wawiwa
React
2022 © Wawiwa Tech | Confidential

wawiwa
23 - Git
2022 © Wawiwa Tech | Confidential

wawiwa

¿Qué es Git?

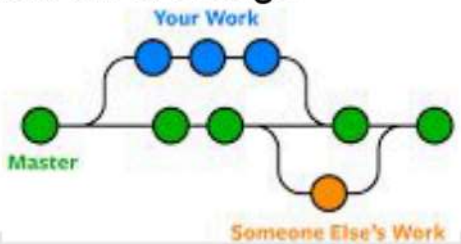
Git es un sistema de control de versión popular.

El Sistema de Control de Versión (VCS) es un software que allina a los programadores a trabajar juntos y mantener una historia completa de su trabajo.

Las funciones de un VCS −

- Permite a los programadores trabajar simultáneamente
- No permite sobrescribir los cambios de cada uno
- Mantiene una historia de cada versión - sigue la pista de cada cambio en el código

Git y GitHub son cosas distintas.



wawiwa

¿Qué es GitHub?

GitHub es una empresa con fines de lucro que ofrece un servicio de alojamiento de repositorios Git basado en la nube.

Esencialmente, facilita el control de versiones y colaboración para equipos e individuales.

La interfaz de GitHub es tan fácil de usar que incluso los programadores novatos pueden aprovechar Git.

Sin GitHub, usar Git requiere un poco mas de conocimiento técnico y uso de línea de comando.

wawiwa

¿Por qué Git?

- Más del 70% de los programadores usan Git!
- Los programadores puede trabajar juntos de cualquier parte del mundo.
- Los programadores pueden ver la historia completa del proyecto.
- Los programadores pueden revertir versiones anteriores de un proyecto.

wawiwa

Empezar con Git

Primero, debes instalar Git siguiendo este sitio:

https://git-scm.com/

wawiwa

Empezar con Git

After installing Git, the first thing to do, is to check if Git is properly installed.

Open CMD or Git Bash which comes included in Git for windows.

Run: **git --version** or **git --v**

wawiwa

Configurar Git

Ahora necesitamos dar a conocer a Git quienes somos. Pasamos nuestro usuario y dirección de email que queremos usar cuando nos registremos en GitHub después:

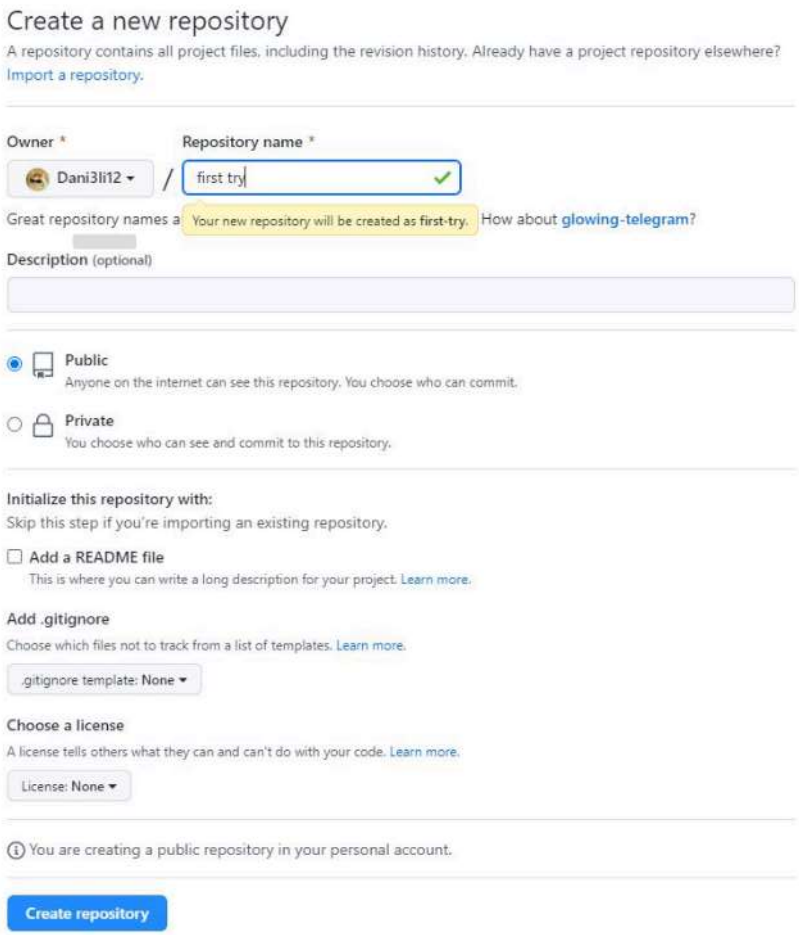
```
git config --global user.name "your-username"

git config --global user.email "email@example.com"
```

wawiwa

Trabajar con GitHub

1. Ve a https://github.com/ y regístrate.
2. Crea un nuevo repositorio - un **repositorio Git sigue la pista y guarda la historia de todos los cambios hechos a los archivos de un proyecto Git**.



wawiwa

Primer Repositorio Git en Github

1. Para inicializar git, corre en la carpeta actual: **git init**

Comando Git add:

El comando “git add” agrega todos los cambios en el directorio en el que se está trabajando al “staging area”. Le dice a Git que quieres incluir todos estas actualizaciones a un archivo particular en el siguiente “commit”.

Puedes escribir el nombre de un archivo específico o decidir agregar todos los cambios.

2. Corre **git add .**

```
PS C:\Users\danie\Documents\my-app> git add .
warning: in the working copy of 'package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'public/index.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/App.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/index.css', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/index.js', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/reportWebVitals.js', LF will be replaced by CRLF the next time Git touches it
```

wawiwa

Primer Repositorio Git en Github

Git commit:

El comando “commit” se usa para guardar tus cambios en el repositorio local con una nota.

3. Corre el comando:

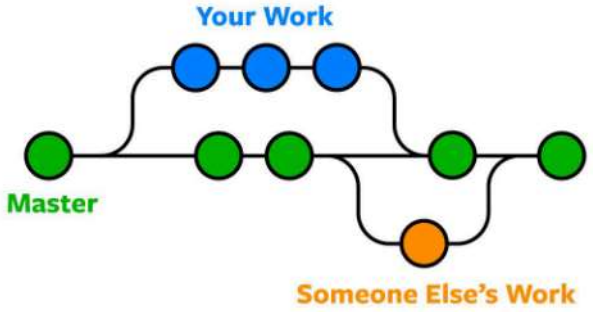
```
git commit -m "first commit"
```

Puedes reemplazar “first commit” con lo que quieras.

```
PS C:\Users\danie\Documents\my-app> git commit -m "first commit"
[master a4f4a20] first commit
 20 files changed, 31785 insertions(+), 19862 deletions(-)
 delete mode 100644 README.md
 delete mode 100644 public/favicon.ico
 delete mode 100644 public/logo192.png
 delete mode 100644 public/logo512.png
 delete mode 100644 public/manifest.json
 delete mode 100644 public/robots.txt
 delete mode 100644 src/App.css
 delete mode 100644 src/App.test.js
 create mode 100644 src/Home.jsx
 create mode 100644 src/Login.jsx
 create mode 100644 src/Register.jsx
 create mode 100644 src/ShoppingCart.js
 create mode 100644 src/db.json
 delete mode 100644 src/logo.svg
 delete mode 100644 src/setupTests.js
 create mode 100644 src/styles.js
```

wawiwa

Primer Repositorio Git en Github



Git branch:

Una "branch" (rama) representa una línea de desarrollo independiente

Las branches te permiten desarrollar funcionalidades, arreglar bugs, o experimentar de manera segura con ideas en una área contenida de tu repositorio.

El commando "git branch" te permite creat, enlistar, renombrar y borrar branches.

4. Para crear una nueva branch, corre el comando: `git branch -M main`

* main - es el nombre de la branch

wawiwa

Primer Repositorio Git en Github

Git remote:

El comando “git remote” te permite crear, ver y borrar conexiones con otros repositorios.

5. Corre la siguiente línea para conectar tu repositorio:

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

```
echo "# fisrt-try" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Dani3li12/fisrt-try.git
git push -u origin main
```

wawiwa

Primer Repositorio Git en Github

5. Git push:

El comando “git push” es usado para subir contenido de un repositorio local a un repositorio remoto. “Pushing” (empujar) es el proceso de transferir “commits” de tu repositorio local a un repositorio remoto.

Para hacer “push” corre esta línea:

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

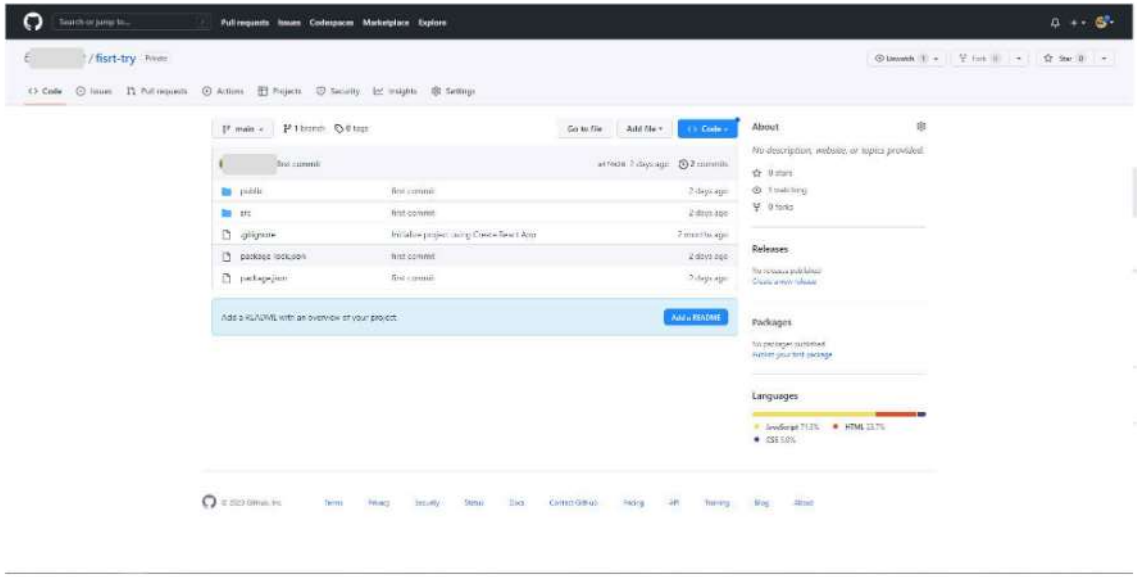
* main - es el nombre de la “branch”

wawiwa

Primer Repositorio Git en Github

¡Felicidades! ¡Tu primer repositorio está listo y ha sido “pushed”!

Trata de hacer “refresh” a la página de github, vas a ver la carpeta de tu proyecto.



wawiwa

Remote

En Git, un “remote” es una referencia a un repositorio localizado en otro servidor, típicamente en internet.

Un repositorio remoto es una copia de un repositorio en el que estás trabajando, pero está guardado en un servidor distinto.

Cuando clonas un repositorio, Git automáticamente crea un remoto llamado "origin", que apunta a repositorio original que clonaste.

wawiwa

Actualizar cambios del código a un repositorio Git

Para hacer “push” a los cambios de tu código a un repositorio Git, sigue estos pasos:

1. Usa el comando "**git add**" para agregar los cambios que quieres hacer “commit” al “staging area”. Si quieres hacer “commit” a todos los cambios en el directorio actual, corre el comando: **git add .** .
2. Usa el comando "**git commit**" para hacer “commit” a los cambios que agregaste al “staging area”. Por ejemplo, corre el comando: **git commit -m “Commit message here"** .
3. Usa el comando "**git push**" para hacer “push” a tus cambios a los que hiciste “commit” al repositorio Git. Por ejemplo, corre el comando: **git push origin master** para hacer push a tus cambios a la branch "master" del remoto "origin".

wawiwa

Práctica

Haz algunos cambios en tu código y haz “push”.

Haz “commit” con el mensaje “second commit”.

wawiwa

.gitignore

Git ignore es una funcionalidad de Git que te permite **especificar archivos y directorios** que quieres que Git ignore cuando haces “commit” a tus cambios a un repositorio.

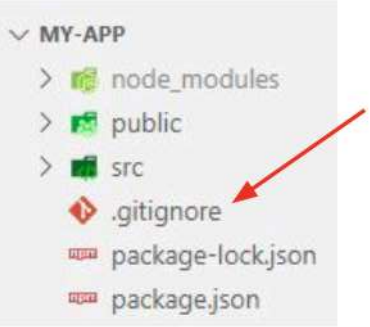
Cuando agregas archivos o directorios a un repositorio Git, Git le sigue la pista a esos archivos y directorios por defecto.

Sin embargo, pueden haber **archivos o directorios que no quieras seguir** o incluir en el repositorio, como **archivos temporales, archivos de “log” o construcción de artefactos.**

wawiwa

.gitignore

Para usar Git ignore, crea un archivo que se llame ".gitignore" en la raíz del directorio de tu repositorio Git.



Este archivo contiene una lista de archivos, directorios o patrones de archivos que Git debería ignorar. Por ejemplo, si quieres que Git ignore todos los archivos con la extensión ".log", puedes agregar la siguiente línea a tu archivo .gitignore: