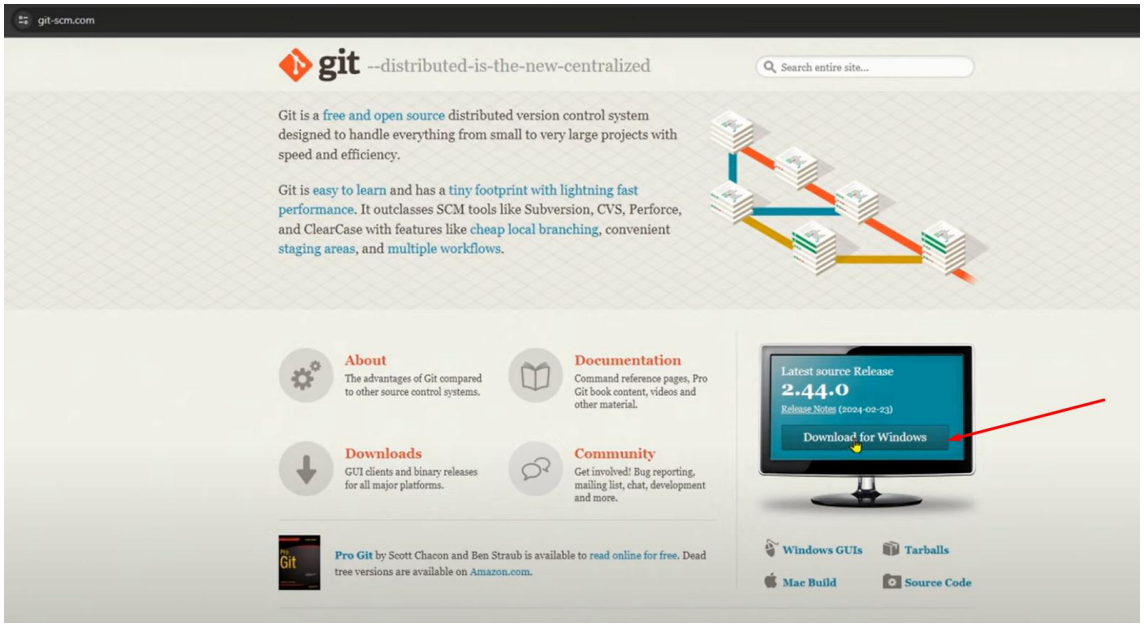
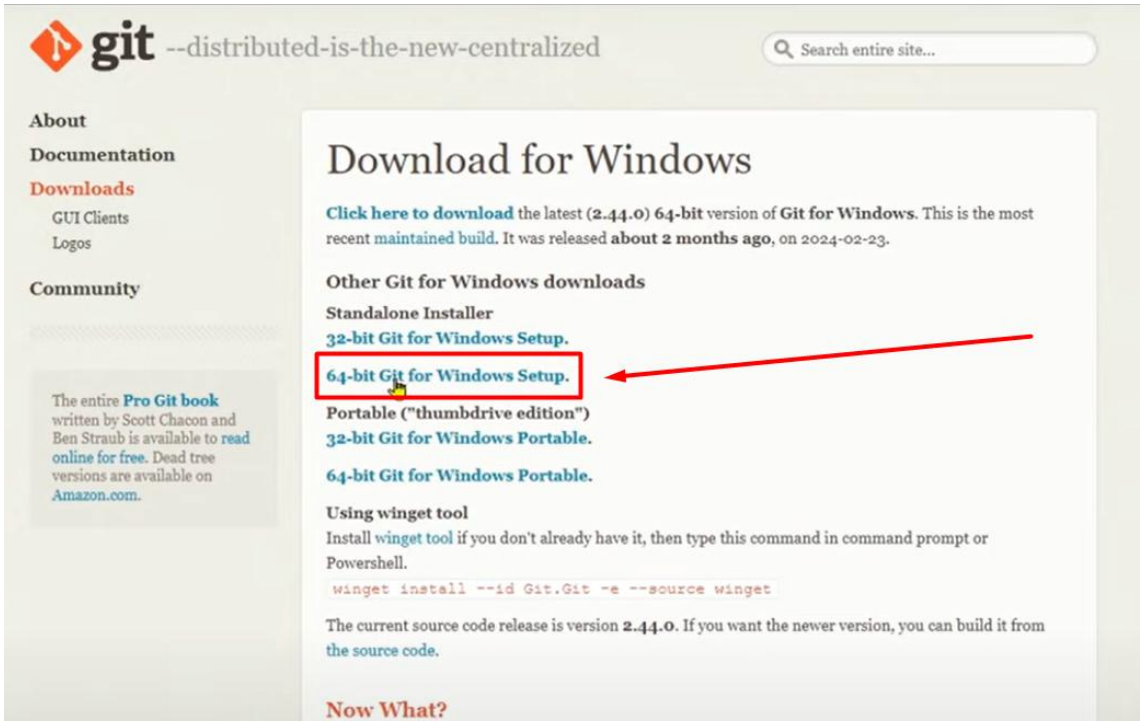
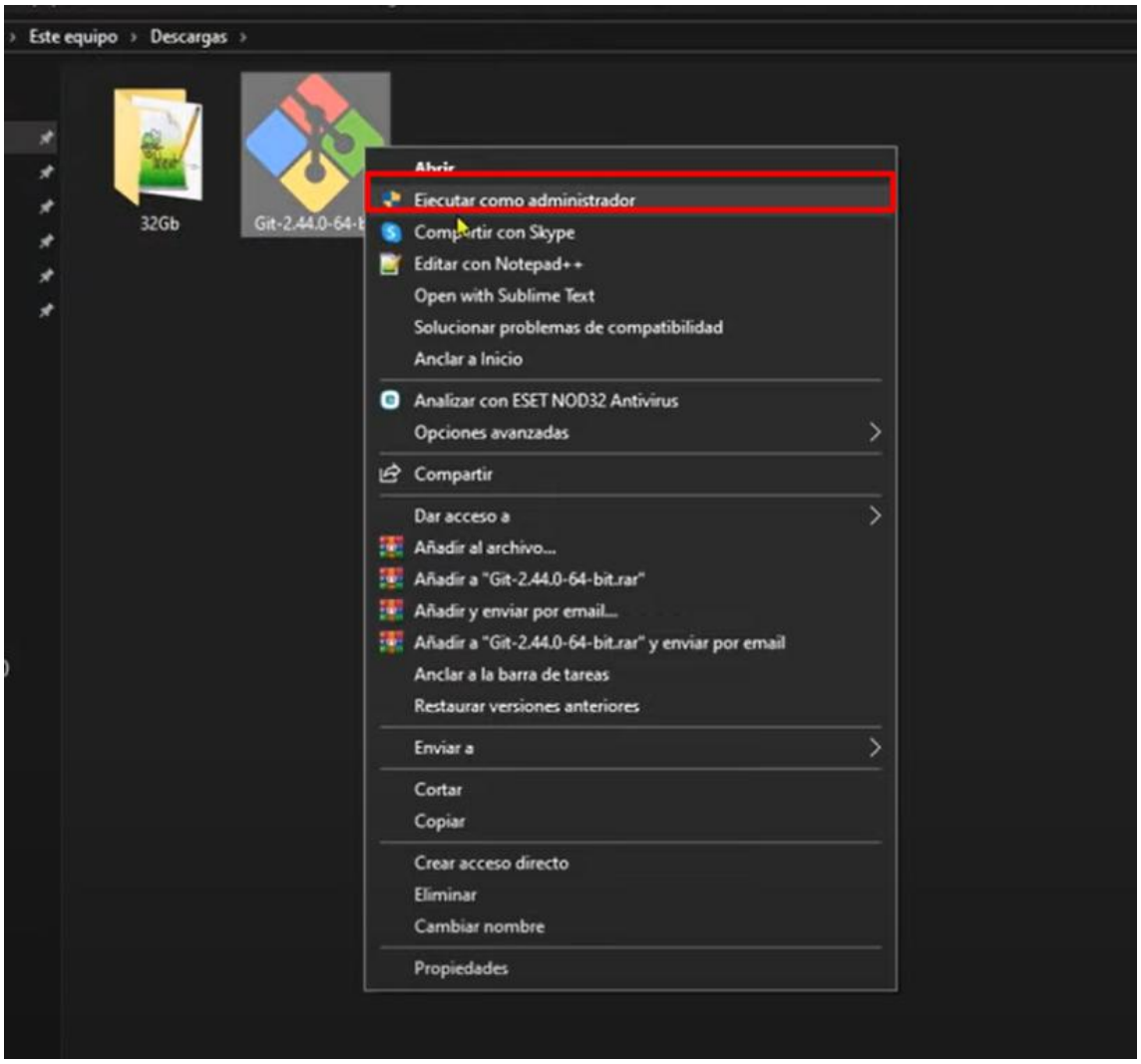
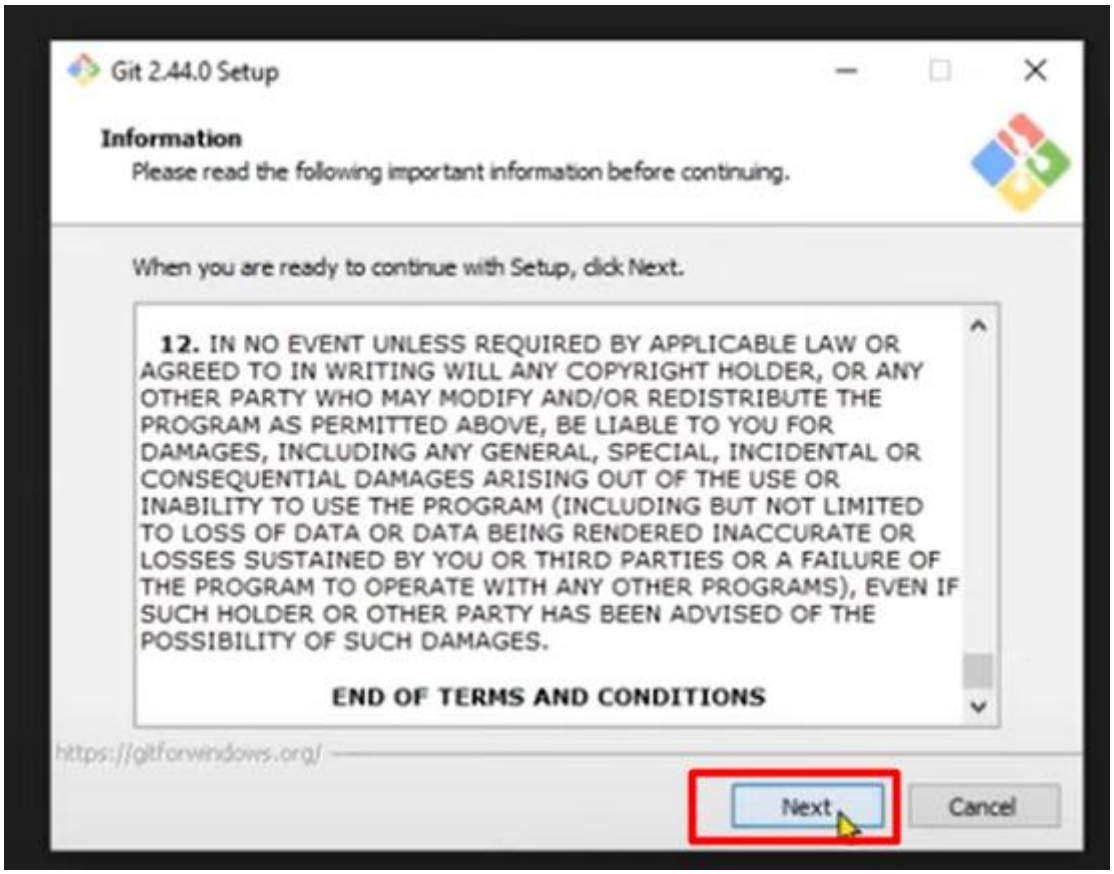
Tutorial Instalación de Git

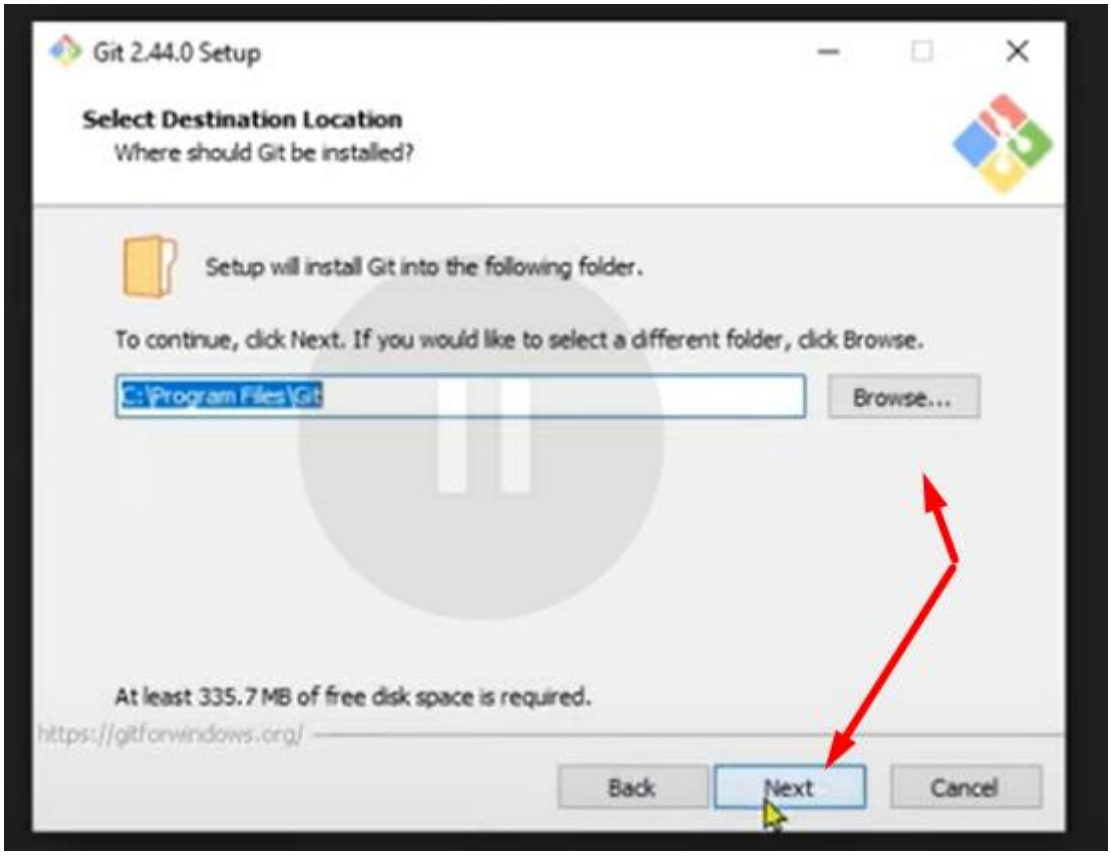
1. Ingresar a la web de git

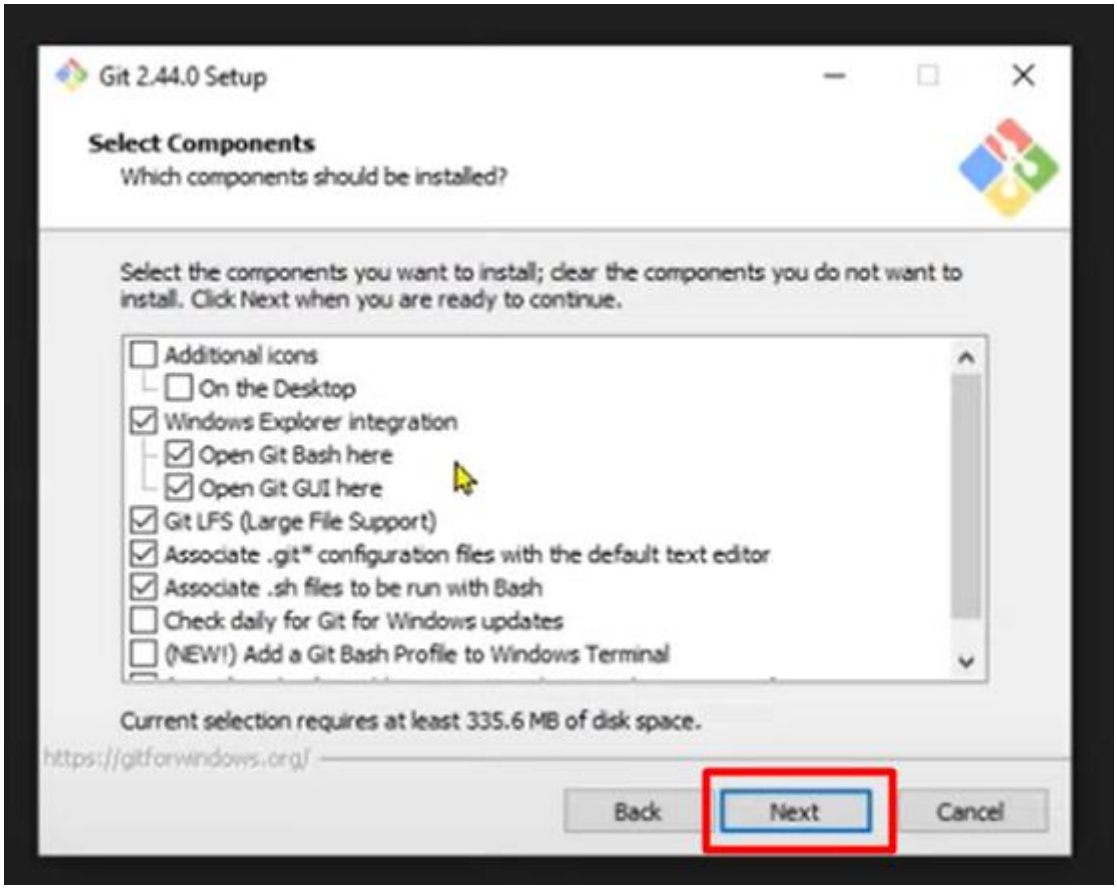




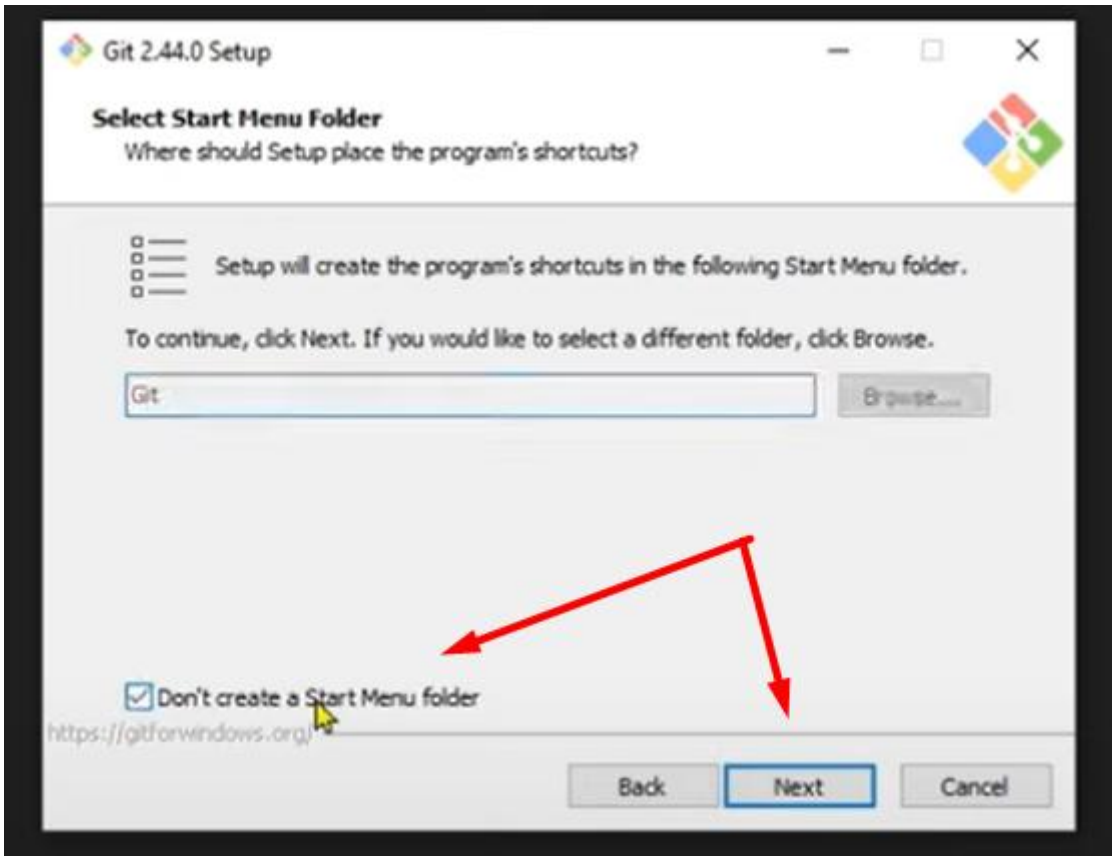

Este equipo
Descargas
32Gb
Git-2.44.0-64-b
Abrir
Ejecutar como administrador
Compartir con Skype
Editar con Notepad++
Open with Sublime Text
Solucionar problemas de compatibilidad
Anclar a Inicio
Analizar con ESET NOD32 Antivirus
Opciones avanzadas
Compartir
Dar acceso a
Añadir al archivo...
Añadir a "Git-2.44.0-64-bit.rar"
Añadir y enviar por email...
Añadir a "Git-2.44.0-64-bit.rar" y enviar por email
Anclar a la barra de tareas
Restaurar versiones anteriores
Enviar a
Cortar
Copiar
Crear acceso directo
Eliminar
Cambiar nombre
Propiedades


Git 2.44.0 Setup
Information
Please read the following important information before continuing.
When you are ready to continue with Setup, click Next.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
END OF TERMS AND CONDITIONS
https://gitforwindows.org/
Next
Cancel

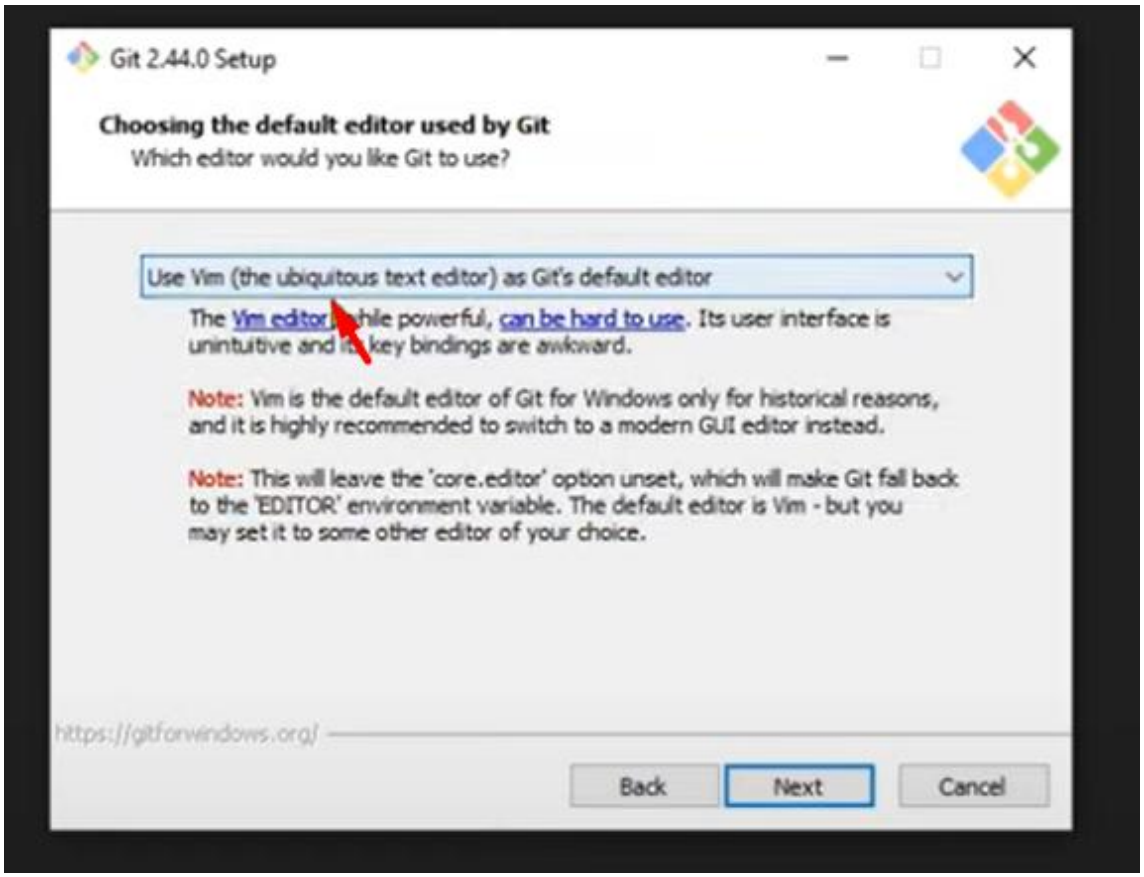

Git 2.44.0 Setup
Select Destination Location
Where should Git be installed?
Setup will install Git into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\Program Files\Git
Browse...
At least 335.7 MB of free disk space is required.
https://gitforwindows.org/
Back
Next
Cancel



Indicar si se desea crear un menú de inicio rápido

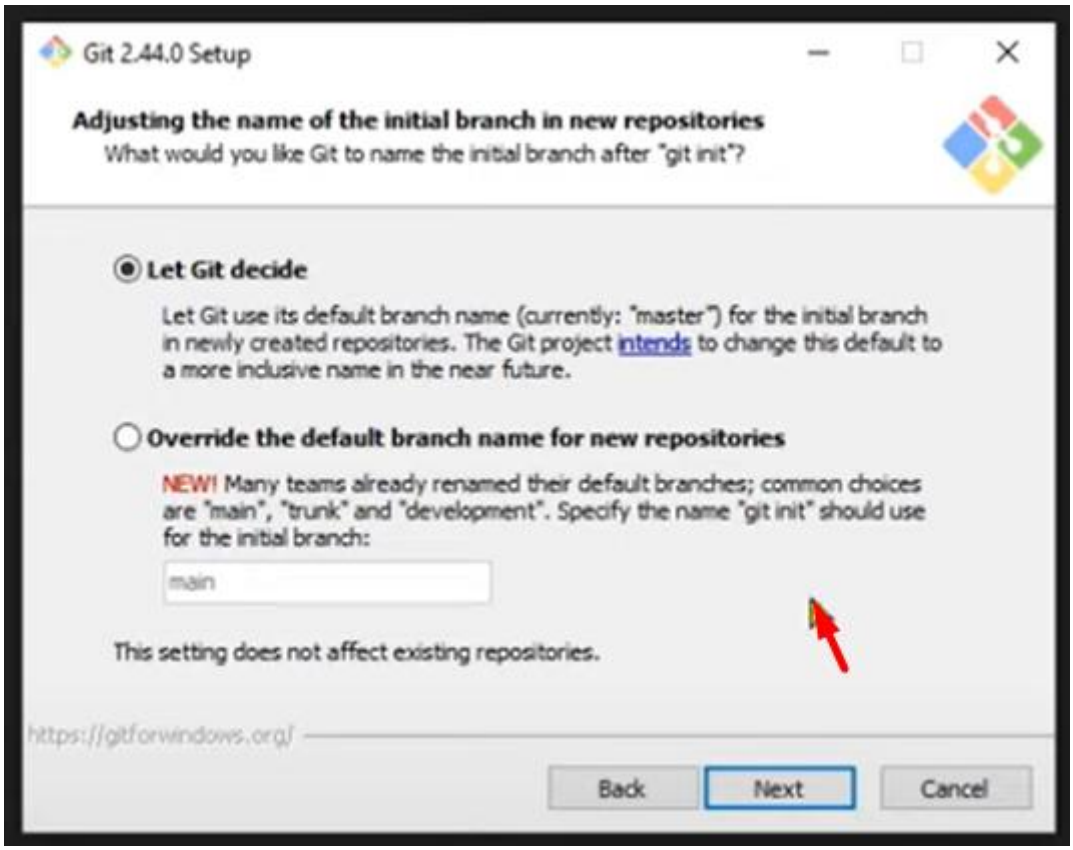


Se selecciona el editor que voy a poder usar para editar comandos dentro de una operación con git



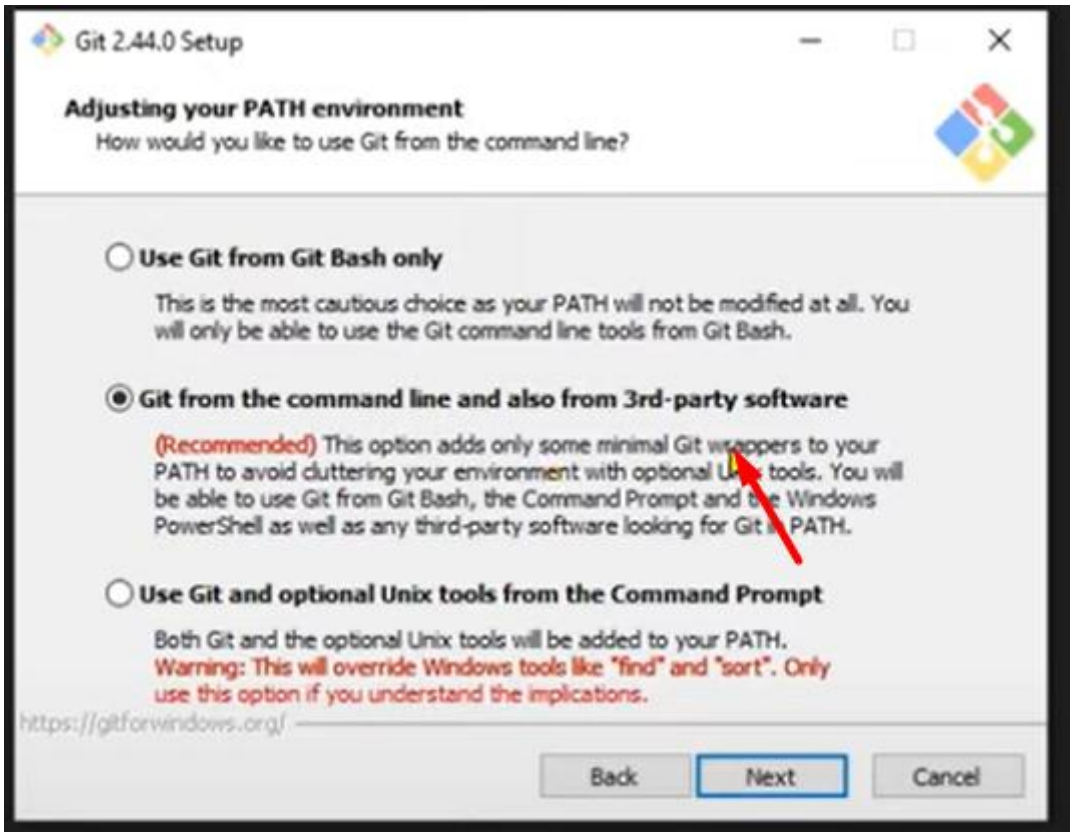
Esta opción es importante porque Git necesita un editor de texto para ciertas operaciones, principalmente:

1. **Escribir mensajes de commit**: Cuando ejecutas git commit sin el parámetro -m, Git abrirá el editor configurado para que escribas un mensaje de commit detallado.
2. **Editar mensajes en rebase interactivo**: Cuando realizas operaciones como git rebase -i, necesitarás un editor para modificar la lista de commits.
3. **Resolver conflictos de fusión**: En algunos casos, el editor se abre para ayudarte a resolver conflictos de fusión (merge conflicts).

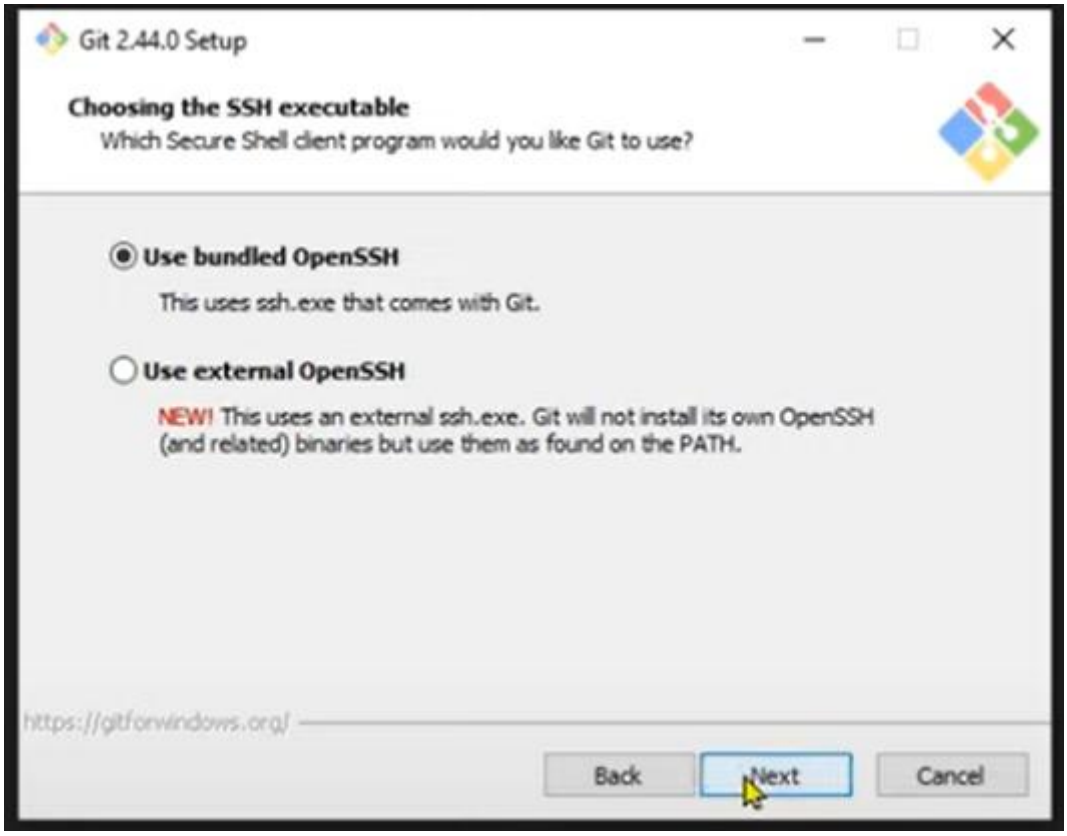


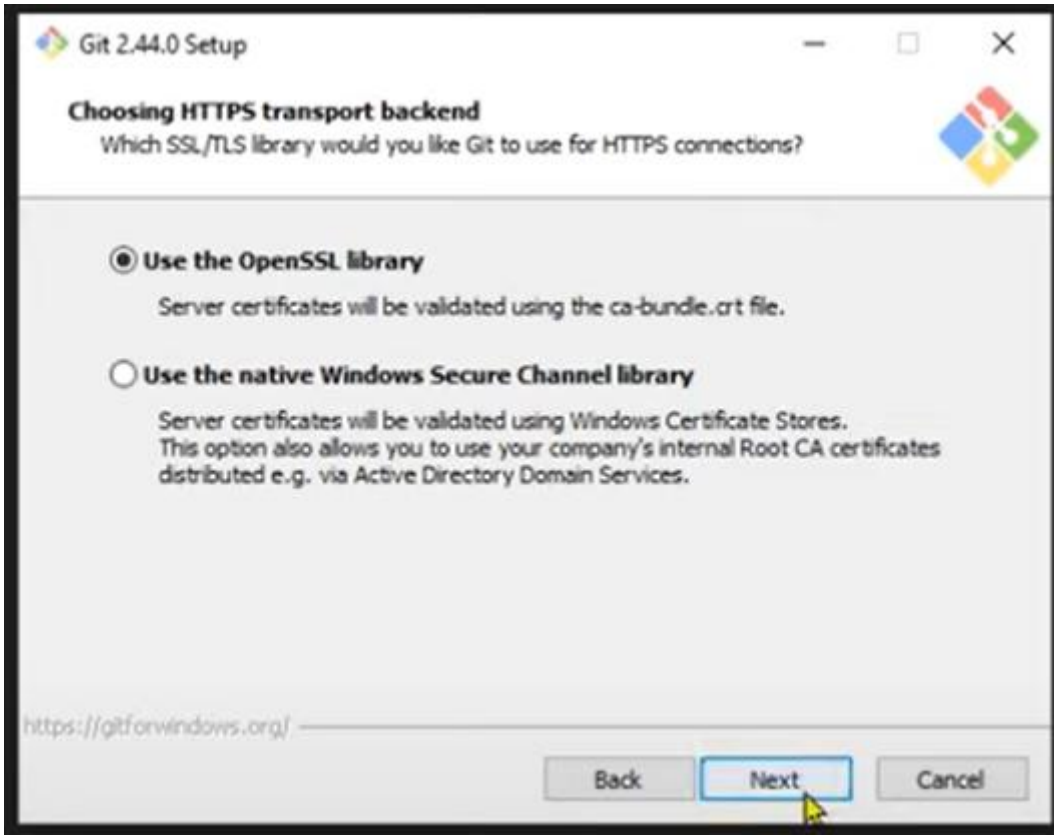
En la imagen, la opción seleccionada es "Let Git decide" (Dejar que Git decida), lo que significa:

1. Git usará su nombre de rama predeterminado (actualmente "master") para la rama inicial cuando crees nuevos repositorios con git init.
2. El texto menciona que el proyecto Git tiene la intención de cambiar este valor predeterminado a un nombre "más inclusivo" en el futuro (en versiones más recientes de Git, este cambio ya se implementó y el nombre predeterminado pasó a ser "main").



Conexión segura



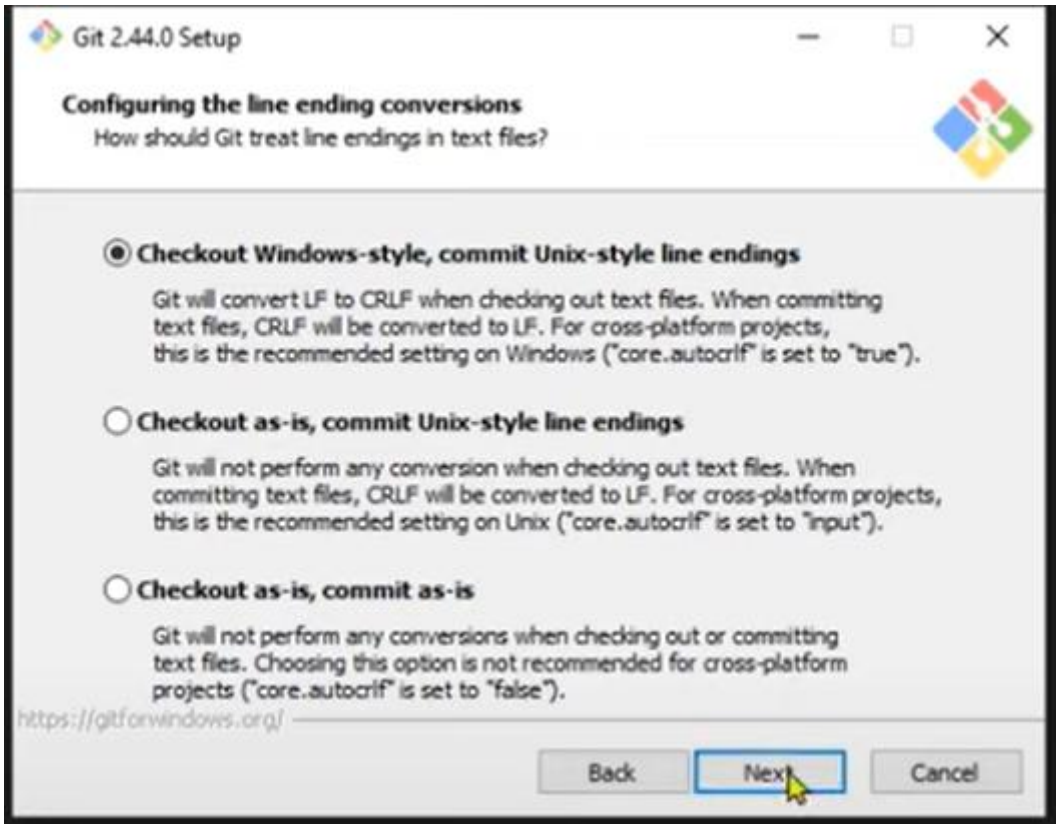


Pantalla del instalador de Git muestra opciones para configurar el backend de transporte HTTPS, es decir, la biblioteca SSL/TLS que Git utilizará para las conexiones seguras cuando interactúes con repositorios remotos a través de HTTPS.

Hay dos opciones disponibles:

1. **Use the OpenSSL library** (opción seleccionada):
    - Utiliza la biblioteca OpenSSL para verificar certificados de servidor
    - Los certificados de servidor se validarán usando el archivo ca-bundle.crt incluido con Git
    - Esta es una opción más universal y generalmente recomendada para uso personal
2. **Use the native Windows Secure Channel library**:
    - Utiliza la biblioteca nativa de Windows (Schannel) para la seguridad

- Los certificados se validarán usando el almacén de certificados de Windows
- Permite usar certificados de autoridades de certificación (CA) internas de tu empresa distribuidos a través de Active Directory o servicios de dominio
- Es especialmente útil en entornos corporativos donde se utilizan certificados personalizados



Esta pantalla del instalador de Git se refiere a la configuración de conversión de finales de línea (line endings) en archivos de texto, lo cual es particularmente importante cuando se trabaja con proyectos entre diferentes sistemas operativos.

La opción seleccionada es **"Checkout Windows-style, commit Unix-style line endings"**, que significa:

- Cuando descargas (checkout) archivos de un repositorio Git en tu sistema Windows, Git convertirá automáticamente los finales de línea de LF (estilo Unix, solo un carácter de salto de línea) a CRLF (estilo Windows, retorno de carro + salto de línea).

- Cuando haces commit de tus cambios, Git convertirá automáticamente los finales de línea de CRLF a LF antes de almacenarlos en el repositorio.

Esta configuración (también conocida como core.autocrlf = true) es la recomendada para usuarios de Windows en proyectos multiplataforma porque:

1. Permite que trabajen con archivos que usan el formato de final de línea nativo de Windows (CRLF) cuando editan localmente.
2. Asegura que el código se almacene en el repositorio con finales de línea de estilo Unix (LF), que es el estándar en la mayoría de los sistemas y herramientas de desarrollo.
3. Evita problemas comunes como diferencias falsas en control de versiones o problemas de interpretación en scripts cuando se ejecutan en diferentes sistemas operativos.

Esta configuración es particularmente útil para equipos con desarrolladores que trabajan en diferentes sistemas operativos (Windows, macOS, Linux), ya que mantiene la consistencia en el repositorio mientras permite a cada desarrollador trabajar con el formato que su sistema prefiere.

Esta pantalla del instalador de Git te permite configurar qué terminal emulador se utilizará con Git en Windows.

La opción seleccionada es "Use MinTTY (the default terminal of MSYS2)", que significa:

Git utilizará MinTTY como su terminal emulador en lugar de la consola predeterminada de Windows.

Las ventajas de MinTTY (la opción seleccionada) incluyen:

Ventana redimensionable libremente

Permite selecciones no rectangulares de texto

Soporta fuentes Unicode para mostrar caracteres especiales correctamente

Mejor experiencia visual y funcional en general

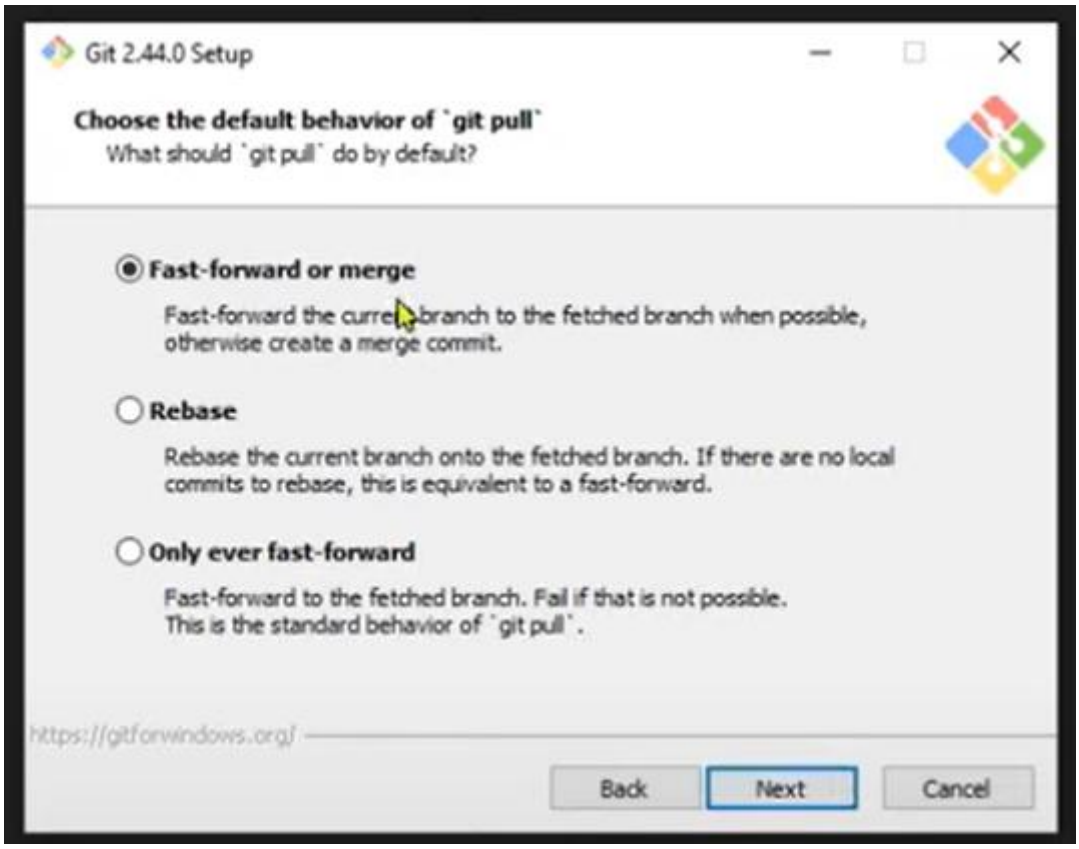
La alternativa no seleccionada ("Use Windows' default console window") utilizaría cmd.exe de Windows, que tiene varias limitaciones:

Capacidad de desplazamiento (scroll-back) muy limitada

Necesita configuración adicional para mostrar caracteres Unicode correctamente

En versiones anteriores a Windows 10, no permitía redimensionar libremente la ventana

Solo permite selecciones rectangulares de texto



Esta pantalla del instalador de Git te permite configurar el comportamiento predeterminado del comando git pull, que es el comando que usas para actualizar tu repositorio local con los cambios del repositorio remoto.

La opción seleccionada es **"Fast-forward or merge"**, que significa:

- Cuando ejecutas git pull, Git intentará primero realizar un "fast-forward" (avance rápido) de tu rama actual hacia la rama remota si es posible. Esto ocurre cuando tu rama local no tiene commits adicionales respecto a la rama remota.
- Si no es posible hacer un fast-forward (porque tienes commits locales que no están en el repositorio remoto), Git creará automáticamente un "commit de fusión" (merge commit) que combina tus cambios locales con los cambios remotos.

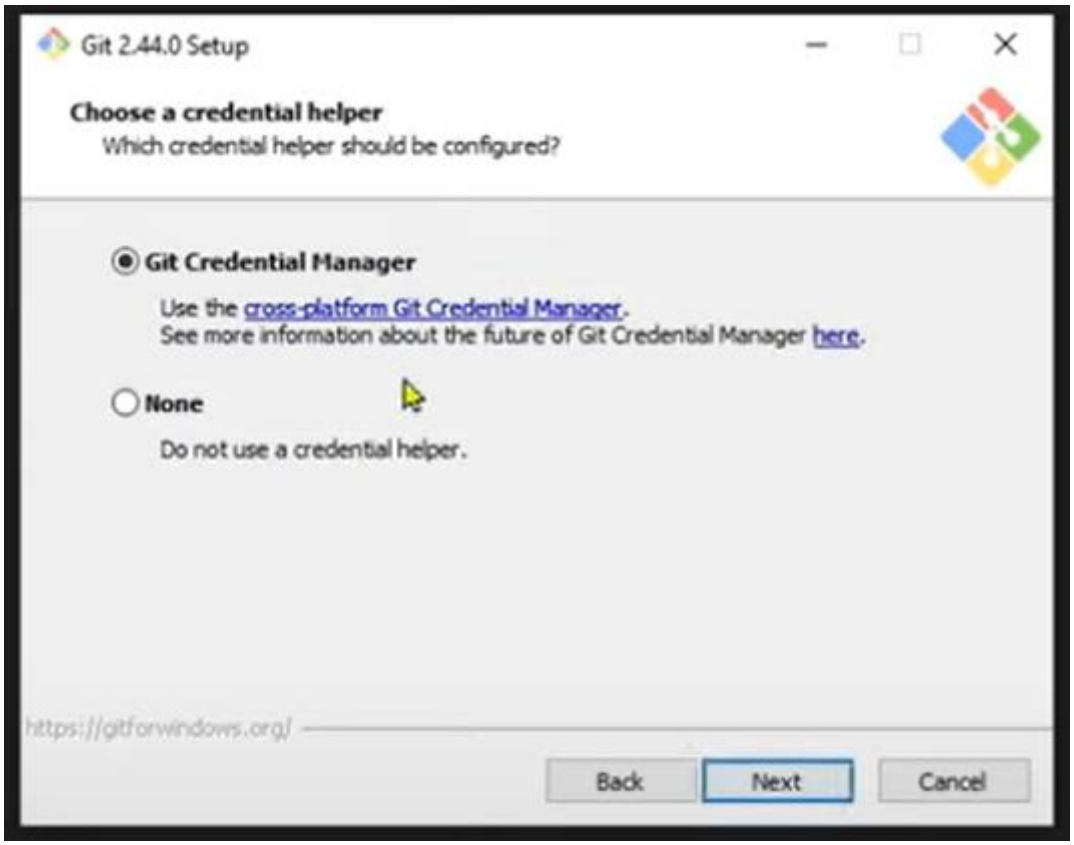
Esta opción es equivalente a configurar git config --global pull.ff only y es un buen equilibrio entre mantener un historial limpio (con fast-forward cuando es posible) y preservar el contexto de las integraciones (con commits de fusión cuando es necesario).

Las otras opciones disponibles son:

- **Rebase**: Reaplica tus cambios locales encima de los cambios remotos, creando un historial lineal pero reescribiendo el historial de commits.

- **Only ever fast-forward**: Solo permite actualizar si se puede hacer un fast-forward simple; si no es posible, el comando git pull fallará y tendrás que resolver manualmente la situación.

La opción seleccionada (Fast-forward or merge) es generalmente la más segura y flexible para principiantes y equipos, ya que evita reescribir el historial de commits (lo que puede causar problemas en repositorios compartidos) y siempre permite la integración de cambios.



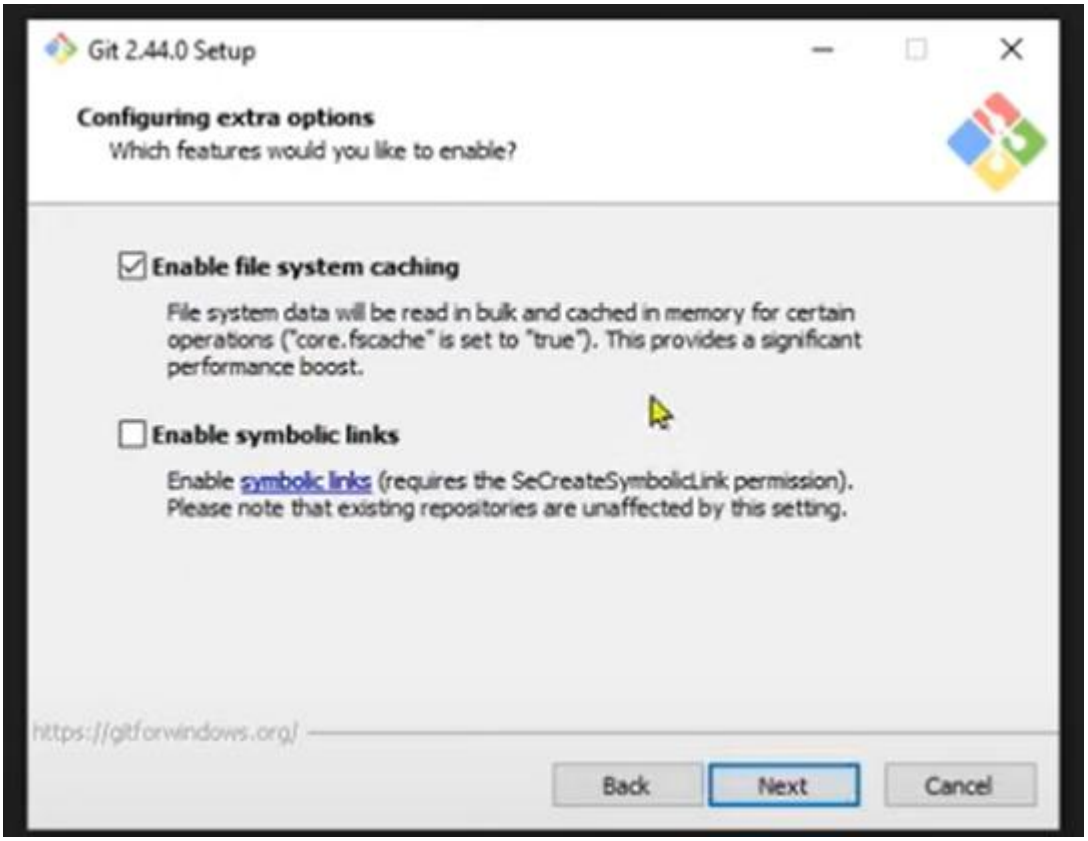
Esta pantalla del instalador de Git te permite seleccionar un "credential helper" (ayudante de credenciales), que es una herramienta que permite almacenar tus credenciales de acceso a repositorios remotos.

La opción seleccionada es **"Git Credential Manager"**, que significa:

- Git utilizará el Credential Manager multiplataforma para gestionar y almacenar de forma segura tus credenciales cuando te conectes a repositorios remotos.

Las ventajas de usar Git Credential Manager incluyen:

1. **Almacenamiento seguro**: Guarda tus nombres de usuario y contraseñas de manera segura en el sistema, utilizando el almacén de credenciales del sistema operativo.
2. **Soporte para autenticación de dos factores (2FA)**: Compatible con métodos modernos de autenticación, incluida la autenticación de dos factores que muchos servicios como GitHub requieren ahora.
3. **Integración con proveedores de servicios Git**: Funciona especialmente bien con GitHub, Azure DevOps, Bitbucket y otros servicios populares de Git.
4. **Multiplataforma**: Como indica el texto, es una herramienta que funciona en diferentes sistemas operativos.



Esta pantalla del instalador de Git te permite configurar opciones adicionales para mejorar el rendimiento y la funcionalidad.

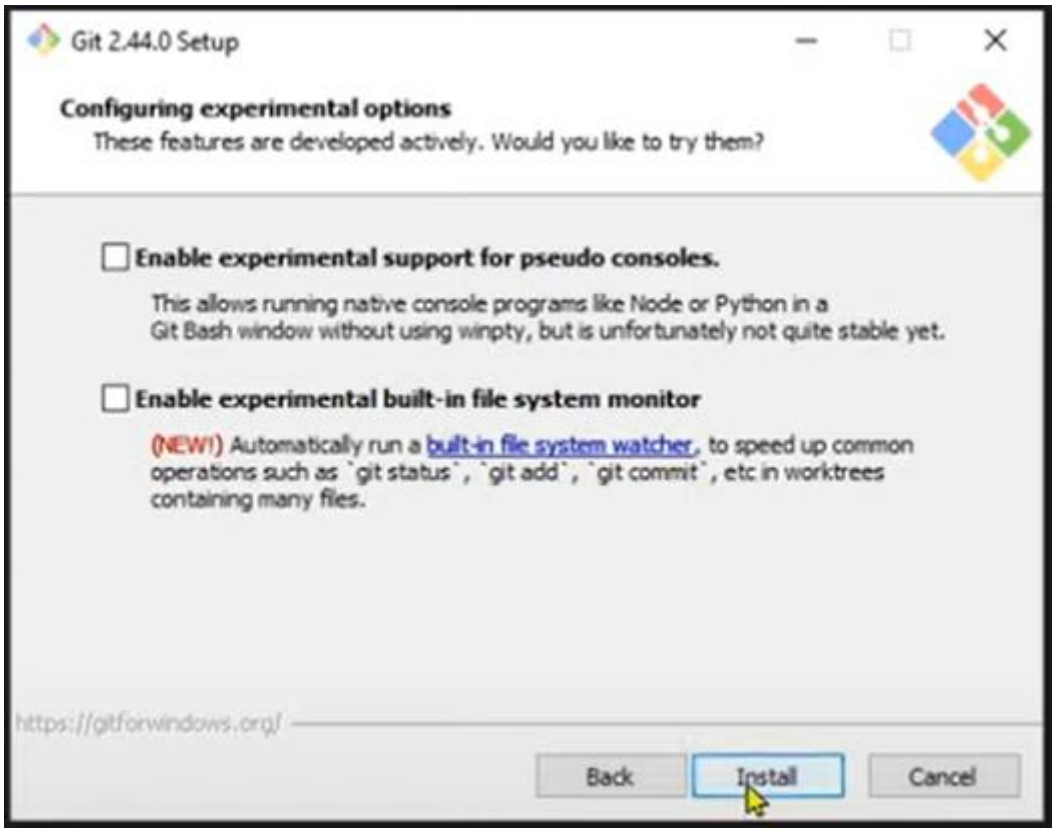
Hay dos opciones disponibles:

1. **Enable file system caching** (opción seleccionada):
    - Esta opción activa el almacenamiento en caché del sistema de archivos (core.fscache se establece en true).

- Los datos del sistema de archivos se leerán en bloque y se almacenarán en memoria para ciertas operaciones.
- Proporciona una mejora significativa de rendimiento, especialmente en repositorios grandes o cuando se realizan operaciones que acceden a muchos archivos.
- Es altamente recomendable mantener esta opción activada para una mejor experiencia.

2. **Enable symbolic links** (opción no seleccionada):
    - Esta opción permitiría a Git crear y gestionar enlaces simbólicos en tu sistema.
    - Requiere el permiso SeCreateSymbolicLink en Windows, que normalmente solo tienen las cuentas de administrador.
    - Los enlaces simbólicos pueden ser útiles en ciertos flujos de trabajo, pero también pueden presentar riesgos de seguridad si no se manejan adecuadamente.
    - Los repositorios existentes no se ven afectados por esta configuración.

Para la mayoría de los usuarios, la configuración mostrada (con solo la caché del sistema de archivos habilitada) es la más adecuada, ya que proporciona un buen rendimiento sin introducir posibles problemas de seguridad relacionados con los enlaces simbólicos. Si necesitas trabajar con enlaces simbólicos en tus proyectos, puedes habilitar esa opción, pero asegúrate de tener los permisos necesarios en tu sistema.



Características de desarrollo fundamentales, no seleccionar ninguna por el momento.

Comandos Base:

Configuración Inicial

Configurar nombre de usuario

git config --global user.name "Tu Nombre"

Configurar email

git config --global user.email "tu.email@ejemplo.com"

Ver configuración

git config –list

Comenzando con un Repositorio

Iniciar un nuevo repositorio

git init

Clonar un repositorio existente

git clone https://github.com/usuario/repositorio.git

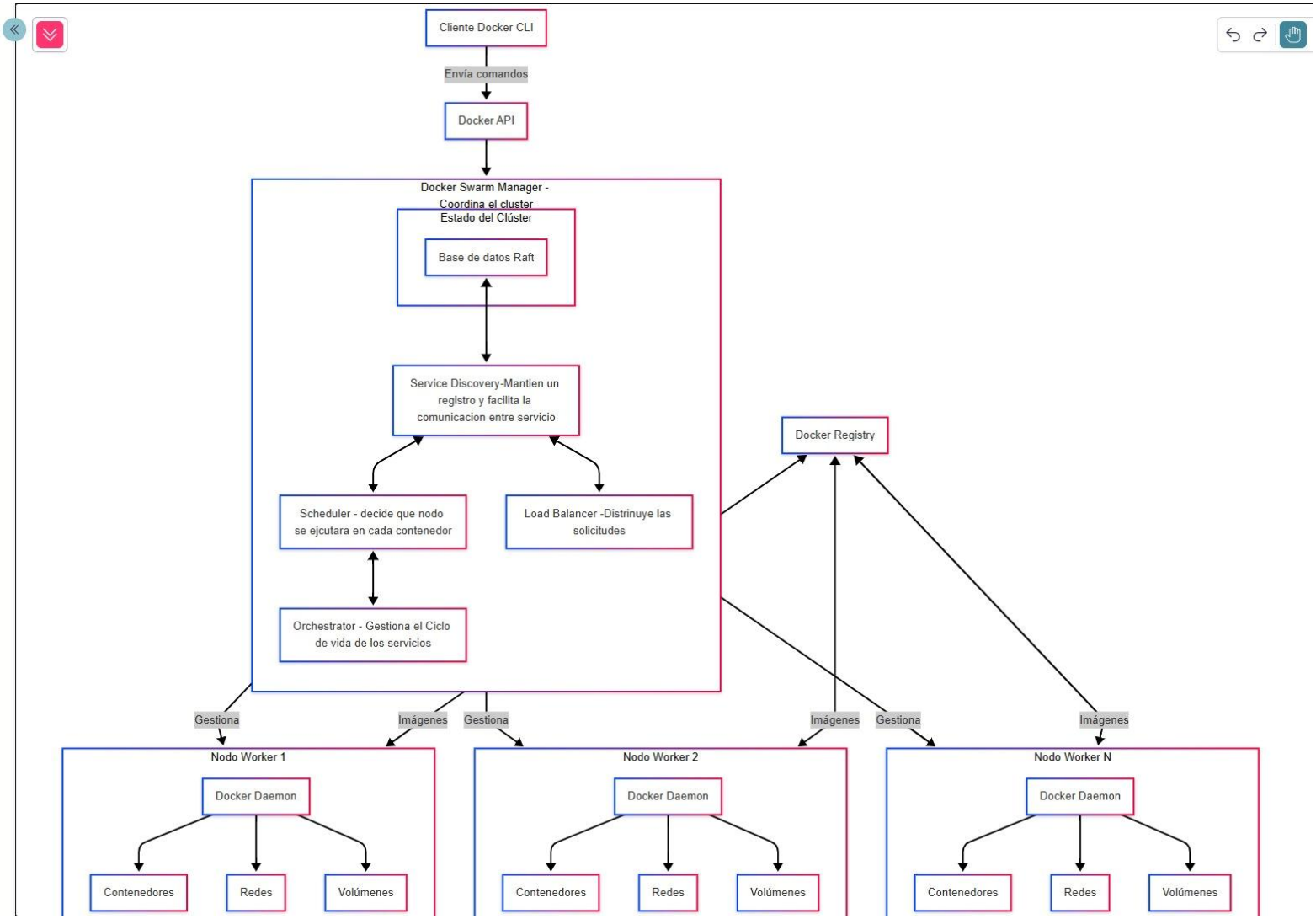
Operaciones Básicas

```
# Ver estado actual
git status
# Añadir archivos al área de preparación
git add archivo.txt      # Añadir un archivo específico
git add .                # Añadir todos los archivos modificados
# Confirmar cambios (commit)
git commit -m "Mensaje descriptivo del cambio"
# Ver historial de commits
git log
git log --oneline        # Formato resumido
```

Trabajando con Ramas

```
# Ver ramas existentes
git branch
# Crear una nueva rama
git branch nueva-rama
# Cambiar a otra rama
git checkout otra-rama
# O con el comando más nuevo
git switch otra-rama
# Crear y cambiar a una nueva rama en un solo paso
git checkout -b nueva-caracteristica
```

DOCKER:



Explicación del flujo de ejecución de Docker

1. El usuario ejecuta un comando a través del Docker CLI

Este es el punto de entrada donde un usuario escribe comandos como docker run, docker build o docker ps en la terminal. El CLI (Command Line Interface) proporciona una interfaz amigable para interactuar con Docker.

2. El CLI se comunica con la Docker API

El Docker CLI traduce los comandos del usuario en llamadas a la API REST de Docker. Esta comunicación generalmente ocurre a través de un socket Unix (en Linux/Mac) o un named pipe (en Windows).

3. La API envía la solicitud al Docker Swarm Manager

Cuando Docker está configurado en modo Swarm (orquestación de clústeres), las solicitudes pasan al Swarm Manager. Este es el nodo maestro que coordina las operaciones en todo el clúster de Docker.

4. El Manager procesa la solicitud y la dirige al nodo correspondiente

El Swarm Manager decide qué nodo trabajador debería manejar la solicitud basándose en la disponibilidad, restricciones y estrategias de programación definidas.

5. Si se necesita una imagen, el nodo la solicita al Docker Registry

Si la operación requiere una imagen que no está disponible localmente, el nodo la descarga desde un registro de Docker (como Docker Hub, o un registro privado).

6. El Docker Daemon en el nodo ejecuta la instrucción

El Docker Daemon (dockerd) es el servicio principal que:

- Gestiona el ciclo de vida de los contenedores
- Maneja la creación de imágenes
- Administra volúmenes, redes y otros recursos
- Interactúa con el sistema operativo para aislar y ejecutar contenedores

Docker Daemon - Componentes Principales

El Docker Daemon (dockerd) contiene varios componentes clave:

1. **containerd**: Gestiona el ciclo de vida de los contenedores (crear, iniciar, detener)
2. **runc**: Herramienta de bajo nivel para crear contenedores según la especificación OCI
3. **Controlador de almacenamiento**: Gestiona sistemas de archivos en capas (overlay2, aufs, etc.)
4. **Controlador de red**: Administra redes virtuales para la comunicación entre contenedores
5. **Controlador de volumen**: Maneja almacenamiento persistente para contenedores
6. **Motor de construcción**: Procesa las instrucciones Dockerfile para crear imágenes

7. **Administrador de recursos**: Limita y monitorea el uso de CPU, memoria, etc.
8. **API REST**: Expone interfaces para la comunicación con clientes

Este flujo y estructura permite que Docker funcione de manera distribuida, escalable y con una experiencia de usuario consistente.

Página Web Simple para Dockerizar:

```
<!DOCTYPE html>
<html>
<head>
  <title>Hola Mundo</title>
</head>
<body>
  <h1>¡Hola Mundo!</h1>
  <p>Esta es una página web muy simple.</p>
</body>
</html>
```

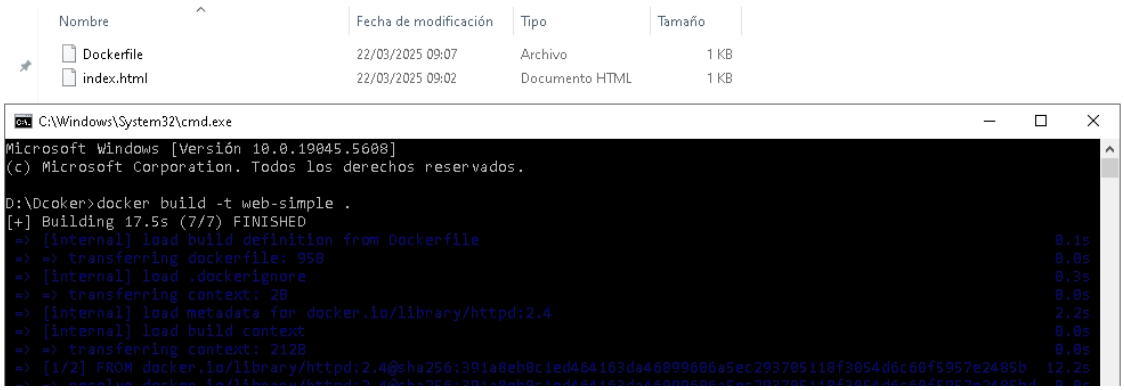
Dockerfile

```
FROM httpd:2.4
COPY index.html /usr/local/apache2/htdocs/
```

Paso 1: Crea estos dos archivos

1. Crea un archivo llamado index.html con el contenido mostrado arriba.
2. Crea un archivo llamado Dockerfile (sin extensión) con esas dos líneas.

Paso 2: Construye la imagen



docker build -t web-simple .

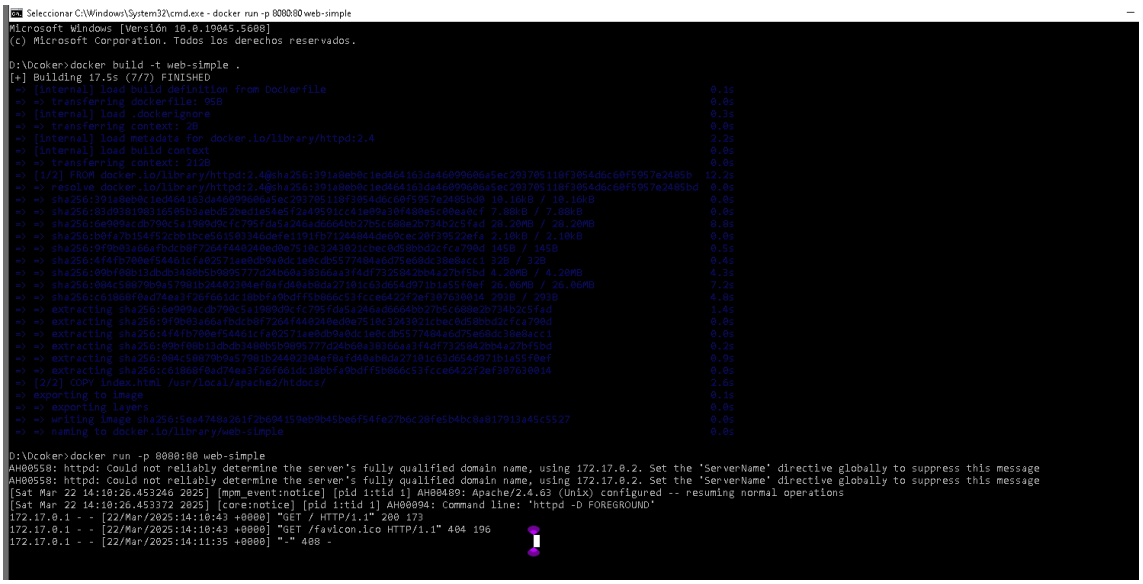
Este comando construye una imagen Docker usando el Dockerfile en el directorio actual (.) y le asigna el nombre web-simple.

Paso 3: Ejecuta el contenedor

docker run -p 8080:80 web-simple

Este comando:

- Crea y arranca un contenedor basado en la imagen web-simple
- Mapea el puerto 8080 de tu computadora al puerto 80 del contenedor

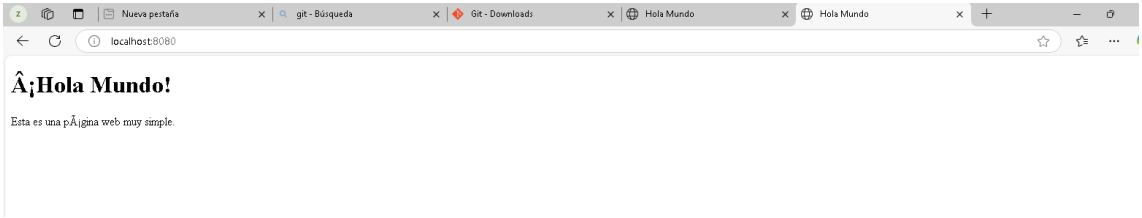


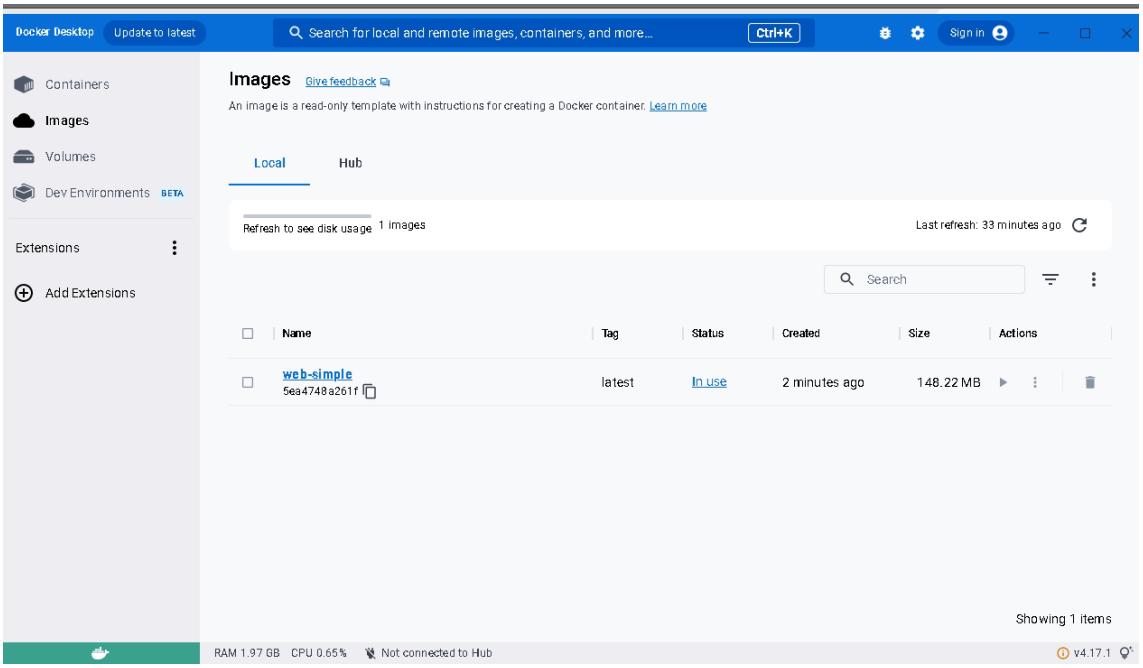
- **Primeras líneas con "sha256:"**: Son los identificadores (hash) de las capas de la imagen Docker que se están descargando o extrayendo. Docker usa un sistema de capas para construir imágenes de manera eficiente.
- **Líneas "Extracting"**: Docker está extrayendo las capas descargadas para crear el contenedor.
- **"[2/2] COPY index.html /usr/local/apache2/htdocs/"**: Muestra que Docker está ejecutando el comando COPY de tu Dockerfile, copiando tu archivo HTML al directorio correcto.
- **"Exporting to image"**: Docker está finalizando la construcción de la imagen.
- **"Naming to docker.io/library/web-simple"**: Docker está etiquetando la imagen con el nombre que le diste.
- **Mensajes de Apache (httpd)**:

    - **"Could not reliably determine the server's fully qualified domain name..."**: Esto es una advertencia común de Apache. No es un error, sino un aviso de que no puede determinar automáticamente el nombre de dominio completo del servidor. No afecta el funcionamiento.
    - **"AH00558: httpd: Could not reliably determine the server's fully qualified domain name..."**: Es el mismo mensaje de arriba, solo que con un código de identificación.
    - **"[core**

      ] [pid ttid 1] AH00094: Command line: "httpd -D FOREGROUND"": Muestra que Apache se está ejecutando en primer plano (esto es importante para que el contenedor no se detenga).

    - **"GET / HTTP/1.1" 200 173"**: Este mensaje indica que alguien (probablemente el navegador) ha solicitado la página principal y Apache respondió con un código de estado 200 (éxito), enviando 173 bytes.
    - **"GET /favicon.ico HTTP/1.1" 404 196"**: intentó automáticamente buscar un favicon (el icono pequeño que aparece en las pestañas), pero Apache respondió con un código 404 (no encontrado) porque no existe ese archivo en tu contenedor.





1. **Ver todos los contenedores en ejecución**:

```
docker ps
```

2. **Ver todos los contenedores (incluyendo los detenidos)**:

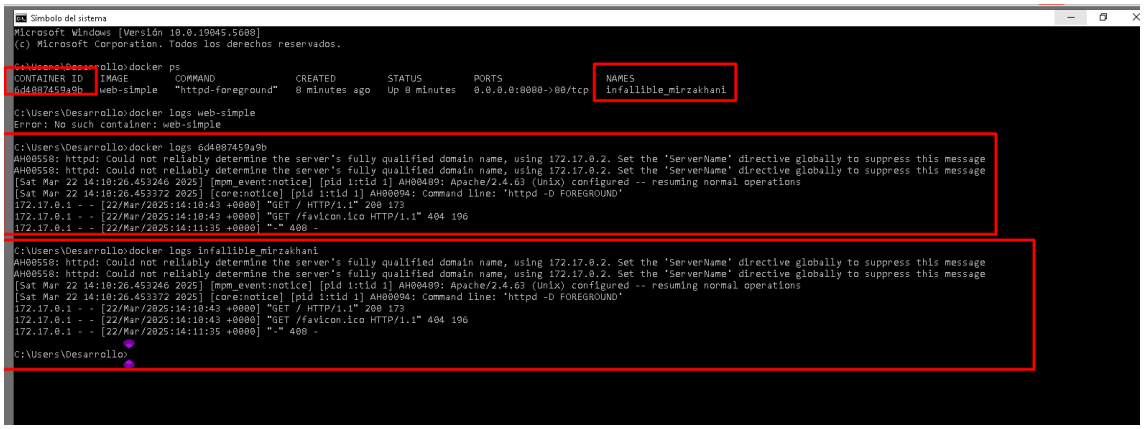
```
docker ps -a
```

3. **Ver los logs de un contenedor**:

```
docker logs web-simple
```

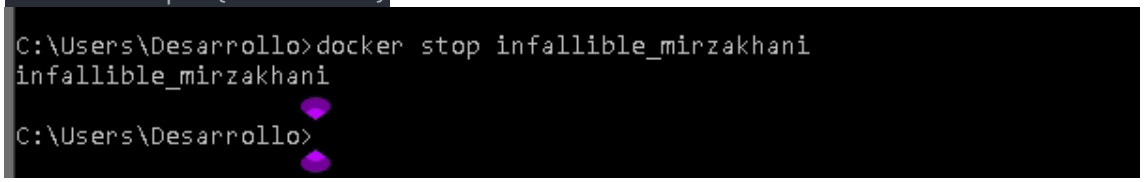
4. **Ver los logs y seguirlos en tiempo real**:

```
docker logs -f {name}
```



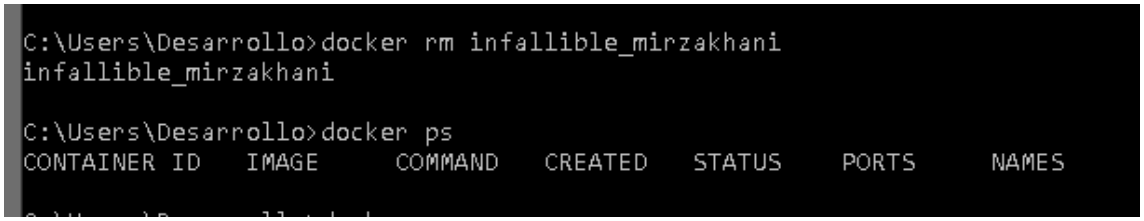
5. **Detener un contenedor**:

```
docker stop  {name o id}
```



6. **Eliminar un contenedor (primero debe estar detenido)**:

```
docker rm {name o id}
```



7. **Eliminar un contenedor forzosamente (incluso si está en ejecución)**:

```
docker rm -f  {name o id}
```

8. **Reiniciar un contenedor**:

```
docker restart {name o id}
```

9. **Ver estadísticas de uso de recursos de los contenedores**:

`docker stats`

10. **Ejecutar un comando dentro de un contenedor en ejecución**:

El comando docker stats muestra estadísticas en tiempo real sobre el rendimiento y uso de recursos de tus contenedores Docker en ejecución. Es una herramienta de monitoreo muy útil que proporciona la siguiente información:

- **CONTAINER ID:** El identificador del contenedor
- **NAME:** El nombre del contenedor
- **CPU %:** Porcentaje de uso de CPU
- **MEM USAGE / LIMIT:** Uso de memoria y límite configurado
- **MEM %:** Porcentaje de uso de memoria respecto al límite
- **NET I/O:** Entrada/salida de red (datos recibidos y enviados)
- **BLOCK I/O:** Entrada/salida de disco (lectura/escritura)
- **PIDS:** Número de procesos en ejecución dentro del contenedor

Este comando es especialmente útil para:

1. Detectar contenedores que consumen demasiados recursos
2. Identificar posibles cuellos de botella en rendimiento
3. Monitorear la salud general de tus aplicaciones contenerizadas
4. Verificar que tus contenedores tienen recursos suficientes para funcionar correctamente

Recrear la imagen con nombre especifico:

```
docker build -t appweb .
```

creando contenedor también con nombre

```
docker run -d -p 8080:80 --name mi-aplicacion appweb
```

Iteramos dentro del contendor:

docker exec -it mi-aplicacion /bin/bash

Para modificar la página web y mostrar tu nombre, sigue estos pasos:

1. Primero, navega al directorio donde se encuentra el archivo HTML si no estás ya allí:

```
bash
Copiar
cd /usr/local/apache2/htdocs/
```

2. Puedes ver el contenido actual del archivo:

```
bash
Copiar
cat index.html
```

3. Para modificar el archivo, necesitas un editor de texto. Apache en su versión ligera normalmente no incluye editores, así que primero debes instalar uno:

```
bash
Copiar
apt-get update
apt-get install -y nano
```

4. Ahora abre el archivo con nano:

```
bash
Copiar
nano index.html
```

5. Edita el archivo para incluir tu nombre. Por ejemplo, cambia:

```
html
Copiar
<h1>¡Hola Mundo!</h1>
```

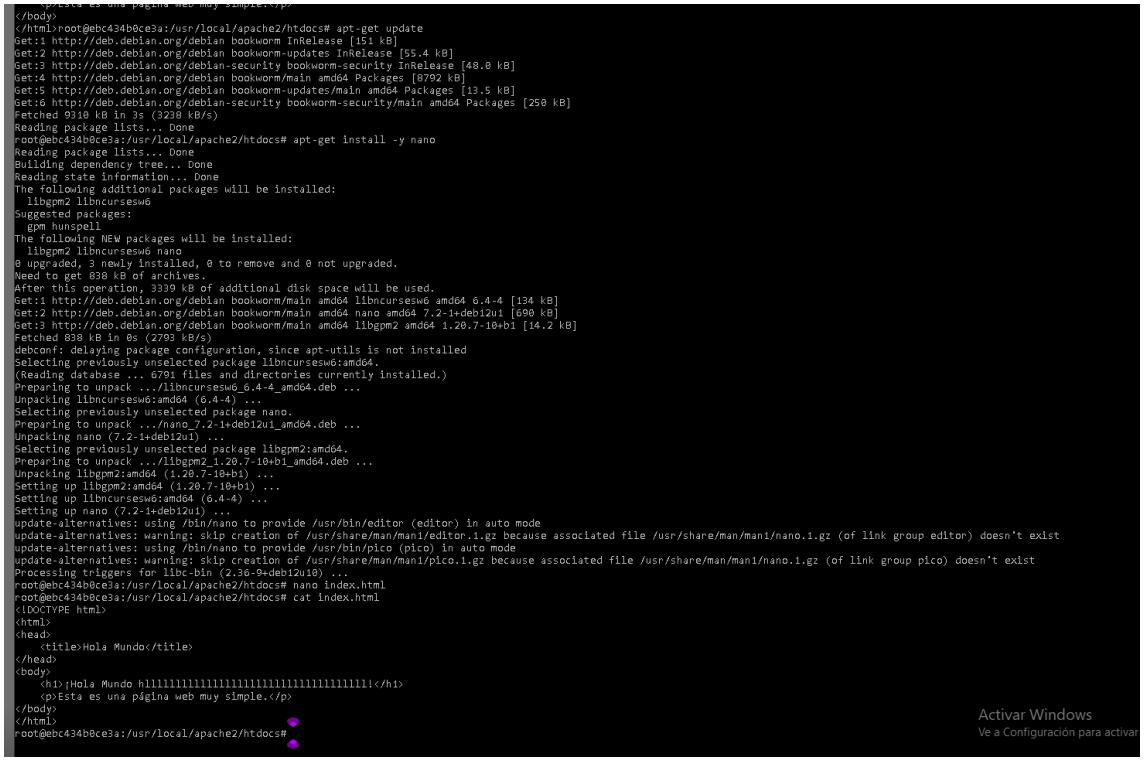
a

```
html
Copiar
<h1>¡Hola Mundo! Soy TU_NOMBRE</h1>
```

6. Para guardar en nano, presiona `Ctrl+O`, luego Enter, y para salir presiona `Ctrl+X`.
7. Verifica los cambios:

```
bash
Copiar
cat index.html
```

8. Ahora puedes acceder a tu sitio web actualizado en http://localhost:8080 en tu navegador.


Activar Windows
Ve a Configuración para activar