Guía para Instalar GitHub y Comprender Git 🚀

La herramienta por excelencia para trabajo colaborativo

Introducción a Git y GitHub 📚

¿Qué es Git? 🤔

Git es una herramienta de control de versiones que permite a los programadores rastrear y gestionar cambios en su código de manera eficiente. Funciona mediante la creación de "instantáneas" de tu proyecto a lo largo del tiempo, lo que facilita el seguimiento de quién hizo qué y cuándo.

¿Qué es GitHub? 🌐

GitHub es una plataforma en línea que utiliza Git como base. Permite alojar repositorios de código y colaborar en proyectos de programación de manera colaborativa y distribuida.

Paso 1: Instalación de Git en tu computadora 💻

Para empezar a utilizar Git, primero debes instalarlo en tu computadora. Aquí tienes los pasos:

1. Descarga Git: Ve a **git-scm.com** y selecciona la versión adecuada para tu sistema operativo (Windows, macOS, Linux). Sigue las instrucciones de instalación.
2. Verifica la instalación: Abre tu terminal o línea de comandos y ejecuta el siguiente comando:

   ```
   shellCopy code
   git --version
   ```

   Deberías ver la versión de Git instalada.

Paso 2: Configuración Inicial de Git 🛠️

Antes de empezar a utilizar Git, configura tu nombre y correo electrónico para que los cambios que realices queden registrados correctamente:

```
shellCopy code
git config --global user.name "Tu Nombre"
git config --global user.email "tu@email.com"
```

Paso 3: Iniciando un Repositorio Git 🏁

Ahora que tienes Git instalado y configurado, puedes crear un repositorio para tu proyecto:

1. Crea una carpeta para tu proyecto en tu computadora.
2. Abre una terminal en esa carpeta y ejecuta:

   ```
   shellCopy code
   git init
   ```

   Esto inicia un nuevo repositorio Git en tu proyecto.

Comandos Básicos de Git 📝

Añadir archivos al área de preparación:

- `git add <nombre_del_archivo>`: Agrega un archivo específico al área de preparación.
- `git add .`: Agrega todos los archivos modificados al área de preparación.

Hacer una confirmación (commit):

- `git commit -m "Mensaje de confirmación"`: Guarda los cambios en el repositorio con un mensaje descriptivo.

Ver el estado de tus archivos:

- `git status`: Muestra el estado actual de los archivos en el repositorio.

Ver el historial de confirmaciones:

- `git log`: Muestra un registro de todas las confirmaciones.

Trabajar con Repositorios Remotos:

- `git remote add origin <URL_del_repositorio>`: Conecta tu repositorio local con un repositorio remoto en GitHub (u otro servicio similar).
- `git push -u origin master`: Sube tus cambios locales al repositorio remoto en la rama "master". Después del primer uso, puedes usar simplemente `git push`.
- `git pull origin master`: Descarga los cambios del repositorio remoto a tu repositorio local. Esto es útil cuando otros colaboradores han realizado cambios.
- `git clone <URL_del_repositorio>`: Crea una copia local de un repositorio remoto en tu computadora.

Ramificaciones (Branches):

- `git branch`: Muestra una lista de todas las ramas locales en tu repositorio.
- `git branch <nombre_de_la_rama>`: Crea una nueva rama con el nombre especificado.
- `git checkout <nombre_de_la_rama>`: Cambia a una rama específica.
- `git merge <nombre_de_la_rama>`: Fusiona los cambios de una rama en la rama actual.

GUÍA PARA EL DÍA A DÍA EN PSP

Clonar un Repositorio:

1. Abre tu terminal o línea de comandos.
2. Navega hasta la ubicación en la que deseas clonar el repositorio. Por ejemplo, para clonarlo en tu directorio de documentos:

```
shellCopy code
cd ~/Documentos
```

3. Clona el repositorio utilizando el comando `git clone` y la URL del repositorio que deseas clonar. Por ejemplo:

```
shellCopy code
git clone https://github.com/nombredeusuario/nombre-del-repositorio.git
```

Esto creará una copia local del repositorio en tu computadora.

Actualizar un Repositorio Clonado:

Una vez que hayas clonado el repositorio, puedes mantenerlo actualizado con los cambios más recientes de la siguiente manera:

1. Navega al directorio del repositorio clonado:

```
shellCopy code
cd nombre-del-repositorio
```

2. Verifica el estado actual del repositorio para asegurarte de que estás en la rama correcta y no tienes cambios locales sin confirmar:

```
shellCopy code
git status
```

3. Para obtener los últimos cambios del repositorio remoto, utiliza el comando `git pull`:

```
shellCopy code
git pull
```

   Esto descargará los cambios más recientes del repositorio remoto y los aplicará en tu copia local.

Ahora, deberías de ser capaz de clonar un repositorio y mantenerlo actualizado con los cambios más recientes. Es importante recordar que debes estar en el directorio correcto del repositorio antes de ejecutar comandos como `git pull` para que funcionen correctamente.

¡A por ello! 🎉

Git puede parecer intimidante al principio, pero con la práctica se vuelve más fácil. No dudes en explorar, cometer errores y aprender de ellos. ¡La colaboración en GitHub te permitirá trabajar en proyectos emocionantes con otros programadores de todo el mundo!

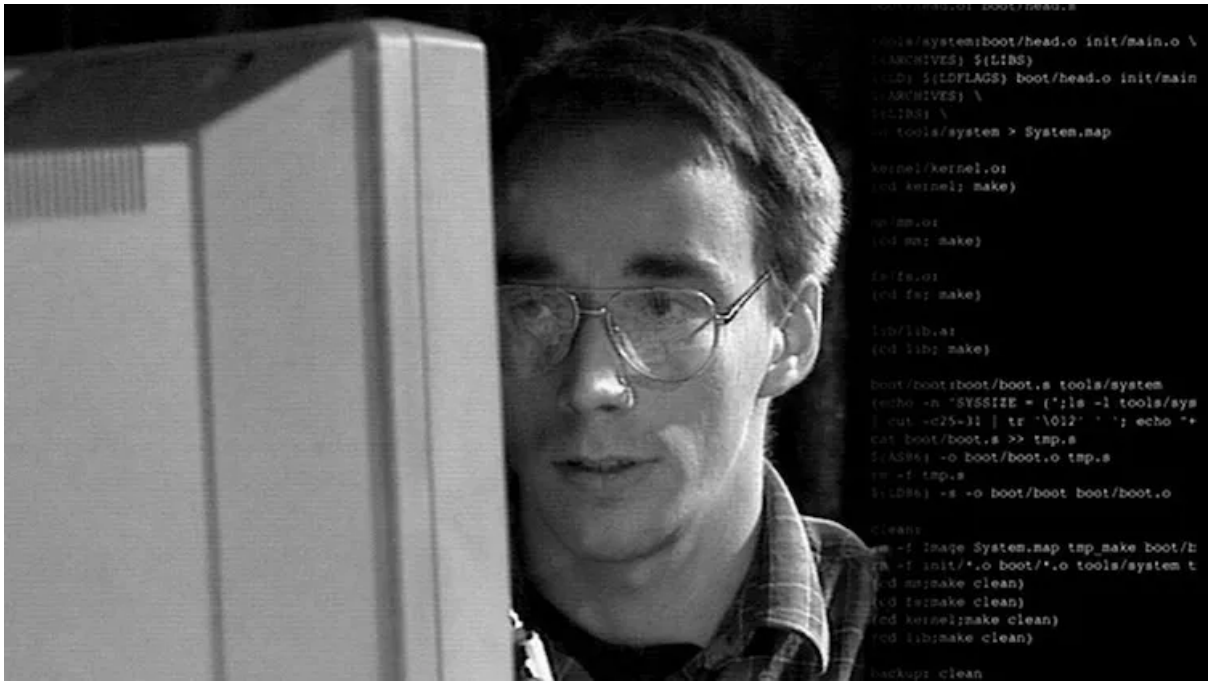
Anécdota

El creador de Git, Linus Torvalds, lo inició porque estaba frustrado con las herramientas de control de versiones existentes. Quería algo rápido y eficiente para el desarrollo de Linux, ¡y así nació Git!

Sobre Linus Tovalds 🔬

Linus Torvalds es un programador finlandés conocido principalmente por crear el núcleo del sistema operativo Linux. En 2005, cuando ya había trabajado en el desarrollo de Linux durante muchos años, necesitaba una herramienta de control de versiones para administrar el código fuente del proyecto.

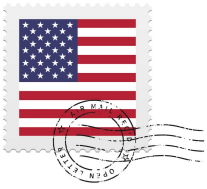
Para que entiendas cómo se creó GIT y con qué misión, te dejo una carta “escrita” por Linus Torvalds (ya os contaré quién la ha escrito realmente, si no lo adiviná is antes).



De: Linus

A: los estudiantes de Retamar

Desde Helsinki a 09 de septiembre de 2023, ¡sábado!



Querido estudiante de Retamar,

Es un placer compartir contigo mi historia sobre cómo surgió Git y cuáles fueron mis motivaciones para crearlo. Espero que encuentres esta historia inspiradora para tu propio viaje en el mundo de la programación y la tecnología.

Hace muchos años, cuando tenía alrededor de tu edad (21 años), me encontraba trabajando en un proyecto de programación llamado Linux. Este proyecto había crecido enormemente y se había vuelto complejo, con miles de archivos y líneas de código. Era un desafío mantenerlo organizado y colaborar con otros programadores.

Mi principal fuente de frustración en ese momento era el control de versiones. Usábamos herramientas de control de versiones como CVS, pero eran lentas y

torpes. Imagina intentar manejar un proyecto gigantesco como Linux, donde cada pequeño cambio debe ser registrado y compartido con otros desarrolladores, y estas herramientas simplemente no lo estaban haciendo bien.

Además, quería que el proceso de colaboración fuera más fluido y que los programadores pudieran trabajar en diferentes partes del proyecto al mismo tiempo, sin problemas.

Así que, motivado por esta frustración y la necesidad de una solución más efectiva, decidí crear mi propia herramienta de control de versiones: Git.

La misión detrás de Git era simple pero poderosa:

1. Velocidad y Eficiencia: Quería que Git fuera increíblemente rápido. Cuando trabajas en un proyecto grande, cada segundo cuenta. Con Git, los desarrolladores pueden realizar cambios, rastrearlos y compartirlos con otros de manera casi instantánea.

2. Colaboración sin Obstáculos: Git estaba destinado a permitir la colaboración sin problemas. Quería que los programadores pudieran trabajar en sus propias ramas de desarrollo y luego fusionar fácilmente sus cambios con el trabajo de otros.

3. Distribución: Git también tenía que ser una herramienta que funcionara de manera distribuida. Esto significa que no estamos atados a una ubicación física específica para trabajar en un proyecto. Cualquier persona en cualquier parte del mundo puede contribuir y tener su propia copia completa del proyecto.

El proceso de creación de Git fue desafiante, pero también emocionante. Pasé horas pensando en cómo diseñar una herramienta que cumpliera con estas metas. Fue como armar un rompecabezas en el que cada pieza debía encajar perfectamente.

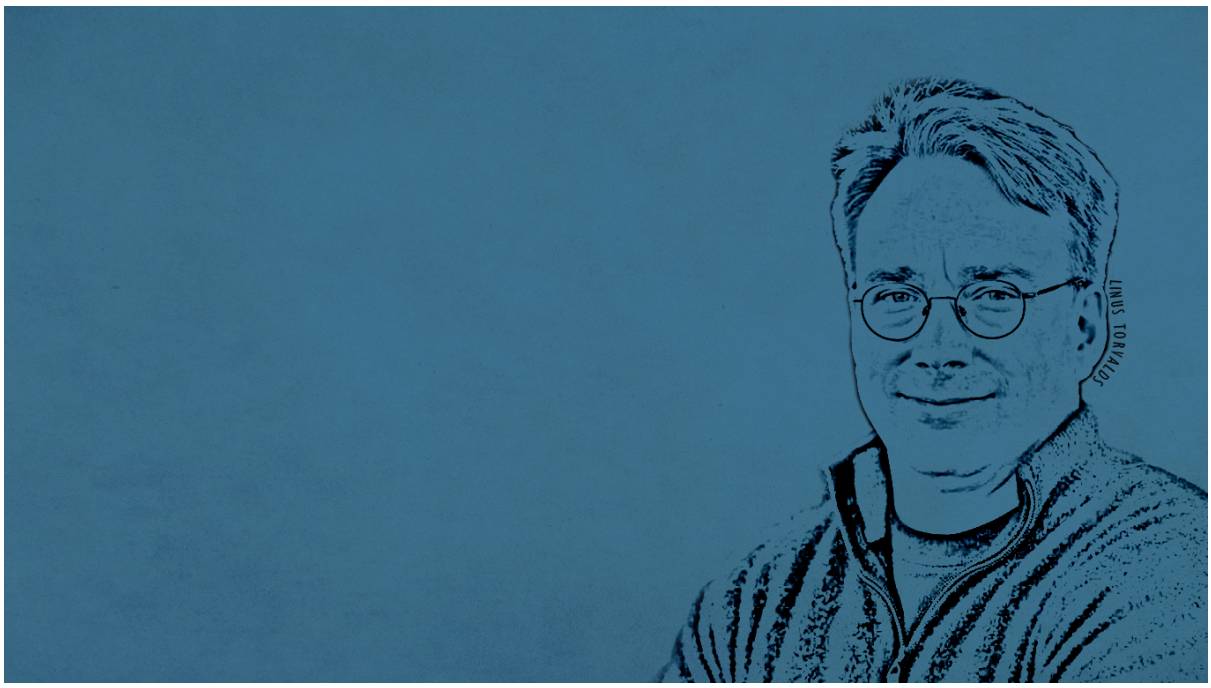
A medida que Git comenzó a tomar forma, la comunidad de desarrolladores comenzó a adoptarlo. Se convirtió en la base de proyectos de código abierto y en una herramienta esencial en el mundo de la programación.

Así que, querido estudiante, mi mensaje para ti es que no tengas miedo de enfrentarte a desafíos y frustraciones en tu viaje como programador. A veces, esas experiencias te llevarán a crear algo nuevo y valioso. Si tienes una visión clara de lo que quieres lograr, puedes superar cualquier obstáculo.

¡Espero que esta historia te inspire en tu propio camino hacia la programación y la tecnología!

Con admiración por tus esfuerzos por aprender tanto cada día,

Linus Torvalds



Frase de Linus sobre los avances de software en el mundo de la programación…. ¿qué pensais?

I often compare software development to biological processes where it really is evolution. I mean, it is not intelligent design. I'm there, in the middle of the thing and I can tell you it is absolutely not intelligent design whatever you want to think.”

Innovation that this industry talks about so much is bullshit. Anybody can innovate. […] Screw that, it's meaningless; 99% of this is just get the work done.

Recuerda, la práctica hace al maestro. ¡Buena suerte en tu aventura con Git y GitHub! 🚀👩‍💻👨‍💻