

# Clase 4 - Introducción a la terminal

# Índice

- **Visual Studio Code:** Concepto, partes, atajos, instalación, extensiones
- **GitBash:** Concepto e Instalación
- **La terminal:** Concepto. Que podemos hacer con ella. Diferencias en windows, Mac y linux. Diferencia entre CLI y CUI
- **Movernos entre carpetas:** Listado de comandos.
- **Usando la terminal de mi S.O:** Demostración.
- **Powershell:** Concepto y funcionalidad.
- **¿Qué es Input y Output?**



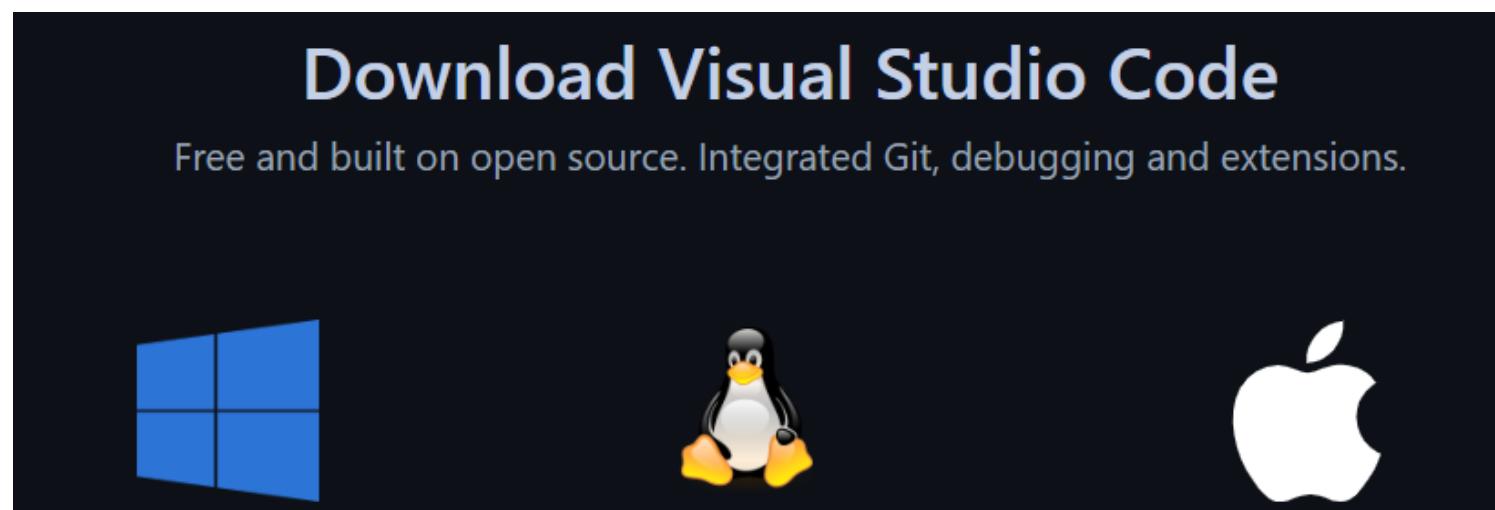
# Visual Studio Code

# ¿Qué es Visual Studio Code?

- Visual Studio Code o VSCode es un **IDE (entorno de desarrollo integrado)** desarrollado por Microsoft.
- Cuando hablamos de un IDE nos referimos a un conjunto de herramientas diseñadas para facilitarnos la creación y el desarrollo de nuestros programas o aplicaciones.

## Compatibilidad

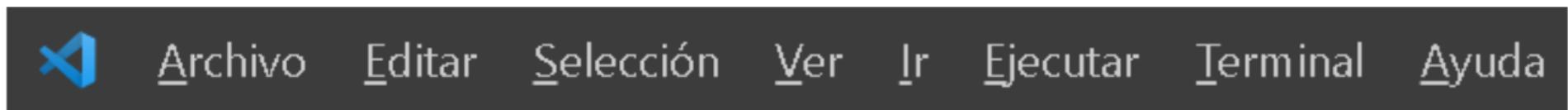
- En el sitio web de Visual Studio Code (<https://code.visualstudio.com/Download>)
- Podemos ver los instaladores para los sistemas operativos más populares del mundo.



# Partes del VSC

## Menú superior

- Nos permite acceder a todas las funcionalidades de VS Code: crear nuevos archivos, guardarlos, edición de nuestro contenido, cambiar vistas, abrir terminales y mucho más.



## Menú lateral (Barra de actividad/Activity bar)

- Nos permite acceder rápidamente a las funcionalidades más utilizadas de VS Code. Repasemos en orden de arriba hacia abajo:



**Explorador (Explorer)**: nos da acceso a visualizar nuestra estructura de carpetas y archivos, y a los archivos que estamos editando (*open editors*).

**Buscar (Search)**: nos permite buscar texto dentro de nuestros archivos, también tiene la función de buscar y reemplazar.

**Control de fuente - versiones (Source control)**: nos va a permitir comparar nuestra versión local con la versión en la nube.

**Ejecutar (Run)**: nos permite correr y debuggear código.

**Extensiones (Extensions)**: nos muestra las extensiones (funcionalidades agregadas) que tenemos instaladas y nos permite buscar e instalar nuevas.

# Partes del VSC

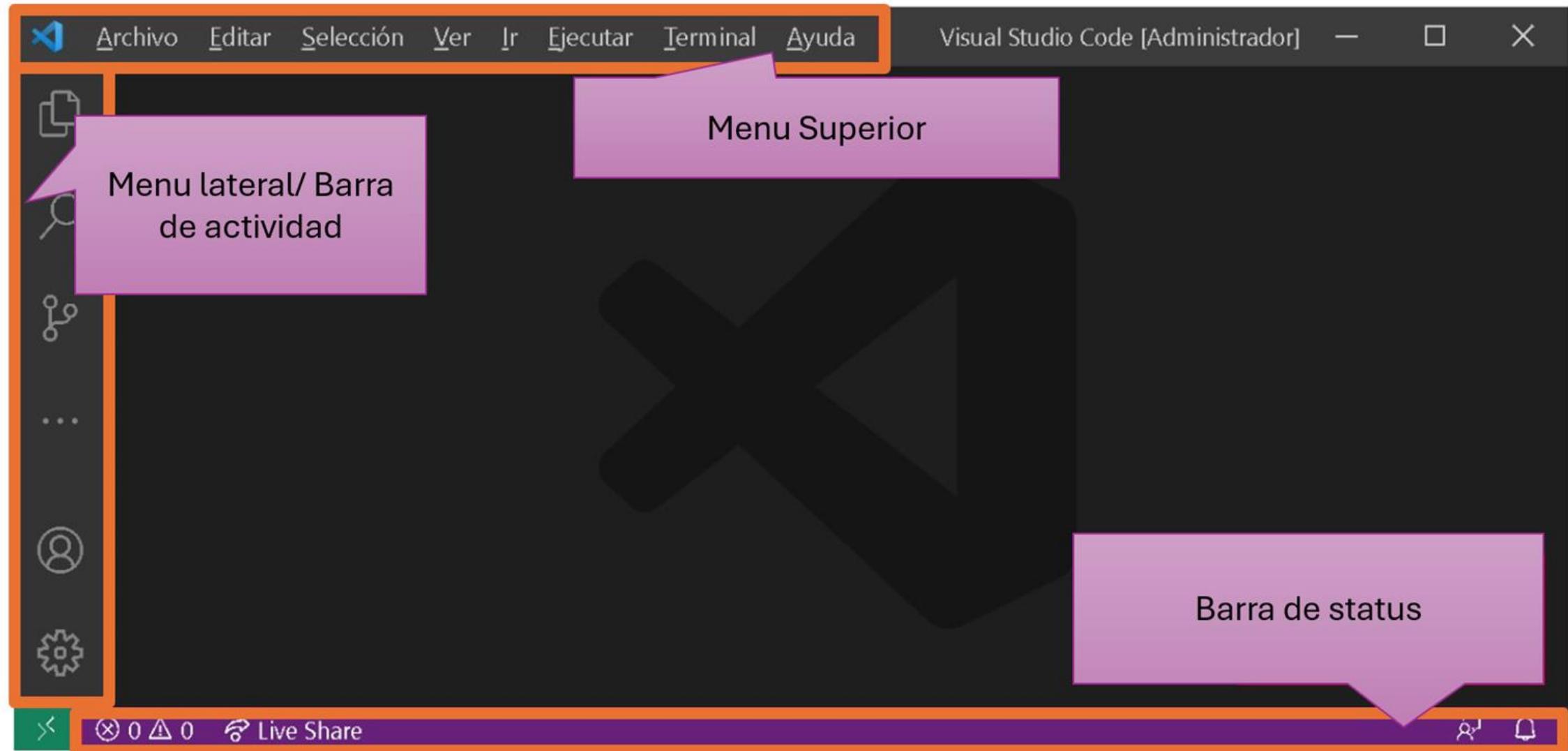
## Barra de estado

- Nos da información del archivo que estamos editando



# Partes del VSC

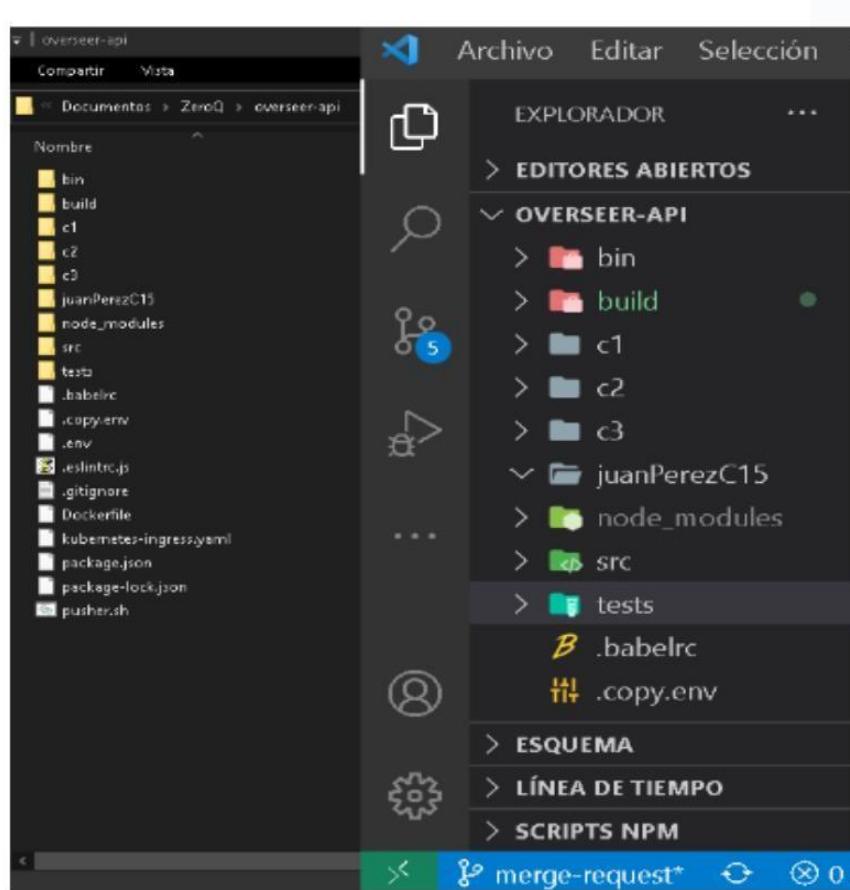
## Entorno



# Partes del VSC

## El explorador y nuestros archivos:

- Nos da información del archivo que estamos editando
- El Visual Studio Code nos muestra la organización de las carpetas de la misma manera que la vemos en el explorador de archivos



# Atajos del VSC



Atajos de teclado para Windows/Linux

## General

<b>Ctrl+Shift+P, F1</b>	Mostrar paleta de comandos
<b>Ctrl+P</b>	Abrir rápido, ir al archivo..
<b>Ctrl+Shift+N</b>	Nueva ventana/instancia
<b>Ctrl+Shift+W</b>	Cerrar ventana/instancia
<b>Ctrl+,</b>	Configuración de usuario
<b>Ctrl+K Ctrl+S</b>	Ver todos los atajos de teclado

## Edición Básica

<b>Ctrl+X</b>	Cortar toda la linea (Sin seleccionar nada)
<b>Ctrl+C</b>	Copiar toda la linea (Sin seleccionar nada)
<b>Alt+↑ / ↓</b>	Mover linea arriba/abajo
<b>Shift+Alt+↑ / ↓</b>	Copiar linea arriba/abajo
<b>Ctrl+Shift+K</b>	Borrar linea
<b>Ctrl+Enter</b>	Insertar linea debajo
<b>Ctrl+Shift+Enter</b>	Insertar linea arriba
<b>Ctrl+Shift+\`</b>	Saltar al corchete de cierre
<b>Ctrl+] / [</b>	Indentar/desindentar linea
<b>Inicio / Fin</b>	Ir al principio/fin de la linea
<b>Ctrl+Inicio</b>	Ir al principio del archivo
<b>Ctrl+Fin</b>	Ir al final del archivo
<b>Ctrl+↑ / ↓</b>	Desplazar la linea hacia arriba/abajo
<b>Alt+PgUp / PgDn</b>	Desplazar la pagina hacia arriba/abajo
<b>Ctrl+K Ctrl+C</b>	Agregar comentario de linea
<b>Ctrl+K Ctrl+U</b>	Eliminar comentario de linea
<b>Ctrl+/</b>	Alternar comentario de linea
<b>Shift+Alt+A</b>	Alternar comentario de bloque
<b>Alt+Z</b>	Alternar envoltorio de palabra

## Navegación

<b>Ctrl+G</b>	Ir a la linea...
<b>Ctrl+P</b>	Ir al archivo...
<b>F8</b>	Ir al error o advertencia siguiente
<b>Shift+F8</b>	Ir al error o advertencia previo
<b>Ctrl+Shift+Tab</b>	Navegar por el historial del grupo de editores
<b>Alt+ ← / →</b>	Ir hacia atrás / adelante

## Buscar y reemplazar

<b>Ctrl+F</b>	Buscar
<b>Ctrl+H</b>	Reemplazar
<b>F3 / Shift+F3</b>	Buscar Siguiente/Anterior
<b>Alt+Enter</b>	Seleccionar todas las ocurrencias de búsqueda
<b>Ctrl+D</b>	Agregar lo seleccionado a la sig. búsqueda
<b>Ctrl+K Ctrl+D</b>	Agregar la último selección a la sig. búsqueda
<b>Alt+C / R / W</b>	Alternar case-sensitive / regex / toda la palabra

## Multi-cursor y selección

<b>Alt+Click</b>	Insertar cursor
<b>Ctrl+Alt+↑ / ↓</b>	Insertar cursor arriba / abajo
<b>Ctrl+U</b>	Deshacer última operación de cursor
<b>Shift+Alt+I</b>	Insertar cursor al final de cada linea seleccionada
<b>Ctrl+L</b>	Seleccionar toda la linea actual
<b>Ctrl+Shift+L</b>	Seleccionar todas las ocurrencias de la selección
<b>Ctrl+F2</b>	Seleccionar ocurrencias de la palabra seleccionada
<b>Shift+Alt+→</b>	Expandir selección
<b>Shift+Alt+←</b>	Contraer selección

## Edición de lenguaje enriquecido

<b>Ctrl+Espacio</b>	Activar sugerencias
<b>Ctrl+Shift+Espacio</b>	Activar pistas de parámetros
<b>Shift+Alt+F</b>	Formatear documento
<b>Ctrl+K Ctrl+F</b>	Formatear selección
<b>F12</b>	Ir a la Definición de la función
<b>Alt+F12</b>	Ver la Definición de la función
<b>Ctrl+K F12</b>	Abrir Definición a un costado
<b>Ctrl+.</b>	Solución Rápida
<b>Shift+F12</b>	Mostrar referencias
<b>F2</b>	Renombrar
<b>Ctrl+K Ctrl+X</b>	Recortar espacios en blanco al final de página
<b>Ctrl+K M</b>	Cambiar lenguaje del archivo

## Gestión del editor

<b>Ctrl+F4, Ctrl+W</b>	Cerrar editor
<b>Ctrl+K F</b>	Cerrar carpeta
<b>Ctrl+`</b>	Dividir editor
<b>Ctrl+1 / 2 / 3</b>	Enfocar al 1°, 2° o 3° grupo editor
<b>Ctrl+K Ctrl+←/→</b>	Enfocar al anterior/siguiente grupo editor
<b>Ctrl+Shift+PgUp/PgDn</b>	Mover editor a la izquierda/derecha
<b>Ctrl+K ← / →</b>	Mover el grupo editor activo

## Administración de archivos

<b>Ctrl+N</b>	Nuevo Archivo
<b>Ctrl+O</b>	Abrir Archivo...
<b>Ctrl+S</b>	Guardar
<b>Ctrl+Shift+S</b>	Guardar Como...
<b>Ctrl+K S</b>	Guardar Todo
<b>Ctrl+F4</b>	Cerrar
<b>Ctrl+K Ctrl+W</b>	Cerrar Todo
<b>Ctrl+Shift+T</b>	Reabrir editor cerrado
<b>Ctrl+K Enter</b>	Mantener la vista previa del editor abierta
<b>Ctrl+Tab</b>	Cambiar a la pestaña siguiente
<b>Ctrl+Shift+Tab</b>	Cambiar a la pestaña anterior
<b>Ctrl+K P</b>	Copiar la ruta del archivo activo
<b>Ctrl+K R</b>	Abrir el archivo activo en el explorador del S.O.
<b>Ctrl+K O</b>	Abrir el archivo activo en una nuevo VSCode

## Mostrar

<b>F11</b>	Cambiar a pantalla completa
<b>Shift+Alt+0</b>	Altemar interfaz del editor (horizontal/vertical)
<b>Ctrl+ = / -</b>	Acercar/alejar zoom
<b>Ctrl+B</b>	Altemar visibilidad de la barra lateral
<b>Ctrl+Shift+E</b>	Mostrar panel de Explorador
<b>Ctrl+Shift+F</b>	Mostrar panel de Búsqueda
<b>Ctrl+Shift+G</b>	Mostrar panel de Control de código(versionado)
<b>Ctrl+Shift+D</b>	Mostrar panel de Ejecución y Depuración
<b>Ctrl+Shift+X</b>	Mostrar panel de Extensiones
<b>Ctrl+Shift+H</b>	Reemplazar palabras en archivos
<b>Ctrl+Shift+J</b>	Altemar muestra de detalles de búsqueda
<b>Ctrl+Shift+U</b>	Mostrar Panel de Salida
<b>Ctrl+Shift+V</b>	Abrir vista previa de MarkDown (archivos .md)
<b>Ctrl+K V</b>	Previsualización al costado de MarkDown (.md)
<b>Ctrl+K Z</b>	Modo Zen (Esc Esc para salir)

## Terminal integrada

<b>Ctrl+`</b>	Mostrar terminal integrada
<b>Ctrl+Shift+`</b>	Crear nueva terminal
<b>Ctrl+C</b>	Copiar selección
<b>Ctrl+V</b>	Pegar en la terminal activa
<b>Ctrl+↑ / ↓</b>	Desplazar arriba/abajo
<b>Shift+PgUp / PgDn</b>	Desplazar pagina arriba/abajo
<b>Ctrl+Inicio / Fin</b>	Desplazar hacia el inicio/final

# Instalación de VSC

## Explicación Paso por Paso:

Paso 1: Ve a la página de Microsoft Visual Studio Code y haz clic en el botón 'Descargar Visual Studio Code' para descargar el archivo de instalación.



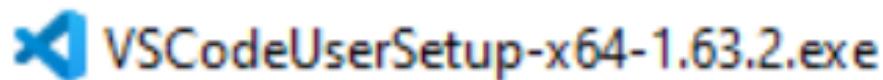
The screenshot shows the official Visual Studio Code download page. At the top, there's a navigation bar with links to Docs, Updates, Blog, API, Extensions, FAQ, GitHub Copilot, and a search bar. On the right side of the header is a 'Download' button. The main content area has a dark background with white text. It features three large icons: a Windows logo, a Linux Tux penguin, and an Apple logo. Below each icon are download links: 'Windows' (Windows 10, 11), '.deb' (Debian, Ubuntu), '.rpm' (Red Hat, Fedora, SUSE), and 'Mac' (macOS 10.15+). Underneath these, there are more detailed download options for different platforms and architectures, such as 'User Installer' (x64, Arm64), 'System Installer' (x64, Arm64), '.zip' (x64, Arm64), 'CLI' (x64, Arm64), '.deb' (x64, Arm32, Arm64), '.rpm' (x64, Arm32, Arm64), '.tar.gz' (x64, Arm32, Arm64), 'Snap' (Snap Store), and 'CLI' (x64, Arm32, Arm64). There are also links for '.zip' (Intel chip, Apple silicon, Universal) and 'CLI' (Intel chip, Apple silicon).

- Link:  
<https://code.visualstudio.com/Download>

# Instalación de vsc - Windows

## Explicación Paso por Paso:

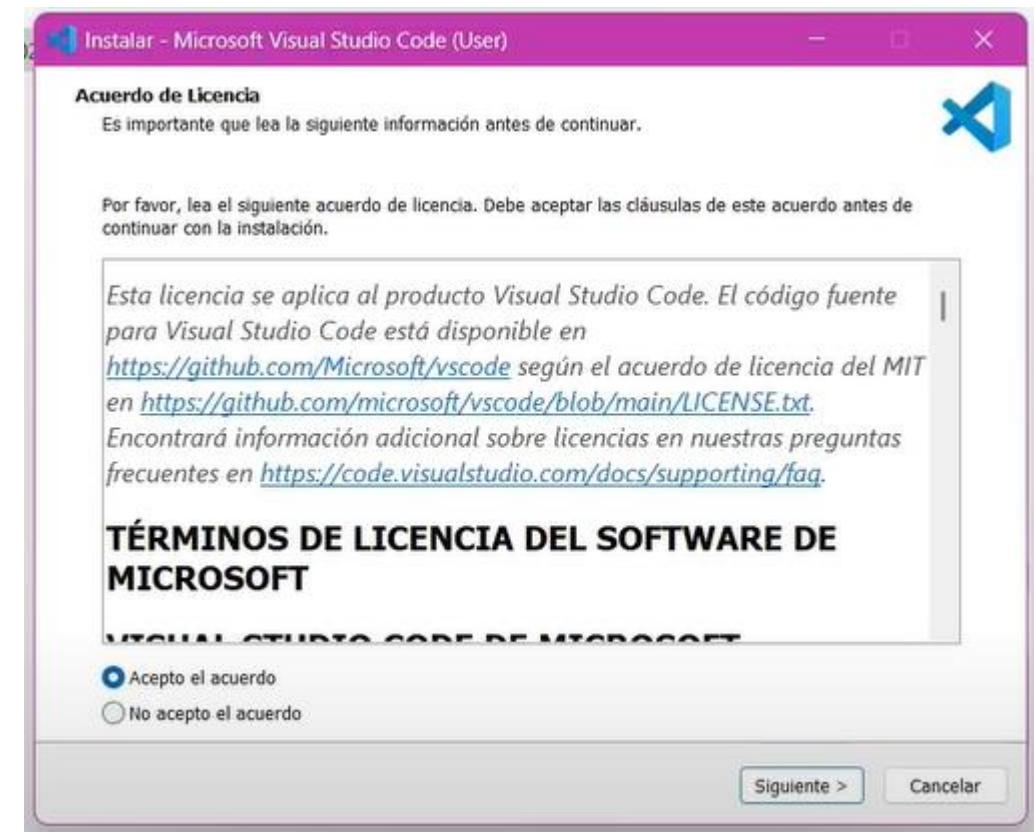
Paso 2: Abre el archivo de instalación .exe en tu carpeta de descargas para iniciar la instalación.



# Instalación de VSC - Windows

## Explicación Paso por Paso:

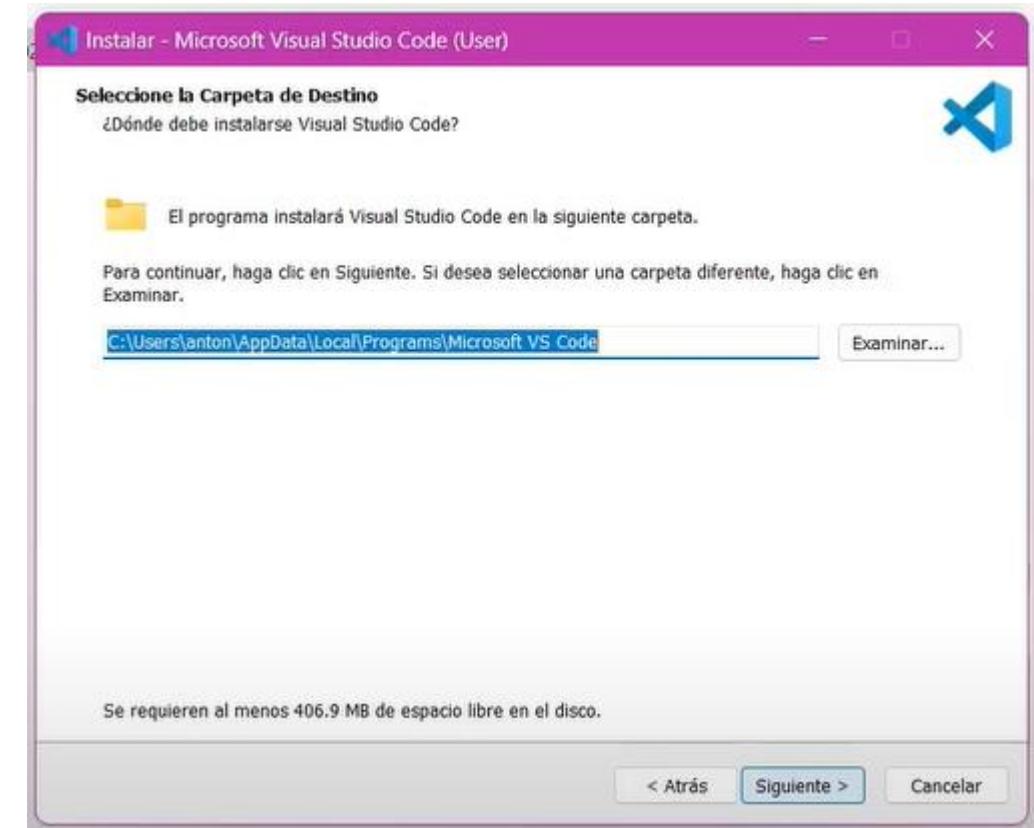
Paso 3: Lee y acepta el acuerdo de licencia.  
Haz clic en Next para continuar.



# Instalación de VSC - Windows

## Explicación Paso por Paso:

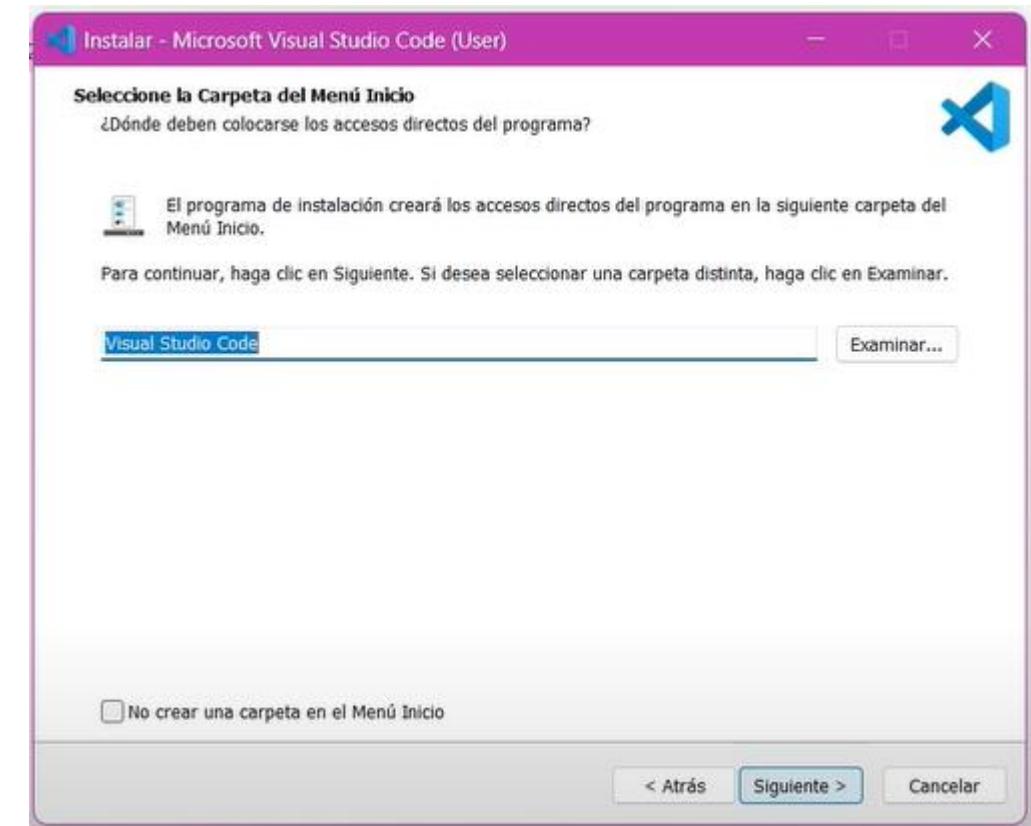
**Paso 4:** Puedes cambiar la ubicación de la carpeta de instalación o mantener la configuración predeterminada. Haz clic en Next para continuar.



# Instalación de VSC - Windows

## Explicación Paso por Paso:

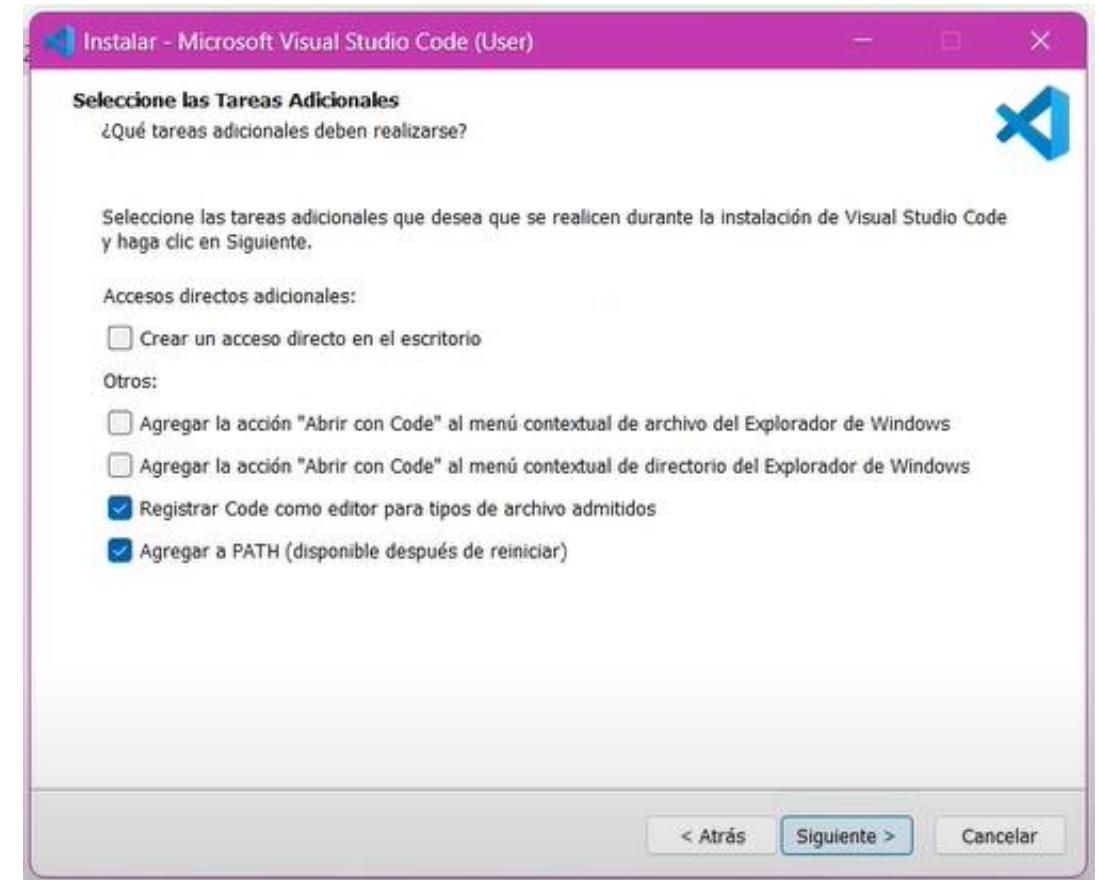
**Paso 5:** Elige si deseas cambiar el nombre de la carpeta de accesos directos en el menú Inicio o si no deseas instalar accesos directos en absoluto. Haz clic en Next.



# Instalación de vsc - Windows

## Explicación Paso por Paso:

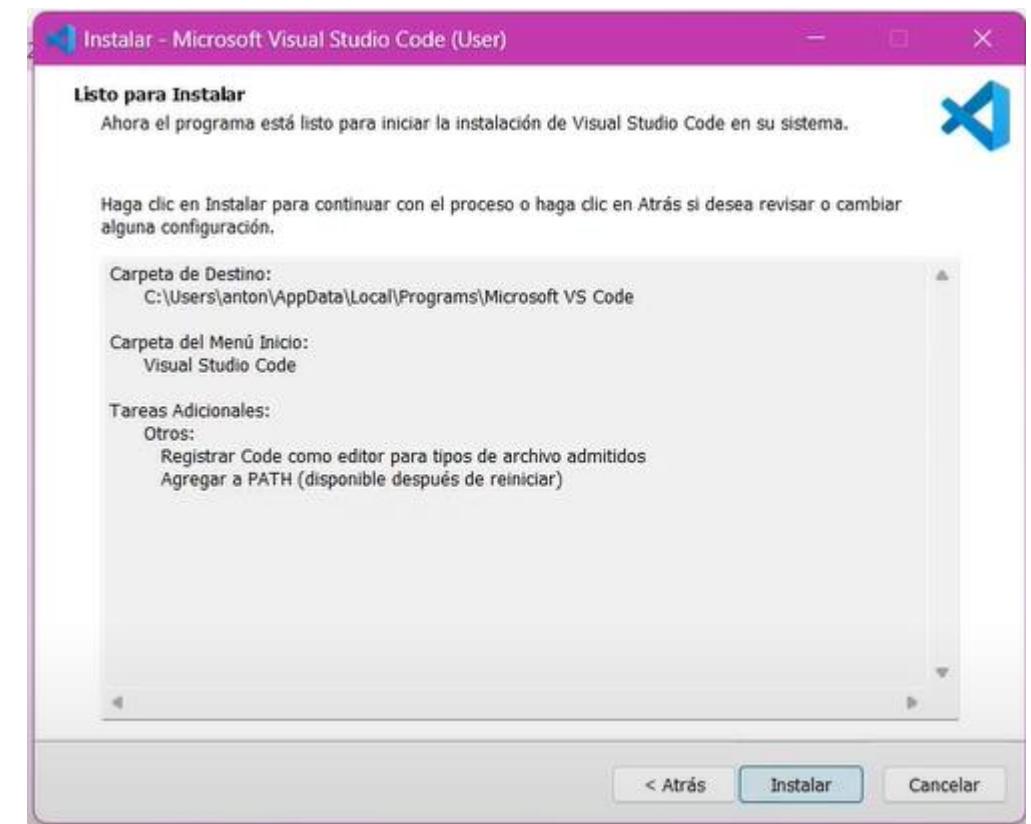
Paso 6: Selecciona las tareas adicionales, por ej. crear un ícono en el escritorio o añadir opciones al menú contextual de Windows Explorer. Haz clic en Next.



# Instalación de vsc - Windows

## Explicación Paso por Paso:

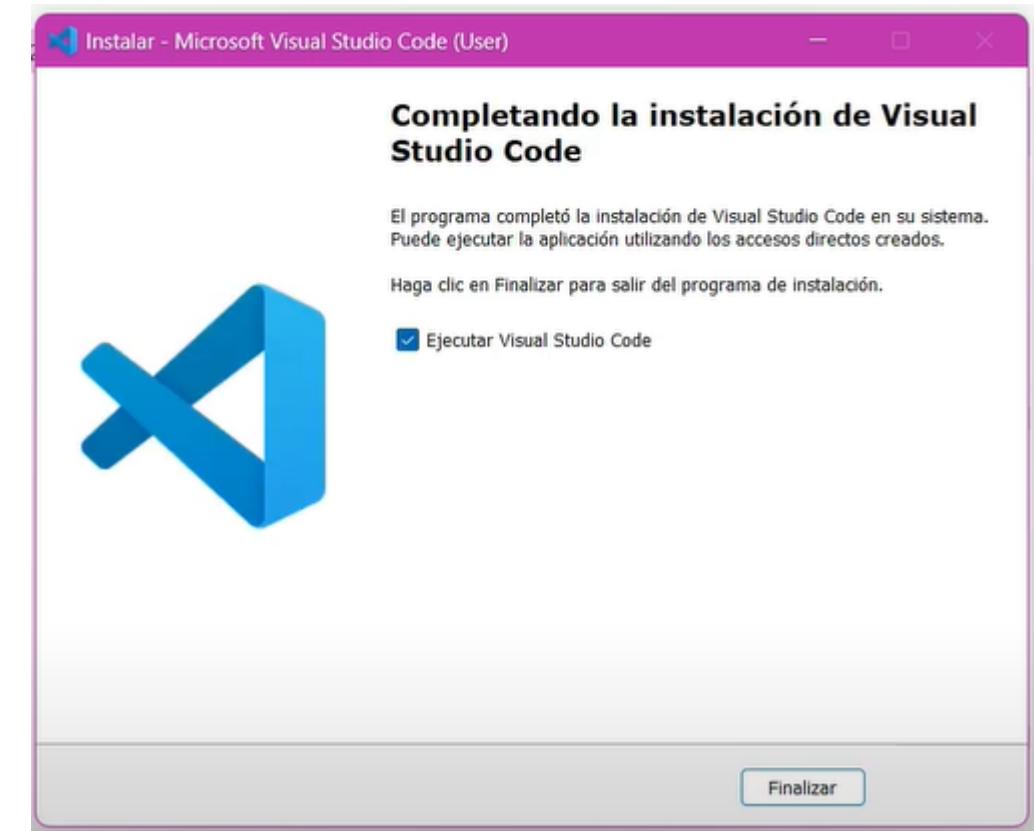
Paso 7: Haz clic en Install para iniciar la instalación.



# Instalación de VSC - Windows

## Explicación Paso por Paso:

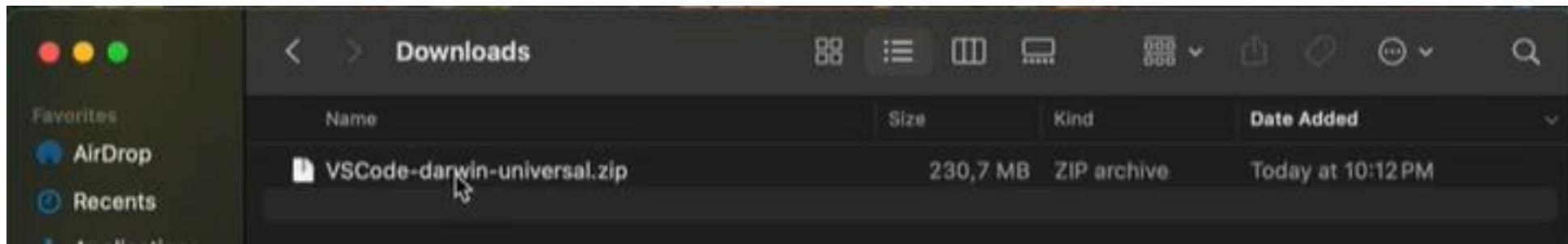
**Paso 8:** El programa está instalado y listo para usar. Haz clic en Finish para finalizar la instalación y lanzar el programa.



# Instalación de VSC – Mac OS

## Explicación Paso por Paso:

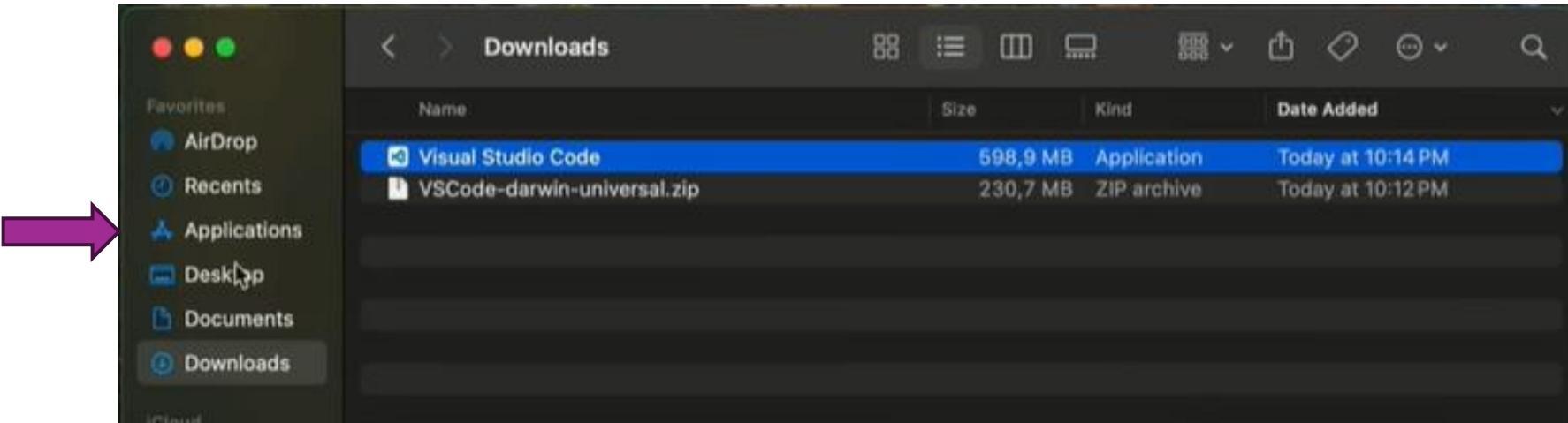
Paso 2: Abre el explorador de archivos y haz doble click en el archivo .zip de VSCode.



# Instalación de VSC – Mac OS

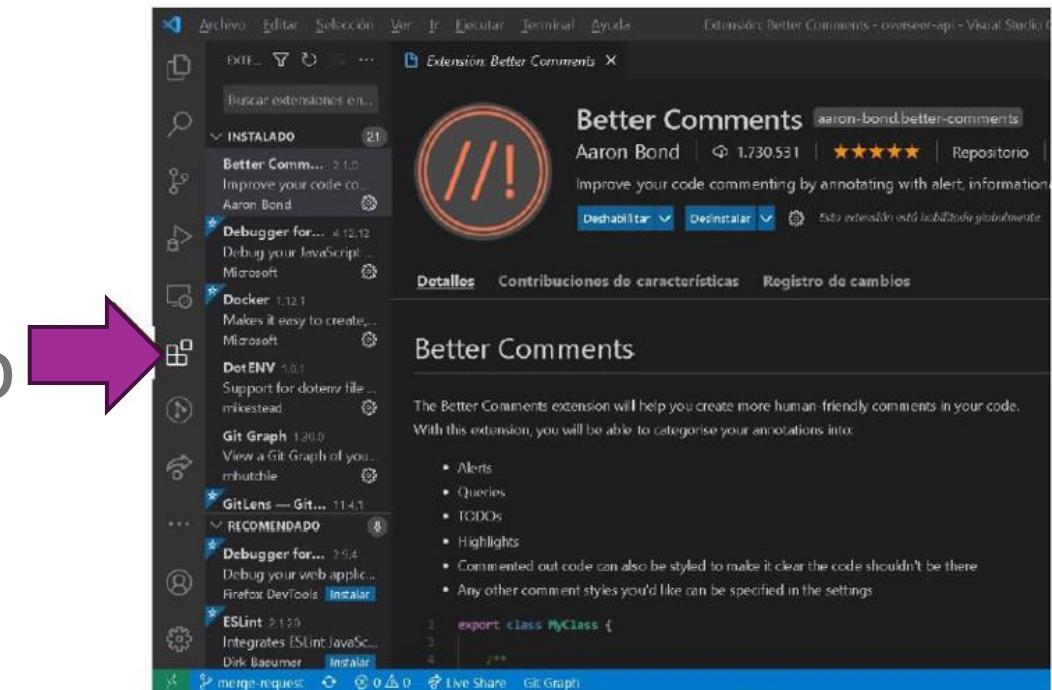
## Explicación Paso por Paso:

Paso 3: Una vez extraído el programa, debes moverlo a la carpeta de Aplicaciones.



# Extensiones de VSC

Una extensión de Visual Studio Code (VSCode) es un complemento o "plugin" que agrega funcionalidades adicionales al editor de código. Estas extensiones permiten personalizar y mejorar el entorno de desarrollo según tus necesidades y el tipo de lenguaje o proyecto con el que estés trabajando.



# Extensiones de VSC

## Explicación Paso por Paso:

Antes de sumergirnos en las extensiones específicas, es importante comprender cómo instalarlas. El proceso es simple:

1. Abre VSCode.
2. Ve a la pestaña «Extensiones» en la barra lateral o presiona Ctrl+Shift+X (Cmd+Shift+X en macOS).
3. Busca la extensión que deseas instalar.
4. Haz clic en «Instalar» junto a la extensión deseada.

# Extensiones de VSC

## Extensiones recomendadas:

### ESLint

Una de las herramientas más útiles es ESLint, un linter que nos permite encontrar problemas en el código JavaScript y arreglarlos de manera automática en muchos de los casos.

### Prettier

Junto con el mencionado ESLint, Prettier también es esencial para el desarrollo con JavaScript. Esta extensión permite formatear el código JavaScript para que se facilite la lectura, pero sobre todo para que los desarrolladores de un proyecto apliquen un mismo estilo en su codificación.

### npm Intellisense

Esta extensión es útil para npm, que te permitirá autocompletar el código cuando estás haciendo un import. Es muy útil para escribir menos y, sobre todo, para no equivocarnos al cargar las librerías de terceros.

### npm

Esta extensión también te facilita el trabajo con npm, pero se centra más en el archivo package.json, ayudando con la revisión de sintaxis y detectando posibles problemas de instalación de dependencias o con sus versiones. También nos ofrece utilidades para ejecutar los scripts npm.

### JavaScript (ES6) code snippets

Con esta extensión tienes una excelente colección de snippets y, si te acostumbras a usarlos, ahorrarás tiempo y no querrás estar sin ellos.

### Import Cost

Esta extensión te permite saber qué ocupa cada una de las librerías que cargas mediante los imports, lo que es muy importante sobre todo para el desarrollo front-end y para ser conscientes del peso que agregamos al JavaScript con cada import que realizamos.

### Quokka.js

Esta extensión es bastante útil para obtener una ejecución rápida del JavaScript a medida que estamos escribiendo el código. Permite ver con anotaciones en el propio editor el resultado de la ejecución de las sentencias o bucles. Es interesante para poder detectar posibles errores en tiempo de código a medida que vas escribiendo, ya sea de sintaxis pero sobre todo de tiempo de ejecución, lo que es todavía más útil.

# GitBash

# Fundamentos

## ¿Qué es Bash?

BASH (Bourne Again SHell) es un intérprete de comandos utilizado en muchos sistemas Unix y Linux.

Es una herramienta poderosa que permite a los usuarios interactuar con el sistema operativo a través de la línea de comandos, automatizar tareas, y administrar sistemas de manera eficiente.



# Fundamentos

## ¿Por qué aprender BASH?

Conocer BASH te permitirá trabajar eficientemente en entornos Unix/Linux, desarrollar scripts para automatización, y comprender cómo funciona la interacción entre software y sistema operativo a bajo nivel.



# Fundamentos

## Un poco de historia de BASH

Bash fue diseñado por **Stephen Bourne** en 1977 y tuvo su primera aparición en Unix v7.

Doce años más tarde, en 1989 **Brian Fox** desarrolla Bourn-Agail Shell para usarse en el proyecto GNU como intérprete de comandos estándar, y continúa actualizándolo hasta 1993 con la ayuda de **Chet Ramey**, quien creó muchos de los arreglos y nuevas características del programa.

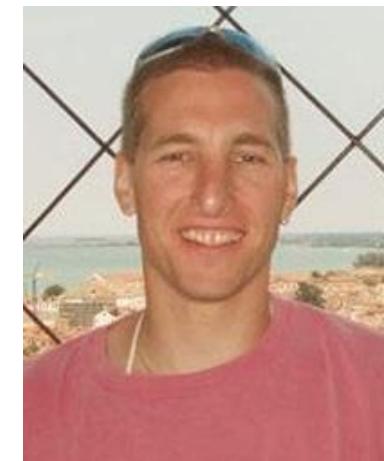
En la actualidad, Bash es el intérprete predeterminado en gran parte de los sistemas GNU/Linux y funciona en la mayoría de sistemas de Unix. Sin embargo, no es el único Shell que existe.



Stephen Bourne



Brian Fox



Chet Ramey

# Git Bash

## ¿Qué es Git Bash?

Git Bash es una aplicación de terminal que se utiliza como interfaz con un sistema operativo mediante comandos escritos (CLI).

Git Bash es básicamente un paquete que instala Bash, algunas utilidades comunes de bash y Git en un sistema operativo Windows

```
MINGW64:/c/Users/singh/Desktop/newRepo
nothing to commit, working tree clean
singh@DESKTOP-PGVSHMF MINGW64 ~/Desktop/newRepo (master)
$ git add .

singh@DESKTOP-PGVSHMF MINGW64 ~/Desktop/newRepo (master)
$ git commit -m "second commit"
[master 9e7f7d0] second commit
 1 file changed, 1 insertion(+)
 create mode 100644 abc/jhvjhbt.txt

singh@DESKTOP-PGVSHMF MINGW64 ~/Desktop/newRepo (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 326 bytes | 65.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To https://github.com/taran910/NewRepo.git
 7a5d54b..9e7f7d0  master -> master

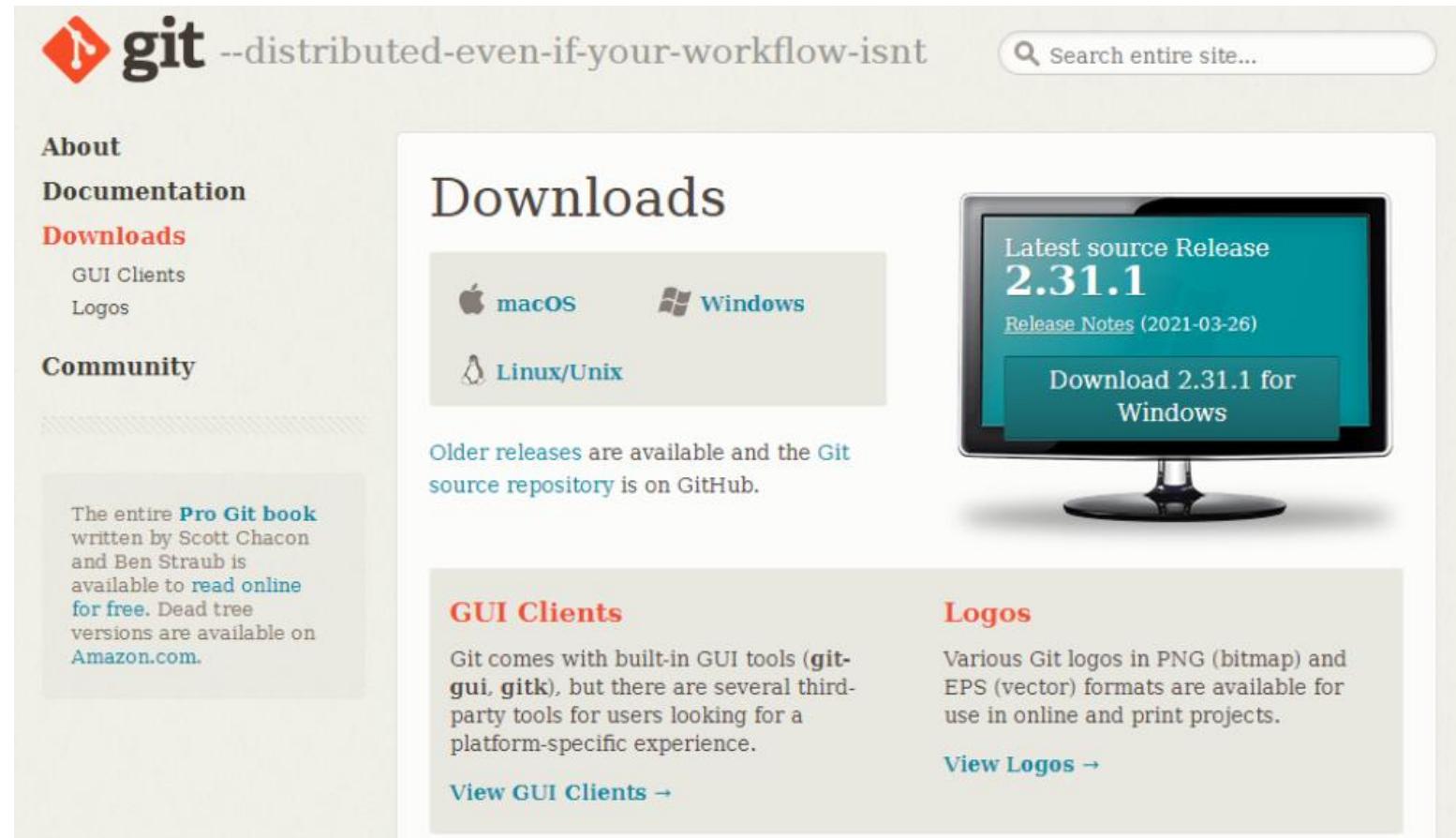
singh@DESKTOP-PGVSHMF MINGW64 ~/Desktop/newRepo (master)
$ |
```



# Git Bash

## Compatibilidad

En el sitio web de git:  
(<https://git-scm.com/downloads>)  
Podemos ver los instaladores para los sistemas operativos más populares del mundo.



The screenshot shows the official Git website at <https://git-scm.com/>. The main navigation menu includes links for About, Documentation, Downloads (which is highlighted in red), and Community. The Downloads section features links for macOS, Windows, and Linux/Unix. A note states that older releases are available and the Git source repository is on GitHub. Below this, a section for GUI Clients explains that Git comes with built-in tools like `git-gui` and `gitk`, and provides a link to view more clients. To the right, a monitor displays a teal window titled "Latest source Release 2.31.1" with a "Download 2.31.1 for Windows" button.

git --distributed-even-if-your-workflow-isnt

Search entire site...

About

Documentation

**Downloads**

GUI Clients

Logos

Community

The entire [Pro Git book](#) written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

## Downloads

macOS Windows

Linux/Unix

Older releases are available and the [Git source repository](#) is on GitHub.

### GUI Clients

Git comes with built-in GUI tools (`git-gui`, `gitk`), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

### Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

# Ahora haremos la instalación paso a paso



# Git Bash - Instalación: Paso por paso Windows

## Control de sistema:

En esta ventana emergente dar clic en “Sí”.



# Git Bash - Instalación: Paso por paso Windows

## Acuerdo de licencia:

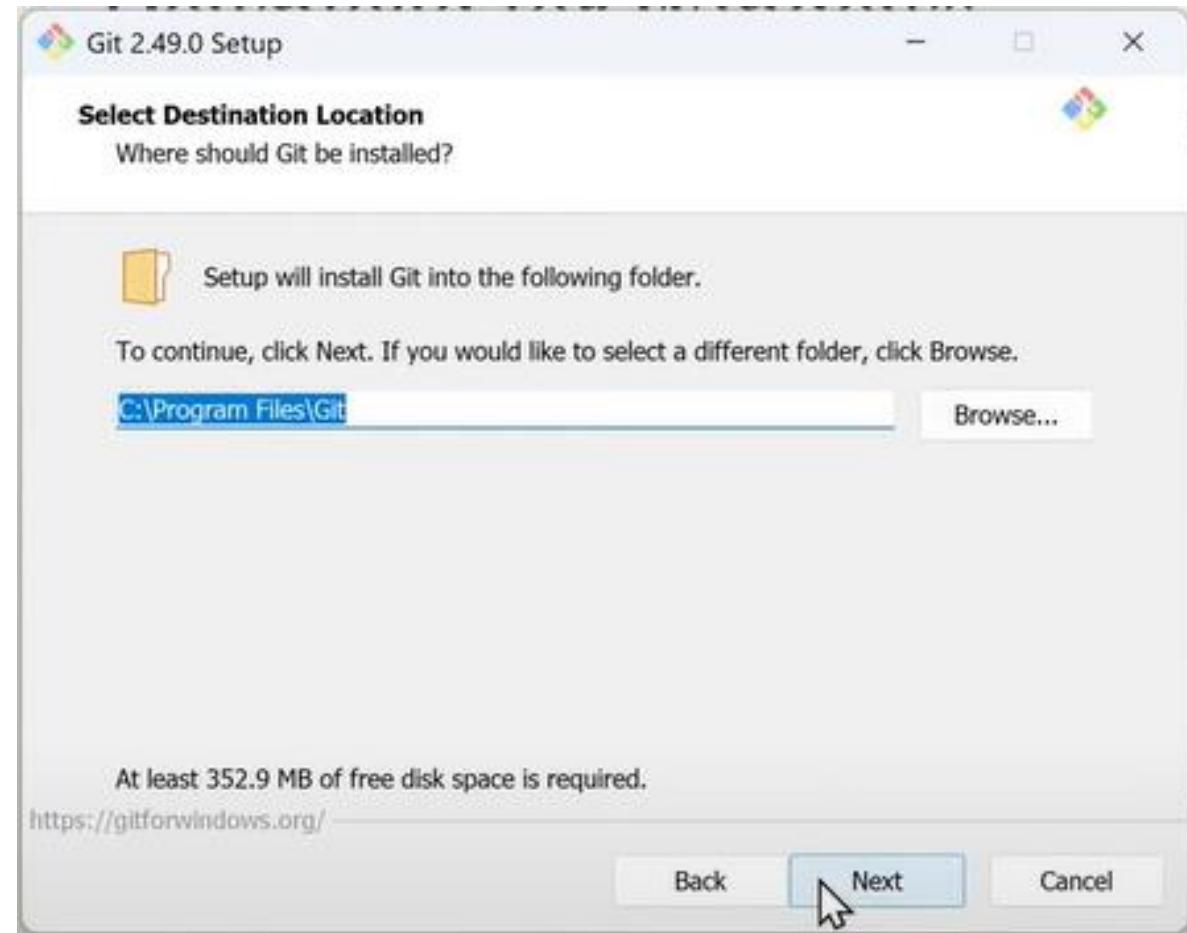
Dar clic en “Next”.



# Git Bash - Instalación: Paso por paso Windows

## Carpeta de destino:

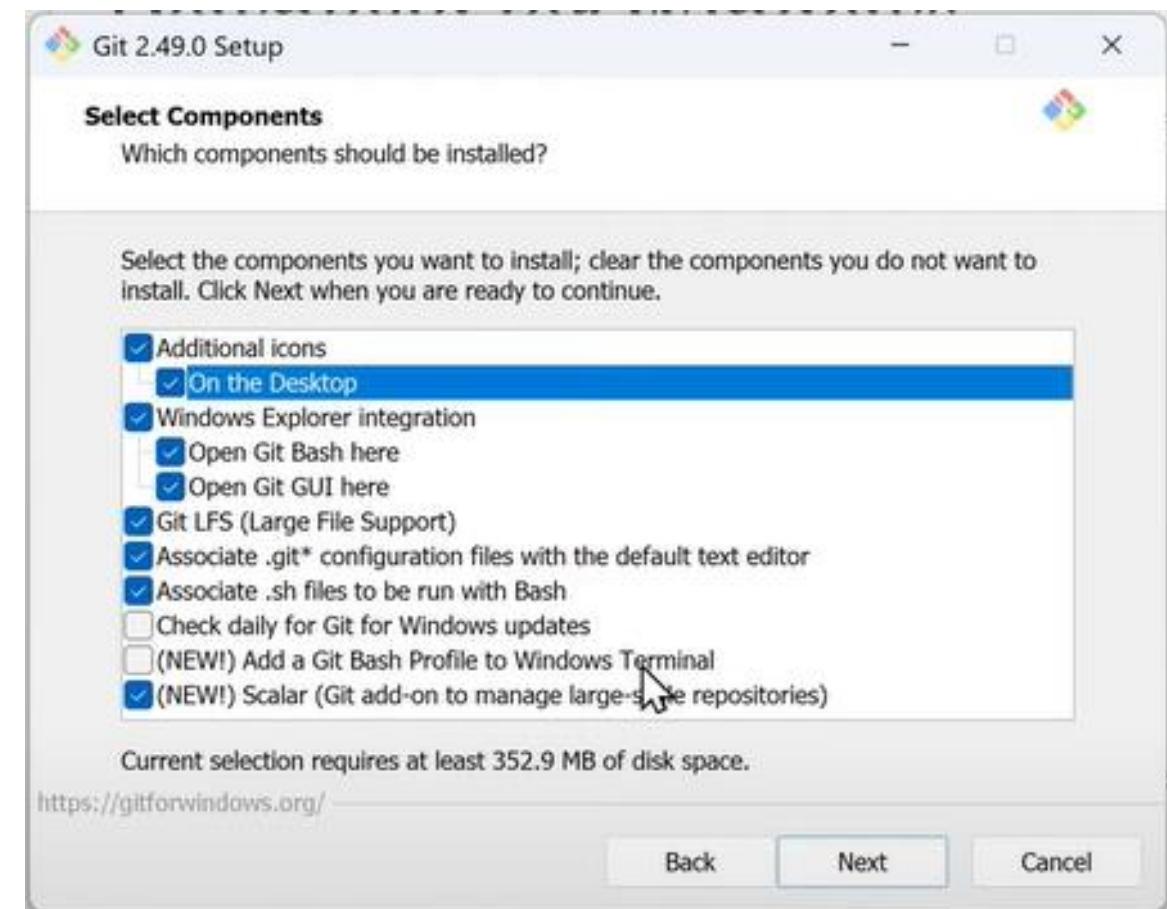
Puedes cambiar la ubicación de la carpeta de instalación o mantener la configuración predeterminada. Haz clic en Next para continuar.



# Git Bash - Instalación: Paso por paso Windows

## Componentes de la Instalación

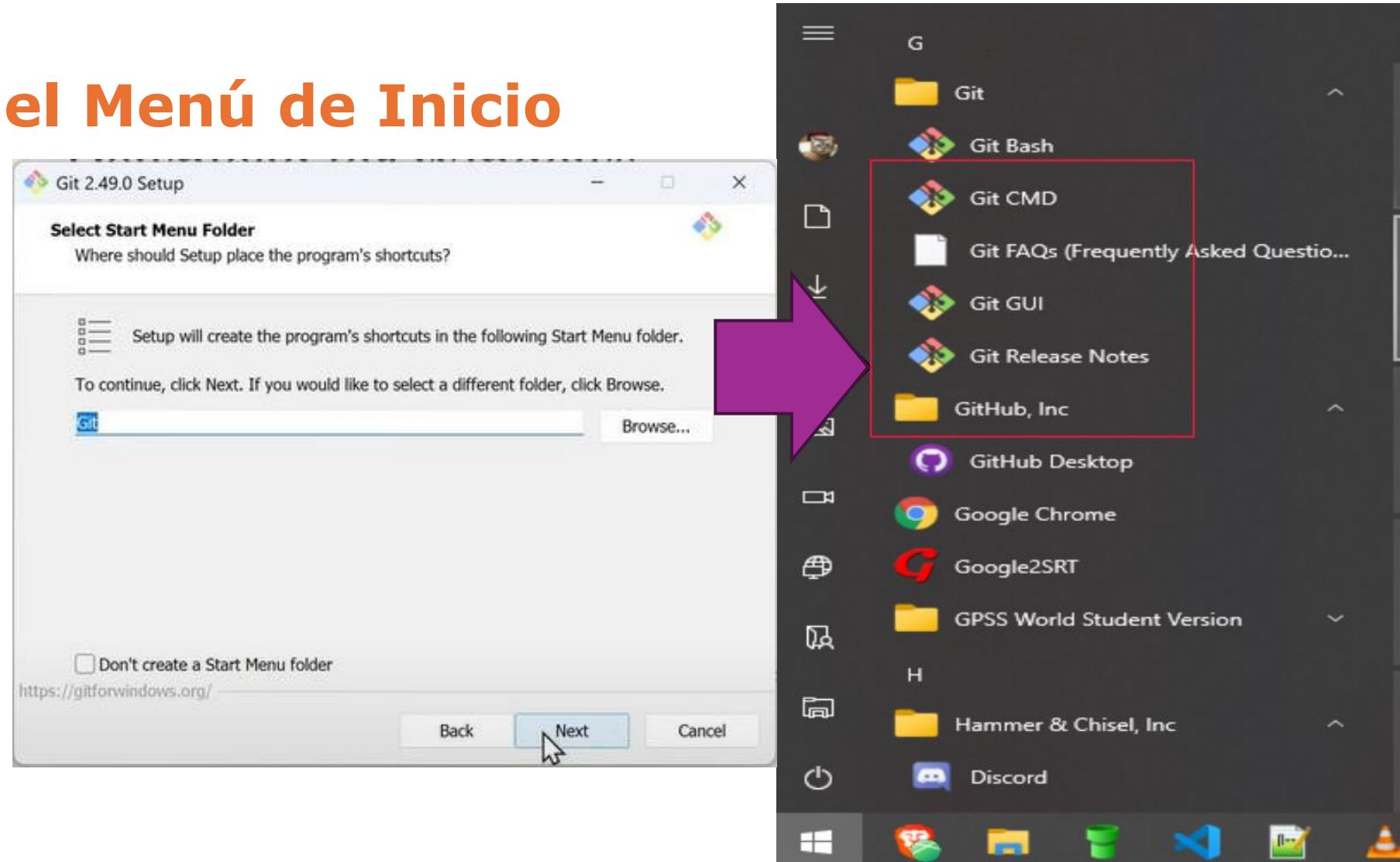
En esta ventana, no es necesario modificar nada, podemos continuar con las opciones marcadas por defecto.



# Git Bash - Instalación: Paso por paso Windows

## Directorio en el Menú de Inicio

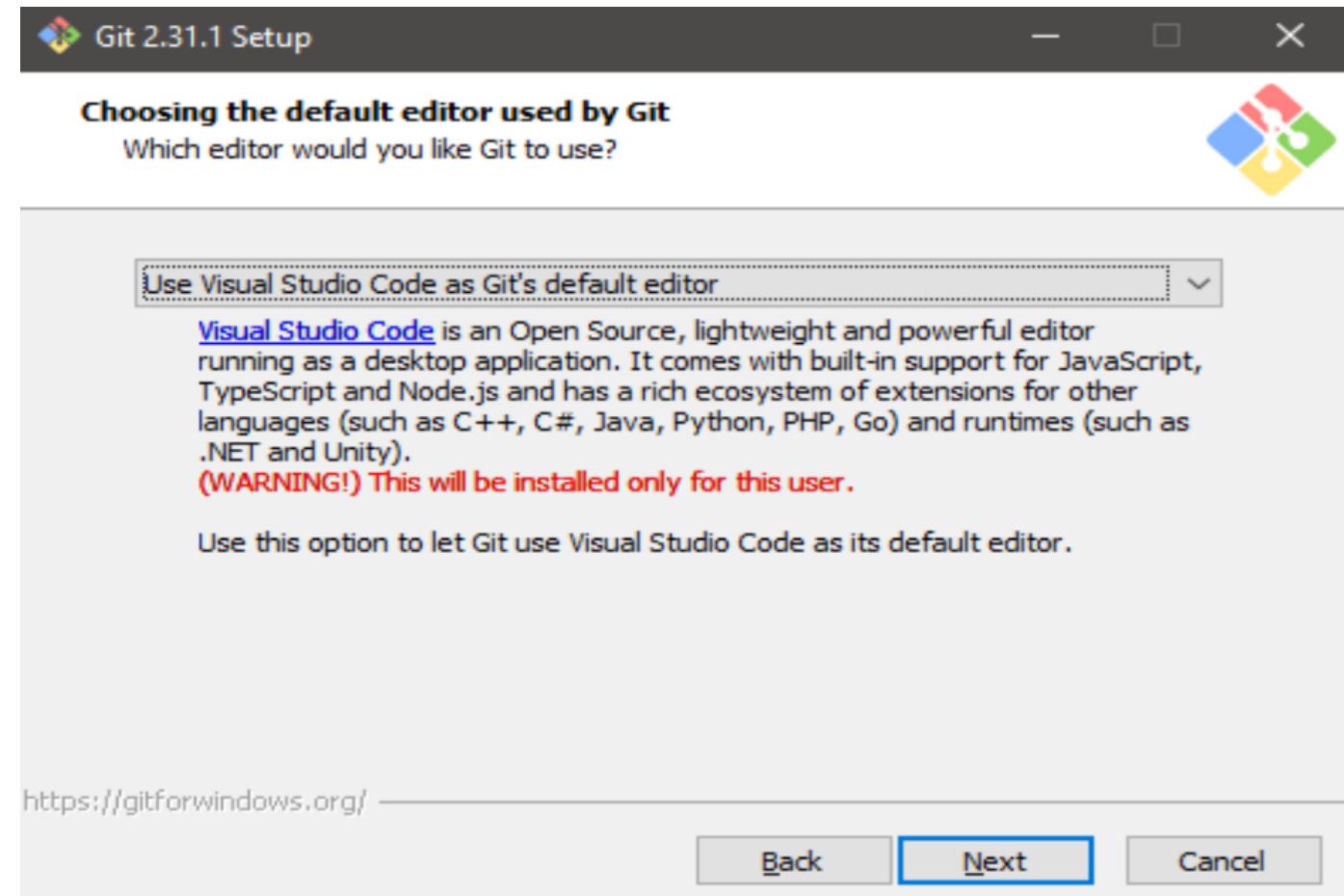
Se recomienda no tildar la opción “Don’t create a Start Menu folder”.



# Git Bash - Instalación: Paso por paso Windows

## Seleccionando el editor de texto por defecto

Se recomienda utilizar **Visual Studio Code** como editor por defecto.

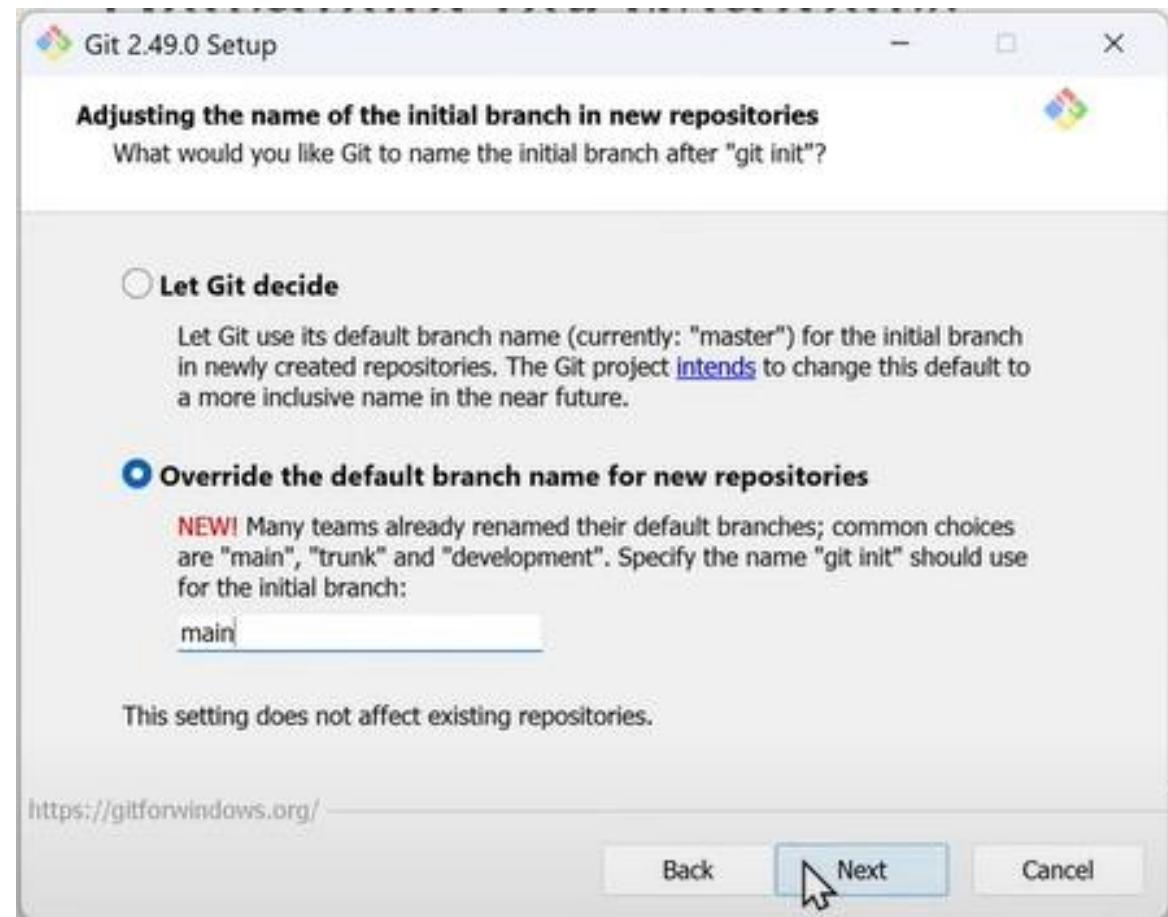


# Git Bash - Instalación: Paso por paso Windows

## Ajuste del nombre de la Rama Inicial

En este punto, es necesario seleccionar la segunda opción que cambia el término **master** por **main**.

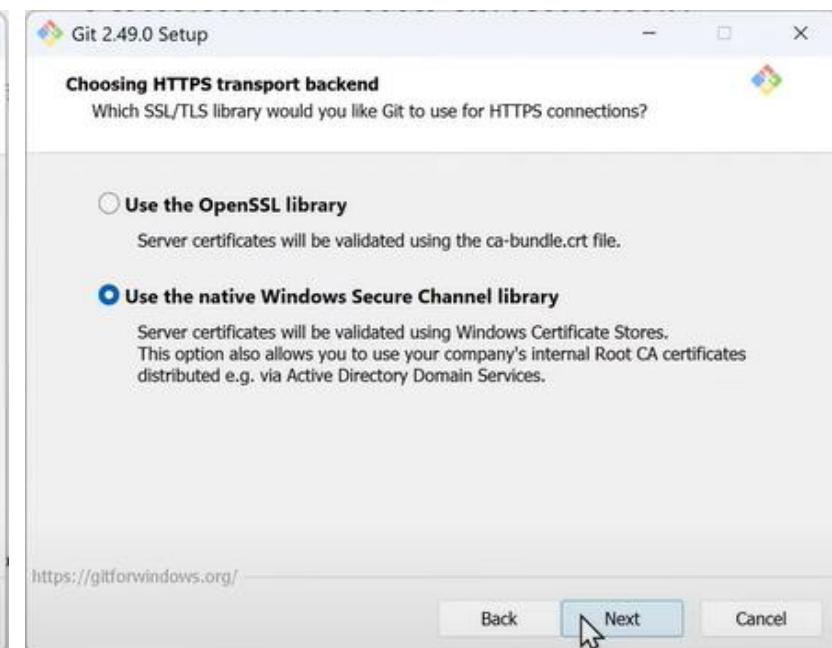
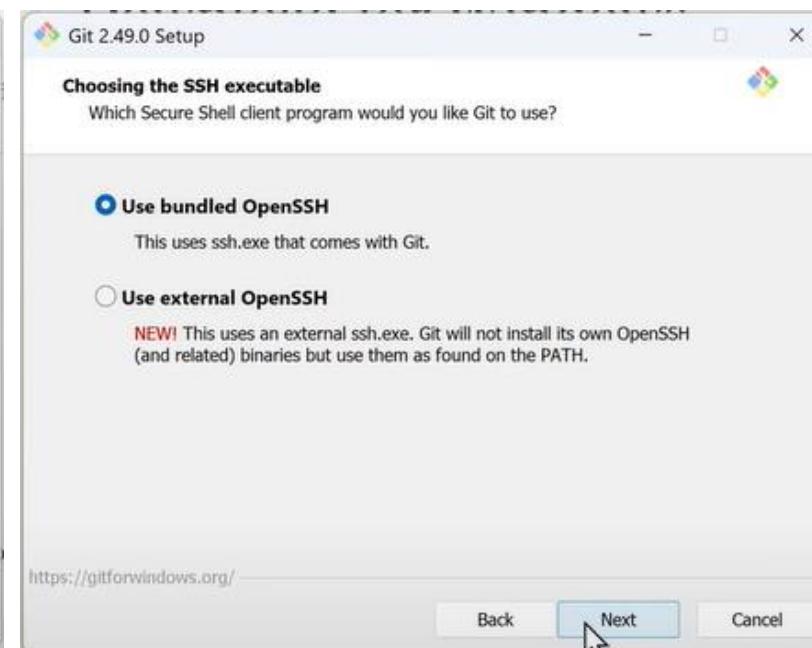
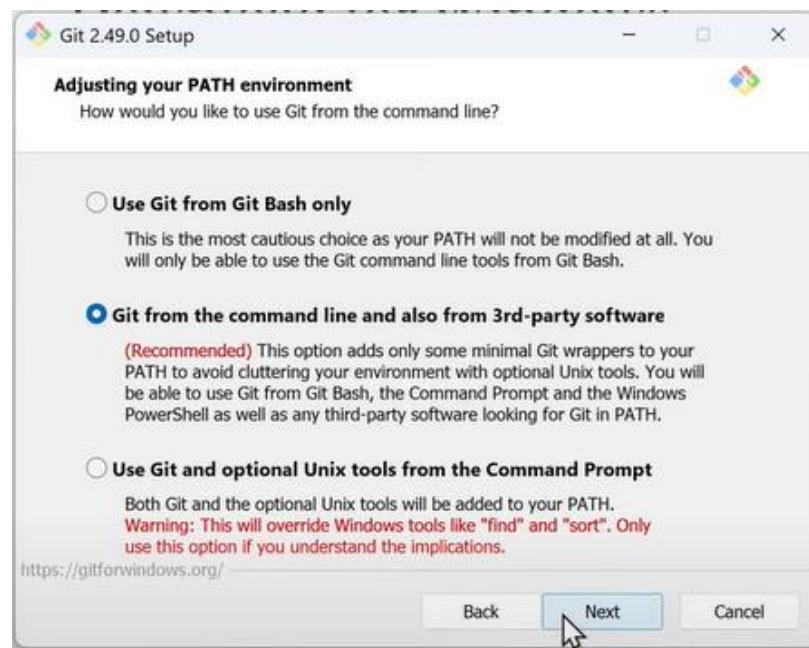
El término "master" tiene connotaciones históricas asociadas con la esclavitud, lo que puede resultar ofensivo o excluyente para algunos grupos, por lo que se escogió el término "Main" ya que es un término más neutral y descriptivo que simplemente indica la rama principal del proyecto.



# Git Bash - Instalación: Paso por paso Windows

## Componentes de la Instalación

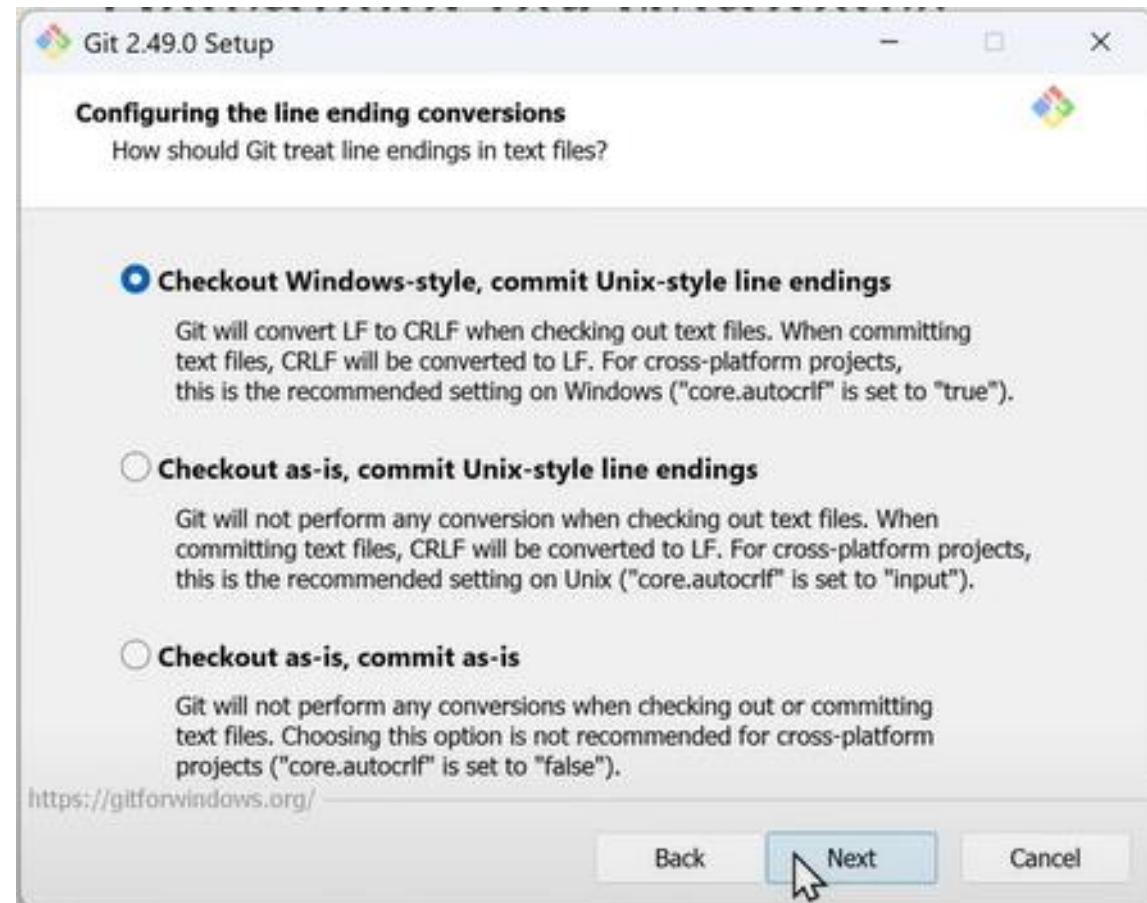
En esta ventana, tampoco es necesario modificar nada, podemos continuar con la opción **recomendada**.



# Git Bash - Instalación: Paso por paso Windows

## Configuración del Final de Línea

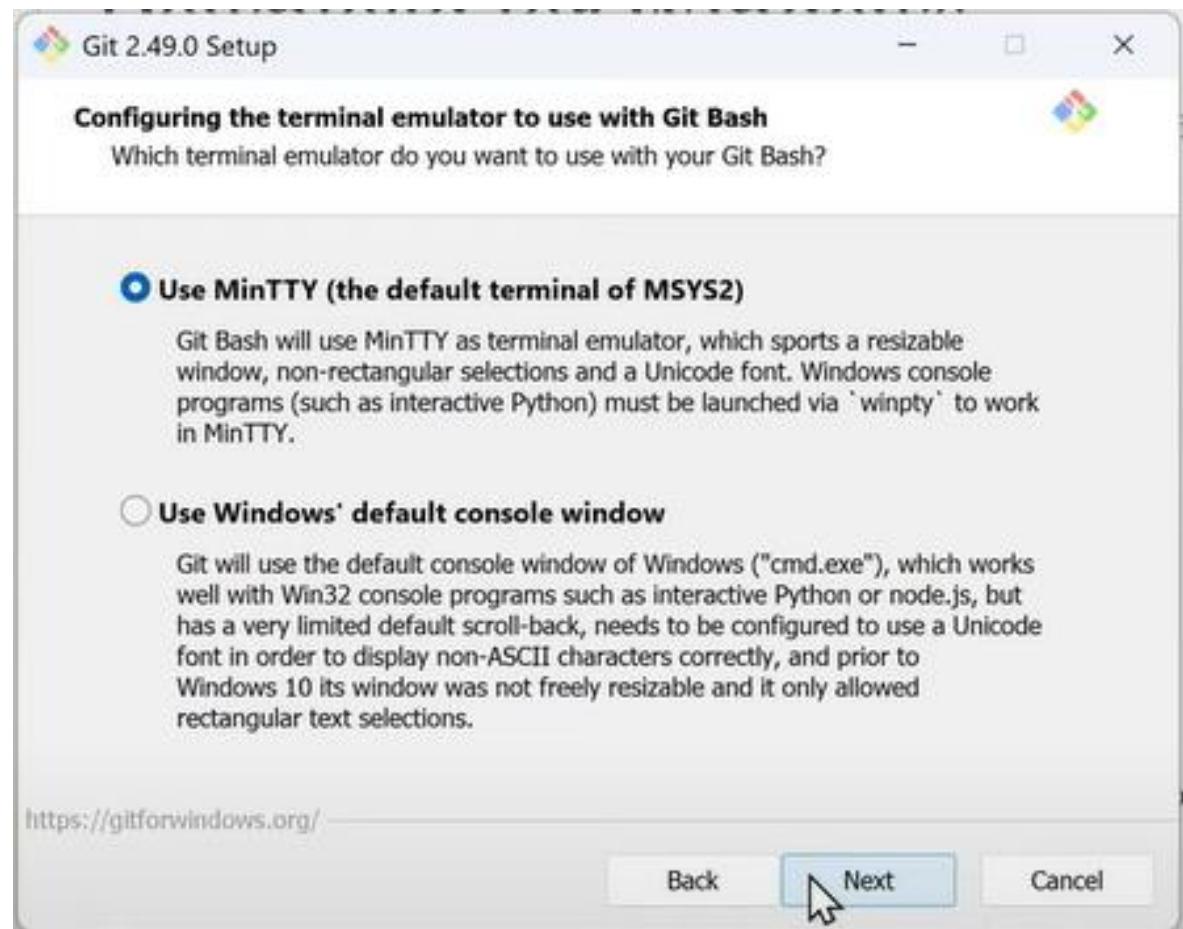
**IMPORTANTE:** Si estamos instalando Git en Windows, será necesario marcar la primera opción para no tener problemas en proyectos en los que trabajemos con personas que utilicen otros sistemas operativos.



# Git Bash - Instalación: Paso por paso Windows

## Componentes de la Instalación

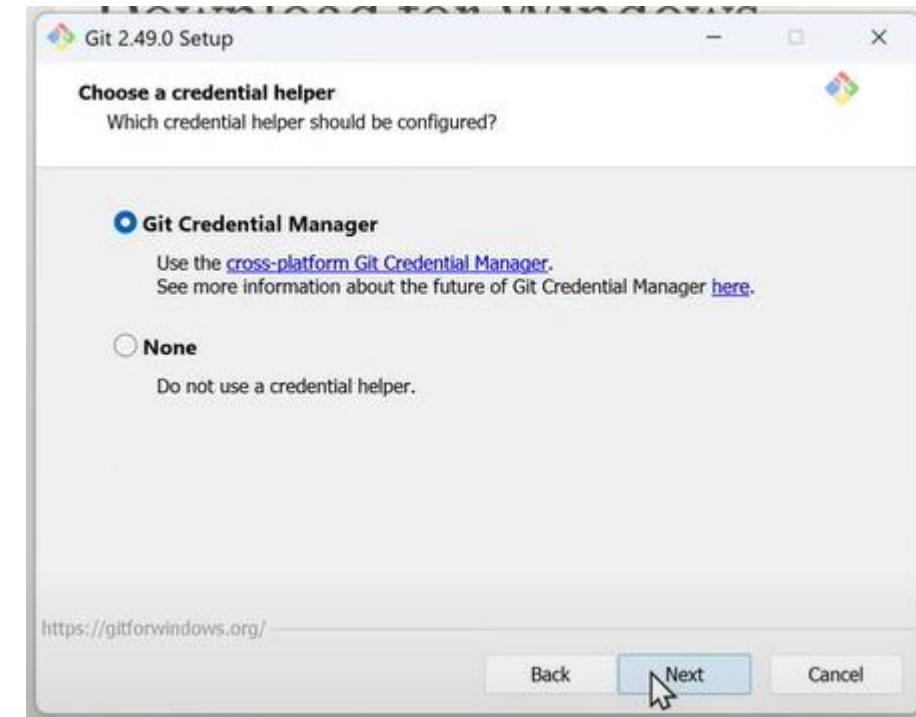
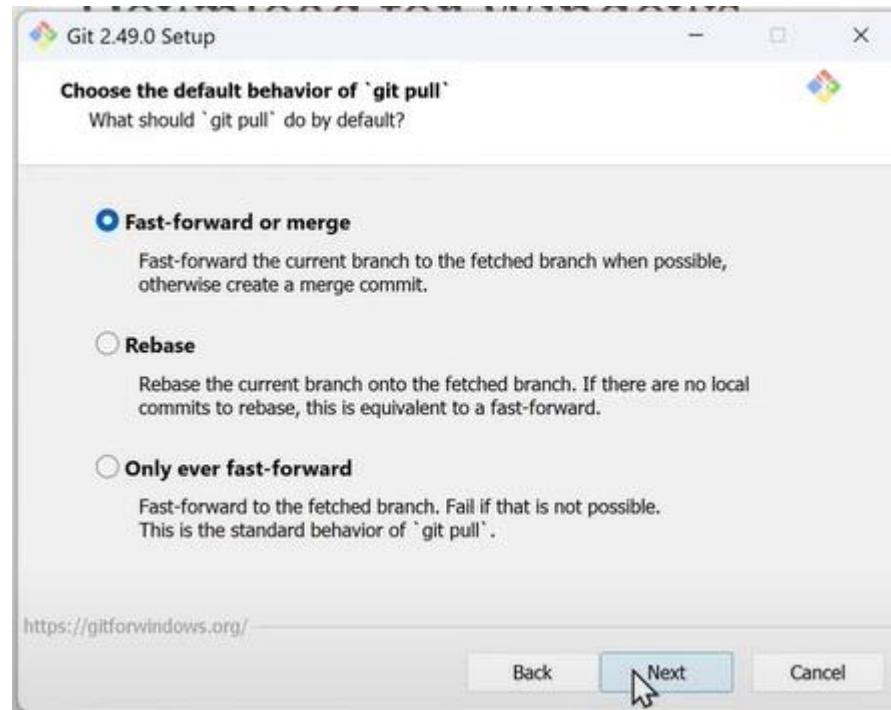
En esta ventana, no es necesario modificar nada, podemos continuar con la opción por defecto.



# Git Bash - Instalación: Paso por paso Windows

## Componentes de la Instalación

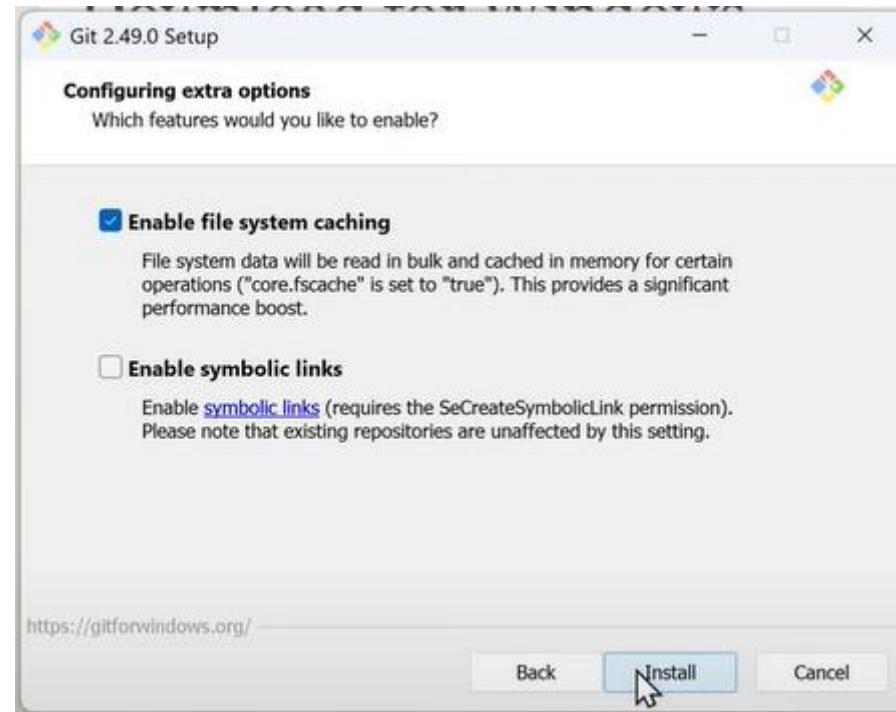
No es necesario modificar nada, podemos continuar con la opción por defecto.



# Git Bash - Instalación: Paso por paso Windows

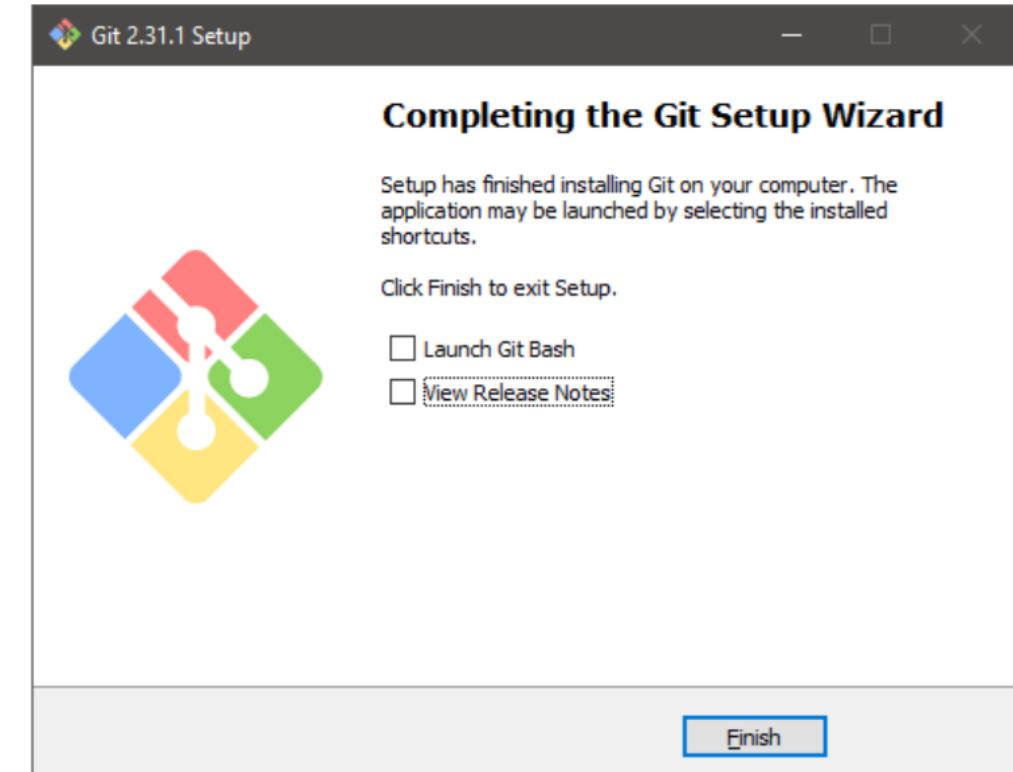
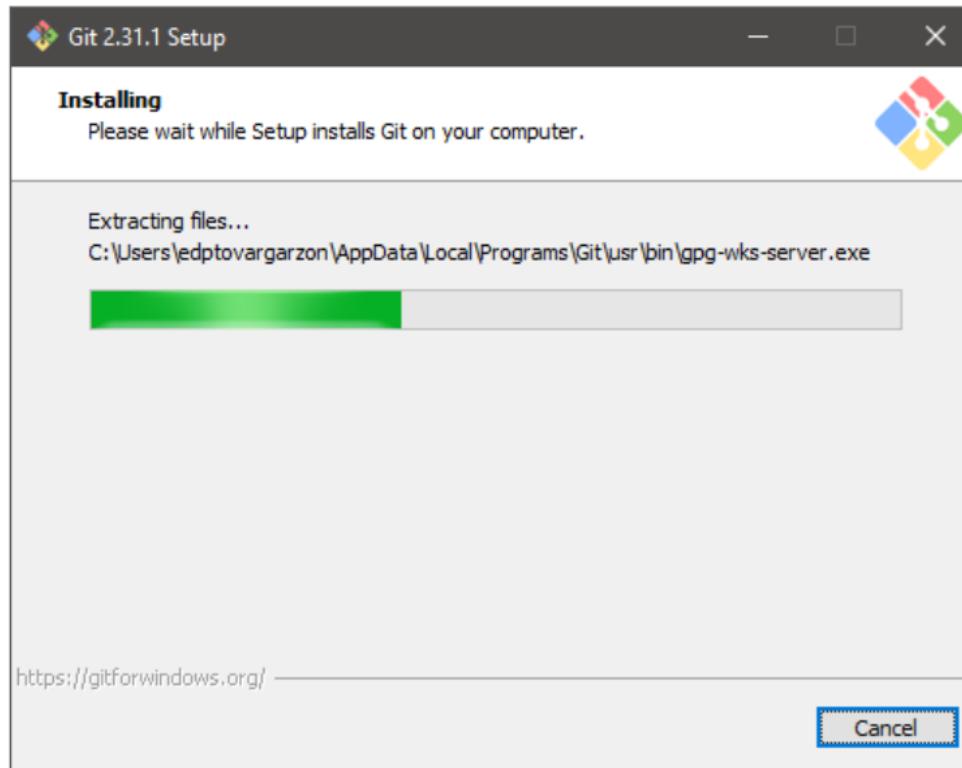
## Configuración Opcional

Tampoco es necesario modificar nada, podemos continuar con la opción por defecto. Finalmente, dar clic en “Install”.



# Git Bash - Instalación: Paso por paso Windows

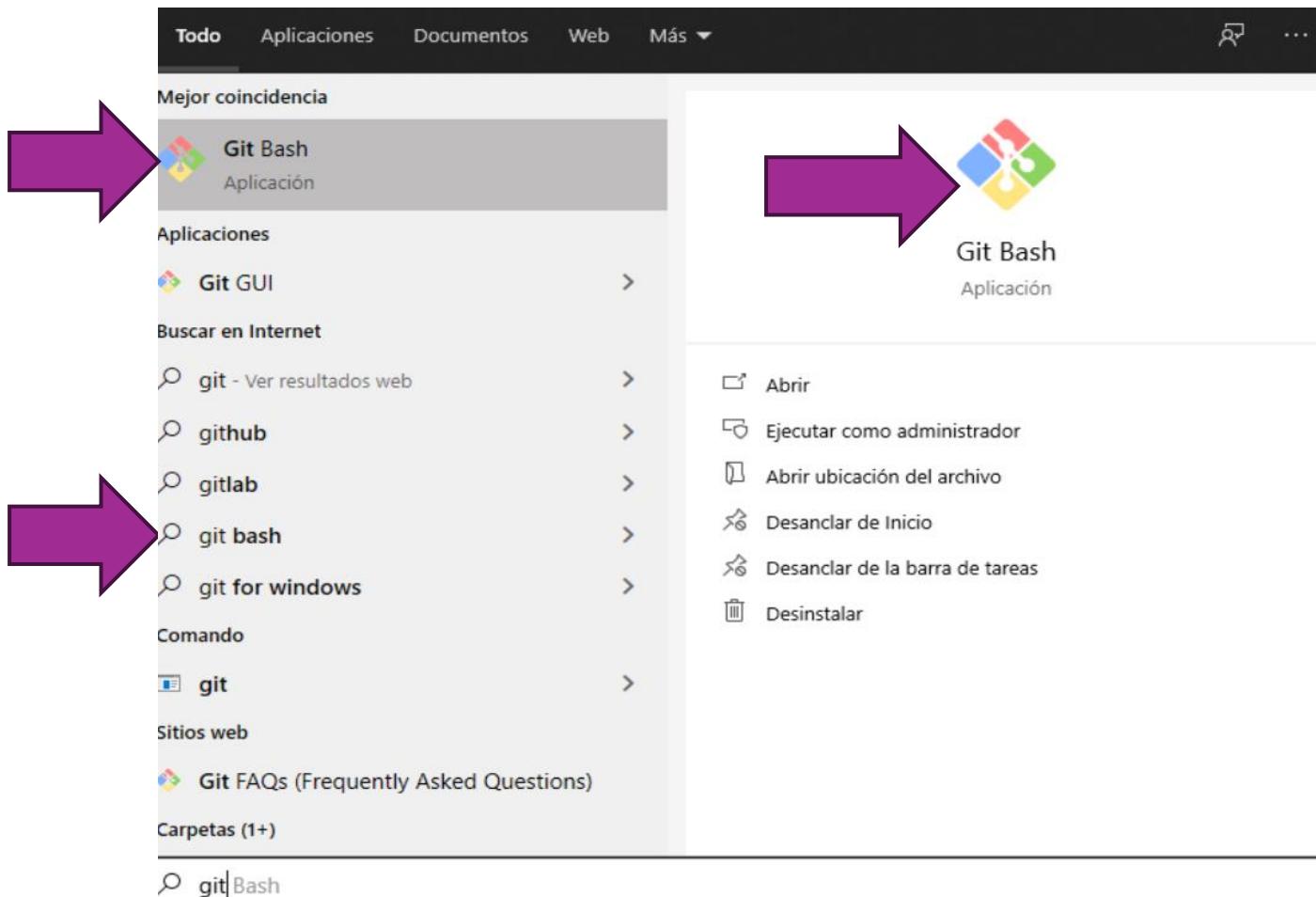
## Finalización de la Instalación



# Ejecutando Git Bash - Windows

## Iniciar Git

Desde el **menú de inicio**



Desde el **menú contextual** (haciendo clic derecho en el explorador de archivos)



Desde el **acceso directo en el escritorio**



# Git Bash en Mac OS

Por lo general, Git viene preinstalado en dispositivos Mac. Para verificarlo, busca en la lista de aplicaciones “Terminal” y ahí utiliza el siguiente comando: “git --version”.

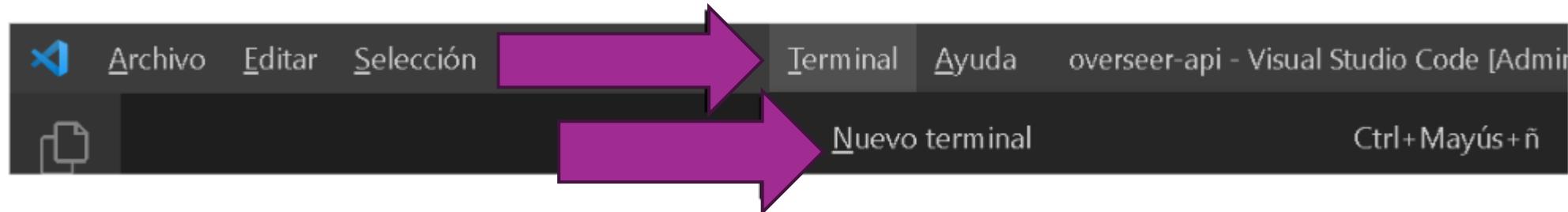


```
Last login: Mon Jul 29 13:30:45 on ttys000
jmalarcon@MacMini ~ % git --version
git version 2.39.3 (Apple Git-146)
jmalarcon@MacMini ~ %
```

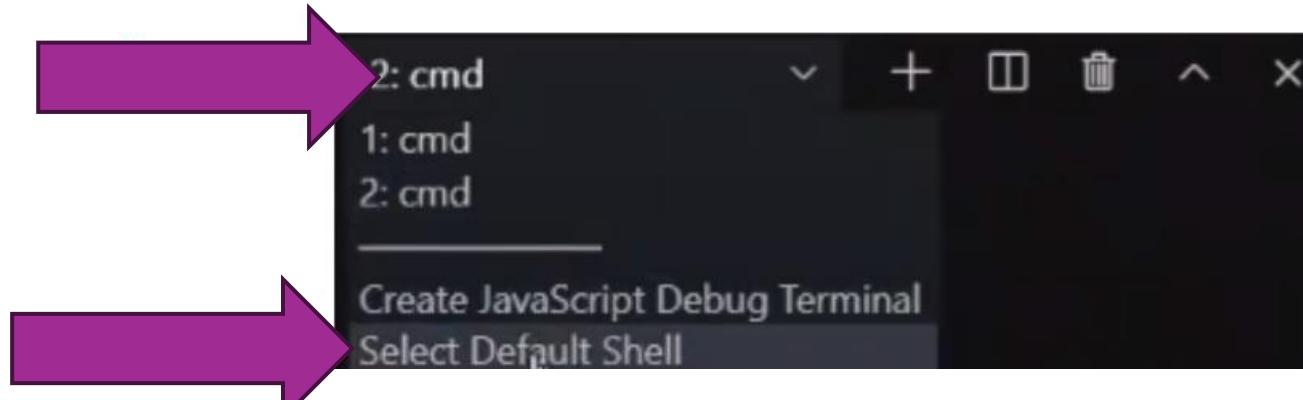
# Git Bash

## Integración con VSCode (parte 1)

- Primero, deberemos abrir la terminal desde la barra superior.
- Seleccionamos **Terminal > Nuevo Terminal**.



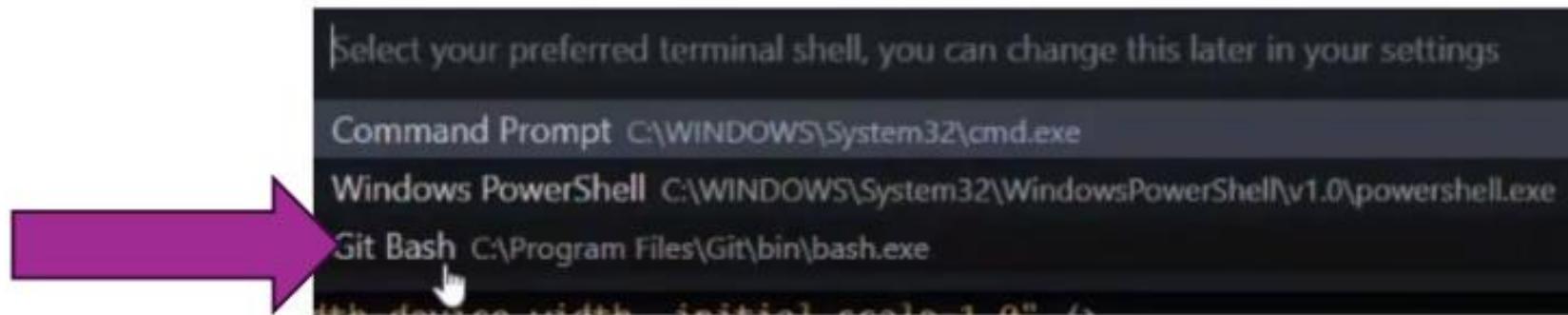
- Se nos abrirá en la parte inferior y deberemos seleccionar en cmd.
- Una vez abierta la lista desplegable deberemos entrar en la última opción.



# Git Bash

## Integración con VSCode (parte 2)

- En la parte superior, seleccionaremos como nuevo shell a **Git Bash**.



- Por último, para que nuestros cambios tengan efectos, abriremos una **Nueva Terminal**.



# La Terminal

# ¿Qué es “La Terminal”?

- Es un programa que está **presente en todos los sistemas operativos**. Usando la terminal podemos darle **órdenes al sistema**. Todo eso que hacemos **usando el mouse** lo podemos lograr a través de **comandos** en la terminal, como por ejemplo, crear una carpeta o un archivo, mover un archivo de lugar y mucho más.
- Un **comando** es una **instrucción específica** que se introduce en la **línea de comandos (CLI)** o en la terminal para realizar una acción o consultar información. Cada comando tiene un **nombre** y puede aceptar **argumentos** y opciones que **modifican su comportamiento**.



# Diferencias GUI - CLI

Graphical User Interface - Command Line Interface



Permite al usuario interactuar con los programas a través de una interfaz gráfica.

Es amigable, intuitiva y fácil de usar para usuarios principiantes.

Requiere más memoria para ejecutar sus componentes gráficos.

Ejecución más lenta debido a la capa gráfica y navegabilidad.

Poco flexible, solo realiza las operaciones para la cual fue diseñada.

Está orientado al público general y al uso cotidiano.

Menor precisión.

Permite al usuario ingresar comandos para interactuar con el software y el sistema operativo.

Requiere conocimientos técnicos de comandos y experiencia.

Necesita poca memoria para su ejecución.

Es muy rápida, ya que reduce al mínimo el uso de recursos y dependencias.

Flexible, permite combinar comandos.

Se utiliza en servidores, clusters, desarrollo y seguridad.

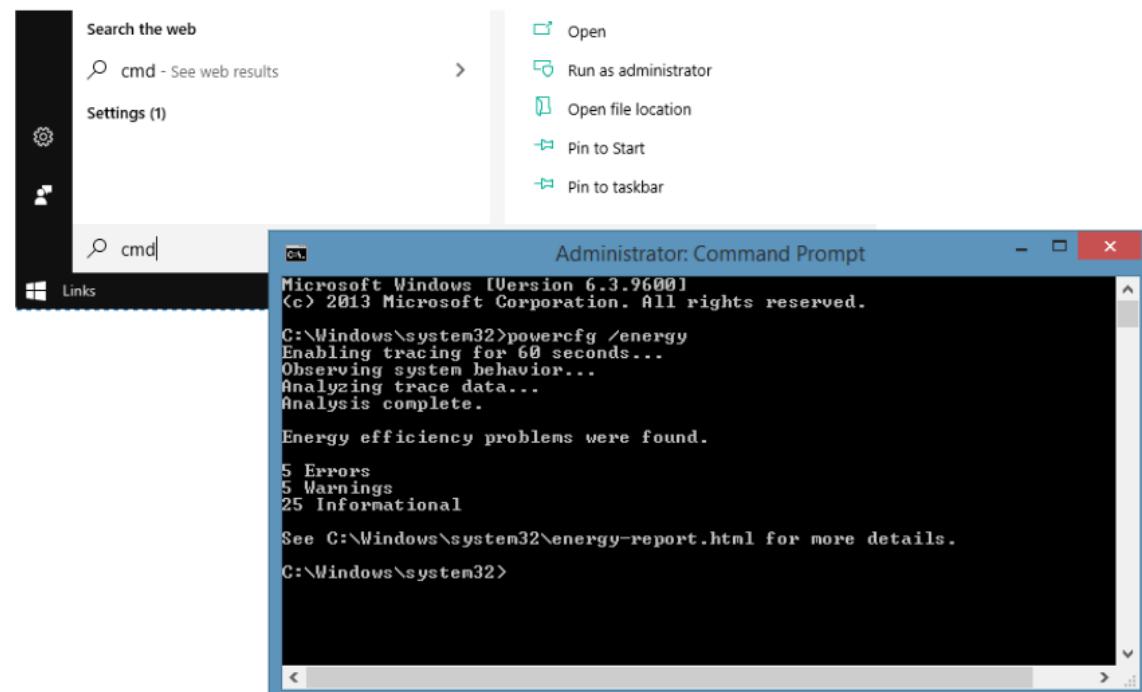
Mayor precisión.



# La Terminal

## La Terminal en Windows:

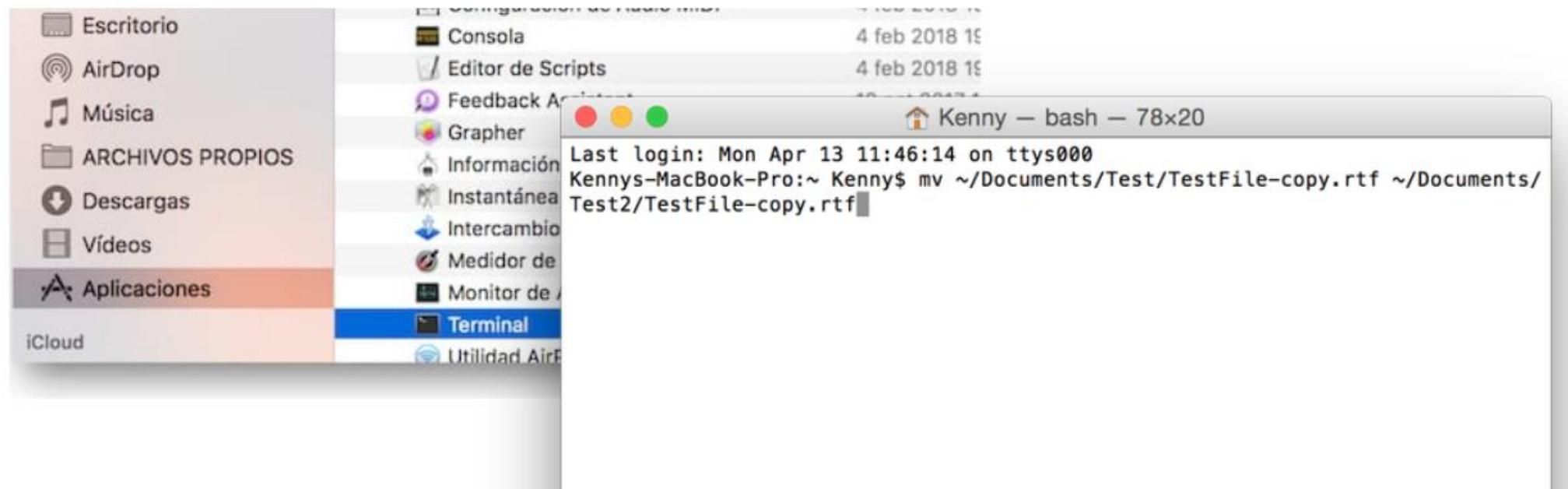
- Para abrir la terminal en Windows, podemos presionar las teclas **Win+R** y, en la ventana emergente, escribir el comando "**cmd.exe**".
- También podemos hacer uso del buscador y escribir la palabra "**cmd**" o "**PowerShell**".



# La Terminal

## La Terminal en MAC:

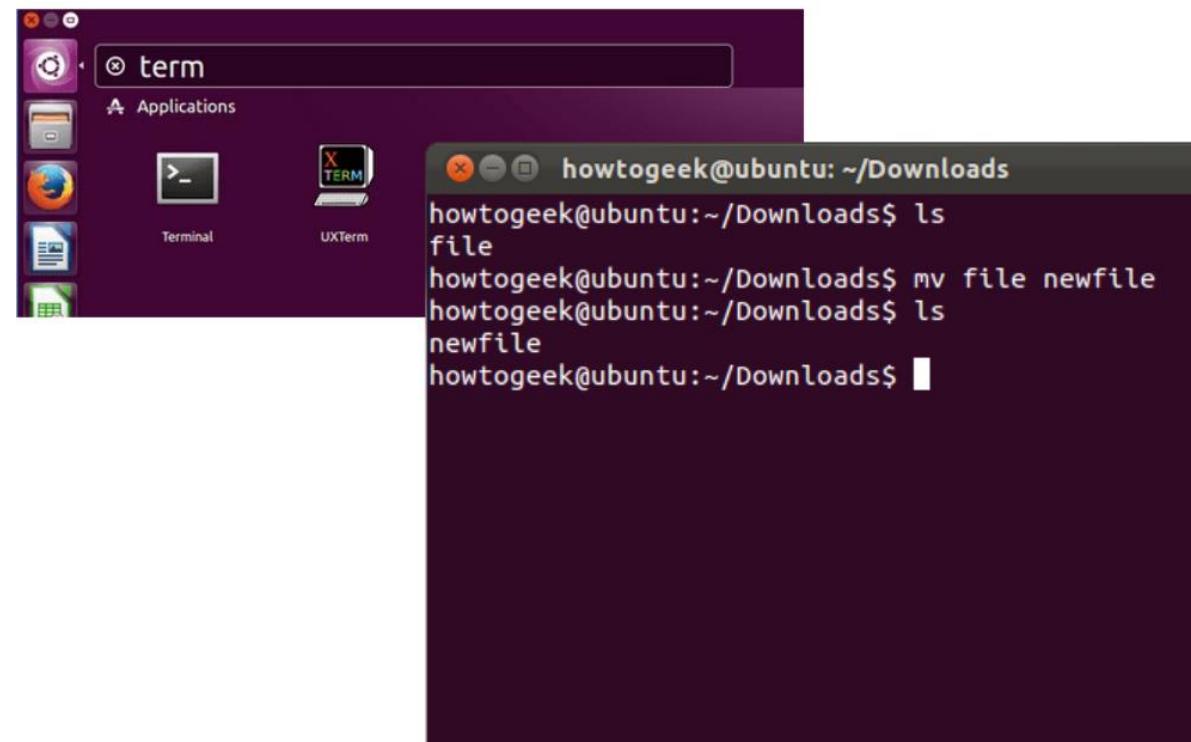
- Para abrir la terminal en Mac, podemos ir al Finder y pulsar en **Archivo > Nueva ventana del Finder** (**⌘N**).
- Luego, en el menú lateral izquierdo, hacer clic en **Aplicaciones** y buscar en el listado la terminal.



# La Terminal

## La Terminal en LINUX:

- Para abrir la terminal en Linux, podemos presionar las teclas **Ctrl+Alt+T** o simplemente ingresar la palabra "**terminal**" en el buscador que ofrece el sistema operativo.



# La terminal

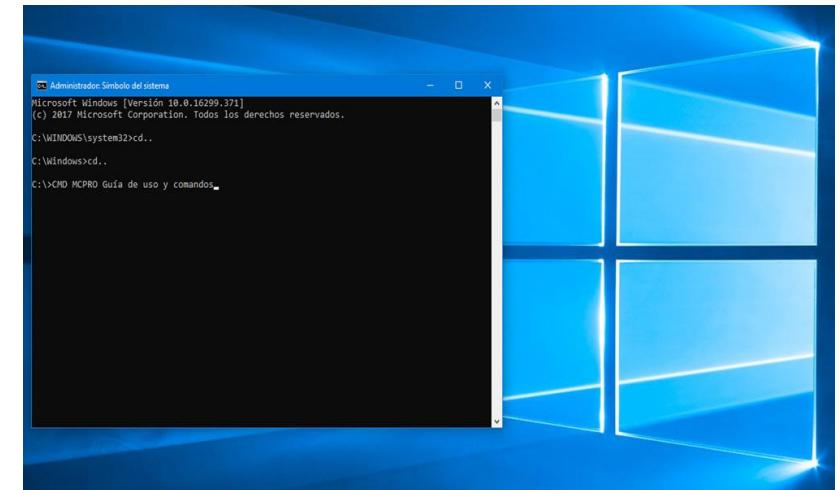
## ¿Qué es un comando?

Un comando en informática es una instrucción o orden que se da a un sistema operativo o programa para realizar una tarea específica.

### Características principales de los comandos:

- Instrucción específica:** Cada comando está diseñado para realizar una tarea particular.
- Interfaz de texto:** Se introducen generalmente a través de una interfaz de línea de comandos (CLI).
- Sintaxis definida:** Tienen una estructura específica que incluye el nombre del comando y, a menudo, argumentos o flags.
- Ejecución inmediata:** Al introducir un comando y presionar Enter, el sistema lo ejecuta de inmediato.
- Versatilidad:** Pueden realizar desde tareas simples hasta operaciones complejas.

```
Windows\system32\cmd.exe - ping 192.168.1.1 -t
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 time=167ms TTL=100
From 192.168.1.1: bytes=32 time=2ms TTL=100
From 192.168.1.1: bytes=32 time=2ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 - MISCOMPARE at offset 1 - time=2ms
From 192.168.1.1: bytes=32 time=4ms TTL=100
From 192.168.1.1: bytes=32 time=5ms TTL=100
From 192.168.1.1: bytes=32 time=387ms TTL=100
From 192.168.1.1: bytes=32 time=2ms TTL=100
From 192.168.1.1: bytes=32 time=2ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 time=109ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 time=2ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
From 192.168.1.1: bytes=32 time=5ms TTL=100
From 192.168.1.1: bytes=32 time=3ms TTL=100
From 192.168.1.1: bytes=32 time=324ms TTL=100
From 192.168.1.1: bytes=32 time=2ms TTL=100
From 192.168.1.1: bytes=32 time=1ms TTL=100
```



# La terminal

## Las flags en los comandos

Son opciones o modificadores que alteran el comportamiento de un comando en la línea de comandos.

Permiten a los usuarios personalizar cómo se ejecuta un comando.

### Características principales de las flags

- **Sintaxis:** Generalmente comienzan con un guion (-) para flags de una letra, o dos guiones (--) para flags de palabra completa.
- **Modifican el comportamiento:** Cambian cómo se ejecuta un comando o qué información muestra.
- **Pueden tener argumentos:** Algunas flags requieren información adicional.
- **Combinables:** En muchos casos, se pueden usar varias flags juntas.

Ejemplos en Listado de archivos (ls):

- ✓ `ls -l`: Lista detallada (long format)
- ✓ `ls -a`: Muestra archivos ocultos (all)
- ✓ `ls -lh`: Lista detallada con tamaños legibles por humanos (long, human-readable)

# Git Bash

## Comandos de la terminal Bash (Parte 1)

Mostrar	
\$ ls	Lista archivos en el directorio o carpeta
\$ ls -a	Lista todos los archivos, incluyendo los archivos ocultos
\$ ls -l	Muestra toda la información de una carpeta: usuario, grupo, permisos, tamaño, fecha y hora de creación.
\$ ls -R	Muestra las carpetas y los archivos contenidos en ellos de manera recursiva
\$ pwd	Muestra la carpeta en la que se está trabajando actualmente

Crear	
\$ mkdir [Carpeta]	Crea una nueva directorio o carpeta
\$ touch [Nombre del archivo]	Crea un nuevo archivo
Eliminar	
\$ rm [Nombre del archivo]	Elimina un archivo
\$ rmdir [Nombre de la carpeta]	Elimina una carpeta vacía
\$ rm -r [Nombre de la carpeta]	Elimina una carpeta y su contenido

Copiar / Mover / Renombrar	
\$ mv [ruta/archivo1] [ruta/archivo2]	Renombra archivos (archivo2 no debe existir o será sobreescrito)
\$ mv [ruta/carpeta1] [ruta/carpeta2]	Renombra la carpeta1 como carpeta2 (carpeta 2 no debe existir)
\$ mv [ruta/carpeta1] [ruta/carpeta2]	Mueve contenido de carpeta1 a carpeta2 (carpeta 2 debe existir)
\$ cp [ruta/archivo1] [ruta/archivo2] \$ cp [ruta/archivo1] [ruta/archivo2]	Copia un archivo o carpeta
opción: -r	Indica que copie recursivamente el contenido de las subcarpetas

Navegación entre carpetas	
\$ cd ..	Sube un nivel de carpeta
\$ cd	Cambia de carpeta
\$ cd/chosen/directory	Cambia a una carpeta específica

Caracteres especiales	
"" (comillas)	Nos permiten utilizar términos que consistan en más de una palabra
. (el punto)	Permite hacer referencia al directorio donde estamos ubicados actualmente

# Git Bash

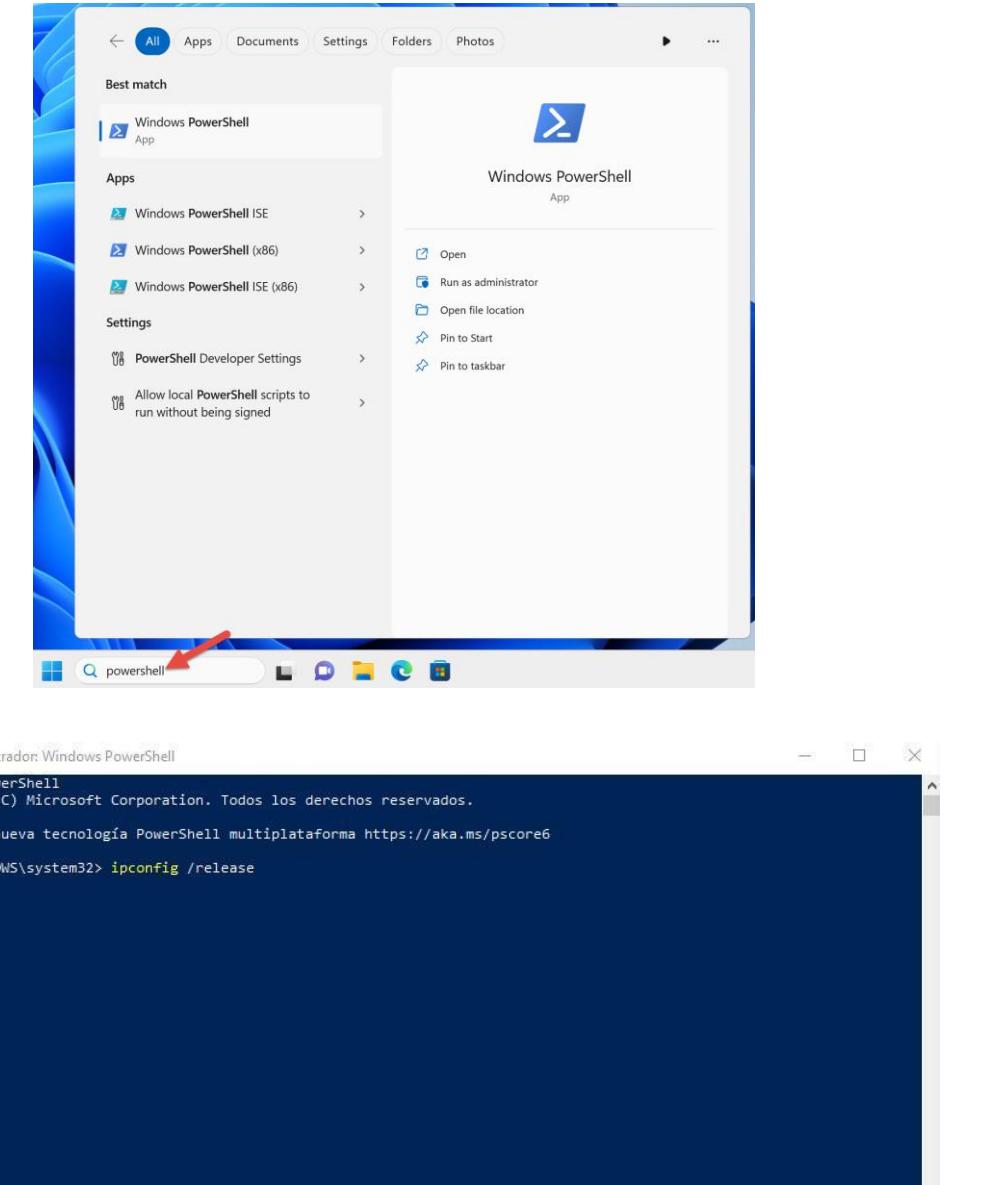
## Comandos de la terminal Bash (Parte 2)

Atajos de teclado		Otros comandos	
\$ <b>ctrl + c</b>	Finaliza un proceso vigente que está corriendo en la terminal	\$ <b>clear</b>	Limpia la pantalla de la terminal
\$ <b>ctrl + l</b>	Limpia la pantalla de la terminal	\$ <b>comando --help</b>	Muestra ayuda del comando

# Powershell

# Powershell

- PowerShell es una **plataforma de automatización de tareas y gestión de configuración desarrollada por Microsoft**. Incluye un lenguaje de scripting y una interfaz de línea de comandos (CLI) que permite a los usuarios realizar tareas administrativas en sistemas Windows y otros sistemas operativos, como Linux y macOS.
- Windows siempre tuvo una interfaz de línea de comandos, pero limitada en funciones y apoyada en un lenguaje casi obsoleto como Batch.



# Powershell

## Los Comandos en Powershell

Los comandos de PowerShell llamados **cmdlets** (por command-applets), están estructurados de la siguiente manera:

- **un verbo y un nombre separados por un guión (-): verbo-nombre.**

Vamos a ver un ejemplo, para ello abrimos nuestra consola de Powershell y ejecutamos:

- **Get-Command**

El verbo (evidentemente en inglés) describe la acción a realizar sobre el nombre.

Lo que hicimos con ese comando es que recuperamos (Get) los comandos (Command). En este caso, el comando nos devolverá una lista de los comandos disponibles de Powershell.

# Powershell

## Los Comandos en Powershell

Con PowerShell encontramos numerosos verbos genéricos tales como **Get, Set, Add, Remove**, entre otros, que se combinan con diferentes nombres como **Path, Variable, Item, Object, Computer, etcétera**.

**Los nombres que constituyen los comandos están siempre en singular** y esto es válido también para los parámetros.

Por lo tanto, es posible, mezclando verbos y nombres, acordarse fácilmente de un buen número de comandos.

Hay que tener en cuenta que los comandos, así como sus parámetros asociados, pueden escribirse indistintamente en **mayúsculas o en minúsculas**.

El analizador sintáctico PowerShell **no es case sensitive**.

# ¿Qué es Input y Output?

# ¿Qué es un INPUT en Programación?

ENTRADA



Un input (o entrada) en programación es cualquier tipo de dato o información que un programa recibe para realizar alguna acción o procesamiento.

Podemos pensarla como cuando le das información a alguien para que haga algo, por ejemplo, como cuando le dices a un amigo tu dirección para que pueda ir a visitarte.

Ejemplos Simples de Input:

## 1). Teclado:

- Cuando escribes algo en la computadora, como tu nombre, esa información es un input.

## 2). Archivo:

- Cuando un programa abre y lee un documento de texto, la información en ese documento es un input.

## 3). Sensores:

- En un teléfono móvil, la información que recibe de un sensor de temperatura es un input.



# ¿Qué es un OUTPUT en Programación?



Un output (o salida) en programación es cualquier tipo de dato o información que un programa produce y muestra o envía después de haber procesado algún input. Es el resultado que ves o recibes después de que el programa haya hecho su trabajo.

Ejemplos Simples de Output:

1). Pantalla:

- Cuando el programa muestra un mensaje en la pantalla, como "Hola, Juan".

2). Archivo:

- Cuando un programa guarda información en un documento de texto.

3). Impresora:

- Cuando un programa envía información a una impresora para que imprima un documento.



# Ejemplo de Input y Output

Imagina que tienes un programa que te pregunta tu nombre y luego te saluda.

## 1). Recepción del Input:

- El programa te pide que ingreses tu nombre. Tú escribes "Juan" en el teclado. Esto es el input.

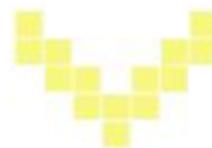
## 2). Procesamiento del Input:

- El programa toma tu nombre ("Juan") y lo usa para crear un mensaje. El mensaje podría ser:  
"Hola, Juan. ¡Bienvenido!".

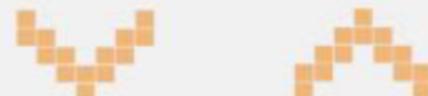
## 3). Generación del Output:

- El programa muestra el mensaje en la pantalla de tu computadora. Tú ves el mensaje:  
"Hola, Juan. ¡Bienvenido!". Esto es el output.

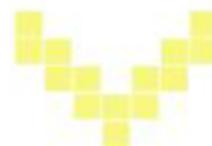
**INPUT**



**STORAGE**



**PROCESS**



**OUTPUT**



Momento de  
poner a prueba  
lo aprendido!