

Inst. Sup. San Pablo N° 9112



Desarrollo de Sistemas Web

Tec. Superior Análisis Funcional de Sistemas

GIT

¿Qué es?

Sistema de control de versiones que nos permite ver que cambios se han hecho en cada versión del desarrollo.

¿Por qué usarlo?

- Sistema distribuído.
- Viajar al pasado.
- Ramas y fusiones.

¿Es gratis?

Es un proyecto de código abierto que fué desarrollado por Linus Torvalds (2005) que hoy cuenta con un desarrollo activo por toda la comunidad.

¿Posee interfaz gráfica?

Sí, tiene GUI pero no es tan amigable como uno quisiera. El fuerte de GIT se encuentra en su formato "bash" o más conocido como *consola*.

Conceptos fundamentales

Repositorio

Lugar donde se almacena el código fuente. Cada dev. puede tener una copia del mismo sin afectar las otras copias.

Ramas

Permite que cada dev. trabaje en distintas versiones al mismo tiempo. Permite trabajar en nuevas funciones o arreglar errores sin afectar la versión en prod.

Commit

Registro de cambio realizado en el repo. Permite realizar un seguimiento de los cambios realizados en el proyecto.



Conceptos fundamentales

Staging area

Espacio en el que los cambios son preparados y organizados. Suele conocerse como área de preparación.

Repositorio remoto

Copias de repositorios locales que se almacenan en un servidor (GitHub, GitLab, Bitbucket).

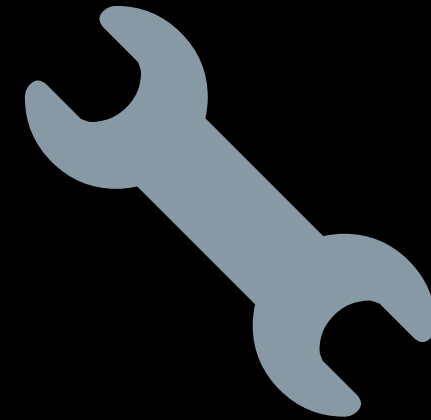
Conflictos de fusión

Ocurre cuando se intenta fusionar dos ramas que han cambiado el mismo archivo. Muchas veces GIT nos resuelve el problema automáticamente pero en otras, necesitaremos hacerlo manualmente.

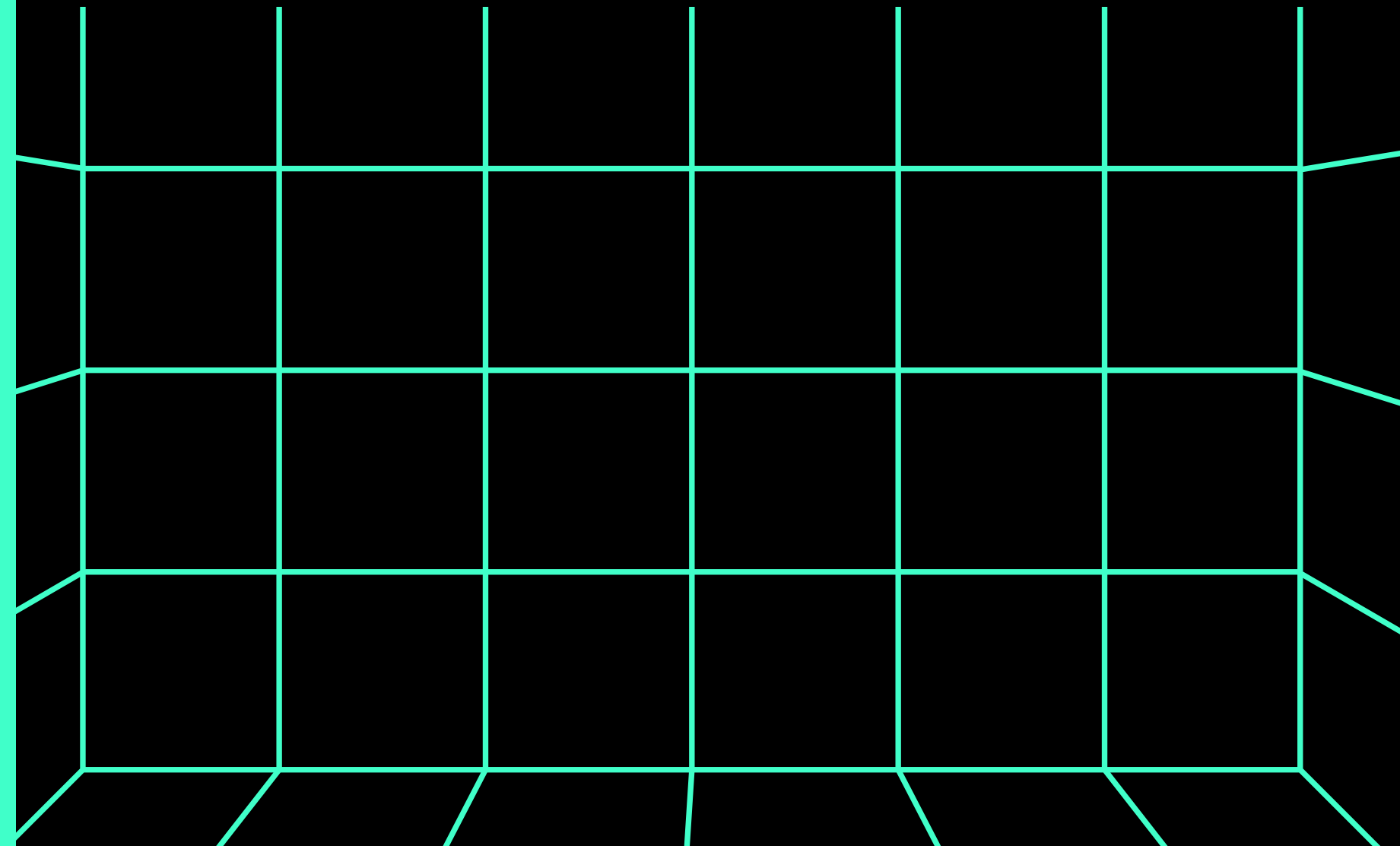




Configuración



Primeros pasos



Configuración



- ✓ Descargar del sitio oficial. (<https://git-scm.com>)
- ✓ Proceder a la instalación. (next > next > ... > finish)
- ✓ Ejecutar "*Git Bash*" (consola de git).
- ✓ Configuración global (git config --global)
- ✓ Iniciar un proyecto.

Pueden encontrar toda la información en la documentación oficial de GIT

1. Descarga



The screenshot shows the Git website homepage. At the top left is the Git logo (a red diamond with a white branching diagram) followed by the text "git --distributed-is-the-new-centralized". To the right is a search bar with the placeholder text "Search entire site..." and a close button (X). Below the header, there are two paragraphs of text. The first paragraph states: "Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency." The second paragraph states: "Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**." To the right of the text is a diagram showing a network of stacks of papers connected by colored lines (red, blue, yellow), representing a distributed version control system. Below the text and diagram, there are four sections with icons and text: "About" (gear icon) with the text "The advantages of Git compared to other source control systems.", "Documentation" (book icon) with the text "Command reference pages, Pro Git book content, videos and other material.", "Downloads" (downward arrow icon) with the text "GUI clients and binary releases for all major platforms.", and "Community" (speech bubble icon) with the text "Get involved! Bug reporting, mailing list, chat, development and more." On the right side, there is a monitor displaying the "Latest source Release" as "2.39.2" with a link to "Release Notes (2023-02-06)" and a button labeled "Download for Windows".

git --distributed-is-the-new-centralized

Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

About
The advantages of Git compared to other source control systems.

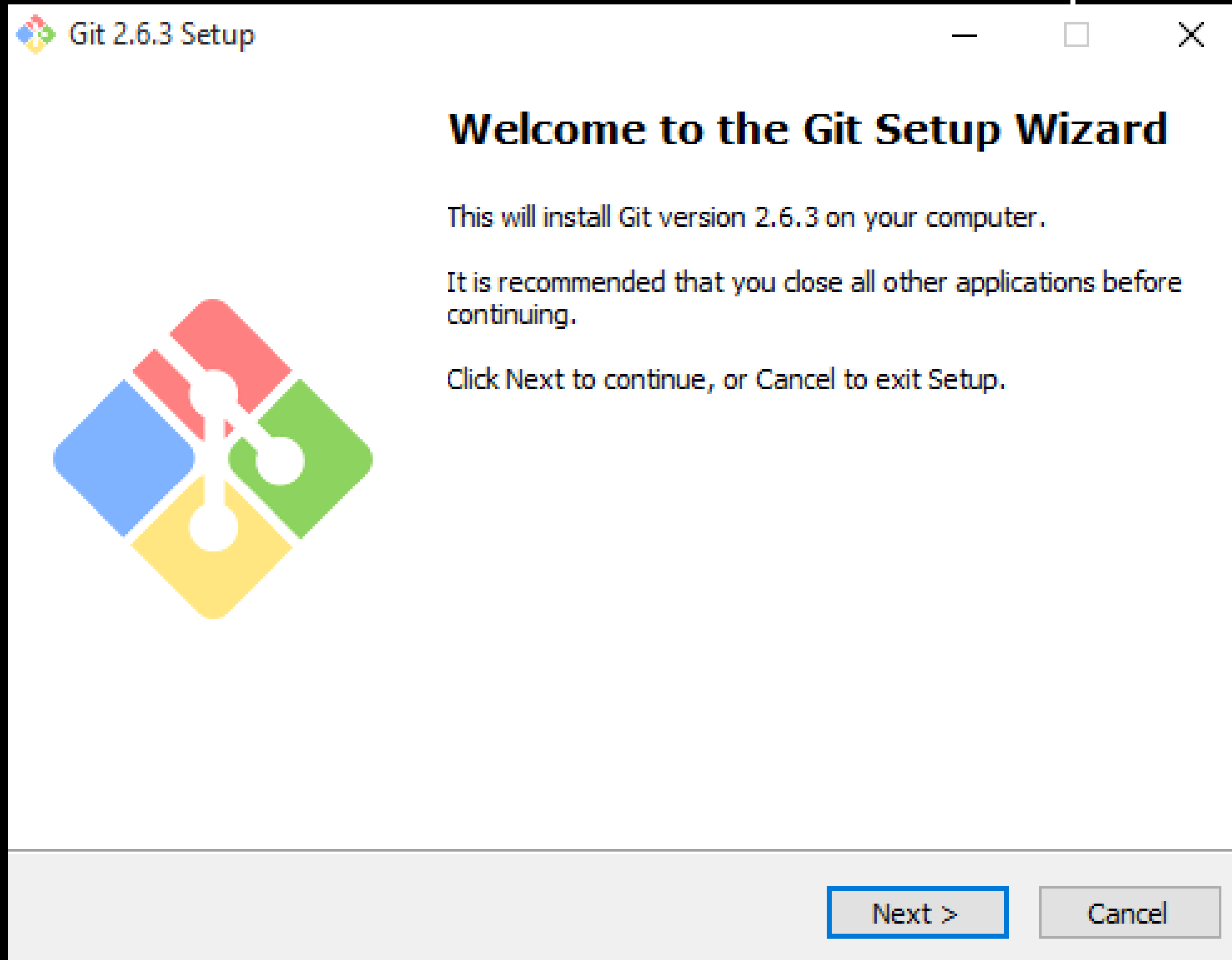
Documentation
Command reference pages, Pro Git book content, videos and other material.

Downloads
GUI clients and binary releases for all major platforms.

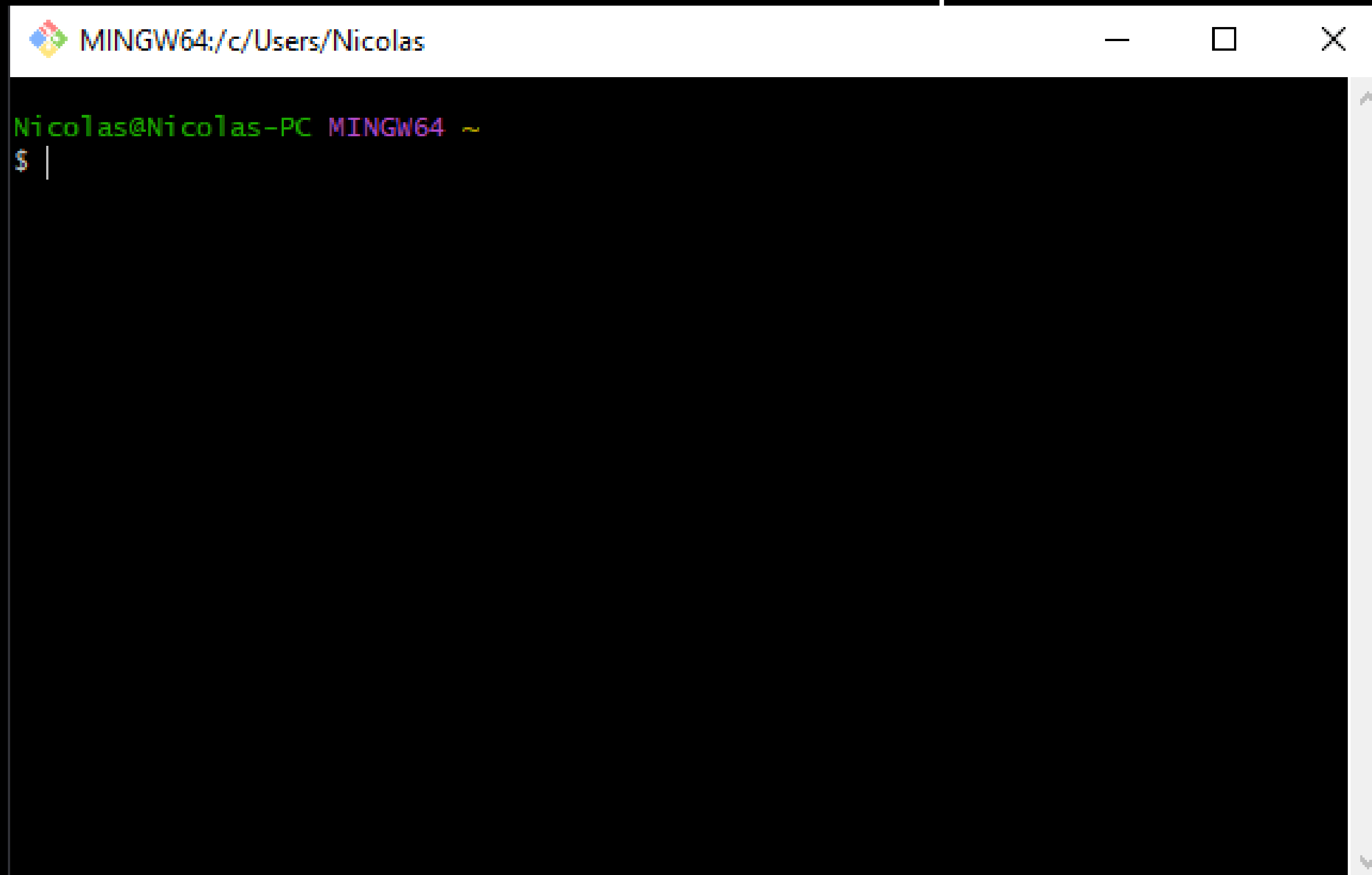
Community
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.39.2
[Release Notes \(2023-02-06\)](#)
Download for Windows

2. Instalación



3. Ejecutar



A screenshot of a MINGW64 terminal window. The title bar at the top reads "MINGW64:/c/Users/Nicolas" and includes standard window controls (minimize, maximize, close). The terminal content shows the prompt "Nicolas@Nicolas-PC MINGW64 ~" followed by a new line starting with "\$ |". A vertical scrollbar is visible on the right side of the terminal window.

```
MINGW64:/c/Users/Nicolas  
  
Nicolas@Nicolas-PC MINGW64 ~  
$ |
```

4. Identidad

```
git config --global user.name "ROTNIC"
```

```
git config --global user.email "rotilinicolas@gmail.com"
```

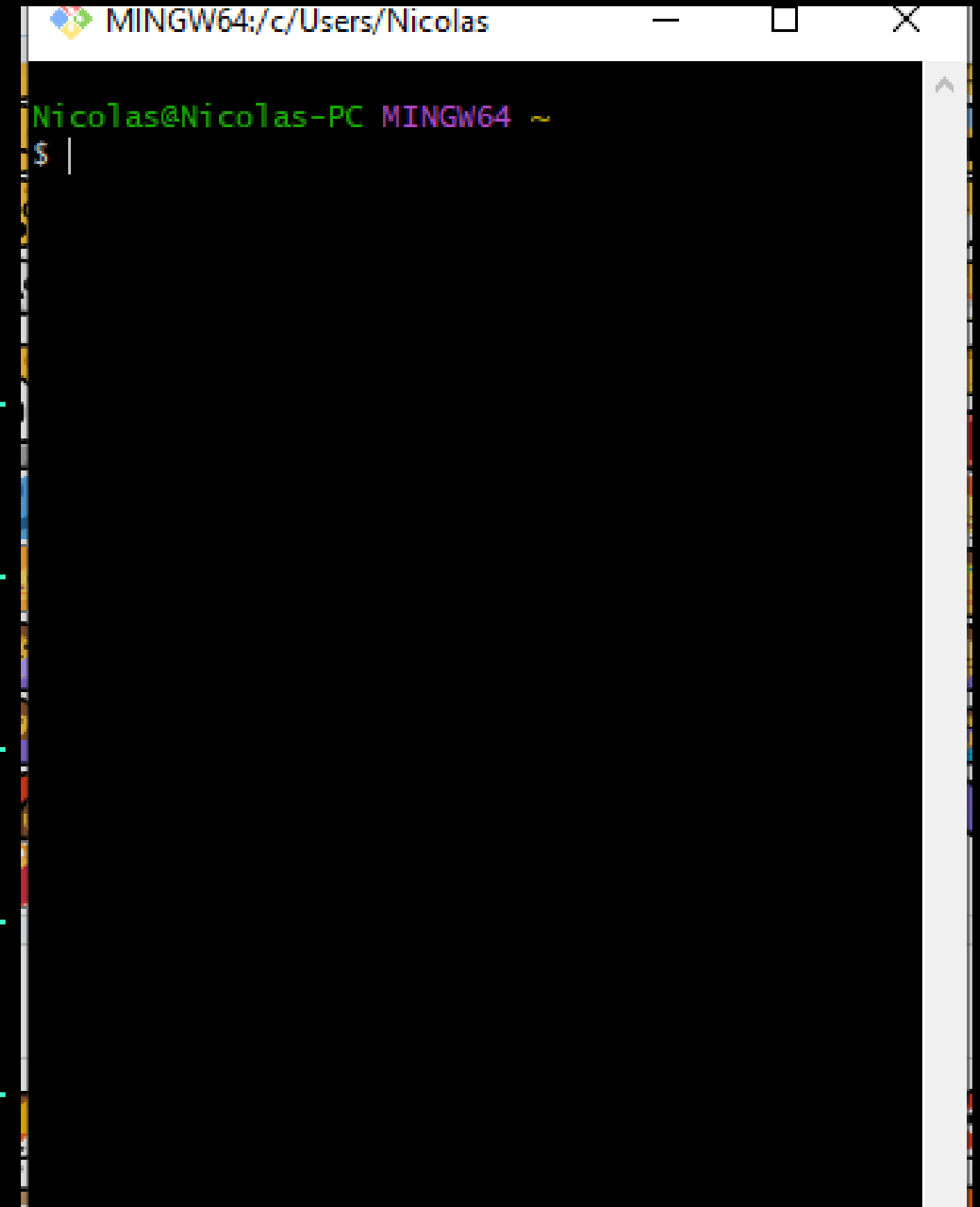
```
git config --list
```

```
git config --global user.name
```

```
git config --global user.name "ROTNIC"
```

CONFIGURACIÓN ESPECÍFICA

Todos los comandos inician anteponiendo la palabra reservada GIT



The screenshot shows a terminal window titled "MINGW64: c:/Users/Nicolas". The prompt is "Nicolas@Nicolas-PC MINGW64 ~" followed by a dollar sign and a cursor. The terminal is decorated with a vertical bar on the left and a vertical bar on the right, both composed of small colored squares.

Comandos básicos



Inicia un proyecto en la carpeta actual < **git init**
Envía los archivos a staging area < **git add**
Confirma los cambios < **git commit**
Muestra el estado actual del repo < **git status**
Muestra historial de confirmaciones < **git log**
Muestra, crea y elimina ramas < **git branch**
Fusiona ramas < **git merge**
Permite cambiar de ramas < **git checkout**

Enviar los cambios al repo remoto < **git push**
Trae los cambios del repo remoto al local < **git pull**



Ejemplo Git

¿Cómo lo usamos?

- Posicionarse en el directorio del proyecto a crear.
 - Iniciamos un proyecto Git.
 - ¿Configuramos datos de identidad?
 - Añadimos un archivo.
 - Add - Commit.
- Creamos nuestra rama de desarrollo.
 - Cambiamos de rama.
- Editamos archivo o creamos nuevos.
 - Fusionamos con rama principal.



Ejercitación



1. Crea un nuevo repositorio en tu computadora.
2. Crea un archivo llamado "index.html" y agrega algo de contenido en él.
3. Agrega el archivo "index.html" al área de preparación.
4. Confirma los cambios en el archivo "index.html".
5. Crea una nueva rama llamada "nueva-rama".
6. Cambia a la nueva rama.
7. Crea un archivo llamado "style.css" en la nueva rama y agrega algo de contenido en él.
8. Agrega el archivo "style.css" al área de preparación.
9. Confirma los cambios en el archivo "style.css".
10. Cambia de nuevo a la rama principal.
11. Fusiona la nueva rama a la rama principal.
12. Agrega un archivo llamado "script.js" y agrega contenido en él.
13. Agrega el archivo "script.js" al área de preparación.
14. Confirma los cambios en el archivo "script.js".



Gracias

¿Dudas?

