:**

- **git init:** Inicializa un repositorio en la carpeta en la que te encuentras.
- **git add:** Agrega el archivo para ser guardado en el próximo commit.
- **git commit -m:** Realiza un commit con los archivos que has agregado con **git add.**
- **git status:** Muestra el estado actual del repositorio git, en este se muestran los archivos modificados, agregados y eliminados.
- **git log:** Muestra el historial de commits realizados en el repositorio.
- **git pull:** Obtener los cambios más recientes del repositorio y fusionarlos con tu rama/repositorio actual.

Aunque existen más comandos para git, estos son los más básicos que se deben conocer con detalle.

Ahora, **Git** incluye algunos comandos que ayudan a subir dicho control de versiones a una nube (un repositorio remoto), donde puedes compartir con diferentes personas y permitir que diferentes personas trabajen al mismo tiempo en un mismo proyecto. Estos comandos son:

- **git clone:** Permite clonar un repositorio remoto de **Github**.
- **git push origin:** Sube los cambios locales al repositorio remoto.
- **git pull origin:** Obtiene los cambios más recientes del repositorio remoto y los fusiona/cambia con la rama local.
- **git remote -v:** Muestra la URL del repositorio remoto.
- **git remote add origin:** Agrega un nuevo "origen remoto" al repositorio local.

Para crear un repositorio remoto, puedes crearte una cuenta en **Github** (www.github.com), una vez creada, puedes ir a la parte derecha arriba y buscar donde dice "Crear un nuevo repositorio" y puedes modificar el **nombre, descripción, privacidad entre otros.**

Una vez creado el repositorio, es necesario copiar el link y realizar el comando **git remove add origin** para linkear el repositorio local (si ya tienes uno, sino, debes inicializar uno) y el repositorio remoto.

En caso de que pida contraseña y usuario, será necesario crear una SSH Key desde la terminal (esto se debe realizar para cada computadora en la cual desees iniciar sesión o trabajar). Los comandos son los siguientes:

- **ssh-keygen:** Genera una llave tipo ssh.
- **ls ~/.ssh:** Nos lleva a la dirección de donde se creó dicha llave.
- **cat ~/.ssh/id_rsa.pub:** Lee el contenido de la llave.

Leemos el contenido de la llave y la copiamos, enseguida vamos a la configuración de la cuenta, enseguida, vamos al apartado de **SSH y GPG llaves.** Una vez que estés dentro, presionas donde dice "agregar SSH key". El título se lo agregas como quieras y finalmente agrega la Key en el apartado donde dice "Key". Añades el SSH key y ya deberá dejar subir los cambios al repositorio remoto.

CONCLUSIÓN

Dentro de programación, es indispensable conocer este tipo de herramientas para poder llevar el control de versiones que se lleva dentro de un programa, así mismo, también para poder organizarse cuando trabajas con diferentes personas y conocer qué tipo de cambios se realizó, cuando y que "bugs" solucionó.

Tanto **Git** y **Github** no solo manejan código como tal, sino que también archivos de cualquier tipo, desde archivos .psd (photoshop), como archivos .c4d (Cinema4D) para animación y fotografía.

Los comandos tanto en Linux como en Windows no cambian tanto si se utiliza la terminal de Git Bash que viene al instalarlo (En Windows) o se pueden agregar al Path para que CMD también tenga dichos comandos.

Aunque existen más comandos y un poco más complejos, estos son los más utilizados y básicos para poder comenzar trabajar con repositorios remotos y llevar un control sobre los cambios que realizas a diferentes archivos.

Así mismo, existen programas de programación que realizan dichos comandos automáticamente, uno de ellos es **Visual Studio Code,** que tiene una herramienta que puedes hasta publicar nuevos repositorios desde el programa tan solo linkeando tu cuenta de github al programa.