

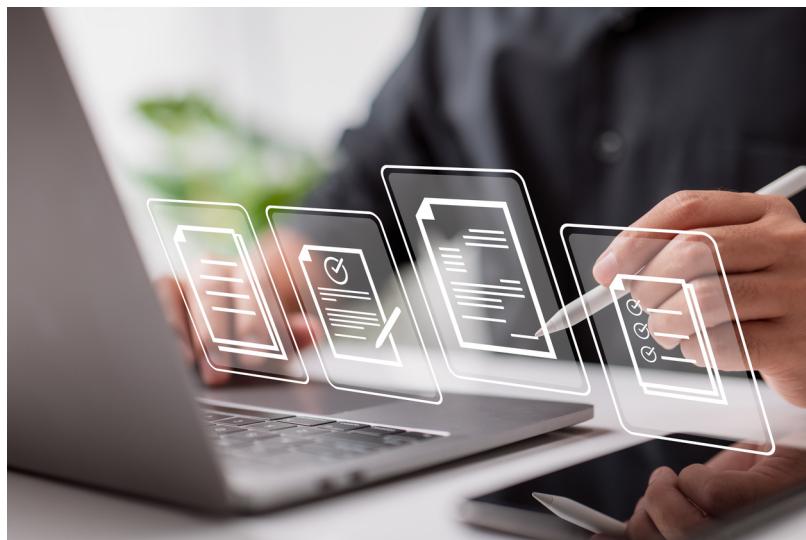


Frontend Fundamentals

Git y Terminal

Introducción

En la industria del software se habla mucho de Git, que es una aplicación útil para quien escribe código y quiere llevar un seguimiento de los avances o cambios que realiza durante el desarrollo de un programa. Git ofrece una interfaz gráfica web para la gestión de cambios, lo que comúnmente se conoce como sistema de control de versiones. Esto quiere decir que permite mantener un historial de cambios que se realicen en el proyecto web y poder acceder a ellos en todo momento.



Imagina que estás trabajando en la redacción de un documento en Word y vas agregando contenido cada cierto tiempo, donde al momento de revisarlo te dan la instrucción de que descartes el último cambio realizado y regreses como se encontraba en la primera revisión, ¿cómo lo harías?

Tal vez pensarías en crear varios documentos conforme vas avanzando en tu redacción, es decir, guardar una versión del documento por cada cambio que realices. Sin embargo, no es lo óptimo, mucho menos cuando se hace desarrollo web, porque siempre van a suceder cambios y será necesario revisar el avance.

Introducción

¿Qué es y para qué sirve Git?

El control de versiones trata sobre la gestión de múltiples versiones correspondientes a un mismo proyecto. Para administrar una versión se debe realizar un seguimiento de cada cambio en los archivos incluidos en el proyecto, es decir, si se agregan nuevos, si se editan los existentes o eliminan otros.

Git es un sistema de control de versiones, más aún, es un sistema distribuido de control de versiones, lo que implica que quien trabaje en un proyecto en Git tiene una copia del historial completo del proyecto y no solo del estado actual de los archivos en él. Está diseñado para ser eficiente y confiable sobre el mantenimiento de versiones de aplicaciones cuando se tiene un gran número de archivos de código fuente.

Pero ¿qué es el control de versiones? Es todo aquello que haces cuando subes código en la nube, añades algo adicional o simplemente editas aquello que no funciona como debería y lo corriges. Todas estas acciones van generando versiones dentro del proyecto y la forma en como ordenas dichos cambios se llama control de versiones y Git es un sistema que te ayuda con este control.

Una de las ventajas de usar este sistema de control de versiones es la facilidad de deshacer cambios. Si por algún motivo alguien del equipo de trabajo comete algún error, es posible corregirlo regresando a un momento previo y recuperar una versión del proyecto anterior (al cambio donde sucedió el error).

Adicionalmente, se tiene un historial completo de los cambios, de tal manera que es posible revisar cómo estaba el proyecto días, semanas, meses o años atrás y el estado exacto de cada uno de los archivos en esos instantes del tiempo.

Con Git también es posible llevar un registro de los cambios realizados, al facilitar la escritura de mensajes cada vez que se actualiza algún archivo dentro de un proyecto o se libera una nueva versión y revisar las diferencias entre archivos en diferentes versiones.

Git trabaja con tres estados del proyecto principales en los que se pueden encontrar tus archivos:

- **Confirmado (committed)**: los datos se encuentran almacenados de manera segura en la base de datos local.
- **Modificado (modified)**: se modificó el archivo, pero aún no se ha confirmado en la base de datos.
- **Preparado (staged)**: se ha marcado un archivo modificado en la versión actual para que vaya en la próxima confirmación.

Esto conlleva a las tres secciones o “árboles” de un proyecto con que trabaja Git:

1. **Directorio de trabajo (working directory)**. Corresponde a una toma instantánea de donde se está actualmente trabajando.
2. **Área de preparación (staging area)**. Es aquella que aloja todos los archivos que se han modificado, agregado o eliminado y están listos para almacenarse en una base de datos.
3. **Directorio git (Git directory)**. Es esa base de datos que guarda el historial de cambios de un proyecto.



Figura 1. Los tres estados de Git.

Con Git, cada integrante de un equipo de trabajo tiene una copia del proyecto original; los cambios realizados también se comparten con todos y se tiene un mecanismo de comunicación regular para compartir los avances, modificaciones o actualizaciones de un proyecto.

Ahora bien, es importante mencionar que la forma en que almacenes tu código es imprescindible para tu proyecto, para esto existen los repositorios, que es el lugar donde guardarás, organizarás y mantendrás todos tus archivos de código.

Puedes tener un repositorio local (en tu propia computadora) o bien, en alguna plataforma que mejor se adapte a tus necesidades. Por mencionar un par de ejemplos, existen GitHub y BitBucket. El repositorio de código que utilices siempre será un papel fundamental en el flujo de trabajo de tu desarrollo.

Comandos básicos en Git

La forma de interactuar con Git, generalmente, es a través de línea de comandos, de los cuales se describen a continuación algunos comandos básicos que te permitirán comenzar a trabajar con Git.

- **git init:** convierte un directorio en un repositorio Git. Este es el primer paso que se realiza al crear un repositorio y después de esto es posible agregar y confirmar archivos y directorios. En otras palabras, es el comando para iniciar un repositorio vacío en alguna carpeta específica.

```
> git init
```

Fuente: Azure DevOps. (2022). ¿Qué es Git? Recuperado de

<https://docs.microsoft.com/es-es/devops/develop/git/what-is-git> con fines educativos.

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

- **git add:** agrega archivos o directorios al área de preparación (staging). Antes de que un archivo o directorio esté disponible para confirmar, se debe ejecutar este comando.

```
> git add
```

Fuente: Azure DevOps. (2022). ¿Qué es Git? Recuperado de

<https://docs.microsoft.com/es-es/devops/develop/git/what-is-git> con fines educativos.

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

- **git add “nombre_de_archivo”:** Sirve para añadir un archivo específico:
- **git add:** Sirve para añadir todos los archivos del directorio:
- **git commit:** guarda (confirma) los cambios hechos a los archivos en el repositorio local. Es una buena práctica incluir un mensaje descriptivo sobre los cambios realizados en cada confirmación.

```
> git commit
```

Fuente: Azure DevOps. (2022). ¿Qué es Git? Recuperado de

<https://docs.microsoft.com/es-es/devops/develop/git/what-is-git> con fines educativos.

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

- **git commit -am “mensaje”:**
Sirve para confirmar los cambios realizados. Generalmente se usa el “mensaje” para asociar al commit una pequeña descripción de los cambios realizados.
- **git revert “hash_commit”:**
Sirve para revertir el commit identificado por “hash_commit”.

- **git status:** muestra el estado actual de un repositorio (rama de trabajo). Si un archivo está en preparación y sin confirmación, el comando lo indicará. Si no hay cambios en archivos o directorios, regresará un “nothing to commit” o “working directory clean”.



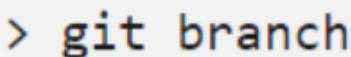
git status

Fuente: Azure DevOps. (2022). ¿Qué es Git? Recuperado de

<https://docs.microsoft.com/es-es/developer/get-started/git/what-is-git> con fines educativos.

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

- **git config:** se usa para configurar las opciones de un repositorio. Se recomienda consultar la documentación de Git para conocer todas las opciones disponibles. Hay dos opciones de importancia a configurar: user.name y user.email, utilizadas en las confirmaciones de cambios.
- **git branch:** se usa para consultar en qué rama se encuentra el repositorio local, crear o eliminar ramas.



> git branch

Fuente: Azure DevOps. (2022). ¿Qué es Git? Recuperado de

<https://docs.microsoft.com/es-es/developer/get-started/git/what-is-git> con fines educativos.

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

- **git merge:** integra los cambios de una rama con otra, por ejemplo, integrar los cambios hechos en la rama “nueva funcionalidad” a la rama “estable”.

Crear tu cuenta

Para comenzar a utilizar Git, primero debes instalarlo en tu computadora de forma local, esto es una condición indispensable. El proceso es muy sencillo ya que es prácticamente igual a la instalación de cualquier otro software que seguramente has realizado. Puedes instalarlo tanto en Sistemas Operativo Windows o de Mac, lo debes bajar de los repositorios de software en la página oficial, considerando la versión de tu sistema operativo:

Como ya se mencionó, se instala como cualquier otro software. Si es Windows se tendrá un asistente al que deberás dar clic en “siguiente”, “siguiente”, “siguiente”, y así hasta acabar el proceso.

Posterior a la instalación de Git, se lanzan algunos comandos de configuración:

```
git config --global user.name "tu_nombre"  
git config --global user.email "tu_email@dominio.com"
```

Al ejecutar estos comandos estás indicando tu nombre de usuario y tu email, lo cual sirve para que cuando hagas commits en el repositorio local, se guarden con la referencia a ti mismo. Esto funciona para que posteriormente cuando se obtenga información de los cambios realizados en los archivos contenidos del repositorio local, aparezca como responsable de estos cambios el usuario y correo que has indicado.

Considera que puedes crear una cuenta en alguno de los sitios GitHub o BitBucket, para hacer uso de estas plataformas; algo que verás en la actividad guiada.



Tecmilenio no guarda relación alguna con las marcas mencionadas como ejemplo. Las marcas son propiedad de sus titulares conforme a la legislación aplicable, se utilizan con fines académicos y didácticos, por lo que no existen fines de lucro, relación publicitaria o de patrocinio.

A lo largo de este tema conociste Git que es un sistema de control de versiones gratuito y de código abierto usado para manejar proyectos de distintos tamaños de forma eficiente.

Git es, sin duda, el más popular de los sistemas de control de versiones en la actualidad y se utiliza para realizar un seguimiento de los cambios en el código fuente, lo que permite que varios desarrolladores trabajen juntos en el desarrollo de un proyecto.



Ahora conoces una nueva herramienta para utilizar en tus futuros proyectos web, para guardar diferentes versiones del código fuente y que puedas volver a una versión anterior cuando sea necesario.

Referencias bibliográficas

- Azure DevOps. (2022). *¿Qué es Git?* Recuperado de <https://docs.microsoft.com/es-es/devops/develop/git/what-is-git>
- DesarrolloWeb. (2022). *Git.* Recuperado de <https://desarrolloweb.com/home/git>
- Git-scm. (2022). *Fundamentos de Git.* Recuperado de <https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Fundamentos-de-Git>

Para saber más

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

Lecturas

Para conocer más acerca de **Instalación y configuración de Git**, te sugerimos leer lo siguiente:

- Docs Microsoft. (2022). *Instalación de Git para Windows.* Recuperado de <https://docs.microsoft.com/es-es/devops/develop/git/install-and-set-up-git#install-git-for-windows>
- Docs Microsoft. (2022). *Instalación de Git para macOS.* Recuperado de <https://docs.microsoft.com/es-es/devops/develop/git/install-and-set-up-git#install-git-for-macos>

Videos

Para conocer más acerca de **Git**, te sugerimos revisar lo siguiente:

- Platzi. (2019, 20 mayo). *¿Qué es Git y GitHub? - Repositorios, ramas y mucho más [Archivo de video].* Recuperado de <https://www.youtube.com/watch?v=DinlgacaWs>
- Programador X. (2021, 14 enero). *Aprende GIT en 15 minutos [Archivo de video].* Recuperado de <https://www.youtube.com/watch?v=M8H-mT4oeAg>

Asegúrate de:

- Comprender qué es Git y cómo funciona.
- Identificar un repositorio.
- Comprender la importancia de un sistema de control de versiones.

Requerimientos Técnicos

- Computadora con acceso a internet.
- Permisos de administrador y/o Git instalado previamente. Studio.

Prework

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

- Revisa previamente la lectura del tema Git y Terminal.
- Descarga la herramienta Git del sitio oficial dependiendo tu sistema operativo.



Git. (s.f.). 1.5 Inicio - Sobre el Control de Versiones -
Instalación de Git. Recuperado de
<https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Instalaci%C3%B3n-de-Git>

El uso y descarga del software deberá apegarse a los términos y condiciones del sitio oficial del fabricante y su uso será responsabilidad de quien lo descargue. Tecmilenio no tiene licencia ni posee los derechos sobre dicho software.

- Una vez instalado Git, trabaja con la consola de comandos para aprender a usar este sistema.
- Crea un directorio principal donde coloques todos tus repositorios y archivos.
- Crea una carpeta específica para tu primer proyecto en Git.

El uso y descarga del software deberá apegarse a los términos y condiciones del sitio oficial del fabricante y su uso será responsabilidad de quien lo descargue. Tecmilenio no tiene licencia ni posee los derechos sobre dicho software.

La obra presentada es propiedad de ENSEÑANZA E INVESTIGACIÓN SUPERIOR A.C. (UNIVERSIDAD TECMILENIO), protegida por la Ley Federal de Derecho de Autor; la alteración o deformación de una obra, así como su reproducción, exhibición o ejecución pública sin el consentimiento de su autor y titular de los derechos correspondientes es constitutivo de un delito tipificado en la Ley Federal de Derechos de Autor, así como en las Leyes Internacionales de Derecho de Autor.

El uso de imágenes, fragmentos de videos, fragmentos de eventos culturales, programas y demás material que sea objeto de protección de los derechos de autor, es exclusivamente para fines educativos e informativos, y cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por UNIVERSIDAD TECMILENIO.

Queda prohibido copiar, reproducir, distribuir, publicar, transmitir, difundir, o en cualquier modo explotar cualquier parte de esta obra sin la autorización previa por escrito de UNIVERSIDAD TECMILENIO. Sin embargo, usted podrá bajar material a su computadora personal para uso exclusivamente personal o educacional y no comercial limitado a una copia por página. No se podrá remover o alterar de la copia ninguna leyenda de Derechos de Autor o la que manifieste la autoría del material.