

**GITHUB, GIT, GIT PAGES, TURORIALES Y ALGUNOS REPOSITORIOS PARA
CREACION DE TU PAGINA WEB.**

CONTENIDO

GITHUB- GIT.....	3
QUE ES GITHUB.....	3
HERRAMIENTAS DE GITHUB.....	5
COMO APRENDEMOS A USAR GITHUB.....	6
GIT.....	8
CARACTERISTICAS DE GIT.....	9
COMO CREAR UN REPOSITORIO.....	12

GITHUB – GIT

Antes de referirnos al tema más de uno nos preguntamos, que es GitHub, para que nos sirve, como la podremos usar.

Acá conocerás todo de esta aplicación que nos servirá y como nos contribuirá

QUE ES GITHUB

Es una plataforma o aplicación de desarrollo colaborativo de software donde en el podemos recrear nuestros proyectos utilizando Git que un software de control de versiones que nos ayudara alojar códigos de fuentes de una aplicación, esta plataforma nos sirve y nos brinda herramientas para nuestros trabajos de en equipo y nuestros proyectos.

Que herramientas nos brinda GitHub, GitHub es ua plataforma creada el 8 de febrero de 2018, para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de computadora. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010 para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente

de programas de computadora. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010. Anteriormente era conocida como *Logical Awesome LLC*. El código de los proyectos alojados en GitHub se almacena típicamente de forma pública, aunque utilizando una cuenta de pago, también permite hospedar repositorios privados “En la actualidad, GitHub es mucho más que un servicio de alojamiento de código. Además de éste, se ofrecen varias herramientas útiles para el trabajo en equipo. Entre ellas, caben destacar” (Copyright 2012, Luciano Castillo). Aparte de lo citado hasta el momento con la plataforma GitHub contribuimos a mejorar el software de los demás.



Copyright 2012, Luciano Castillo.

HERRAMIENTAS DE GITHUB

1. Una wiki para el mantenimiento de las distintas versiones de las páginas.
2. Un sistema de seguimiento de problemas que permiten a los miembros de tu equipo detallar un problema con tu software o una sugerencia que deseen hacer.
3. Una herramienta de revisión de código, donde se pueden añadir anotaciones en cualquier punto de un fichero y debatir sobre determinados cambios realizados en un commit específico.
4. Un visor de ramas donde se pueden comparar los progresos realizados en las distintas ramas de nuestro repositorio.

Su uso principal es creando programas de código abierto, fomentando.



Copyright 2012, Luciano Castillo.

En esta página podremos crear una cuenta gratuita y comenzar a subir repositorios de código (o crearlos desde 0), para que con la ayuda de todos esos proyectos mejore; así como también fortalecer los proyectos de los demás para crecer como grupo.

COMO APRENDEMOS A USAR GITHUB

1. Creamos una cuenta

A screenshot of the GitHub website's registration page. The page has a dark background with a light-colored registration form on the right. The form contains fields for 'Nombre de usuario' (Pick a username), 'Email' (you@example.com), and 'Contraseña' (Create a password). Below the password field is a note: 'Asegúrese de que tenga más de 15 caracteres o al menos 8 caracteres, incluido un número y una letra minúscula. Más información.' A green button labeled 'Registrarse en GitHub' is at the bottom of the form. To the left of the form, the text 'Construido para desarrolladores' is prominently displayed, followed by a paragraph about GitHub's platform. The top navigation bar includes links like '¿Por qué GitHub?', 'Empresa', 'Explorar', 'Mercado', 'Precios', a search bar, and buttons for 'Iniciar sesión' and 'Registrarse'.

2. Llenamos el formato

Nombre de usuario

Email

Contraseña

Asegúrese de que tenga más de 15 caracteres. O al menos 8 caracteres, incluido un número y una letra minúscula. [Más información...](#)

Registrarse en GitHub

Al hacer clic en "Registrarse en GitHub", usted acepta nuestros [términos de servicio](#) y [declaración de privacidad](#). Ocasionalmente le enviaremos correos electrónicos relacionados con la cuenta.

3. Entramos a nuestra cuenta y empezamos a usar gratuitamente la pagina

The screenshot shows the GitHub homepage for a user named 'malejacastellon10...'. The interface includes a top navigation bar with links for 'Solicitudes de extracción', 'Cuestiones', 'Mercado', and 'Explorar'. A central banner promotes learning Git and GitHub without code, featuring a 'Leer la guía' button and an 'Iniciar un proyecto' button. On the right, a 'Bienvenido al nuevo panel de control' message is displayed above a list of recommended repositories, including 'rieles / webpack', 'hotosm / tasking-manager', and 'Microsoft / accessibility-insights-web'. The bottom right corner indicates '5 notificaciones nue'.

GIT

es un software diseñado por Linus_Torvalds, “pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de **código fuente**. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos.

Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o **front end** como **Cogito** o **StGIT**. Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena. Hay algunos proyectos de mucha relevancia que ya usan Git, en particular, el grupo de **programación del núcleo Linux**.

El **mantenimiento del software** Git está actualmente (2009) supervisado por Junio Hamano, quien recibe contribuciones al código de alrededor de 280 programadores. En cuanto a derechos de autor Git es un **software libre** distribuible bajo los términos de la versión 2 de la **Licencia Pública General de GNU**”.
(Linus Torvalds (8 de abril de 2005).

CARACTERISTICAS DE GIT

1. Fuerte apoyo al desarrollo no lineal, por ende, rapidez en la gestión de ramas y mezclado de diferentes versiones. Git incluye herramientas específicas para navegar y visualizar un historial de desarrollo no line.
2. Los repositorios Subversion y svn se pueden usar directamente con git-svn.
3. Los almacenes de información pueden publicarse por HTTP, FTP, rsync o mediante un protocolo nativo, ya sea a través de una conexión TCP/IP simple o a través de cifrado SSH. Git también puede emular servidores CVS, lo que habilita el uso de clientes CVS pre-existentes y módulos IDE para CVS pre-existentes en el acceso de repositorios Git.
4. Realmacenamiento periódico en paquetes. Esto es relativo frente a la escritura de cambios y relativamente ineficiente para lectura si el reempaquetado (con base en diferencias) no ocurre cada cierto tiempo.
5. Los repositorios Subversion y svn se pueden usar directamente con git-svn.

A continuación, algunas de las ordenes que podemos darle a Git con sus respectivos significados:

1. `git fetch`:

Descarga los cambios realizados en el repositorio remoto.

2. `git merge <nombre_rama>`:

Impacta en la rama en la que te encuentras parado, los cambios realizados en la rama “nombre_rama”.

3. `git pull`:

Unifica los comandos *fetch* y *merge* en un único comando.

4. `git commit -am "<mensaje>":`

Confirma los cambios realizados. El “mensaje” generalmente se usa para asociar al *commit* una breve descripción de los cambios realizados.

5. `git push origin <nombre_rama>:`

Sube la rama “nombre_rama” al servidor remoto.

6. `git status`:

Muestra el estado actual de la rama, como los cambios que hay sin commitear.

7. `git add <nombre_archivo>:`

Comienza a trackear el archivo “nombre_archivo”.

8. `git checkout -b <nombre_rama_nueva>:`

Crea una rama a partir de la que te encuentres parado con el nombre “nombre_rama_nueva”, y luego salta sobre la rama nueva, por lo que quedas parado en esta última.

9. `git checkout -t origin/<nombre_rama>:`

Si existe una rama remota de nombre “nombre_rama”, al ejecutar este comando se crea una rama local con el nombre “nombre_rama” para hacer un seguimiento de la rama remota con el mismo nombre.

10. `git branch`:

Lista todas las ramas locales.

11. `git branch -a:`

Lista todas las ramas locales y remotas.

12. `git branch -d <nombre_rama>:`

Elimina la rama local con el nombre “nombre_rama”.

13. `git push origin <nombre_rama>:`

Commitea los cambios desde el branch local origin al branch “nombre_rama”.

14. `git remote prune origin:`

Actualiza tu repositorio remoto en caso de que algún otro desarrollador haya eliminado alguna rama remota.

15. `git reset --hard HEAD:`

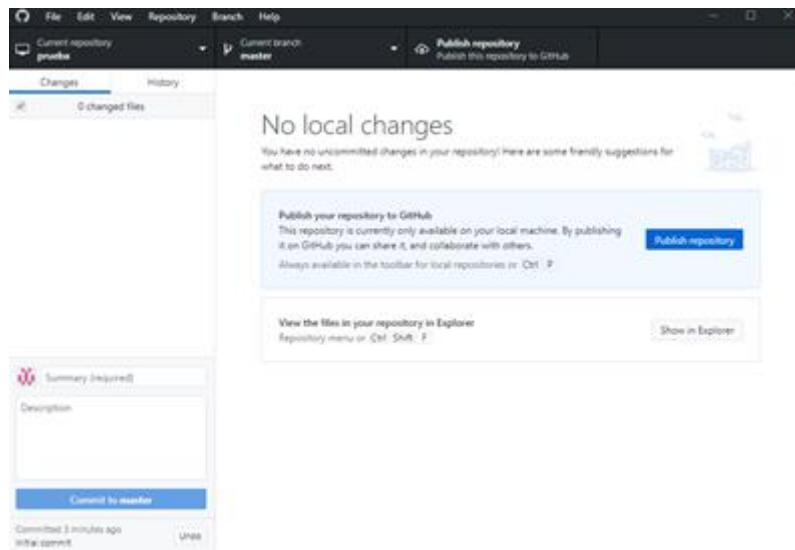
Elimina los cambios realizados que aún no se hayan hecho *commit*.

16. `git revert <hash_commit>:`

Revierte el *commit* realizado, identificado por el “hash_commit”. (Linus Torvalds (8 de abril de 2005).


COMO CREAR UN REPOSITORIO

Para crear un repositorio en GitHub, solo hay que seleccionar el botón “Create a New Repo”, de la barra de herramientas, habiendo entrado a GitHub con tu cuenta:



1. habrá que llenar dos datos, Nombres del repositorio, Descripción (opcional).

Owner

 **LuchoCastillo** ▾


Repository name


Repositorio ✓

Great repository names are short and memorable. Need inspiration? How about **furry-octo-nemesis**.

Description (optional)

Soy un repositorio :)

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

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

☐ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately.

Add .gitignore: **None** ▾

Create repository

(Copyright 2012, Luciano Castillo).