

1. ¿Cuál es la diferencia entre la función de "Branch" y la de "Tag" que tienen los sistemas de control de versiones?

Una *Branch* o *rama* es una línea independiente de desarrollo que se deriva del código fuente principal y se utiliza para trabajar bien en características, bien en corrección de errores, sin afectar al código en la rama principal.

Una *etiqueta* o *Tag* es un metadato que ayuda a describir un ítem y se puede usar para documentar diferentes aspectos determinados del código, como parámetros. Se usa para marcar momentos evolutivos clave en el desarrollo del software. Además, facilita encontrar fácilmente una versión específica del software más adelante, sin tener que recordar detalles específicos sobre la revisión.

2. Explica para qué se usa la etiqueta @since. Indica también si hay alguna etiqueta que documente qué excepciones propaga un método.

Etiqueta JavaDoc que guarda la información de la versión en la cual se produjeron cambios en el código asociados a una clase, una interfaz, un método o un campo.

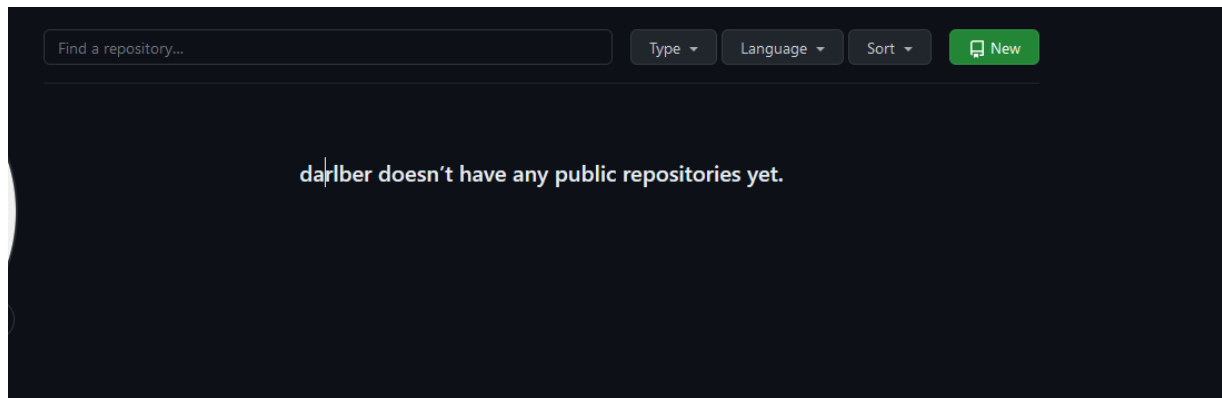
La etiqueta que documenta las excepciones que propaga un método, es @throws.

3. Indica las etiquetas obligatorias usadas en la cabecera de una clase.

No existen etiquetas obligatorias en JavaDoc, no obstante, las más utilizadas y las que los entornos de desarrollo añaden de forma automática, son @author y @version.

GIT


Configurar GIT para el proyecto. Crear un repositorio público en GitHub.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Required fields are marked with an asterisk ().*


Owner *  darlber / **Repository name ***

✔ dam_ENDES is available.

Great repository names are short and memorable. Need inspiration? How about [studious-meme](#) ?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:


☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

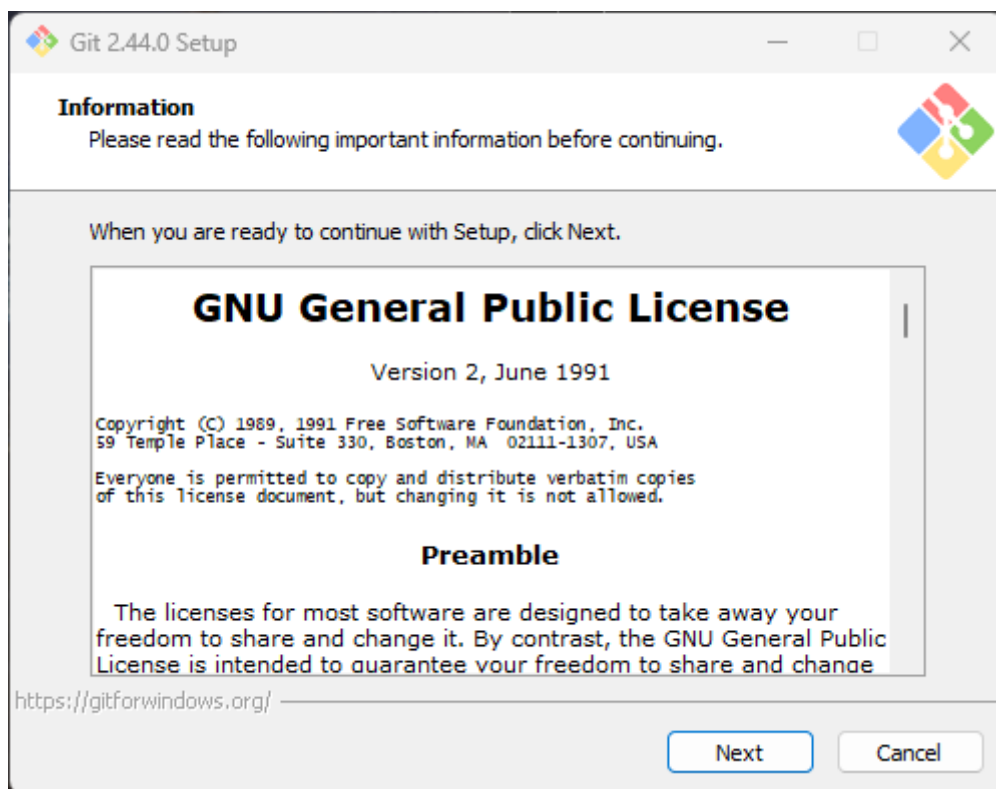
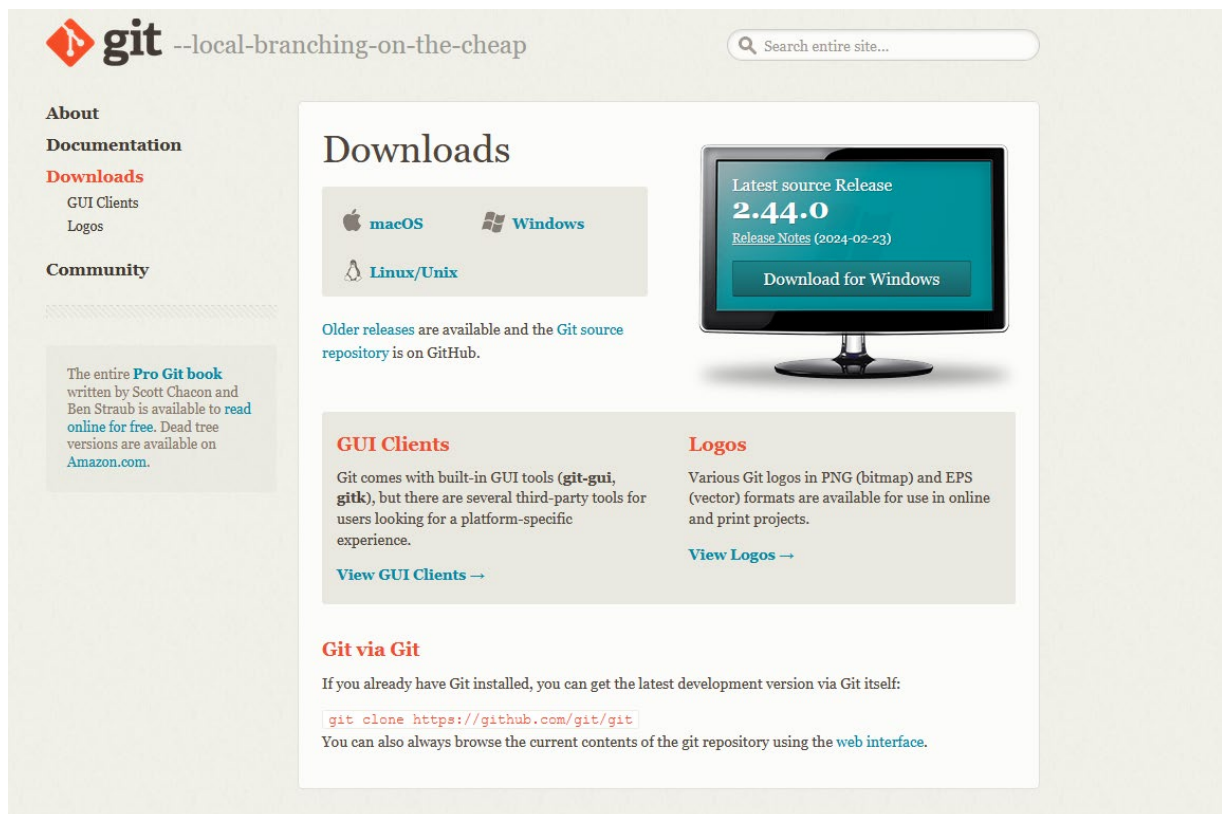
Add .gitignore

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

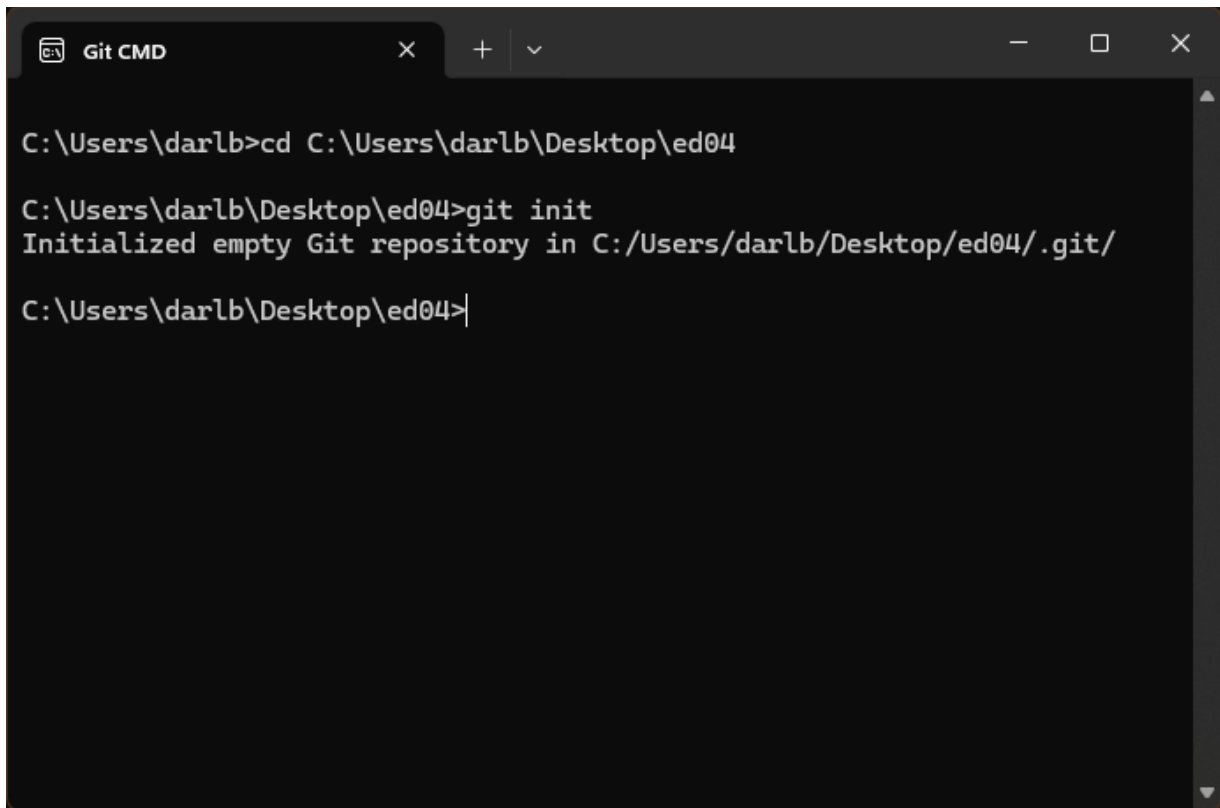
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.



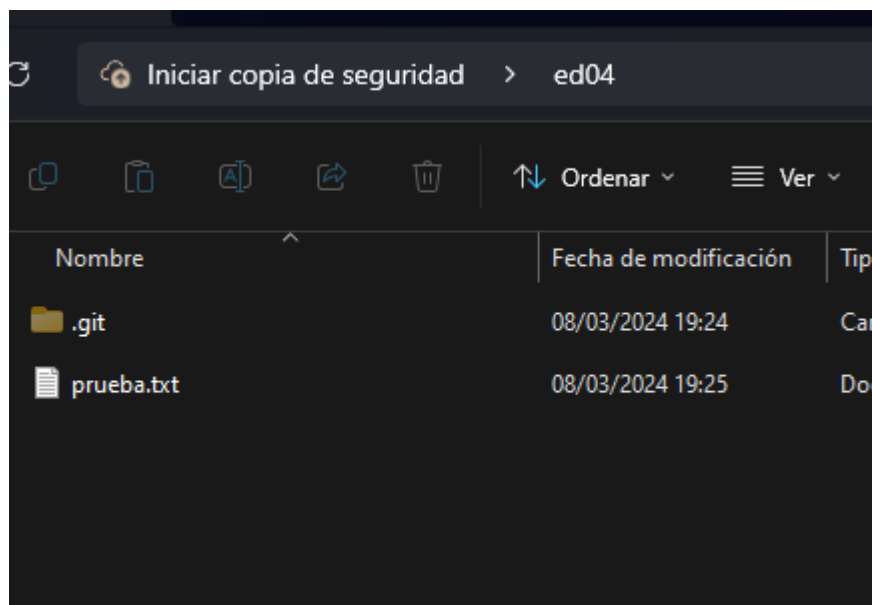
Creamos el repositorio online e instalamos la herramienta Git para nuestro Windows

Realizar, al menos, una operación commit. Comentando el resultado de la ejecución



```
Git CMD
C:\Users\darlb>cd C:\Users\darlb\Desktop\ed04
C:\Users\darlb\Desktop\ed04>git init
Initialized empty Git repository in C:/Users/darlb/Desktop/ed04/.git/
C:\Users\darlb\Desktop\ed04>
```

Inicializamos el repositorio mediante el comando git init



Mediante los comandos git add y git commit confirmamos los cambios de añadir el archivo que hemos creado al repositorio

```
Git CMD
C:\Users\darlb>cd C:\Users\darlb\Desktop\ed04

C:\Users\darlb\Desktop\ed04>git init
Initialized empty Git repository in C:/Users/darlb/Desktop/ed04/.git/

C:\Users\darlb\Desktop\ed04>git add prueba.txt

C:\Users\darlb\Desktop\ed04>git status
On branch master

No commits yet

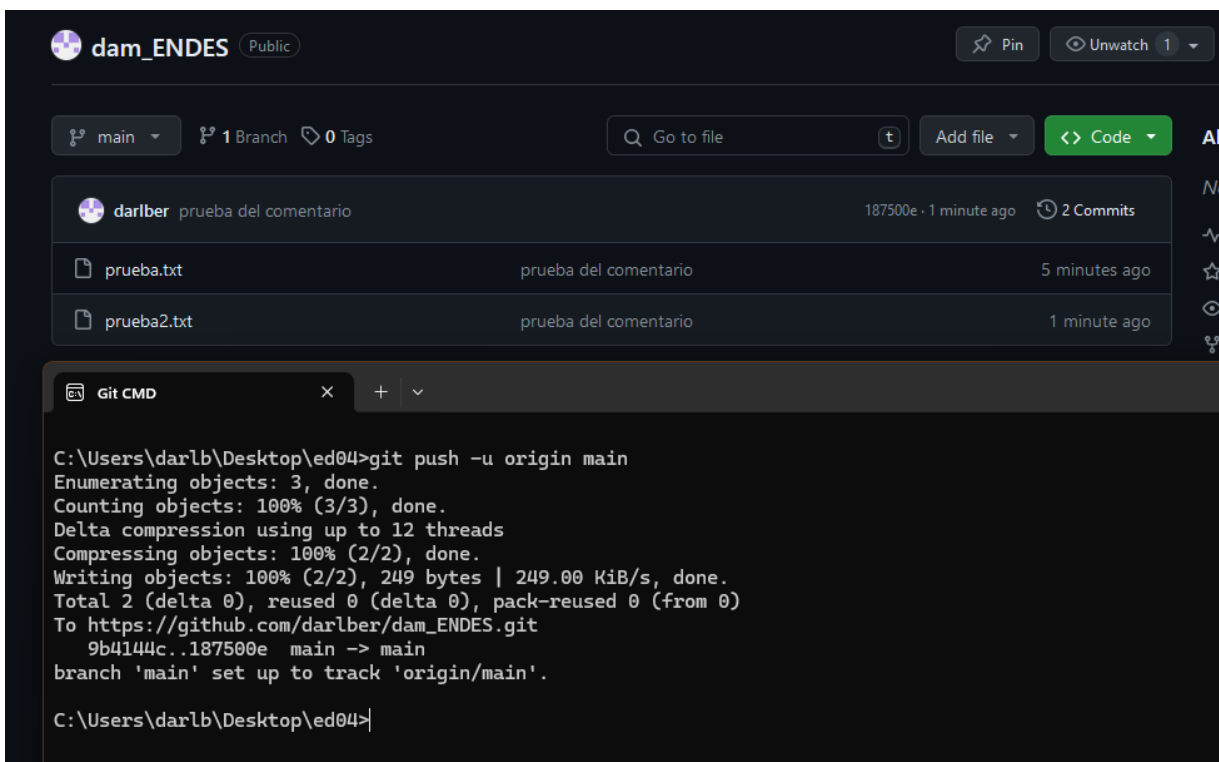
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   prueba.txt

C:\Users\darlb\Desktop\ed04>git commit -m "prueba del comentario"
[master (root-commit) 9b4144c] prueba del comentario
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 prueba.txt

C:\Users\darlb\Desktop\ed04>git status
On branch master
nothing to commit, working tree clean

C:\Users\darlb\Desktop\ed04>|
```

Podemos subir los archivos al repositorio online mediante el comando git push -u y comprobamos



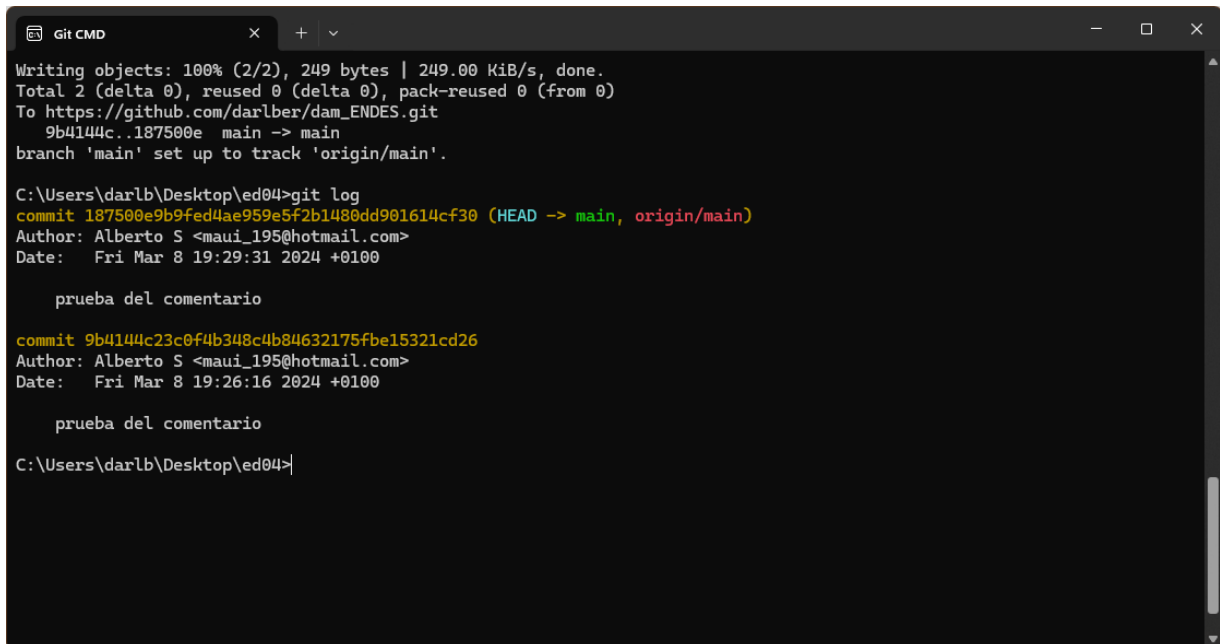
The screenshot shows a GitHub repository interface for a user named 'dam_ENDES' (Public). The repository has 1 branch (main) and 0 tags. A commit by 'darlber' titled 'prueba del comentario' is shown, with a commit hash of 187500e, made 1 minute ago, and containing 2 commits. The commit details show two files: 'prueba.txt' and 'prueba2.txt', both with the message 'prueba del comentario'. Below the repository view, a terminal window (Git CMD) shows the execution of the command 'git push -u origin main'. The output indicates that objects were enumerated, counted, and compressed successfully, and the branch 'main' was set up to track 'origin/main'.

```
Git CMD
C:\Users\darlb\Desktop\ed04>git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 249 bytes | 249.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/darlber/dam_ENDES.git
 9b4144c..187500e  main -> main
branch 'main' set up to track 'origin/main'.

C:\Users\darlb\Desktop\ed04>|
```

Mostrar el historial de versiones para el proyecto mediante un comando desde consola

Esto lo realizaremos con el comando git log



```
Git CMD
Writing objects: 100% (2/2), 249 bytes | 249.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/darlb/dam_ENDES.git
 9b4144c..187500e  main -> main
branch 'main' set up to track 'origin/main'.

C:\Users\darlb\Desktop\ed04>git log
commit 187500e9b9fed4ae959e5f2b1480dd901614cf30 (HEAD -> main, origin/main)
Author: Alberto S <maui_195@hotmail.com>
Date:   Fri Mar 8 19:29:31 2024 +0100

    prueba del comentario

commit 9b4144c23c0f4b348c4b84632175fbe15321cd26
Author: Alberto S <maui_195@hotmail.com>
Date:   Fri Mar 8 19:26:16 2024 +0100

    prueba del comentario

C:\Users\darlb\Desktop\ed04>
```