Introducción conceptual a

UNLaM

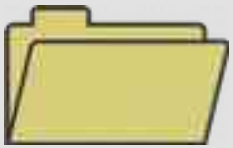
LACIE
Laboratorio
Abierto
de la **C**arrera de
Ingeniería
Electrónica

DIIT
Departamento de Ingeniería e Investigaciones Tecnológicas

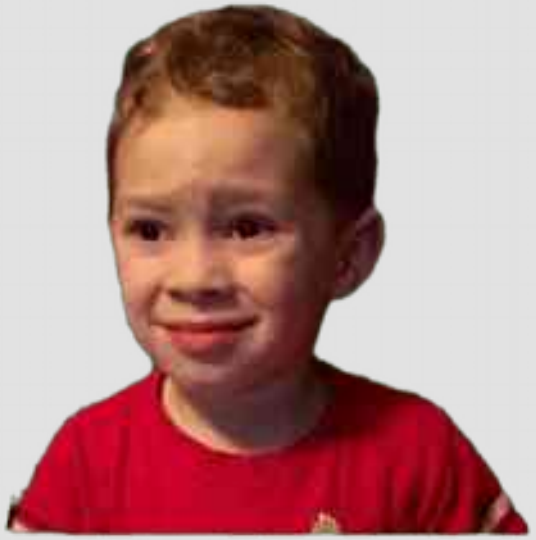
Autor: Diego Brengi
Colaboración: Federico Vallés
Fecha: 19/09/20

El Problema

- Cuando se realiza un diseño de software o hardware, el diseñador va realizando cambios (mejoras) a los archivos.
- Para agregar cambios, pero sin perder por error el trabajo previo realizado, es conveniente guardar cada vez que se hacen cambios significativos que implican varias horas de trabajo.



Mi_Circuito_V1.sch
Mi_Circuito_V2.sch
Mi_Circuito_V2b.sch
Mi_Circuito_V3a.sch
Mi_Circuito_funciona.sch
Mi_Circuito_funciona_ultimo.sch
Mi_circuito_funciona_ultimo_final.sch
Mi_circuito_funciona_ultimo_posta.sch
Mi_circuito_definitivo.sch
Mi_circuito_final_final_posta.sch



Las leyes de Murphy

UNLaM

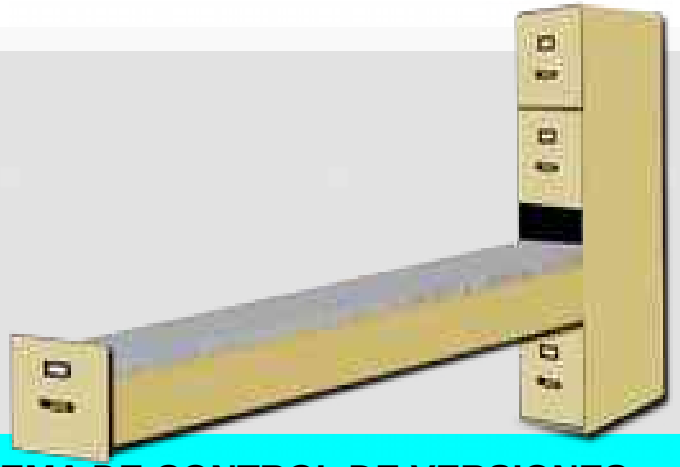
Tarde o temprano:
- Borramos el archivo incorrecto.
- Comenzamos a trabajar desde una versión anterior a la última.
- Nos olvidamos cuál es la última o que significa cada archivo.

Se hace más complejo cuando:
- Se manejan varios archivos por proyecto.
- Participan varias personas.
- Los participantes trabajan en distintas partes del mundo.

Sistema de Control de versiones

UNLaM

- Un sistema de control de versiones registra el archivo cada vez que se lo indicamos.
- Cada archivo se registra con fecha, hora y autor.
- En nuestra carpeta solo veremos la última versión.



Mi_Circuito.sch

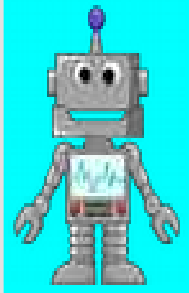
Regístrese y archívese

SISTEMA DE CONTROL DE VERSIONES

Mi_Circuito.sch : v1 Fecha y hora – Autor
Mi_Circuito.sch : v2 Fecha y hora – Autor
Mi_Circuito.sch : v3 Fecha y hora – Autor
Mi_Circuito.sch : v4 Fecha y hora – Autor
Mi_Circuito.sch : v5 Fecha y hora – Autor

Permite:

- Volver a versiones anteriores.
- Saber cuándo y quiénes han aplicado cambios.
- Comparar versiones.
- Conocer el “historial” de un archivo.
- Guardar en un lugar seguro.



El software se encarga de registrar y archivar las versiones.

Tipos de Archivo

Los sistemas de versionado trabajan mejor con archivos de texto plano como por ejemplo:

- Código C/C++, Python, scripts, etc.
- Archivos latex.
- Archivos de configuración.
- Listados.

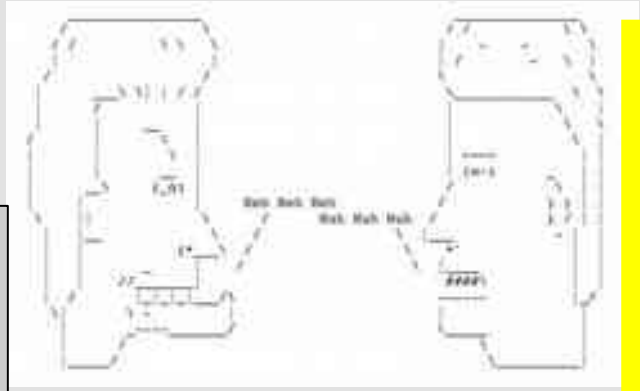
Son menos eficientes con archivos binarios como:

- Fotos e imágenes.
- Pdfs.
- Documentos word,excel, presentaciones, libreoffice, etc.
- Ejecutables.
- zip y rar.
- etc., etc.

Porque guardan solamente las diferencias entre un archivo y el siguiente.

```
void main()
//rutina main
{
int cont;
}
```

```
void main(void)
{
int cont;
return 0;
}
```



Cuidado: por cada nueva versión de uno de estos archivos se guardan las copias completas de las versiones anteriores (aunque los cambios sean pequeños). Eso puede hacer crecer mucho el tamaño del repositorio si no somos cuidadosos.

Sistemas de control de versiones

UNLaM

Existen muchos sistemas de control de versiones.
Sin embargo en la última década *git* se convirtió en el más popular.

El sistema de control de versiones *git*

“One git to rule them all”

¿Qué es Git?

UNLaM

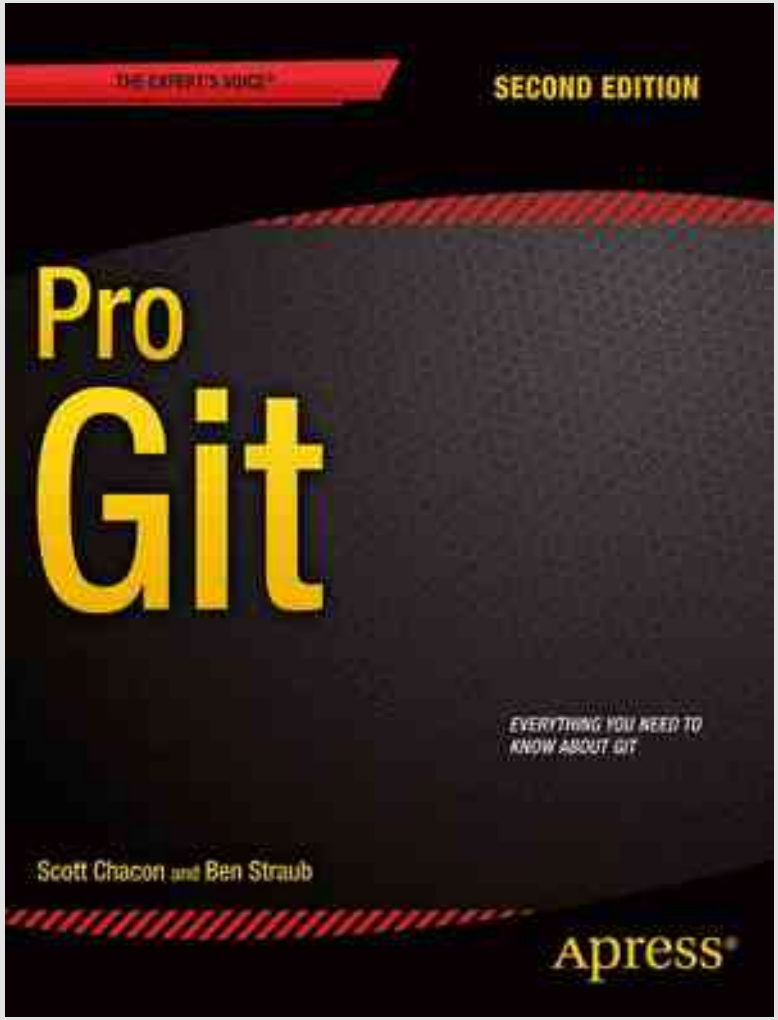
GIT es un sistema distribuido de control de versiones

URL oficial:
http://git-scm.com/

Libro

El libro de referencia sobre GIT es de descarga gratuita:

Libro oficial en inglés y español:
https://git-scm.com/book/es/v2



Origen

Linus Torvalds

Creador de

LINUX

- 61.725 archivos
- 25.584.633 líneas de código.
- Más de 19.000 desarrolladores

Creador de

git

Para que todos los programadores puedan trabajar en el desarrollo de LINUX sin recurrir a herramientas pagas.

Repositorios git

UNLaM

- Son carpetas donde se guardan todos los archivos de un proyecto.
- Están en “servidores git”.

La abeja Maya usando git

Servidor git

Repositorios git

Servidores de repositorios

- Los servidores contienen los repositorios git.
- Se acceden a través de una URL.
- Están en un equipo físico en alguna parte.

Servidores en una empresa

Servidor git interno

Servidores públicos

Servidor git de GITLAB.COM

Servidor git de GITHUB.COM

Servidor git de BITBUCKET.ORG

Acceder a un repositorio

Cómo acceder a un repositorio git sin instalar nada.

Permisos de acceso

Si el servidor es **público** y el repositorio es **abierto**, solo hay que saber la URL.

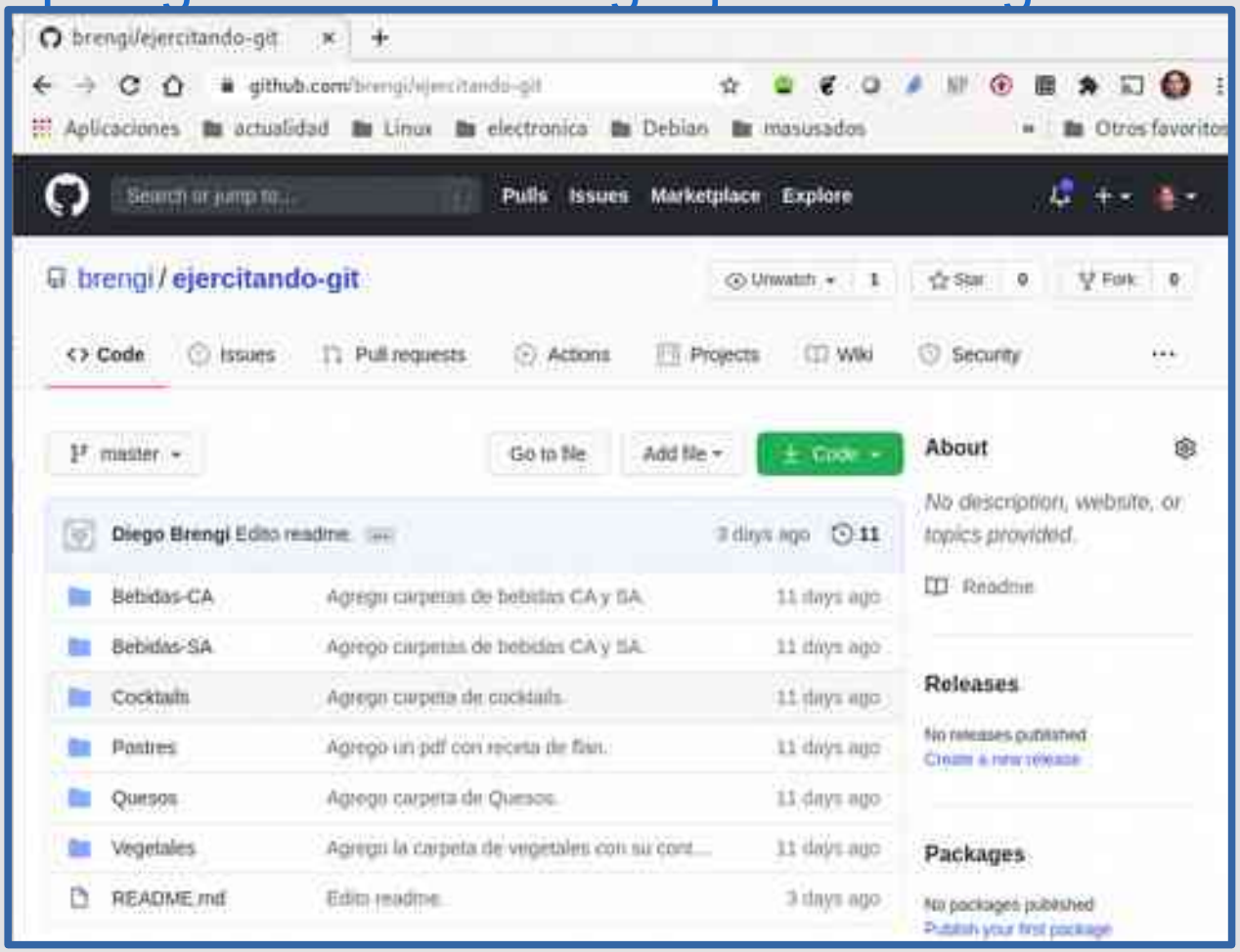
Si el repositorio es privado se necesita:

1. **Estar registrado en el servidor git con una cuenta de acceso (usuario y clave que te gestiona el admin).**
2. **Solicitar permisos al dueño del repositorio.**
3. **Url del repositorio.**

Acceso mediante página web

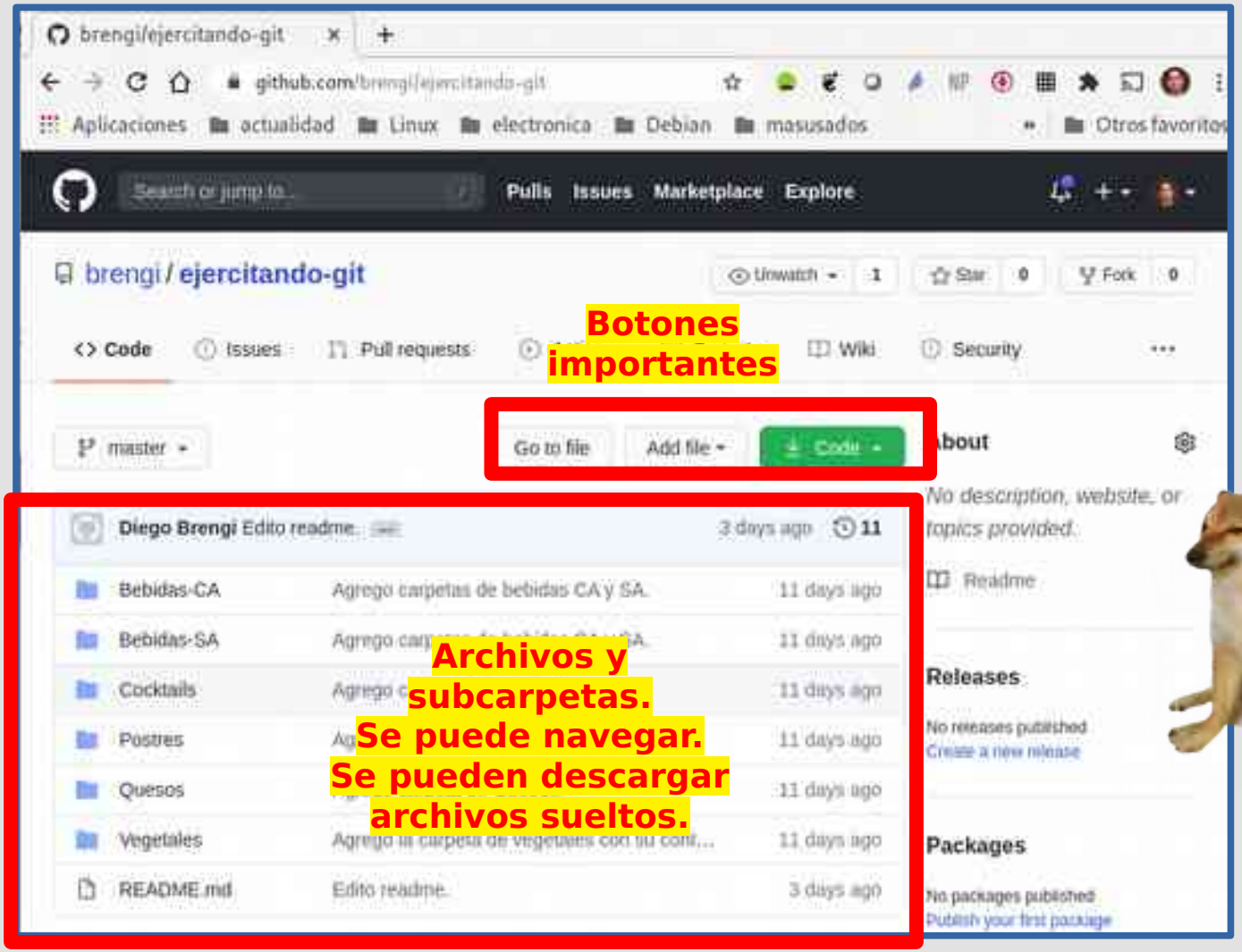
UNLaM

https://github.com/brengi/ejercitando-git

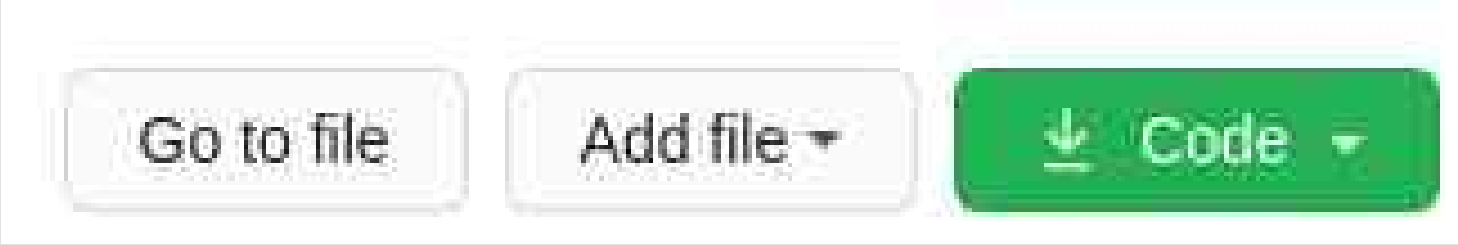


Acceso web

UNLaM



Botones importantes



Listado de archivos

Agregar un archivo al repo

a) Bajar un zip.
b) Bajar todo el repositorio a la computadora (con historial). Se necesita instalar previamente un software.

Agregar un archivo al repo

WARNING

Una vez ingresado al repositorio es muy complicado borrar el archivo. Se puede borrar a simple vista, pero siempre quedará en el historial. Para borrarlo completamente el propietario deberá correr algunos riesgos.

Usar el repo desde la web

Yo instalé git y me aprendí unos comandos básicos.

Yo mejor lo uso desde la web.

Instalar git en la PC

Instalar git para explorar su verdadera potencialidad.

Programas disponibles

UNLaM

Sobre éste funcionan todos los demás.

Desde la web

Lo vimos anteriormente. Tiene muchas limitaciones.

Clásico

https://git-scm.com/downloads

Línea de comandos y aplicaciones gráficas simples y básicas.

Debian/Ubuntu: sudo apt-get install git
Fedora: sudo dnf install -y git

GUI interfaces gráficas

https://git-scm.com/downloads/guis

Hay aprox. 40 interfaces gráficas disponibles. Algunas son libres y gratuitas y otras no.

- SamartGUI
- TortoiseGit
- GitKraken
- Git-cola
- Ungit

Interfaces gráficas para git

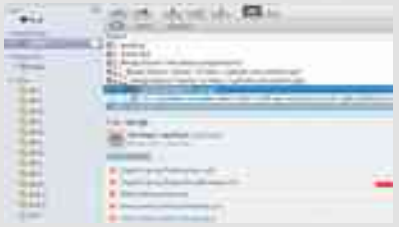
UNLaM

GUI interfaces gráficas

Si no te agrada la línea de comandos, elegí la GUI que más te guste.

https://git-scm.com/downloads/guis

- Permiten utilizar git sin usar la línea de comandos.
- Se representa visualmente la información y para muchas operaciones le facilita la vida del usuario.
- No todas despliegan la capacidad completa de git.
- Algunas veces ocultan las operaciones reales al usuario.
- Si no se tienen los conceptos primero, puede ser difícil entender qué está pasando.



Instalar git clásico

UNLaM

Git clásico

Pedir la guía de instalación para windows.

Bajo linux, instalar la versión que venga con la distro.

Para basados en Debian el comando es: sudo apt-get install git

En este cursito vamos a ver algunos comandos básicos como para empezar.

Configuración inicial

UNLaM

Luego de instalar, hay que configurar git para que cada operación que hagamos quede correctamente registrada (en especial agregar y modificar archivos):

Se hacen solo una vez. Queda guardado en la configuración global de git.

Configurar el nombre y apellido:

```
git config --global user.name "Diego Capusotto"
```

Realmente podemos poner cualquier valor tanto en el nombre como en el email. Son los datos que se mostrarán en el registro de cambios.

Configurar el email:

```
git config --global user.email "dcapusotto@inti.gob.ar"
```

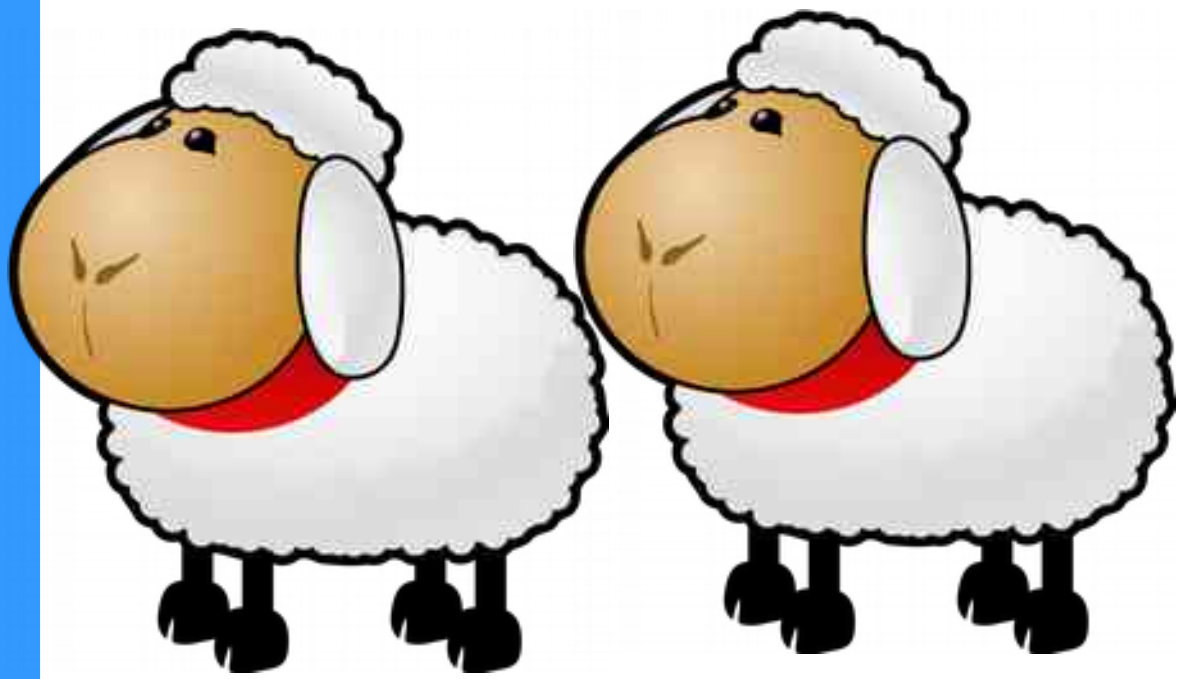
Caché de claves:

```
git config --global credential.helper  'cache --timeout=28800'
```

Guarda las claves que ingresemos por 8 horas y nos evita ingresarlas por cada operación en el servidor.

Clonar un repositorio

Copiar un repositorio completo en nuestra PC.



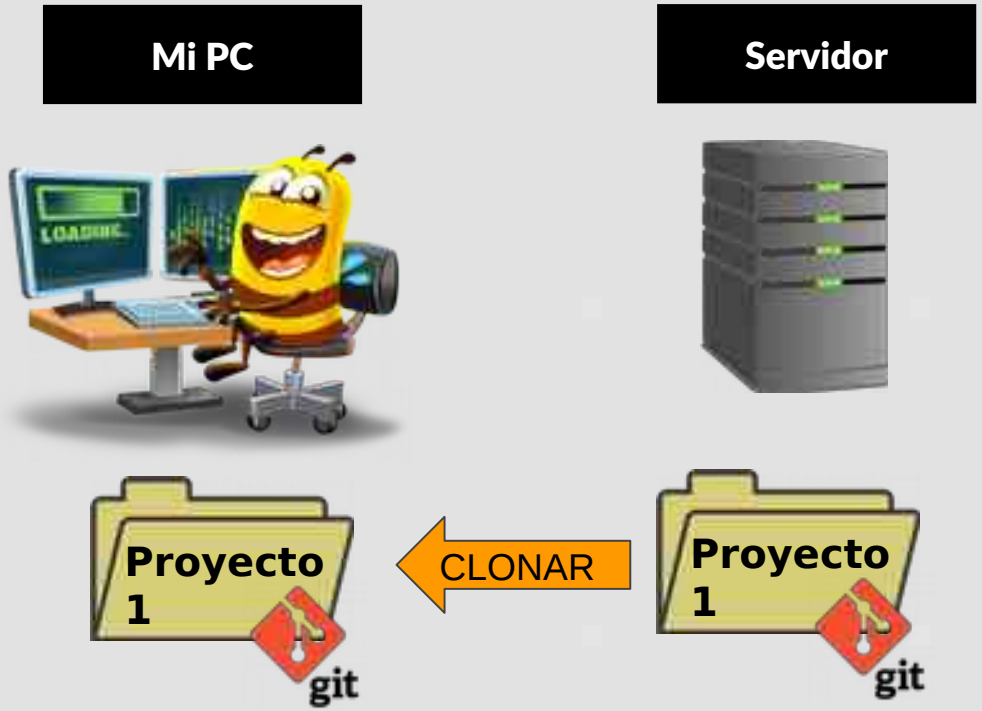
Qué es clonar

UNLaM

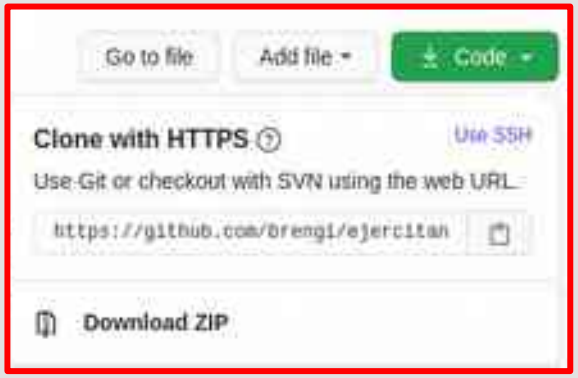
Clonar un repositorio significa hacerte una copia completa local en tu disco.

COMPLETA:
Están todos los archivos con su historial completo.

LOCAL:
Todo está ahora en una carpeta en el disco de tu PC.



Como clonar



git clone https://github.com/brengi/ejercitando-git.git

```
Clonando en 'ejercitando-git'...
remote: Enumerating objects: 41, done.
remote: Counting objects: 100% (41/41), done.
remote: Compressing objects: 100% (29/29), done.
remote: Total 41 (delta 6), reused 41 (delta 6), pack-reused 0
Desempaquetando objetos: 100% (41/41), listo.
```

Mi PC

GITHUB.COM

CLONAR

Resultado de la clonación

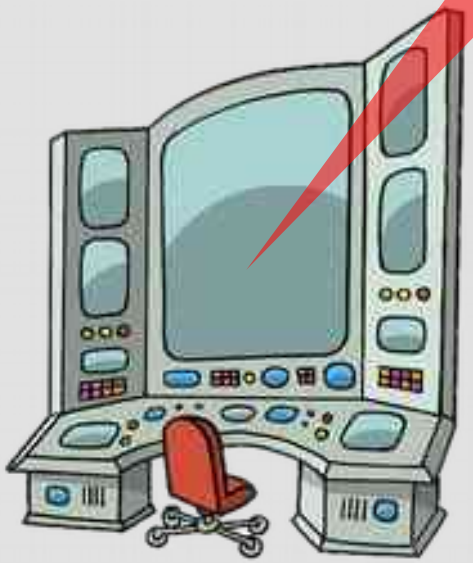
UNLaM

- Ya no necesitamos conectarnos al servidor. No hay ningún tipo de sincronización automática.
- Tenemos una carpeta con todos los archivos del proyecto. La carpeta estará en el directorio desde donde ejecutamos el comando “git clone”.
- Los archivos que vemos, son los **“archivos de trabajo”**.
- En una **subcarpeta oculta** dentro del directorio principal, tendremos todas las versiones anteriores, y mucha más información.

La carpeta .git

Nunca borres o toques dentro de la carpeta “.git”. Es el centro de control de ***git***.

git no es un servicio de sistema ni un daemon. Es un comando que va a realizar operaciones solamente cuando ejecutes el comando “git”.



Centro de control de git

```
ejercitando-git/
+-- .git
+-- Bebidas-CA
|   +-- bebidas.txt
|   \-- vodka.jpg
+-- Bebidas-SA
|   +-- agua.jpg
|   \-- bebidas.txt
+-- Cocktails
|   +-- caipi.txt
|   +-- mojito.jpg
|   +-- mojito.txt
|   \-- negroni.txt
+-- Postres
|   +-- Flan.jpg
|   +-- flan_receta.pdf
|   \-- postres.txt
+-- Quesos
|   +-- quesos.txt
|   \-- Roquefort.jpg
+-- README.md
\-- Vegetales
    +-- frutas.txt
    \-- vegetales.txt
```

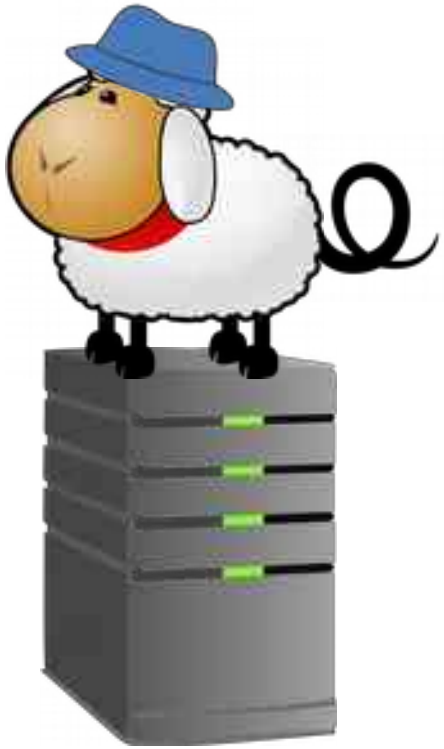
Y ahora que pasa?

UNLaM

- Ahora podés utilizar las herramientas que siempre usás en tu PC.
- Para ver o editar los textos, usá el editor de texto plano que usás siempre (por ejemplo notepad, notepad++, gedit o vim).
- Para ver o editar imágenes utiliza el programa de visualización o edición que usás siempre.
- Lo mismo aplica para cada tipo de archivo.

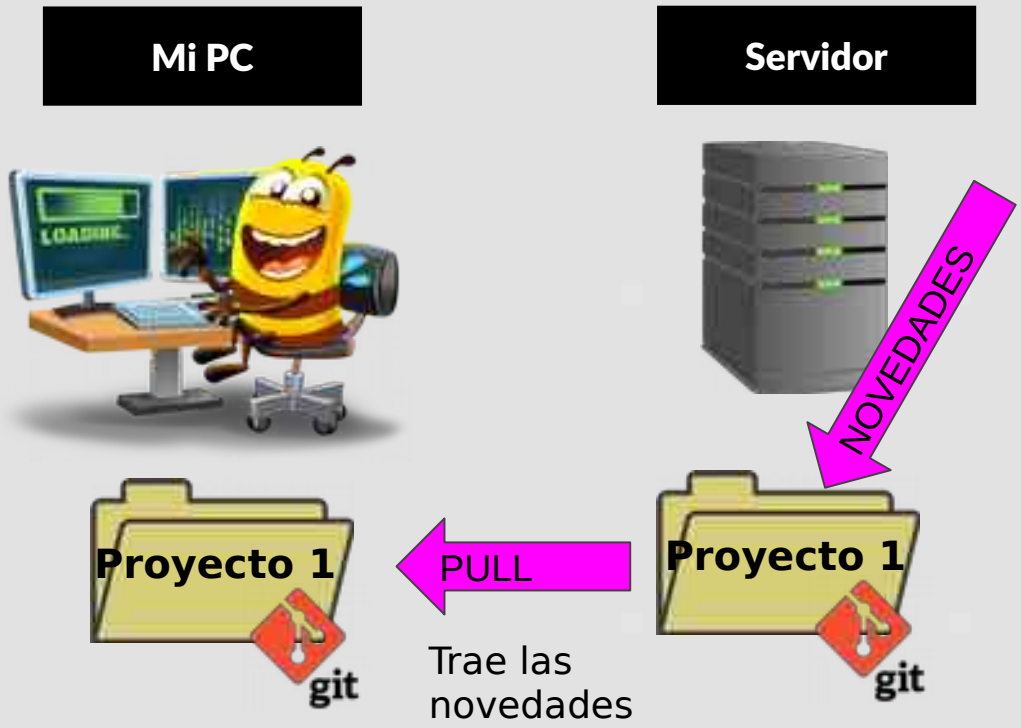
Actualizar un repositorio

Traer a la PC las novedades que están en el servidor.



Actualizar un repo

- Luego de clonar un repositorio, el resto de los desarrolladores seguirá trabajando y subirán sus cambios al servidor.
- Esos cambios en el servidor NO se sincronizan con mi repo automáticamente. Hay que realizarlo manualmente, cuando yo lo decido.
- La operación “git pull” va a traer las novedades desde el servidor y las va a reflejar en los archivos de trabajo.



El comando git pull

No hay novedades

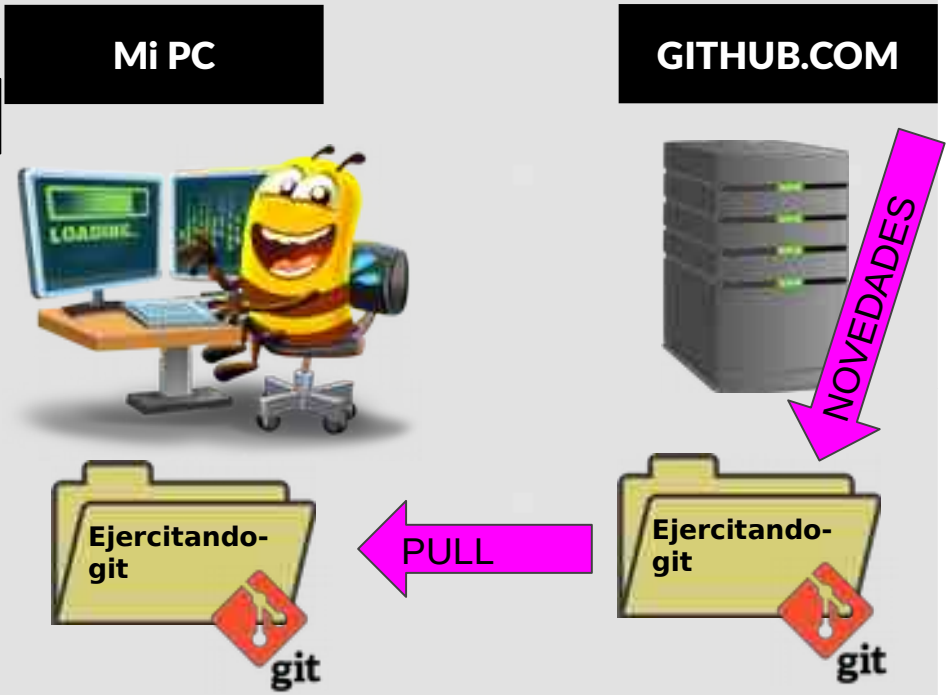
git pull

```
Ya está actualizado.
```

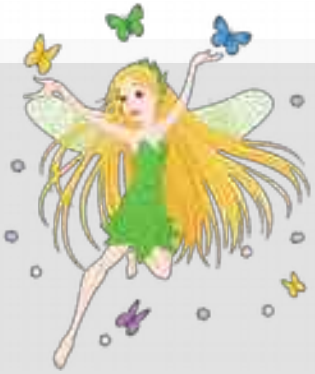
Hay novedades

git pull

```
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Desempaquetando objetos: 100% (3/3), listo.
Desde https://github.com/brengi/ejercitando-git
   f9d7442..7e2cda1  master     -> origin/master
Actualizando f9d7442..7e2cda1
Fast-forward
 README.md | 5 ++---
 1 file changed, 2 insertions(+), 3 deletions(-)
```



Lo que hace “git pull”

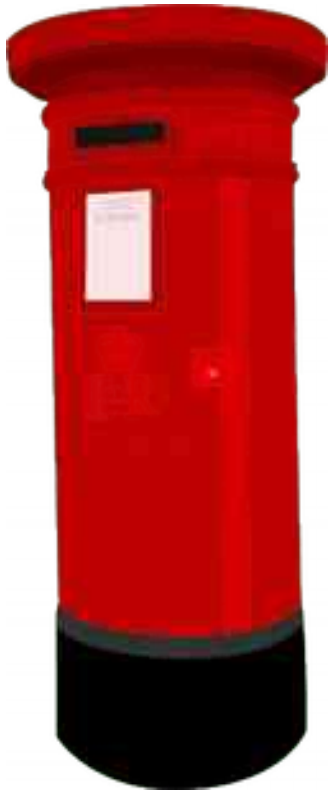


El concepto de “git pull” se explica simple para este curso introductorio. Sin embargo son varias las cosas que pasan al hacer un git pull.

1. Se consulta al servidor.
2. Se traen los cambios y se los copia dentro de la carpeta .git (fetch).
3. Se analizan las novedades y se analiza el contenido del directorio de trabajo.
4. Si no hay conflictos se actualizan los archivos de trabajo (merge).
5. Si hay conflictos se avisa al usuario y no se actualizan los archivos de trabajo.

El camino de las contribuciones

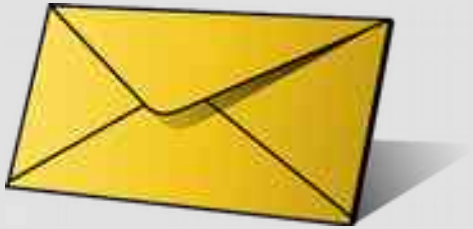
Git tiene etapas internas que debemos conocer para entender los comandos



Analogía de la carta

Cuando trabajamos escribimos hojas.

Son nuestro trabajo, nuestra producción que se refleja en los archivos de trabajo.



Las hojas “PREPARAMOS ” en sobres.

Decidimos que tenemos cosas importantes para guardar.

Depositamos los sobres en el buzón del barrio.

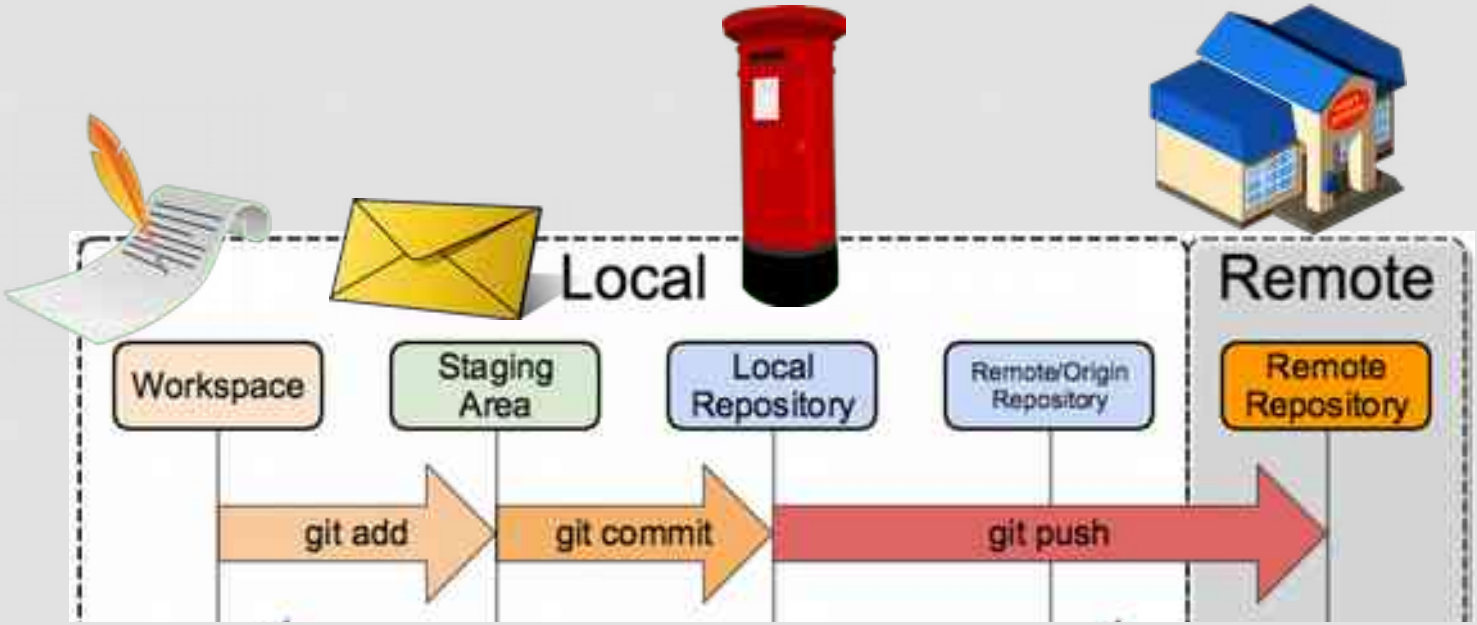
Consolidamos nuestro trabajo localmente, en un repo en nuestra PC.

Los sobres llegan al correo central.

Enviamos ese trabajo al servidor.

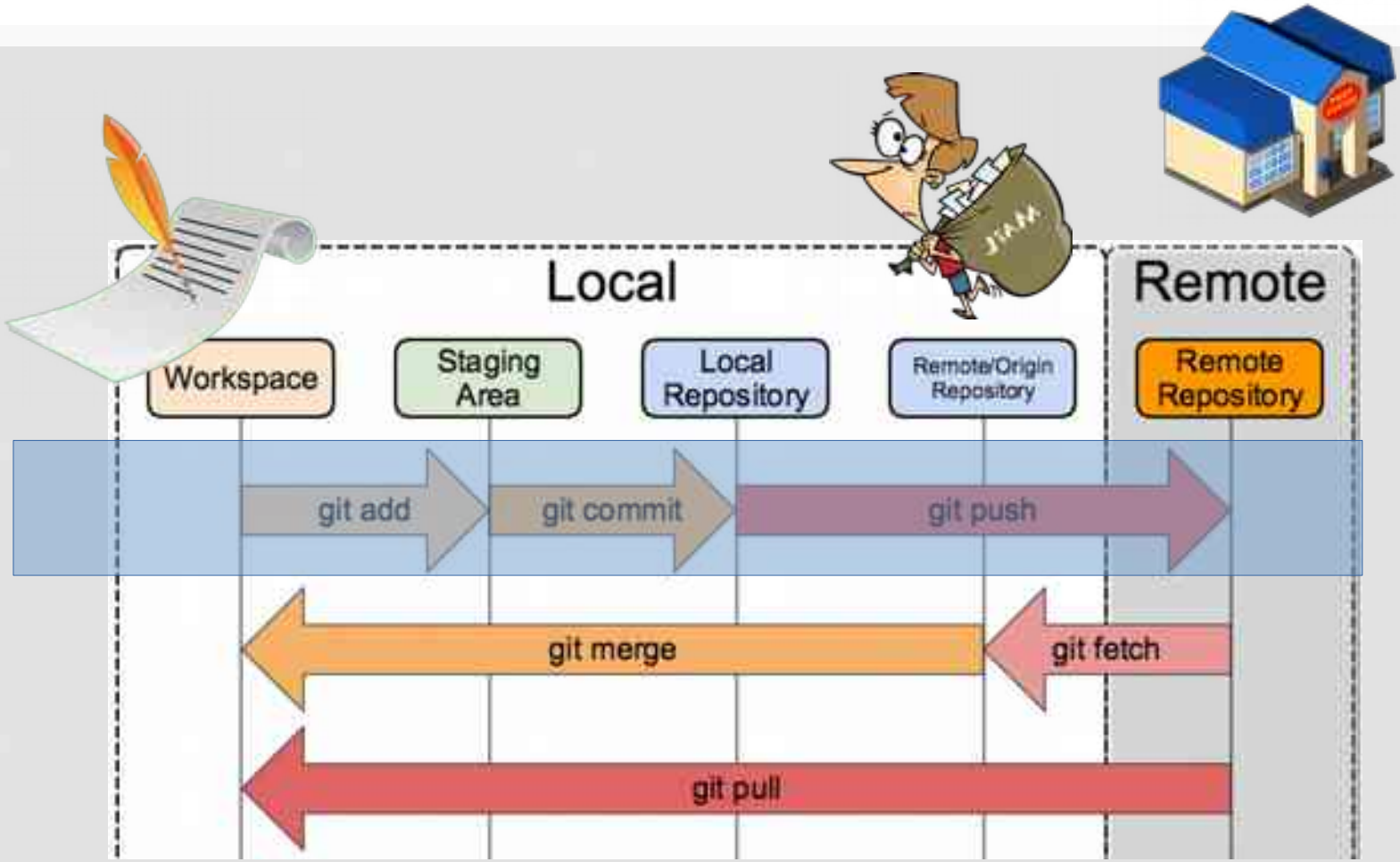
Enviando al servidor

UNLaM



Recibiendo novedades

UNLaM



Mi espacio de trabajo (workspace)

Los archivos en mi espacio de trabajo.

El workspace

UNLaM

El espacio de trabajo o workspace es donde cloné el repositorio.

Ahí voy a tener archivos de todo tipo que iré trabajando.

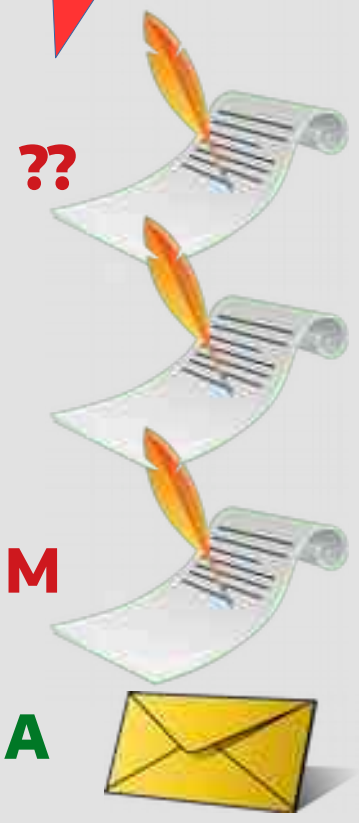
Son los archivos que veo en el disco.

Los puedo editar, modificar y borrar.

Los estados de los archivos

Avisos de git

Git va a llevar control de los archivos y les va a asignar un estado.



?? UNTRACKED

UNMODIFIED

M MODIFIED

A STAGED

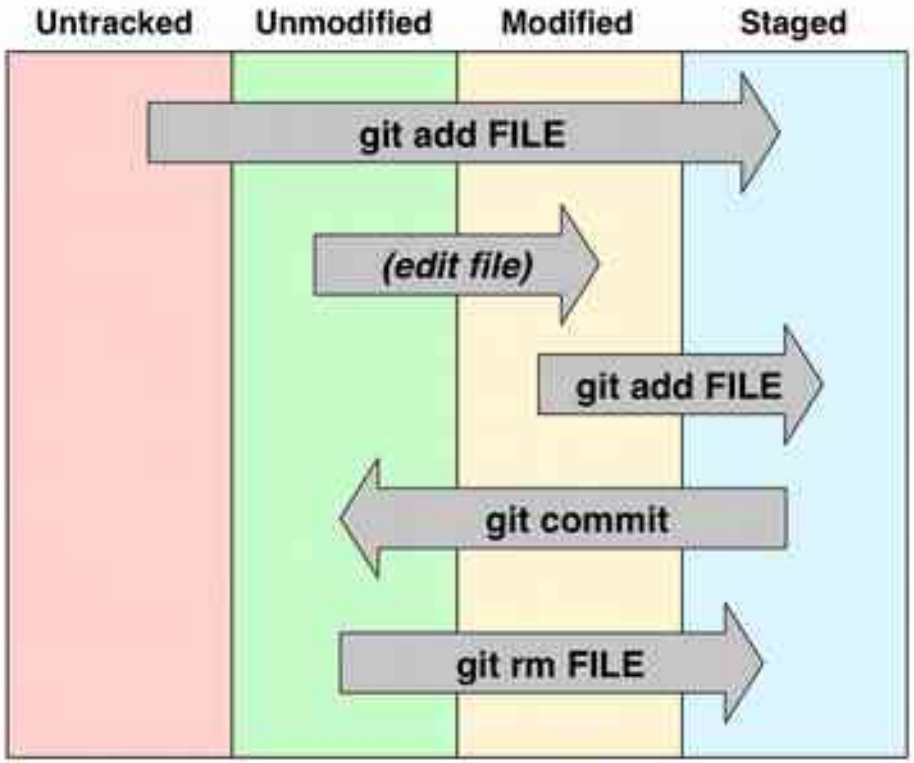
Es unarchivo nuevo o desconocido para git. Nunca se ingresó al sistema.

Es un archivo que está sin modificar, es decir que está igual que el que está en el repo.

Es un archivo que está en el repo pero que el usuario ha modificado y entonces hay diferencias.

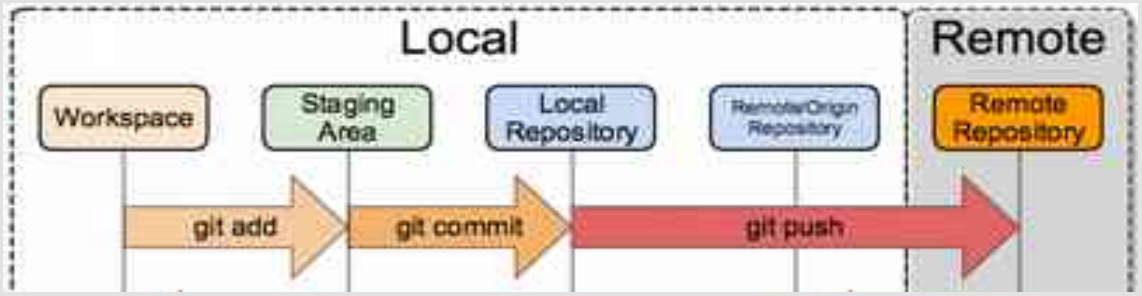
Es un archivo (nuevo o modificado) pero que ya está en el “sobre”, preparado para ingresar al repo.

Los estados



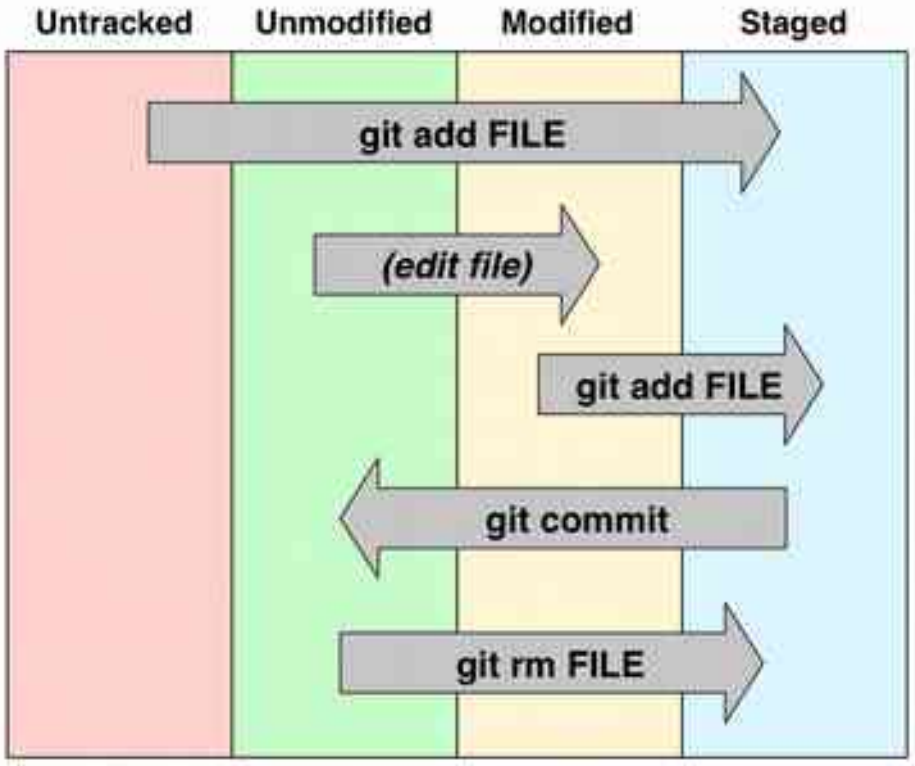
Trabajando en el repo haremos que los archivos pasen al estado de MODIFIED.

Luego, con comandos “git” los pasaremos a stage para luego consolidarlos.



Los estados

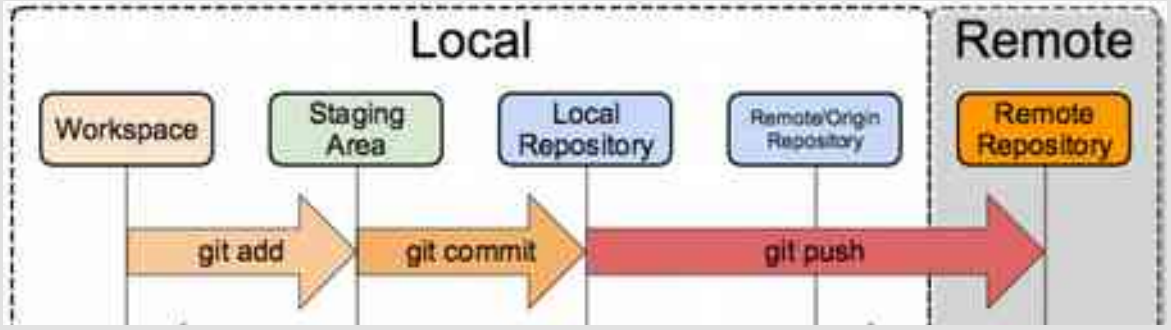
UNLaM



Si es un archivo nuevo, lo preparamos con “git add”.

Si es un archivo modificado, lo preparamos también con “git add”

Si ya está preparado, lo consolidamos con “git commit”. Luego de consolidarse quedará como UMODIFIED.



¡Muchas gracias!

LACIE

Laboratorio
Abierto
de la **C**arrera de
Ingeniería
Electrónica