**FUNDAMENTOS DE SISTEMAS DE INFORMACÓN**

SEGUIMIENTO #1

**PRESENTADO POR:**

ELVER ARROYAVE

ALEJANDRO BUILES

JEFERSON VELASQUEZ

**PRESENTADO A:**

CORONADO GARCIA ROBINSON

**ENTIDAD EDUCATIVA**

UNIVERSIAD DE ANTIOQUIA

**PROGRAMA:**

INGENIERÍA DE SISTEMAS



**FECHA:**

14-septiembre del 2020

# ¿Qué es Git?

Git es un sistema de control de versiones o VCS como se conoce en sus siglas. La funcionalidad de un VCS es poder crear un historial de nuestros proyectos que nos permita tener de forma ordenada un versiona miento del código o proyecto que estemos trabajando, un ejemplo de algunos VCS son Maven, SourveSafe, Bazza, Mercurial, Git (El más usado por mucho).

Si hablamos de Git podemos decir que es tan usado que cuando hablamos de un VCS de una ves pensamos en Git. Fue creado inicialmente hace más o menos 15 años por quien si no más él gran Linus Torvalds, el cual nosotros los amantes del Software libre le debemos mucho. Y obvio Git es un software libre.

Las principales características de git y que lo hacen muy esplendido es que es que podemos crear un repositorio de forma local en nuestras máquinas de una manera tan sencilla y manejable que hasta un niño de primaria lo haría. 😊

Con este repositorio podemos controlar el versiona miento de nuestro código, documentos, carpetas imágenes entre otras cosas que ya no se me ocurre más.

Hoy en día creo que la gran mayoría de los desarrolladores del mundo lo utilizan y en muchas partes es un requisito indispensable para laborar como desarrollador en una empresa.

Algunos ejemplos de las cosas que podemos hacer en git son:

* Podemos crear ramas para desarrollar en nuestro código, donde podemos realizar cambios si tener que tocar la rama principal de nuestro código, con esto estamos asegurando que esta rama principal no tendrá ningún problema.
* Podemos fusionar las ramas luego de que hayamos terminado de hacer la modificación, corrección o agregación de una nueva funcionalidad.
* Podemos recuperar código que hayamos perdido.
* Volver a puntos de la historia de nuestro proyecto que queramos, de una forma simple.}
* Podemos trabajar de forma remota con otros desarrolladores en otros lugares del mundo y aun así aplicar todas las funcionalidades mencionadas anteriormente.

Hasta ahora hemos hablado de git como un **repositorio local** y es esplendido aquí desarrollamos la mayoría de las operaciones en nuestras máquinas y nos hemos dado cuenta de que podemos crear un repositorio en nuestros PC sin tener uno remoto. Ahora si no tes has sorprendido lo suficiente también podemos crear un **repositorio remoto o global** (Más adelante hablaremos de una potente herramienta que nos permite hacer esto) en el cual nuestro repositorio estará en la nube donde lo podremos compartir con otras personas de nuestro equipo de desarrollo o a la comunidad en general para que lo puedan ver, descargar, editar y muchas cosas más, todo esto dependiendo las opciones que habilitemos y queramos que se hagan.

**¿Qué es Github?**

pero agrega muchas características propias como interfaz gráfica basada en la web, control de acceso y varias funciones de colaboración. Si Git es el corazón, Hub es como su alma, es lo que convierte una línea de comandos Git en la red social más grande para desarrolladores.

Aloja los repositorios de código y brinda herramientas muy útiles para el trabajo en equipo, puedes contribuir a la mejora del software de otros, para esto provee de algunas funcionalidades para hacer un fork y solicitar un pulls.

Su principal funcionalidad es la bifurcación, copia un repositorio de la cuenta de un usuario a otro mediante un **fork**, es la clonación de un repositorio ajeno para trabajar en él, corregir, mejorar o implementar algo, después de realizar modificaciones puedes enviar un **pull** al dueño del proyecto, este podrá analizar los cambios y los podrá adjuntar al repo original.

Además del servicio de alojamiento, ofrece varias herramientas útiles para el trabajo en equipo como:

* **Wiki:** para el mantenimiento de las distintas versiones de las páginas.
* **Sistema de seguimientos de problemas**: permite a todos los miembros del equipo detallar problemas de tu software o sugerencias que quieran hacer.
* **Herramienta de revisión de código**: puede añadir anotaciones en cualquier punto de un fichero y debatir sobre determinados cambios realizados en un commit.
* **Visor de ramas**: se pueden comparar los progresos realizados en las distintas ramas de nuestro repo.

Referencias

* <https://codeburst.io/git-and-github-in-a-nutshell-b0a3cc06458f>
* <https://www.drauta.com/5-softwares-de-control-de-versiones>
* <https://conociendogithub.readthedocs.io/en/latest/data/introduccion/>
* <https://git-scm.com/about>
* <http://www.7sabores.com/blog/git-comandos-basicos>