



# UNIVERSIDAD MAYOR DE SAN ANDRÉS

## CARRERA DE INFORMÁTICA



# **GIT**

# **GITHUB**



Lic. Arnaldo Muñoz Mendoza

# ÍNDICE

- **Instalación**
- **Conceptos básicos**
- **Comandos básicos**
- **Ramas**
- **Github**

# GIT

La página oficial de git es:

**<https://git-scm.com/>**



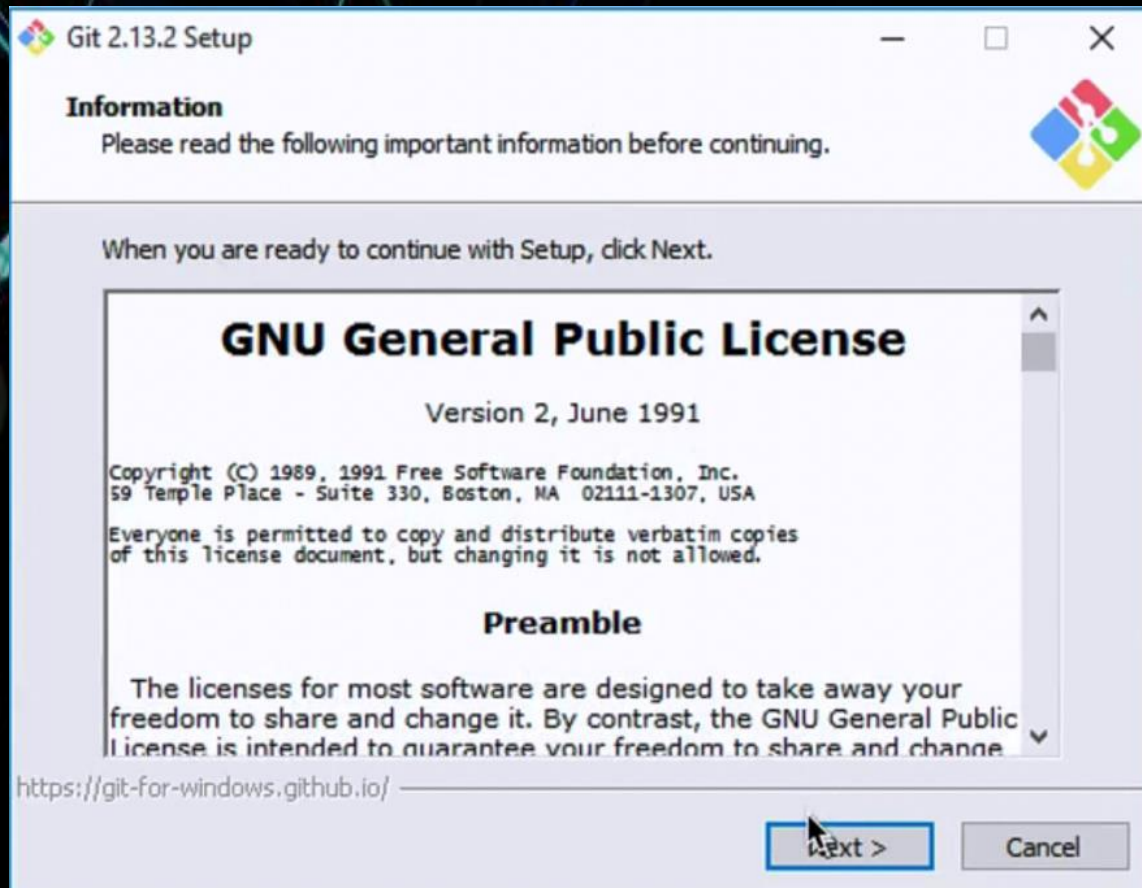


# INSTALACIÓN



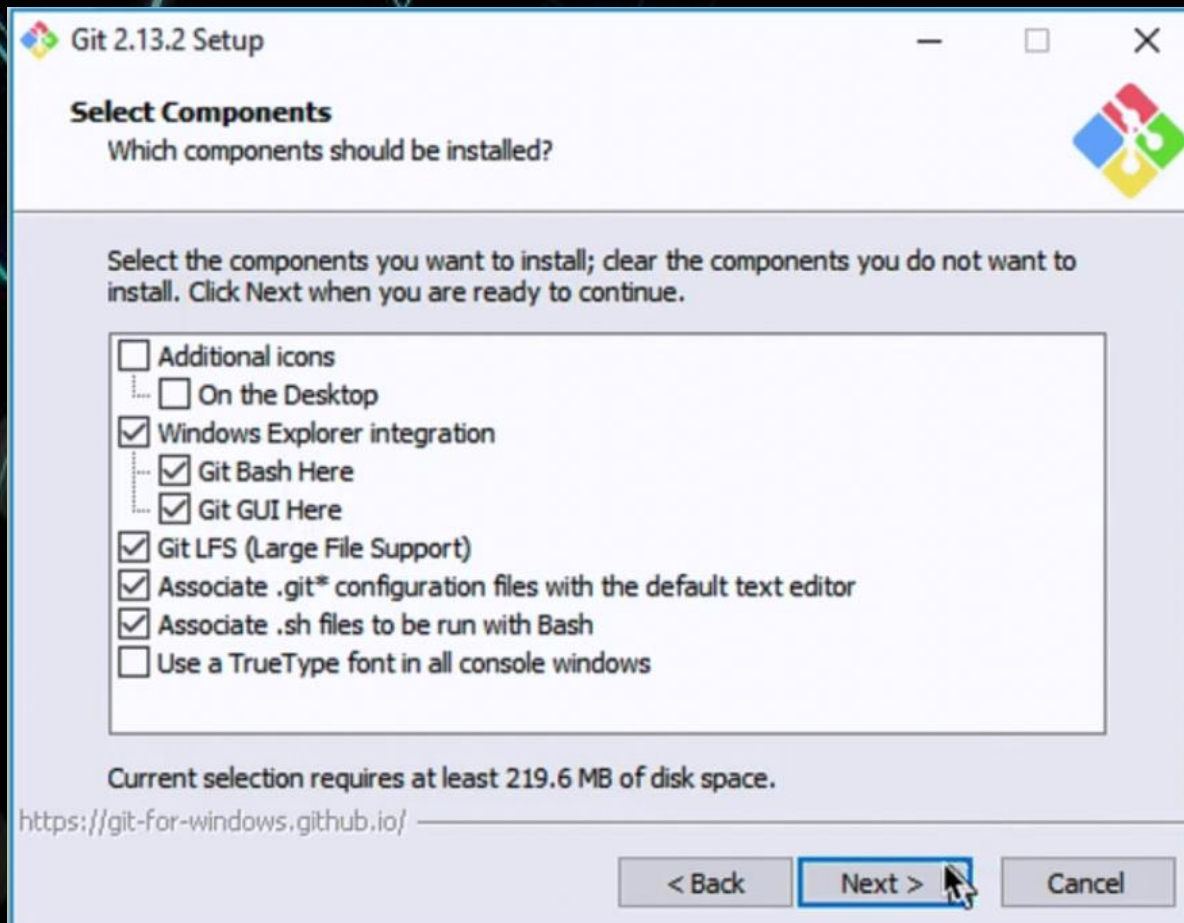
El enlace de descarga de git de la página oficial es:  
<https://git-scm.com/download>

# INSTALACIÓN

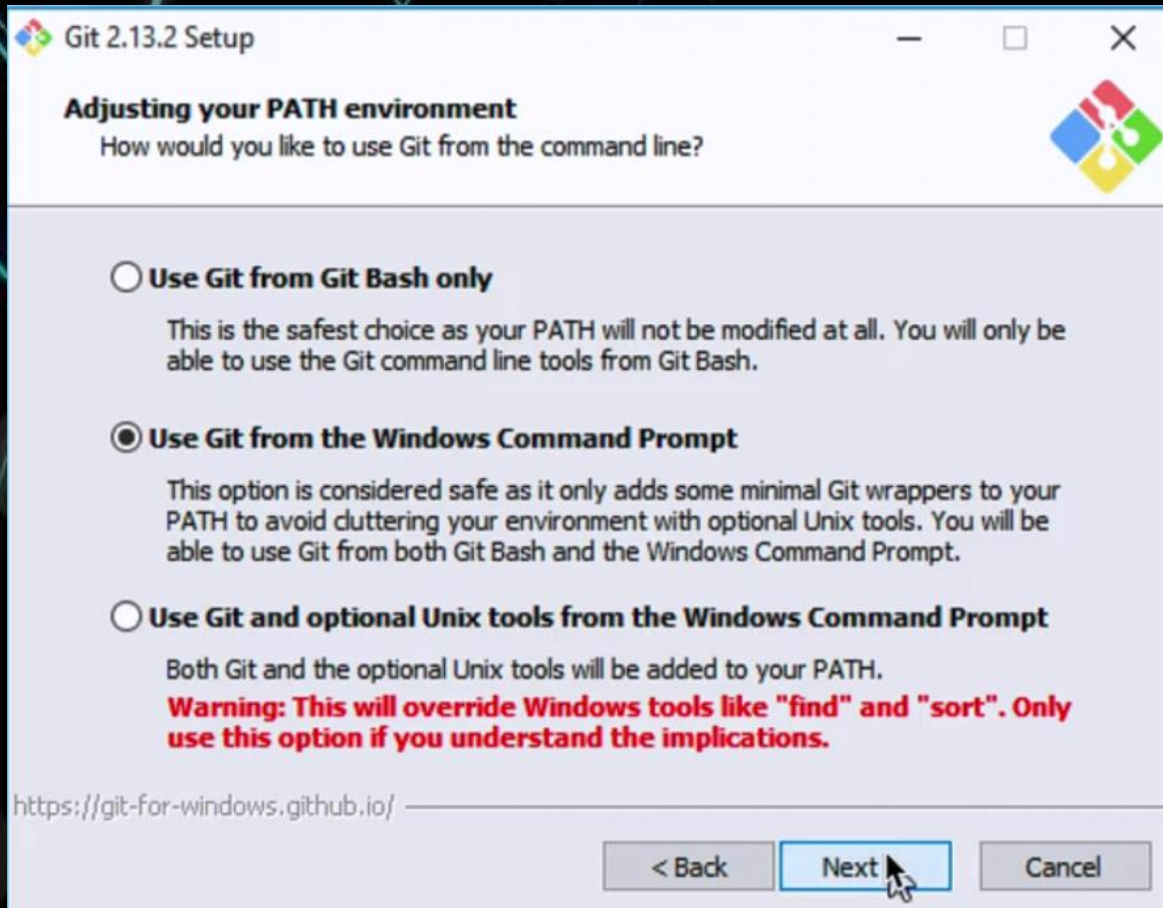




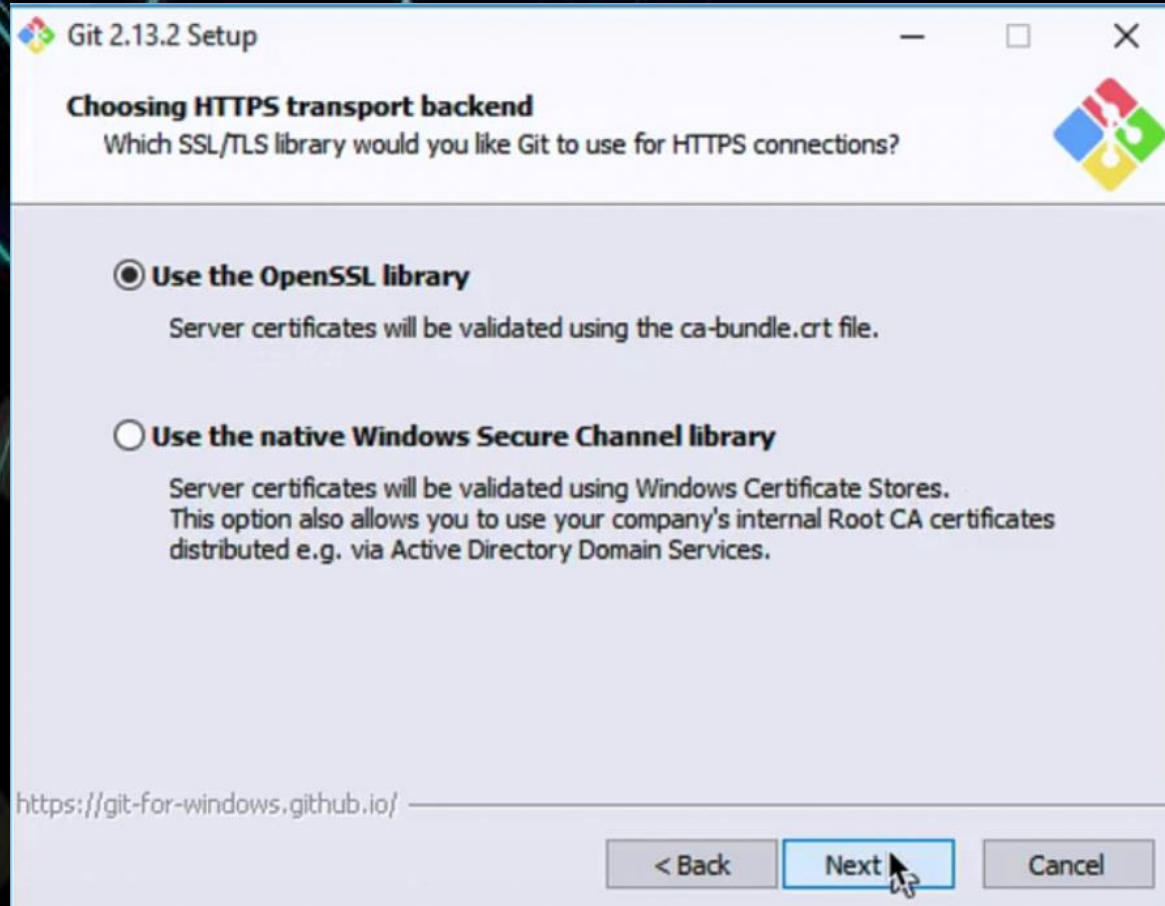
# INSTALACIÓN



# INSTALACIÓN

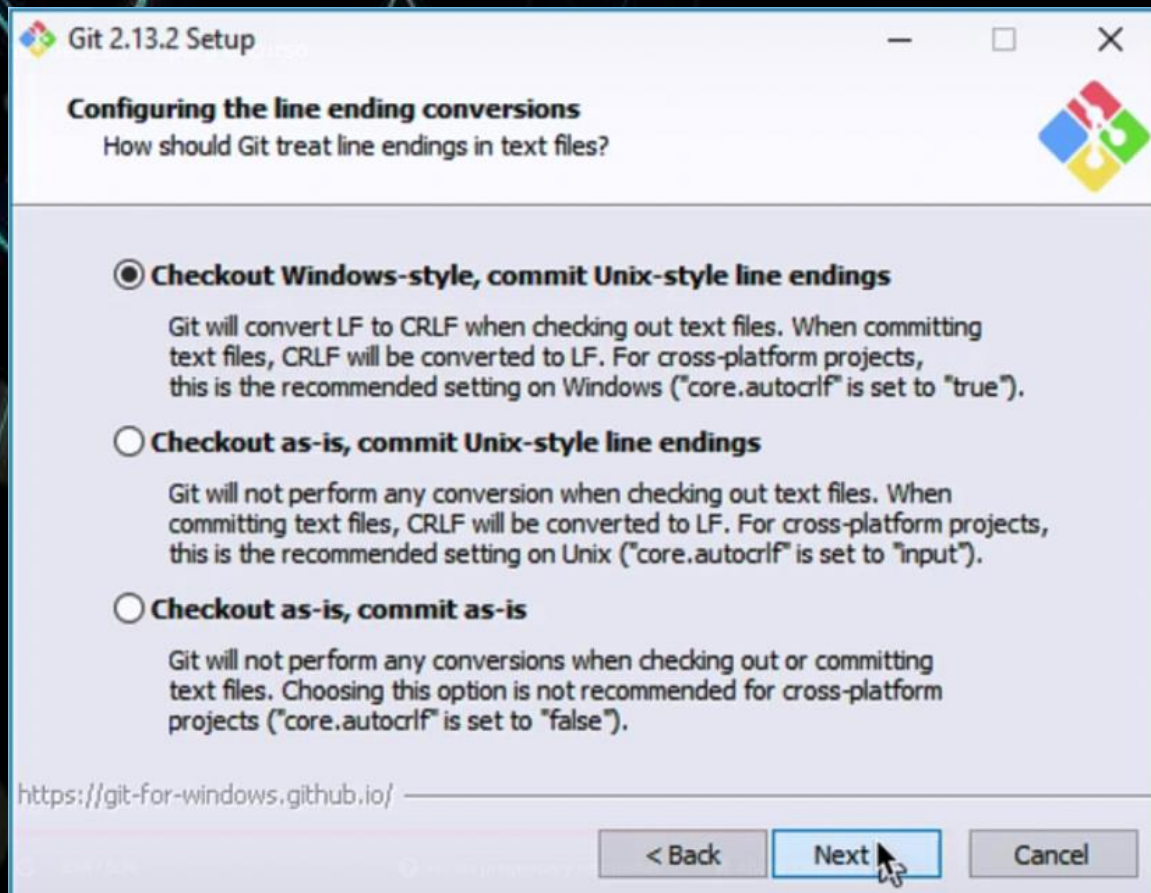


# INSTALACIÓN





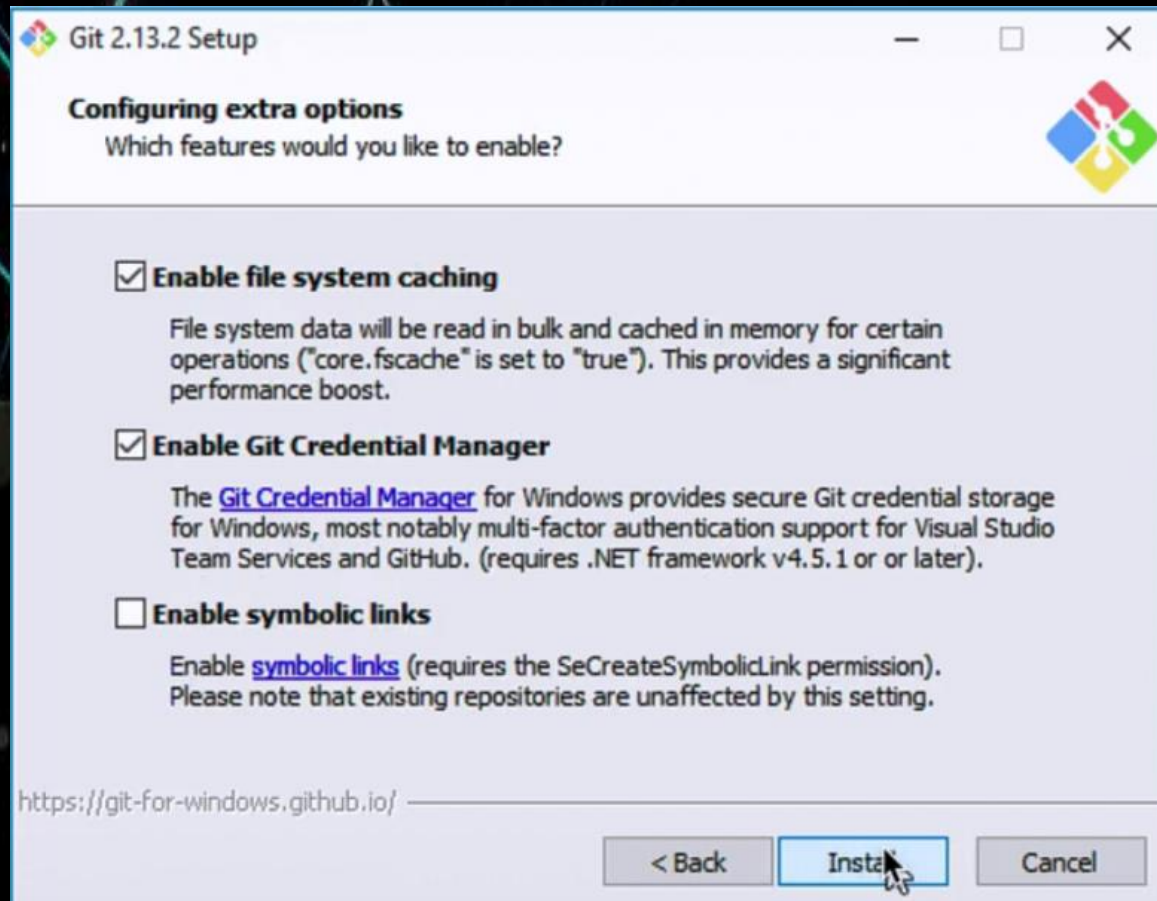
# INSTALACIÓN



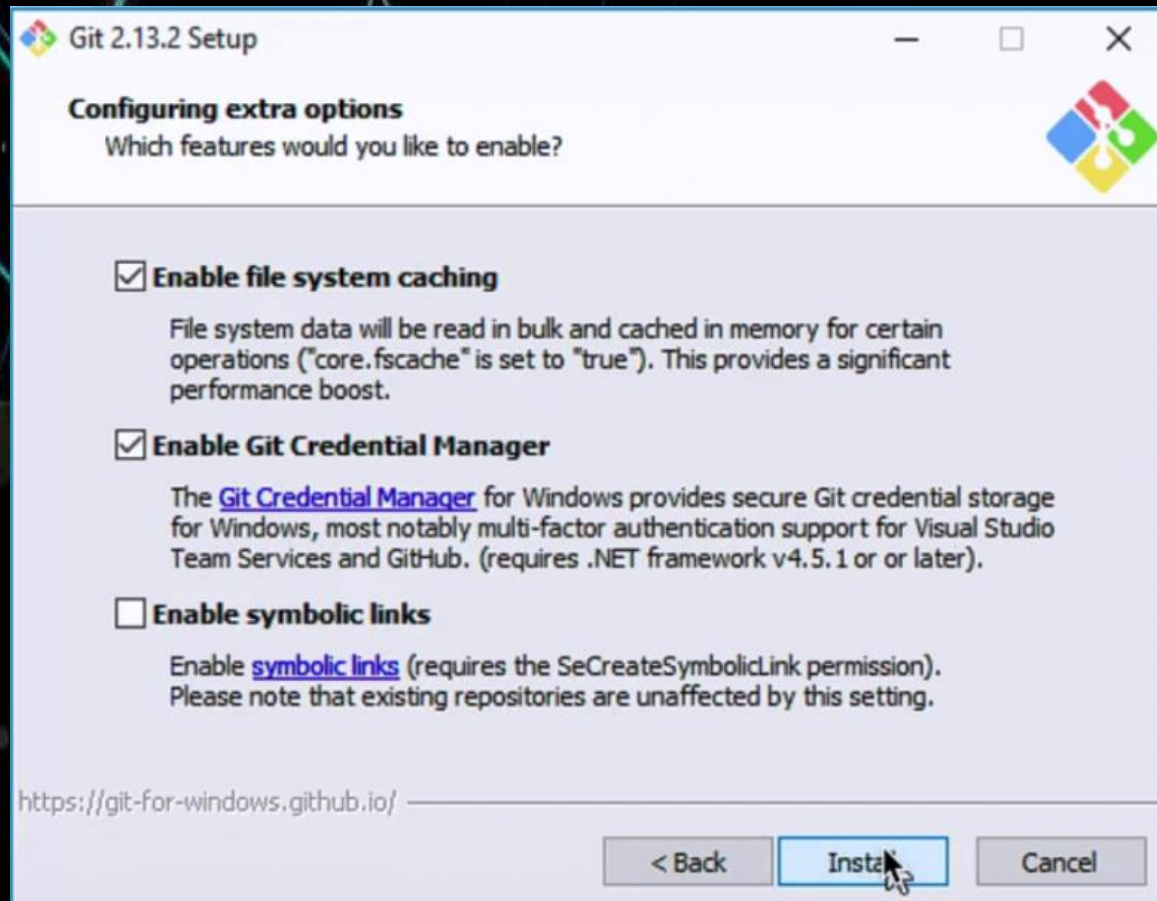
# INSTALACIÓN



# INSTALACIÓN

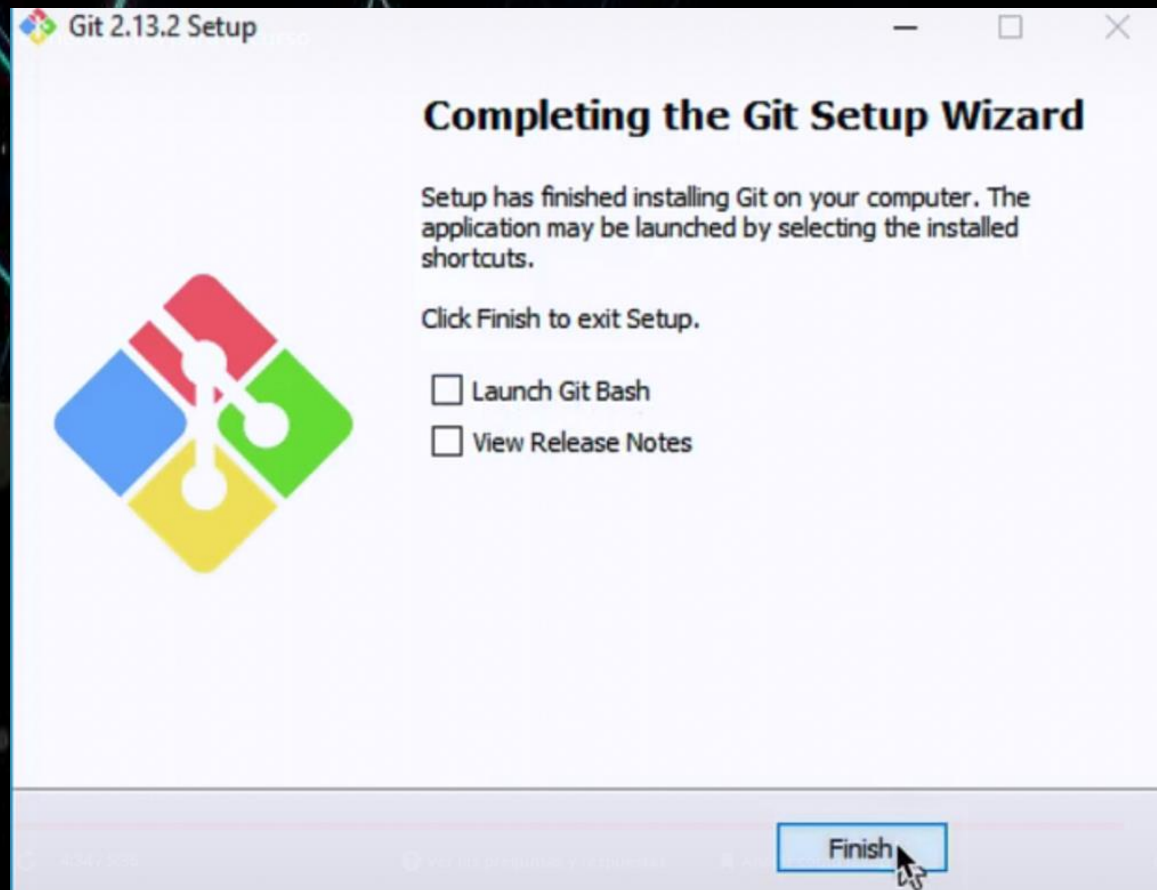


# INSTALACIÓN





# INSTALACIÓN





# CONCEPTOS BÁSICOS

¿Qué es GIT?

¿Por qué nos interesa aprender git?



# EJEMPLO

Se realizara una página web sencilla a la cual se le hará un seguimiento con **git**.

Con este ejemplo se comprenderá mejor la importancia de manejar **git** en cualquier proyecto.



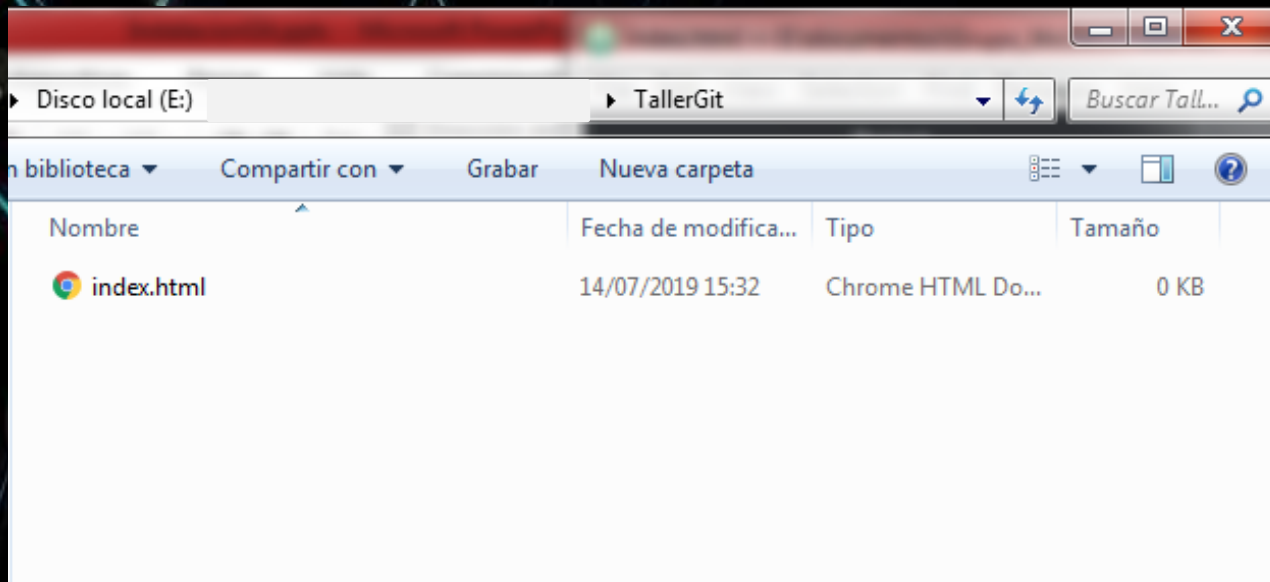
# EJEMPLO

Crear una carpeta (en cualquier directorio), y dentro de la carpeta crear un documento \*.html.

Abrir el editor de texto de su preferencia y abrir la carpeta que se creo anteriormente.

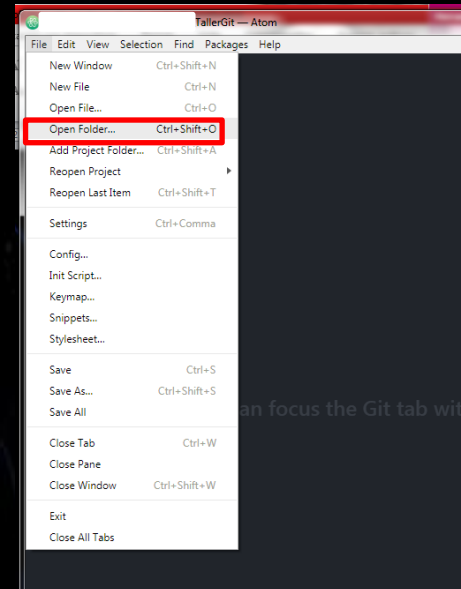
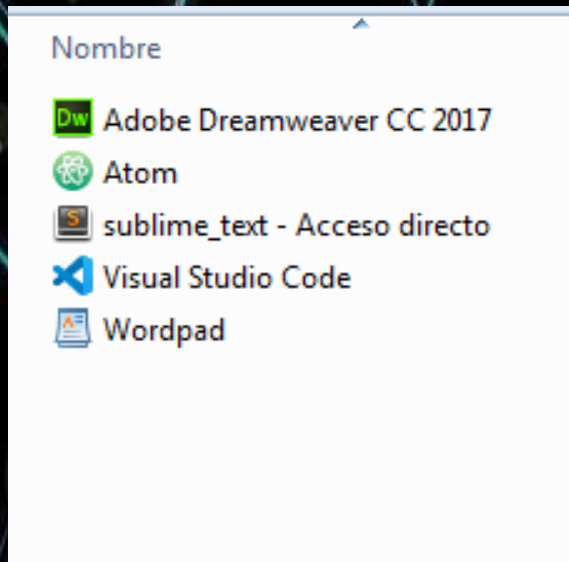


# EJEMPLO



Se creó la carpeta **tallerGit** y dentro de la carpeta se creó el archivo **index.html**.

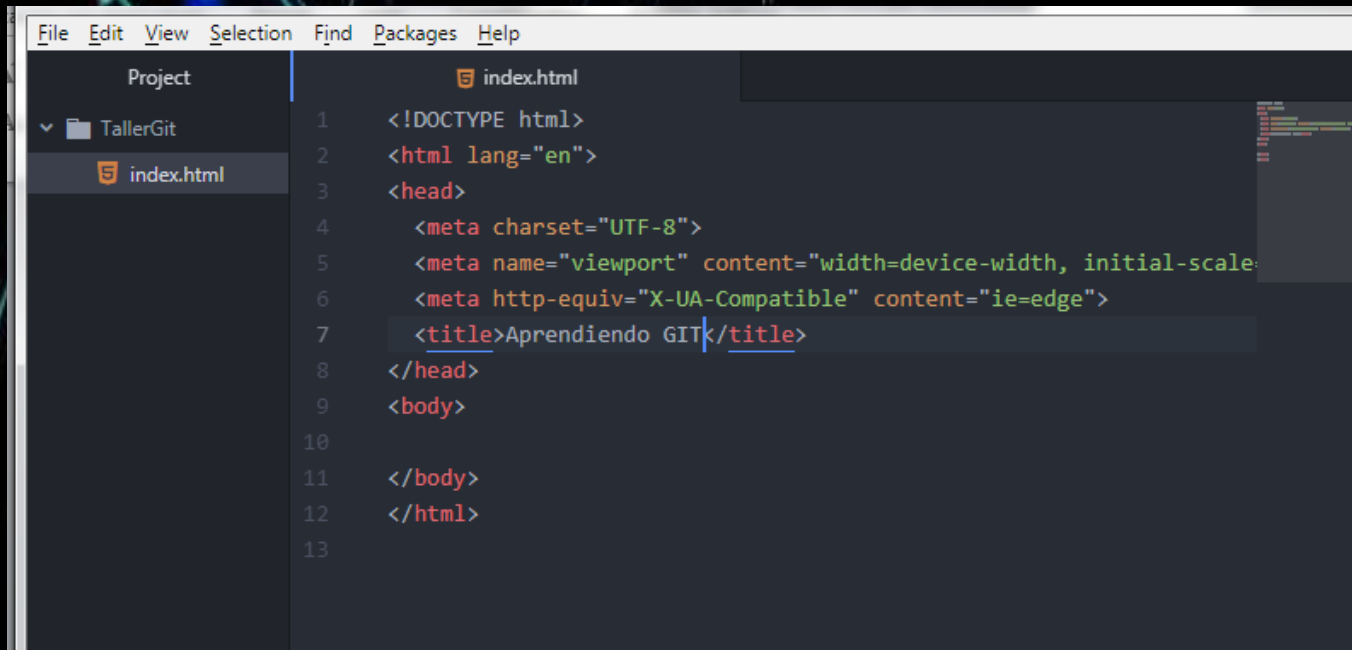
# EJEMPLO



Abrir el editor de texto de su preferencia y luego desde el editor de texto abrir la carpeta que se creó.

# EJEMPLO

El archivo **.html** debe tener la siguiente estructura:

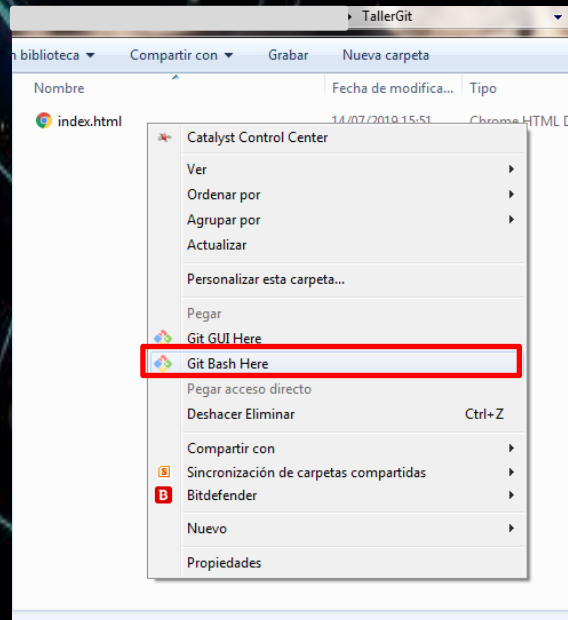
A screenshot of a code editor window. The left sidebar shows a project named 'TallerGit' with a file 'index.html' selected. The main editor area displays the HTML code for 'index.html'. The code is as follows:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta http-equiv="X-UA-Compatible" content="ie=edge">
7   <title>Aprendiendo GIT</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
13
```

# COMANDOS BÁSICOS

## Abrir el Git Bash

Dentro de la carpeta donde se creó el archivo `.html` abrir el gitbash.





# COMANDOS BÁSICOS

## Git Bash

Una vez abierto el **git bash** verificar si el directorio es el correcto. El comando a ejecutar es **pwd**.

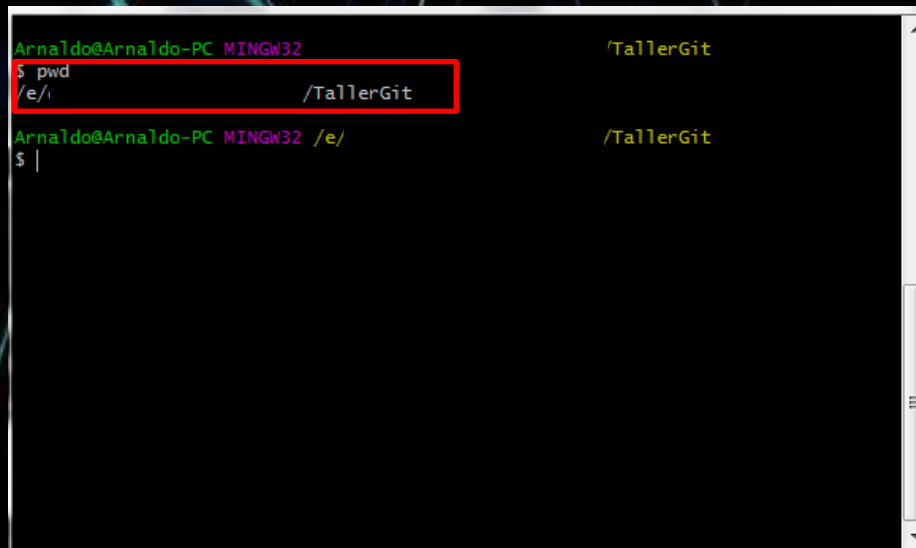
A screenshot of a Git Bash terminal window. The window has a white title bar and a black background. The text inside the terminal is as follows: the first line shows the user 'Arnaldo' at host 'Arnaldo-PC' in a 'MINGW32' environment, followed by the current directory path '/TallerGit' in green. The second line shows a prompt character '\$' followed by a vertical bar '|', indicating the terminal is ready for input.

```
Arnaldo@Arnaldo-PC MINGW32 /TallerGit  
$ |
```

# COMANDOS BÁSICOS

## Git Bash

Una vez abierto el **git bash** verificar si el directorio es el correcto. El comando a ejecutar es **pwd**.

A screenshot of a Git Bash terminal window. The window has a title bar with 'TallerGit' on the right. The terminal shows the prompt 'Arnaldo@Arnaldo-PC MINGW32' followed by the command '\$ pwd'. The output is '/e/\\', which is highlighted by a red rectangular box. Below this, the prompt changes to 'Arnaldo@Arnaldo-PC MINGW32 /e/' and a new line with '\$ |' is shown.

```
Arnaldo@Arnaldo-PC MINGW32 /TallerGit
$ pwd
/e/\\
Arnaldo@Arnaldo-PC MINGW32 /e/
$ |
```

# COMANDOS BÁSICOS

## Credenciales

Antes de inicializar un repositorio primero se deben llenar los credenciales. Estos credenciales ayudaran a tener conocimiento de las personas que realizan algún cambio en el proyecto.

# COMANDOS BÁSICOS

## Credenciales

- `git config --global user.name "nombre de usuario"`
- `git config --global user.email "correo electrónico"`



# COMANDOS BÁSICOS

## Credenciales

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit
$ git config --global user.name "Arnaldo Muñoz Mendoza"

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit
$ git config --global user.email "          @gmail.com"
```

# COMANDOS BÁSICOS

## Credenciales

Para ver si los credenciales fueron llenados correctamente la instrucción a ejecutar es :

- `git config --global -l`

Para editar los credenciales por si existe un error la instrucción a ejecutar es:

- `git config --global -e`

# COMANDOS BÁSICOS

## Credenciales

Al editar los credenciales primero se debe guardar los cambios y luego salir.

La instrucción a ejecutar es:

- **ESC :wq**

En la siguiente imagen se podrá ver mejor esta instrucción.

# Credenciales

[illegible]

# COMANDOS BÁSICOS

## Inicializando Repositorio

Ya llenado los credenciales ahora se debe inicializar un repositorio. La instrucción es :

- `git init`

Al inicializar un repositorio aparecerá una carpeta **oculta** con nombre **.git**, es recomendable no manipular esta carpeta.



# COMANDOS BÁSICOS

## Añadiendo archivos al escenario.

A los archivos que se quieren hacer un seguimiento se les debe añadir al escenario. La instrucción es:

- `git add “nombre de archivo”`

Si se quiere añadir todos los archivos entonces la instrucción será:

- `git add .`

# COMANDOS BÁSICOS

## Commits .

Para que los archivos se encuentren versionados o capturados en el tiempo se debe ejecutar la siguiente instrucción:

- `git commit -m "nombreDelCOmmit"`

# COMANDOS BÁSICOS

## Listando commits .

Para listar todos los commits capturados en el tiempo se debe ejecutar:

- `git log`

# COMANDOS BÁSICOS

## Verificando archivos .

Si se quiere saber el estado de los archivos (archivos con seguimiento/ archivos sin seguimiento). Se debe ejecutar la siguiente instrucción:

- `git status`

Los archivos que no se encuentran en el escenario estarán de color **rojo**.

# EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit
$ git init
Initialized empty Git repository in C:/Users/Arnaldo/Desktop/TallerGit/.git/

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    index.html

nothing added to commit but untracked files present (use "git add" to track)

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git add index.html

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master

No commits yet

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

    new file:   index.html

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git commit -m "Iniciando proyecto"
[master (root-commit) e22a823] Iniciando proyecto
 1 file changed, 12 insertions(+)
 create mode 100644 index.html

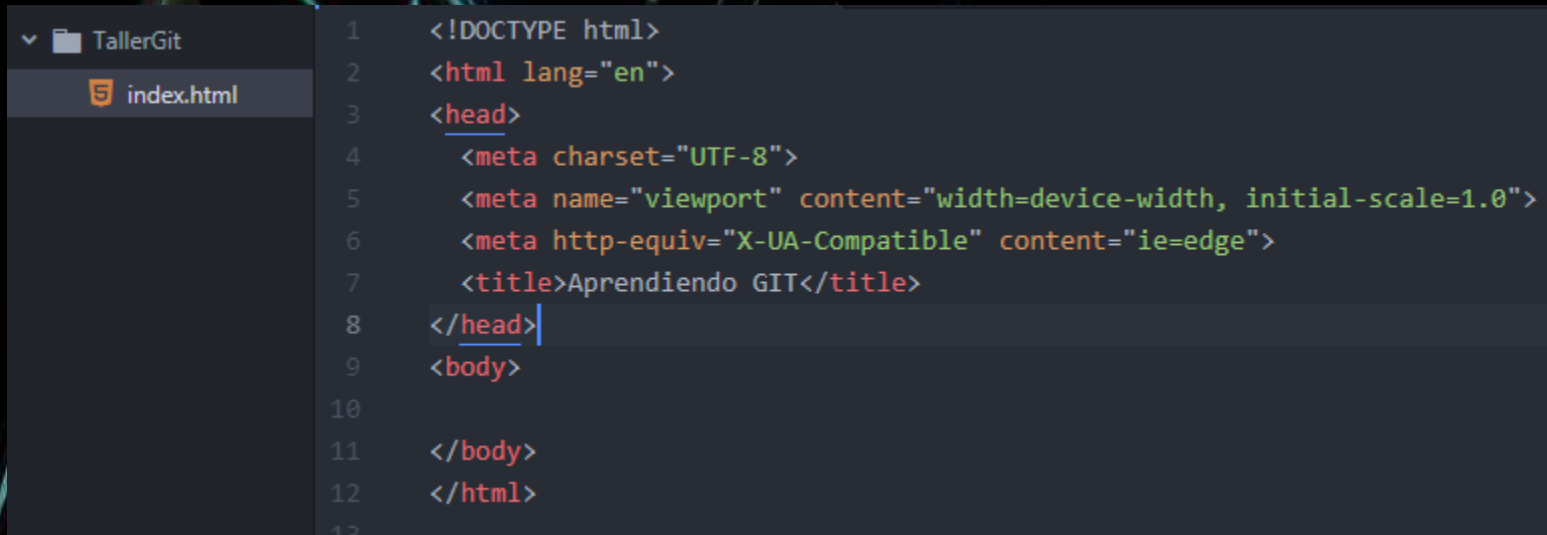
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master
nothing to commit, working tree clean

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ |
```



# EJEMPLO

El primer commit (Iniciando proyecto) capturo el momento en el que el proyecto se encontraba en este estado.

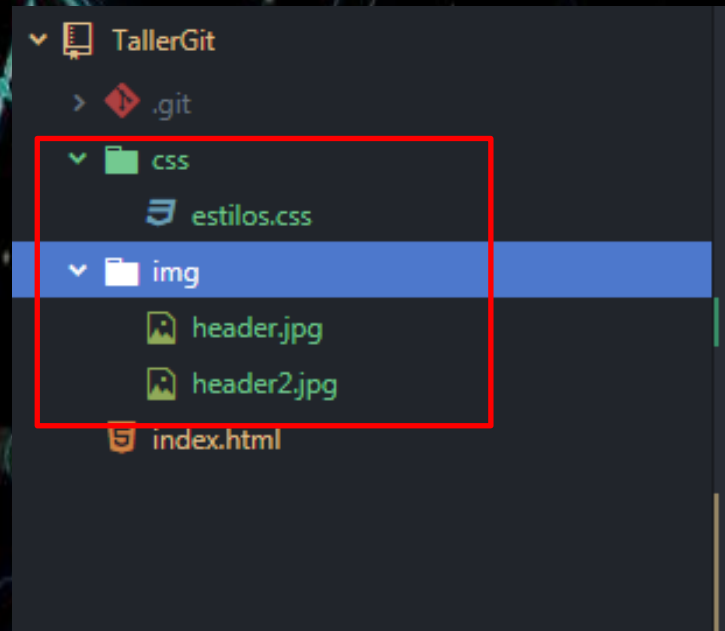


The image shows a code editor interface with a dark theme. On the left, a file explorer shows a folder named 'TallerGit' containing a file named 'index.html'. The main editor area displays the content of 'index.html' with line numbers 1 through 13. The code is a basic HTML5 boilerplate with a title 'Aprendiendo GIT'.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <meta http-equiv="X-UA-Compatible" content="ie=edge">
7    <title>Aprendiendo GIT</title>
8  </head>
9  <body>
10
11 </body>
12 </html>
13
```

# EJEMPLO

En la carpeta tallerGit se creó dos subcarpetas **css** y **img**. Dentro de css se creó un documento estilos.css y dentro de img se copiaron dos imágenes. Quedando el proyecto en este estado.



# EJEMPLO

Como se hizo algunas modificaciones y/o adiciones al proyecto, se debe realizar una captura o un versionamiento al estado actual del proyecto.

# EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git add .

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git commit -m "Añadiendo Recursos"
[master 4193110] Añadiendo Recursos
4 files changed, 1 deletion(-)
create mode 100644 css/estilos.css
create mode 100644 img/header.jpg
create mode 100644 img/header2.jpg

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master
nothing to commit, working tree clean

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$
```

La instrucción **git add .** Añade todos los archivos que se encuentren en el directorio al cual se esta haciendo un seguimiento.

# RAMAS

Una rama es una línea en el tiempo de commits. Anteriormente se trabajó con una rama llamada (master).

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)  
$
```



# RAMAS

## Creando ramas .

Para crear una rama se debe ejecutar la siguiente instrucción:

- `git branch nombreRama`

# RAMAS

## Listando ramas .

Para poder visualizar todas las ramas existentes se debe ejecutar la siguiente instrucción:

- `git branch`

# RAMAS

## Cambiando de ramas .

La rama por defecto es el master, pero para poder cambiar de rama se debe ejecutar la instrucción:

- `git checkout ramaDestino`

# RAMAS

## Listando ramas .

Para poder visualizar todas las ramas existentes se debe ejecutar la siguiente instrucción:

- `git branch`

# RAMAS

## Borrando ramas .

Para poder borrar una rama se debe ejecutar la siguiente instrucción:

- `git branch -d nombreDeRamaABorrar`



# RAMAS

## Uniendo ramas .

Para poder unir ramas primero se debe identificar en que rama se encuentra, se debe posicionar en la rama en la que se quiere unir las ramas y se debe ejecutar la siguiente instrucción:

- `git merge nombreRamaQueSeQuiereUnir`

# EJEMPLO

Se creara una nueva rama con nombre **rama header**. Y el proyecto quedara de esta manera:



# EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git branch ramaHeader

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git branch
* master
  ramaHeader

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git checkout ramaHeader
Switched to branch 'ramaHeader'

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$ git branch
* master
  ramaHeader
```

**Crear rama, listar ramas, cambiar rama.**

# EJEMPLO

Se añadió el header y el menú de navegación en el index.html.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta http-equiv="X-UA-Compatible" content="ie=edge">
7   <link rel="stylesheet" href="css/estilos.css">
8   <title>Aprendiendo GIT</title>
9 </head>
10 <body>
11   <header>
12   </header>
13   <nav class="menu">
14     <a href="#">Inicio</a>
15     <a href="#">Nosotros</a>
16     <a href="#">Blog</a>
17     <a href="#">Login</a>
18   </nav>
19 </body>
20 </html>
```

# EJEMPLO

En el archivo estilos.css se le aplican estilos al header y al menú de navegación.

```
html {
  box-sizing: border-box;
}
*, *:before, *:after {
  box-sizing: inherit;
}

/*Header*/
header{
  background-image: url(../img/header.jpg);
  height: 50vh;
  width: 100%;
  background-size: cover;
  background-repeat: no-repeat;
  background-position: center center;
  max-height: 50vh;
  position: relative;
}
```

```
/*Menu*/
.menu{
  position: absolute;
  display: flex;
  justify-content: space-around;
  top: 25px;
  background: rgba(0,0,0,.3);
  height: 60px;
  align-items: center;
  width: 99%;
}
.menu a{
  text-decoration: none;
  color: white;
  font-size: 2rem;
}
```



# EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$ git status
On branch ramaHeader
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   css/estilos.css
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$
```

Se puede ver que en la rama Header existen cambios que aun no se capturaron.

# EJEMPLO

Todo lo que se adiciono en el proyecto será capturado en la rama header.

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$ git add .

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$ git commit -m "Añadiendo Header y Nav"
[ramaHeader 32a2f37] Añadiendo Header y Nav
2 files changed, 43 insertions(+)

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$ git status
On branch ramaHeader
nothing to commit, working tree clean

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$
```

# EJEMPLO

Se unirá la rama header a la rama master. Para eso, se debe cambiar a la rama master que es la rama en donde se quieren los cambios.

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaHeader)
$ git checkout master
Switched to branch 'master'

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master
nothing to commit, working tree clean

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git branch
* master
  ramaHeader

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git merge ramaHeader
Updating 4193110..32a2f37
Fast-forward
 css/estilos.css | 34 +++++
 index.html      |  9 +++++
 2 files changed, 43 insertions(+)

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$
```

# EJEMPLO

Se realizo un cambio en el archivo estilos.css al header se le cambio de imagen y al menú se le aumento su transparencia. Git reconocerá estos cambios.

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   css/estilos.css

no changes added to commit (use "git add" and/or "git commit -a")

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ |
```

# EJEMPLO

## Capturando cambios.

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   css/estilos.css

no changes added to commit (use "git add" and/or "git commit -a")

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git commit -am "Aplicando cambios en el header"
[master 4a75c11] Aplicando cambios en el header
1 file changed, 2 insertions(+), 2 deletions(-)

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git status
On branch master
nothing to commit, working tree clean

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ |
```

La instrucción **git commit -am** “”, añade los archivos que sufrieron cambios y a la vez genera un commit.



# EJEMPLO

## **Viajando en el tiempo.**

El header inicial tenia una apariencia distinta al header actual, pero gracias a git se puede viajar en el tiempo y escoger con cualquiera de los headers.

# EJEMPLO

## Viajando en el tiempo

Para viajar en el tiempo primero se debe conocer el **dash** del commit. El dash es lo que hace único a cada commit, para poder verlo se debe listar los commits. Se debe copiar el dash y ejecutar la siguiente instrucción.

- **git checkout dashDelCommit**

# EJEMPLO

## Viajando en el tiempo

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git log
commit 4a75c11bf438fc673180738fd33044cb93080f43 (HEAD -> master)
Author: Arnaldo Muñoz Mendoza <[email address]>
Date: 2019 -0400
```

Aplicando cambios en el header

```
commit 32a2f37c0556c9ef61bd4add1f0377a63a22154c (ramaHeader)
Author: Arnaldo Muñoz Mendoza <[email address]>
Date: 2019 -0400
```

Añadiendo Header y Nav

```
commit 19311014c9db9b3a16fb28bc900bcb9d069450
Author: Arnaldo Muñoz Mendoza <[email address]>
Date: 2019 -0400
```

Añadiendo Recursos

```
commit e22a823feb9c7d39c28f42310a67979fe627f437
Author: Arnaldo Muñoz Mendoza <[email address]>
Date: 2019 -0400
```

Iniciando proyecto

# EJEMPLO

## Viajando en el tiempo

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git checkout 32a2f37c0556c9ef61bd4add1f0377a63a22154c
Note: checking out '32a2f37c0556c9ef61bd4add1f0377a63a22154c'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-b` with the checkout command again. Example:

```
git checkout -b <new-branch-name>
```

```
HEAD is now at 32a2f37 Añadiendo Header y Nav
```

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit ((32a2f37...))
$ |
```

Se copio el dashed del commit de nombre  
“Añadiendo Header y Nav”



# EJEMPLO

## Header Inicial.



## Header Actual.





# EJEMPLO

## Viajando en el tiempo

Puede notarse que el header y el menú de la página cambiaron. Con esto se volvió al header anterior. Para volver al estado de la página anterior (Volviendo al futuro XD) se debe ejecutar la siguiente instrucción:

- `git checkout master`

# EJEMPLO

## Viajando en el tiempo

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit ((32a2f37...))  
$ git checkout master  
Previous HEAD position was 32a2f37 Añadiendo Header y Nav  
Switched to branch 'master'
```

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)  
$ git branch  
* master  
  ramaHeader
```

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)  
$
```

# TAREA

Crear la rama **ramaCuerpo**, cambiar de la rama **master** a la rama **ramaCuerpo** y añadir el siguiente código al documento **index.html** (Ver siguiente imagen)

# TAREA

## RamaCuerpo (index.html)

```
<div class="cuerpo">
  <div class="contenido">
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Autem magnam offi</p>
  </div>
  <div class="aside">
    
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Explicabo soluta</p>
  </div>
</div>
```

Para generar el texto Lorem .... Solo se debe escribir Lorem y luego presionar tabulador. Las etiquetas **<p>** tienen su cierre **</p>** **que no se ven en la imagen.**

# TAREA

## RamaCuerpo (estilos.css)

```
/*Cuerpo*/  
.cuerpo{  
  position: relative;  
  display: flex;  
}  
.cuerpo.contenido{  
  position: relative;  
  width: 65%;  
  top: 0;  
}
```

```
.cuerpo .contenido p{  
  padding-left: 20px;  
  text-align: justify;  
  padding-right: 20px;  
}  
.cuerpo .aside img{  
  width: 150px;  
  height: 150px;  
  padding-top: 20px;  
}  
.cuerpo .aside p{  
  text-align: justify;  
  padding-right: 20px;  
}
```



# TAREA

Una vez realizado esas adiciones crear un commit y unir la rama **ramaCuerpo** a la rama **master**.

# EJEMPLO

Por último se creara el pie de pagina en otra rama de nombre **ramaPiePagina**. En la siguiente imagen se muestra lo que se debe adicionar en el index.html y en los estilos.css

# EJEMPLO

## RamaPiePagina (index.html)

```
<footer>  
  <p>Todos los derechos reservados &copy; ArniCode-2019</p>  
</footer>
```

## RamaPiePagina (estilos.css)

```
/*Pie de página*/  
footer{  
  background:#0270B2;  
  height: 35px;  
}  
footer p{  
  color:#fff;  
  font-size: 20px;  
  text-align: center;  
  padding-top:5px ;  
}
```

# EJEMPLO

Con los cambios efectuados se creara la rama ramaPiePagina, se capturara los cambios (commit) y se unirá la rama creada a la rama master.

# EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git branch ramaPiePagina

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git checkout ramaPiePagina
Switched to branch 'ramaPiePagina'
M      css/estilos.css
M      index.html

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaPiePagina)
$ git add .

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaPiePagina)
$ git commit -m "Añadiendo pie de pagina"
[ramaPiePagina 03cd1df] Añadiendo pie de pagina
2 files changed, 15 insertions(+)

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaPiePagina)
$ git status
On branch ramaPiePagina
nothing to commit, working tree clean

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (ramaPiePagina)
$ git checkout master
Switched to branch 'master'

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git merge ramaPiePagina
Updating 9bbdd7d..03cd1df
Fast-forward
 css/estilos.css | 12 ++++++++
 index.html      |  3 +++
 2 files changed, 15 insertions(+)
```



# TAGS

Un tag no es mas que una referencia en un commit en especifico. Para crear tags la instrucción es la siguiente:

- `git tag nombreTag`

Para crear un tag con opciones la instrucción es la siguiente:

- `git tag -a version -m “descripciónDeTag ”`

# TAGS

Se creara un tag en el último commit generado.

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)  
$ git tag -a v1.0.0 -m "Versión de prueba"
```

# GITHUB

La página oficial de github es

<https://github.com/>



# GITHUB

Es una plataforma de desarrollo colaborativo de software para alojar proyectos.

Desde enero de 2019 github permite crear repositorios privados de manera gratuita.

Para su uso se necesita estar conectado a internet.

# GITHUB

Para crear una cuenta en github se debe dirigir al menú **SIGN UP**, esto hará que se despliegue una nueva ventana en el que se deben llenar los datos correspondientes, una vez llenado los datos y haber creado la cuenta se debe verificar en el correo electrónico la activación de la cuenta Github.



# GITHUB



Why GitHub? ▾ Enterprise Explore ▾ Marketplace Pricing ▾

Search GitHub



Sign in

Sign up

## Built for developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside 36 million developers.

Username

Email

Password

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account related emails.

# GITHUB

## Join GitHub

The best way to design, build, and ship software.



Step 1:  
Set up your account



Step 2:  
Choose your subscription



Step 3:  
Tailor your experience

### Create your personal account

Username \*

This will be your username. You can add the name of your organization later.

Email address \*

We'll occasionally send updates about your account to this inbox. We'll never share your email address with anyone.

Password \*

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more](#).

### You'll love GitHub

Unlimited public repositories  
Unlimited private repositories

- ✓ Limitless collaboration
- ✓ Frictionless development
- ✓ Open source community

### Verify account



# GITHUB

## Welcome to GitHub

You've taken your first step into a larger world,



**Completed**

Set up a personal account



**Step 2:**

Choose your plan



**Step 3:**

Tailor your experience

### Choose your personal plan



Unlimited public repositories for free.



Unlimited private repositories for \$7/month. ([view in HNL](#))

Don't worry, you can cancel or upgrade at any time.

#### ☐ Help me set up an organization next

Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.

[Learn more about organizations.](#)

**Continue**

### Both plans include:

- ✓ Collaborative code review
- ✓ Issue tracking
- ✓ Open source community
- ✓ Unlimited public repositories
- ✓ Join any organization

# GITHUB

## Welcome to GitHub

You'll find endless opportunities to learn, code, and create,



**Completed**

Set up a personal account



**Step 2:**

Choose your plan



**Step 3:**

Tailor your experience

How would you describe your level of programming experience?

- ☐ Very experienced      ☐ Somewhat experienced      ☐ Totally new to programming

What do you plan to use GitHub for? (check all that apply)

- ☐ Development      ☐ School projects      ☐ Design  
☐ Research      ☐ Project Management      ☐ Other (please specify)

Which is closest to how you would describe yourself?

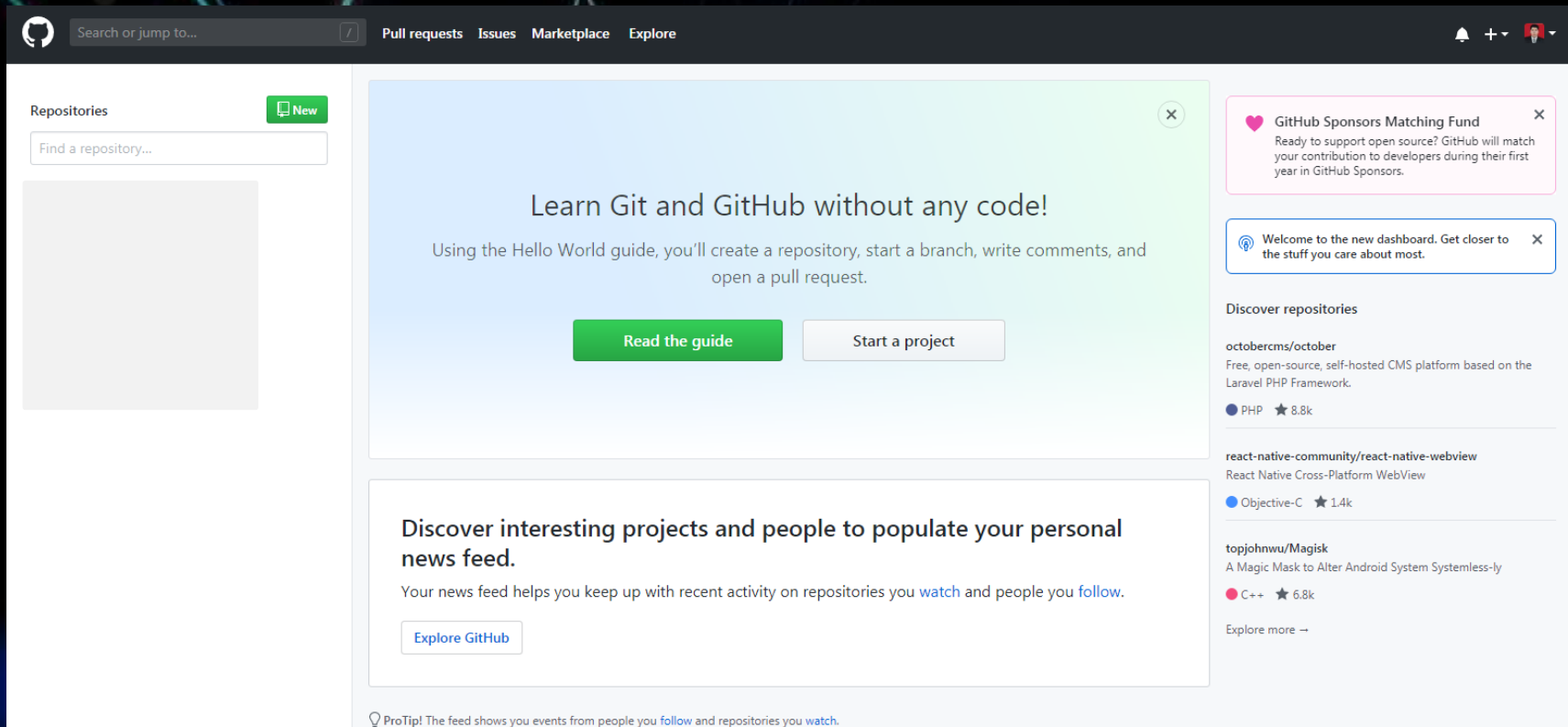
- ☐ I'm a hobbyist      ☐ I'm a student      ☐ I'm a professional  
☐ Other (please specify)

# GITHUB

Una vez ya creado la cuenta de github se comenzara a interactuar con esta plataforma, se creará un repositorio y se subirá la página de prueba creada al repositorio.



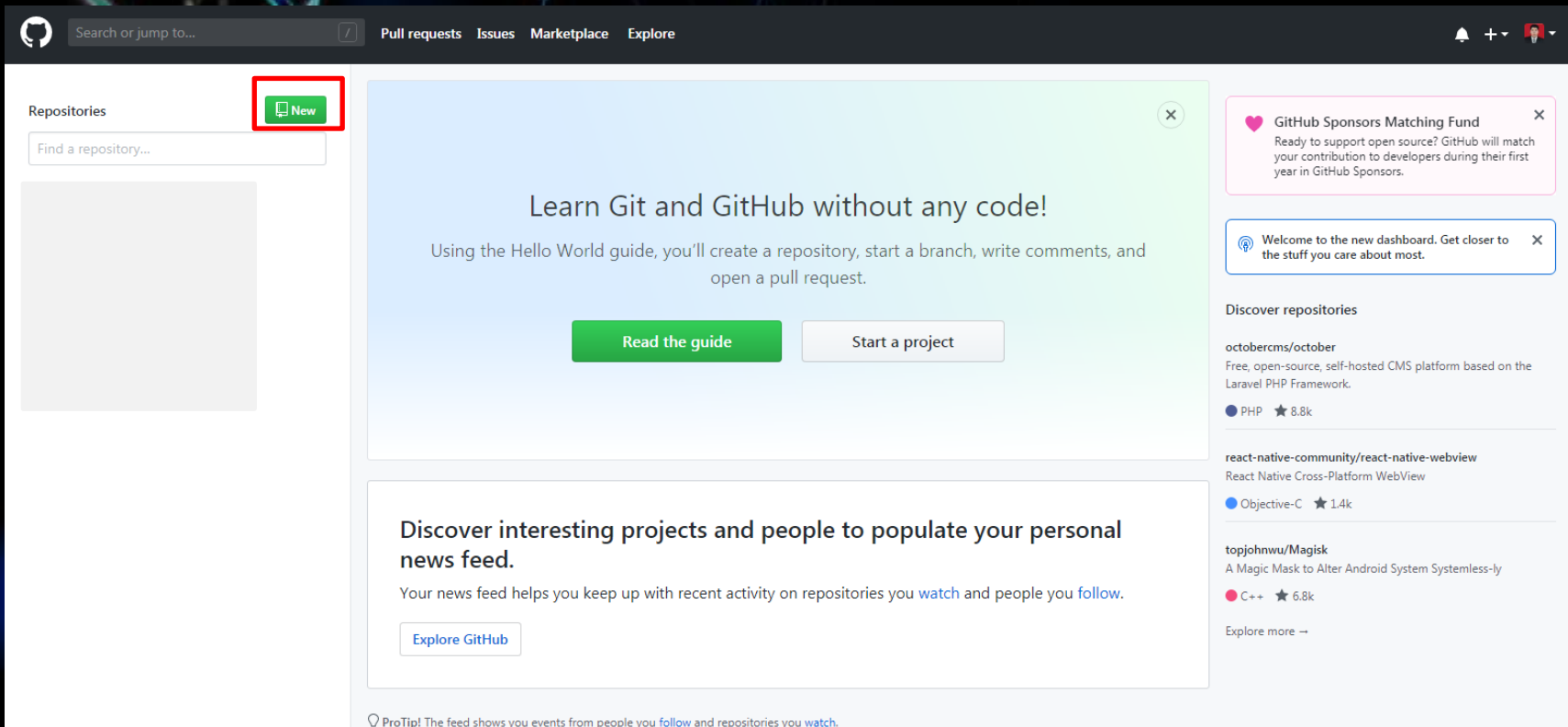
# GITHUB



Ventana principal de la cuenta de Github.

# GITHUB

## Creando un repositorio



The screenshot shows the GitHub homepage. In the top-left sidebar, under the 'Repositories' section, there is a green button with a plus icon and the word 'New'. This button is highlighted with a red rectangular box. The main content area features a large light blue box with the text 'Learn Git and GitHub without any code!' and two buttons: 'Read the guide' (green) and 'Start a project' (light blue). Below this is a section titled 'Discover interesting projects and people to populate your personal news feed.' with an 'Explore GitHub' button. The right sidebar contains several notifications and a list of repositories under the heading 'Discover repositories'.

Search or jump to... / Pull requests Issues Marketplace Explore

Repositories **New**

Find a repository...

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

[Read the guide](#) [Start a project](#)

Discover interesting projects and people to populate your personal news feed.

Your news feed helps you keep up with recent activity on repositories you [watch](#) and people you [follow](#).

[Explore GitHub](#)

ProTip! The feed shows you events from people you [follow](#) and repositories you [watch](#).

GitHub Sponsors Matching Fund

Ready to support open source? GitHub will match your contribution to developers during their first year in GitHub Sponsors.

Welcome to the new dashboard. Get closer to the stuff you care about most.

Discover repositories

octobercms/october

Free, open-source, self-hosted CMS platform based on the Laravel PHP Framework.

PHP ★ 8.8k

react-native-community/react-native-webview

React Native Cross-Platform WebView

Objective-C ★ 1.4k

topjohnwu/Magisk

A Magic Mask to Alter Android System Systemless-ly

C++ ★ 6.8k

[Explore more](#) →


# GITHUB

## Creando un repositorio

### Create a new repository

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



Owner

 ArmyWorld ▾

Repository name \*

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


Description (optional)

- ☒  **Public**  
Anyone can see this repository. You choose who can commit.
- ☐  **Private**  
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

- ☐ **Initialize this repository with a README**  
This will let you immediately clone the repository to your computer.

Add .gitignore: **None** ▾


Add a license: **None** ▾ 

Create repository

# GITHUB

## Creando un repositorio

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

 Set up in Desktop or ☐ HTTPS ☒ SSH `https://github.com/ArnyWorld/tallerGit.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 "# tallerGit" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/ArnyWorld/tallerGit.git
git push -u origin master
```


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

```
git remote add origin https://github.com/ArnyWorld/tallerGit.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](#)

 **ProTip!** Use the URL for this page when adding GitHub as a remote.

# GITHUB

## Git Remote

Para subir el repositorio local al repositorio remoto Github se deben ejecutar algunas instrucciones:

- `git remote add origin https://.....`

Esta instrucción establecerá una conexión entre nuestro repositorio local y el repositorio remoto que se encontrara en `https://.....`



# GITHUB

## Git Push

La instrucción `git push` permitirá subir el repositorio local al repositorio remoto, la instrucción es:

- `git push -u origin master`

# GITHUB EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git remote add origin https://github.com/ArnyWorld/tallerGit.git
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git push -u origin master
```

El enlace marcado en recuadro rojo es el enlace del repositorio remoto. Este enlace se encuentra detallado en la [diapositiva 83](#).

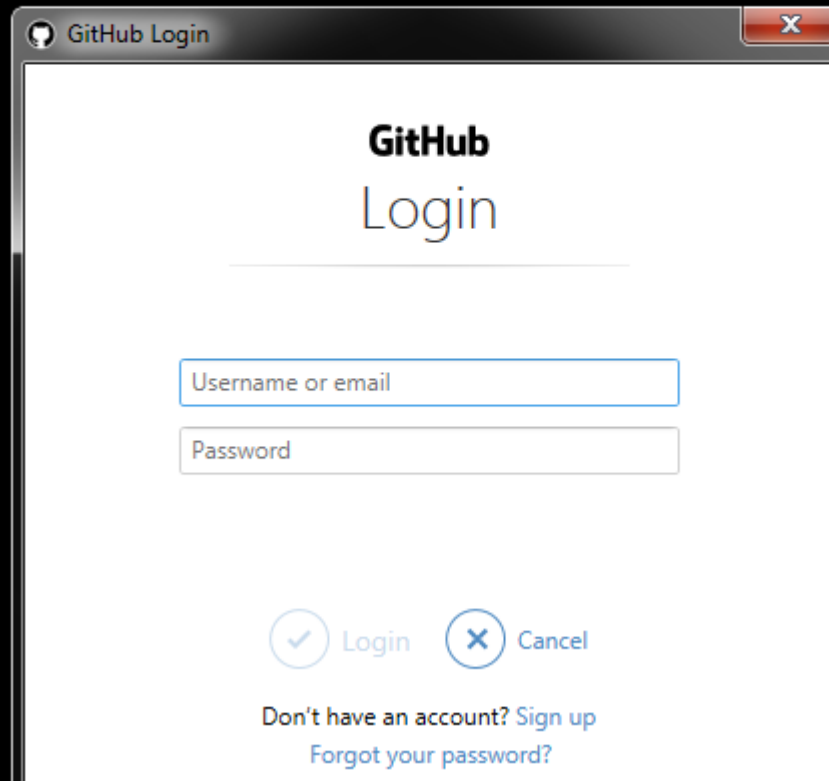
# GITHUB EJEMPLO

Al ejecutar la instrucción **git push -u origin master**, se desplegara una ventana en la cual se debe registrar con el nombre de usuario y contraseña de github.

# GITHUB EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git remote add origin https://github.com/ArnyWorld/tallerGit.git

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git push -u origin master
```



The image shows a screenshot of a Windows-style dialog box titled "GitHub Login". The dialog has a white background and a dark gray title bar with a close button (X) in the top right corner. The main content area is white and contains the GitHub logo (an octocat) and the text "Login" in a large, bold, sans-serif font. Below the text, there are two input fields: "Username or email" and "Password". At the bottom of the dialog, there are two buttons: "Login" (with a checkmark icon) and "Cancel" (with an X icon). Below these buttons, there are two links: "Don't have an account? Sign up" and "Forgot your password?".

GitHub Login

GitHub  
Login

Username or email

Password

✓ Login ✕ Cancel

Don't have an account? [Sign up](#)  
[Forgot your password?](#)

# GITHUB EJEMPLO

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git remote add origin https://github.com/ArnyWorld/tallerGit.git

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git push -u origin master
Enumerating objects: 32, done.
Counting objects: 100% (32/32), done.
Delta compression using up to 8 threads
Compressing objects: 100% (25/25), done.
Writing objects: 100% (32/32), 350.43 KiB | 10.95 MiB/s, done.
Total 32 (delta 7), reused 0 (delta 0)
remote: Resolving deltas: 100% (7/7), done.
To https://github.com/ArnyWorld/tallerGit.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ .....
```

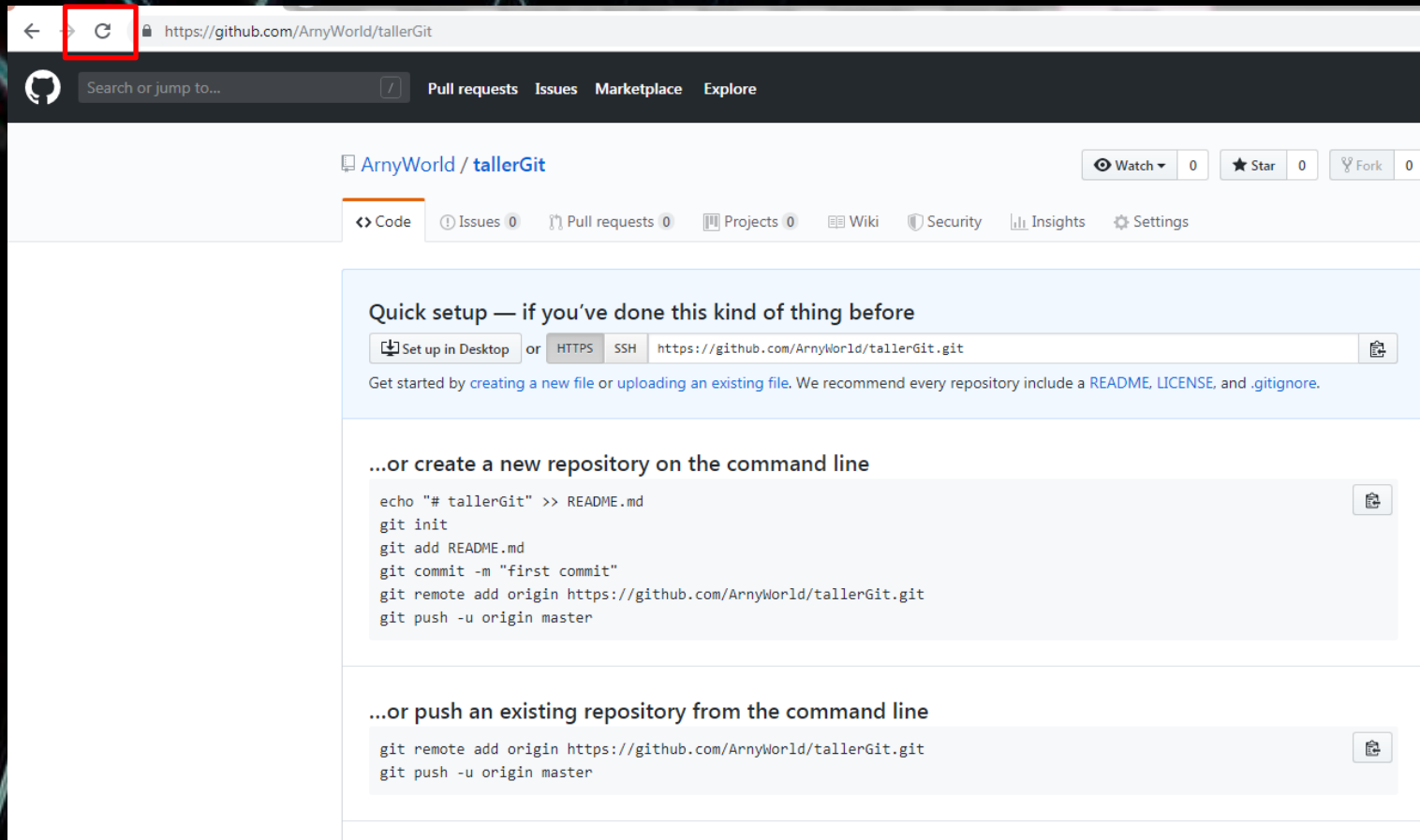
Después de *loguearse* se subirá el repositorio local al repositorio remoto.



# GITHUB EJEMPLO

Luego de ejecutar las instrucciones mencionadas anteriormente, se debe dirigir a su repositorio remoto y actualizar la página.

# GITHUB EJEMPLO



Repositorio remoto sin actualizar.

# GITHUB EJEMPLO

The screenshot shows a GitHub repository page for 'ArmyWorld / tallerGit'. At the top, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork' (0). Below these are tabs for 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Security', 'Insights', and 'Settings'. The main content area shows a message 'No description, website, or topics provided.' with an 'Edit' button. Below this is a 'Manage topics' link. A progress bar shows '6 commits', '1 branch', '0 releases', and '1 contributor'. Below the progress bar are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find File', and 'Clone or download'. A list of files is shown: 'css' (Añadiendo pie de pagina, 10 hours ago), 'img' (Añadiendo el cuerpo de la página, 11 hours ago), and 'index.html' (Añadiendo pie de pagina, 10 hours ago). At the bottom, there is a prompt to 'Add a README'.

ArmyWorld / tallerGit

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. Edit

Manage topics

6 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find File Clone or download

ArmyWorld Añadiendo pie de pagina Latest commit 03cd1df 10 hours ago

css	Añadiendo pie de pagina	10 hours ago
img	Añadiendo el cuerpo de la página	11 hours ago
index.html	Añadiendo pie de pagina	10 hours ago

Help people interested in this repository understand your project by adding a README. Add a README

© 2019 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

Repositorio remoto actualizado.

# GITHUB EJEMPLO

## Verificando Commits

ArmyWorld / tallerGit

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. [Edit](#)

[Manage topics](#)


6 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find File Clone or download

ArmyWorld Añadiendo pie de pagina Latest commit 03cd1df 10 hours ago

css	Añadiendo pie de pagina	10 hours ago
img	Añadiendo el cuerpo de la página	11 hours ago
index.html	Añadiendo pie de pagina	10 hours ago

Help people interested in this repository understand your project by adding a README. [Add a README](#)

© 2019 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#)  [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

# GITHUB EJEMPLO

## Verificando Commits

The screenshot displays the GitHub interface for the repository 'ArmyWorld / tallerGit'. At the top, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork' (0). Below these are navigation tabs: 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Security', 'Insights', and 'Settings'. A dropdown menu shows the current branch as 'master'. The commit history is organized by date, with two sections: 'Commits on Jul 15, 2019' and 'Commits on Jul 14, 2019'. Each commit entry includes a description, the user 'ArmyWorld', the time since commit, a file icon, a commit hash, and a code comparison link.

Commit Date	Commit Message	User	Time Ago	File Icon	Commit Hash	Code Comparison
Commits on Jul 15, 2019	Añadiendo pie de pagina	ArmyWorld	11 hours ago		03cd1df	<a href="#">Code</a>
	Añadiendo el cuerpo de la página	ArmyWorld	11 hours ago		9bbd7d	<a href="#">Code</a>
	Aplicando cambios en el header	ArmyWorld	12 hours ago		4a75c11	<a href="#">Code</a>
Commits on Jul 14, 2019	Añadiendo Header y Nav	ArmyWorld	12 hours ago		32a2f37	<a href="#">Code</a>
	Añadiendo Recursos	ArmyWorld	13 hours ago		4193110	<a href="#">Code</a>
	Iniciando proyecto	ArmyWorld	14 hours ago		e22a823	<a href="#">Code</a>



# GITHUB EJEMPLO

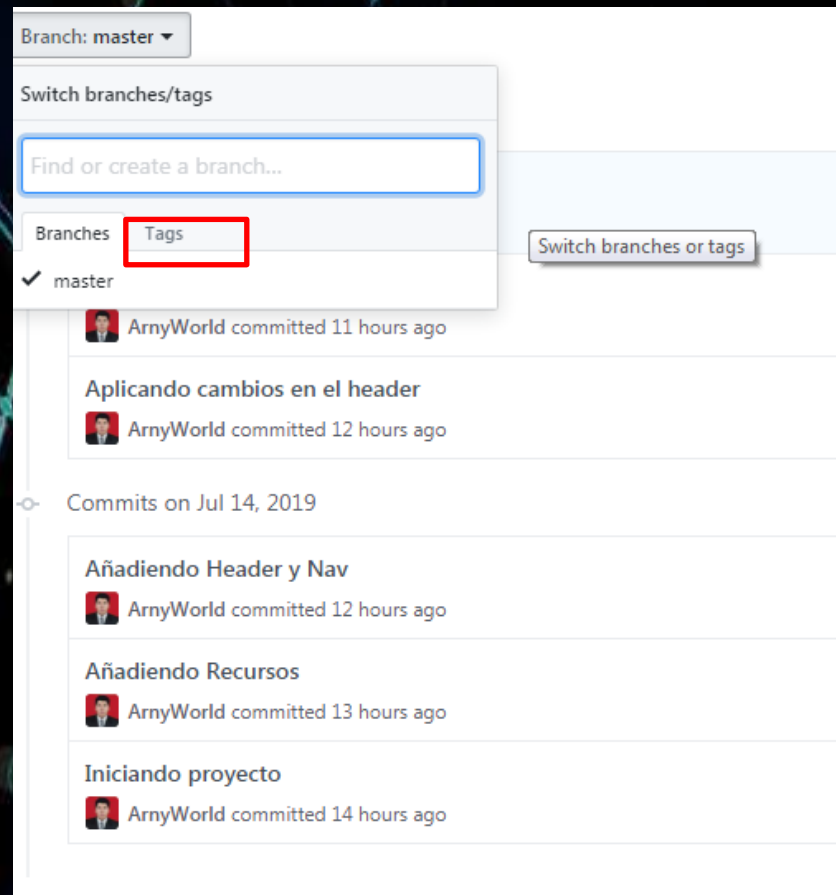
## Verificando Tags

The screenshot shows the GitHub interface for the repository 'ArnyWorld / tallerGit'. At the top, there are buttons for 'Watch', 'Star', and 'Fork', each with a count of 0. Below these are navigation tabs: 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. A dropdown menu shows 'Branch: master'. The commit history is displayed in two sections: 'Commits on Jul 15, 2019' and 'Commits on Jul 14, 2019'. Each commit entry includes a description, the user 'ArnyWorld', the time since commit, a file icon, a commit hash, and a code icon.

Commit Date	Commit Message	User	Time	Hash
Jul 15, 2019	Añadiendo pie de pagina	ArnyWorld	11 hours ago	03cd1df
	Añadiendo el cuerpo de la página	ArnyWorld	11 hours ago	9bdd7d
	Aplicando cambios en el header	ArnyWorld	12 hours ago	4a75c11
Jul 14, 2019	Añadiendo Header y Nav	ArnyWorld	12 hours ago	32a2f37
	Añadiendo Recursos	ArnyWorld	13 hours ago	4193110
	Iniciando proyecto	ArnyWorld	14 hours ago	e22a823

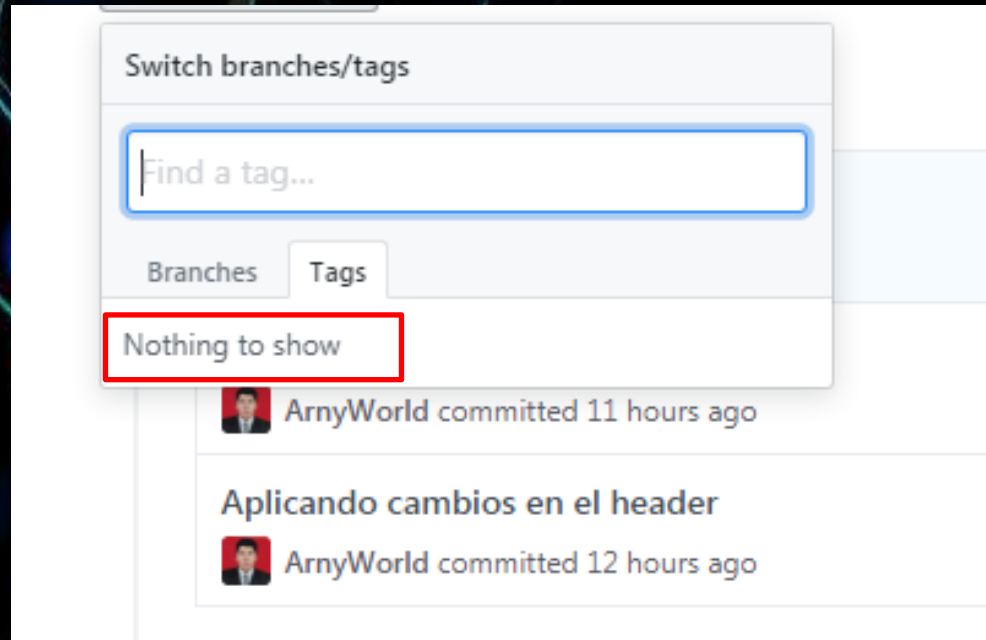
# GITHUB EJEMPLO

## Verificando Tags



# GITHUB EJEMPLO

## Verificando Tags



Se puede notar que no se encuentra el tag que se creó en el repositorio local.

# GITHUB EJEMPLO

## Subiendo Tags

La instrucción para subir los tags creados al repositorio remoto es:

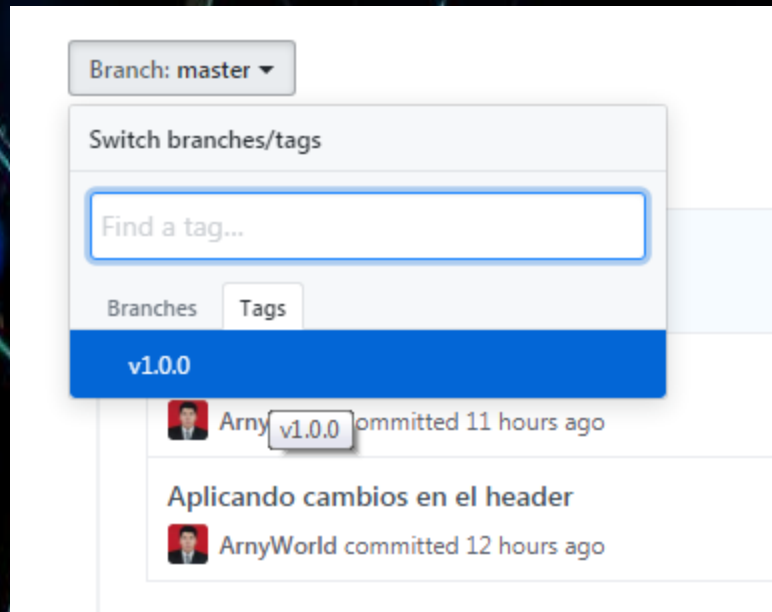
- `git push --tags`

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
$ git push --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 187 bytes | 187.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0)
To https://github.com/ArnyWorld/tallerGit.git
 * [new tag]          v1.0.0 -> v1.0.0
```

```
Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/TallerGit (master)
```

# GITHUB EJEMPLO

## Verificando Tags



Se puede notar que ya se encuentra el tag en el repositorio remoto.



# GITHUB EJEMPLO

## Clonando repositorios

The screenshot shows a GitHub repository page for 'ArmyWorld'. At the top, it displays '6 commits', '1 branch', '1 release', and '1 contributor'. Below this, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find File', and a highlighted 'Clone or download' button. A dropdown menu is open from the 'Clone or download' button, showing 'Clone with HTTPS' (highlighted with a yellow box) and 'Use SSH'. Below this, it says 'Use Git or checkout with SVN using the web URL.' and shows the URL 'https://github.com/ArmyWorld/tallerGit.g' (highlighted with a yellow box). There are also buttons for 'Open in Desktop' and 'Download ZIP' (highlighted with a blue box). At the bottom, there is a section for adding a README.

6 commits   1 branch   1 release   1 contributor

Branch: master   New pull request   Create new file   Upload files   Find File   **Clone or download**

**Clone with HTTPS**   Use SSH

Use Git or checkout with SVN using the web URL.

`https://github.com/ArmyWorld/tallerGit.g`

Open in Desktop   **Download ZIP**

Help people interested in this repository understand your project by adding a README.   Add a README

Dirección del repositorio a clonar.

Descargar repositorio .ZIP.

# GITHUB EJEMPLO

## Clonando repositorios

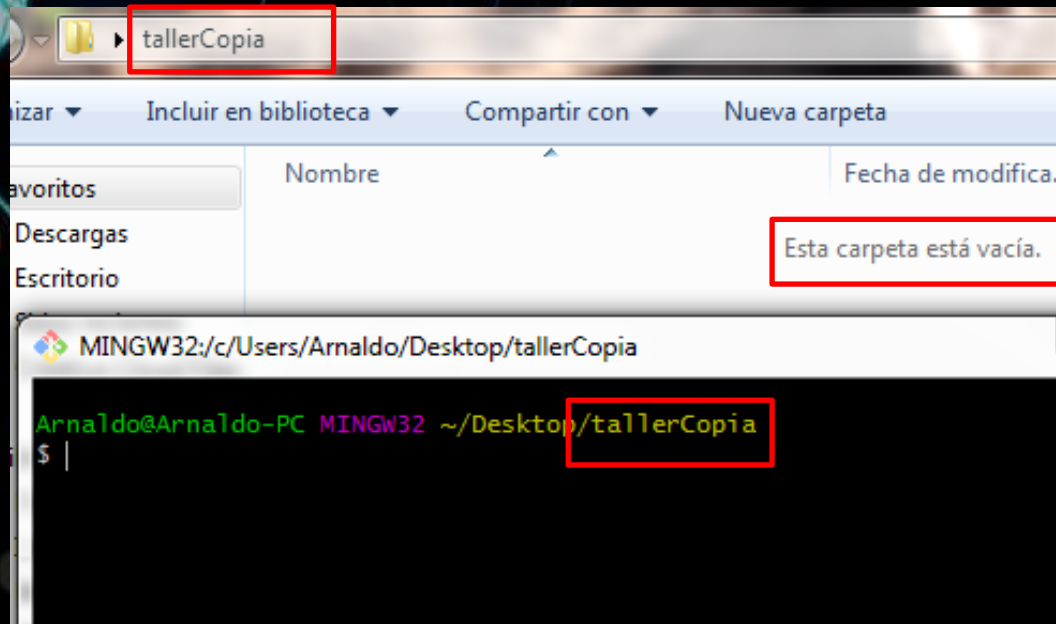
La instrucción para clonar un repositorio remoto es:

- `git clone dirección NuevoNombreRepositorio`

Para mostrar el funcionamiento de esta instrucción se creará una nueva carpeta con nombre tallerCopia, dentro de esta carpeta se encontrara el repositorio clonado.

# GITHUB EJEMPLO

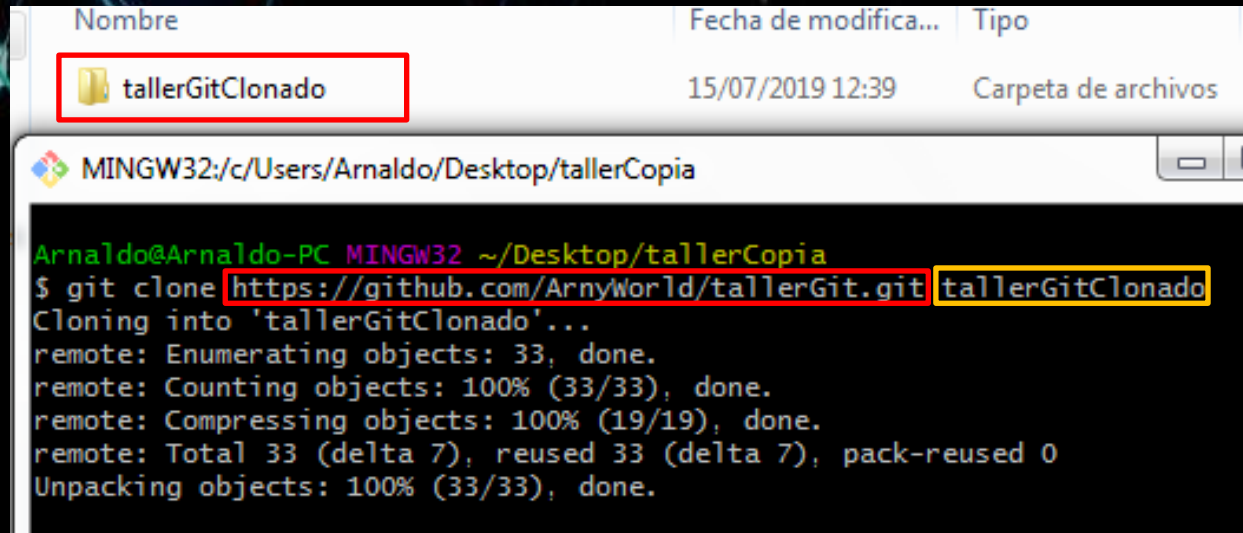
## Clonando repositorios



Carpeta tallerCopia vacia

# GITHUB EJEMPLO

## Clonando repositorios



The image shows a Windows File Explorer window and a terminal window. The File Explorer window displays a folder named 'tallerGitClonado' with a modification date of 15/07/2019 12:39 and a type of 'Carpeta de archivos'. The terminal window shows the command 'git clone https://github.com/ArnyWorld/tallerGit.git tallerGitClonado' being executed. The output of the command is displayed below the command line.

```
Nombre | Fecha de modifica... | Tipo
tallerGitClonado | 15/07/2019 12:39 | Carpeta de archivos

MINGW32:/c/Users/Arnaldo/Desktop/tallerCopia

Arnaldo@Arnaldo-PC MINGW32 ~/Desktop/tallerCopia
$ git clone https://github.com/ArnyWorld/tallerGit.git tallerGitClonado
Cloning into 'tallerGitClonado'...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 33 (delta 7), reused 33 (delta 7), pack-reused 0
Unpacking objects: 100% (33/33), done.
```

Dirección del repositorio a clonar.

Nuevo nombre del repositorio clonado

# **GIT - GITHUB**

## **Aclaración**

Todas las instrucciones aprendidas en este taller no son las únicas, existen mas instrucciones tanto en Git como en GitHub.

Para conocer mas sobre Git puede visitar su página oficial y curiosar un poco sobre todas sus instrucciones <https://git-scm.com/docs>



# **GIT - GITHUB**

## **Despedida**

Espero les sea útil esta pequeña guía de Git y Github agradecerles por su atención e interes en aprender nuevas tecnologías y si quieren saber mas sobre este u otros temas relacionados no olviden seguir la página de Facebook

<https://www.facebook.com/WeDevsCommunity/>

nuestro grupo de whatsapp

<https://chat.whatsapp.com/D1tyerX4c3mH2KoccV7nGy>



# GRACIAS

