

IES Punta del Verde

Trabajo Git

Álvaro Maseri Apraiz

2ºASIR

¿Qué es Git?

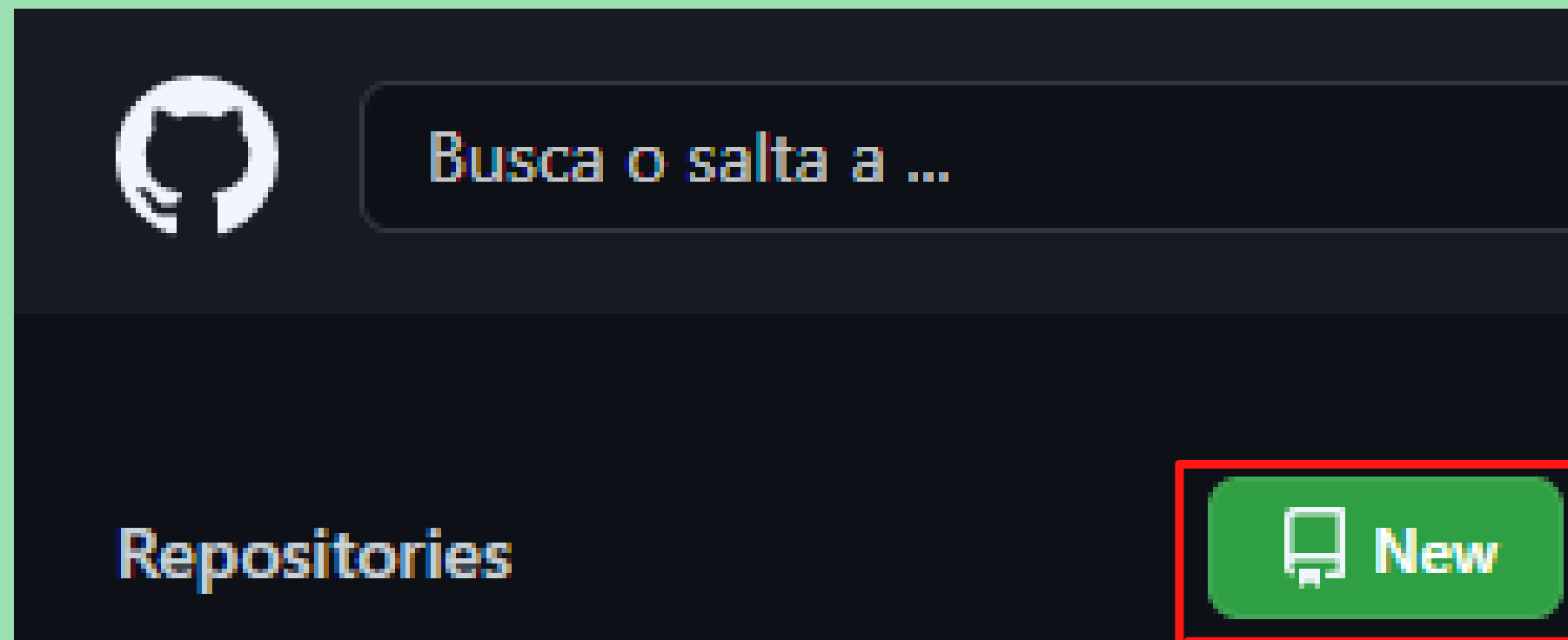
Git es un sistema de control de versiones creado por Linus Torvalds, actualmente es el más utilizado en el mundo entero.



Git es un proyecto de código abierto maduro y con un mantenimiento activo

Como crear repositorio en GitHub

Hacemos click en new




Rellenamos los datos

Los archivos README se utilizan normalmente para añadir información importante sobre el proyecto al que se refiere

Crea un nuevo repositorio

Un repositorio contiene todos los archivos del proyecto, incluido el historial de revisiones. ¿Ya tienes un repositorio de proyectos en otro lugar? [Importar un repositorio](#).

Dueño *


 maseri420 ▾

/


Nombre del repositorio *

Los grandes nombres de repositorios son breves y fáciles de recordar. ¿Necesitas inspiración? ¿Qué tal una **risita brillante** ?

Descripción (opcional)

☒  **Público**

Cualquiera en Internet puede ver este repositorio. Tú eliges quién puede comprometerse.

☐  **Privado**

Tú eliges quién puede ver y comprometerse con este repositorio.

Inicialice este repositorio con:

Omita este paso si está importando un repositorio existente.

☒ **Agregar un archivo README**


Aquí es donde puede escribir una descripción larga de su proyecto. [Aprende más](#).

☐ **Agregar .gitignore**

Elija qué archivos no rastrear de una lista de plantillas. [Aprende más](#).

☐ **Elija una licencia**

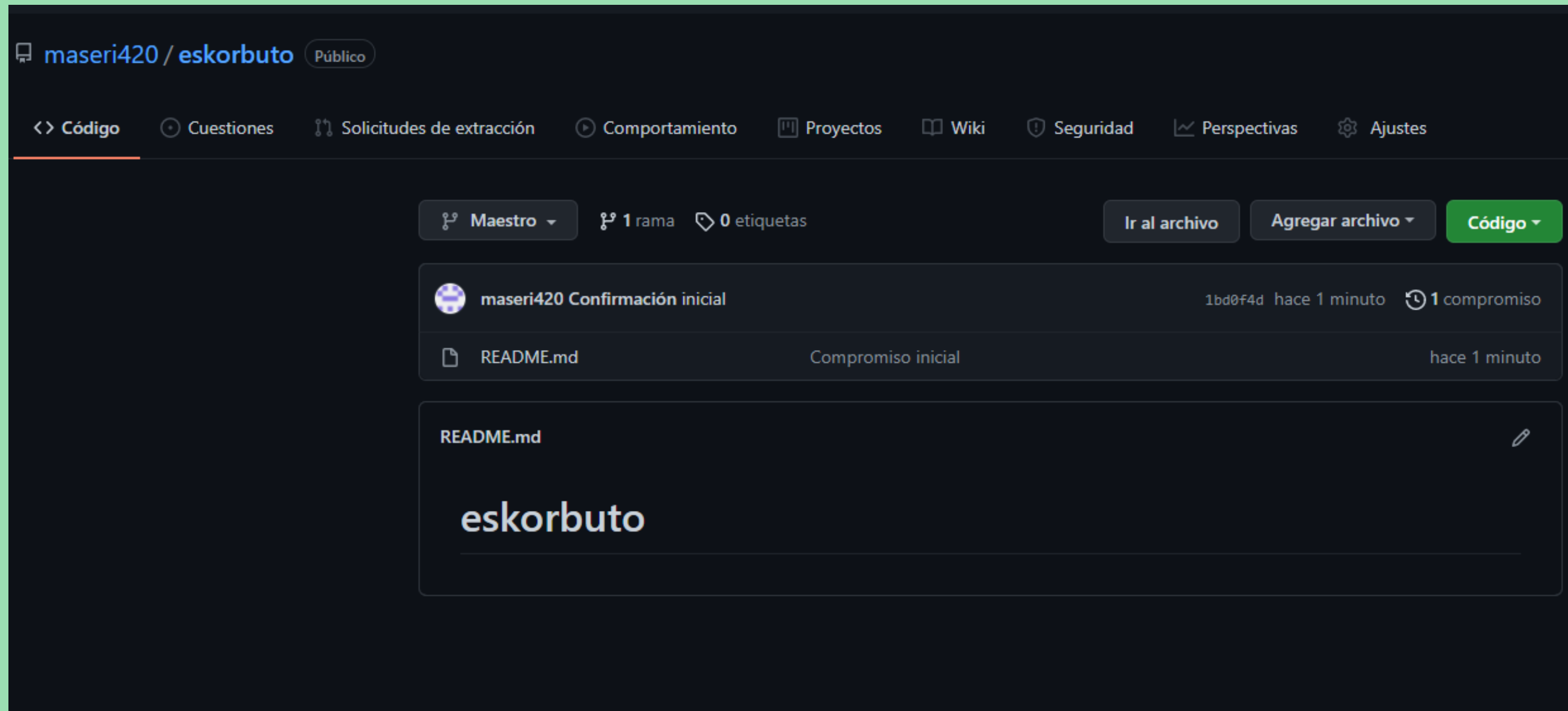
Una licencia les dice a otros lo que pueden y no pueden hacer con su código. [Aprende más](#).

Esto establecerá  **Maestro** como rama predeterminada. Cambie el nombre predeterminado en su [configuración](#).

Crear repositorio

Los archivos .gitignore se utiliza para indicarle a Git que archivos debe ignorar de un proyecto

Aquí se encuentra el repositorio en donde
se encuentra únicamente el archivo
README.md añadido anteriormente



Para editar el archivo README:
Click encima

Maestro ▾

2 ramas

0 etiquetas

Ir al archivo

Agregar archivo ▾

Código ▾

 maseri420

Confirmación inicial


1bd0f4d · anteayer · 1 compromiso

 README.md

Compromiso inicial

anteayer

Click en el lápiz

 maseri420

Actualización README.md


Última confirmación e645893 · ahora · Historia


 1 colaborador


0 líneas (0 sloc) | 1 byte


Crudo

Culpar









1

Escribimos lo deseado, acepta lenguaje html




```
1  # eskorbutto
2  < h1 > hola gente < h1 >
3
```

Cometemos cambios


Cometer cambios


Cancelar

Y ya tenemos algo dentro del archivo

 **maseri420** Actualización README.md


Última confirmación 9e078cd ahora [Historia](#)


 1 colaborador


 2 líneas (2 sloc) | 33 bytes


Crudo

Culpar











```
eskorbuto

hola gente
```


Para usar el gitignore necesitamos un archivo con extension .gitignore

 ejemplo1	23/09/2021 19:05	Documento de te...	1 KB
 ignorar	23/09/2021 19:36	Documento de te...	0 KB

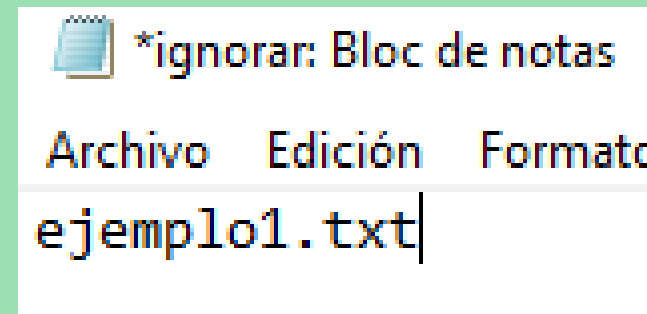
voy a realizar un git status para ver como tengo el arbol

```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   ejemplo1.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ignorar.gitignore
```

Como podemos ver no hice commit de ejemplo1.txt

Escribimos dentro de archivo .gitignore los archivos que no deseamos ver en git



Ahora hago un git status y como puede ver no detecta ejemplo1.txt

```
No commits yet

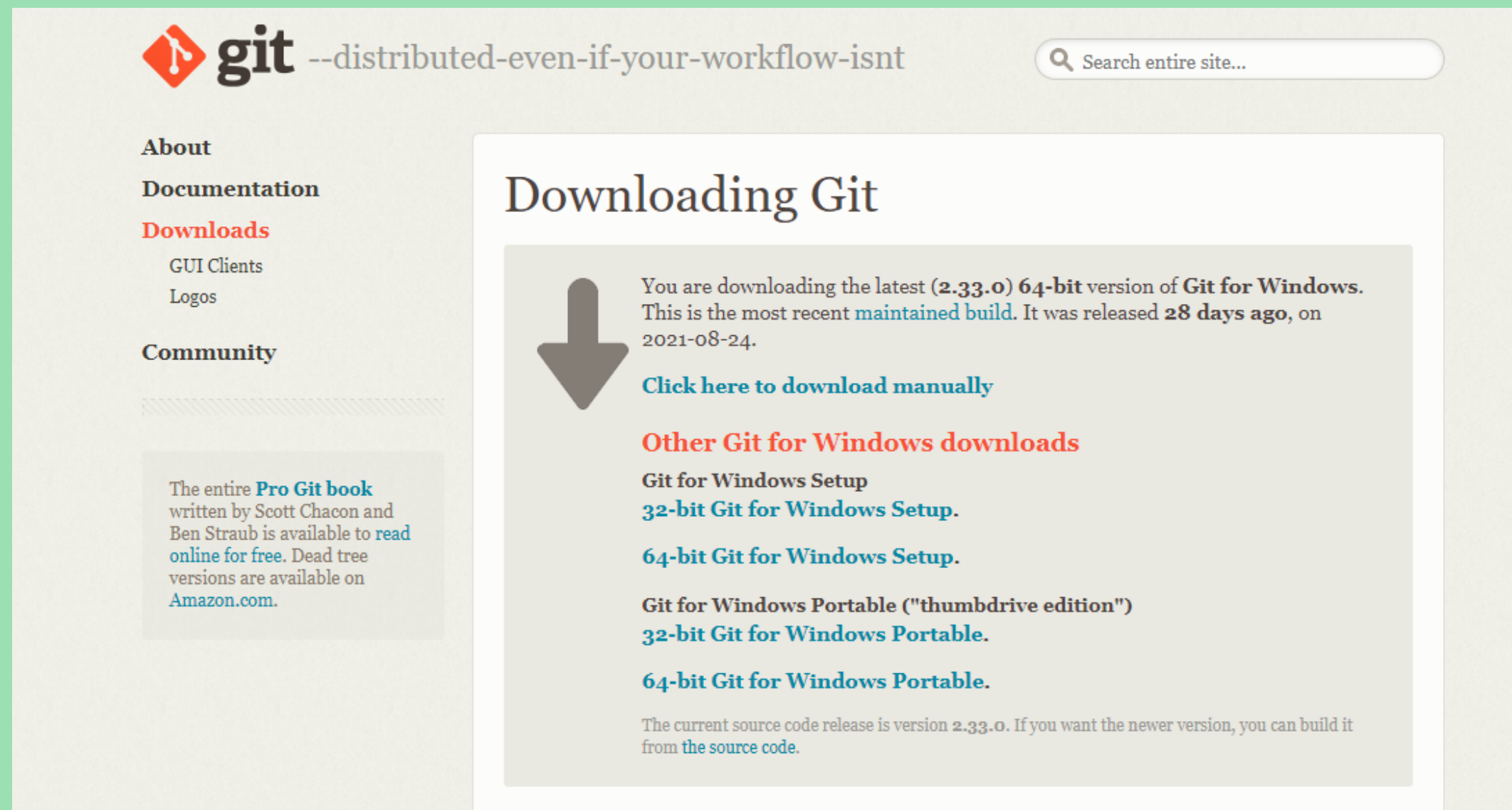
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  .gitignore
```

Es importante mencionar que el archivo .gitignore debe estar al mismo nivel que el .git y solo puedes ignorar archivos que este a ese nivel

Git Download

El proceso de instalación se realiza mediante el siguiente enlace donde encontrarás:

<https://git-scm.com/download/win>



The screenshot shows the Git website's download page for Windows. The header features the Git logo and the tagline "--distributed-even-if-your-workflow-isnt", along with a search bar. The left sidebar contains navigation links: "About", "Documentation", "Downloads" (highlighted in red), "GUI Clients", "Logos", and "Community". A promotional box for the "Pro Git book" is also visible. The main content area is titled "Downloading Git" and features a large downward arrow. The text informs the user they are downloading the latest (2.33.0) 64-bit version of Git for Windows, released 28 days ago on 2021-08-24. It provides a link to "Click here to download manually" and lists other download options: "Git for Windows Setup", "32-bit Git for Windows Setup.", "64-bit Git for Windows Setup.", "Git for Windows Portable ('thumbdrive edition')", "32-bit Git for Windows Portable.", and "64-bit Git for Windows Portable.". At the bottom, it mentions the current source code release is version 2.33.0 and provides a link to "the source code".

git --distributed-even-if-your-workflow-isnt

Search entire site...

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read [online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloading Git

You are downloading the latest (**2.33.0**) **64-bit** version of **Git for Windows**. This is the most recent [maintained build](#). It was released **28 days ago**, on 2021-08-24.

[Click here to download manually](#)

Other Git for Windows downloads

Git for Windows Setup

32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

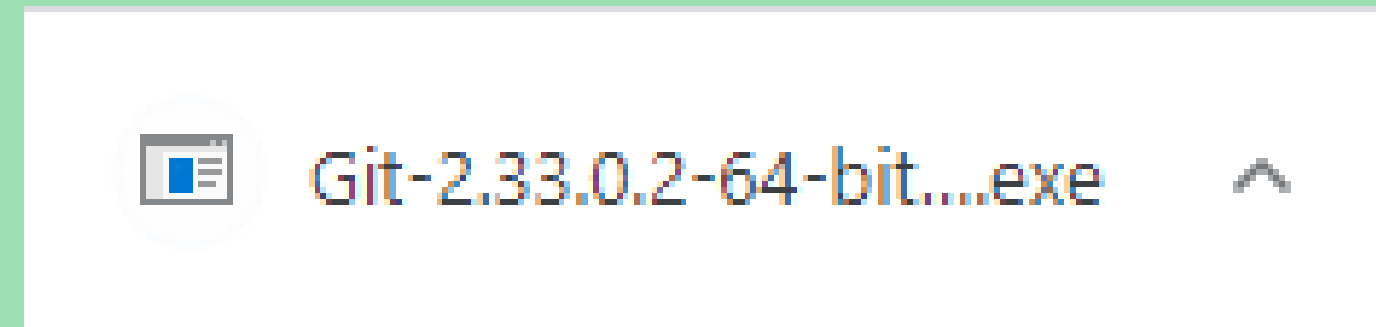
Git for Windows Portable ("thumbdrive edition")

32-bit Git for Windows Portable.

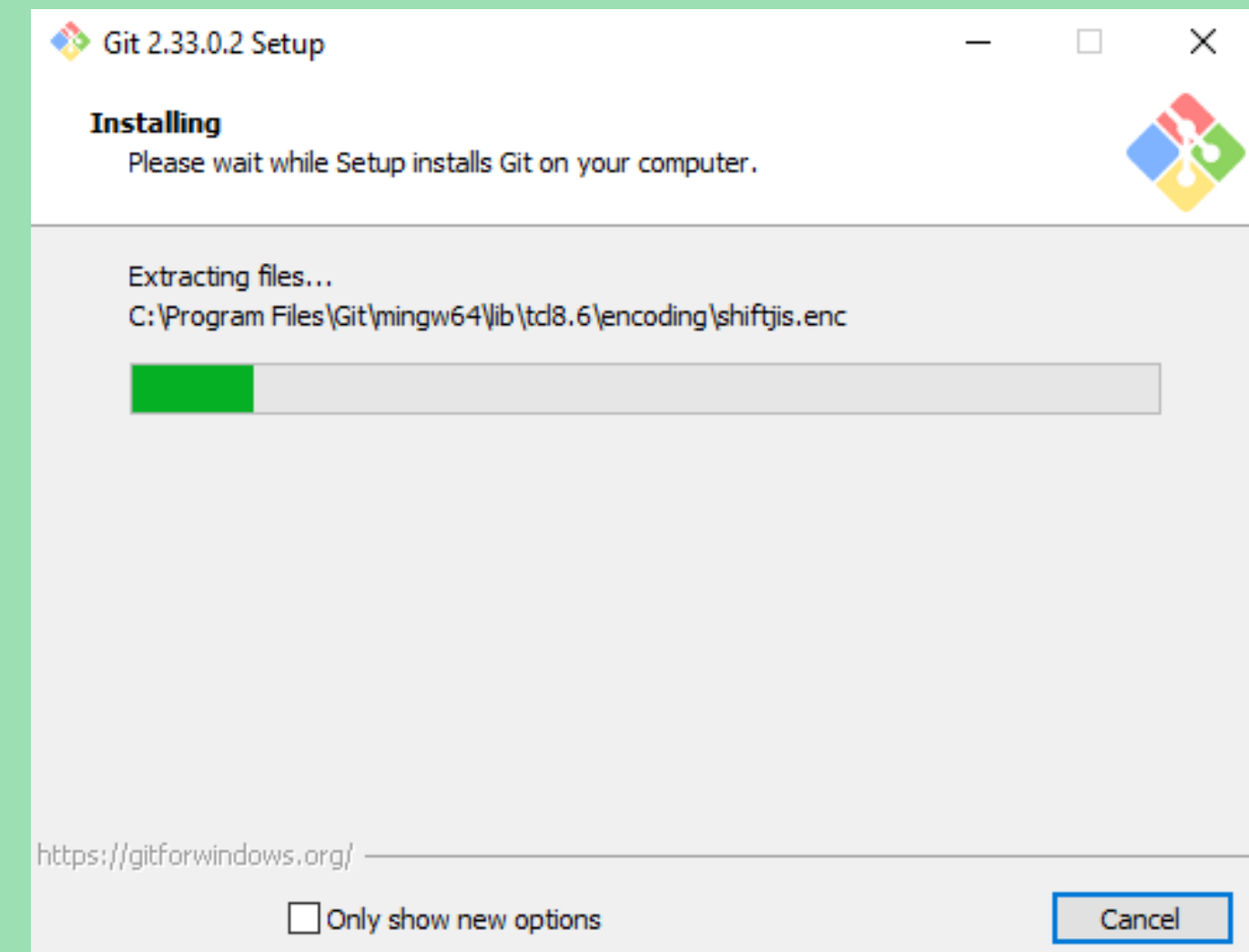
64-bit Git for Windows Portable.

The current source code release is version **2.33.0**. If you want the newer version, you can build it from [the source code](#).

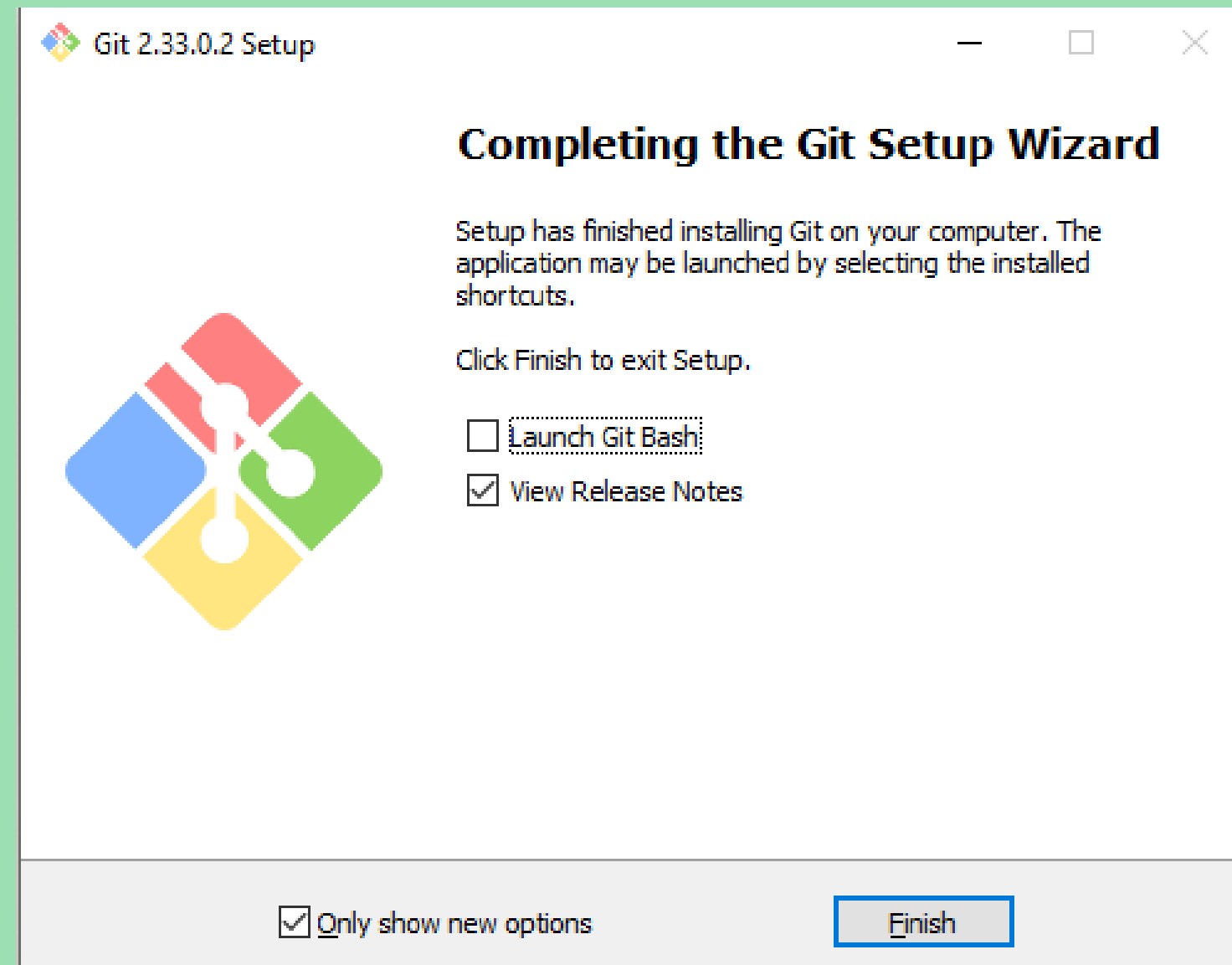
Seleccionamos el archivo correspondiente y una vez descargado lo ejecutamos



A continuación aceptamos la configuración como más nos convenga y descargamos.



Una vez termine la descarga ya habremos terminado el proceso de instalación.



Términos

A continuación desarrollare los distintos comandos de Git
con un sencillo ejemplo

- Primero debemos abrir la terminal, cómo estoy trabajando en Windows entramos en Windows PowerShell.



Windows PowerShell

Aplicación

- El comando Git init crea un nuevo repositorio de Git. Puede utilizarse para convertir un proyecto existente y sin versión en un repositorio de Git, o para inicializar un nuevo repositorio vacío.

```
PS C:\Users\usuario\Desktop\PUNTA_DEL_VERDE\base de datos\1º trimestre\Documentaciones\1> git init
Initialized empty Git repository in C:/Users/usuario/Desktop/PUNTA_DEL_VERDE/base de datos/1º trimestre/Documentaciones/1/.git/
```

- Nos registramos en Git con nuestro nombre global(usuario de Github) y correo electrónico

```
git config --global user.name maseri420
git config --global user.email alvaromaseriapraiz@gmail.com
```

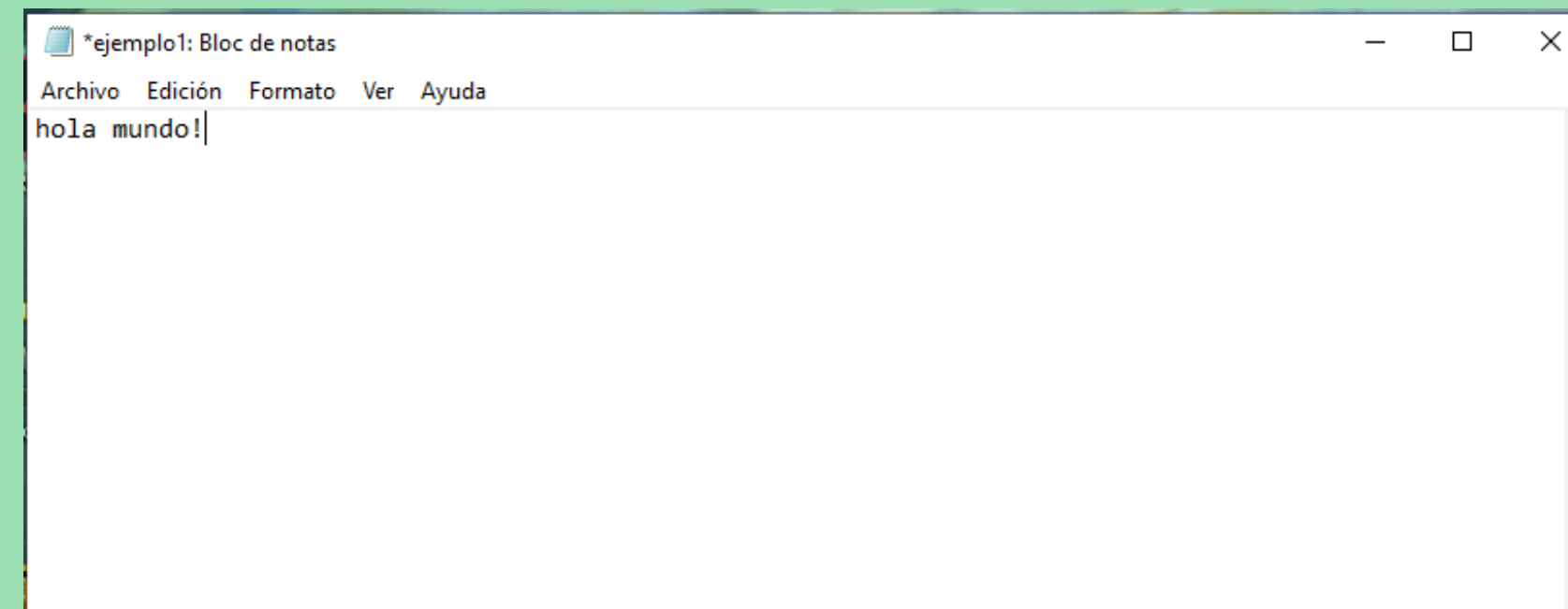
- Creamos un archivo de texto vacío llamado ejemplo1.txt

```
new-item ejemplo1.txt
```

- Añadimos el archivo del área de trabajo al repositorio mediante el comando Git add, esto es solo necesario hacerlo 1 vez por archivo

```
git add .\ejemplo1.txt
```

- Editamos el contenido de su interior y guardamos



- Confirmamos los cambios en el repositorio mediante el comando
Git commit -m

El parámetro -m permite agregarle un comentario al guardado

```
git commit -m "cambio1"
```

Además con el parámetro -a podemos ejecutar el comando Git add al realizar el propio commit, por ejemplo:

- Realizo otros cambios en el archivo y entoces

```
git commit -a -m "cambio2"
```

- El comando git status muestra el estado del directorio de trabajo y del área del entorno de ensayo.

```
git status
```

```
On branch master  
nothing to commit, working tree clean
```

Si ejecutáramos un git status sin que hayamos añadido y guardado los cambios nos saltaría el siguiente error

```
On branch master  
Changes not staged for commit:  
  (use "git add <file>..." to update what will be committed)  
  (use "git restore <file>..." to discard changes in working directory)  
        modified:   ejemplo1.txt
```

Ahora subiremos los archivos al repositorio remoto en GitHub

Utilizaremos el comando Git branch el cual te permite crear o eliminar ramas, con esto dicho aplicándole el parámetro -M creamos una rama nueva con el nombre deseado

```
git branch -M main
```

Seguidamente añadimos el remoto

```
git remote add origin https://github.com/maseri420/e
```

- Este es un comando que empuja a los commits de la rama local al remoto llamado origin, es decir renombra el repositorio remoto pero en el local para referenciarlo con el nuevo nombre

Continuamente nos solicitarán el elemento de acceso para entrar. Otra manera de hacerlo es desde el navegador (sing in with your browser)

Connect to GitHub

×

GitHub

Sign in

Sign in with your browser

or

Personal Access Token

Sign in

Don't have an account?

[Sign up](#)

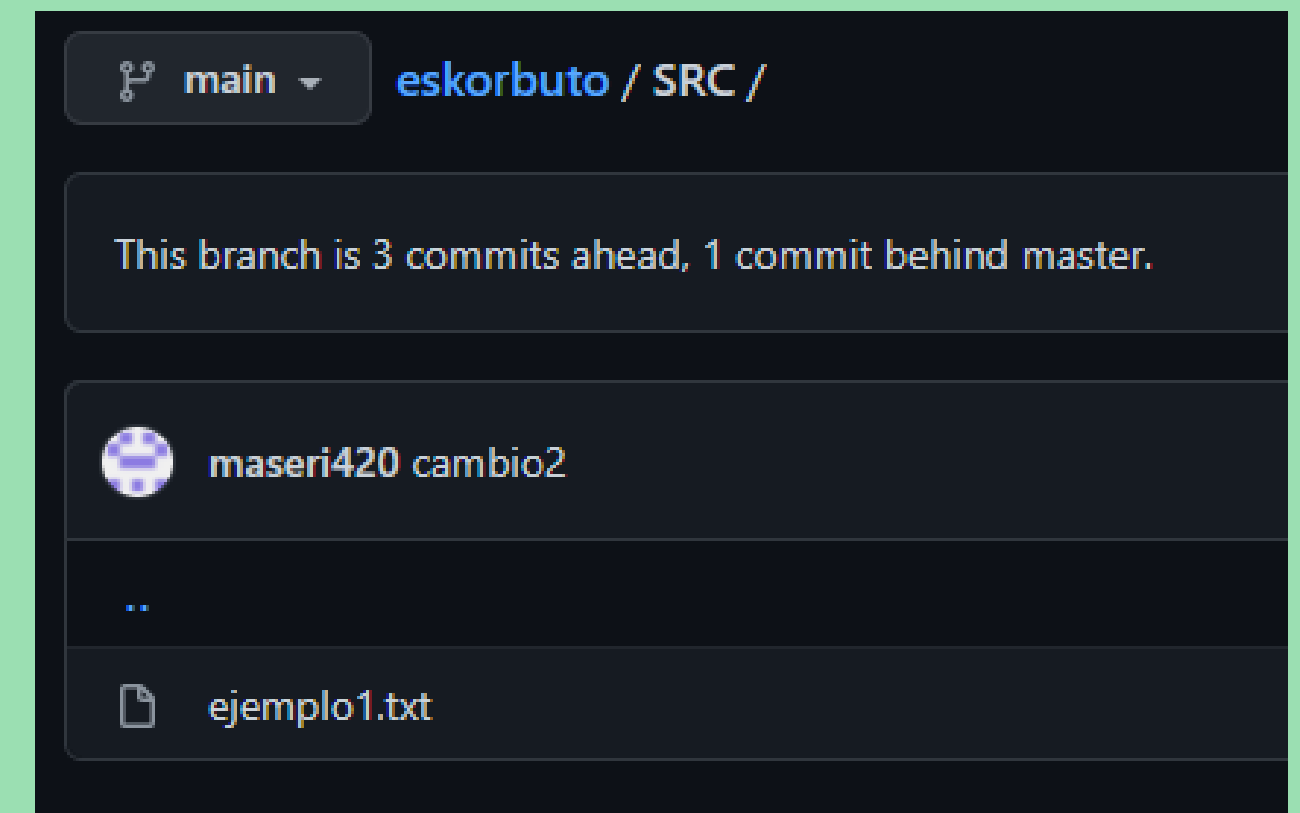
Y ya lo tendríamos conectado!!!

Ahora subiremos los commits para ello utilizaremos el comando Git push el cual te permite subir estos commits al repositorio remoto

```
git push -u origin main
```

El parámetro -u produce un seguimiento de las ramas que ya estén actualizadas o enviadas con éxito.

Pasándole de manera continua el nombre del remoto y de la rama local



Sin embargo si en vez de subir queremos descargarnos los archivos de un repositorio remoto utilizaremos el comando Git pull

```
git pull origin main
```

Añadiéndole el nombre del repositorio y la rama.

```
remote: Enumerating objects: 12, done.  
remote: Counting objects: 100% (12/12), done.  
remote: Compressing objects: 100% (3/3), done.  
remote: Total 12 (delta 0), reused 12 (delta 0), pack-reused 0  
Unpacking objects: 100% (12/12), 800 bytes | 2.00 KiB/s, done.  
From https://github.com/maseri420/eskorbuto  
* branch          main          -> FETCH_HEAD  
* [new branch]    main          -> origin/main
```

Descarga realizada;

Cuando en vez de descargar un proyecto, deseamos clonarlo se utiliza el comando git clone

```
git clone https://github.com/maseri420/eskorbuto.git
```

Se descarga y ya lo tenemos clonado!!

```
Cloning into 'eskorbuto'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 15 (delta 0), reused 12 (delta 0), pack-reused 0
Receiving objects: 100% (15/15), done.
```



Como dije anteriormente con git branch se pueden crear ramas nuevas
pues si queremos cambiar de ramas utilizamos git checkout

```
git branch rama1
```

Ahora nos movemos a la rama1

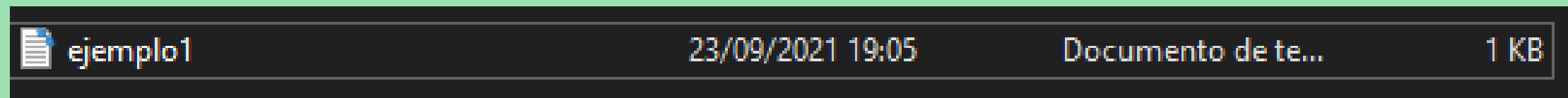
```
git checkout rama1
```

Nueva rama llamada rama1 en donde creo un archivo de texto llamado
ejemplo2.txt, después de añadirlo y aplicarle el commit voy a ir a la rama
main

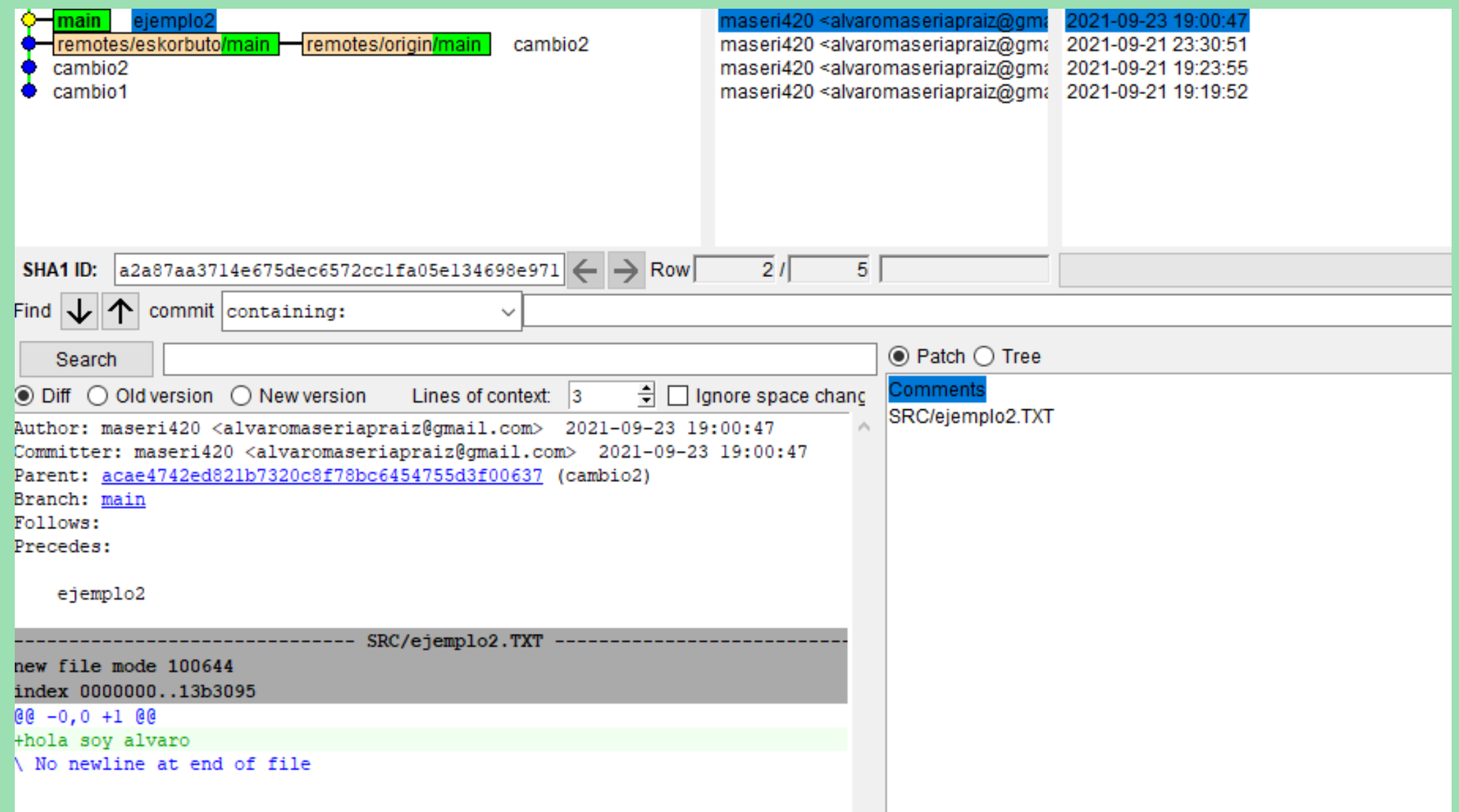
 ejemplo1	21/09/2021 19:31	Documento de te...	1 KB
 ejemplo2	23/09/2021 18:58	Documento de te...	1 KB

```
git checkout main
```


Como podrás observar aquí solo tengo el ejemplo1.txt debido a que el ejemplo2.txt



Con el comando gitk nos saltara una interfaz gráfica para ver todo el proceso realizado



Ahora voy a unir las dos ramas a partir de git merge, es importante estar en la rama donde quieres que este todos los archivos ya que estos estarán sobre la rama que haces el merge

```
git merge rama1
```

Con gitk se ve de manera sencilla

