

Guía de ejercicios 2



¡Hola! Te damos la bienvenida a esta nueva guía de estudio.

¿En qué consiste esta guía?

La siguiente guía de estudio tiene como objetivo practicar y ejercitar los contenidos que hemos visto en clase.

¡Vamos con todo!



Tabla de contenidos

Guía de ejercicios 2	1
¿En qué consiste esta guía?	1
Tabla de contenidos	1
Documentación de código con Markdown	1
Encabezados en Markdown	2
Lista de elementos en Markdown	2
Formato de código	2
Flujo de trabajo con Github	3
Actividad guiada N° 1: Utilizando Github	4
¡Manos a la obra! - Repositorio	7



¡Comencemos!

Documentación de código con Markdown

Documentar el código es una práctica fundamental para el desarrollo de software, ya que ofrece numerosos beneficios tanto para los desarrolladores individuales como para los equipos de trabajo.

- Para documentar el código en Github podemos hacer uso del archivo `readme.md`. Este archivo trabaja con una extensión llamada Markdown por sus siglas **md**.
- Al documentar con Markdown debemos tomar consideración los siguientes elementos:
 - Título claro y descriptivo
 - Descripción breve del código
 - Notas importantes o consideraciones
 - Enlaces a recursos adicionales
 - Código formateado

Encabezados en Markdown

Para generar títulos en Markdown debemos seguir la siguiente sintaxis

```
# Título 1
## Título 2
### Título 3
#### Título 4
##### Título 5
```

- El signo de gato o numeral # debe agregarse antes del texto definido como título.
- El tamaño de cada título irá variando según la numeración, donde 1 es el más grande y 5 el más pequeño.

Lista de elementos en Markdown

Para crear lista de elementos podemos utilizar la sintaxis de - para listas referenciadas con viñetas y con números para listas referenciadas de manera numérica.

```
- Elemento de viñeta o lista desordenada
1. Elemento de lista ordenada numéricamente
```

Formato de código

Para dar formato de código en Markdown debemos triples comillas simples que envuelven un bloque de código.

```
```  
Aquí va el bloque de código
```
```

- En las triples comillas simples de apertura podemos indicar el lenguaje por el cual debe ser formateado dicho código.
- Te compartimos un ejemplo de un bloque que debe ser formateado como HTML.

```
```html  
<p>Hola mundo</p>
```
```

Con Markdown es posible formatear texto con muchas opciones, incluso para el formato de código, para ello puedes visitar la siguiente [documentación](#) donde se muestran más alternativas.

Con estos conocimientos básicos de Markdown, te invitamos a acceder a algún repositorio que tengas en GitHub y accedas al archivo README.md e implementes la sintaxis vista hasta el momento. Intenta documentar mediante los elementos y consideraciones mencionadas anteriormente.

Flujo de trabajo con Github

1. Conectarse a un repositorio.
2. Crear una rama nueva distinta a la rama principal para no afectarla. Esto permite tener un espacio de trabajo seguro que no afecte al código principal.
3. Realizar los cambios, en este paso del flujo debemos ejecutar todas aquellas acciones que se identifiquen como mejoras al código, corrección de errores o eliminación de funcionalidades. Estos cambios deben realizarse en la rama creada distinta a la principal. Una vez realizados los cambios podemos hacer commit.
4. Generar un pull request y que los colaboradores puedan entregar feedback del código subido en la rama nueva. Este feedback te permitirá volver atrás y corregir alguna implementación.
5. Dirigir la revisión de comentarios, los revisores del repositorio dejarán preguntas, comentarios y sugerencias. Los revisores pueden comentar todo el pull request o agregar comentarios a líneas o archivos específicos.

6. Fusionar la rama creada con la principal, una vez que los cambios hayan finalizado y los comentarios hayan sido abordados, es posible fusionar la rama creada con la principal.



Actividad guiada N° 1: Utilizando Github

Como vimos en el capítulo anterior, Git trabaja a modo de repositorio local y al finalizar cada cambio, podremos guardar los avances en el repositorio remoto de Github.

Para esto, ejercitaremos creando un repositorio remoto para el proyecto, lo añadiremos por consola y subiremos sus cambios. Sigue atentamente el paso a paso que te presentamos a continuación:

- **Paso 1:** una vez que ya tengas configurada la cuenta en GitHub, podrás generar un nuevo repositorio. Para ello, dentro de tu perfil en GitHub presiona el botón **+** y selecciona **new repository**:

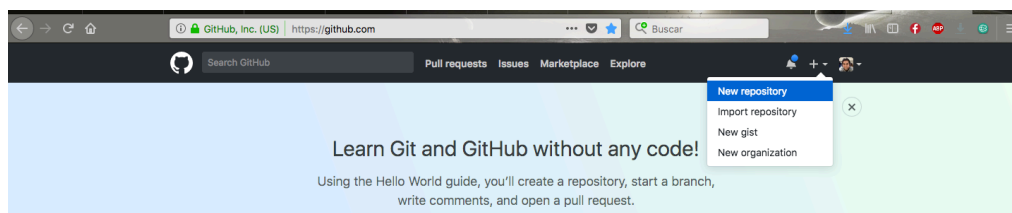


Imagen 1. Añadir nuevo repositorio.

Fuente: Desafío Latam.

Debes indicarle un nombre y una descripción:

Create a new repository

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

Owner * [alegonzalezcelis](#) Repository name *

Great repository names are short Your new repository will be created as Meet-Coffee. musical-octo-spork?

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

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

☐ Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

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

[Create repository](#)

Imagen 2. Crear el nuevo repositorio.

Fuente: Desafío Latam.

- **Paso 2:** luego de crearlo, te entregará algunos ejemplos de comandos para subir el código.

alegonzalezcelis / Meet-Coffee

Unwatch 1 Star 0 Fork

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

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

[Set up in Desktop](#) or [HTTPS](#) [SSH](#) <https://github.com/alegonzalezcelis/Meet-Coffee.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 "# Meet-Coffee" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/alegonzalezcelis/Meet-Coffee.git
git push -u origin main
```

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

```
git remote add origin https://github.com/alegonzalezcelis/Meet-Coffee.git
git branch -M main
git push -u origin main
```

...or import code from another repository

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

[Import code](#)

Imagen 3. Repositorio creado.

Fuente: Desafío Latam.

Si ya tienes un proyecto con git iniciado como es nuestro caso, solo debemos indicarle el nuevo repositorio remoto y subir los cambios.

- **Paso 3:** abre la terminal y dirígete a la carpeta del proyecto:

```
cd Desktop/meet\&coffee/
```



Nota: GitHub en sus inicios definió la rama `master` como su rama principal o por defecto, sin embargo, desde octubre de 2020, han implementado cambios en su rama principal, sustituyendo gradualmente la rama `master` por la nueva rama `main`, este cambio afecta directamente a los nuevos repositorios. Según se puede apreciar en su documentación oficial, el cambio responde a la implementación de un nombre más corto y a la interpretación más óptima en otros lenguajes. Para más información acerca de esta actualización visita el siguiente link: <https://github.com/github/renaming>

Es por eso que desde ahora comenzaremos a trabajar en nuestra rama `main`:

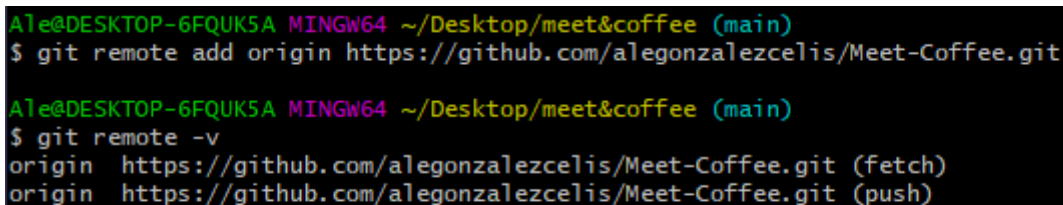
- **Paso 4:** renombra la rama `master` por `main` con el siguiente comando:

```
git branch -M main
```

- **Paso 5:** añade el repositorio remoto recién creado al proyecto desde la terminal. Reemplaza dirección con la URL o git del repositorio recién creado.

```
git remote add origin [dirección del repositorio remoto]
```

Al presionar enter, no recibirás ninguna confirmación, pero si quieres ver si está ingresado en el repositorio remoto, puedes usar el comando : `git remote -v` y verás algo como esto:



```
Ale@DESKTOP-6FQUK5A MINGW64 ~/Desktop/meet&coffee (main)
$ git remote add origin https://github.com/alegonzalezcelis/Meet-Coffee.git

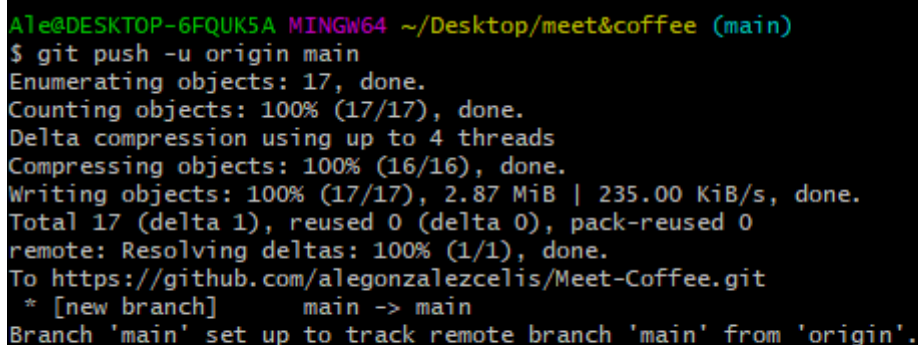
Ale@DESKTOP-6FQUK5A MINGW64 ~/Desktop/meet&coffee (main)
$ git remote -v
origin https://github.com/alegonzalezcelis/Meet-Coffee.git (fetch)
origin https://github.com/alegonzalezcelis/Meet-Coffee.git (push)
```

Imagen 4. Uso de git remote
Fuente: Desafío Latam

- **Paso 6:** ahora puedes subir el proyecto repositorio remoto:

```
git push -u origin main
```

De esta forma todo lo que tienes en local, será subido al repositorio recién creado y ya estás compartiendo tu código con el mundo.



```
Ale@DESKTOP-6FQUK5A MINGW64 ~/Desktop/meet&coffee (main)
$ git push -u origin main
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 4 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (17/17), 2.87 MiB | 235.00 KiB/s, done.
Total 17 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/alegonzalezcelis/Meet-Coffee.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
```

Imagen 5. Uso de Git Push.
Fuente: Desafío Latam.



¡Manos a la obra! - Repositorio

Github, desde agosto de 2020 nos permite crear un perfil personalizado a través de un archivo `readme.md`, en un repositorio con tu nombre de usuario, y es lo que haremos en el siguiente ejercicio.

Mira el siguiente [video](#), donde se explica cómo debes crear el repositorio y el archivo. Si tienes dudas, puedes consultar la solución en material complementario.