



Open in app

Get started



Published in DevOps with Valentine

You have 1 free member-only story left this month. [Sign up for Medium and get an extra one](#)



Valentin Despa

Follow

May 25, 2021 · 6 min read · ✨ · Listen

Save



# [2022] How to install Git on Windows 10 / 11 (step by step guide)

If you want to be able to collaborate on Git projects from your own computer, you need to have Git installed. It's not hard. Let me guide you through the process.

*Latest update: May 2022*

## Step 1 — Install Git

Open any terminal and check if you already have Git installed by typing:

```
git --version
```

If you are getting back an error message, you need to install Git. I would anyway recommend installing/updating Git anyway.



[Open in app](#)[Get started](#)

```
script file, or operable program. Check the spelling of the name, or if a
path was included, verify that the path is correct and try again.
At line:1 char:1
+ git --version
+ ~~~
+ CategoryInfo          : ObjectNotFound: (git:String) [], CommandNotFou
ndException
+ FullyQualifiedErrorId : CommandNotFoundException
```

Git is not present on the system yet.

Go ahead and open <https://git-scm.com/>. The latest version I see is 2.30.0.

Now open the installer you have downloaded and go through the installation process. Unless you know what you are doing, **leave all settings to their defaults.**

### License Information

You need to accept the GNU GPL open source license to continue.



[Open in app](#)[Get started](#)

### Information

Please read the following important information before continuing.



When you are ready to continue with Setup, click Next.

## GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.  
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies  
of this license document, but changing it is not allowed.

### Preamble

The licenses for most software are designed to take away your  
freedom to share and change it. By contrast, the GNU General Public  
License is intended to guarantee your freedom to share and change

<https://gitforwindows.org/>

Next

Cancel

### Select Destination Location

Unless otherwise instructed by your system administrator, use the installation path suggested by Git.



[Open in app](#)[Get started](#)

## 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.

At least 264.0 MB of free disk space is required.

<https://gitforwindows.org/>

## Select Components

Apart from the pre-selected options, you may also want to select “Add a Git Bash Profile to Windows Terminal”. This can be useful later on, especially if you plan to use Visual Studio Code or other IDEs that have a built-in terminal window.



[Open in app](#)[Get started](#)

## Select Components

Which components should be installed?



Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- Additional icons
  - On the Desktop
- Windows Explorer integration
  - Git Bash Here
  - Git GUI Here
- Git LFS (Large File Support)
- Associate .git\* configuration files with the default text editor
- Associate .sh files to be run with Bash
- Check daily for Git for Windows updates
- (NEW!) Add a Git Bash Profile to Windows Terminal

Current selection requires at least 263.9 MB of disk space.

<https://gitforwindows.org/>

[Back](#)[Next](#)[Cancel](#)

## Start Menu Folder

At this step, you can select the name of the start menu folder. The default value is Git. Right below, you have the option of not creating such an entry.



[Open in app](#)[Get started](#)

### Select Start Menu Folder

Where should Setup place the program's shortcuts?



Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

Git

Browse...

Don't create a Start Menu folder

<https://gitforwindows.org/>

Back

Next

Cancel

### The default editor used by Git

Sometimes Git will ask you to insert or edit some text, for example, when entering a commit message. The default editor used is Vim. However, if you are a beginner and have never used Vim before, I **highly recommend** selecting another text editor you are familiar with, such as Notepad++, Visual Studio Code, or the good ol' Notepad.



[Open in app](#)[Get started](#)

## Choosing the default editor used by Git

Which editor would you like Git to use?



Use Vim (the ubiquitous text editor) as Git's default editor

The [Vim editor](#), while powerful, [can be hard to use](#). Its user interface is unintuitive and its key bindings are awkward.

**Note:** Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.

**Note:** This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.

<https://gitforwindows.org/>

Back

Next

Cancel

### The initial branch name

The entire software development community is trying to be more inclusive. Most Git repositories still use `master` as their primary branch. There is a movement to replace `master` with `main`, but Git still uses `master` as a default.

Major Git hosting companies (such as GitHub and GitLab) have already renamed their default branch name from `master` to `main`.

Currently, Git still used `master` as a default. However, I highly recommend overriding this setting, as shown in the image below.



[Open in app](#)[Get started](#)

## Adjusting the name of the initial branch in new repositories

What would you like Git to name the initial branch after "git init"?



**Let Git decide**

Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project [intends](#) to change this default to a more inclusive name in the near future.

**Override the default branch name for new repositories**

**NEW!** Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:

This setting does not affect existing repositories.

<https://gitforwindows.org/>

[Back](#)[Next](#)[Cancel](#)

## PATH environment

Whenever you wish to use a tool from a command line without specifying the full location path, you need to tell Windows about this by adding that directory to the PATH environment variable. Git is no different. I recommend using the default option.





[Open in app](#)[Get started](#)

## Adjusting your PATH environment

How would you like to use Git from the command line?



**Use Git from Git Bash only**

This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

**Git from the command line and also from 3rd-party software**

**(Recommended)** This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

**Use Git and optional Unix tools from the Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.  
**Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.**

<https://gitforwindows.org/>

Back

Next

Cancel

### SSH executable

Unless you know that you have OpenSSH already installed, it is best to use the OpenSSH that comes with Git.



[Open in app](#)[Get started](#)

### Choosing the SSH executable

Which Secure Shell client program would you like Git to use?



**Use bundled OpenSSH**

This uses `ssh.exe` that comes with Git.

**Use external OpenSSH**

**NEW!** This uses an external `ssh.exe`. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.

<https://gitforwindows.org/>

Back

Next

Cancel

### HTTPS transport backend

Unless you have specific instructions, use the OpenSSL library. In case you later face issues cloning private repositories from within your organization due to certificate validation issues, you may want to come back to this section.



[Open in app](#)[Get started](#)

### Choosing HTTPS transport backend

Which SSL/TLS library would you like Git to use for HTTPS connections?



**Use the OpenSSL library**

Server certificates will be validated using the `ca-bundle.crt` file.

**Use the native Windows Secure Channel library**

Server certificates will be validated using Windows Certificate Stores.  
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

<https://gitforwindows.org/>

Back

Next

Cancel

### Line ending conventions

Windows and Unix systems use a different encoding for line endings. Unless you have other instructions, it is best to go with the default here.



[Open in app](#)[Get started](#)

## Configuring the line ending conversions

How should Git treat line endings in text files?



**Checkout Windows-style, commit Unix-style line endings**

Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

**Checkout as-is, commit Unix-style line endings**

Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

**Checkout as-is, commit as-is**

Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

<https://gitforwindows.org/>

Back

Next

Cancel

## Terminal emulator for Git Bash

The MinTTY terminal emulator is much more modern and for most use-cases the best choice.



[Open in app](#)[Get started](#)

## Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?



**Use MinTTY (the default terminal of MSYS2)**

Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `wintpy` to work in MinTTY.

**Use Windows' default console window**

Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

<https://gitforwindows.org/>

Back

Next

Cancel

### Git pull behavior

By default, pulling new changes from a remote Git repository does not attempt a rebase. Since most Git manuals and tutorials assume you use the default option, it is best to leave it as it is (for now).



[Open in app](#)[Get started](#)

### Choose the default behavior of `git pull`

What should `git pull` do by default?



**Default (fast-forward or merge)**

This is the standard behavior of `git pull`: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

**Rebase**

Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.

**Only ever fast-forward**

Fast-forward to the fetched branch. Fail if that is not possible.

<https://gitforwindows.org/>

Back

Next

Cancel

### Git credential helper

Using a credentials helper is a great way to use multi-factor HTTPS authentication with Git. This is great for working with external Git repositories if you are working over HTTPS (and not SSH).

To quote the documentation, the [Git Credential Manager](#) “provides multi-factor authentication support for Azure DevOps, Azure DevOps Server (formerly Team Foundation Server), GitHub, Bitbucket, and GitLab.”





Open in app

Get started

### Choose a credential helper

Which credential helper should be configured?



**Git Credential Manager**

Use the [cross-platform Git Credential Manager](#).

See more information about the future of Git Credential Manager [here](#).

**None**

Do not use a credential helper.

<https://gitforwindows.org/>

Back

Next

Cancel

### Extra options

This is way to technical, so we will go with the defaults.





Open in app

Get started

### Configuring extra options

Which features would you like to enable?



**Enable file system caching**

File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

**Enable symbolic links**

Enable [symbolic links](#) (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.

<https://gitforwindows.org/>

Back

Next

Cancel

### Experimental options

At this point, I don't recommend enabling any of the experimental features.





[Open in app](#)[Get started](#)

### Configuring experimental options

These features are developed actively. Would you like to try them?



**Enable experimental support for pseudo consoles.**

**(NEW!)** This allows running native console programs like Node or Python in a Git Bash window without using winpty, but it still has known bugs.

**Enable experimental built-in file system monitor**

**(NEW!)** Automatically run a [built-in file system watcher](#), to speed up common operations such as ``git status``, ``git add``, ``git commit``, etc in worktrees containing many files.

<https://gitforwindows.org/>

Back

Install

Cancel

### Installation process

This should not take more than one minute to complete.



[Open in app](#)[Get started](#)

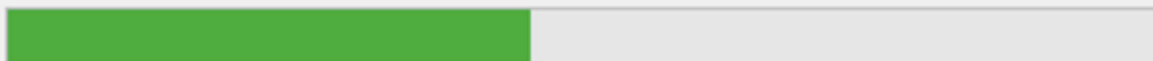
## Installing

Please wait while Setup installs Git on your computer.



Extracting files...

C:\Program Files\Git\usr\bin\msys-crypto-1.1.dll



<https://gitforwindows.org/>

Cancel

This wizard has also installed a new tool called **Git Bash**, a terminal alternative for Cmd or Powershell. I will use it to demonstrate the upcoming steps.



[Open in app](#)[Get started](#)

## Completing the Git Setup Wizard

Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.



Click Finish to exit Setup.

- Launch Git Bash
- View Release Notes

Only show new options

Finish

Click on the checkbox to launch Git Bash and click on Finish.

From Git Bash or your terminal of choice, run the following command.

```
git --version
```

```
MINGW64:/c/Users/valentin  
valentin@win10v2 MINGW64 ~  
$ git --version
```



[Open in app](#)[Get started](#)

## Step 2 — Configure Git

Before we move forward, adapt the following commands with your name and email. They will be part of any changes you make to any Git repository. Do it now, otherwise, your work colleagues will give you a minus point.

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

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

## Conclusion

I hope this tutorial helped you get started with configuring your Git installation in Windows 10 to work with GitLab CI. Leave a comment in the section below if you have any questions. I would love to hear from you!

Thank you for sticking with this article until the end. If you enjoyed it, please leave a comment, share, and press that 🙌 a few times (up to 50 times). It will help others discover this information and maybe it will help someone else as well.

Follow me on [Medium](#) and [YouTube](#) if you're interested in more tutorials like this one.

---

**Sign up for DevOps with Valentine**

