1. **GIT**
   1. **¿Qué es Git?**

Git es una herramienta que realizar el sistema de control de versiones de código de forma distribuida. Es de código abierto, con mantenimiento activo y la herramienta de este tipo más empleada en el mundo.

Muchos proyectos de software dependen de Git para el control de versiones, incluidos proyectos comerciales y de código abierto.  Esta herramienta fue creada por Linus Torvalds, el famoso creador del kernel del sistema operativo Linux, en 2005

Este sistema funciona muy bien en una amplia variedad de sistemas operativos e IDE (entornos de desarrollo integrados). Git cuenta con una arquitectura distribuida. En lugar de tener un único espacio para todo el historial de versiones del software, propio de los sistemas de control como CVS o Subversion, aquí la copia de trabajo del código de cada desarrollador es un repositorio que puede albergar el historial completo de todos los cambios.

Rendimiento, seguridad y flexibilidad son las bases sobre las que se ha diseñado Git. Es una herramienta muy potente, rápida, ágil y de software libre. Tiene un sistema de trabajo con ramas que lo hace especialmente potente. Estas ramas se destinan a hacer proyectos divergentes de un proyecto principal, para hacer experimentos o para probar nuevas funcionalidades. Cada rama puede tener una línea de progreso diferente de la rama principal donde está el core de nuestro desarrollo. Podemos algunas de esas mejoras o cambios en el código y hacer una fusión a nuestro proyecto principal.

* 1. **¿Cuál es su Propósito Principal?**

Su propósito es llevar registro de los cambios en archivos de computadora incluyendo coordinar el trabajo que varias personas realizan sobre archivos compartidos en un repositorio de código.

Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o [front end](https://es.wikipedia.org/wiki/Front-end_y_back-end" \o "Front-end y back-end) como [Cogito](https://es.wikipedia.org/wiki/Cogito_(software)) o [StGIT](https://es.wikipedia.org/w/index.php?title=StGIT&action=edit&redlink=1" \o "StGIT (aún no redactado)). [2](https://es.wikipedia.org/wiki/Git#cite_note-2)​Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena. [3](https://es.wikipedia.org/wiki/Git#cite_note-3)​Hay algunos proyectos de mucha relevancia que ya usan Git, en particular, el grupo de [programación](https://es.wikipedia.org/wiki/Programaci%C3%B3n) del [núcleo Linux](https://es.wikipedia.org/wiki/N%C3%BAcleo_Linux).

* 1. **Explique los Conceptos Básicos de control de Versiones**

Un control de versiones es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante. Aunque en los ejemplos de este libro usarás archivos de código fuente como aquellos cuya versión está siendo controlada, en realidad puedes hacer lo mismo con casi cualquier tipo de archivo que encuentres en una computadora.

* **Sistema de Control de Versiones Locales**

Este método es muy común porque es muy sencillo, pero también es tremendamente propenso a errores. Es fácil olvidar en qué directorio te encuentras y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no querías.

Para afrontar este problema los programadores desarrollaron hace tiempo VCS locales que contenían una simple base de datos, en la que se llevaba el registro de todos los cambios realizados a los archivos.

* **Sistema de Control de Versiones Centralizadas**

Los sistemas de Control de Versiones Centralizados (CVCS por sus siglas en inglés) fueron desarrollados para solucionar este problema. Estos sistemas, como CVS, Subversion y Perforce, tienen un único servidor que contiene todos los archivos versionados y varios clientes que descargan los archivos desde ese lugar central. Este ha sido el estándar para el control de versiones por muchos años.

* **Sistema de Control de Versiones Distribuidos**

Los sistemas de Control de Versiones Distribuidos (DVCS por sus siglas en inglés) ofrecen soluciones para los problemas que han sido mencionados. En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no solo descargan la última copia instantánea de los archivos, sino que se replica completamente el repositorio. De esta manera, si un servidor deja de funcionar y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios disponibles en los clientes puede ser copiado al servidor con el fin de restaurarlo.

1. **GITHUB**
   1. **¿Qué es GitHub?**

GitHub es una plataforma de desarrollo colaborativo que aloja proyectos en la nube utilizando el sistema de control de versiones llamado Git. Ayuda a los desarrolladores a almacenar y administrar el código llevando un registro de cambios. Generalmente el código es abierto, lo que permite realizar proyectos compartidos y mantener el seguimiento detallado de su progreso. La plataforma GitHub también funciona como red social conectando a los desarrolladores con los usuarios. Como usuario puedes descargar programas o aplicaciones, y de la misma manera puedes aportar a su desarrollo ofreciendo mejoras y discutir las cuestiones que te interesan en foros temáticos.

* 1. **Cómo se Relaciona con Git**

Git es una herramienta de control de versiones distribuida que puede gestionar el historial de código fuente de un proyecto de desarrollo, mientras que GitHub es una plataforma basada en la nube construida alrededor de la herramienta Git.

* 1. **Enumere algunas Características Claves de GitHub**
* **Repositorios**: Los repositorios son la característica principal de GitHub y permiten a los desarrolladores almacenar y administrar el código.
* **Ramificaciones**: Las ramificaciones permiten a los desarrolladores crear copias separadas del mismo repositorio para probar cambios nuevos y evitar que los cambios nuevos afecten el código principal.
* **Colaboración**: GitHub permite a los desarrolladores trabajar juntos en el mismo proyecto al mismo tiempo, lo que les permite realizar cambios simultáneamente.
* **Seguimiento de errores**: GitHub proporciona una herramienta de seguimiento de errores para ayudar a los desarrolladores a identificar y corregir errores en el código.
* **Integración con herramientas de terceros**: Los desarrolladores pueden conectar GitHub con otras herramientas de terceros para acelerar el proceso de desarrollo.
* **Mantenimiento de la versión**: GitHub permite a los desarrolladores rastrear y administrar las versiones anteriores del código, lo que les permite volver a versiones anteriores si es necesario.

1. **USO COMÚN** 
   1. **En que Situaciones se Utiliza Git y GitHub**

Git y GitHub funcionan juntos, pero no son lo mismo.

GitHub es una plataforma donde puedes crear y mantener software e interactuar con otros desarrolladores. Git es el programa que utiliza GitHub para hacer que su plataforma sea perfecta y funcional.

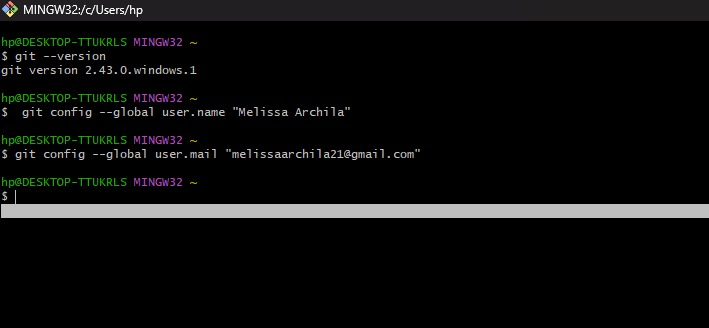
* 1. **¿Cómo Beneficia a los Desarrolladores y Equipos de Desarrollo?**

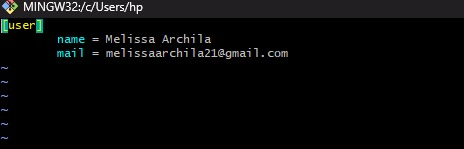
Git y GitHub facilitan que los desarrolladores colaboren en proyectos, incluso si están ubicados en diferentes partes del mundo. GitHub proporciona una plataforma para que los desarrolladores compartan su código, realicen un seguimiento de los problemas y administren las solicitudes de incorporación de cambios. Con Git,**los desarrolladores pueden trabajar en un proyecto simultáneamente y el sistema de control de versiones ayuda a mantener a todos en sintonía.**

C++ es un lenguaje de programación de alto nivel, orientado a objetos y de propósito general que ayuda a los desarrolladores a escribir programas rápidos y portables. A pesar de tener 40 años (fue creado en 1979), se encuentra entre los lenguajes de programación más utilizados en 2020.

**Descargar e Instalar Git**

* Configurar el nombre de usuario y el correo electrónico usando los comandos:
  + git config --global user.name "Tu Nombre"
  + git config --global user.email "tu@email.com"





**Descargar e Instalar GitHub**

* Crear una cuenta en [GitHub](https://github.com/).
* Configurar la autenticación de dos factores (si es deseado).

