Source Control with git

What is source control?

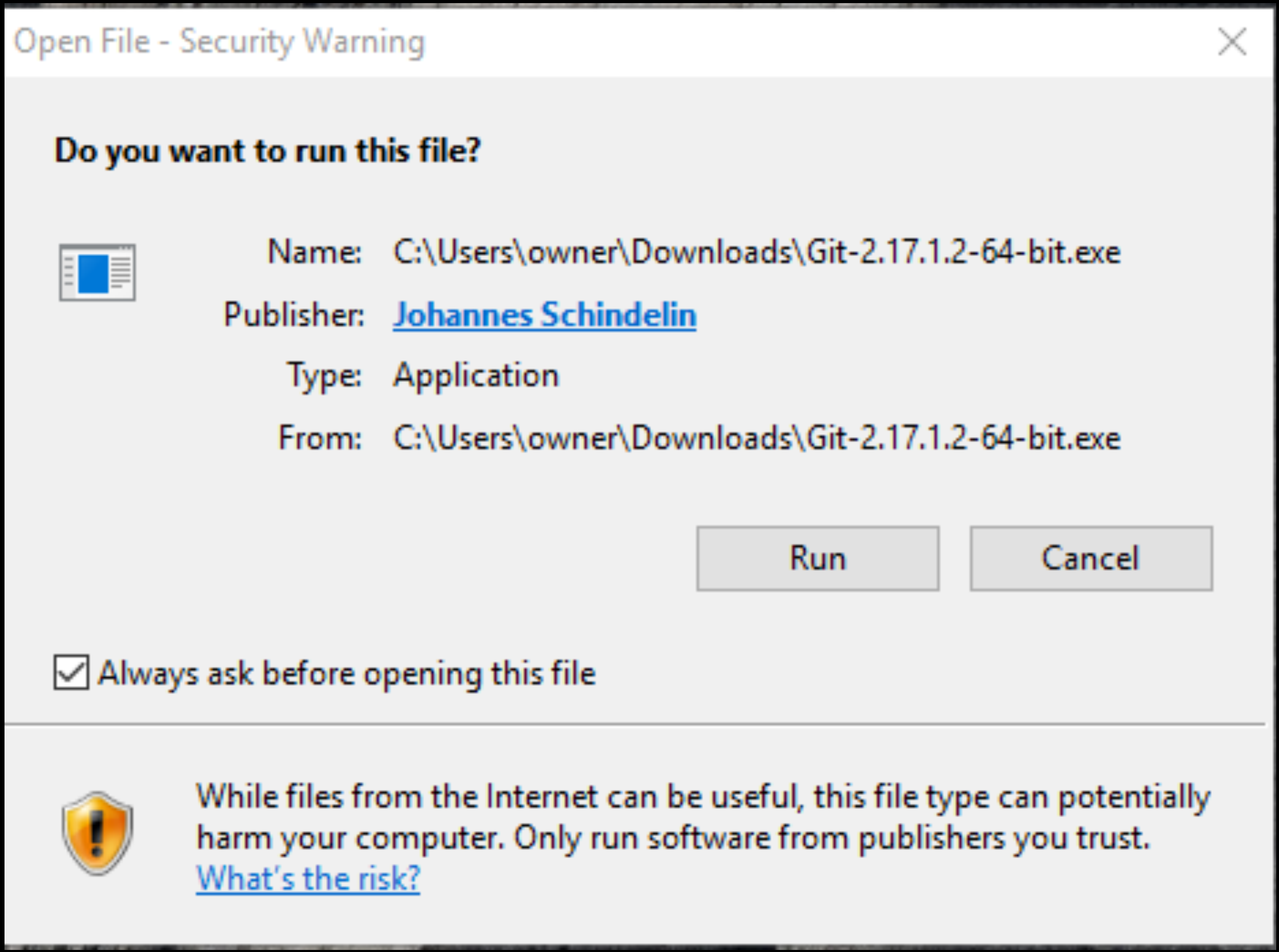
- a way to keep track of changes to source code
- what is source code?
- a way to keep it safe and backed up in multiple places
- a way to provide for multiple versions or feature sets
- a way for automation to occur on your source code (builds, deployments, etc)
- most importantly, a way to collaborate and share with other developers

Why git?

- distributed
- cross platform
- very fast and efficient
- easy branching and merging
- great tooling
- github
- it’s awesome (and everybody uses it now)

let’s get it installed!

- if you have a empower laptop, you are done!
- if you already have git installed, you are done!
- if you are on windows, use the local installer
- mac users, hang tight, we will do an install at the end (it is only 2 steps!)



Open File - Security Warning

Do you want to run this file?

Name: C:\Users\owner\Downloads\Git-2.17.1.2-64-bit.exe

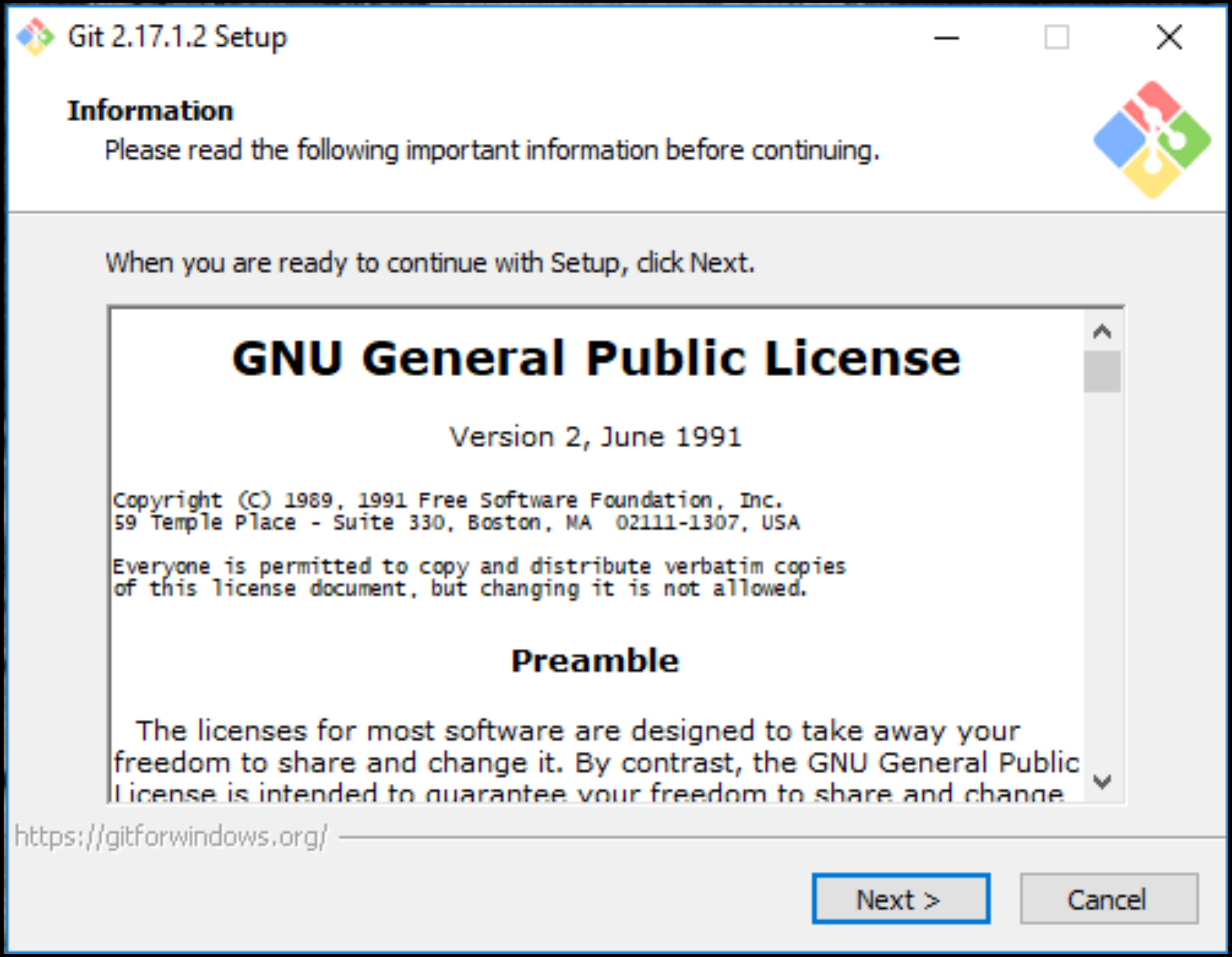
Publisher: Johannes Schindelin

Type: Application

From: C:\Users\owner\Downloads\Git-2.17.1.2-64-bit.exe

☑ Always ask before opening this file

While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?



Information

Please read the following important information before continuing.

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

GNU General Public License

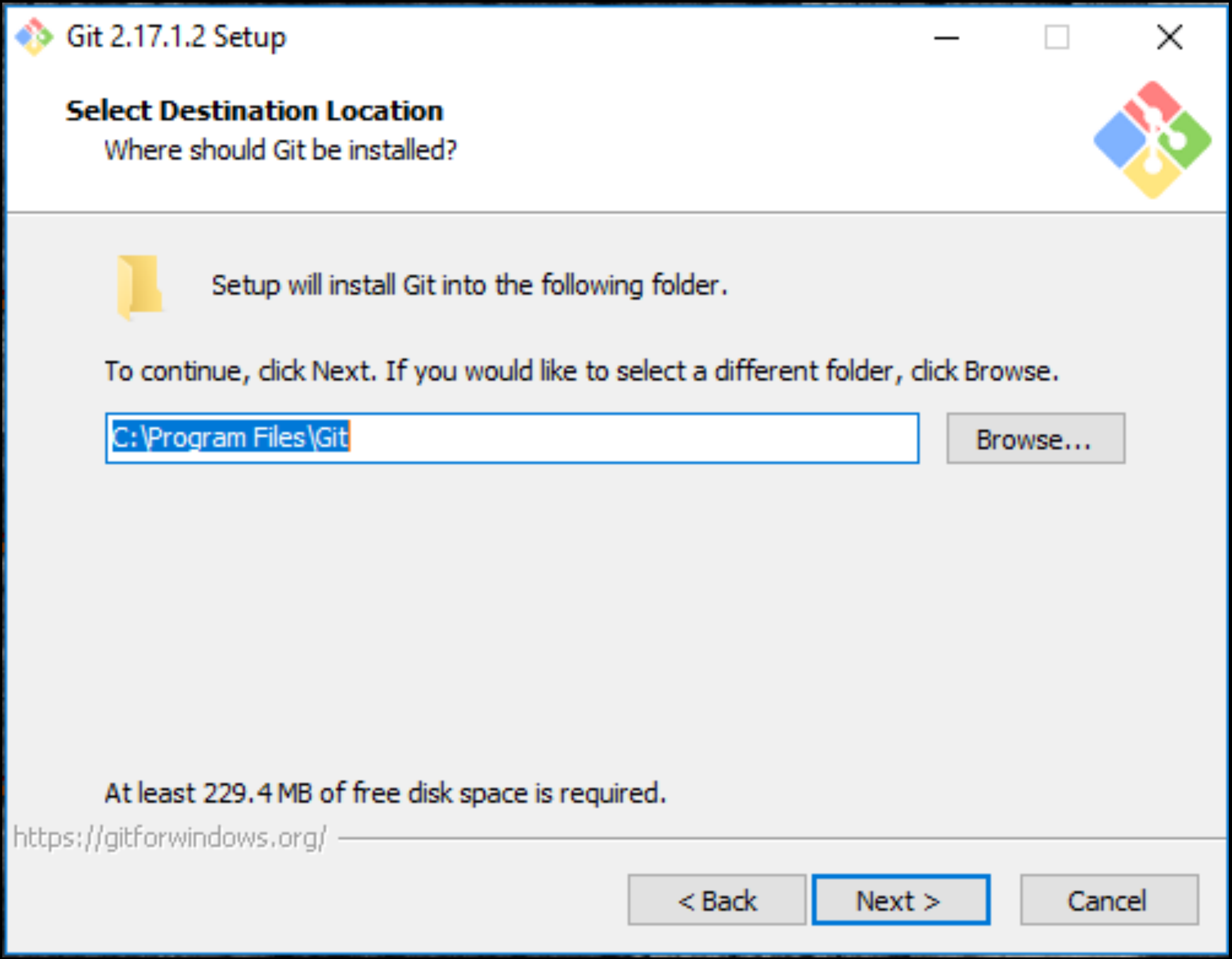
Version 2, June 1991

```
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



Select Destination Location

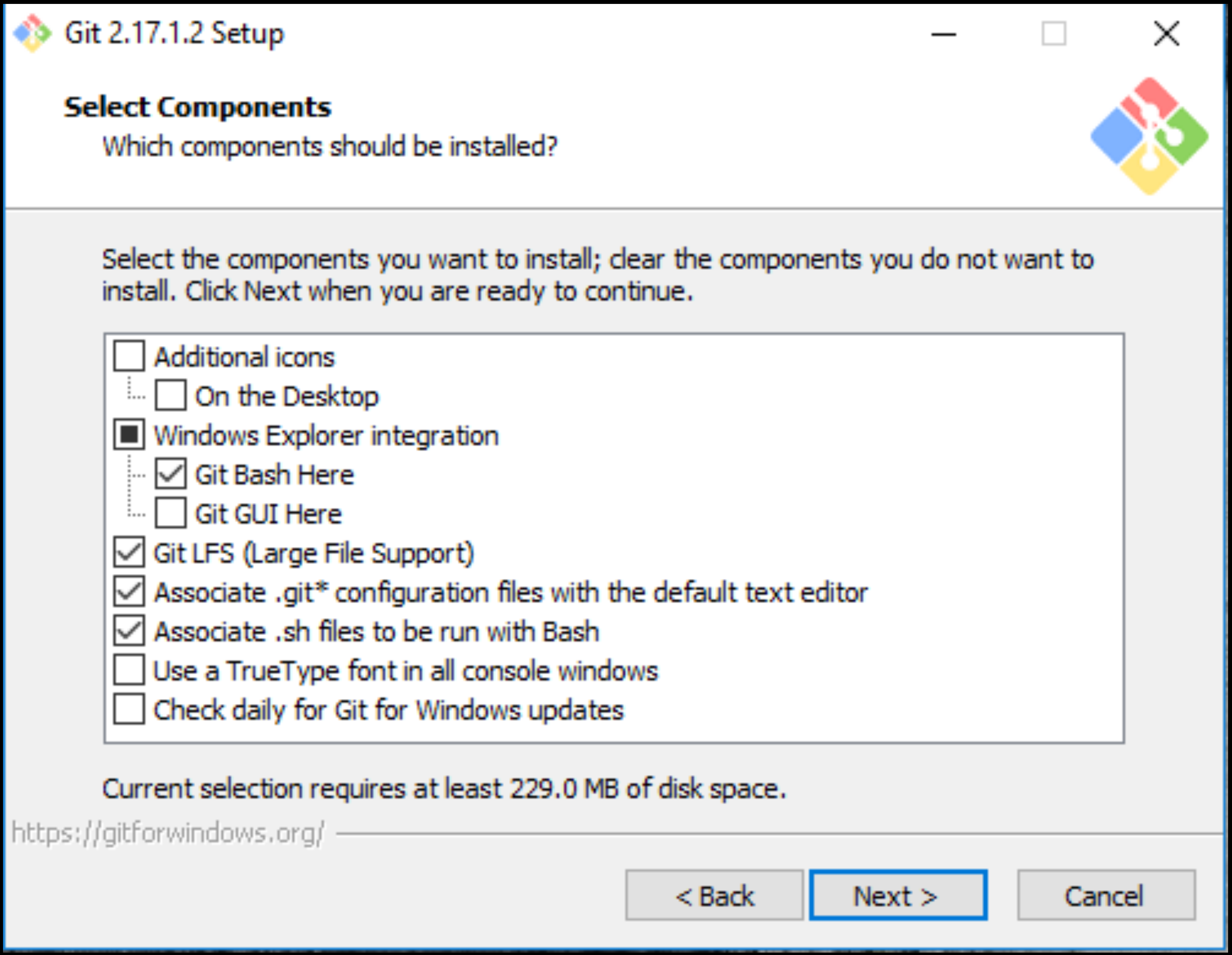
Where should Git be installed?

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

At least 229.4 MB of free disk space is required.



Git 2.17.1.2 Setup

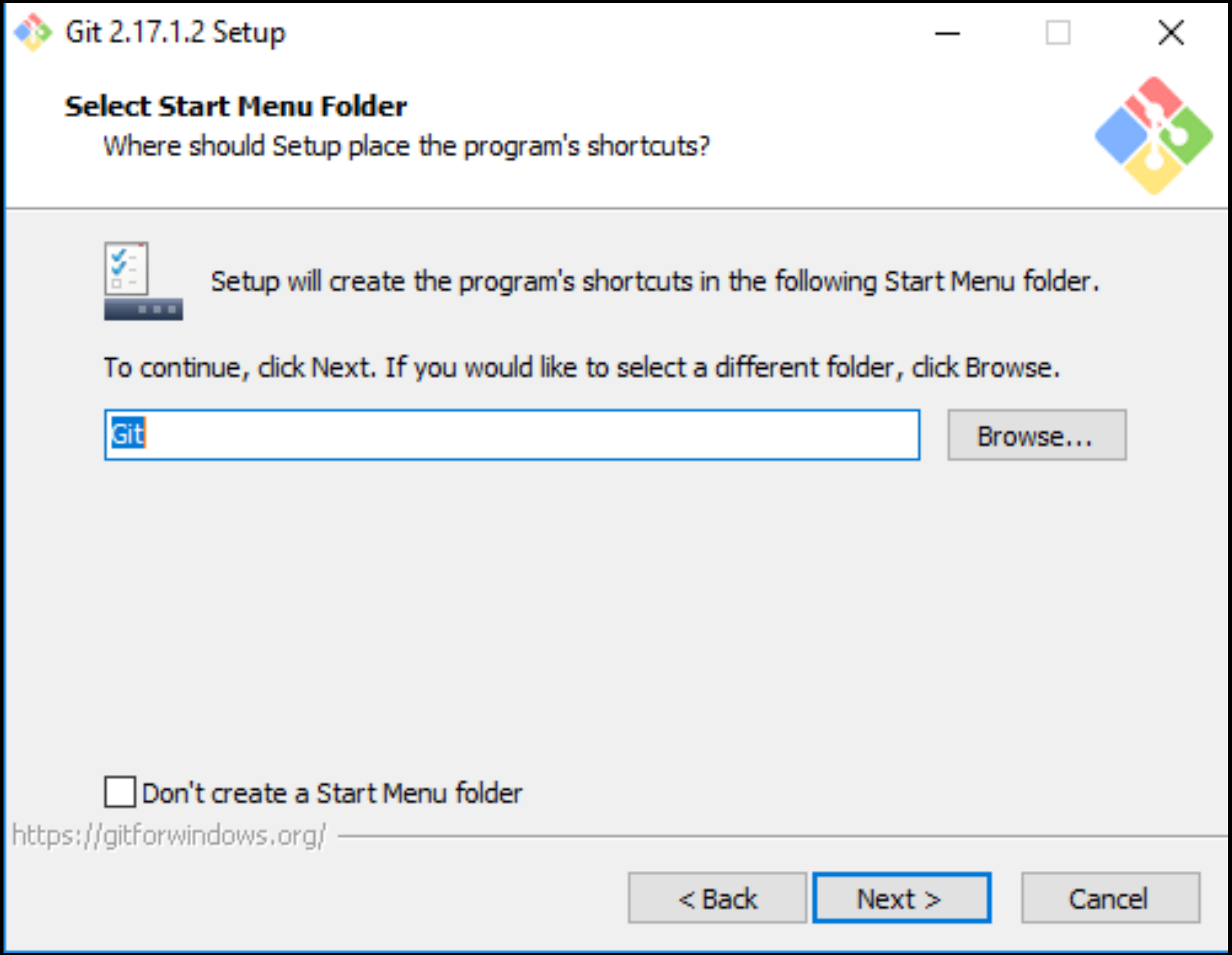
Select Components

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
- □ Use a TrueType font in all console windows
- □ Check daily for Git for Windows updates

Current selection requires at least 229.0 MB of disk space.



Git 2.17.1.2 Setup

Select Start Menu Folder

Where should Setup place the program's shortcuts?

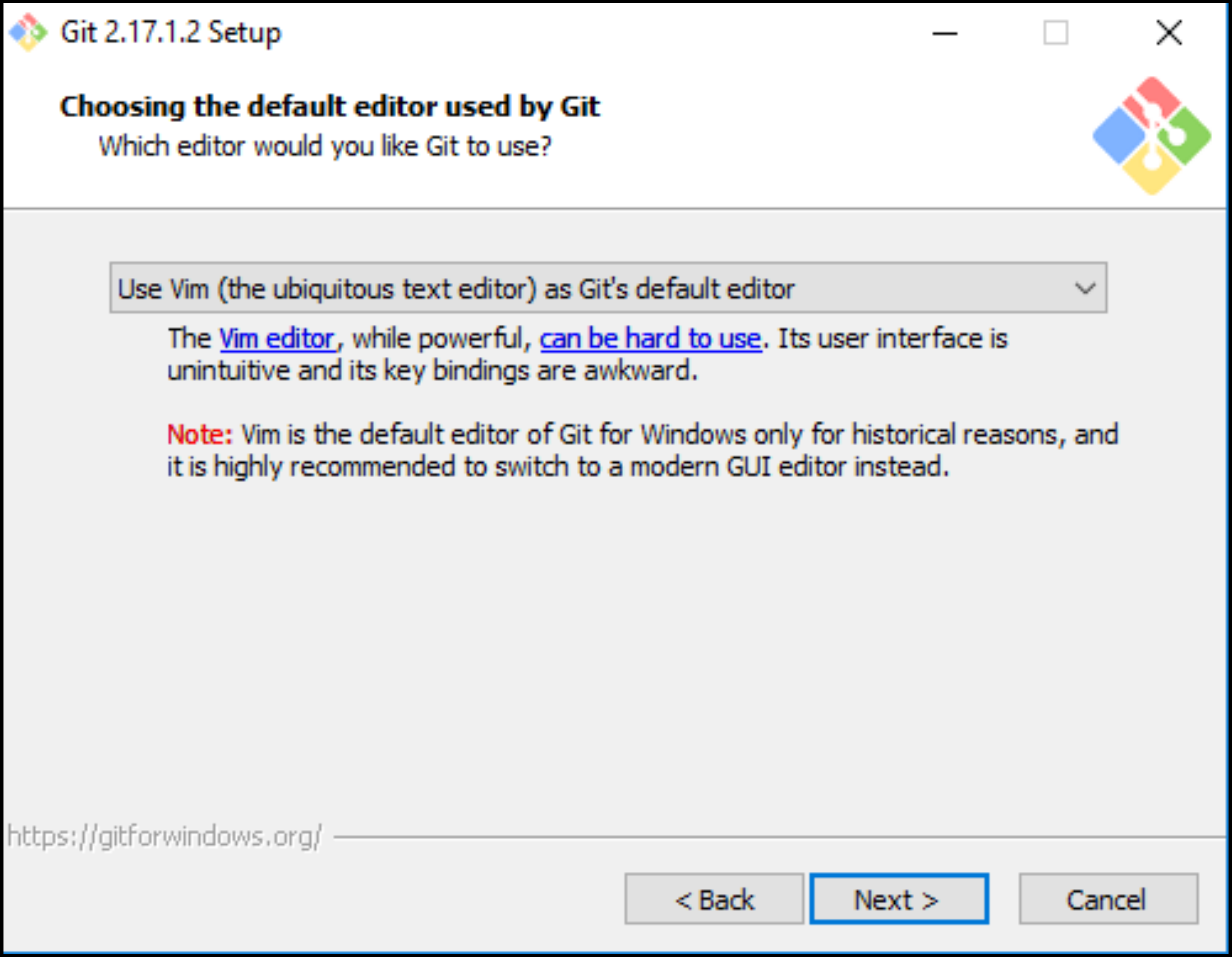
Setup will create the program's shortcuts in the following Start Menu folder.

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

Git

□ Don't create a Start Menu folder

https://gitforwindows.org/



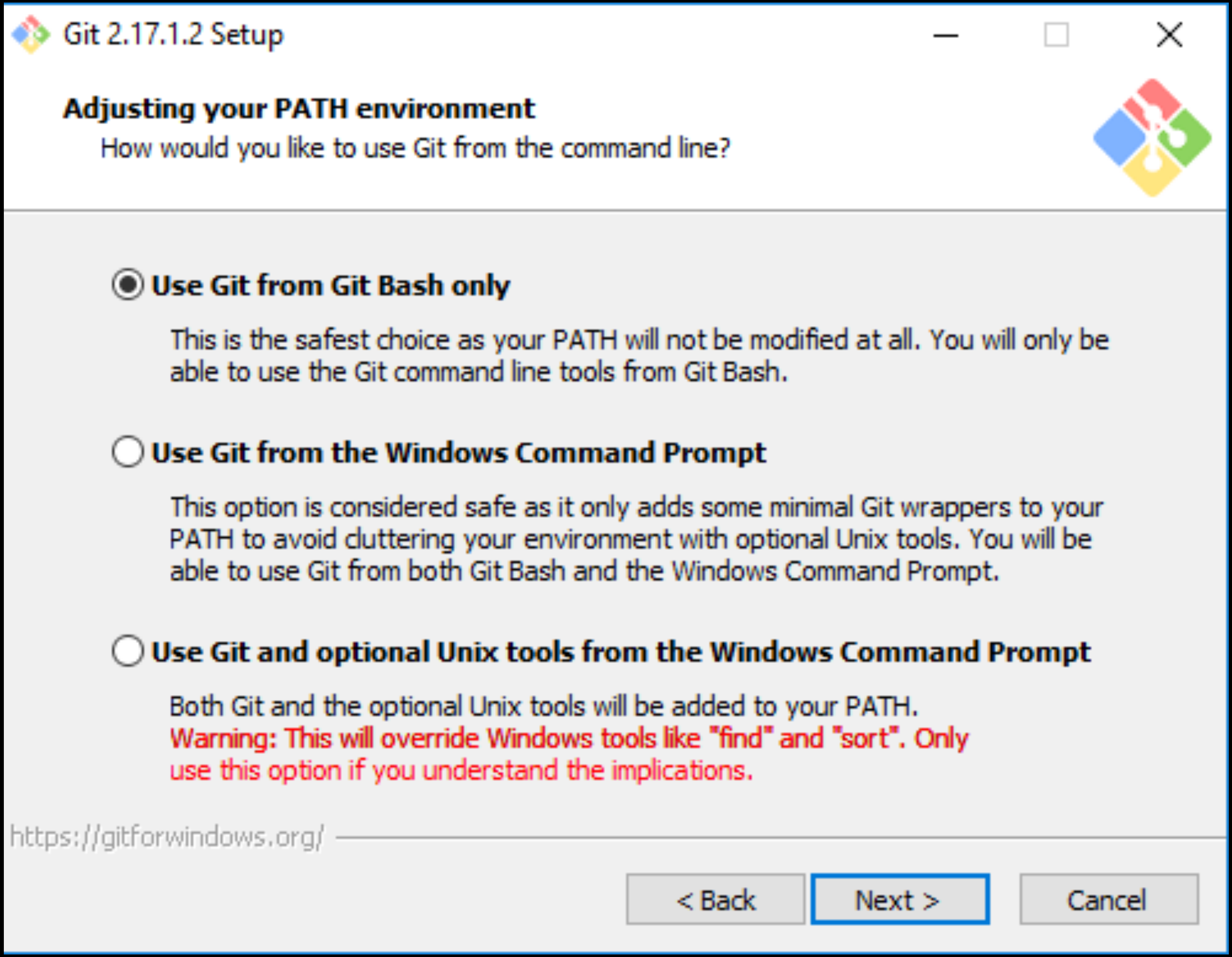
Choosing the default editor used by Git

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.



Adjusting your PATH environment

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

● **Use Git from Git Bash only**

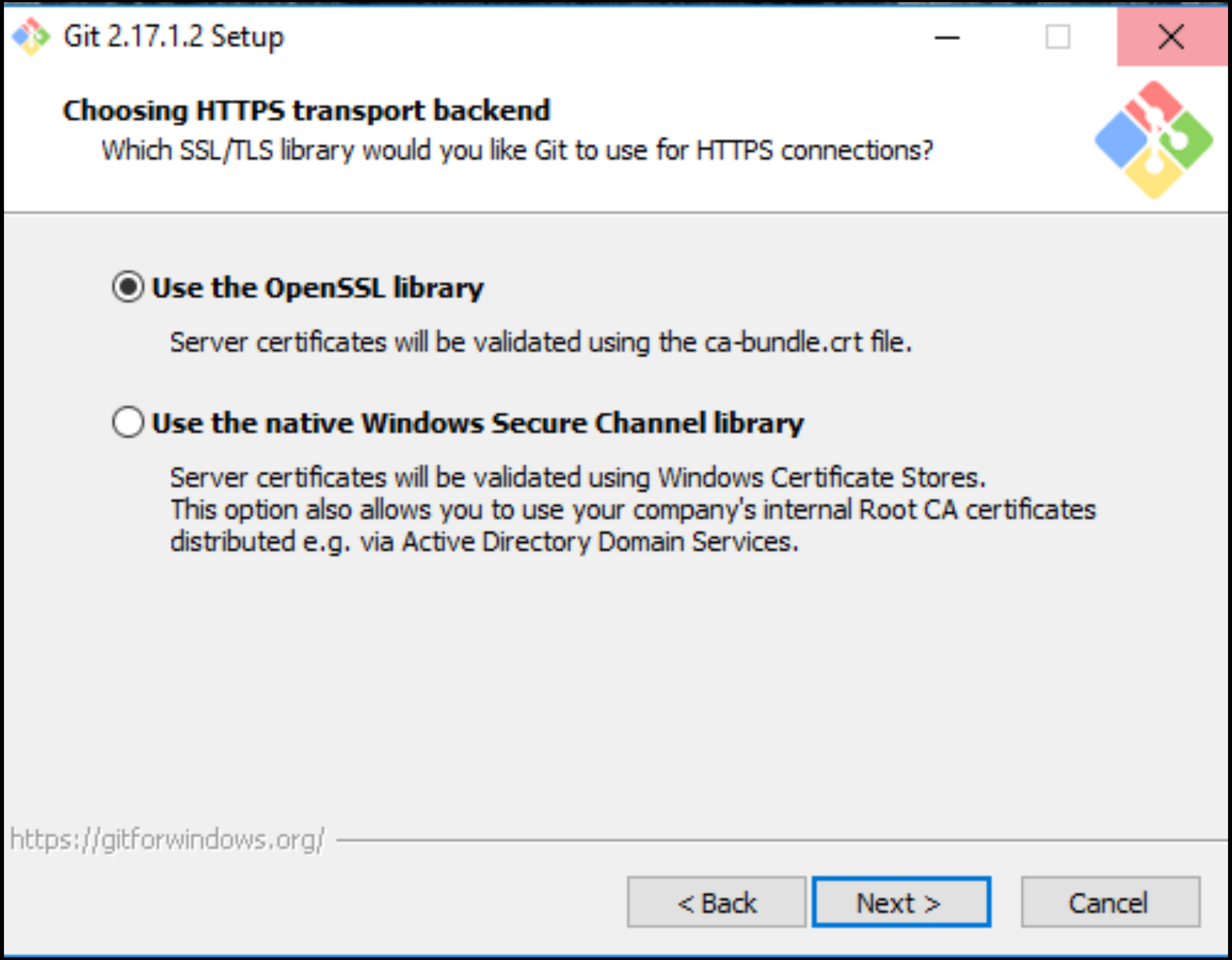
This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

○ **Use Git from the Windows Command Prompt**

This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from both Git Bash and the Windows Command Prompt.

○ **Use Git and optional Unix tools from the Windows 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.



Choosing HTTPS transport backend

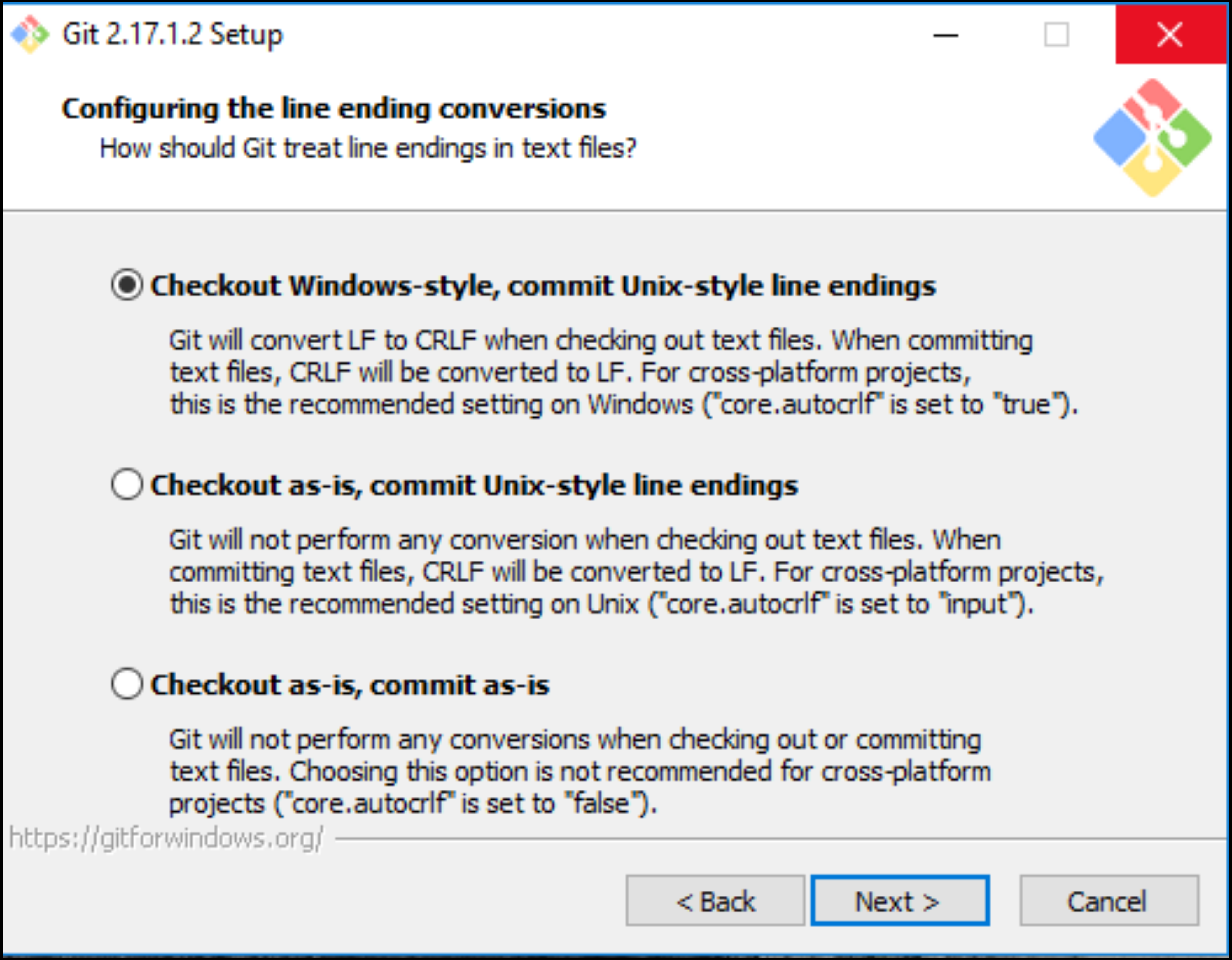
Which SSL/TLS library would you like Git to use for HTTPS connections?

● **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.



Configuring the line ending conversions

How should Git treat line endings in text files?

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

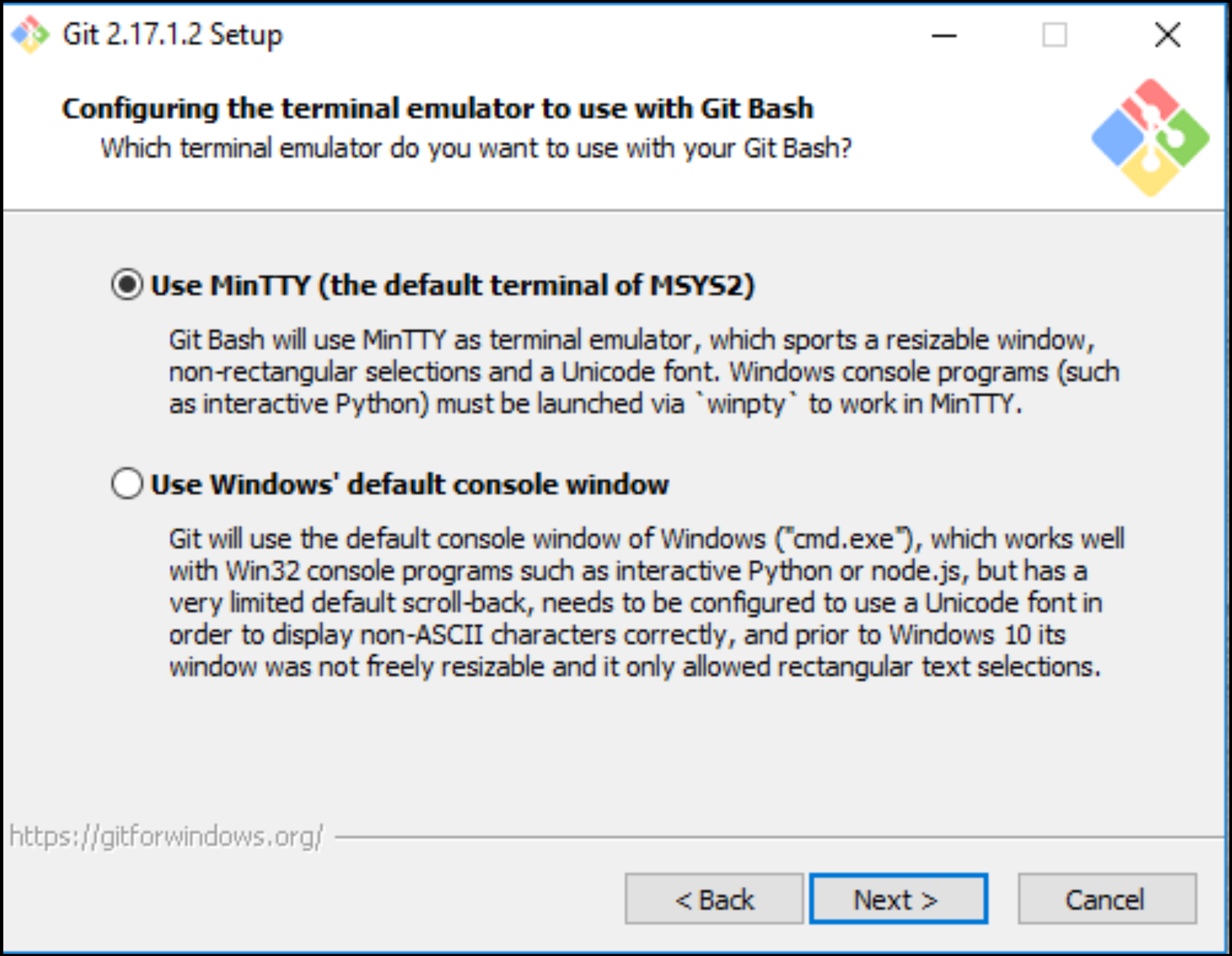
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

○ **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").



Configuring the terminal emulator to use with Git Bash

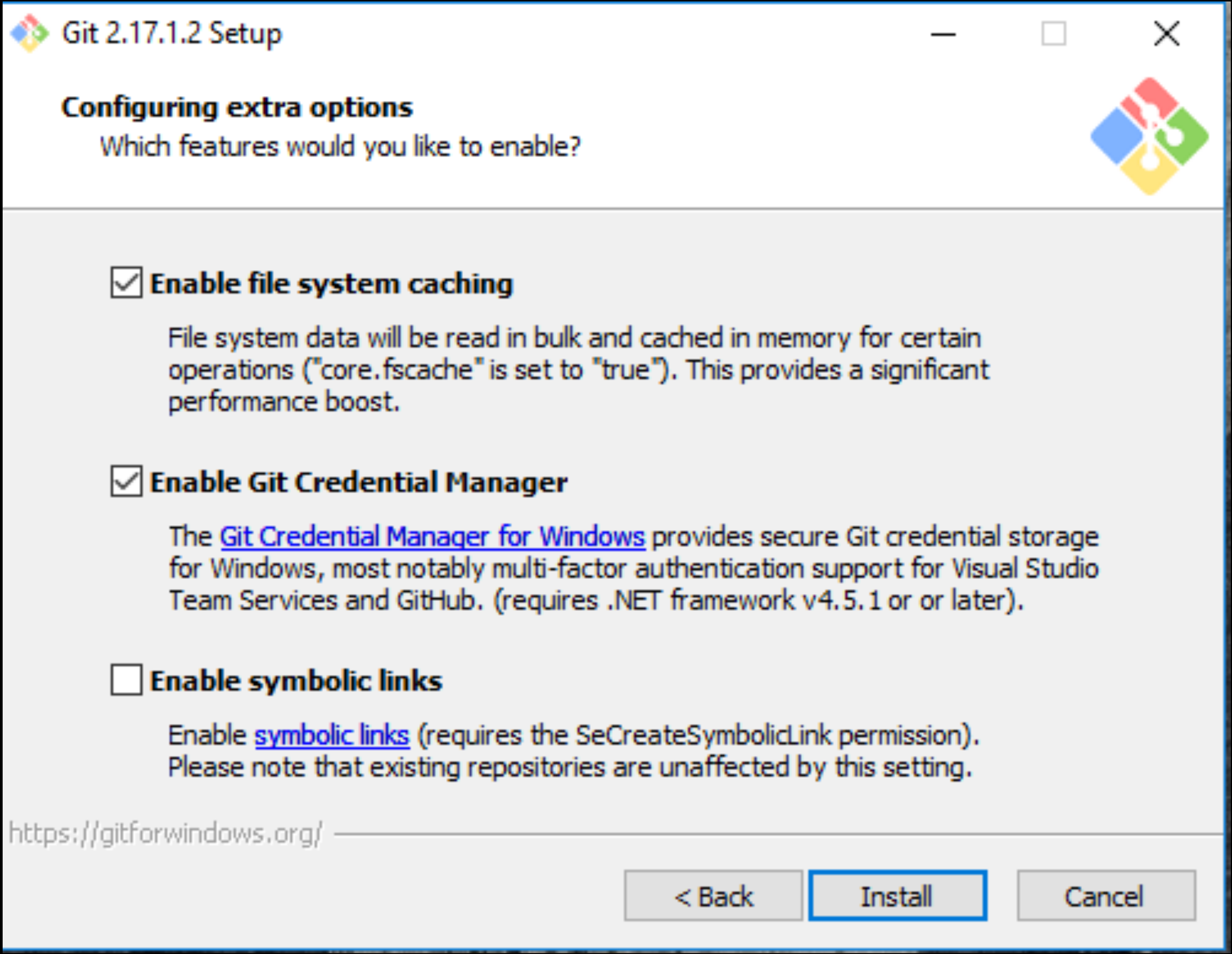
Which terminal emulator do you want to use with your Git Bash?

● **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 `winpty` 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.



Configuring extra options

Which features would you like to enable?

☑ **Enable file system caching**

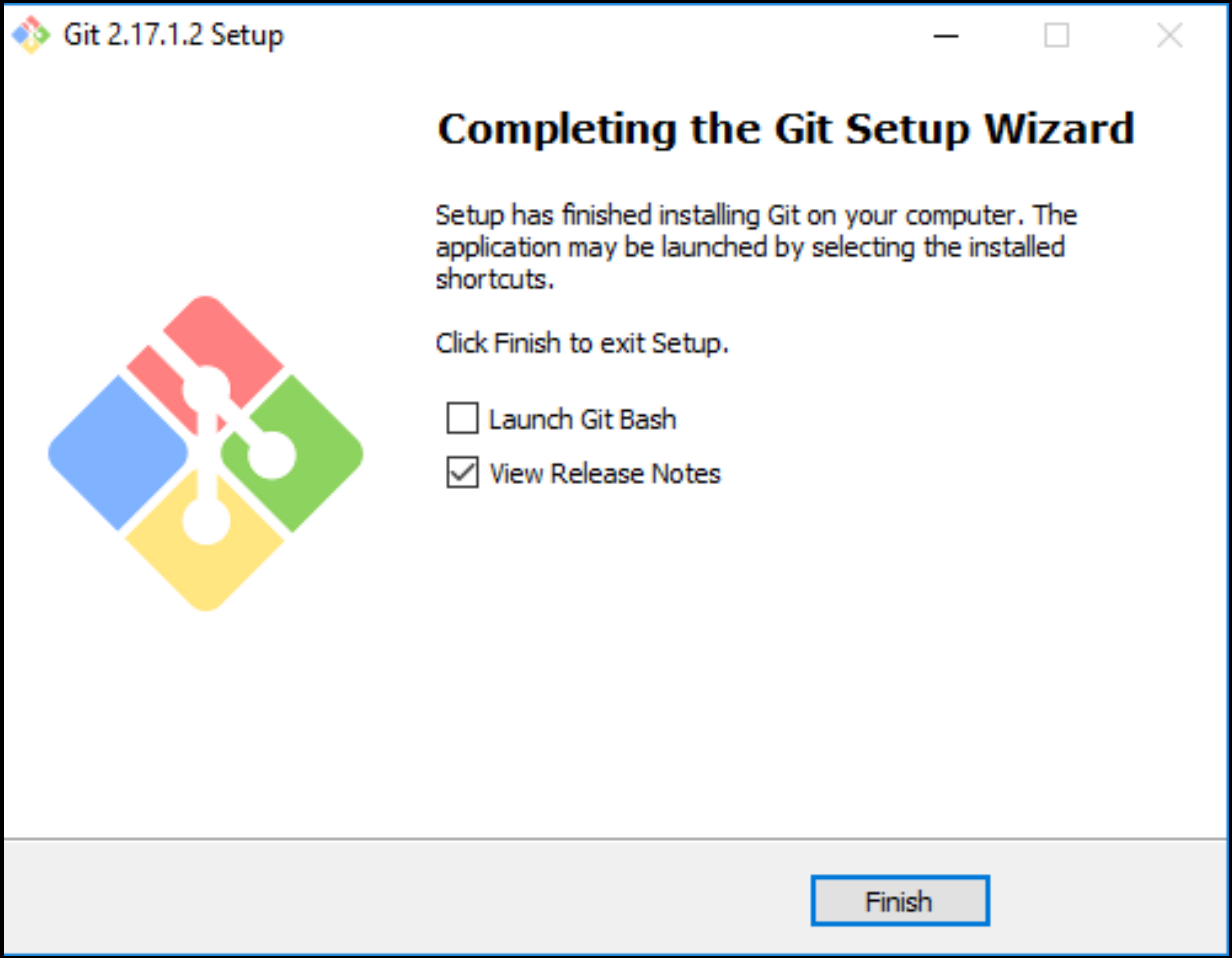
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

☑ **Enable Git Credential Manager**

The Git Credential Manager for Windows provides secure Git credential storage for Windows, most notably multi-factor authentication support for Visual Studio Team Services and GitHub. (requires .NET framework v4.5.1 or or later).

□ **Enable symbolic links**

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



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

install on a mac

- in your browser go to https://brew.sh/
- open a terminal session
- copy this line that you see in the browser: /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
- paste that line in the terminal and hit enter to install homebrew
- once homebrew is installed, type this in your terminal: brew install git

let’s configure it!

ssh key and configuration

- open this link in your browser - https://bit.ly/1UiZIkj
- follow the “Generating a new SSH key” directions
- take the contents of the id_rsa.pub file and put it in your github profile under settings / ssh and gpg keys
- in git bash, configure your name and email like so:
- git config --global user.name “FirstName LastName”
- git config --global user.email email@address.here

Victory!

Now let’s see what it can do