



**TRABAJANDO CON GIT**

# ÍNDICE

---

- 1.- Operaciones básicas con Git
- 2.- Características de JavaScript
- 3.- Preparación del entorno
- 4.- Ejecución de JavaScript
- 5.- Depuración del código
- 6.- Transpiladores y polyfills

# 1

## OPERACIONES BÁSICAS CON GIT



git

## Versión de git

Con la orden **git versión** podemos ver la versión de git que tenemos instalado en el equipo.

```
PS C:\Users\Victor> git version  
git version 2.37.3.windows.1
```

## Configuración inicial

Lo primero que hay que hacer tras instalar **git** es actualizar la configuración con el nombre y el correo electrónico utilizando la orden **git config**.

```
PS C:> git config --global user.name "Victor J. González"  
PS C:> git config --global user.email "vgonzalez165@gmail.com"
```

## Inicialización del repositorio

La orden **git init** inicializa un repositorio en la carpeta en la que nos encontremos.

```
PS D:\proyectos\guia_git> git init  
Initialized empty Git repository in D:/proyectos/guia_git/.git/
```

La inicialización de un proyecto simplemente consiste en que se crea un fichero denominado `.git` que está oculto y contendrá todos los archivos necesarios para gestionar el repositorio.

```
PS D:\proyectos\guia_git> Get-ChildItem -hidden
```

```
Directory: D:\proyectos\guia_git
```

Mode	LastWriteTime	Length	Name
----	-----	-----	----
d--h-	14/09/2022 13:42		.git

Eliminar el repositorio es tan fácil como eliminar este archivo.

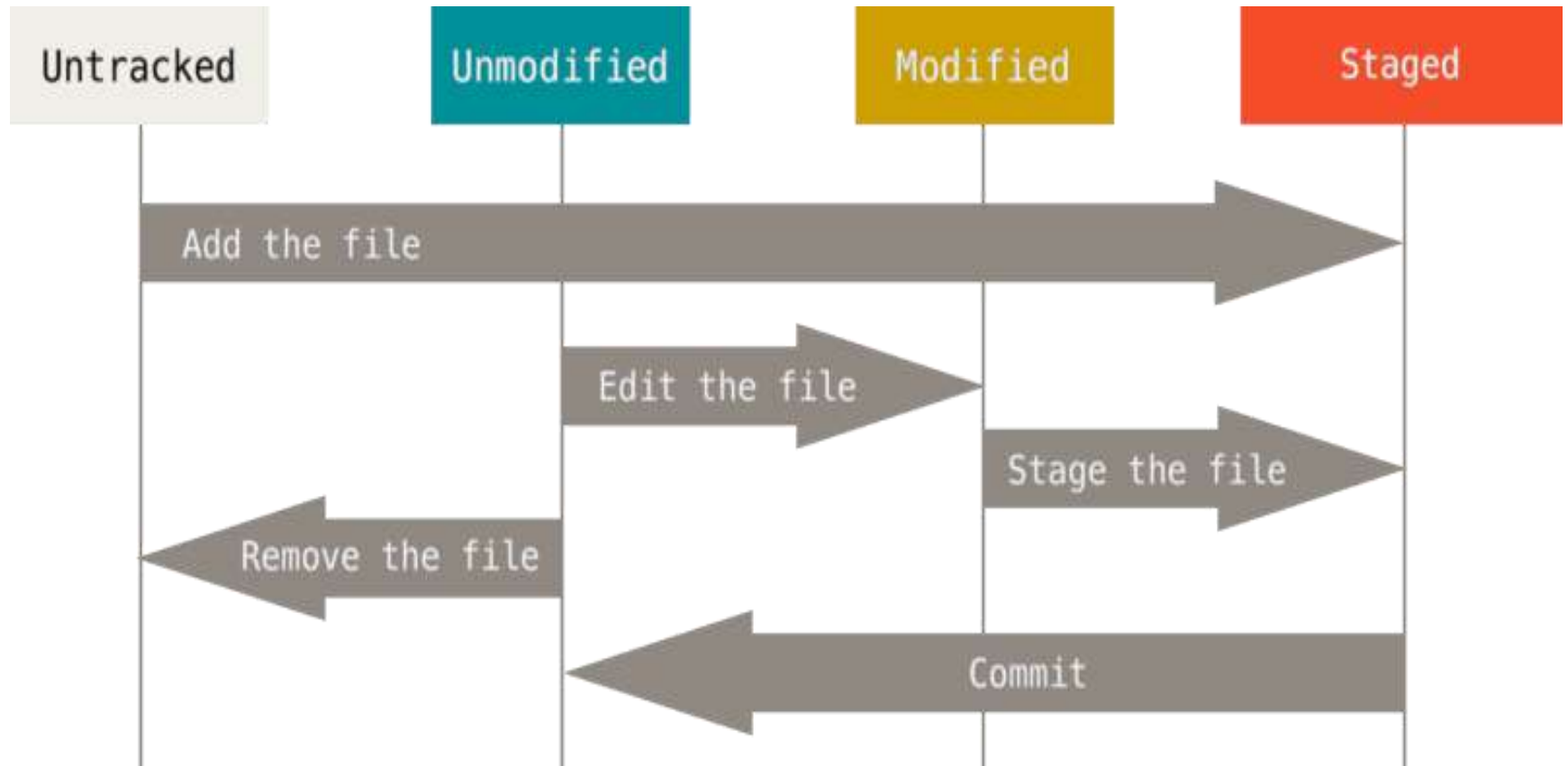
## Ciclo de vida de los archivos

En un repositorio git podemos considerar las siguientes áreas:

**Espacio de trabajo (workspace):** son los directorios y archivos que vemos y con los que trabajamos. Cada directorio o archivo puede estar en una de estas situaciones:

- **Untracked:** no se realiza seguimiento del archivo.
- **Unmodified:** se realiza seguimiento pero no hay sido modificado
- **Modified:** el fichero hay sido modificado





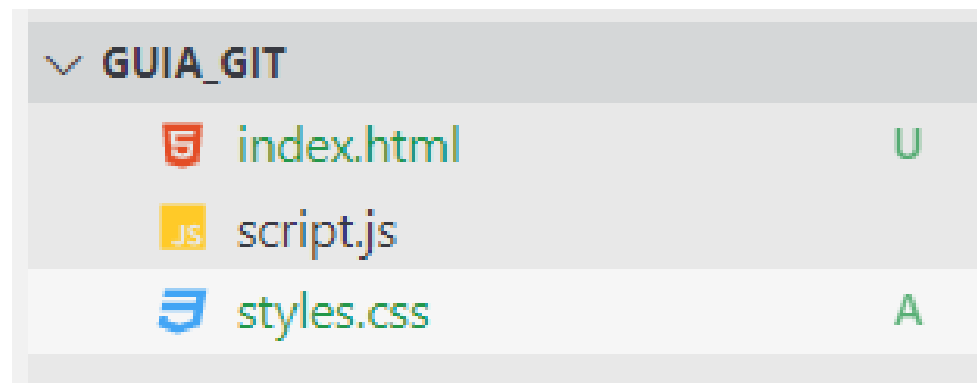
**Stage área (área de preparación):** es una zona intermedia temporal donde se guardan los archivos como paso previo a ser almacenados en el repositorio.

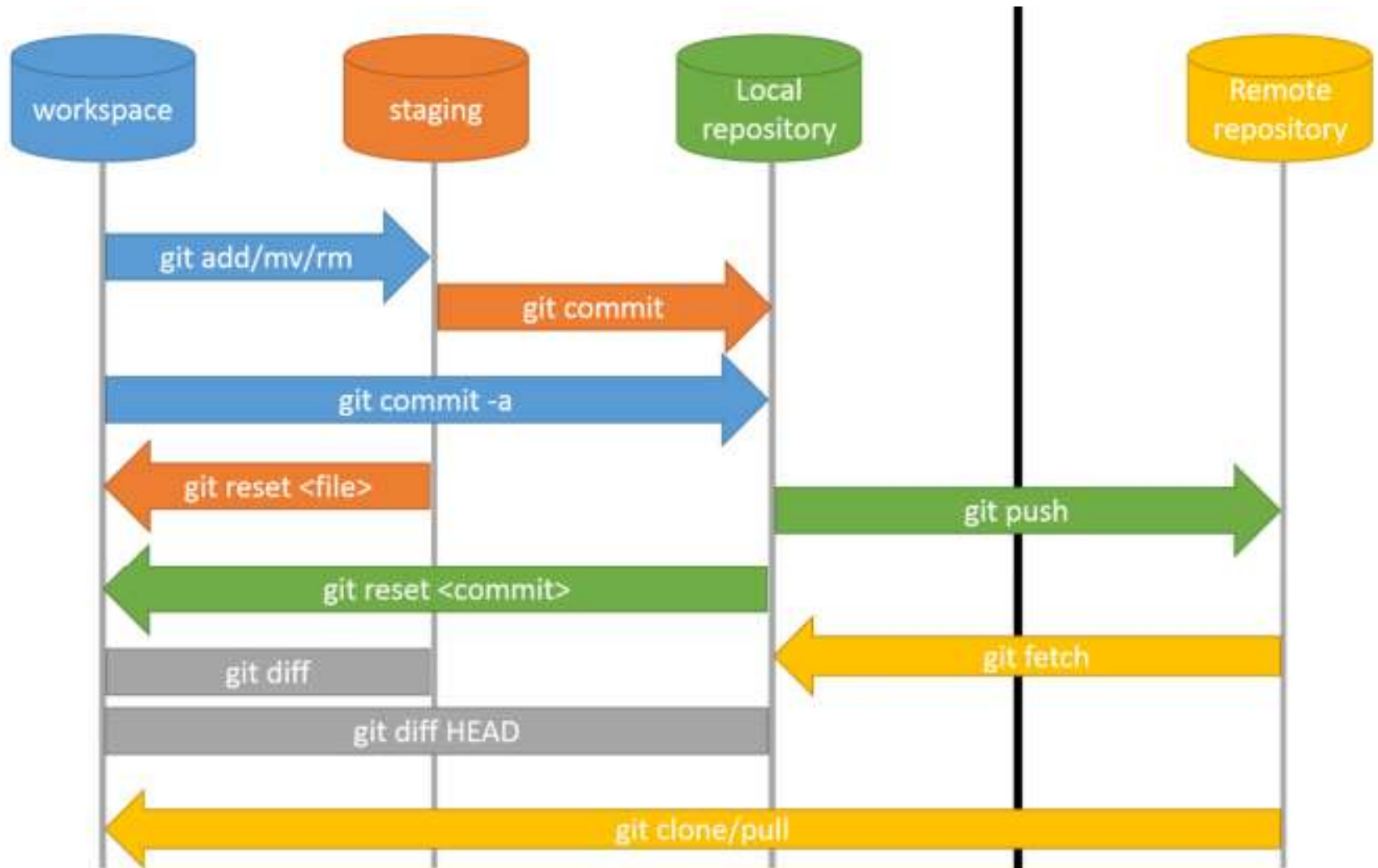
**Repositorio local:** es el espacio donde se guardan la diversas versiones creadas de los archivos en el equipo local.

**Repositorio remoto:** podemos sincronizar nuestro repositorio local con un repositorio remoto (Gibhub, GitLab, ...). Esto permite que varios desarrolladores trabajen con el mismo código.

Aquí vemos como se muestra en VisualStudio los archivos según su estado:

- *Index.html*: untracked
- *Script.js*: en repositorio local
- *Styles.css*: en el stage





## Añadir un archivo al seguimiento

Con el comando **git add <filename>** marcamos ficheros para su seguimiento, de forma que pasan al *stage área*.

```
● PS D:\proyectos\guia_git> git add index.html  
○ PS D:\proyectos\guia_git>
```

Podemos agregar todos los archivos del directorio actual y directorios hijo con la orden **git add .**

```
● PS D:\proyectos\guia_git> git add .  
○ PS D:\proyectos\guia_git>
```

## Añadir archivos al *stage area*

Con **git status** vemos todos los archivos que se encuentran actualmente en el *stage área*.

```
● PS D:\proyectos\guia_git> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   index.html
```

Aquí vemos todos los cambios que hay en el *stage area* desde el último *commit*.

## Quitar el seguimiento de archivos

Si queremos quitar el seguimiento de un archivo (es decir, enviarlo de nuevo a *untracked*) debemos usar **git rm --cached <filename>**

```
● PS D:\proyectos\guia_git> git rm --cached .\index.html
rm 'index.html'
● PS D:\proyectos\guia_git> git status
On branch master

No commits yet

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

## El archivo `.gitignore`

Normalmente en todos los proyectos hay archivos sobre los que no queremos hacer seguimiento. Por ejemplo:

- **Archivos compilados** (como los `.pyc` de Python)
- **Binarios** como diagramas o similares.
- **Archivos privados** del desarrollados o que tengan claves.
- **Archivos de sistemas de gestión de dependencias**, como la carpeta `node_modules` de node.



Aunque podemos evitar añadirlos al hacer **git add**, esto nos obliga a estar muy pendiente de ello, siendo fácil que se escape algún archivo.

Para evitar estos problemas Git dispone de la posibilidad de crear el archivo **.gitignore**.

Este es un archivo de texto donde cada línea contiene un elemento que Git debe ignorar y no agregar nunca a seguimiento. Se puede:

- Indicar directamente el nombre de un archivo
- Utilizar comodines, p.e. \*.pyc
- Señalar directorios completos, p.e. node\_modules/

## Enviar archivos al repositorio local

Cuando queramos enviar todos los cambios pendientes que hay en el *stage area* al repositorio utilizamos la orden **git commit -m <mensaje>**.

- PS D:\proyectos\guia\_git> git commit -m "Primer commit"  
[master (root-commit) 89e71ed] Primer commit  
1 file changed, 13 insertions(+)  
create mode 100644 index.html
- PS D:\proyectos\guia\_git> git status  
On branch master  
nothing to commit, working tree clean

## Mostrar listado de *commits*

Se puede ver un historial de todos los *commits* que hemos hecho al repositorio con **git log** o añadiendo el parámetro - - **oneline** para que muestre una versión más reducida.

```
● PS D:\proyectos\guia_git> git log
commit 89e71ed72ad4d54c58d0f86cd0e6a8b5b05edaaa (HEAD -> master)
Author: Víctor J. González <vgonzalez165@gmail.com>
Date:   Wed Sep 14 17:20:46 2022 +0200
```

Primer commit

```
● PS D:\proyectos\guia_git> git log --oneline
89e71ed (HEAD -> master) Primer commit
```

En ocasiones la salida puede ser excesivamente larga, por lo que podemos limitarnos a los últimos *n commits* con **git log -<n>**

```
● PS D:\proyectos\guia_git> git log -2
commit e68aa14c640a674a508ef9ce133328d35cf12857 (HEAD -> master)
Author: Victor J. González <vgonzalez165@gmail.com>
Date:   Wed Sep 14 17:26:18 2022 +0200

    Creado fichero styles.js

commit 6e2e693367c883af8830e24b6bd8cf88bedb091e
Author: Victor J. González <vgonzalez165@gmail.com>
Date:   Wed Sep 14 17:25:56 2022 +0200

    Creado fichero script.js
```

Se puede mostrar información sobre los cambios que se han guardado en cada *commit* con el parámetro **-p**

```
PS D:\proyectos\guia_git> git log -2 -p
commit e68aa14c640a674a508ef9ce133328d35cf12857 (HEAD -> master)
Author: Victor J. González <vgonzalez165@gmail.com>
Date:   Wed Sep 14 17:26:18 2022 +0200

    Creado fichero styles.js

diff --git a/styles.css b/styles.css
new file mode 100644
index 0000000..e69de29
```

## Detalle de los cambios de un *commit*

Se pueden ver los cambios en el código que hay en cada *commit* con la orden **git show <id>**

El campo <id> es el identificador del *commit*, que se muestra cuando hacemos **git log**. Se puede abreviar poniendo únicamente los primeros caracteres (mínimo 4)

```
● PS D:\proyectos\guia_git> git log -1
commit 66154d882a0d1382f57cf6bf5f03235d2164f68b (HEAD -> master)
Author: Victor J. González <vgonzalez165@gmail.com>
Date:   Wed Sep 14 17:35:51 2022 +0200
```

Color de fondo

```
PS D:\proyectos\guia_git> git show 66154
commit 66154d882a0d1382f57cf6bf5f03235d2164f68b (HEAD -> master)
Author: Victor J. González <vgonzalez165@gmail.com>
Date:   Wed Sep 14 17:35:51 2022 +0200
```

Color de fondo

```
diff --git a/styles.css b/styles.css
index e69de29..639762b 100644
--- a/styles.css
+++ b/styles.css
@@ -0,0 +1,3 @@
+body {
+  background-color: azure;
+}
\ No newline at end of file
```

## Modificar el último *commit*

En ocasiones podemos olvidarnos de agregar algo a un *commit* y en vez de crear uno nuevo es preferible añadirlo al último.

Esto podemos hacerlo con la orden **git commit --amend**

```
● PS D:\proyectos\guia_git> git add .  
● PS D:\proyectos\guia_git> git commit --amend  
[master 0fe7801] Color de fondo  
Date: Wed Sep 14 17:35:51 2022 +0200  
1 file changed, 4 insertions(+)
```

Al modificarlo podemos cambiar el mensaje del *commit* o mantenerlo.



# 2

SINCRONIZAR  
CON  
REPOSITORIO  
REMOTO



git

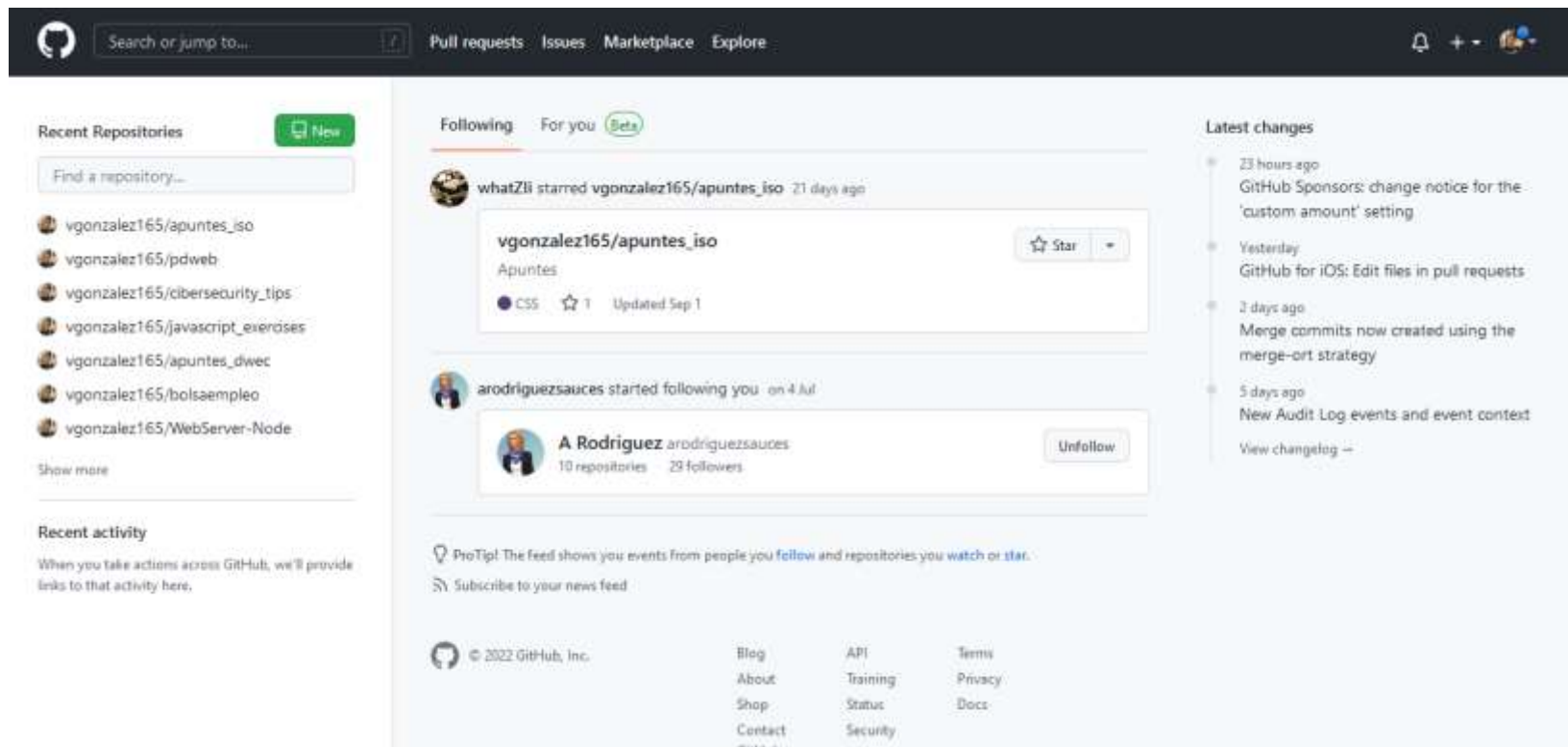
Hasta aquí hemos trabajado con nuestro **repositorio local**, pero podemos conectarnos a un **repositorio remoto** en el que guardaremos también nuestros *commits*.

Los más conocidos son GitHub, Bitbucket y GitLab. En este curso utilizaremos el primero



**github**  
SOCIAL CODING

Tras iniciar sesión vemos una página parecida a esta, donde lo más importante está a la izquierda donde se mostrarán todos los repositorios asociados a nuestra cuenta.



Tras iniciar sesi3n vemos una ṕgina parecida a esta, donde lo ḿs importante est́ a la izquierda donde se mostrarán todos los repositorios asociados a nuestra cuenta.

Recent Repositories



Find a repository...

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

---

### Repository template

Start your repository with a template repository's contents.

No template ▼

---

Owner \*



vgonzalez165 ▼



Repository name \*

guia\_git



Great repository names are short and memorable. Need inspiration? How about [expert-octo-guide?](#)

Description (optional)

Haciendo ejercicios para aprender Git



### Public

Anyone on the internet can see this repository. You choose who can commit.



### Private

You choose who can see and commit to this repository.

---

### Initialize this repository with:

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



#### Add a README file

This is where you can write a long description for your project. [Learn more.](#)

### Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▼

### Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▼

---

Tras darle a crear tendremos un repositorio remoto en blanco. Tenemos tres posibilidades:

- Crear un repositorio en blanco en nuestro equipo y sincronizarlo con este.

**...or create a new repository on the command line**

```
echo "# guia_git" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:vgonzalez165/guia_git.git
git push -u origin main
```

- Subir un repositorio local que tengamos y sincronizarlo con este.

### ...or push an existing repository from the command line

```
git remote add origin git@github.com:vgonzalez165/guia_git.git  
git branch -M main  
git push -u origin main
```

- O importar el código desde otro sistema de control de versiones.

### ...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code



También podemos escoger si el acceso al repositorio remoto lo vamos a hacer mediante HTTP o SSH

Quick setup — if you've done this kind of thing before



Set up in Desktop

or

HTTPS

SSH

`git@github.com:vgonzalez165/guia_git.git`

Como ya tenemos nuestro repositorio en local, escogemos la segunda opción, así que copiamos el código que nos indica en la terminal.

```
PS D:\proyectos\guia_git> git remote add origin https://github.com/vgonzalez165/g
● uia_git.git
● PS D:\proyectos\guia_git> git branch -M main
● PS D:\proyectos\guia_git> git push -u origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (11/11), 1.16 KiB | 594.00 KiB/s, done.
Total 11 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/vgonzalez165/guia_git.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Si recargamos el navegador podemos ver que nuestro código ya está en el repositorio remoto.

The screenshot shows a GitHub repository page for 'vgonzalez165 / guia\_git' (Private). The repository is on the 'main' branch, has 1 branch and 0 tags. The commit history shows 4 commits. The latest commit, 0fe7801, made 1 hour ago, includes changes to 'index.html', 'script.js', and 'styles.css'. A blue banner at the bottom suggests adding a README.

vgonzalez165 / guia\_git (Private) Unwatch

<> Code Issues Pull requests Actions Projects Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

vgonzalez165 Color de fondo 0fe7801 1 hour ago 4 commits

index.html	Primer commit	1 hour ago
script.js	Creado fichero script.js	1 hour ago
styles.css	Color de fondo	1 hour ago

Add a README with an overview of your project. Add a README

## Sincronizar el repositorio local al remoto

Se pueden subir todos los cambios que tengo en el repositorio local al remoto con la orden **git push**. Esto hará que se suban todos los *commits* que se han hecho desde el último *push*.

```
● PS D:\proyectos\guia_git> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 24.79 KiB | 12.40 MiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/vgonzalez165/guia_git.git
   0fe7801..7ddadd  main -> main
```

## Sincronizar el repositorio remoto al local

Una característica de Git es que puede haber múltiples desarrolladores, cada uno con su repositorio local, y todos sincronizados al mismo repositorio remoto.

En esos casos necesitaremos descargar en nuestro equipo los cambios que hayan subido otros desarrolladores, y eso lo haremos con la orden **git pull**.

```
● PS D:\proyectos\guia_git> git pull
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 688 bytes | 57.00 KiB/s, done.
From https://github.com/vgonzalez165/guia_git
   7ddaddd..0ea1fe7  main      -> origin/main
Updating 7ddaddd..0ea1fe7
Fast-forward
 README.md | 4 ++++
 1 file changed, 4 insertions(+)
 create mode 100644 README.md
```

# 3

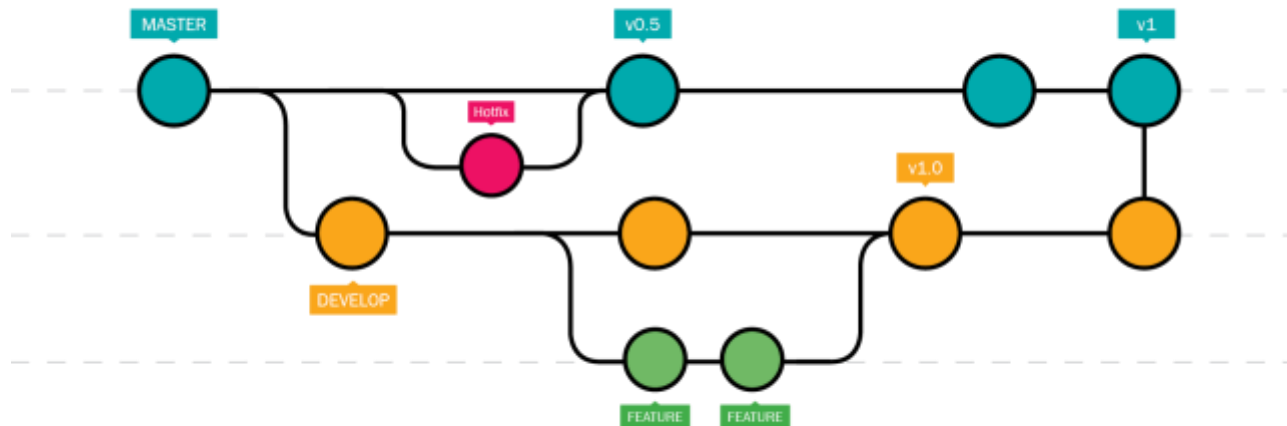
RAMAS CON  
GIT



git

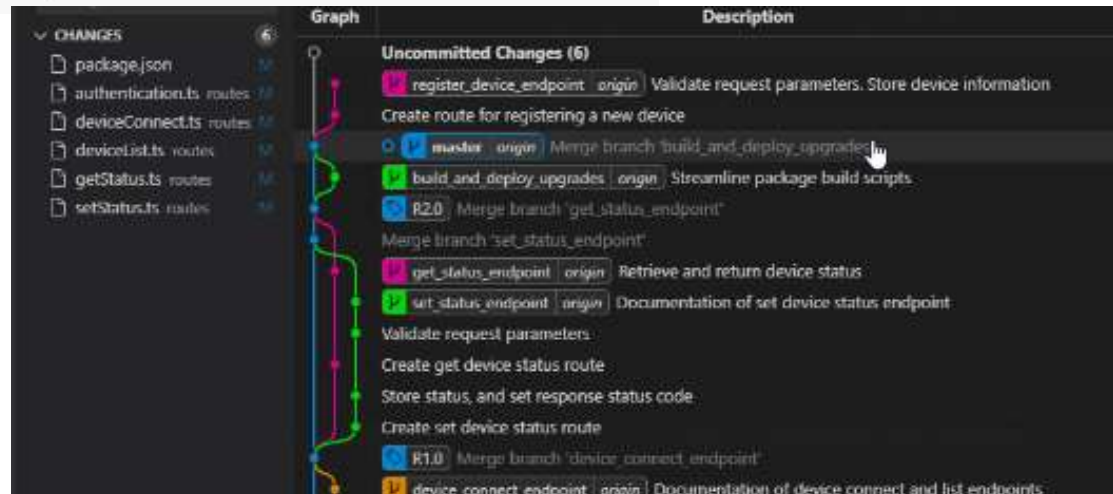
Una de las características más importantes de Git es la posibilidad de crear **ramas** o **bifurcaciones del código**.

Por ejemplo, un desarrollador puede implementar alguna funcionalidad experimental en una rama sin necesidad de modificar el código de la rama principal e incorporarlo únicamente cuando esté plenamente funcional.





Ahora que vamos a trabajar con ramas tal vez sea interesante instalar alguna extensión de VisualStudio que las muestre de forma gráfica, como **Git Graph**



## Ver las ramas del repositorio

Podemos ver las ramas que tenemos en el repositorio con **git branch**.

```
● PS D:\proyectos\guia_git> git branch  
* main
```

Como aún no hemos creado ninguna rama en nuestro proyecto únicamente está la rama principal, que por defecto se denomina **master**.

GitHub está promoviendo la denominación de esta rama como **main**, tal como se ve en el código superior.

## Creación de una rama

Creamos una nueva rama con **git branch <nombre\_rama>**

```
● PS D:\proyectos\guia_git> git branch developer
● PS D:\proyectos\guia_git> git branch
  developer
* main
```

Con **git show-branch** vemos los *commits* de cada rama

```
● PS D:\proyectos\guia_git> git show-branch
! [developer] Create README.ms
* [main] Create README.ms
--
+* [developer] Create README.ms
```

## Movernos entre ramas

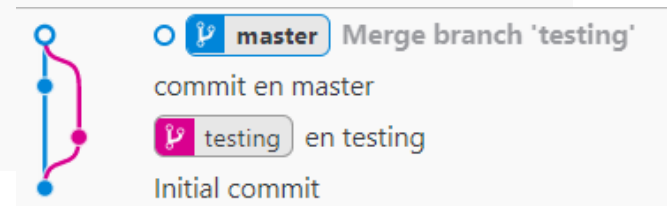
Para movernos entre ramas usaremos **git checkout** **<nombre\_rama>**

```
● PS D:\proyectos\guia_git> git branch
  developer
* main
● PS D:\proyectos\guia_git> git checkout developer
Switched to branch 'developer'
● PS D:\proyectos\guia_git> git branch
* developer
  main
```

## Fusión de ramas

Con el tiempo llegará el momento de fusionar los cambios con la rama principal. Esto se hace con **git merge** <nombre\_rama>. Hay que tener la precaución de **estar en la rama principal** antes de hacer esto.

- PS D:\proyectos\guia\_git> git branch  
\* master  
testing
- PS D:\proyectos\guia\_git> git merge testing  
Merge made by the 'ort' strategy.  
testing.html | 12 ++++++++  
1 file changed, 12 insertions(+)  
create mode 100644 testing.html



En ocasiones habrá **conflictos** al fusionar, por ejemplo, si hay cambios en el mismo fichero en dos ramas diferentes.

```
❌ PS D:\proyectos\guia_git> git merge testing -m "Fusionando"
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
○ PS D:\proyectos\guia_git> █
```

```
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
<<<<<<< HEAD | (Current Change)
|
|   <h1>Estoy en MASTER</h1>
|
=====
|   <h1>Estoy en TESTING</h1>
|   >>>>>>> testing (Incoming Change)
```

## Subir la rama al remoto

Si intentamos subir una rama que hemos creado a GitHub veremos que nos da error.

```
⊗ PS D:\proyectos\guia_git> git push
fatal: The current branch experimental has no upstream branch.
To push the current branch and set the remote as upstream, use
```

```
git push --set-upstream origin experimental
```


```
To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.
```

Si queremos subir una rama tenemos que hacerlo con la orden **git push -u origin <nombre\_rama>**.

```
● PS D:\proyectos\guia_git> git push -u origin experimental
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 314 bytes | 314.00 KiB/s, done.
Total 3 (delta 0), reused 1 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'experimental' on GitHub by visiting:
remote:   https://github.com/vgonzalez165/guia_git/pull/new/experimental
remote:
To https://github.com/vgonzalez165/guia_git.git
 * [new branch]      experimental -> experimental
branch 'experimental' set up to track 'origin/experimental'.
```

Esta operación solo hay que hacerla una vez, quedando ya vinculada la rama al origen para sucesivos *pushes*.



 **experimental** had recent pushes 2 minutes ago

[Compare & pull request](#)

 **main** ▾  2 branches  0 tags


[Go to file](#)

[Add file ▾](#)

[Code ▾](#)



**vgonzalez165** Cambios en el MASTER

66e8dd1 6 minutes ago  8 commits



index.html

Cambios en el MASTER


6 minutes ago

Add a README with an overview of your project.

[Add a README](#)

#### Default branch



**main**  Updated 6 minutes ago by vgonzalez165

Default



#### Your branches

**experimental**  Updated 5 minutes ago by vgonzalez165

0 | 1

[New pull request](#)



#### Active branches

**experimental**  Updated 5 minutes ago by vgonzalez165

0 | 1

[New pull request](#)



## Descarga de ramas en remoto

En ocasiones trabajamos en un equipo sobre el mismo repositorio y otro miembro ha subido una rama.

Si queremos descargar esa rama debemos indicar expresamente con el comando **git fetch**.

# 4

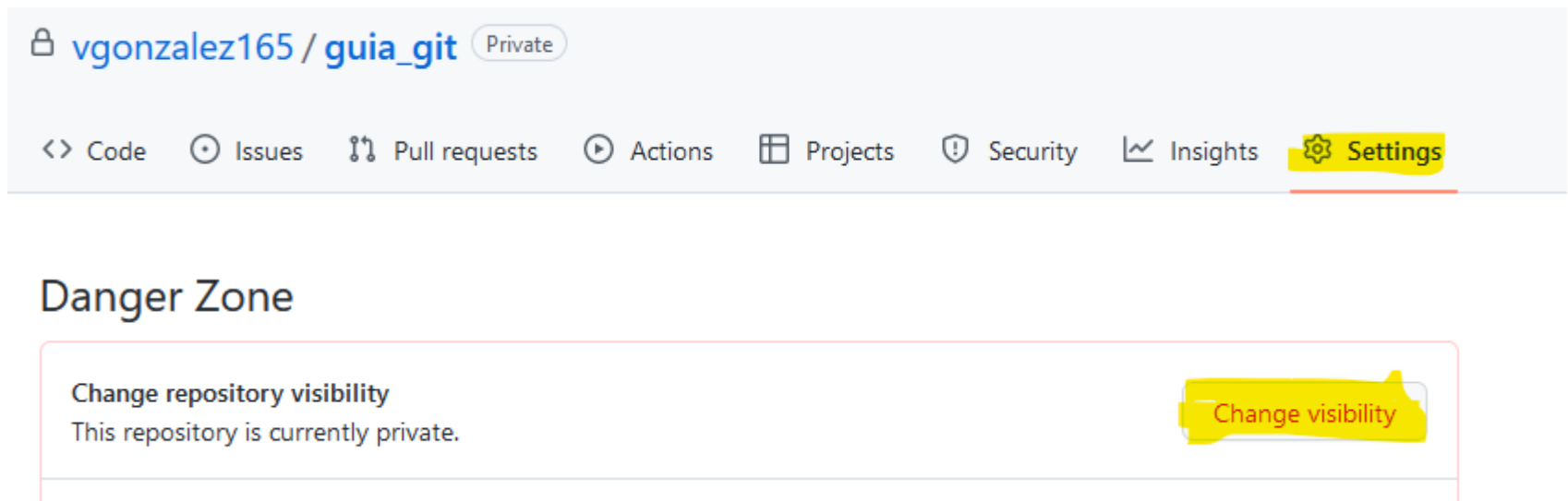
PUBLICAR EN  
GITHUB PAGES



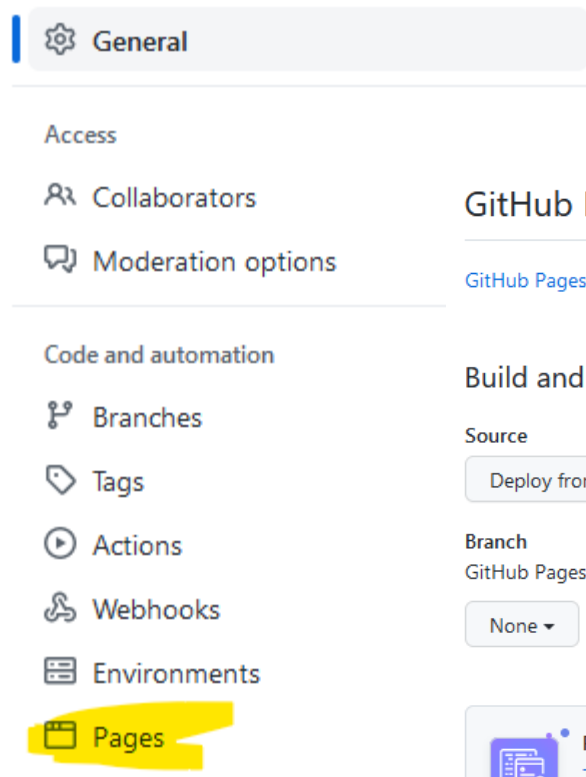
git

GitHub dispone de un servicio llamado **GitHub Pages** que permite alojar páginas web estáticas para ser publicadas en Internet.

Lo primero que debemos hacer es asegurarnos de que nuestro repositorio es **público**.



A continuación vamos a la opción **Pages** del menú principal.



## GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

### Build and deployment

#### Source

Deploy from a branch ▼

#### Branch

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more.](#)

None ▼

Save



Publish privately to people with read access to this repository

Try risk-free for 30 days using a GitHub Enterprise organization, or [learn more about changing the visibility of your GitHub Pages site.](#) ✕

Aquí simplemente escogemos qué rama queremos publicar y si el fichero **index.html** se encuentra en el directorio raíz o en algún subdirectorio.

#### Branch

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more.](#)

 main ▾  / (root) ▾ Save

Select branch

Select branch

✓ main

experimental

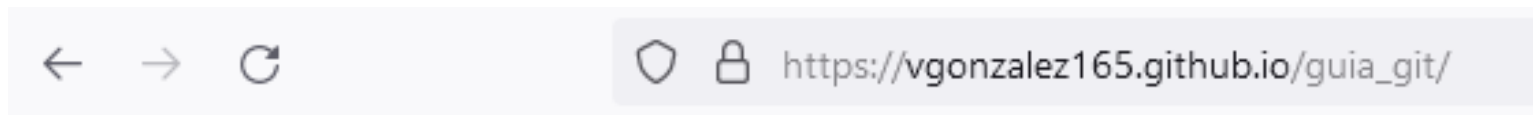
None

ad access to this repository

terprise organization, or [learn more about changing the visibility of your GitHub](#)

Y ya tendremos la ṕgina disponible en la URL de la forma

**`https://<nombre_usuario>.github.io/<nombre_repositorio>`**



## **GitHub Pages**

**Esto es lo que se puede publicar**

# 5

## MARKDOWN



git