

# Plataformas online para git

## GitHub

Si bien hay varias plataformas online para trabajar con git (y nada nos impide crear una nosotros), vamos a nombrar las tres más difundidas:

- GitHub ( <https://github.com> )
- GitLab ( <https://about.gitlab.com/> )
- BitBucket ( <https://bitbucket.org/product> )

En este apunte, se siguen los pasos para trabajar con GitHub<sup>1</sup>, pero cada estudiante puede explorar el uso de otra plataforma si lo desea.

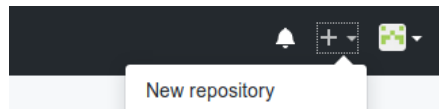
## Creando una cuenta en github

El procedimiento para crear una cuenta en GitHub, no difiere demasiado del que se utiliza para cualquier otro servicio Web, como el correo electrónico o las redes sociales. Dejamos a cargo de cada estudiante la creación de su propia cuenta. Es importante que creamos una cuenta por cada estudiante, y no una por grupo de trabajo.

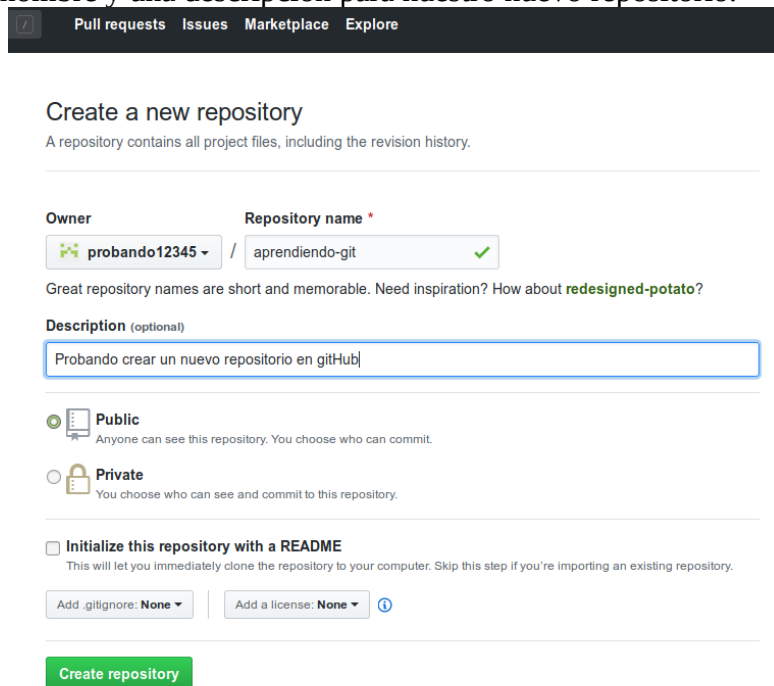
## Creando un repositorio en tu cuenta de GitHub

Una vez creada la cuenta de GitHub y verificado nuestro mail, estaremos en condiciones de crear nuestro primer repositorio.

1) Vamos al signo + que encontraremos arriba a la derecha, y allí elegimos “New repository”.



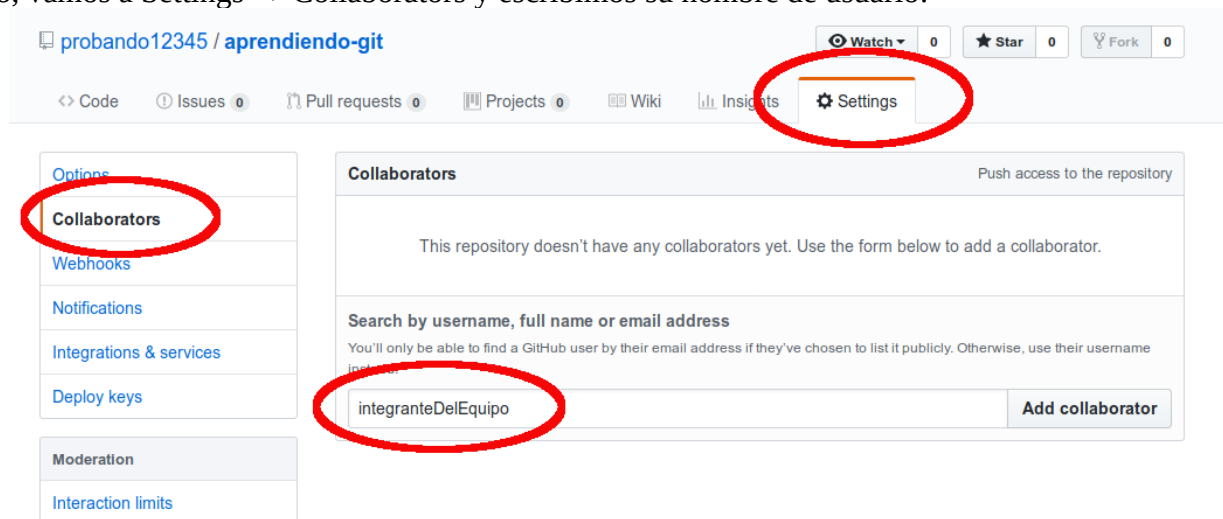
2) Luego, elegiremos un nombre y una descripción para nuestro nuevo repositorio.

A screenshot of the 'Create a new repository' form on GitHub. The form has a dark header with navigation links: 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the header, the title 'Create a new repository' is followed by a subtitle 'A repository contains all project files, including the revision history.' The form contains several fields: 'Owner' (a dropdown menu showing 'probando12345'), 'Repository name' (a text input field with 'aprendiendo-git' and a green checkmark), 'Description' (a text input field with 'Probando crear un nuevo repositorio en github'), 'Public/Private' (radio buttons with 'Public' selected), 'Initialize this repository with a README' (a checkbox), 'Add .gitignore' (a dropdown menu with 'None'), and 'Add a license' (a dropdown menu with 'None'). At the bottom is a green 'Create repository' button.

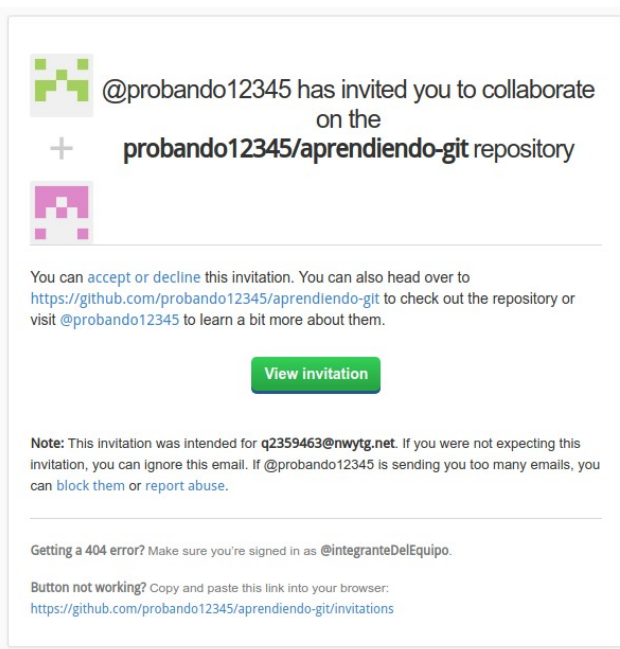
---

<sup>1</sup> La pronunciación es algo parecido a /guítjab/

3) Ahora, la persona que ha creado el repositorio, puede autorizar a sus compañeros de equipo a escribir en él. Para eso, vamos a Settings → Collaborators y escribimos su nombre de usuario:



4) El compañero invitado debe revisar su correo electrónico, donde le llegará la invitación.



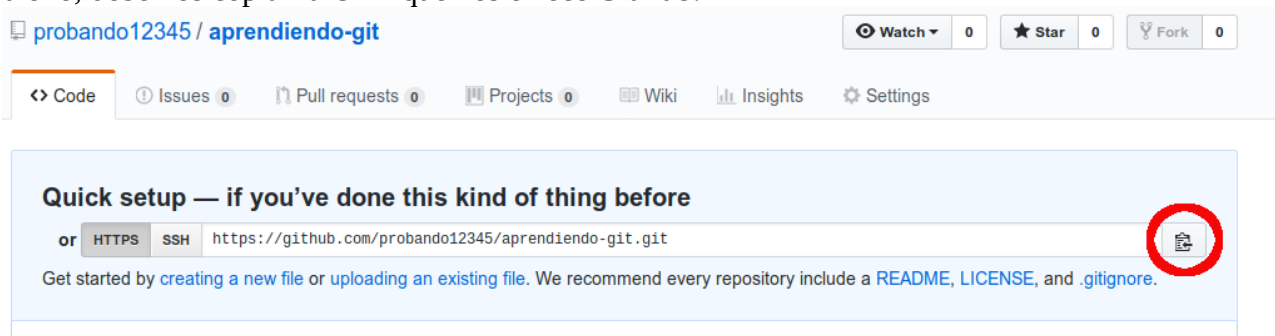
5) Al hacer clic en el enlace, podemos aceptar o declinar la invitación. Aceptaremos.

6) Debemos repetir los pasos 3; 4 y 5 con el resto de los integrantes del equipo.

## Clonar localmente un repositorio de GitHub

Clonar es simplemente copiar el repositorio de GitHub en nuestro entorno de trabajo local.

1) Para ello, debemos copiar la URL que nos ofrece GitHub:



2) Con esa URL, volveremos a la terminal de GNU/Linux, o al GitBash en Windows, y clonaremos el repositorio:

```
usuario@PC ~ $ git clone AAAA
```

donde “AAAA” es la dirección recién copiada en el paso anterior. Por ejemplo:

```
usuario@PC ~ $ git clone https://github.com/probando12345/aprendiendo-git.git
```

Clonando en 'aprendiendo-git'...

warning: Parece haber clonado un repositorio sin contenido.

La salida del comando nos advierte que el repositorio no tiene contenido, porque aún no hemos hecho ningún commit.

3) Nos movemos dentro de la carpeta que contiene el repositorio

```
usuario@PC ~ $ cd aprendiendo-git
```

(Hay que reemplazar aprendiendo-git por el nombre que hayamos elegido para el repositorio. Este comando no tiene salida)

4) Verificamos el estado del repositorio:

```
usuario@PC ~/aprendiendo-git $ git status
```

En la rama master

No hay commits todavía

no hay nada para confirmar (crea/copia archivos y usa "git add" para hacerles seguimiento)

5) Todos los integrantes del equipo pueden repetir los pasos anteriores, para que todos tengan el repositorio.

## Actualizando un repositorio con push

Ahora, haremos que uno de los integrantes del equipo actualice el repositorio local, y que luego *suba* esos cambios a GitHub.

1) Uno de los integrantes del equipo, escribe algo en un archivo cualquiera, y guarda en el directorio de trabajo.

2) Comprobamos el estado del repositorio:

```
usuario@PC ~/aprendiendo-git $ git status
```

En la rama master

No hay commits todavía

Archivos sin seguimiento:

(usa "git add <archivo>..." para incluirlo a lo que se será confirmado)

**unArchivoCualquiera.txt**

no hay nada agregado al commit pero hay archivos sin seguimiento presentes (usa "git add" para hacerles seguimiento)

3) Comenzamos a hacer seguimiento al nuevo archivo:

```
usuario@PC ~/aprendiendo-git $ git add unArchivoCualquiera.txt
```

4) Comprobemos nuevamente el estado del repositorio:

```
usuario@PC ~/aprendiendo-git $ git status
```

En la rama master

No hay commits todavía

Cambios a ser confirmados:

(usa "git rm --cached <archivo>..." para sacar del área de stage)

**nuevo archivo: unArchivoCualquiera.txt**

5) Ahora podemos hacer el primer commit a nuestro repositorio:

```
usuario@PC ~/aprendiendo-git $ git commit -m "Primer commit"
```

1 file changed, 1 insertion(+)

create mode 100644 unArchivoCualquiera.tx

En este punto, nuestro repositorio **local**, está **adelantado** al repositorio remoto (alojado en GitHub). Es decir que nuestro repositorio local registra un commit que nuestro repositorio GitHub aún no conoce.

6) Para poner al día nuestro repositorio, podemos usar el comando **git push**:

```
usuario@PC ~/aprendiendo-git $ git push
```

Username for 'https://github.com': probando12345

Password for 'https://probando12345@github.com':

Contando objetos: 3, listo.

Delta compression using up to 4 threads.

Comprimiendo objetos: 100% (2/2), listo.

Escribiendo objetos: 100% (3/3), 232 bytes | 232.00 KiB/s, listo.

Total 3 (delta 0), reused 0 (delta 0)

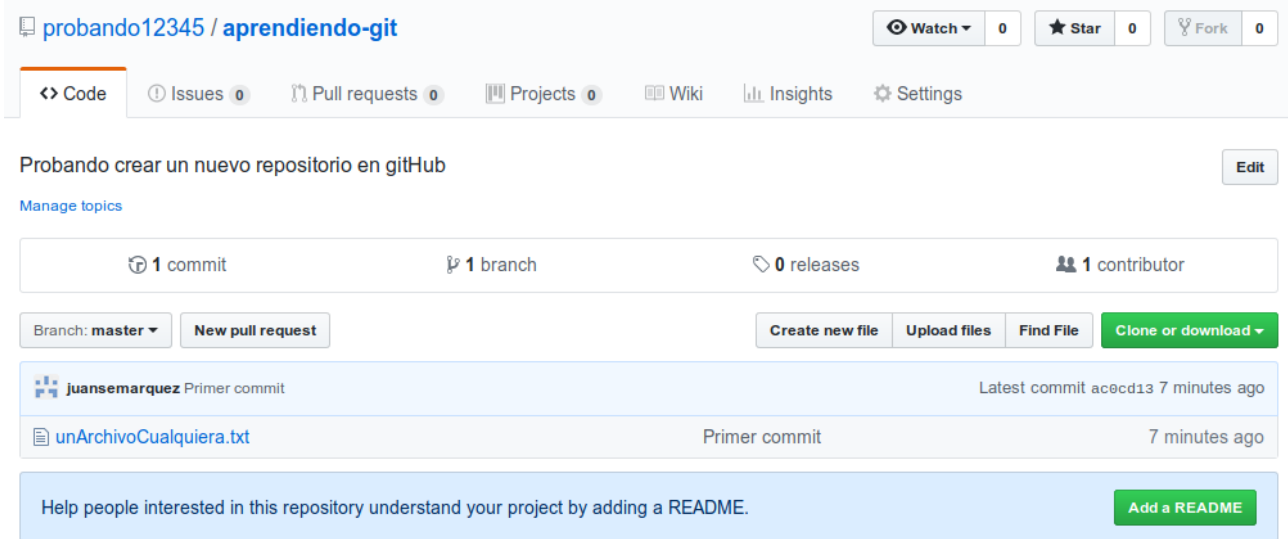
To https://github.com/probando12345/aprendiendo-git.git

\* [new branch] master -> master

Lo que está subrayado es el nombre de usuario de GitHub. También debemos ingresar la contraseña. (Puede ocurrir que, mientras escribimos la contraseña, nada cambia en la pantalla de la terminal. Esto es normal.).

Una vez ingresado el usuario y la contraseña de GitHub, el sistema procede a actualizar el repositorio remoto.

7) Podemos comprobarlo visitando nuevamente la página de GitHub:



Vemos que ahora se ve reflejado el cambio en GitHub.

## Actualizando el repositorio local con git pull

Este apartado lo realizaremos con otro usuario, distinto al que realizó el apartado anterior.

Supondremos que el usuario había clonado el repositorio **antes** de que se hiciera el git push del paso 6) del apartado anterior.

1) Le pediremos al sistema que “se ponga al día” con el estado del repositorio remoto:

```
integrante@PC ~/aprendiendo-git $ git remote update
```

Extrayendo origin

remote: Enumerating objects: 5, done.

remote: Counting objects: 100% (5/5), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0

Desempaquetando objetos: 100% (3/3), listo.

Desde https://github.com/probando12345/aprendiendo-git

ac0cd13..252f898 master -> origin/master

Este comando no modifica en absoluto el estado de nuestro repositorio, sino que simplemente se anoticia de los cambios que hubieran ocurrido en GitHub.

2) Comprobemos el estado del repositorio:

```
integrante@PC ~/aprendiendo-git $ git status
```

En la rama master

Tu rama está detrás de 'origin/master' por 1 commit, y puede ser avanzada rápido.

(usa "git pull" para actualizar tu rama local)

nada para hacer commit, el árbol de trabajo está limpio

Se nos informa que estamos “detrás” de origin, es decir, que hay commits que nuestro repositorio remoto tiene guardados, pero que nuestro repositorio local aún no ha registrado.

3) Podemos actualizar el repositorio local, para que quede idéntico al repositorio remoto, con git pull:

```
integrante@PC ~/aprendiendo-git $ git pull
```

Actualizando ac0cd13..252f898

Fast-forward

unArchivoCualquiera.txt | 1 +

1 file changed, 1 insertion(+)

Ahora, el repositorio local ha quedado idéntico al repositorio remoto, por lo que sabemos que estamos trabajando con la última versión.

## Resolución de conflictos

En este punto, puede aparecer la pregunta por los conflictos. ¿Qué pasa si dos colaboradores de un mismo repositorio actualizan las mismas líneas del mismo archivo en el mismo momento? Git tiene un sofisticado mecanismo para resolver estos conflictos, que serán abordados en próximas clases.