

# Git y GitHub

para desarrolladores



# git



# GitHub

# Linus Torvalds

Lanzado el 7 de abril de 2005












Para ayudar en el desarrollo del  
kernel de Linux



# Es un VCS

(Version Control System)

- Sistema de control de versiones
- Un sistema para gestionar los cambios de un proyecto de desarrollo de software
- Mantiene archivos de tus cambios
- Permite el trabajo colaborativo
- Permite saber quien hizo que cambios y que
- Te permite viajar en el tiempo y revertir cualquier cambio y regresar a un estado previo

 Metodos	3/15/2018 11:25 A...	File folder
 MetodosN	4/27/2018 11:23 PM	File folder
 MetodosNumericos	3/3/2018 10:47 PM	File folder
 MNum	2/15/2018 6:16 PM	File folder
 M-todosNumericos	3/21/2018 6:41 PM	File folder
 M-todosNumericos-master	2/15/2018 10:13 A...	File folder
 Proyecto Mysql	4/30/2013 8:16 AM	File folder
 ProyectoBD-master	6/3/2018 9:22 PM	File folder
 ProyectoBueno	4/12/2018 11:59 A...	File folder
 ProyectoColas	10/10/2017 11:57 ...	File folder
 ProyectoPilas	10/10/2017 12:31 ...	File folder

# JETBRAINS

Professional desktop IDEs: IntelliJ IDEA, PyCharm, and more

**Benefit** A free subscription for students, to be renewed annually.



Affordable registration, hosting, and domain management

**Benefit** One year SSL certificate (normally \$9/year)

**Benefit** One year domain name registration on the .me TLD (normally \$18.99/year)



Email infrastructure as a service

**Benefit** Student plan 15K free emails/month (normally limited to 200 free emails/day) while you're a student



Track errors in every language, framework, and library

**Benefit** 500,000 events/month with unlimited projects and members while you're a student



Web and mobile payments, built for developers


**Benefit** Waived transaction fees on first \$1000 in revenue processed



Dynamic A/B testing, smart push notifications and custom analytics for native mobile apps

**Benefit** Complete access to the suite of tools for native mobile apps. Unlimited access to the platform free for 6 months.

# Cuenta gratis vs cuenta de desarrollador



Developer

MXN

\$140.98


per month

[\(view in USD\)](#)

**Includes:**

- Personal account
- Unlimited public repositories
- Unlimited private repositories
- Unlimited collaborators

Free for students as part of the [Student Developer Pack](#).



Free

\$0

per month

**Includes:**

- Personal account
- Unlimited public repositories
- Unlimited collaborators

There are millions of public projects on GitHub. Join one or start your own for free.



# Repository (Repositorio)

Es una colección de todos los archivos y el historial de los mismos

Consiste en todos tu **commits**

El lugar en donde todo tu trabajo duro esta almacenado

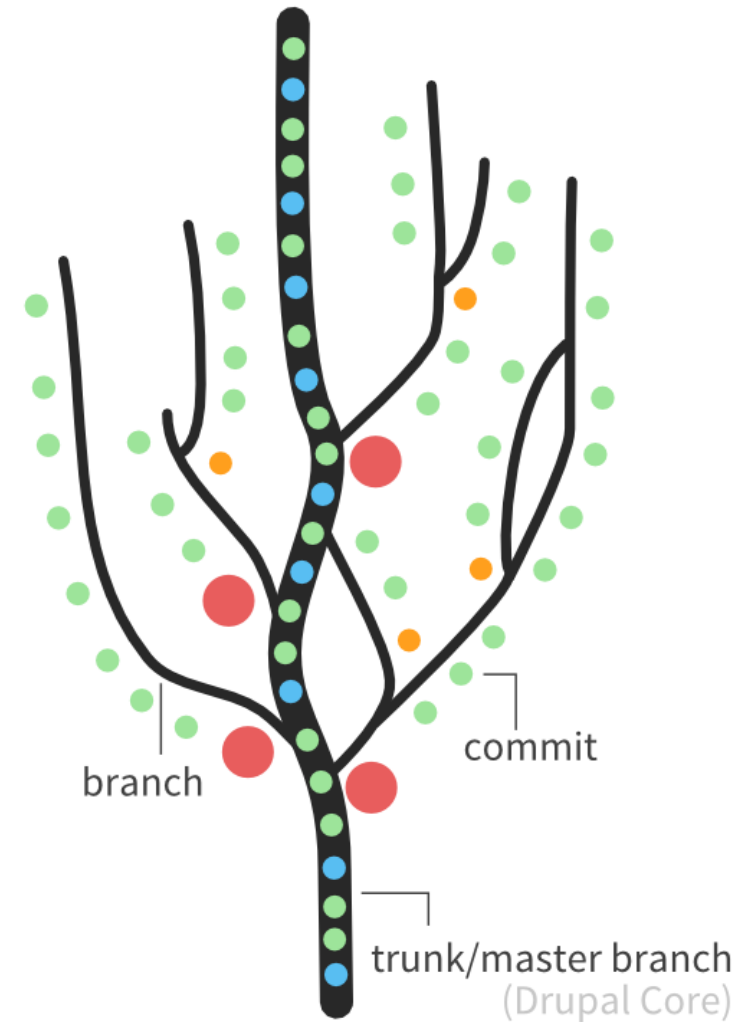
Cuando copiamos un repositorio de un servidor remoto se le llama **cloning ó clone repo**

# Branches (ramas)

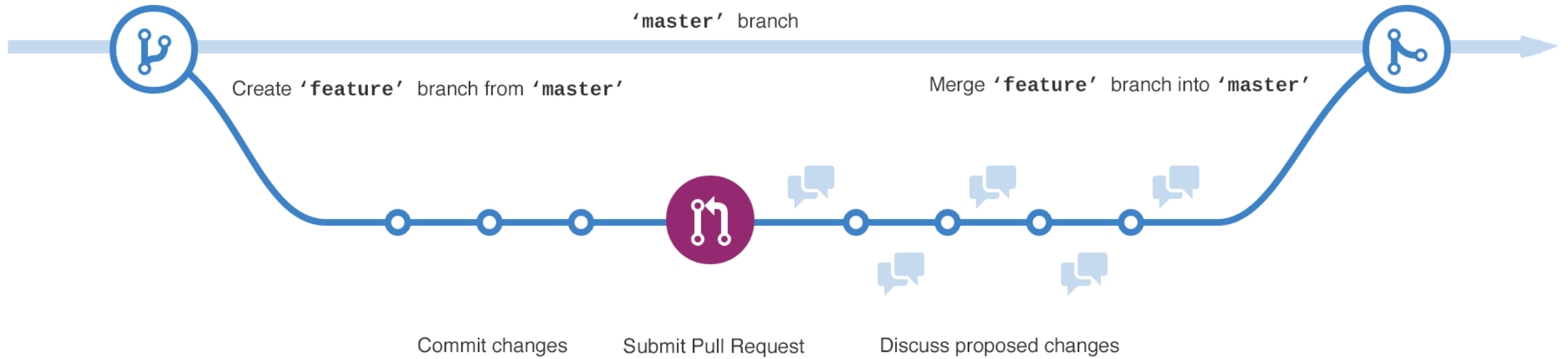
Las ramas en git son versiones en las cuales se almacenan los **commits**.

La rama principal se llama **master**

Pueden haber tantas ramas como se quieran, por lo general cada nueva rama añade una funcionalidad o se arregla un **bug** o problema en el proyecto







# Git Flow

# Buena práctica

Una buena práctica  
es la siguiente:

Utilizar 4 tipos de  
ramas: **Master,**  
**Development,**  
**Features,** y **Hotfix.**

A grayscale background image showing a hand pointing at a document. The document contains a table with multiple rows and columns, and some text below it. The hand is in the foreground, pointing towards the table.

# Master

- Es la rama principal. Contiene el repositorio que se encuentra publicado en producción, por lo que debe estar siempre estable.

# Development

- Es una rama sacada de master. Es la rama de integración, todas las nuevas funcionalidades se deben integrar en esta rama.
- Luego que se realice la integración y se corrijan los errores (en caso de haber alguno), es decir que la rama se encuentre estable, se puede hacer un merge de development sobre la rama master.

A grayscale background image showing a hand pointing at a laptop screen. The screen displays lines of code, likely in a programming language like Python or JavaScript. The hand is in the foreground, with the index finger pointing towards the screen. The laptop is open, and the screen is the primary focus of the hand's gesture.

## Features:

- Cada nueva funcionalidad se debe realizar en una rama nueva, específica para esa funcionalidad. Estas se deben sacar de development.
- Una vez que la funcionalidad esté desarrollada, se hace un merge de la rama sobre development, donde se integrará con las demás funcionalidades.

A grayscale background image showing a hand pointing at a laptop screen. The screen displays lines of code, likely from a version control system like Git. The hand is in the foreground, pointing towards the screen. The overall tone is technical and focused on software development.

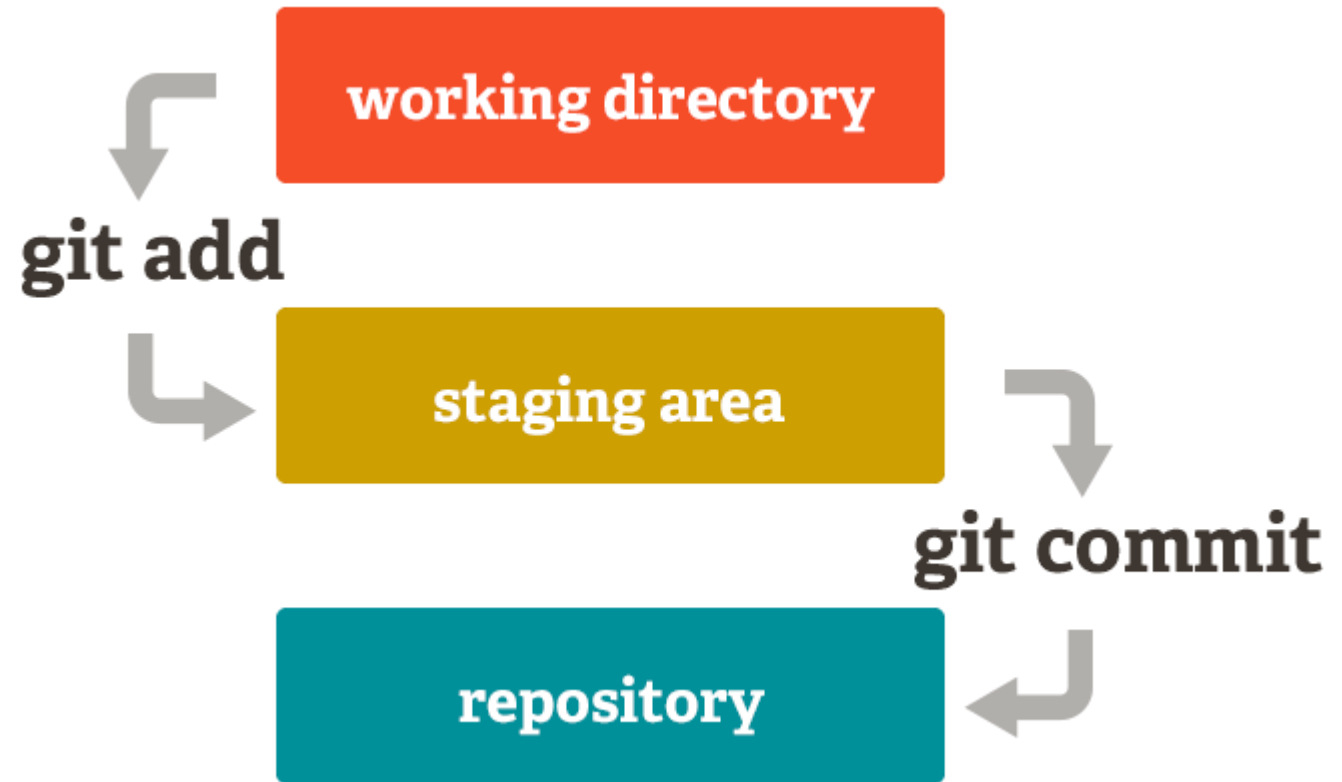
## Hotfix:

- Son bugs que surgen en producción, por lo que se deben arreglar y publicar de forma urgente. Es por ello, que son ramas sacadas de master.
- Una vez corregido el error, se debe hacer un merge de la rama sobre master. Al final, para que no quede desactualizada, se debe realizar el merge de master sobre development.



# Commit

- Almacena los cambios que has hecho en el repositorio de trabajo



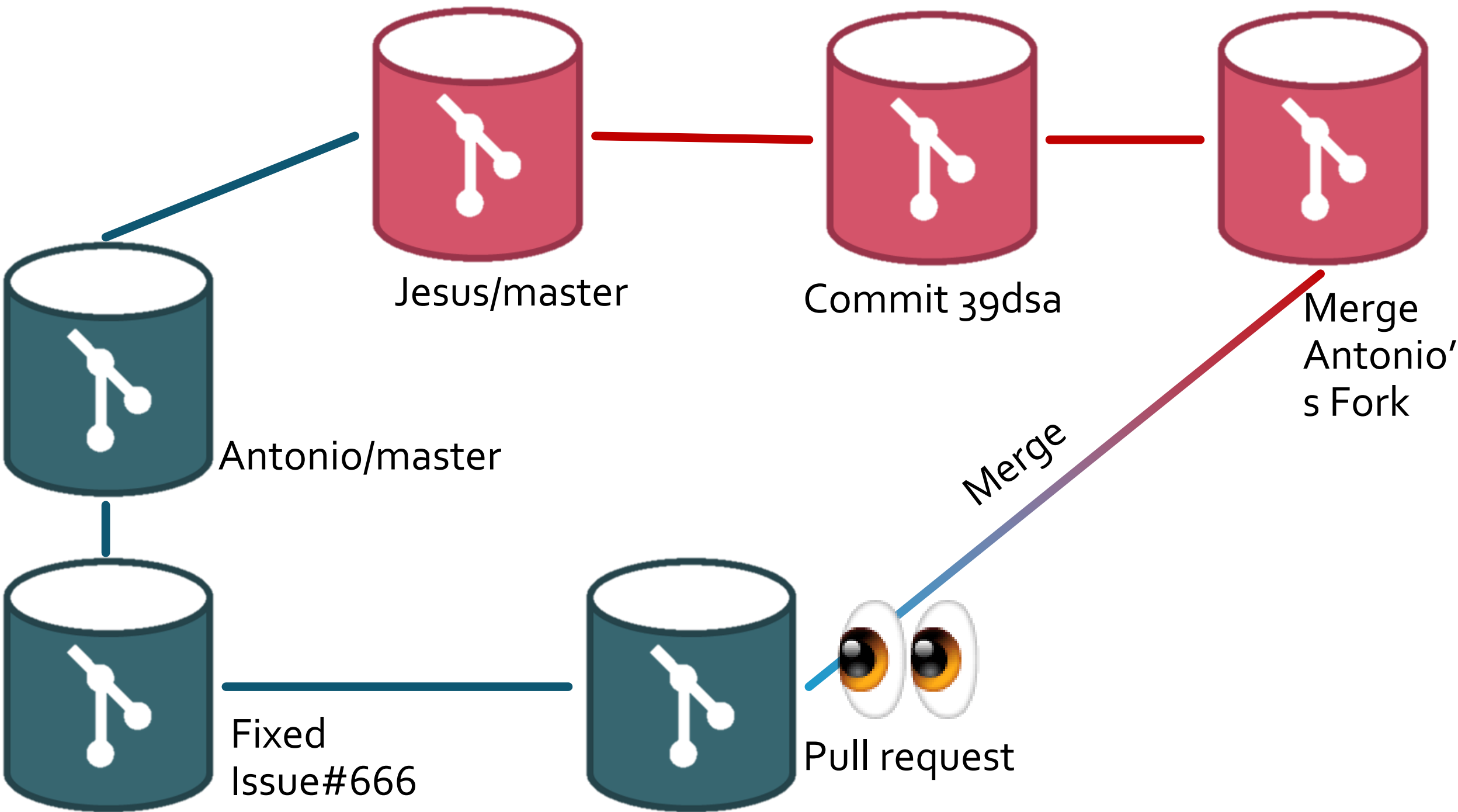
# When you revert a commit



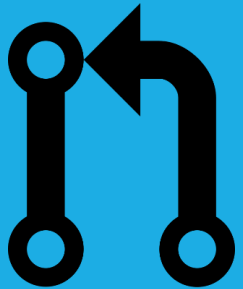


- Fork o bifurcación en español
- Hace una copia exacta del repositorio original que podemos utilizar como un repositorio git cualquiera.
- Tendremos dos repositorios git idénticos pero con distinta URL uno será el nuestro
- Este repositorio tendrá exactamente la misma historia que en el momento en el que lo hayamos **forkeado**

 **Antonio072 / javascript-algorithms**  
forked from [trekhleb/javascript-algorithms](#)



## Pull request



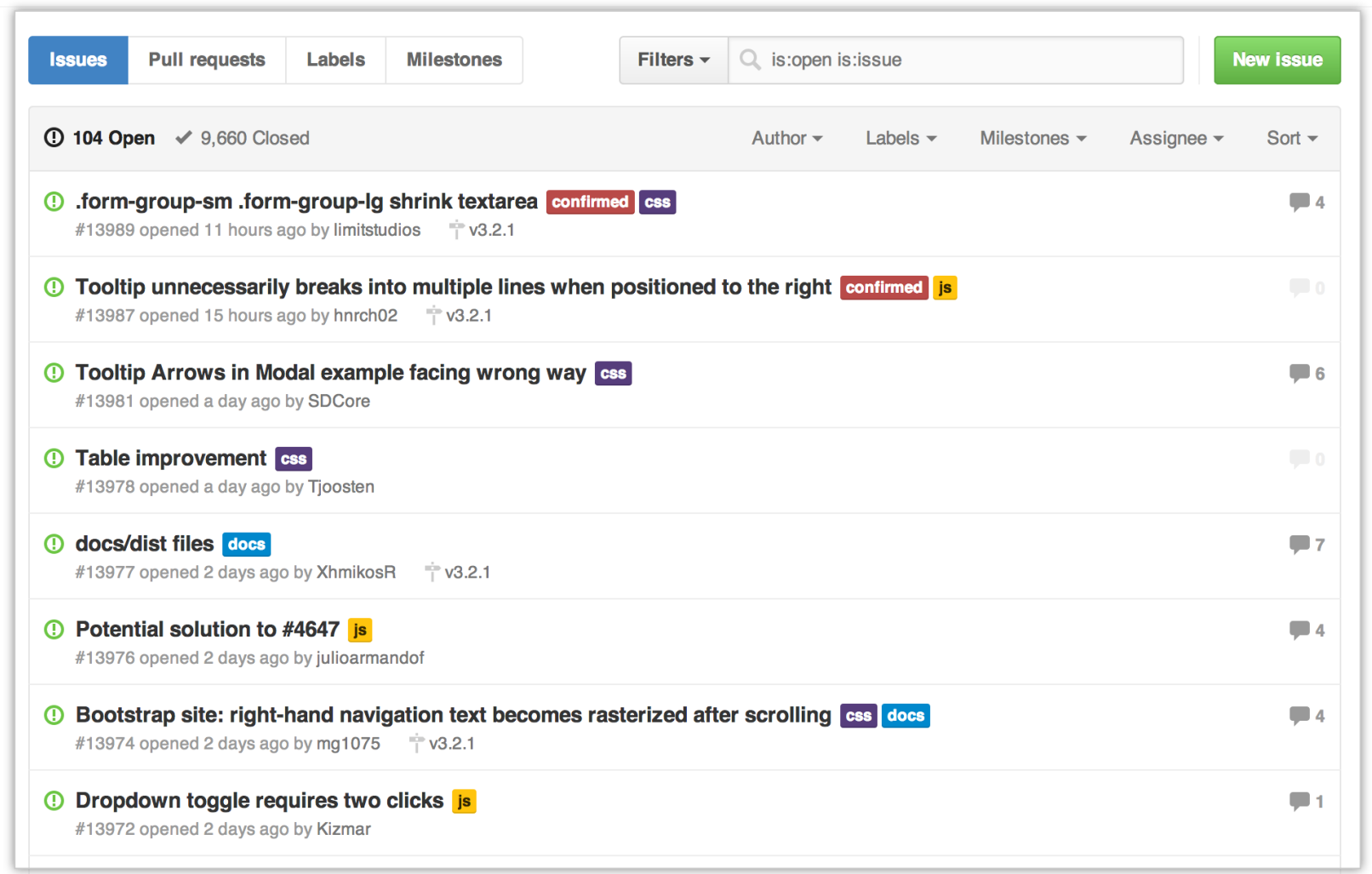
Es una petición para que los propietarios de un **repositorio** incorporen los cambios que hemos realizado.

Una parte fundamental cuando se unen 2 **branches** debido a que permite la retroalimentación en nuestros cambios

# Issues

Son una gran manera de mantener el seguimiento de las tareas, mejoras y bugs para los proyectos.

Cada **repo** tiene su propia seccion



The screenshot shows the GitHub Issues interface. At the top, there are tabs for 'Issues', 'Pull requests', 'Labels', and 'Milestones'. To the right of these tabs is a search bar with the text 'is:open is:issue' and a 'New Issue' button. Below the tabs, the main content area displays a list of issues. The first issue is '#13989 .form-group-sm .form-group-lg shrink textarea' with labels 'confirmed' and 'css'. The second issue is '#13987 Tooltip unnecessarily breaks into multiple lines when positioned to the right' with labels 'confirmed' and 'js'. The third issue is '#13981 Tooltip Arrows in Modal example facing wrong way' with label 'css'. The fourth issue is '#13978 Table improvement' with label 'css'. The fifth issue is '#13977 docs/dist files' with label 'docs'. The sixth issue is '#13976 Potential solution to #4647' with label 'js'. The seventh issue is '#13974 Bootstrap site: right-hand navigation text becomes rasterized after scrolling' with labels 'css' and 'docs'. The eighth issue is '#13972 Dropdown toggle requires two clicks' with label 'js'. Each issue entry includes the issue number, title, labels, the user who opened it, the time it was opened, and the number of comments.

Issue Number	Title	Labels	Author	Time	Comments
#13989	.form-group-sm .form-group-lg shrink textarea	confirmed, css	limitstudios	11 hours ago	4
#13987	Tooltip unnecessarily breaks into multiple lines when positioned to the right	confirmed, js	hnrch02	15 hours ago	0
#13981	Tooltip Arrows in Modal example facing wrong way	css	SDCore	a day ago	6
#13978	Table improvement	css	Tjoosten	a day ago	0
#13977	docs/dist files	docs	XhmikosR	2 days ago	7
#13976	Potential solution to #4647	js	julioarmandof	2 days ago	4
#13974	Bootstrap site: right-hand navigation text becomes rasterized after scrolling	css, docs	mg1075	2 days ago	4
#13972	Dropdown toggle requires two clicks	js	Kizmar	2 days ago	1





**Ahora si se viene lo chido**

# Command Line

Primero vamos a instalar Git  
<https://git-scm.com/downloads>



# Configurar nuestro usuario y contraseña

Abriremos la ventana de comandos

Añadiremos nuestro usuario

```
git config --global user.name "tu usuario sin comillas"
```

Podemos confirmar el usuario que introdujimos con:

```
git config --global user.name
```

Añadiremos nuestra dirección de correo

```
git config --global user.email "tu correo"
```

Añadiremos la contraseña

```
git config --global user.password "tu contraseña"
```



# Clone repo

Podemos clonar todo un repositorio y hacerlo nuestro

Si en el repositorio no tenemos permisos de escritura o no somos colaboradores nos mostrará un mensaje de acceso denegado,

Podemos clonar todos los repositorios que estén públicos pero no podremos modificarlos, para ello es mejor usar la opción fork

```
MINGW64:/c/Users/usuario/Desktop/New folder/testable-projects-fcc
usuario@Lenovo MINGW64 ~/Desktop/New folder
$ ls
testable-projects-fcc/

usuario@Lenovo MINGW64 ~/Desktop/New folder
$ cd testable-projects-fcc/

usuario@Lenovo MINGW64 ~/Desktop/New folder/testable-projects-fcc (master)
$ git commit -m "First commit"
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

usuario@Lenovo MINGW64 ~/Desktop/New folder/testable-projects-fcc (master)
$ git push
remote: Permission to freeCodeCamp/testable-projects-fcc.git denied to Antonio072.
fatal: unable to access 'https://github.com/freeCodeCamp/testable-projects-fcc.git/': The requested URL returned error: 403

usuario@Lenovo MINGW64 ~/Desktop/New folder/testable-projects-fcc (master)
$ |
```



Cuando colaboramos en un repositorio, y es nuestro o somos colaboradores podemos hacer un **pull** (jalar los cambios de una rama) o un **push** (subir los cambios guardados de nuestros **commits**)



De lo contrario si clonamos un repositorio o lo **forkeamos** tendremos que hacer un **pull request**

`git request-pull v1.0`  
<https://git.ko.xz/project/master>



Hacer un **pull** es conveniente si no se han realizado cambios en la rama la cual clonamos y de la original si, esto para no tener conflictos al realizar un **merge**

# Pull/push

# Git init

Podemos inicializar un repositorio en cualquier parte de nuestra computadora desde la consola de comandos,.

Al utilizar el commando **git init** le indicamos al Sistema que todo el contenido que existen dentro de esa carpeta forma un **repositorio**





# Git add

Cuando realizamos un **git init** por defecto nos marca que ningún archivo dentro de esa carpeta será subido al repositorio o será guardado en un **commit**, todos los que no serán subidos serán señalados en la consola

```
pip-18.0-py2.py3-none-any.whl
pyaudio-0.1.0.exe
pyaudio-0.2.8.py33.exe
pythonbook.pdf
reporte simulacion 2.docx
setup-x86_64.exe
simulacion(1).zip
simulacion.zip
solucionarioecuacionesdiferencialesdenniszill7aedicion-120713235939-phpa
pp01.pdf
talkative copybase.PSD
tc500setup(1).exe
tc500setup.exe
temario curso online pdf.pdf
tiristores.pptx
vs_community__1561936280.1535847941.exe
wordtojpeg.zip
~$Day of the Dead.ppt
~$RetoCE.pptx

nothing added to commit but untracked files present (use "git add" to track)

usuario@Lenovo MINGW64 ~/Downloads (master)
$ |
```

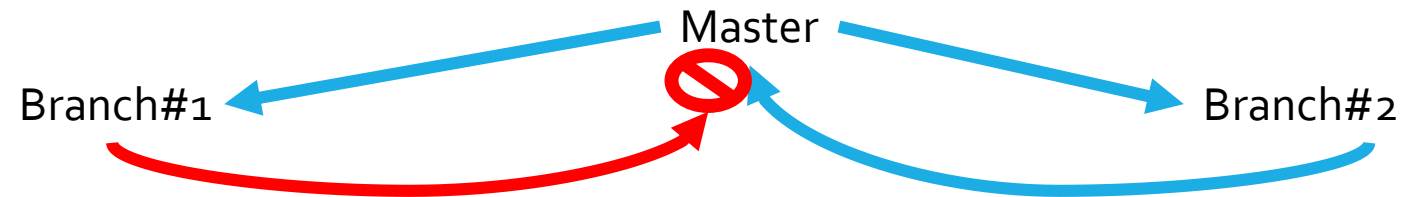
# Merge conflict

Cuando creamos una nueva rama a partir de otra ya sea **master** o cualquiera, podemos volver hacer una **unión** o **merge** para combinar los cambios que hemos realizado con la rama objetivo

Ocurre un conflicto cuando la rama original de la cual hemos creado otra se ha visto modificada antes de que subamos nuestros cambios esto es:

Pasos:

1. Se crea la rama maestra
2. La rama 1 y 2 se crean al mismo tiempo
3. En ambas ramas se realizan cambios
4. La rama 2 sube sus cambios a la master antes de que lo haga la rama 1
5. Al querer subir los cambios de la rama 1 a master esta difiere en cuanto a su estado "original"



# Git remotes

- Podemos agregar un repositorio local y subirlo a alguna plataforma alojada a internet para poder tener acceso a ellos desde cualquier lugar.
- Para ellos solo necesitamos:
  1. Ir a la carpeta en donde tengamos nuestro repositorio (si no lo tenemos podemos hacer un **git init** para inicializar uno nuevo)
  2. Agregar todos los archivos que deseamos subir de acuerdo a la siguiente tabla

	New Files	Modified Files	Deleted Files	
<code>git add -A</code>	✓	✓	✓	Stage All (new, modified, deleted) files
<code>git add .</code>	✓	✓	✓	Stage All (new, modified, deleted) files
<code>git add --ignore-removal .</code>	✓	✓	✗	Stage New and Modified files only
<code>git add -u</code>	✗	✓	✓	Stage Modified and Deleted files only

# Git remotes

3. Una vez que hayamos agregado los archivos hacemos un commit  
`git commit -m "Primer commit"`
4. Agregamos el repositorio remote  
`git remote add origin`  
<https://github.com/usuario/repo.git>
5. Hacemos el **push** hacia el repositorio remote  
`Git push -u origin master`

**VOILÀ !**







**GitHub** Pages


# Github pages

- Creamos un Nuevo repo

Read the guide

Start a project

- Debera tener el siguiente nombre **usuario.github.io**

 Antonio072 ▾ / antonio072.github.io| ✓

Great repository names are short and memorable. Need inspiration? How about **crispy-potato**.

- Una vez creado subimos el contenido de nuestra página clonando el repositorio que hemos creado, añadimos todo el contenido y **pusheamos el repo**
- Y visitamos la página