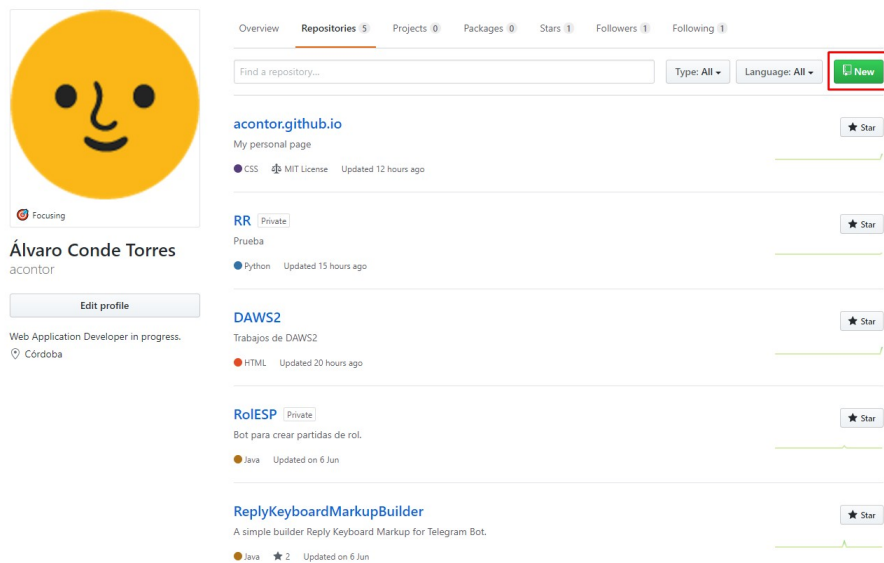


GITHUB

Guía rápida de uso por Álvaro Conde Torres

1. Crear repositorio.

Para crear un repositorio, **nos dirigiremos hacia nuestros repositorios dentro de Github**. Encontraremos por la parte superior un botón para poder crearlo.



En la siguiente página, le colocamos un **nombre**, una **descripción**, indicamos si queremos que sea **público o privado**, seleccionamos el checkbox para que nos cree un **README** y le añadimos una **licencia**.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner: acontor / Repository name:

Great repository names are short and memorable. Need inspiration? How about [fantastic-potato?](#)

Description (optional):

☒ Public
Anyone can see this repository. You choose who can commit.

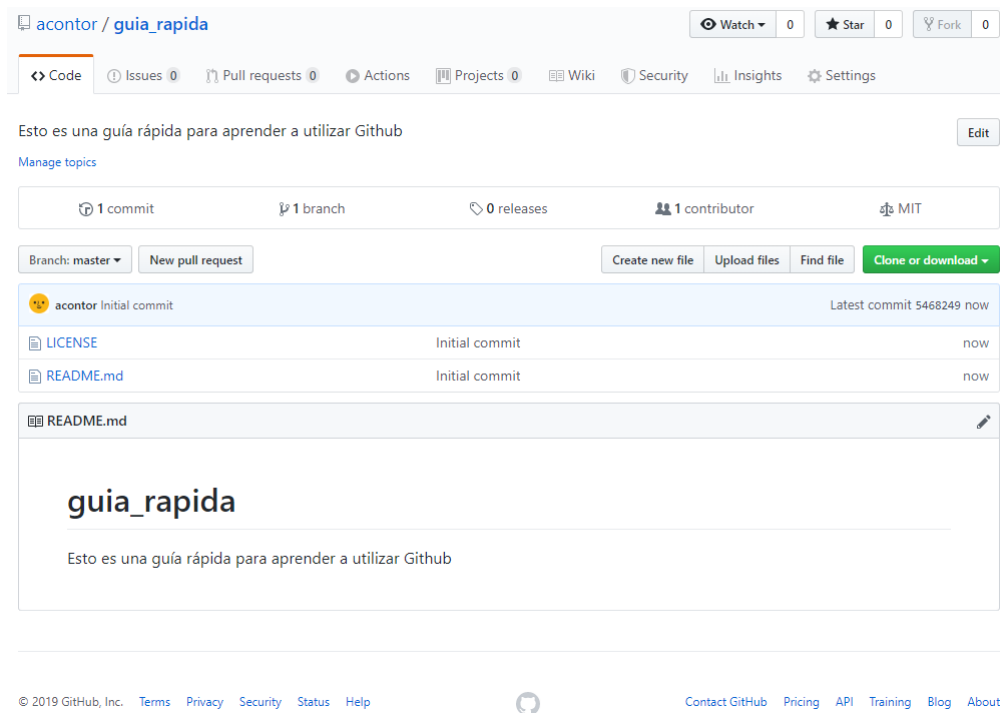
☐ Private
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☒ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: Add a license:

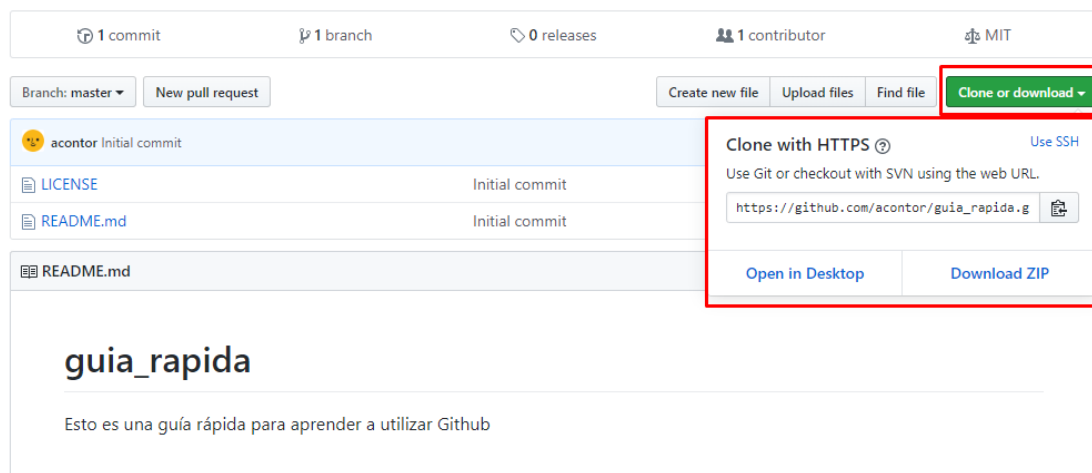
Ya nos aparecerá la web principal de nuestro repositorio.



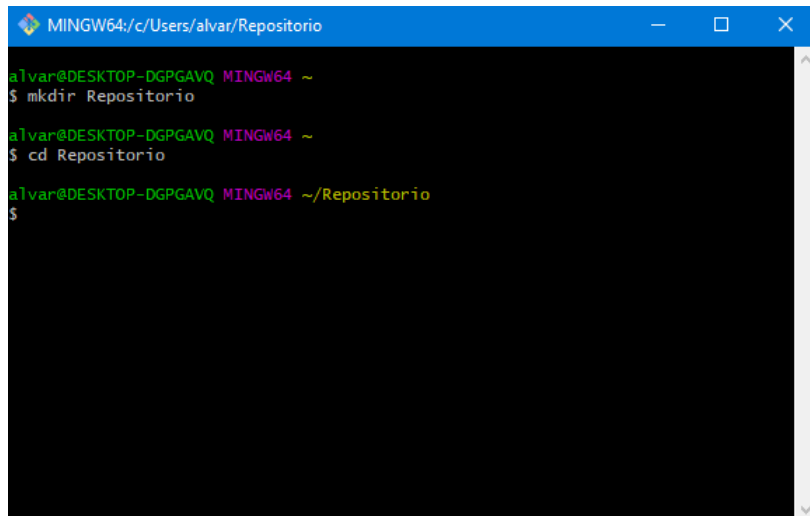
** Si en el nombre de repositorio utilizamos xxx.github.io, nos creará un sitio web accesible desde el navegador.

2. Clonar repositorio.

Para clonar un repositorio tenemos que tener instalada la **herramienta Git**. Utilizaremos el comando **git clone 'url de repositorio'**. La url la encontramos en la página principal del repositorio.

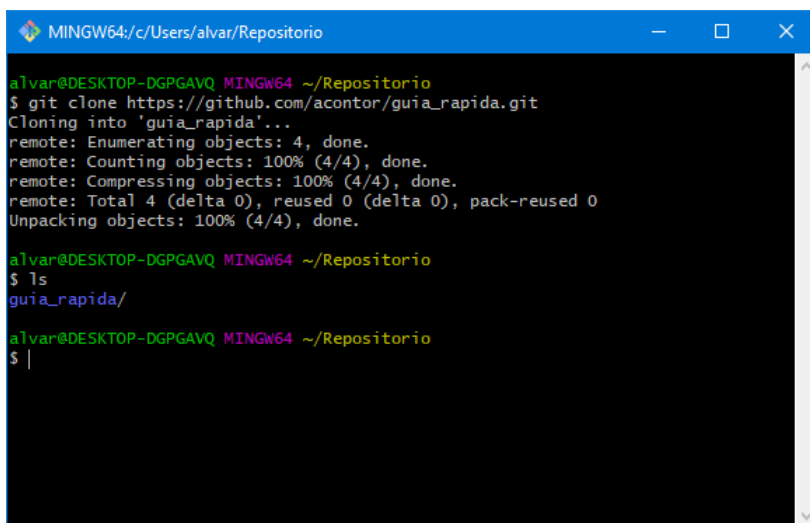


Abrimos **Git Bash**, que es el **cmd** que nos proporciona la herramienta **Git**, y creamos una **carpeta** donde guardar nuestros repositorios y entramos a ella.



```
MINGW64:/c/Users/alvar/Repositorio
alvar@DESKTOP-DGPGAVQ MINGW64 ~
$ mkdir Repositorio
alvar@DESKTOP-DGPGAVQ MINGW64 ~
$ cd Repositorio
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio
$
```

Pasamos a utilizar el comando **git clone** y ejecutamos un **ls** para ver si ha sido creada la carpeta con el nombre del repositorio creado.



```
MINGW64:/c/Users/alvar/Repositorio
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio
$ git clone https://github.com/acontor/guia_rapida.git
Cloning into 'guia_rapida'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), done.
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio
$ ls
guia_rapida/
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio
$ |
```

3. Actualizar repositorio.

Una vez manipulemos el contenido de la carpeta **guia_rapida**, vamos a actualizar el repositorio de github desde la herramienta **Git**.

Para ello vamos a hacer uso de los comandos **add**, **commit** y **push**.

También podremos ir divisando los cambios con **status**.

Entramos a la carpeta del repositorio y creamos un archivo, en nuestro caso, index.html.

```
MINGW64:/c/Users/alvar/Repositorio/guia_rapida
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio
$ cd guia_rapida/

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ ls
index.html  LICENSE  README.md

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ |
```

Vamos a utilizar por primera vez el comando `git status` que nos indicará los cambios realizados en el repositorio.

```
MINGW64:/c/Users/alvar/Repositorio/guia_rapida
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

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

        index.html

nothing added to commit but untracked files present (use "git add" to track)
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$
```

Para añadir los cambios es necesario utilizar primero el comando **git add**. Este comando añadirá los cambios realizados a la carpeta llamada `.git` que se encuentra oculta dentro de nuestra carpeta local. Podemos añadir todos los cambios con **git add .** o añadir un solo cambio con **git add index.html**. Siempre es recomendable la primera opción.

```
MINGW64:/c/Users/alvar/Repositorio/guia_rapida
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ git add .

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   index.html

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$
```

Ahora es el turno de **git commit**. Vamos a **guardar los cambios en el repositorio local**, listos para ser enviados a nuestro github. Hacemos uso de la **opción -m para agregar un comentario al commit** de forma rápida.

```
MINGW64: c:/Users/alvar/Repositorio/guia_rapida

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ git commit -m 'Primer commit de nuestra guía'
[master ed99385] Primer commit de nuestra guía
1 file changed, 12 insertions(+)
create mode 100644 index.html

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

nothing to commit, working tree clean

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$
```

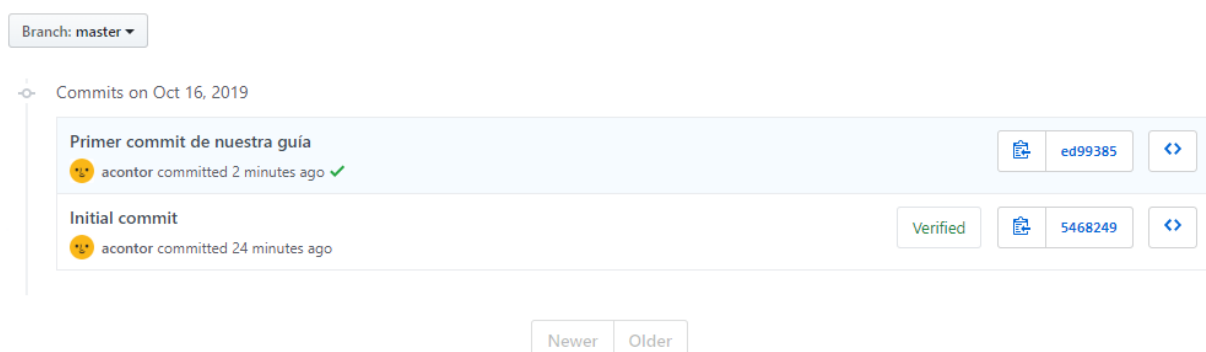
Y, por último, **enviaremos las modificaciones al repositorio de github**. Para ello, utilizamos el comando **git push**.

```
MINGW64: c:/Users/alvar/Repositorio/guia_rapida

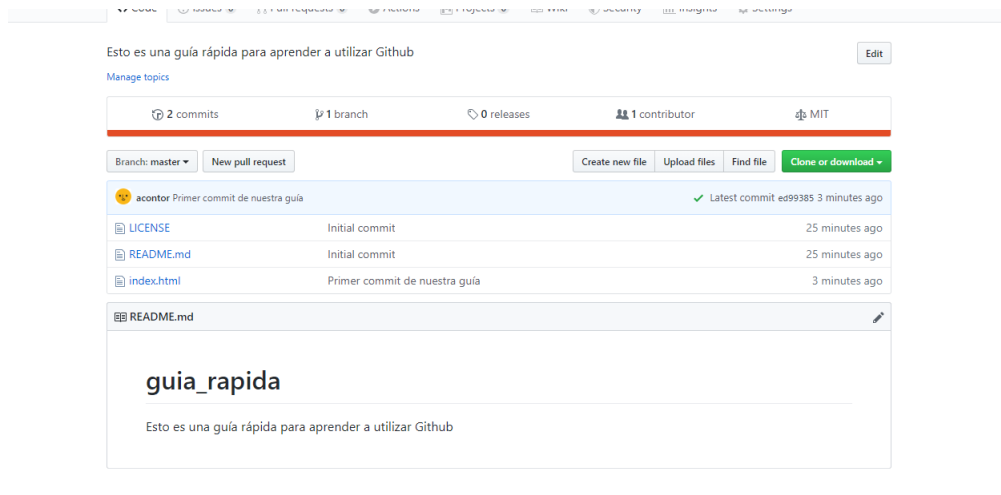
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 562 bytes | 562.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/acontor/guia_rapida.git
  5468249..ed99385  master -> master

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ |
```

Quizás la primera vez que se realice el proceso pida una autenticación. Es solo colocar vuestros datos de acceso a github. Ya veremos el commit en la web del repositorio en github.



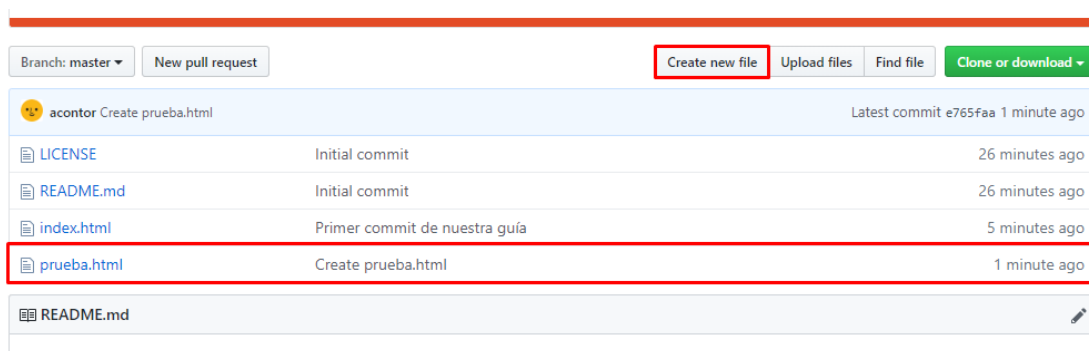
Y, en el code del repositorio, veremos el index.html creado.



4. Actualizar repositorio local

Si realizamos cambios en el repositorio online de Github, podemos utilizar el comando **git pull** para **descargarnos las modificaciones a nuestro repositorio local**.

Hemos creado un archivo prueba.html desde la web.



Y ahora ejecutamos el comando git pull en Git Bash.

```
MINGW64/c/Users/alvar/Repositorio/guia_rapida
alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ git pull
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/acontor/guia_rapida
   ed99385..e765faa  master    -> origin/master
Updating ed99385..e765faa
Fast-forward
 prueba.html | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 prueba.html

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ ls
index.html  LICENSE  prueba.html  README.md

alvar@DESKTOP-DGPGAVQ MINGW64 ~/Repositorio/guia_rapida (master)
$ |
```