

## What Is Git?

Before diving into the installation steps, let's take a brief moment to understand what Git is all about. Created by [Linus Torvalds](#) in 2005, [Git](#) is a distributed version control system that helps you keep track of changes in your source code and allows for seamless collaboration among developers.

It is a powerful tool that stores your source code in repositories, which are just like folders where you can store files. When multiple developers are working on the same project, they can all “push” and “pull” changes to the repository to keep everyone's version of the code up-to-date.

## Who Uses Git?

Git is used by a wide range of individuals, including software developers, open-source project contributors, and system administrators. It's an essential tool for managing large codebases, coordinating work among team members, and maintaining a project's history.

Many well-known companies use Git, including:

- [Microsoft](#): Git powers the version control behind many of Microsoft's software projects, including the open-source .NET Core and [Visual Studio Code](#).
- [Google](#): Google relies on Git for managing the source code for Android, Chromium, and numerous other projects.
- [Adobe](#): Adobe utilizes Git for version control and collaboration across their software development teams.
- [Airbnb](#): Git is an integral part of Airbnb's software [development workflow](#), enabling their engineering teams to collaborate on various projects.

## Advantages of Using Git

Git offers several advantages that make it the go-to choice for version control and collaboration. Some of these benefits include:

- Efficient and fast: Git's fast and efficient operations make it ideal for managing large, complex codebases with many contributors. Its design prioritizes performance while minimizing resource usage.
- Distributed nature allows for offline work: [Developers](#) can work on their local Git repositories by making changes and committing them even when offline due to Git's distributed nature. After reconnecting to the internet, the changes can be synchronized with the remote repository.
- Branching and merging capabilities: The use of Git [branching and merging](#) features allows developers to work on distinct features or bug fixes in separate branches to maintain stability in the main codebase. Merging changes back into the main branch is a simple process.
- Collaboration support: Git is built with collaboration in mind, allowing multiple developers to work on the same project simultaneously without conflicts. It also helps in tracking changes and attributing them to the correct contributor.
- Widely used and supported: The Git version control system is popular and widely used by many developers. Its extensive documentation and large community make it a versatile choice, with many tools and services built for integration.

## Git Prerequisites

Before we jump into the installation process, let's quickly go over some prerequisites to ensure you're all set to get started with Git. Don't worry, you won't need much:

1. Basic computer knowledge: Familiarity with using a computer is essential. You don't need to be an expert, but knowing your way around your operating system and file management is helpful.
2. Command line/terminal experience: Git relies heavily on the command line (or terminal) for its operation. If you haven't used the command line before, you might want to brush up on some basics. Don't stress, though – you'll pick it up as you go along.
3. Programming experience (optional): While having some [programming](#) background can be helpful when working with Git, it's not mandatory. If you're new to coding, Git can still be a valuable tool for managing your projects as you learn.

## How To Install Git

In this section, we'll guide you through the process of installing Git on your system. We'll start with installing Git on Windows, followed by macOS and Linux.

### How To Install Git on Windows

Installing Git on Windows is a fairly straightforward process and involves the following steps:

1. Download the Windows installer
2. Run the installer
3. Verify installation

#### Step 1: Download the Windows Installer

Visit the [official Git website](#) to download the latest version of the Git installer for Windows. The download should start automatically when you visit the page.

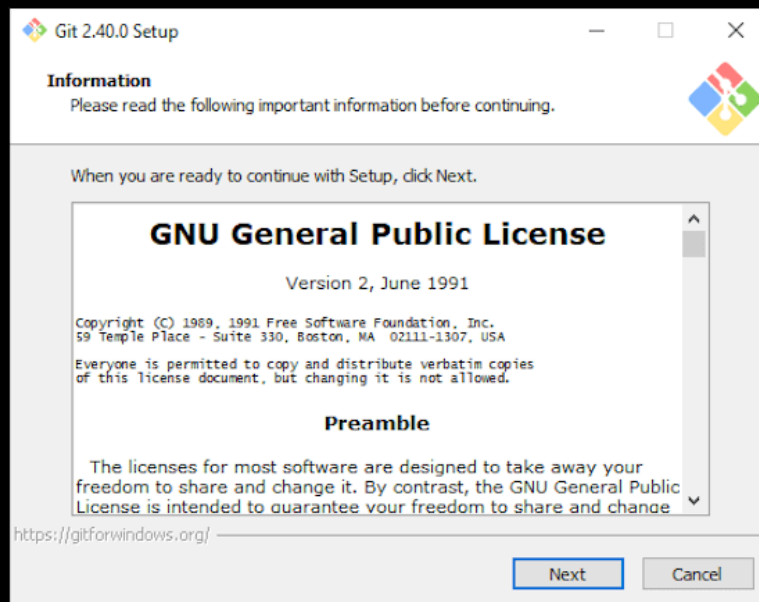


Downloading Git for Windows.

## Step 2: Run the Installer and Select Options

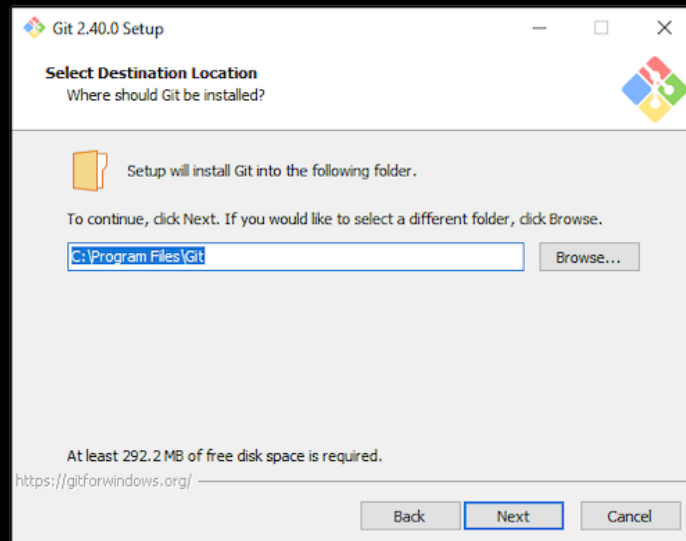
Launch the downloaded installer and follow the installation wizard. Confirm that the app can make changes to your device by clicking Yes on the User Account Control dialog that appears.

Read the GNU General Public License, and when you're prepared to install, click Next.



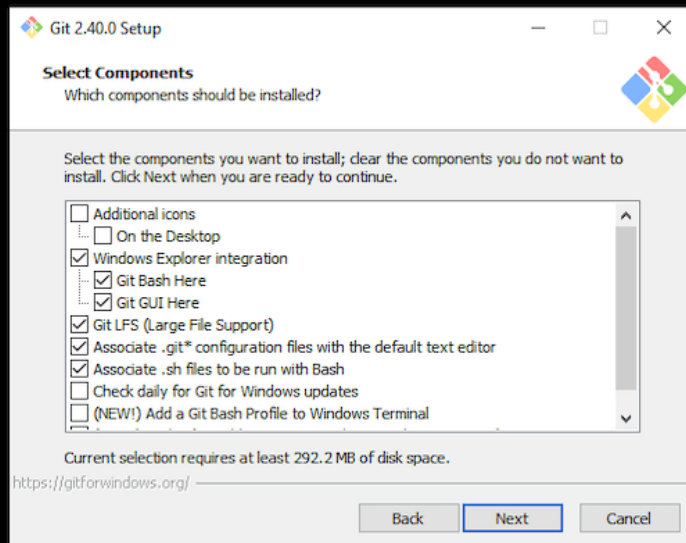
Accept the GNU license.

The installer will request an installation location. Keep the default unless you need to change it, and click Next.



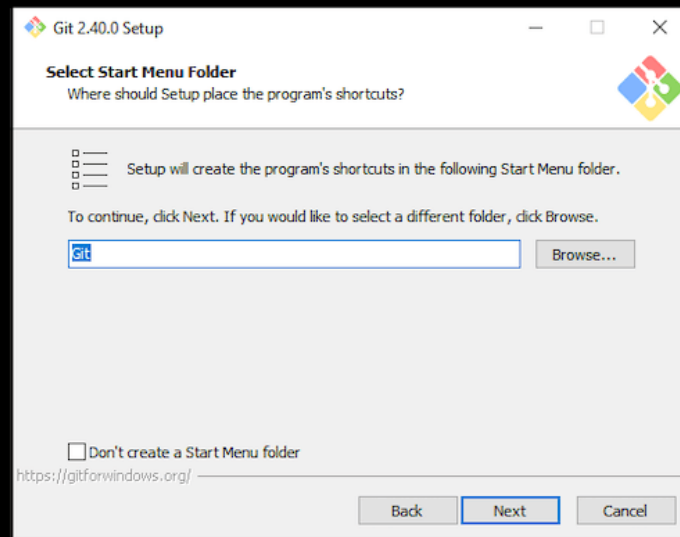
Select the destination location for Git on your computer.

A component selection screen will be displayed. Keep the default settings unless you need to modify them, and click Next.



Select the components you'd like to install with Git.

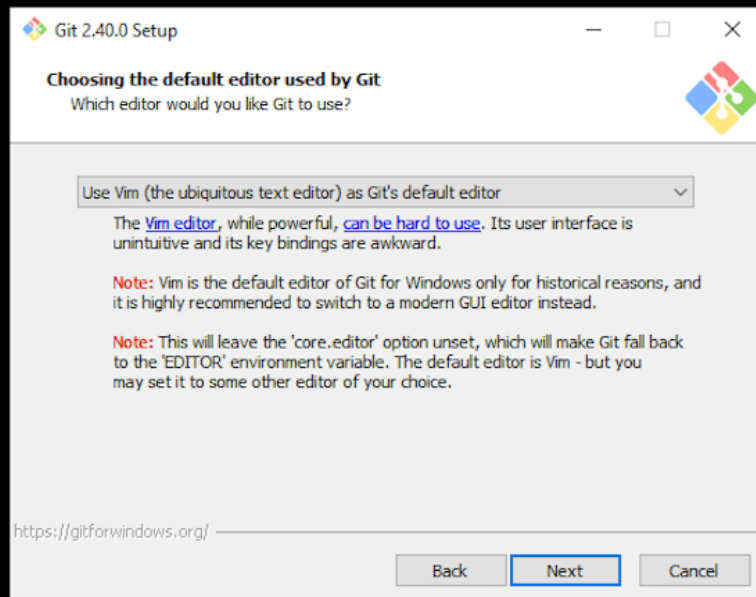
You'll be prompted to create a start folder. Leave it as is and click Next.



You can adjust the start folder name if you'd like.

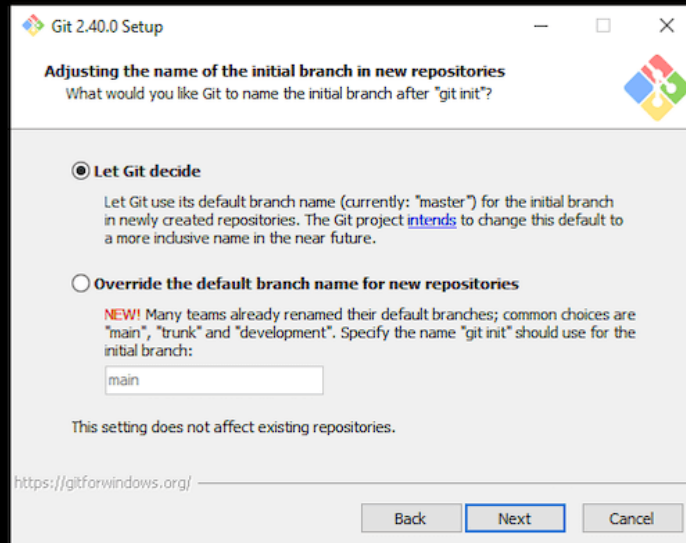
Choose a text editor to use with Git. Click on the drop-down menu to pick the text editor you like to use like Vim, Notepad++, etc and click Next.





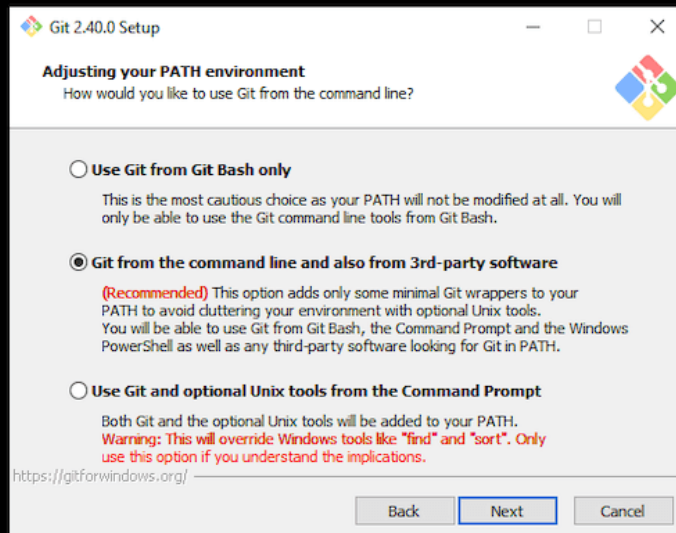
Select the default editor used by Git.

In this next step, you can opt to **rename** your initial branch. The default is master. Leave the default (unless you'd really like to change it) and click Next.



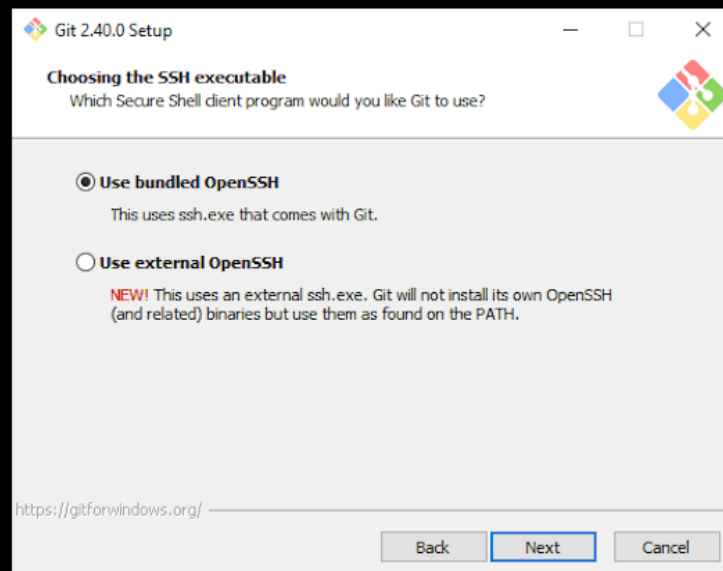
You can adjust the name of the initial branch in new repositories.

Now you're on to modifying the PATH environment. Leave this on the recommended selection, Git from the command line and also from 3rd party software and click Next.



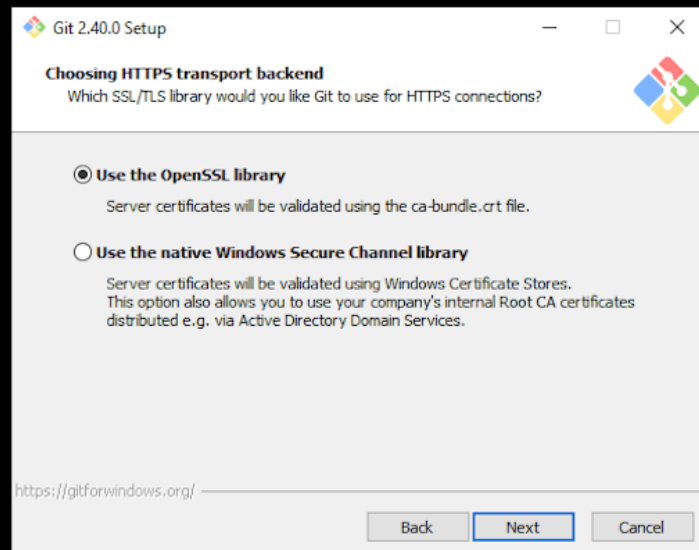
Selecting the path environment

On the next screen, you need to pick the [SSH program](#) you want to use. Git does come with its own SSH client, so leaving the default settings checked is fine. Click Next.



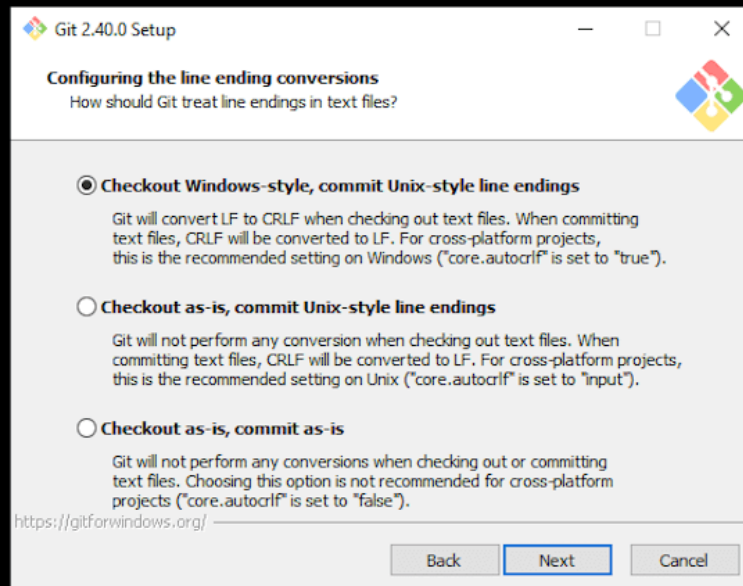
Choosing the SSH executable in the Windows Git installer.

The following option concerns server certificates. The majority of users should choose the default, Use the OpenSSL library. Click Next.



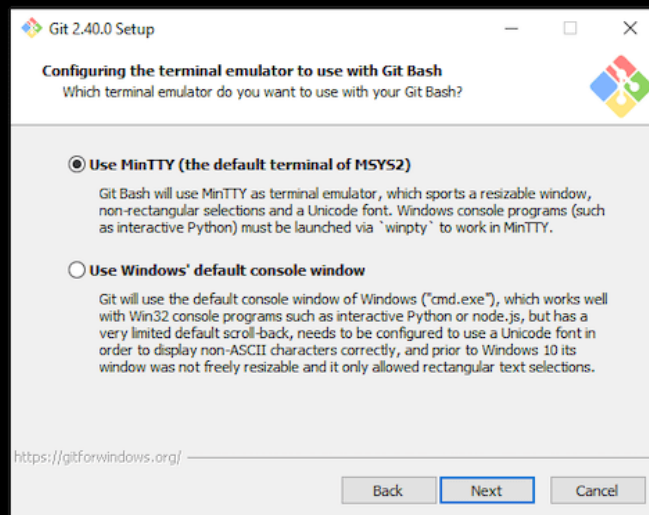
Most should select Use the OpenSSL Library

The next screen deals with line ending conversions. Leave it set to the default option, Checkout Windows-style, commit Unix-style line endings. Changing this option might cause issues. Click Next.



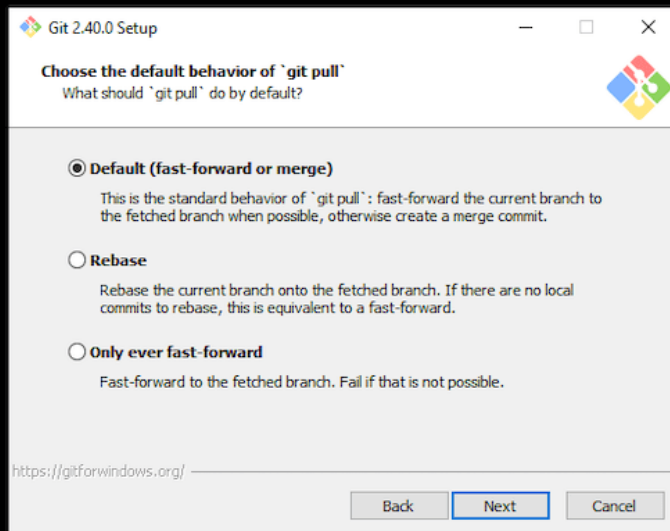
Configuring the line ending conversions.

Next up, you'll need to select the terminal emulator. The default MinTTY is recommended. Click Next.



Configuring the terminal emulator to use with Git Bash.

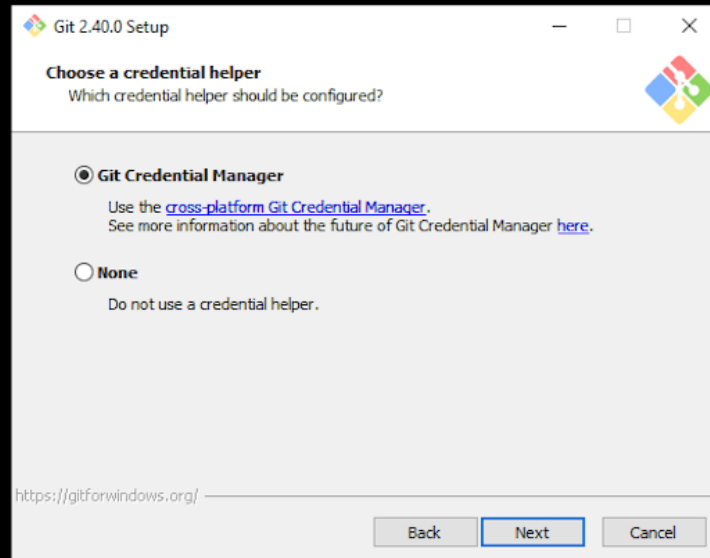
On the next screen, you'll be asked what the git pull command should do. Again, the default option is recommended. Click Next.



Select the default behavior of the 'git pull' command.

You'll need to choose which credential helper to use next. Keep the default option selected and click Next.



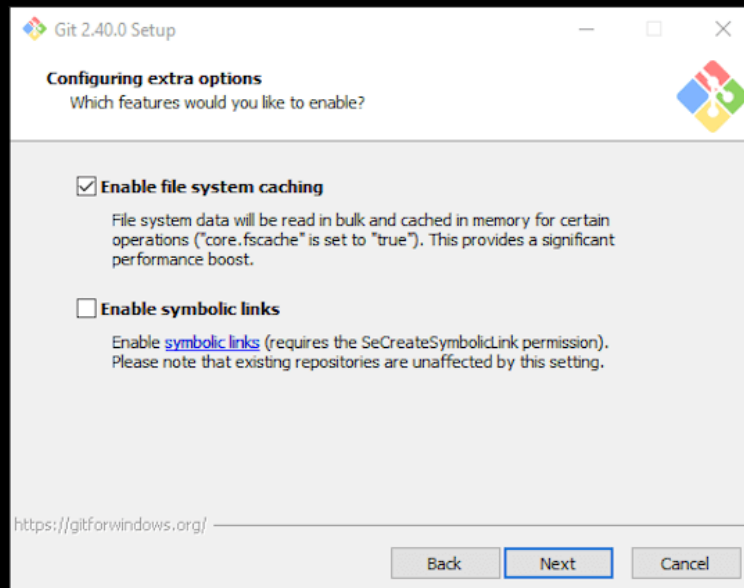


Choosing a credential helper.

Next, you'll be presented with some extra options to customize your installation, including:

- Enable file system caching
- Enable symbolic links

Once done making your selections, click Next.

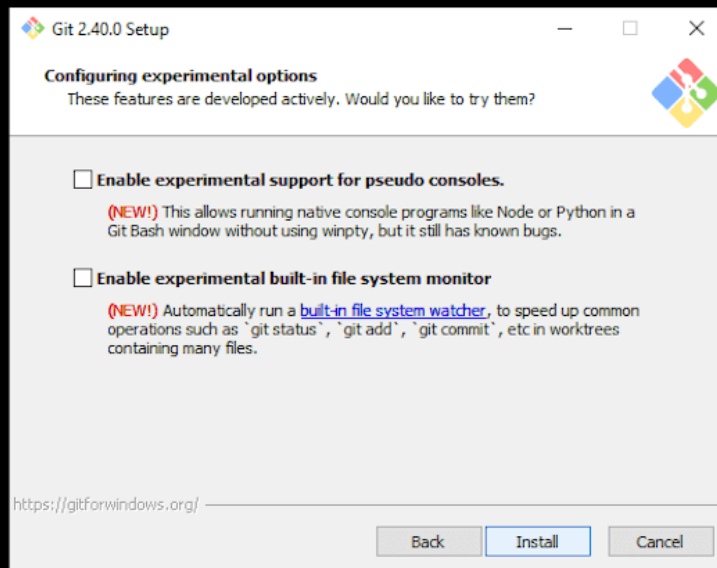


Select extra options.

If you're installing a newer version of Git, it'll next prompt you to select experimental features. As of this writing, the options include:

- Support for [pseudo consoles](#)
- Built-in file system monitor

Check the boxes to whichever you'd like (or none), then click Install.



Configuring experimental options

On the last screen of the installer, you can opt to view the Release Notes or Launch Git Bash. Check the boxes next to the options you prefer, then click Finish.

### Step 3: Verify the installation with Git Bash

To ensure that Git has been installed correctly, open Git Bash and type the following command:

Unset

```
git --version
```

Press Enter, and the name of the version of Git you just installed should appear.

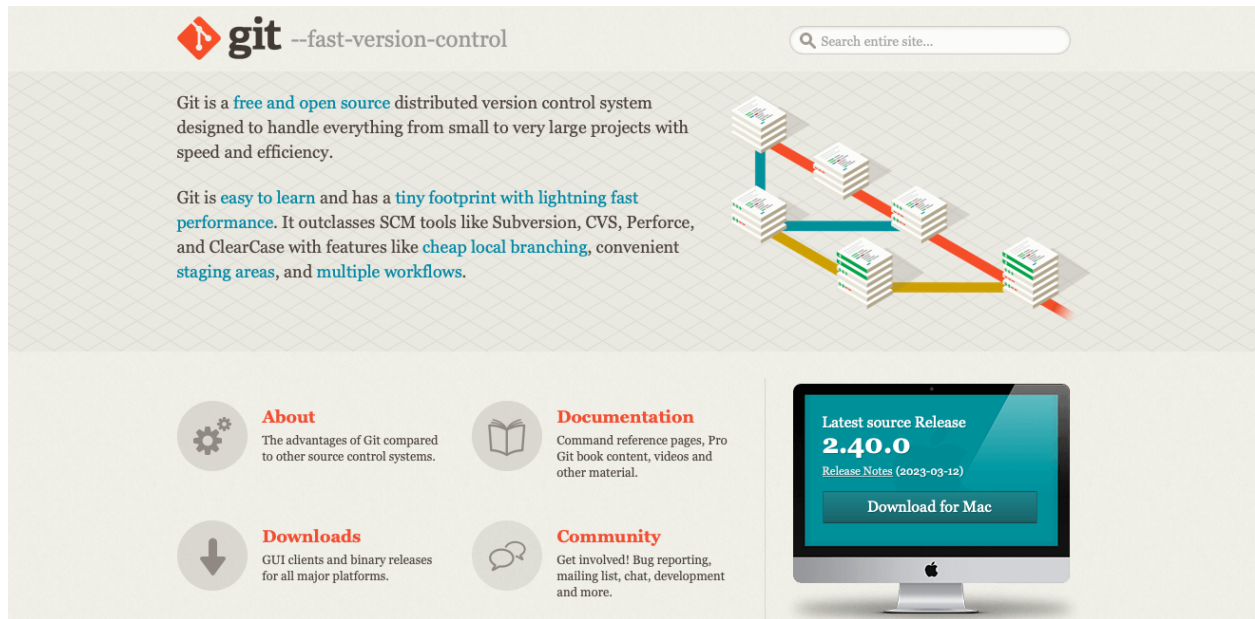
## How To Install Git on macOS

Just as with the installation process for Windows, installing Git on a Mac is straightforward as well:

1. [Download the macOS installer](#)
2. [Complete installation](#)
3. [Install via Homebrew \(optional\)](#)
4. [Verify installation through Terminal](#)

### Step 1: Download the macOS Installer

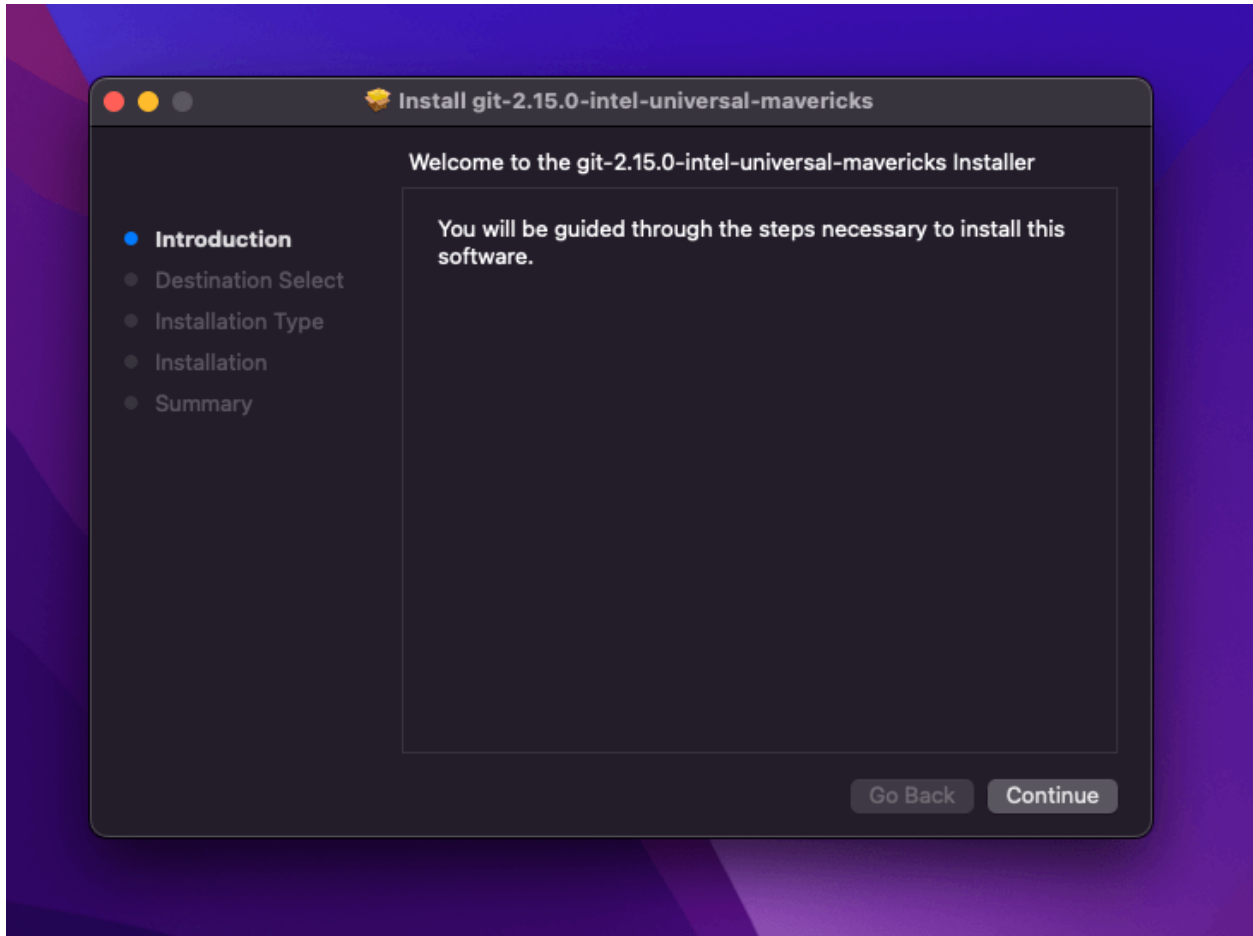
Visit the [official Git website](#) to download the latest version of the Git installer for macOS.



Downloading Git for macOS.

## Step 2: Complete Installation Instructions

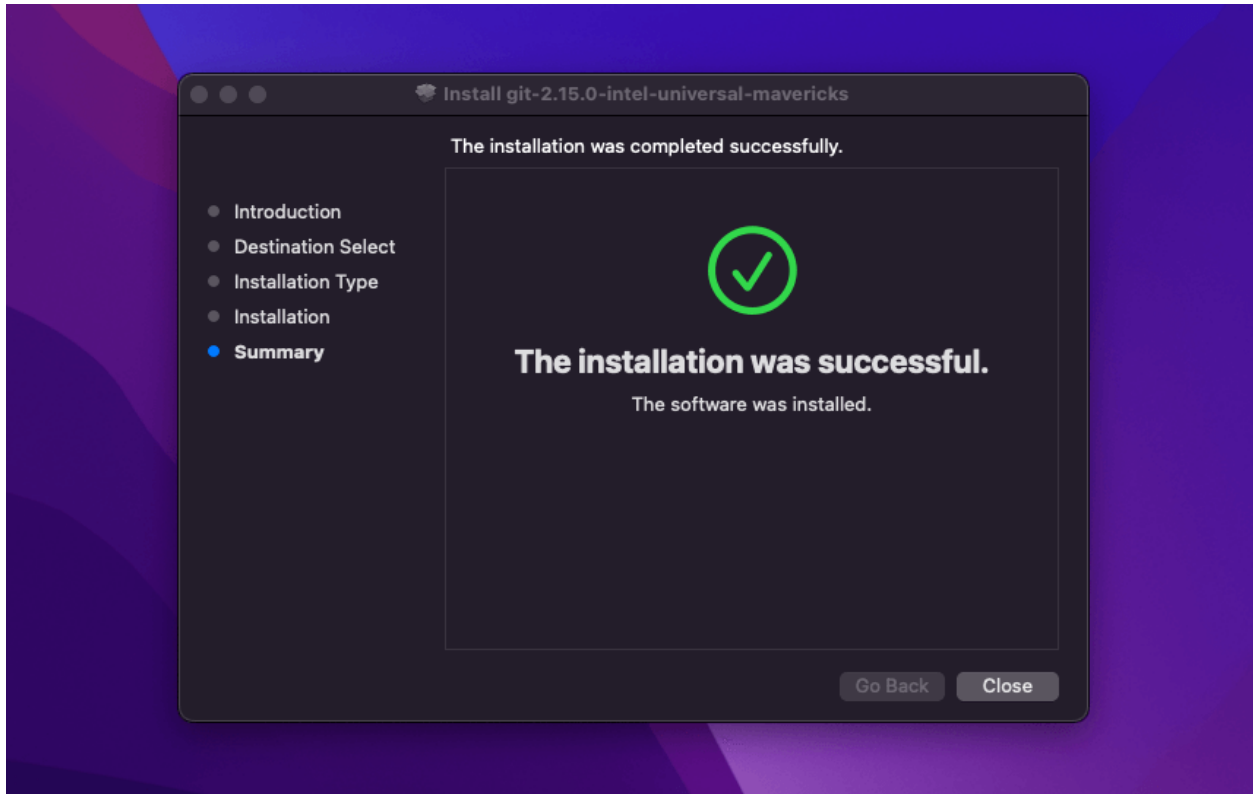
Once the installer is downloaded, open the .dmg file and follow the installation instructions. You will be guided through the process. On the first screen, click Continue.



Git installer for macOS.

Then select a destination for where you'd like Git to be located on your system. Confirm your selection by clicking Install. you'll then be prompted to input your system password.

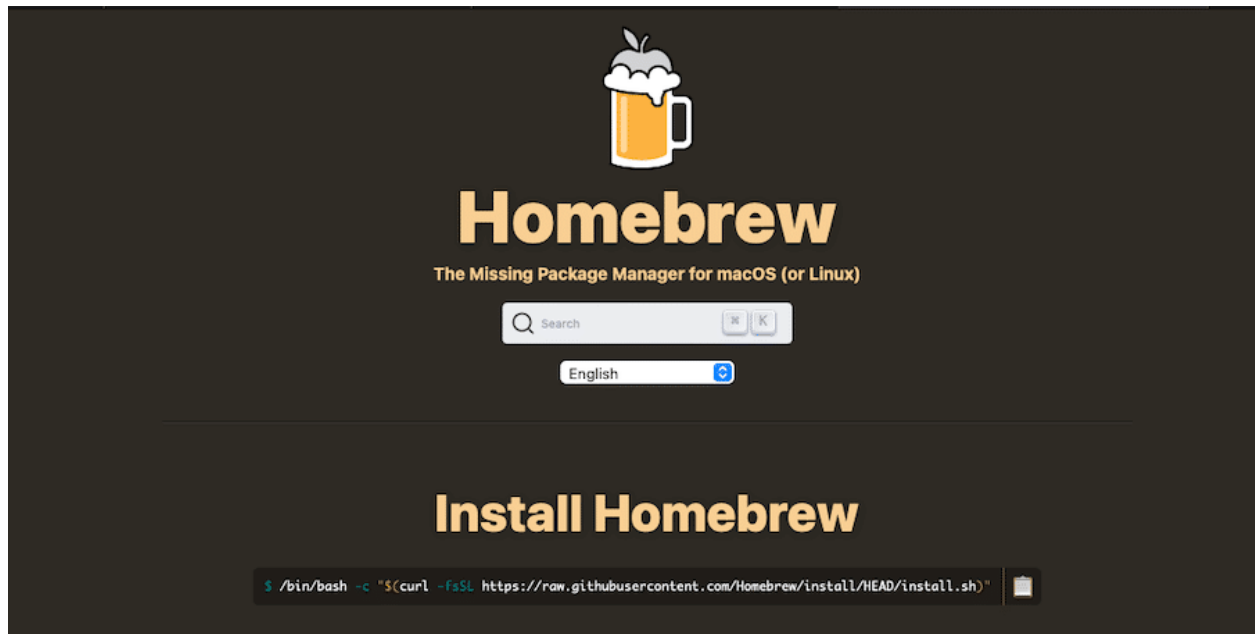
After a few moments, the installation will be complete and you'll be presented with the following confirmation:



A confirmation message that Git installation was successful.

### Step 3: Install via Homebrew (optional)

If you prefer, you can install Git using [Homebrew](#), a popular package manager for macOS. If you haven't already installed Homebrew, you can do so easily by first checking your macOS version.



The Homebrew website.

You can check your macOS version by clicking on the Apple menu in the top-left corner of your screen and selecting About This Mac. You need to have 10.9 (Mavericks) or newer to install Homebrew.

Then open Terminal and install Xcode Command Line Tools. These are required for Homebrew to function properly. With the Terminal open, input the following:

Unset

```
xcode-select --install
```

Follow the on-screen instructions to complete the installation.



Then to install Homebrew, all you need to do is then paste the following command in Terminal:

Unset

```
/bin/bash -c "$(curl -fsSL  
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.s  
h)"
```

Press Enter to run the command. The installation script will download and install Homebrew on your system.

You can verify installation worked by running the following in Terminal:

Unset

```
brew --version
```

If the installation was successful, you should see the Homebrew version number displayed.

Once you have Homebrew installed, you can install Git with the following command:

Unset

```
brew install git
```

Homebrew will download and install the latest version of Git on your system.

#### **Step 4: Verify Installation With Terminal**

To ensure that Git has been installed correctly, open Terminal and type the following command:

Unset

```
git --version
```

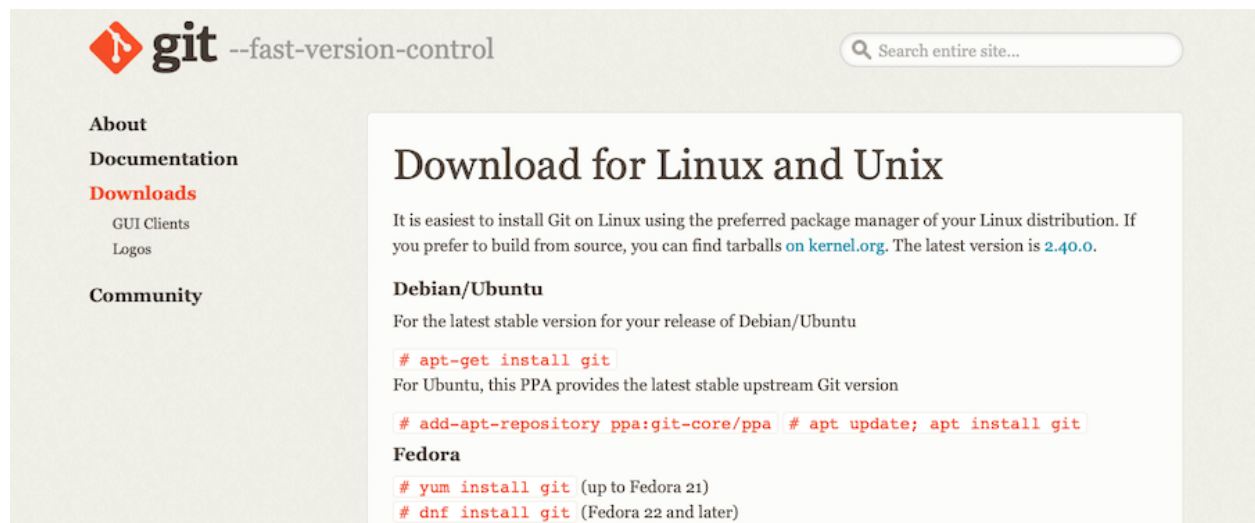
Press Enter, and you should see the version of Git you installed displayed on the next line.

## How To Install Git on Linux

Installing Git on a [Linux system](#) looks a bit different than the other processes detailed here for Windows and macOS. To complete this process, you can expect to:

1. [Install via package manager](#)
2. [Verify the installation](#)

### Step 1: Install Via Package Manager (apt, yum, etc.)



The screenshot shows the Git website's "Download for Linux and Unix" page. The header features the Git logo and the tagline "--fast-version-control", along with a search bar. A left sidebar contains links for "About", "Documentation", "Downloads" (highlighted), "GUI Clients", "Logos", and "Community". The main content area is titled "Download for Linux and Unix" and explains that the easiest way to install Git on Linux is using the preferred package manager. It provides instructions for Debian/Ubuntu, including the command `# apt-get install git`, and for Fedora, including `# yum install git` (up to Fedora 21) and `# dnf install git` (Fedora 22 and later).

**git** --fast-version-control

Search entire site...

**About**  
**Documentation**  
**Downloads**  
GUI Clients  
Logos  
**Community**

## Download for Linux and Unix

It is easiest to install Git on Linux using the preferred package manager of your Linux distribution. If you prefer to build from source, you can find tarballs [on kernel.org](#). The latest version is **2.40.0**.

### Debian/Ubuntu

For the latest stable version for your release of Debian/Ubuntu

```
# apt-get install git
```

For Ubuntu, this PPA provides the latest stable upstream Git version

```
# add-apt-repository ppa:git-core/ppa # apt update; apt install git
```

### Fedora

```
# yum install git (up to Fedora 21)
# dnf install git (Fedora 22 and later)
```

Git installation instructions for Linux on the Git website.

The easiest way to install Git on Linux is through the package manager for your distribution. For Debian-based distributions like [Ubuntu](#), you can use the apt package manager:

Unset

```
sudo apt-get install git
```

For Red Hat-based distributions like Fedora, you can use the yum or dnf package manager:

Unset

```
sudo yum install git
```

or

Unset

```
sudo dnf install git
```

## Step 2: Verify the Installation

And as always, your last step in the installation process is to verify this. To do this, open Terminal and type in the following:

Unset

```
git --version
```

On the next line, the Git version you just installed will be listed.

## Git Configuration

After successfully installing Git on your system, it's important to configure your Git settings to ensure a smooth and efficient workflow – and to ensure it can be used properly with [GitHub](#). The following steps apply to all operating systems (Windows, macOS, and Linux).

### Step 1: Set Your Name and Email Address

To set your name and email address for Git, open your terminal or Git Bash (for Windows users) and enter the following commands, replacing Your Name and youremail@example.com with your actual name and email address:

Unset

```
git config --global user.name "Your Name"
```

Unset

```
git config --global user.email "youremail@example.com"
```

These settings will be used to identify your commits in the Git history.

### Step 2: Configure Line Endings (Optional)

Git can [automatically handle line endings](#) depending on your operating system. This is important because Windows and Unix-based systems (such as macOS and Linux) use different line ending characters.

To configure Git to handle line endings appropriately for your system, run the following command:

For Windows users:

Unset

```
git config --global core.autocrlf true
```

For macOS and Linux users:

Unset

```
git config --global core.autocrlf input
```

### Step 3: Verify Your Configuration

To verify that your Git configuration is set up correctly, run the following command:

Unset

```
git config --list
```

This command will display your current Git configuration settings. Ensure that your name, email address, text editor, and line ending settings are correct.

With your Git configuration complete, you're now ready to start using Git for your projects.