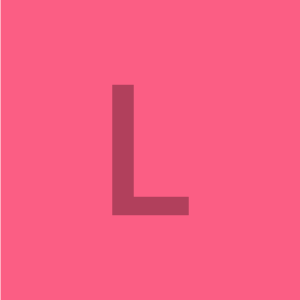
Iniciando un repositorio con Git



Luzdalis Lopez

5 de Diciembre



*Antes de comenzar a desarrollar este contenido, es conveniente que sepa que este tema tiene una secuencia de tres artículos, donde este es el primero. Por si te interesan los otros dos: el segundo es*[*Creando un repositorio en GitHub*](https://www.aluracursos.com/blog/creando-repositorio-remoto-en-github?utm_source=gnarus&utm_medium=timeline)*y el tercero es*[*Clonando un repositorio con Git y GitHub*](https://www.aluracursos.com/blog/clonando-un-repositorio-remoto?utm_source=gnarus&utm_medium=timeline)

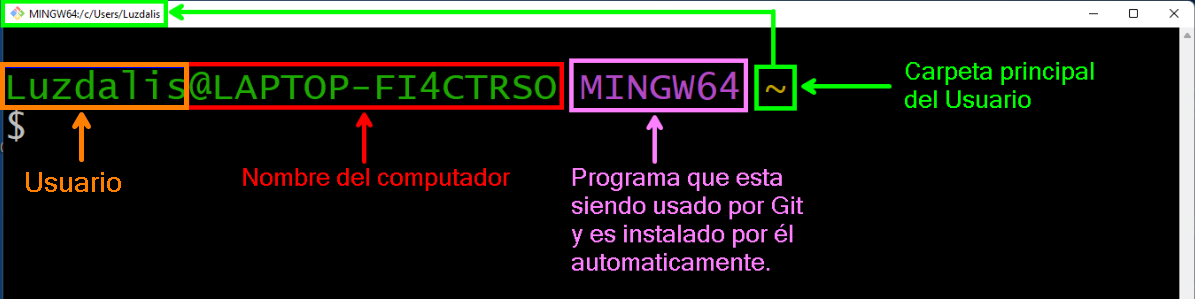
Todo principiante en programación debe tener en cuenta, la necesidad de saber qué es [Git y Github](https://www.aluracursos.com/blog/git-y-github-que-son-y-primeros-pasos), ya que son herramientas que te serán de mucha utilidad a lo largo de tu carrera; por lo que es conveniente que puedas usarlos.

¿Has creado tu primer proyecto y aún no lo has versionado? Si quieres que git comience a rastrear tu código para llevar un registro y control de los cambios realizados al mismo tiempo, pero no te queda muy claro cómo hacerlo; ¡Listo! ¡Este es el artículo ideal que te guiará sobre cómo llevar a cabo este versionado!

**Iniciando un nuevo repositorio en Git (máquina local)**

En primer lugar, si no tiene instalado Git, le recomiendo que [lo descargue e instale en su computadora](https://git-scm.com/downloads); en general se instalará por defecto en tu disco duro C y ahí se basará esta práctica.

Con Git instalado, abra la terminal **Git Bash**, que se iniciará de forma predeterminada en el directorio principal (~) o /c/Users/<User-of-windows-account>/ de su usuario, como se muestra en la siguiente imagen:



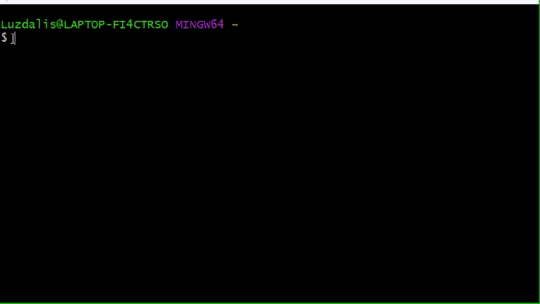
Si desea determinar el directorio actual, escriba pwd en la línea de comando $, como se muestra a continuación:



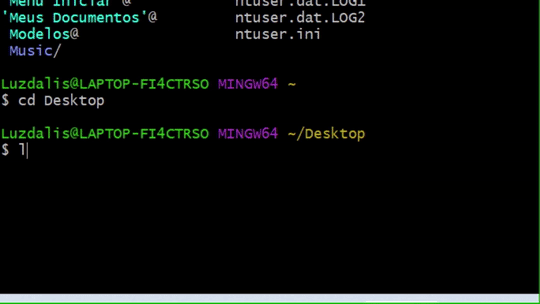
***Notas importantes:***

* *Si vas a usar Git por primera vez, primero deberás configurarlo, estableciendo tu nombre de usuario y dirección de correo electrónico, ya que esta información se usará de manera inalterable en los "commits" que envíes. A continuación les dejo los*[*comandos*](https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Configurando-Git-por-primera-vez)*para configurar estas variables globales:*
* $ git config --global user.name "Tu nombre de Usuario"
* $ git config --global user.email "tu@email.com"
* *Previamente, debes conocer la ruta o la ruta donde se encuentra almacenado tu proyecto****(carpeta o directorio)****, para evitar frustraciones al usar la terminal.*

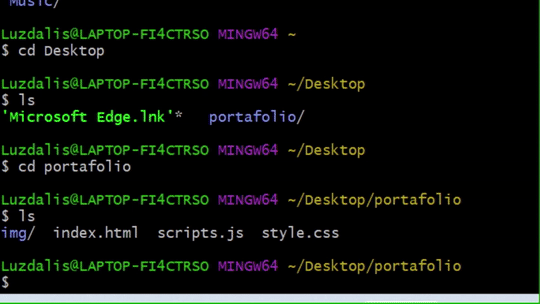
1. Se listarán los archivos del directorio principal y luego se accederá a la carpeta **Desktop**; allí se encuentra el proyecto, en el cual, se quiere iniciar el repositorio:



1. A continuación se mostrarán los archivos de la alfombra de Escritorio y entrarás al **portafolio** del proyecto donde se visualizarán tus archivos:



1. Finalmente, se ejecutará el comando: git init, para comenzar a rastrear git en el repositorio indicado:



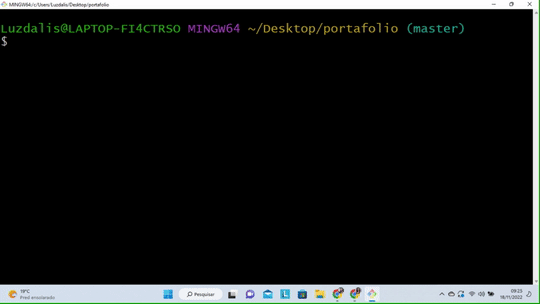
Una vez ejecutado el comando aparecerá el siguiente mensaje indicando que se ha iniciado el repositorio:

`Initialized empty Git repository in C:/Users/Luzdalis/Desktop/portafolio/.git/`

Git mostrará el nombre de la rama principal (rama) que está en uso, que generalmente se llama **master**.

\***Nota:** Para garantizar la compatibilidad futura, se recomienda actualizar el nombre de la rama de **master** a **principal** con el siguiente comando:

$ git branch -M main



**Guardar cambios en el nuevo repositorio local**

Es importante que tengas en cuenta que cada archivo de tu repositorio puede tener uno de los estados: rastreado o no rastreado.

Los archivos rastreados (***tracked files*** en inglés) son los archivos que git está viendo y pueden estar sin modificar (**Unmodified**), modificados (**Modified**) o preparados **(Staged )**.

Por otro lado, los archivos sin seguimiento (***Untracked files***) son todos aquellos archivos que no tienen seguimiento y no se encuentran en el área de preparación.

Para ilustrar estos estados se muestra la siguiente imagen, extraída de la documentación oficial de Git:



Si quieres profundizar un poco más en cómo: [guardar cambios en el Repositorio, puedes encontrarlos aquí](<https://git-scm.com/book/es/v2/Fundamentos-de-Git-Saving>- cambios-en-el-Repositorio).

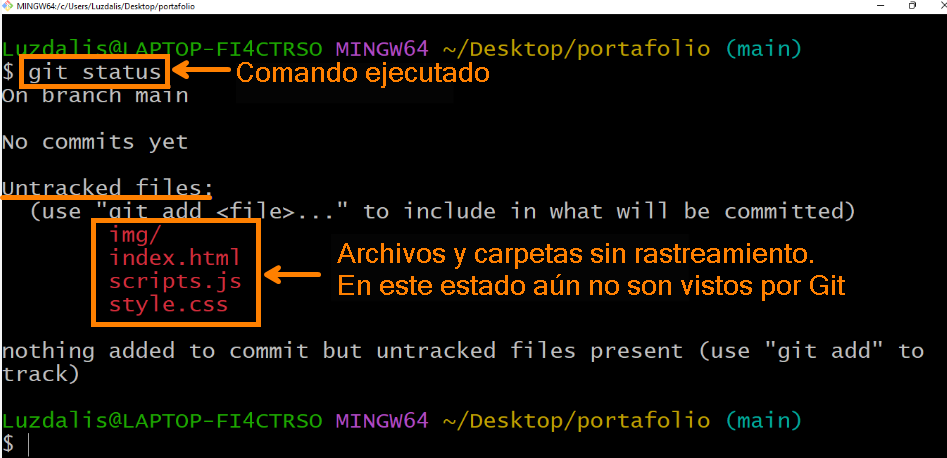
Dentro de este orden de ideas se iniciará el control y registro de versiones del proyecto:

**Primer paso: ver el estado del repositorio**

Para conocer el estado en el que se encuentran los archivos del repositorio utilice el siguiente comando:

* $ git status

Cabe señalar que al iniciar un repositorio; la rama principal **main** está vacía, es decir, no hay archivo, Git no está rastreando ninguna alfombra.

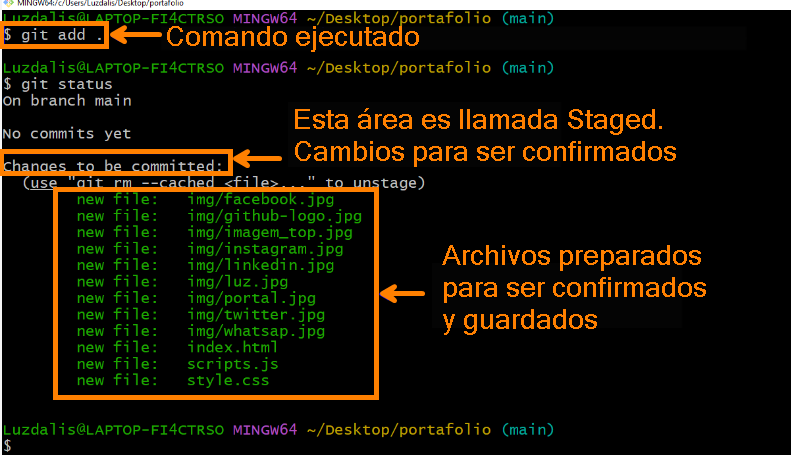


​ **Untracked files** (Archivos sin rastreamiento).

**Segundo paso: agregar archivos para confirmar**

Para comenzar a rastrear los archivos, se ejecutará el comando **git add** para agregar contenido a la próxima confirmación. Se puede utilizar de dos formas:

* $ git add nombre\_del\_archivo /\*\*\*Añade un archivo específico\*\*\*/
* $ git add . /\*\*\*Adiciona todos los archivos de una sola vez.\*\*\*/

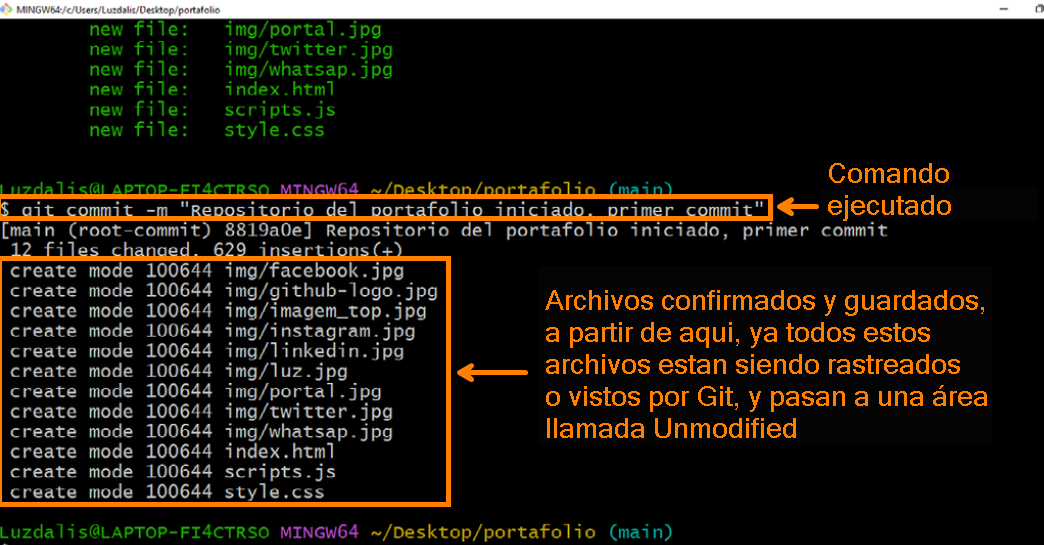


​ **Changes to be committed** (cambios preparados para confirmación).

**Tercer paso: confirmar los archivos**

Para confirmar los cambios, utilice el comando:

$ git commit -m "Mensaje descriptivo del commit"



**Lista de comandos básicos más usados en el versionado de Git**

* **git config:** Esto es lo primero que debe hacer al instalar Git, defina su nombre de usuario y dirección de correo electrónico.

$ git config --global user.name "Tu nombre de Usuario"

$ git config --global user.email "tu@email.com"

* **git init**: Inicia o crea un repositorio.

$ git init

* **git status**: Visualiza el estado del repositorio.

$ git status

* **git add**: Prepara contenido para la próxima confirmación.

$ git add nombre\_del\_archivo

$ git add .

* **git commit**: Guarda contenido actual junto con un mensaje de registro de usuario que describe los cambios.

$ git commit -m "Mensaje descriptivo"

* **git clone**: Clona un repositorio existente.

$ git clone [url]

* **git branch**: Una branch no es mas que un puntero para desplazamiento de un commit a otro, se puede utilizar de varias formas:

$ git branch //Muestra las branches exitentes en un repositorio

$ git branch nombre\_branch //Crea una nueva branch

$ git branch -M nombre\_branch //para renombrar la branch actual

* **git checkout**: Para cambiar de una branch para otra.

$ git checkout

$ git checkout -b nombre\_branch //Crea una nueva branch y apunta para ella.

* **git log**: Muestra el historial de commit, se puede usar de varias formas:

$ git log

$ git log --oneline

* **git remote**: Muestra el repositorio remoto:

$ git remote

$ git remote -v

* **git pull**: Descarga el contenido de repositorio remoto y actualiza automáticamente el repositorio local.

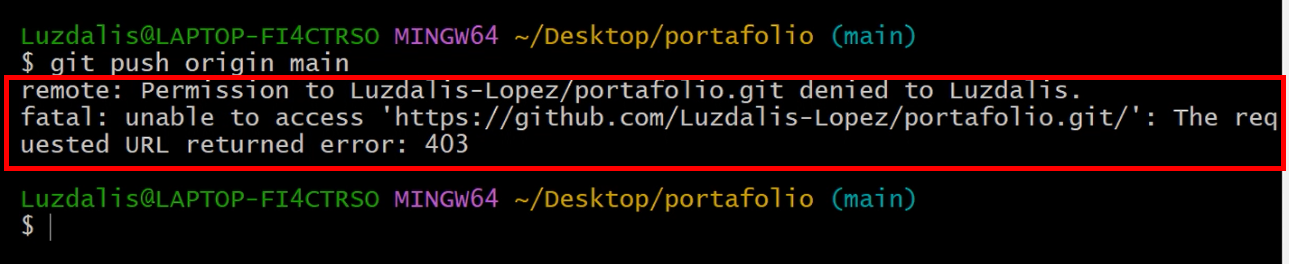
$ git pull

* **git push**: Envia contenido desde el repositorio local y actualiza automaticamente el repositorio remoto

$ git push

**Algunas recomendaciones:**

* Para la creación de alfombras o directorios, evite poner espacios y acentos en el nombre, esto evitará roturas o errores, al ejecutar los comandos en la terminal.
* Si eres nuevo usuario de Git y GitHub, es muy importante que realices las configuraciones de las variables globales y de la clave SSH, para que te evites dolores de cabeza y no logres hacer el versionado correctamente; uno de los errores más comunes vistos como resultado de perder algunos de ellos, es el siguiente en la imagen:



*Permission to... denied to... (Permiso denegado)*

Este error aparece cuando hemos clonado un repositorio con la dirección URL HTTP vista anteriormente, y estamos intentando enviar actualizaciones al repositorio remoto, solo que GitHub no reconoce quién está enviando los datos; para solucionar este problema debes crear una clave SSH que te autentique para que puedas hacer cambios en ella, luego debes hacer un clon con la dirección URL SSH y listo. Solo necesitarás hacerlo una vez; a partir de ahí tu usuario quedará autenticado para enviar cambios a cualquier repositorio existente en tu perfil de GitHub sin ningún inconveniente.