

TEMA VI

CONTROL DE VERSIONES

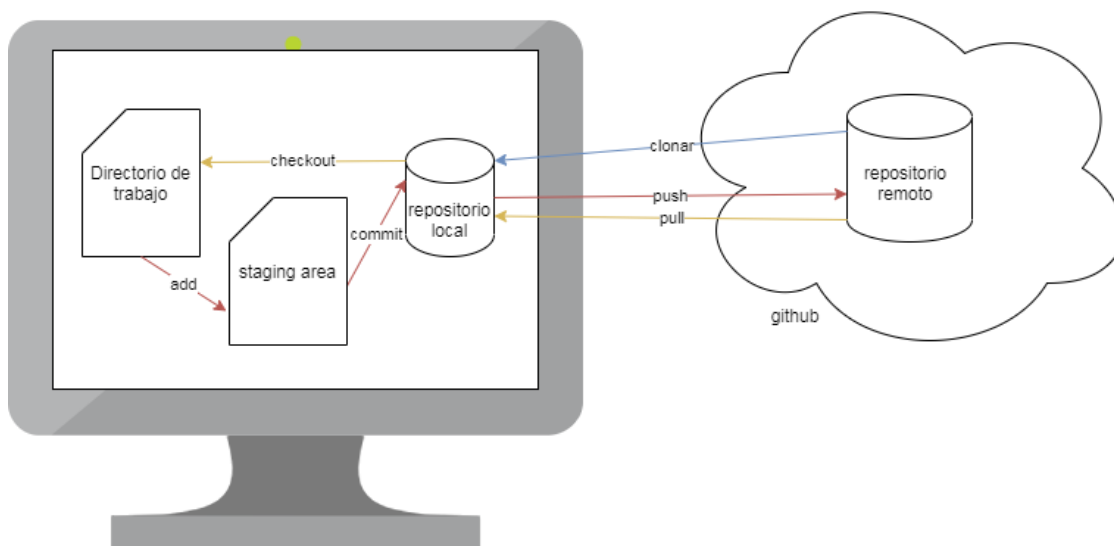
GITHUB

Git es “distribuido”, es decir que cada programador debe tener en local su repositorio. Obviamente, es necesario que esas instantáneas que vamos teniendo no estén solo en nuestro disco duro, necesitamos subirlas a la nube. Para ello usaremos github. Se trata de conectar nuestros repositorios locales con un repositorio remoto. Este repositorio remoto es github.

Una URL remota es la manera de Git de decir "el lugar donde se almacena tu código". Esa URL podría ser tu repositorio en GitHub o la bifurcación de otro usuario o incluso en un servidor completamente diferente.

Git asocia una URL remota con un nombre y tu remoto predeterminado generalmente se llama origin

Git es la tecnología y GitHub es un hosting de repositorios

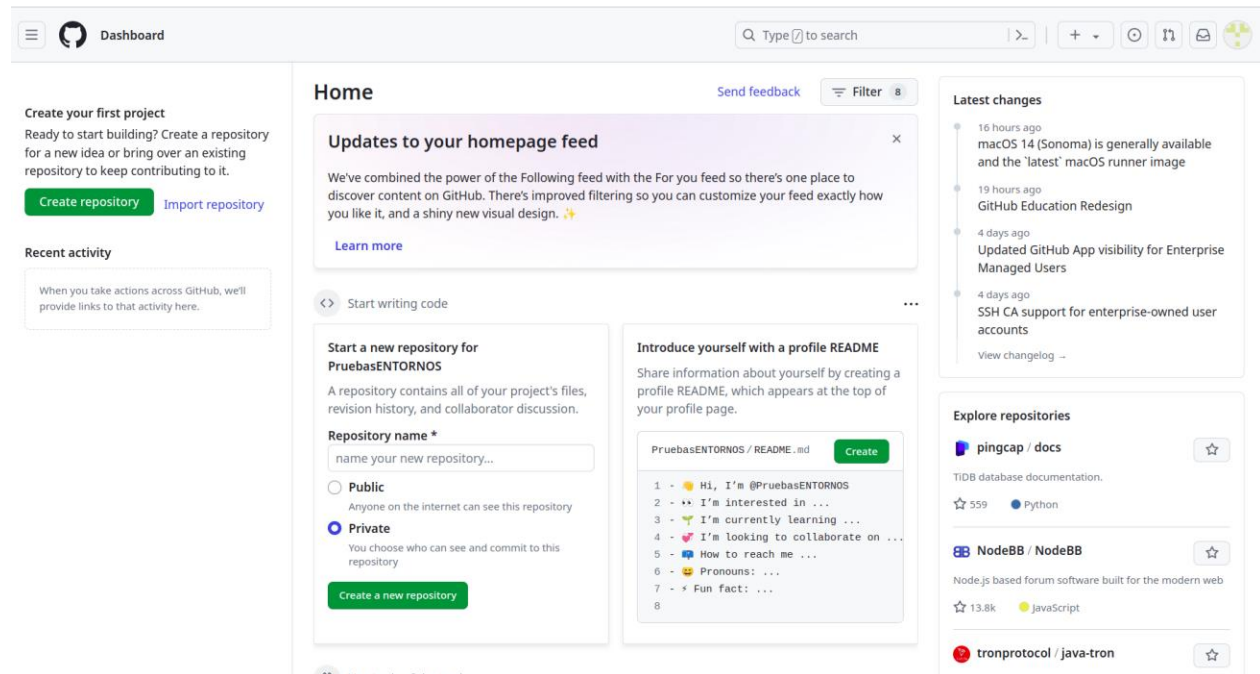


Debemos registrarnos en su web: github.com

Nos pide una dirección de email, password y nombre de usuario

Hay que verificar la cuenta, a veces este es un proceso tedioso.

Antes de entrar al repositorio quizás pida alguna configuración (estudiante, profesor, número de personas que trabajarán con nosotros en el repositorio...)

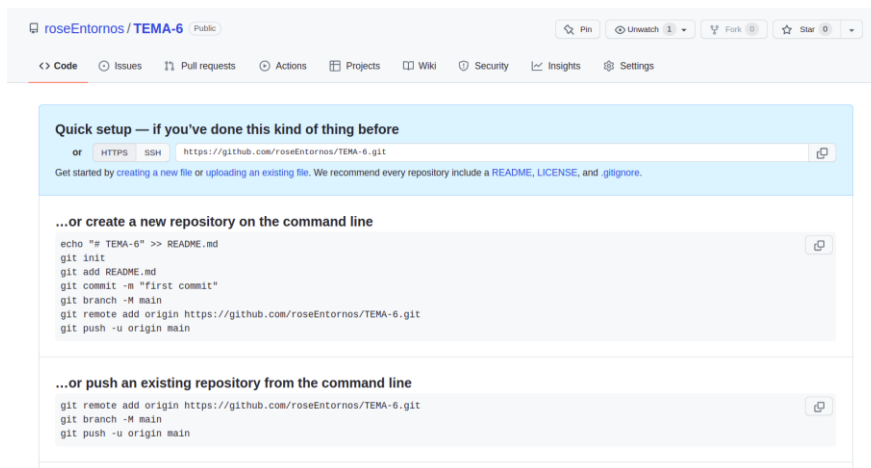


Ahora sólo nos falta crear el repositorio (puedo tener varios asociados a esta cuenta y usuario)

Trabajar con el repositorio

- Darle un nombre, será el nombre de usuario/nombrererepositorio
- La descripción es opcional
- Decidir si es público o privado
- El resto de las opciones las veremos más adelante

A continuación, nos indica los comandos según las operaciones que queramos hacer



- Podemos crear uno nuevo, se puede crear desde cero el proyecto directamente en github
- Podemos crearlo a partir de uno local
- Podemos importarlo de otro repositorio remoto

Nuestro caso es el segundo

1. Con la primera instrucción, git asocia la url de mi repositorio remoto con un nombre, generalmente se pone origin pero puedo darle el que yo quiera.

git remote add nombre url

A partir de este momento puedo usar ese nombre en lugar de la URL. (Sólo lo haremos la primera vez)

Podemos usar el comando **git remote -v** para listar todas las asociaciones que tenemos hechas

```
prueba https://github.com/roseEntornos/PRUEBAS.git (fetch)
prueba https://github.com/roseEntornos/PRUEBAS.git (push)
rosa@rosa-Essential14L:~/Escritorio/GIT/Documentos$
```

Ha asociado dos URL (lectura y escritura) al nombre prueba

Si genero varios remotos, asociados a repositorios de colaboradores, puedo trabajar con cada uno de ellos. Puedo tener distintos permisos en cada repositorio.

Si quiero traerme algo que está en el repositorio de un colaborador

git fetch "nombre"

Por ejemplo, puedo traerme a mi repositorio alguna rama de un colaborador. Después hablaremos de este comando

Si quiero eliminar algún remoto **git remote rm nombre**

```
origin https://github.com/roseEntornos/PRUEBAS.git (fetch)
origin https://github.com/roseEntornos/PRUEBAS.git (push)
otro https://github.com/roseEntornos/PRUEBAS.git (fetch)
otro https://github.com/roseEntornos/PRUEBAS.git (push)
prueba https://github.com/roseEntornos/PRUEBAS.git (fetch)
prueba https://github.com/roseEntornos/PRUEBAS.git (push)
rosa@rosa-Essential14L:~/Escritorio/GIT/Documentos$
```

Esto no tiene mucho sentido porque tengo el mismo repositorio remoto asociado a muchos "nombres". Debería borrar y dejar sólo uno.

```
rosa@rosa-Essential14L:~/Escritorio/GIT/Documentos$ git remote rm origin
rosa@rosa-Essential14L:~/Escritorio/GIT/Documentos$ git remote rm otro
rosa@rosa-Essential14L:~/Escritorio/GIT/Documentos$ git remote -v
prueba https://github.com/roseEntornos/PRUEBAS.git (fetch)
prueba https://github.com/roseEntornos/PRUEBAS.git (push)
rosa@rosa-Essential14L:~/Escritorio/GIT/Documentos$
```

2. De las ramas hablaremos más tarde, ahora sólo tendremos la principal. Con esta instrucción se llamará main en lugar de master

git branch -M main

3. Con la tercera instrucción, subimos el contenido de nuestro repositorio local al remoto

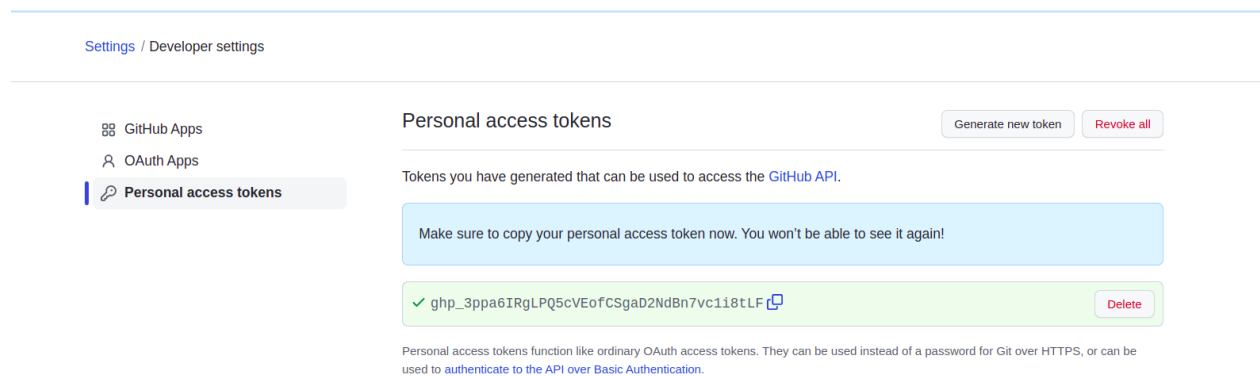
git push -u origin main

En nuestra consola ejecutaremos esas tres instrucciones.

Estas operaciones requieren autenticación, por motivos de seguridad github desde 2020 no permite autenticarse con contraseñas. Lo hace a través de un token personal que te proporciona. Necesitamos obtenerlo

<https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>

En esa dirección nos dicen como hacerlo



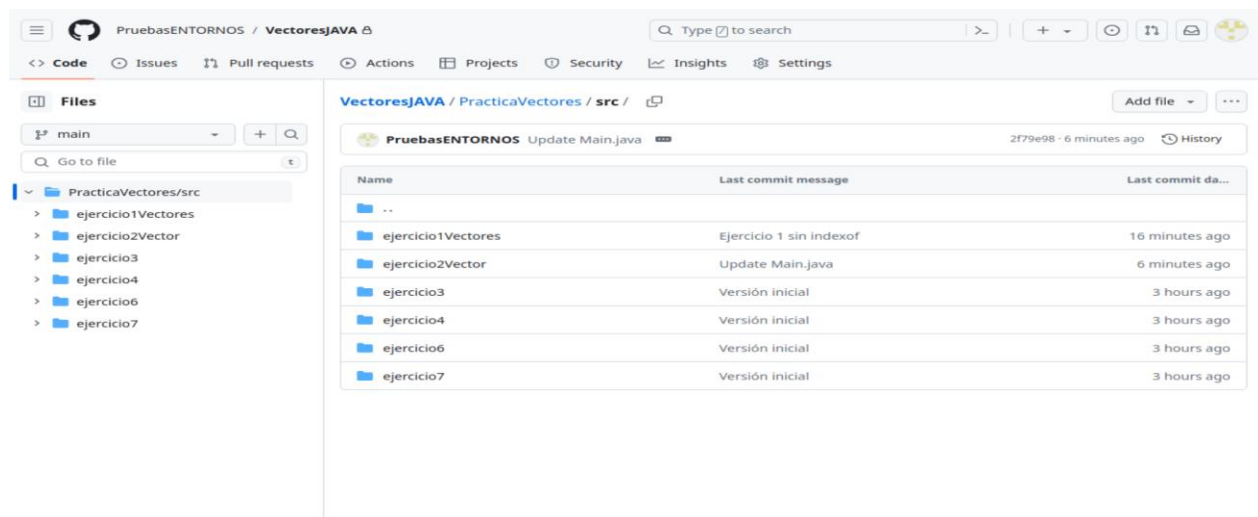
El resultado es el que se muestra en la imagen. Puedo tener tantos repositorios como sea necesario.

Hay que marcar la opción repo

Una vez que tenemos adjudicado nuestro token, cuándo nos pida la password de nuestro usuario daremos este dato

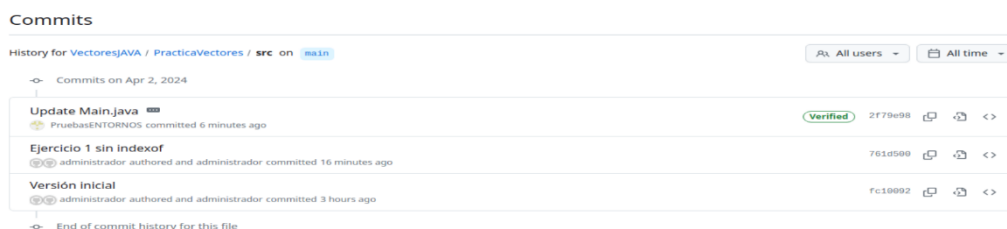
```
rosa@rosa-Essential14L:~/Escritorio/JAVA$ git push -u origin main -f
Username for 'https://github.com': roseEntornos
Password for 'https://roseEntornos@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/en/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for information on currently recommended modes of authentication.
fatal: Autenticación falló para 'https://github.com/roseEntornos/EJEMPLOENTORNOS/'
rosa@rosa-Essential14L:~/Escritorio/JAVA$
rosa@rosa-Essential14L:~/Escritorio/JAVA$ git push -u origin main -f
Username for 'https://github.com': roseEntornos
Password for 'https://roseEntornos@github.com':
Enumerando objetos: 89, listo.
Contando objetos: 100% (89/89), listo.
Compresión delta usando hasta 8 hilos
Comprimiendo objetos: 100% (71/71), listo.
Escribiendo objetos: 100% (89/89), 40.14 KiB | 4.46 MiB/s, listo.
Total 89 (delta 12), reusados 0 (delta 0), pack-reusados 0
remote: Resolving deltas: 100% (12/12), done.
To https://github.com/roseEntornos/EJEMPLOENTORNOS
 + 3bd8148...decd43f main -> main (forced update)
Rama 'main' configurada para hacer seguimiento a la rama remota 'main' de 'origin'.
rosa@rosa-Essential14L:~/Escritorio/JAVA$
```

Ya tenemos subido en remoto nuestro repositorio local.



Name	Last commit message	Last commit da...
..		
ejercicio1Vectores	Ejercicio 1 sin indexof	16 minutes ago
ejercicio2Vector	Update Main.java	6 minutes ago
ejercicio3	Versión inicial	3 hours ago
ejercicio4	Versión inicial	3 hours ago
ejercicio6	Versión inicial	3 hours ago
ejercicio7	Versión inicial	3 hours ago

Podemos ver los commits (History) que tenemos hechos.



Commit Message	Author	Timestamp
Update Main.java	PruebasENTORNOS	6 minutes ago
Ejercicio 1 sin indexof	administrador	16 minutes ago
Versión inicial	administrador	3 hours ago

Pulsando en uno de ellos veremos el código del archivo que hemos cambiado

Commit

The screenshot shows a GitHub commit diff for the file 'PracticaVectores/src/ejercicio1Vectores/Main.java'. The commit is titled 'Ejercicio 1 sin indexof' and was authored and committed by 'administrador' 18 minutes ago. The diff shows changes to the 'main' method, including imports for Scanner and Vector, and a new 'buscarAlumno' method. The diff is color-coded: green for additions, red for deletions, and blue for changes. The 'main' method now includes a loop that calls 'buscarAlumno' and prints the result. The 'buscarAlumno' method is a new addition that searches for a student in a vector and returns its index or -1 if not found.

Podemos realizar modificaciones en github, seleccionamos el archivo y lo editamos. Si añadimos o cambiamos algo, para guardar en remoto los cambios hacemos un commit

The screenshot shows the 'Commit changes' dialog in GitHub. It has a title bar with a purple icon. Below the title bar, there is a text input field for the commit message, which currently contains 'Create EjemploGIT_1.java'. Below the input field, there is a checkbox labeled 'Commit directly to the main branch.' which is checked. Below the checkbox, there is a link that says 'Create a new branch for this commit and start a pull request. Learn more about pull requests.' At the bottom of the dialog, there are two buttons: 'Commit changes' (green) and 'Cancel' (red).

Commits

The screenshot shows the GitHub commit history for the file 'Main.java'. The history is displayed as a list of commits. The first commit is 'Update Main.java' by 'RTORNOS' committed 6 minutes ago, with a green 'Verified' badge. The second commit is 'Ejercicio 1 sin indexof' by 'administrador' committed 16 minutes ago. The third commit is 'Versión inicial' by 'administrador' committed 3 hours ago. The history is filtered by 'All users' and 'All time'. A black arrow points to the 'Update Main.java' commit.

Con esto, obviamente no hemos sincronizado con nuestro repositorio local, para ello ejecutamos

git pull

Este comando mira si hay cambios en el repositorio remoto frente al local, en caso de haberlos actualiza nuestro repositorio local.

Parece que hace lo mismo que el comando git fetch pero no es así:

El comando **git fetch [repositorio]** nos va a permitir recuperar todos los ficheros de un repositorio remoto que hayan sido modificados por otros colaboradores del proyecto y de los que no disponemos. Este comando tan sólo recupera la información del repositorio remoto y **la ubica en una rama oculta de tu repositorio local**, por lo que no la fusionará automáticamente con tu repositorio local. En este caso tenemos que saber que por cada repositorio remoto que tengamos configurado también tendremos una rama oculta de este.

Después tenemos que fusionar esta rama oculta con la rama local en la que estamos trabajando actualmente y para ello necesitamos hacer uso del comando **git merge**.

El comando **git pull** es una forma de abreviar los procesos que realizan los dos comandos anteriores, por lo que nos permite ahorrar tiempo. Estamos sincronizando y trayéndonos todos los cambios del repositorio remoto a la rama en la que estemos trabajando actualmente, sin necesidad de ejecutar ningún comando extra.

Etiquetas (tag)

Supongamos que ya tenemos listo nuestro proyecto para considerarlo como versión 1.0 en su estado actual, queremos tenerlo en el repositorio para que se lo puedan descargar, lo puedan ver.... Para ello añadimos un tag

git tag nombre [-m descripción]

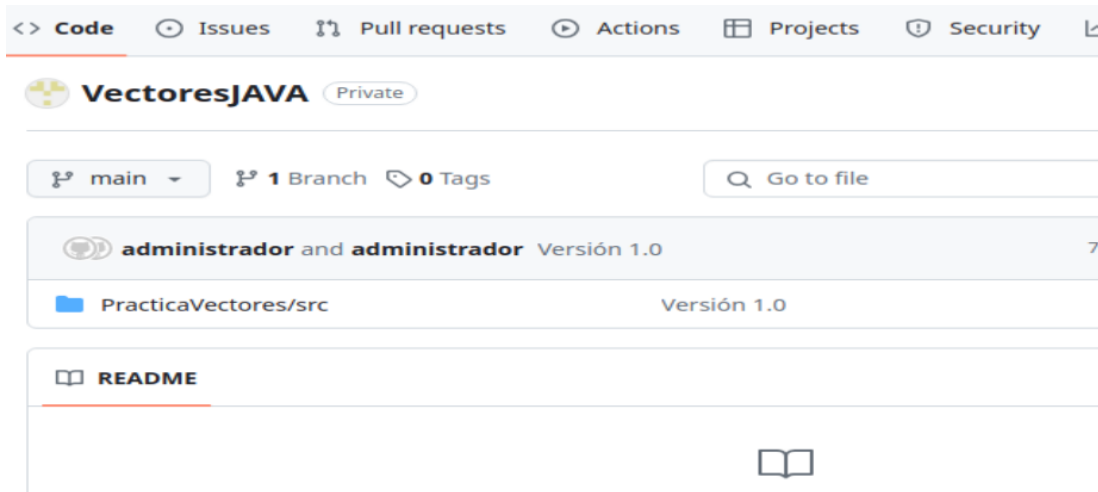

```
rosa@rosa-Essential14L:~/Escritorio/GIT/EjemploGIT_1$ git tag 09-03-2022v1
rosa@rosa-Essential14L:~/Escritorio/GIT/EjemploGIT_1$ git tag
09-03-2022v1
rosa@rosa-Essential14L:~/Escritorio/GIT/EjemploGIT_1$ git tag v1.0 -m "09-03.2022"
rosa@rosa-Essential14L:~/Escritorio/GIT/EjemploGIT_1$ git tag
09-03-2022v1
v1.0
rosa@rosa-Essential14L:~/Escritorio/GIT/EjemploGIT_1$
```

Ya sólo nos queda actualizar en remoto

git push -tags

```
rosa@rosa-Essential14L:~/Escritorio/GIT/EjemploGIT_1$ git push --tags
Username for 'https://github.com': entornosRose
Password for 'https://entornosRose@github.com':
Enumerando objetos: 1, listo.
Contando objetos: 100% (1/1), listo.
Escribiendo objetos: 100% (1/1), 167 bytes | 167.00 KiB/s, listo.
Total 1 (delta 0), reusado 0 (delta 0)
To https://github.com/roseEntornos/OTROREPO.git
* [new tag]          09-03-2022v1 -> 09-03-2022v1
* [new tag]          v1.0 -> v1.0
rosa@rosa-Essential14L:~/Escritorio/GIT/EjemploGIT_1$
```

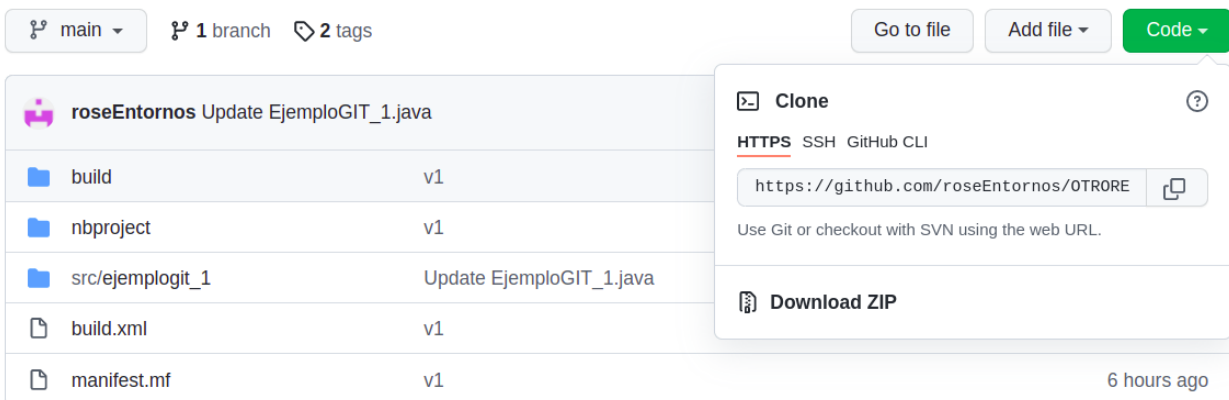
Si vamos a github veremos nuestros tags, pulsando en ellos vemos que nos da la posibilidad de descargar



Clonar un repositorio

Otra funcionalidad de github es la posibilidad de clonar un repositorio. Esto puede ser útil en varias circunstancias: he perdido mi repositorio local (daño en disco, borrado de carpeta...), me incorporo a un proyecto ya empezado, yo no tengo nada del proyecto

En definitiva, tengo que trabajar en un proyecto y no lo tengo localmente.



Copiamos esa URL y desde consola ejecutamos del comando git clone URL

```
rosa@rosa-Essential14L:~/Escritorio/GIT$ git clone https://github.com/roseEntornos/OTROREPO.git
Clonando en 'OTROREPO'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 27 (delta 2), reused 20 (delta 0), pack-reused 0
Desempaquetando objetos: 100% (27/27), 18.23 KiB | 1.52 MiB/s, listo.
rosa@rosa-Essential14L:~/Escritorio/GIT$
```

El repositorio se clonará en la ruta en la que estemos. Se clona todo, código y los commits. También se clonarán las ramas.

A continuación, una imagen que ilustra lo que hemos visto hasta ahora

