Introducción a Git

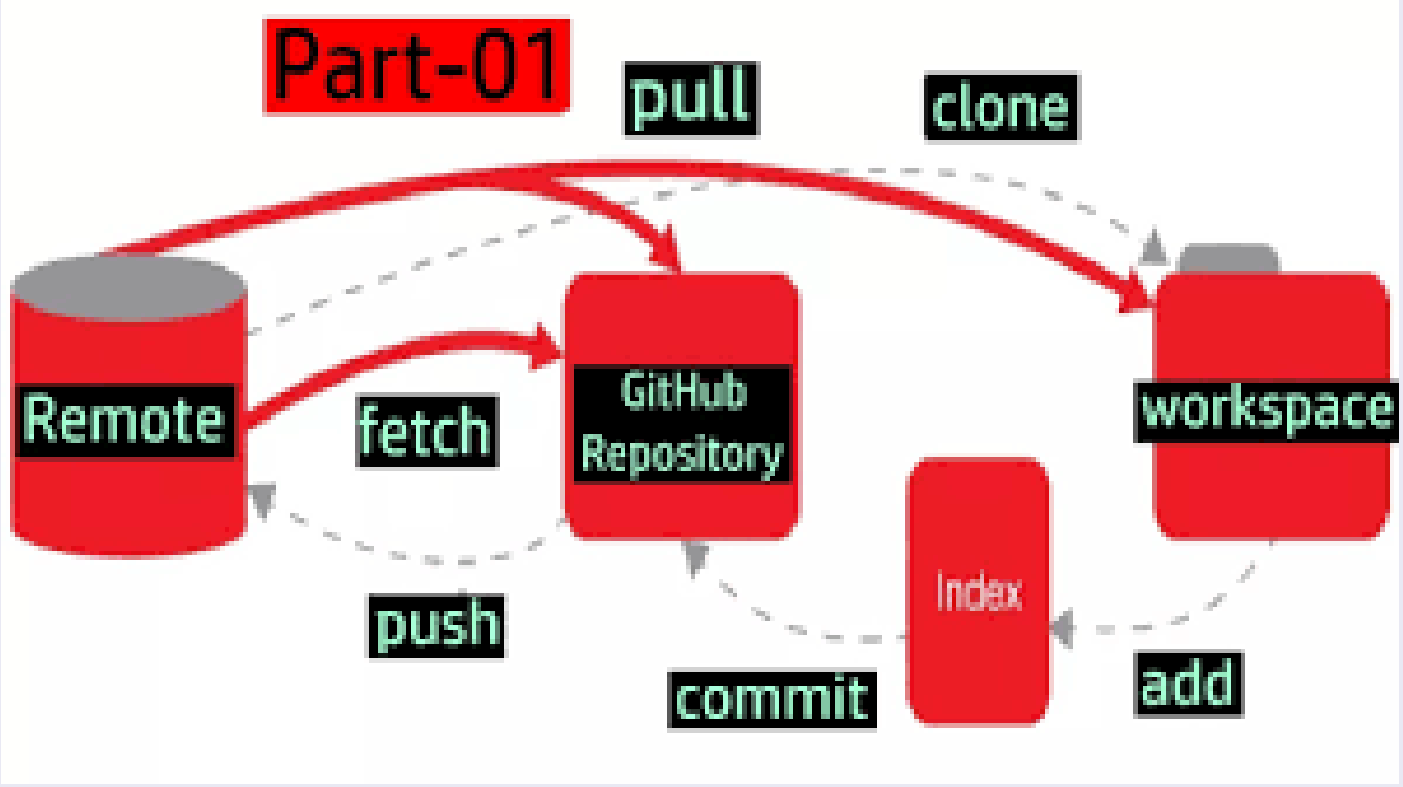
Que es git

Sistema de control de versione, administrar mis cambios.

- Control de versiones: antes, ahora, futuro
- Multiusuario
- Control de cambios por usuario
- Regresar a versiones anteriores cuando quiera
- Repositorios Locales y remotos
- Crea snapshots
- Yo decido cuando cambio

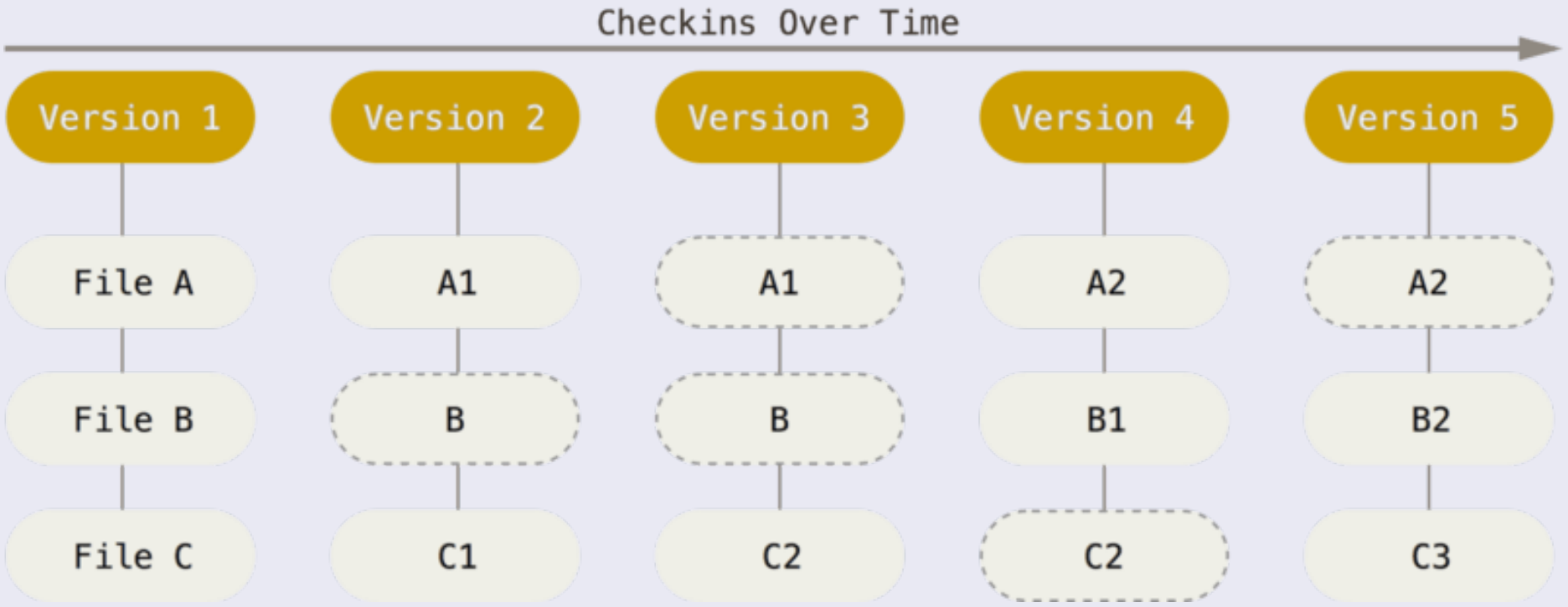
Si se ve complejo, probablemente lo es.

Por muy complejo que se vea el esquema de uso de git.



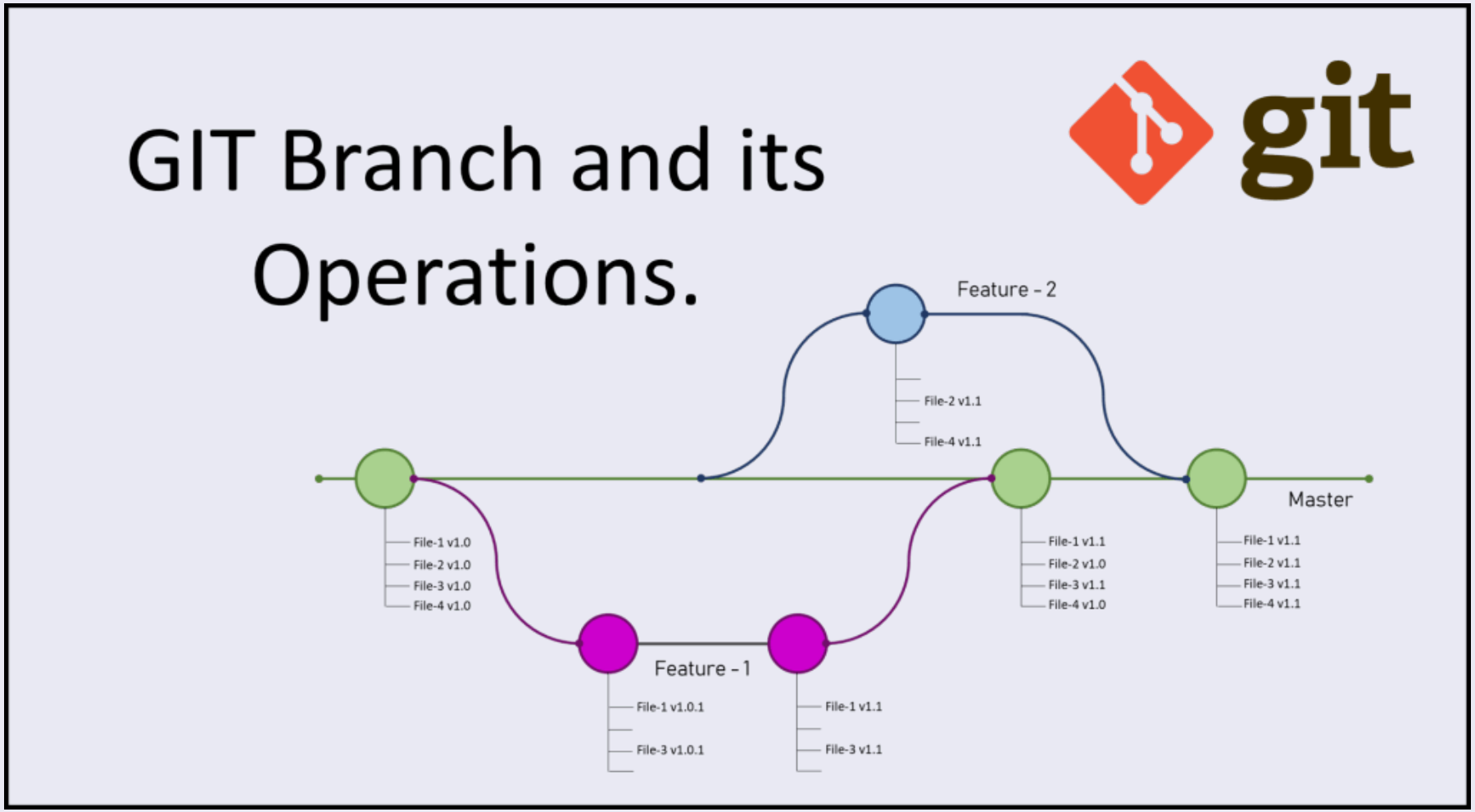
Si se ve complejo, probablemente lo es.

Lo podemos ver por medio de versiones.



Si se ve complejo, probablemente lo es.

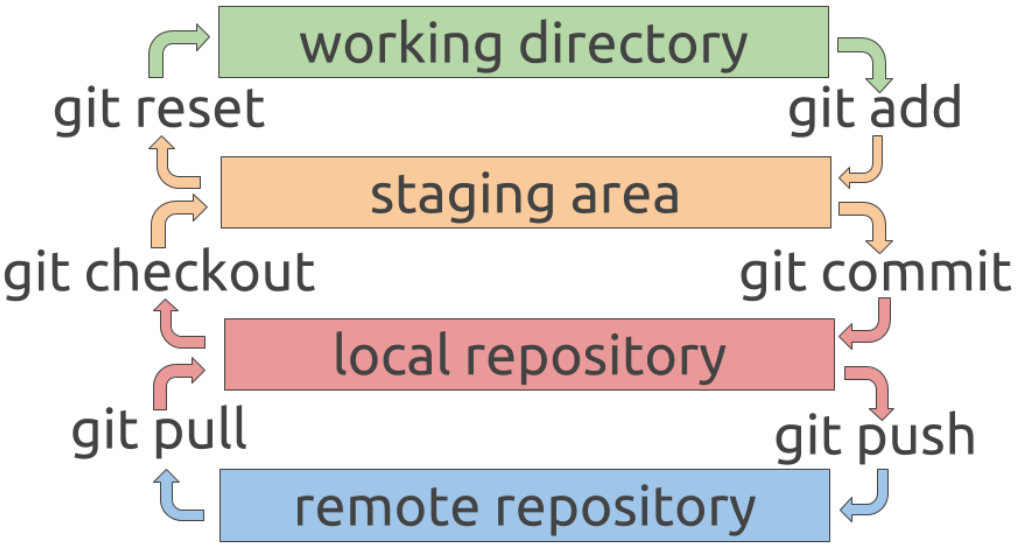
Podemos crear ramas para secciones alternativas.



Distribución

Todo inicia con:

git init

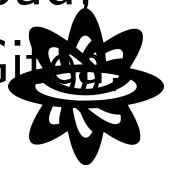


Para verificar como esta todo:

git status

Se distribuye en 4 ambientes: 3 estados locales, 1 remoto

- Working directory: donde estoy trabajando.
- Staging area: Agrego archivos y preparo.
- Repository: almaceno mis versiones finales
- Repositorio local, dentro de mi computadora.
- Repositorio remoto en la nube.
- Servidor propio, GitHub, GitLab, Bitbucket, Launchpad, SourceForge, Git Kraken, Gitea, Git Bucket.



Como inicio a usar git?

Instalación

$sudo apt install git-all

Configurar quien soy

$git config −−global user.name "Juan Caminante"
$git config −−global user.email johnyw@example.com

Ejemplo 1: Crear archivos y moverlos entre los ambientes de git

En una terminal creamos dos archivos y modificamos su flujo de trabajo.

- mkdir ejegit && cd ejegit
  - Crea carpeta e ingresa a ella
- touch file1 file2
  - creamos file1 y file2
- echo hola >> file1
  - escribimos en file1
- git init
  - iniciamos git, carpeta se convierte en Working directory
- git status
  - revisamos el contenido
- git add file1
  - agregamos file1 a staging area
- git status
  - muestra estados de los archivos
- git add file2
  - agregamos file2 a staging area
- git status
  - muestra estados de los archivos
- git commit
  - abre su editor de texto, siempre agregar un comentario. Se mueve todo al repositorio
- $git log
  - vemos los elementos de nuestro commit junto a su hash

Ejemplo 2: Deshacer y ver diferencias en cambios de archivo

Utilizando los archivos del ejemplo anterior modificamos un archivo y luego comparamos con el commit realizado.

- git status
  - muestra que no hay nada a que hacer commit
- echo nueva linea >> file1
  - escribimos en file1
- git status
  - muestra que hay un archivo en el WD
- git checkout -- file1
  - Deshace la modificación, remplazando con lo del ultimo commit
- git status
  - muestra que no hay modificaciones
- echo 42 u infidel >> file1
  - modificamos file1
- cat file1
  - muestra diferencia entre file1 del WD y el Rep
- git diff file1
  - agregamos file1 a staging area
- git add file1
  - abre su editor de texto, agregar un comentario. Se mueve todo al repositorio
- git commit
- git log
  - vemos los elementos de nuestros commits junto a cada hash

Ejemplo 3: ignorar archivos

Utilizando los archivos del ejemplo anterior, crearemos una carpeta y dos archivo para que sean ignorados por git.

- mkdir nover
- touch nover.txt
- touch .gitignore
- echo nover >> .gitignore
- echo nover.c >> .gitignore
- git status
- touch nover/otro.txt
- git status
- git add .gitignore
- git commit -m “comentario”
- git log

- creamos una carpeta
- creamos un archivo
- necesitamos un archivo de nombre .gitignore
- escribimos el nombre de la carpeta y del archivo dentro del archivo .gitignore
- muestra que solo existe .gitignore
- creo un archivo dentro de la carpeta ignorada
- muestra que solo existe .gitignore
- agregamos .gitignore a staging area
- se mueve todo al repositorio con la instrucción (-m) podemos agregar un comentario en linea
- vemos los elementos de nuestros commits junto a cada hash

Ejemplo 4: Ramificaciones

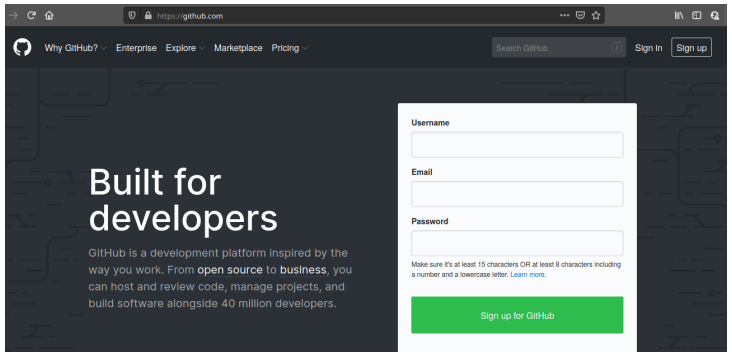
Utilizando los archivos del ejemplo anterior, crearemos una rama básica.

- git branch - muestra que solo existe la rama master
- git branch OTRAR - creamos otra rama en nuestro proyecto
- git branch - muestra las ramas existentes
- git checkout OTRAR - cambiamos a la rama OTRAR
- git branch - muestra que estamos en rama OTRAR
- touch f1 f2 f3 f4 f5 - creamos varios archivos
- git add . - agregar los archivos del proyecto usamos (.)
- git status - muestra los files en staging area
- git commit -m "comentar" - se mueve todo al repositorio.
- git log - vemos los elementos de nuestros commits junto a cada hash
- git checkout master - cambiamos a la rama master

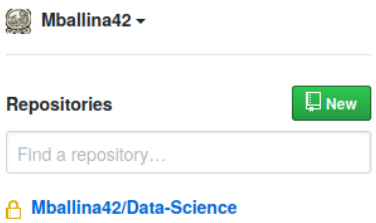
Para unir las dos ramas, utilizar la instrucción *git merge nombre-rama*

GitHub

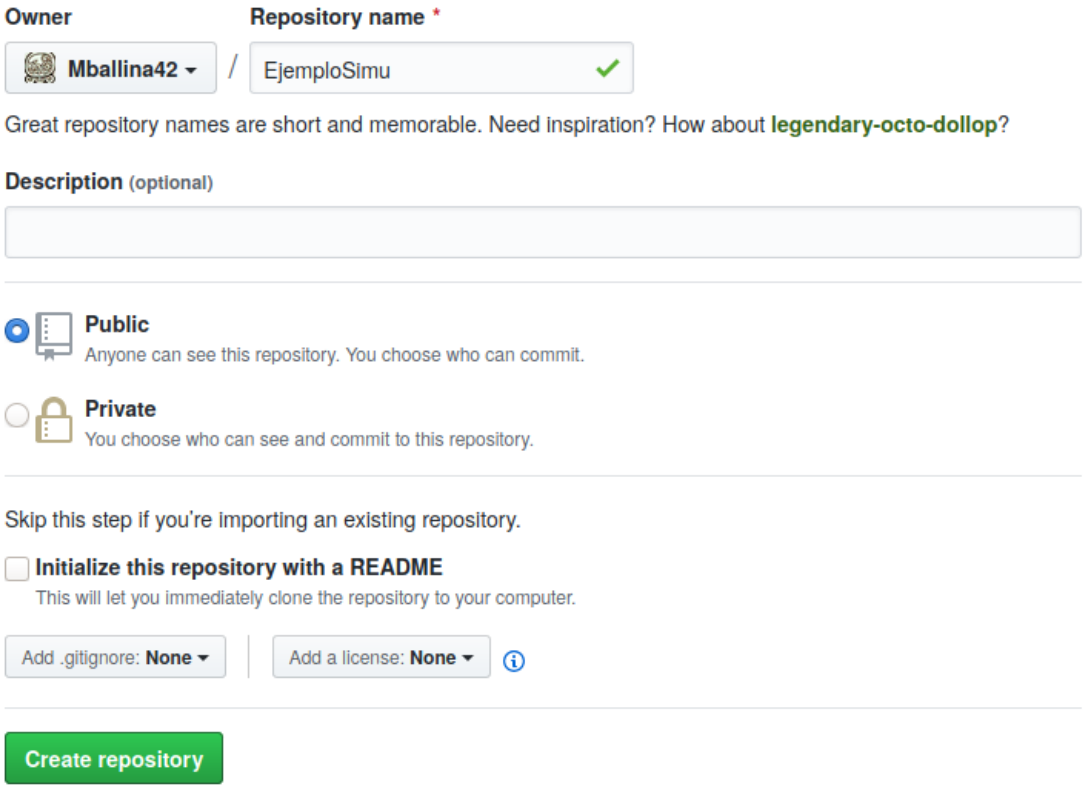
Creamos una cuenta.



Creamos un nuevo repositorio.



Configuramos repositorio.



Solo sigue la instrucciones

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

or HTTPS SSH https://github.com/Mballina42/EjemploSimu.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

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

```
echo "# EjemploSimu" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/Mballina42/EjemploSimu.git
git push -u origin master
```

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

```
git remote add origin https://github.com/Mballina42/EjemploSimu.git
git push -u origin master
```

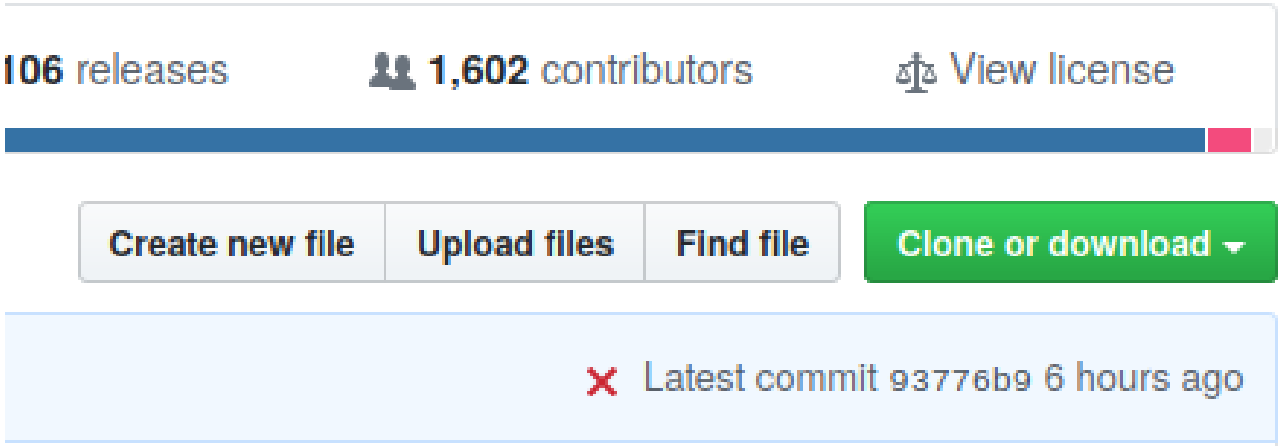
...or import code from another repository

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

Import code

Clonar Repositorios

Dentro de mi terminal debo utilizar
git clone https://github.com/usuario/EjemploSimu.git



La dirección se obtiene dentro del repositorio de git en el boton:
Clone or download