



**PUCP**

# Desarrollo Web con Python

*CETAM - PUCP*



# Sesión 3:

- Git y Github
- Python: Estructuras y POO
- Django Framework



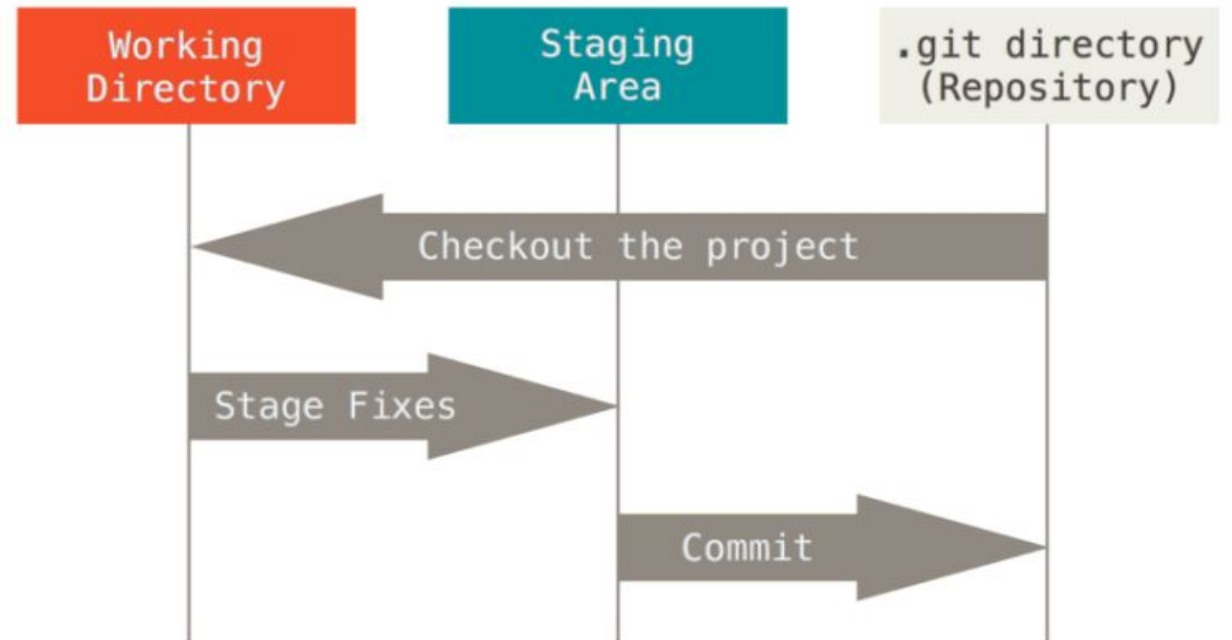
# Sistema de control de versiones GIT

*Desarrollo Web con Python - Sesión 3*

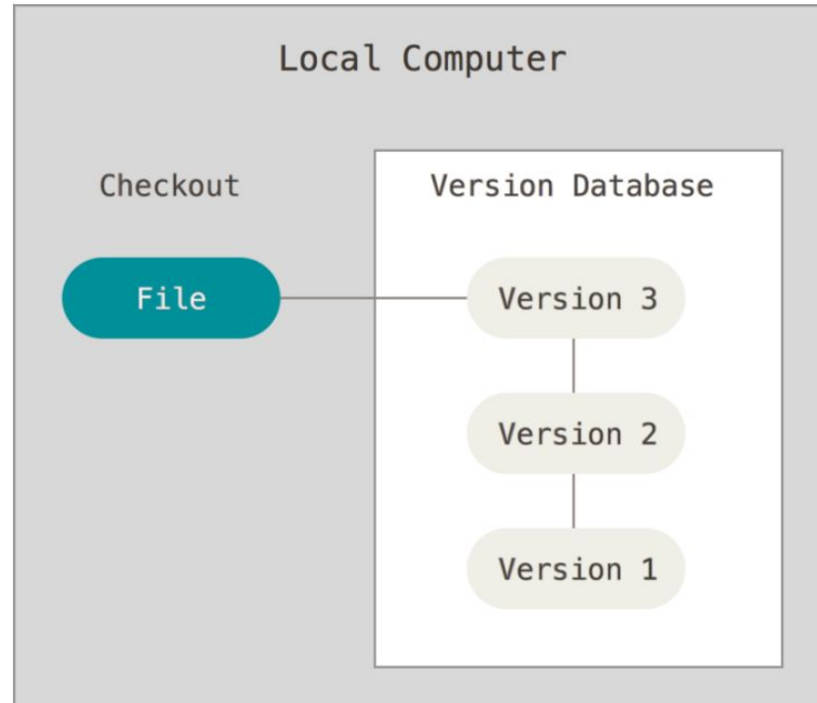


# ¿Qué es GIT?

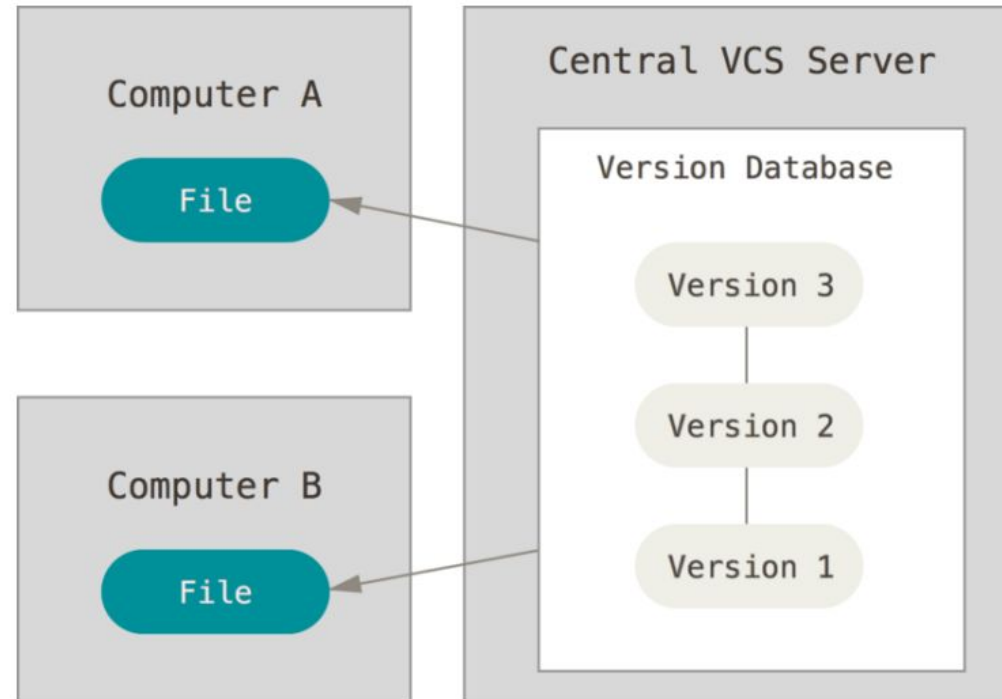
Es un sistema de control de versiones que permite mantener el historial de cambios entre los archivos de un determinado proyecto. Esta característica permite que los cambios realizados puedan ser revertidos en caso se detecten errores.



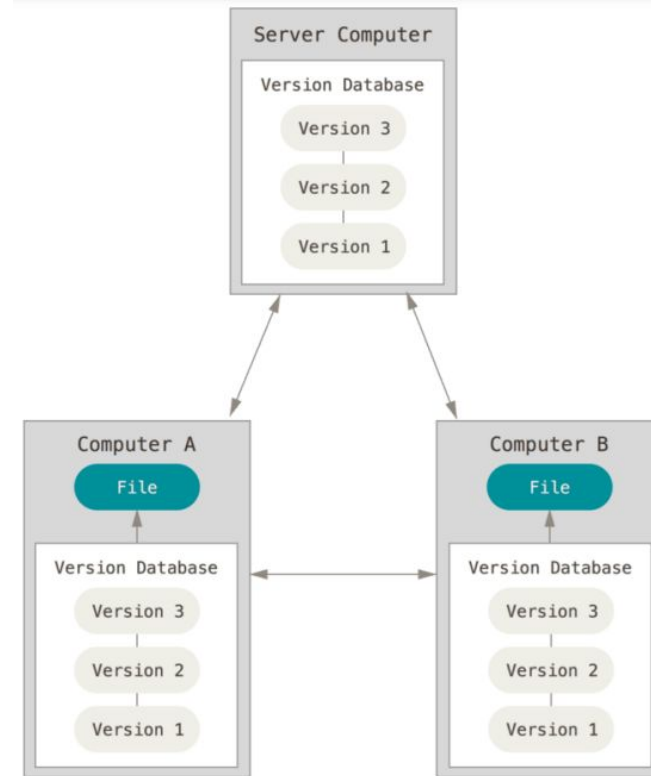
# Sistema local de control de versiones



# Sistema centralizado de control de versiones



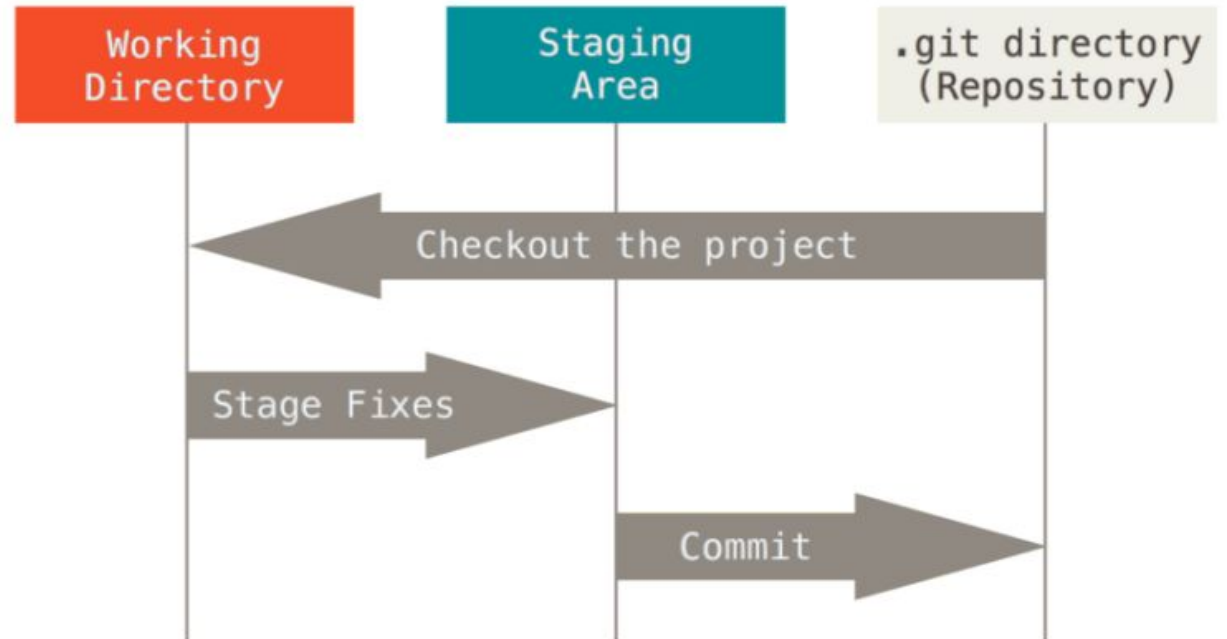
# Sistema distribuido de control de versiones





# Estados de un repositorio GIT

- Archivos no trackeados.- Son archivos de los cuales no se lleva un registro de sus versiones y son ignorados por el sistema
- Archivos agregados.- Son archivos colocados en el área de carga, poseen modificaciones pero aún no se encuentran trackeados
- Archivos trackeados.- Son archivos del cual se lleva el registro de sus versiones; todos los archivos agregados pasan a este estado luego de un "commit".



# Características de GIT

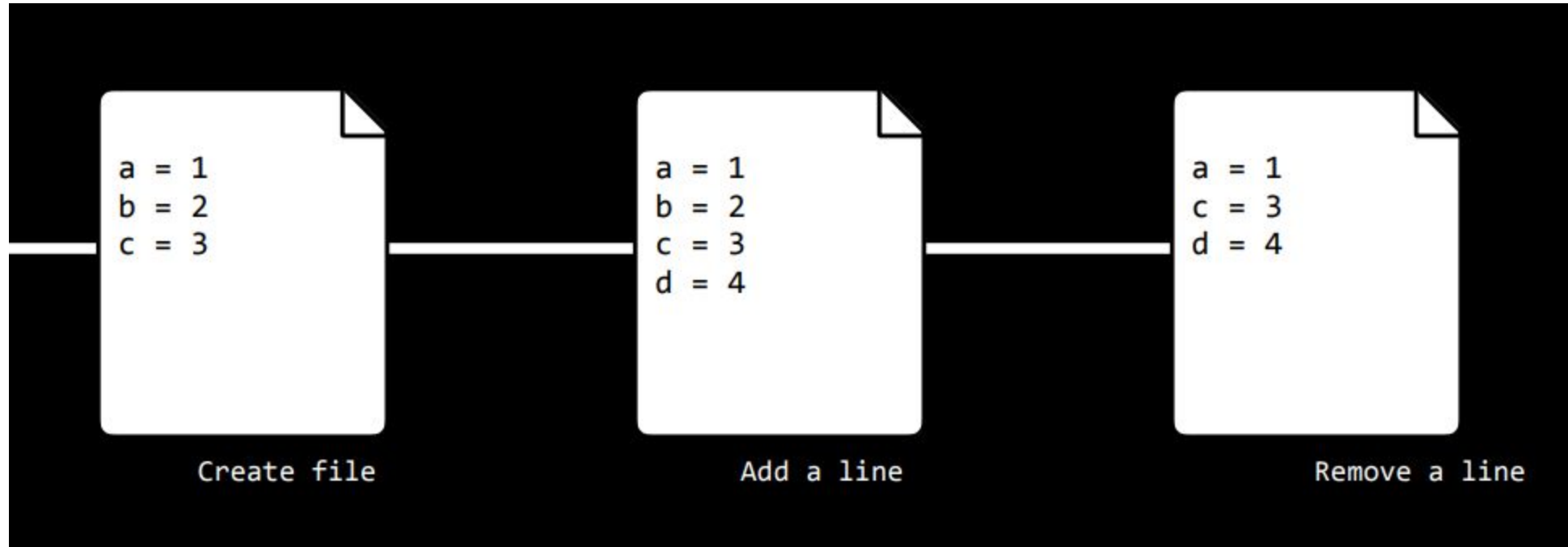
Dentro de las características de git tenemos lo siguiente:

- Verificar los cambios entre diferentes versiones
- Sincronizar los cambios entre diferentes desarrolladores
- Crear ramas de experimentación del proyecto
- Regresar a versiones anteriores del proyecto

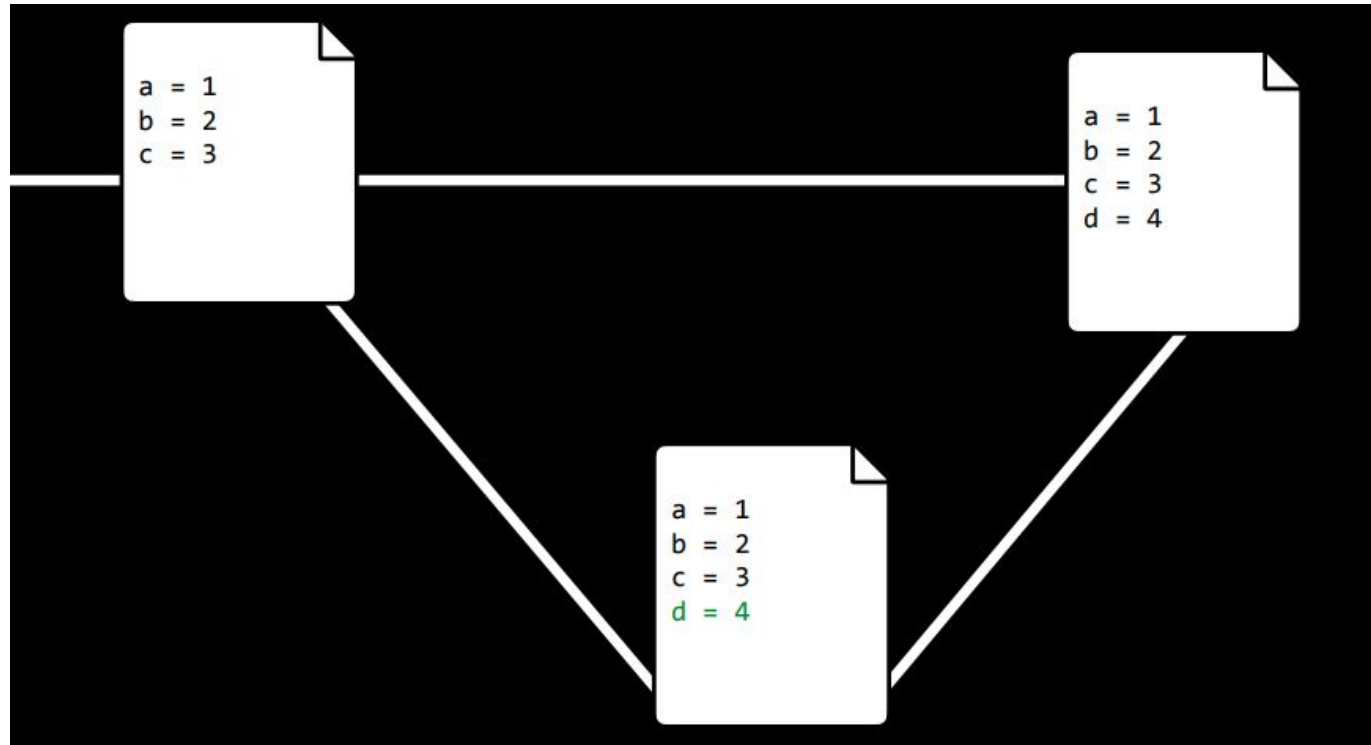


Documentacion Git: <https://git-scm.com/docs>

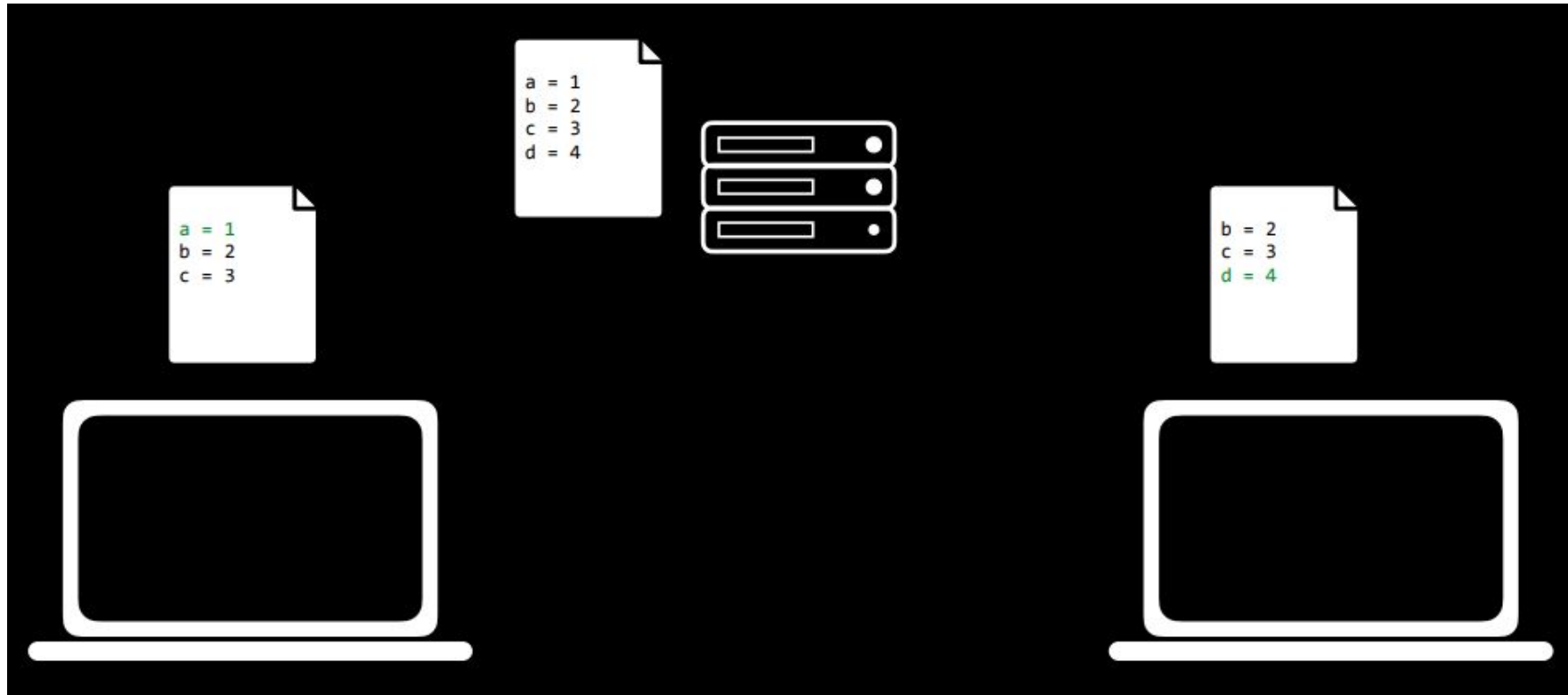
# Verificar los cambios de código



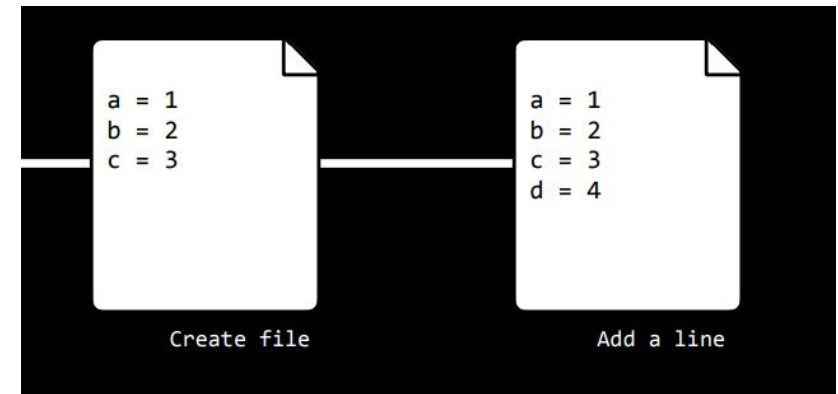
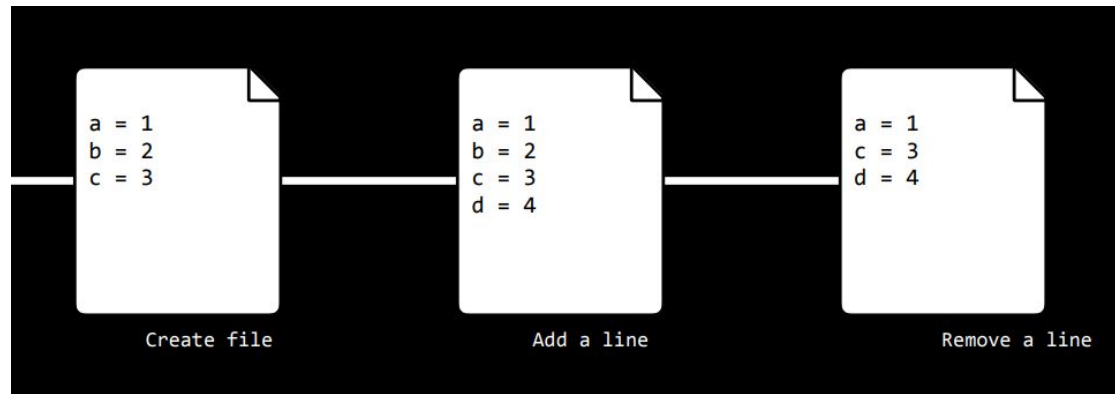
# Verificar cambios en el código sin perder la versión original



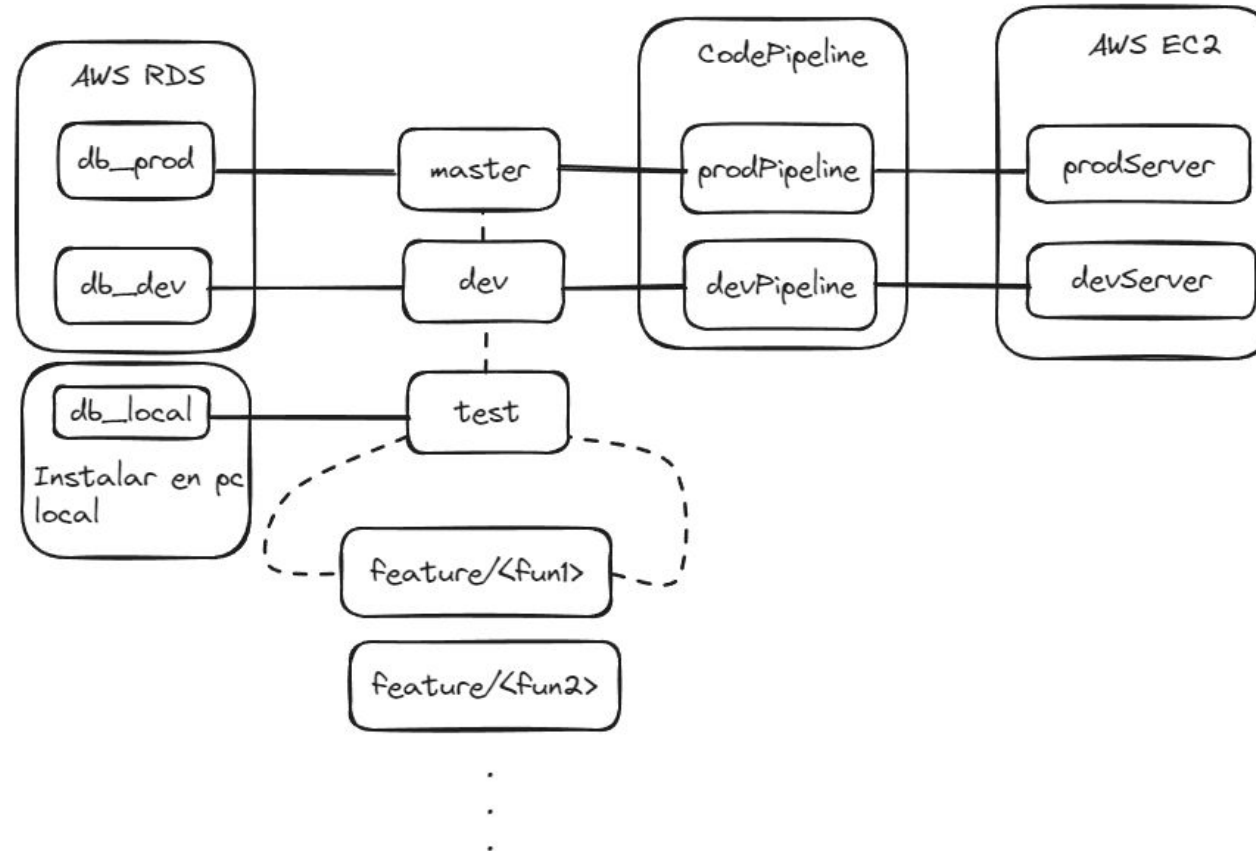
# Sincronizar cambios entre diferentes desarrolladores



# Regresar a versiones anteriores del código

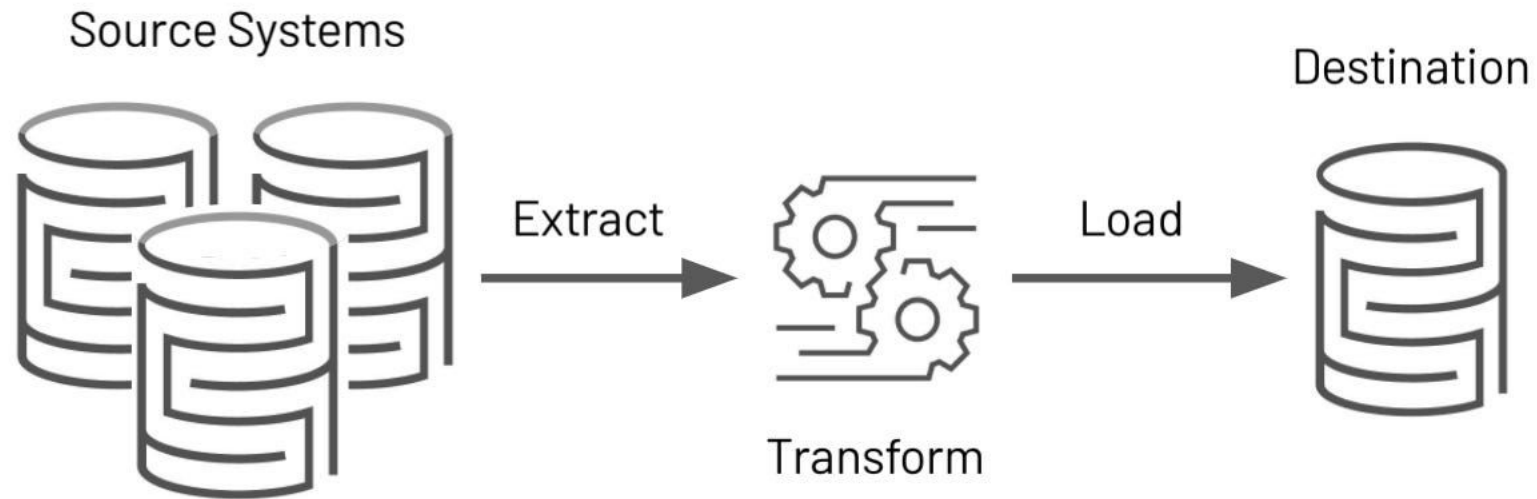


# Ejemplos de aplicación: Proyecto Web Metalprotec



# Ejemplos de aplicación: Procesos ETL

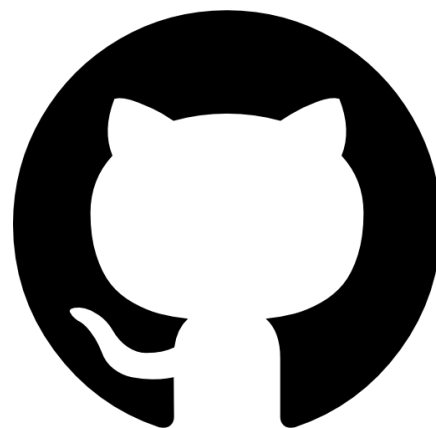
## ETL Process





# Ejemplos de aplicación: Proyecto Teleatiendo





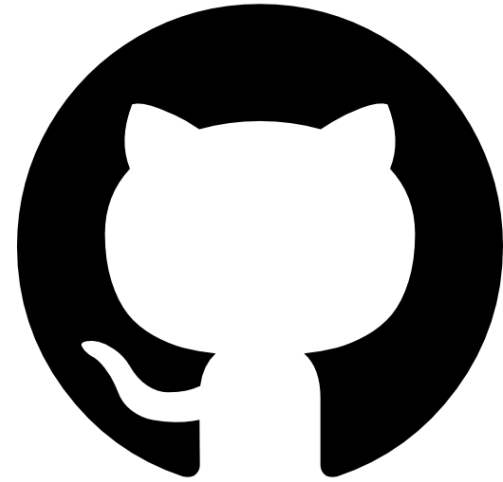
**GitHub**

*Desarrollo Web con Python - Sesión 3*



## ¿Qué es Github?

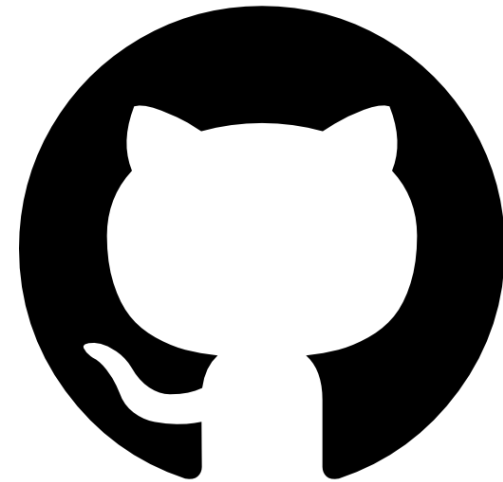
Github es un servidor de almacenamiento para repositorios de git, en donde se visualizan los diferentes proyectos realizados. Además posee un herramientas gráficas para revisar las diferentes versiones de un repositorio. Se pueden encontrar repositorios de diferentes librerías y proyectos desarrollados por programadores a lo largo del mundo



# Repositorios en Github

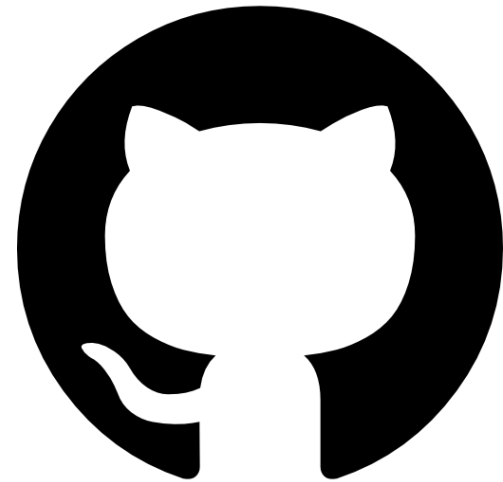
Verificando algunos repositorios notables:

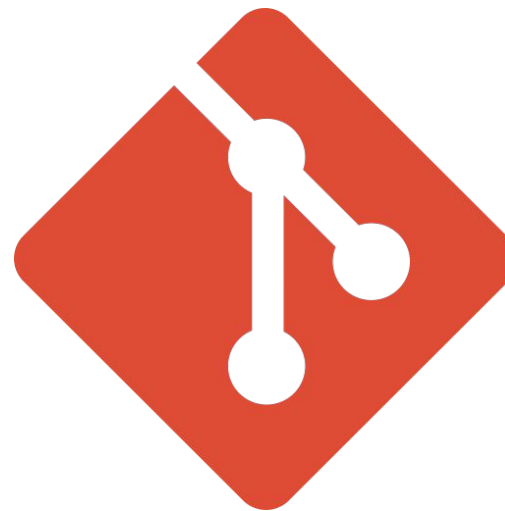
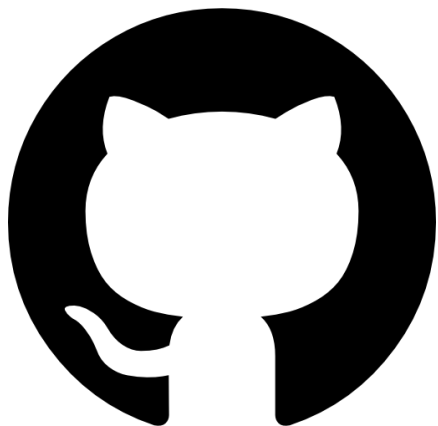
- <http://rafalab.github.io/>
- <https://github.com/brianyu28>



# Git push y Git clone

Los comandos Git Push y Git Clone permiten extraer y subir los cambios realizados a un repositorio de github. Cada repositorio posee en su configuración un origen remoto al cual se hace la petición al momento de clonarlo o actualizarlo. La autenticación de los repositorios se da a través de un token generado una sola vez por la aplicación





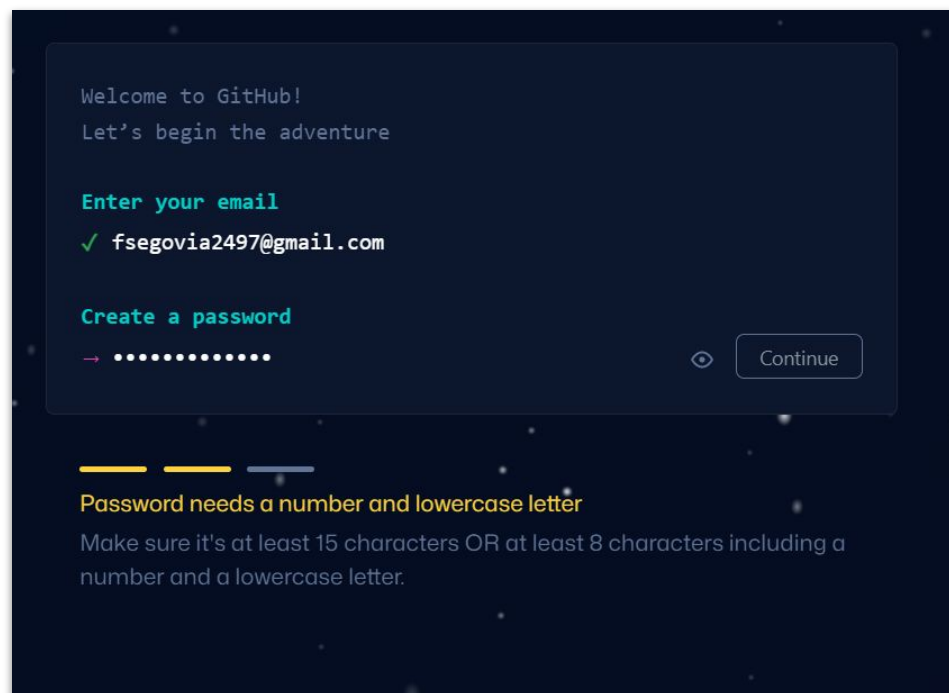
# Aplicaciones Git y Github

*Desarrollo Web con Python - Sesión 3*



# Crear una cuenta en github y generar el token

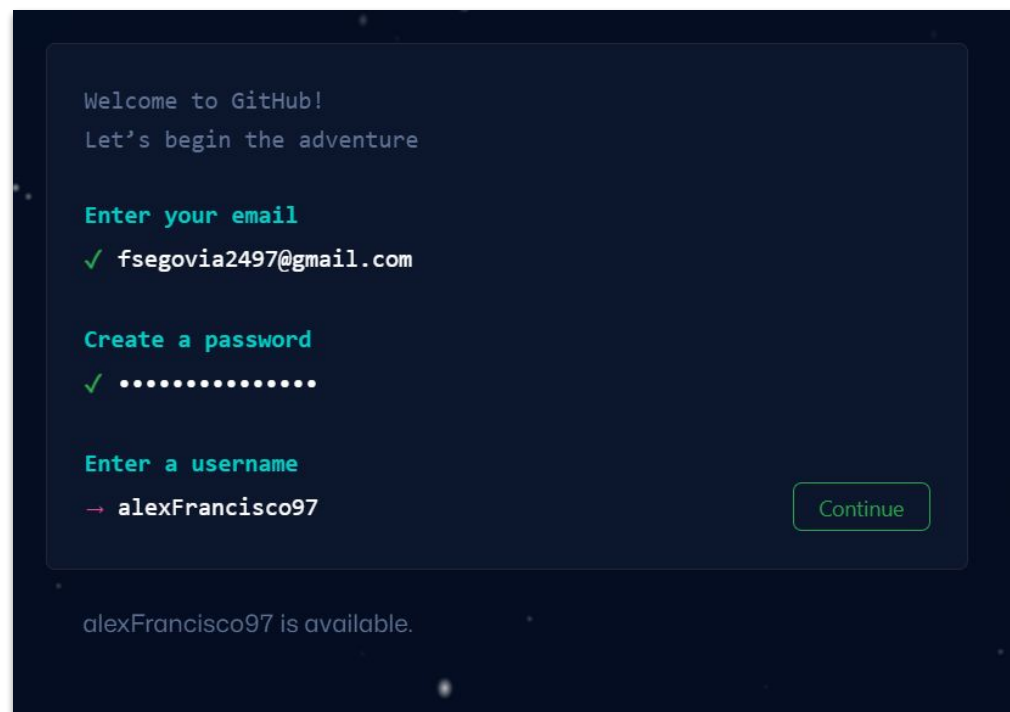
Ingresar nuestro Email y crear un password



The screenshot shows the GitHub account creation page. It has a dark blue background with a starry pattern. The text 'Welcome to GitHub! Let's begin the adventure' is at the top. Below it, 'Enter your email' is followed by a green checkmark and the email 'fsegovia2497@gmail.com'. Then, 'Create a password' is followed by a red arrow, a series of dots, an eye icon, and a 'Continue' button. At the bottom, there are three progress bars (two yellow, one blue) and a warning: 'Password needs a number and lowercase letter. Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter.'

# Crear una cuenta en github y generar el token

Ingresa nuestro Email, genera una contraseña y un nombre de usuario



The screenshot shows the GitHub account creation process on a dark-themed interface. It includes a welcome message, followed by three input fields: 'Enter your email' with the value 'fsegovia2497@gmail.com', 'Create a password' with masked characters, and 'Enter a username' with the value 'alexFrancisco97'. A 'Continue' button is visible next to the username field. At the bottom, a message states 'alexFrancisco97 is available.'

Welcome to GitHub!  
Let's begin the adventure

Enter your email  
✓ fsegovia2497@gmail.com

Create a password  
✓ .....

Enter a username  
→ alexFrancisco97

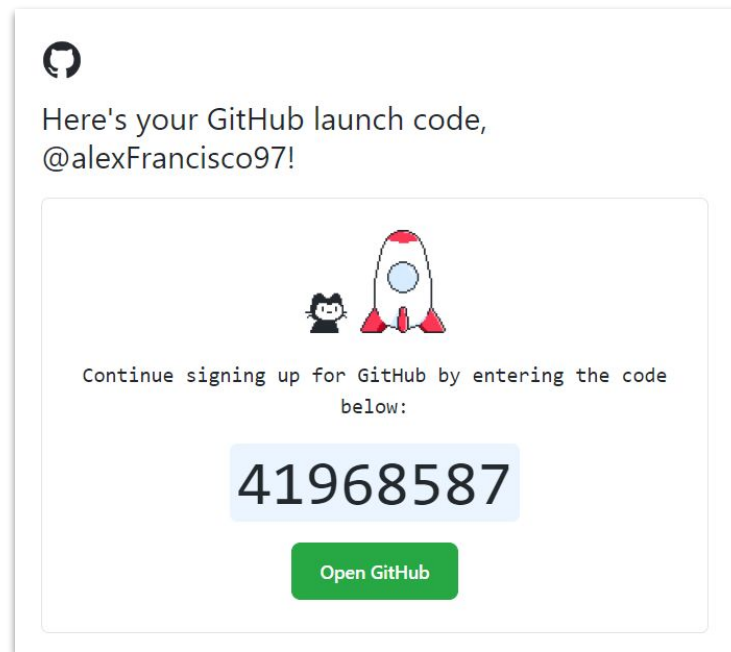
Continue

alexFrancisco97 is available.



# Crear una cuenta en github y generar el token

Confirmar el correo electrónico



# Crear una cuenta en github y generar el token

En configuración ingresar a 'Opciones de desarrollador' y 'Generar nuevo token'

- Applications
- Scheduled reminders
- Archives
- Security log
- Sponsorship log
- Developer settings

All of the fields on this page are optional and can be out, you're giving us consent to share this data where our [privacy statement](#) to learn more about how we use it.

[Update profile](#)

## Contributions & Activity

- ☐ **Make profile private and hide activity**  
Enabling this will hide your contributions and activity from leaderboards and releases.
- ☐ **Include private contributions on my profile**  
Your contribution graph, achievements, and activity will be visible on your profile.

Settings / Developer settings

- GitHub Apps
- OAuth Apps
- Personal access tokens**

## Personal access tokens

[Generate new token](#) [Revoke all](#)

Tokens you have generated that can be used to access the [GitHub API](#).

[commits\\_agosto](#) — `admin:pgp_key, delete_packages, delete_repo, project, repo, write_packages` Last used within the last week [Delete](#)  
Expires on **Tue, Sep 20 2022**.

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

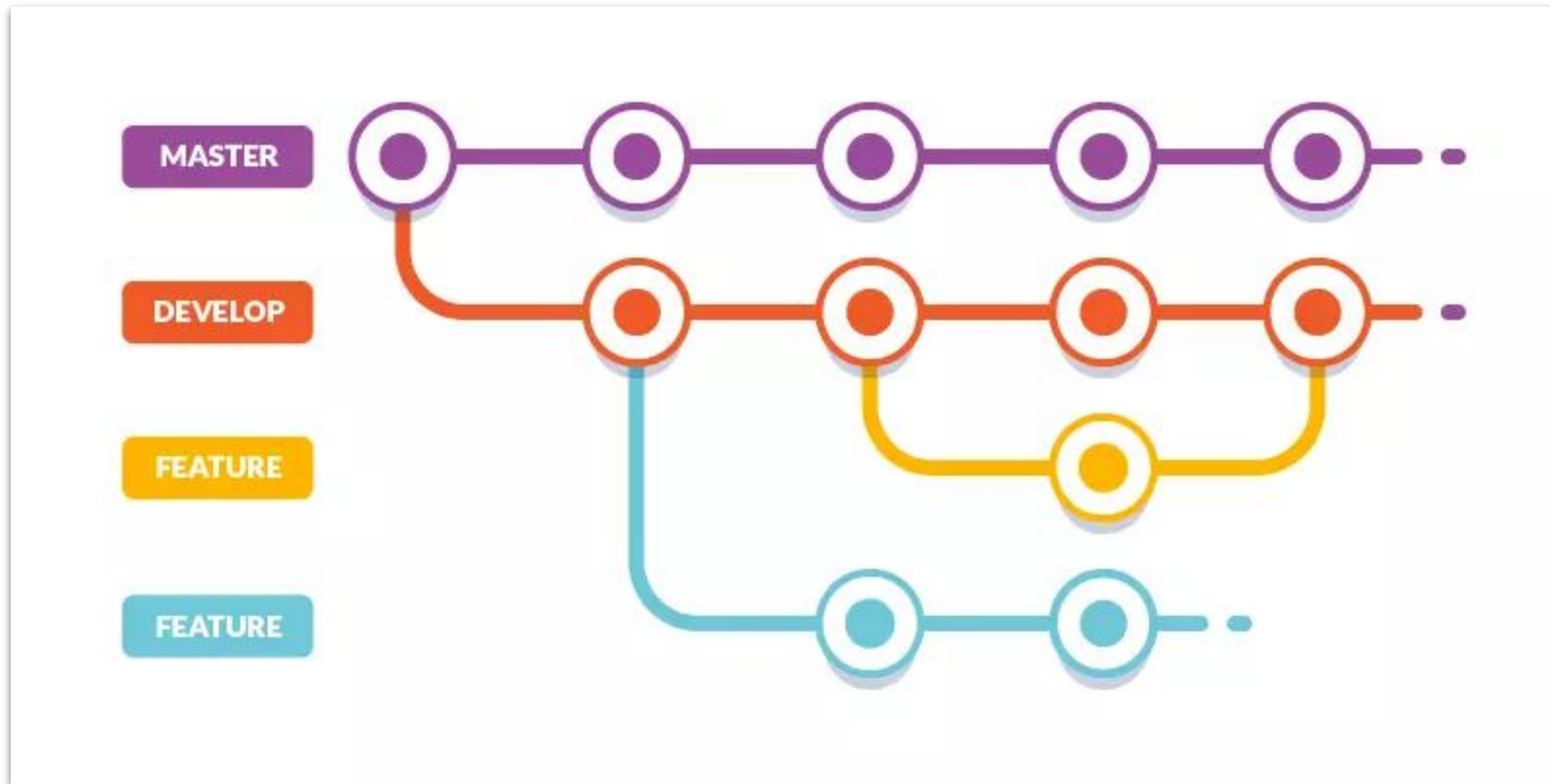
# Crear y trackear un repositorio

git init	#Inicializa un repositorio
git log	#Muestra el historial de versiones del repo
git status	#Muestra el estado de los archivos en el repo
git add	#Agrega archivos al área de pre-carga
git commit	#Crea una nueva versión del repositorio
git clone	#Crea una copia local de un repositorio remoto

# Comandos más comunes en Git

<code>git log</code>	<code>#Muestra el historial de versiones del repo</code>
<code>git status</code>	<code>#Muestra el estado de los archivos en el repo</code>
<code>git clone</code>	<code>#Crea una copia local de un repositorio remoto</code>
<code>git branch</code>	<code>#Permite crear diferentes ramas en el repo</code>
<code>git checkout</code>	<code>#Permite visualizar las diferentes versiones</code>
<code>git reset</code>	<code>#Eliminar los cambios posteriores</code>
<code>git merge</code>	<code>#Permite combinar dos ramas diferentes</code>

# Ramas en un repositorio



# Subir cambios a un repositorio remoto

`git remote`

`#Visualiza las fuentes remotas`

`git remote add <url>`

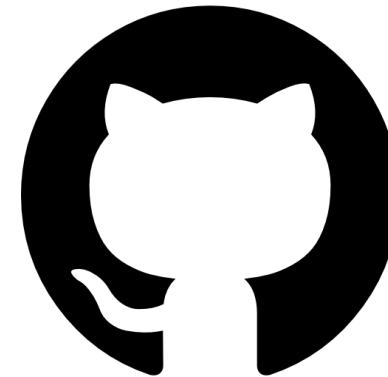
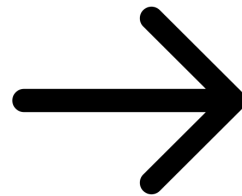
`#Permite agregar una fuente remota al repo`

`git push origin master`

`#Empuja los cambios locales al repo remoto`



**git**



# Gracias