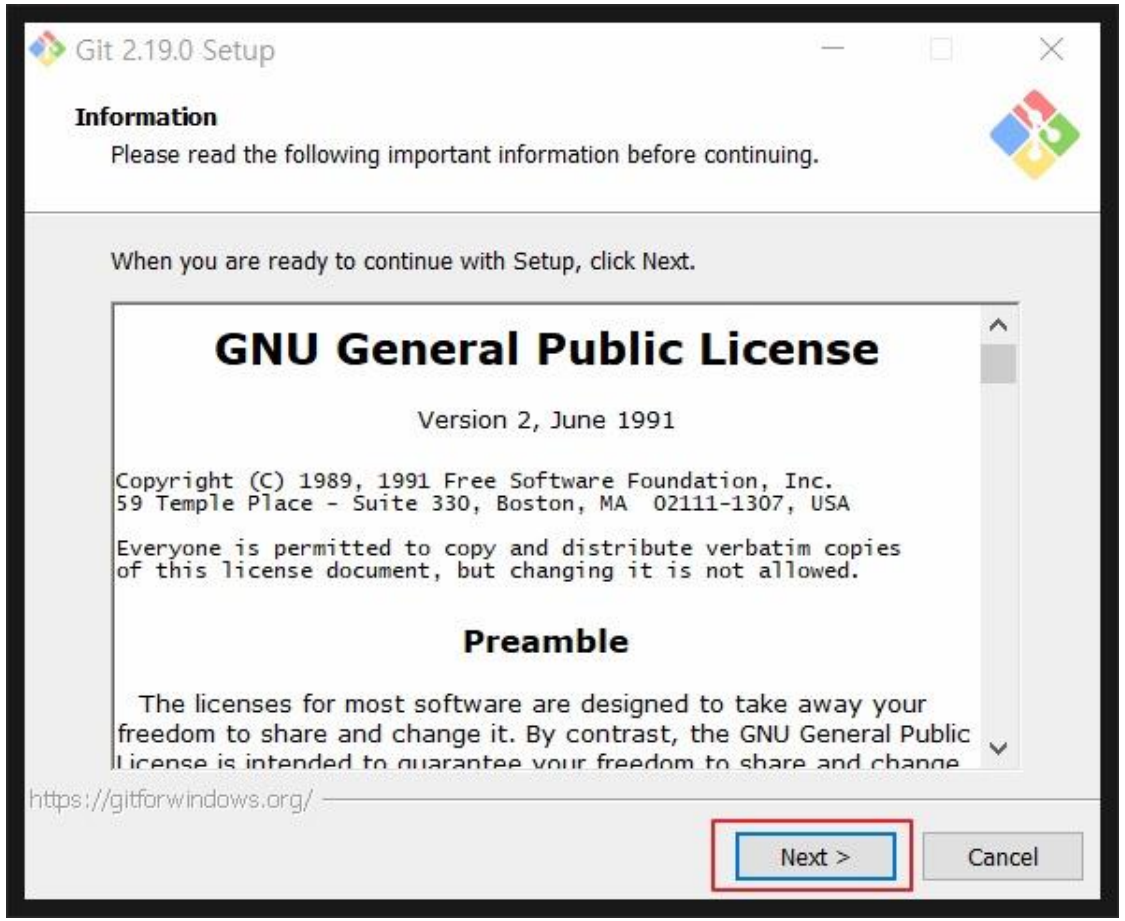
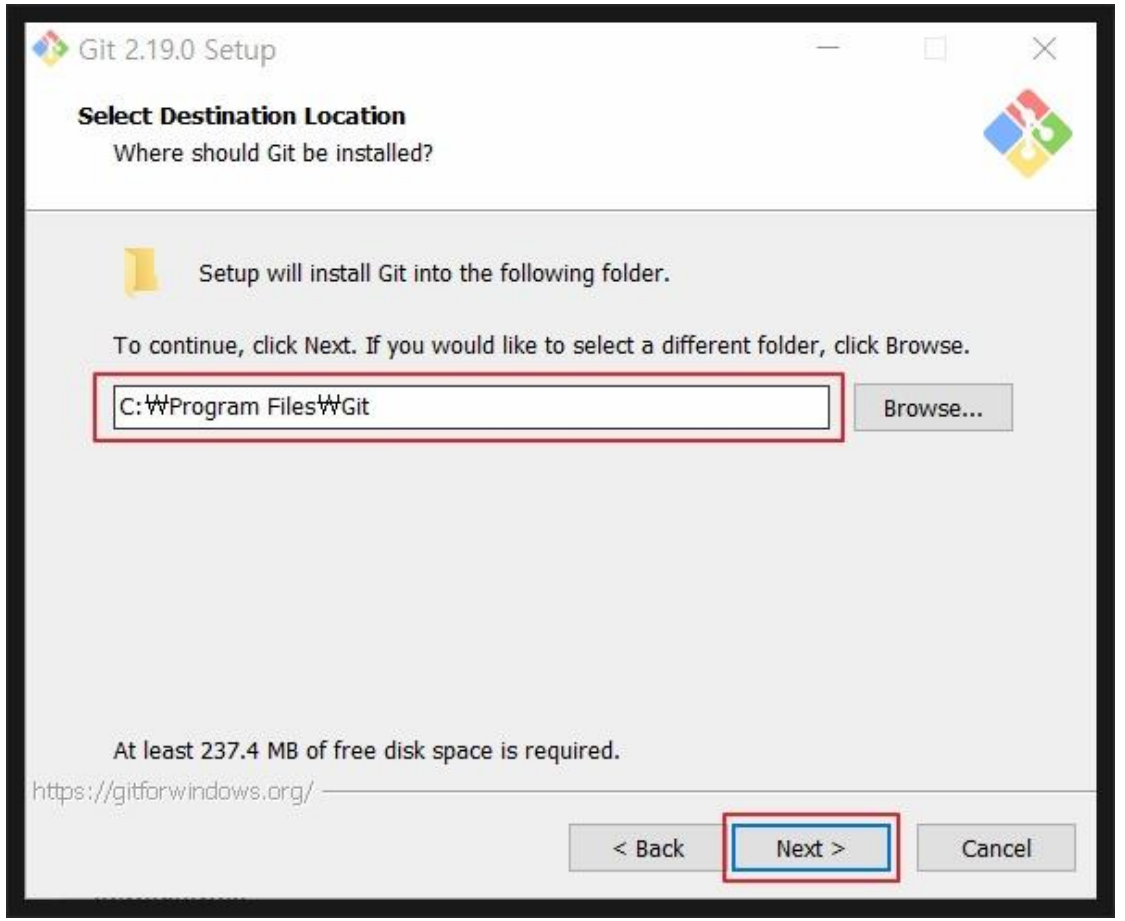
Git 기본 사용법

Git bash 설치

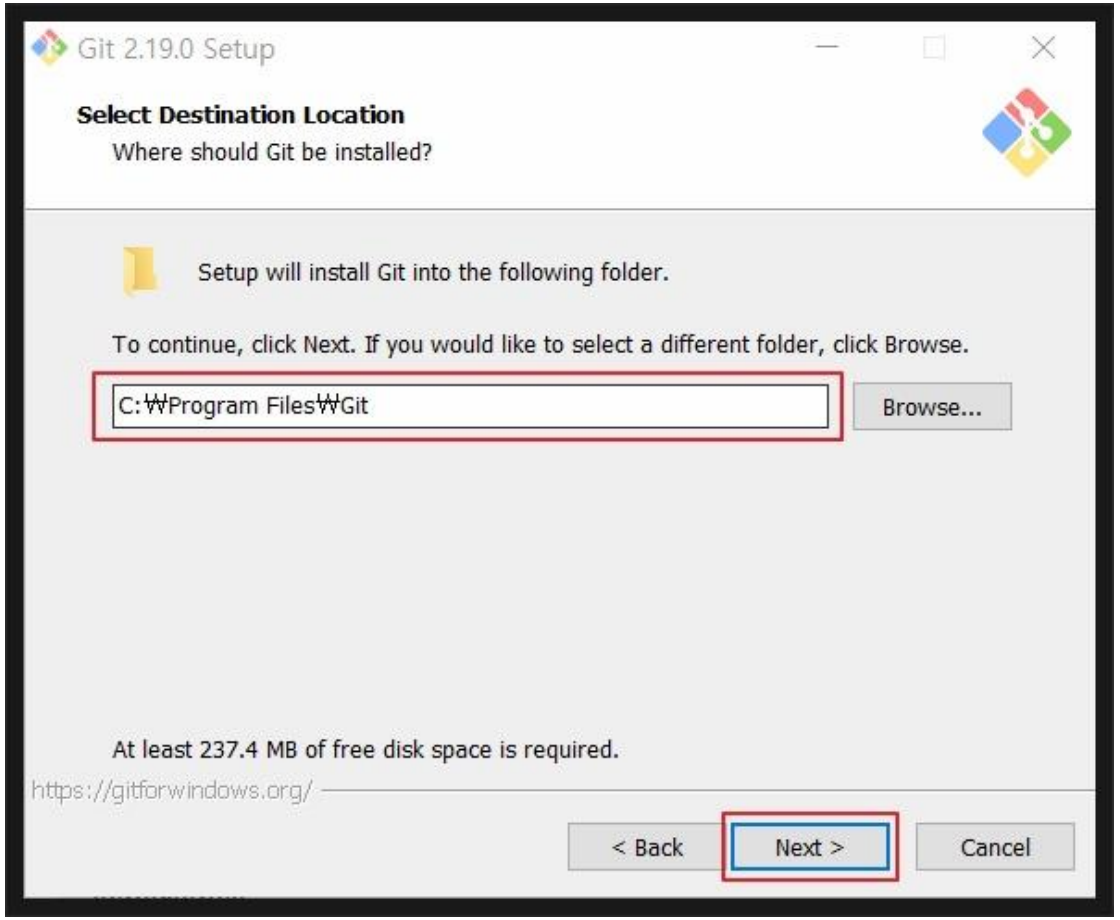
- https://git-scm.com/downloads 접속 후 window 버전 클릭

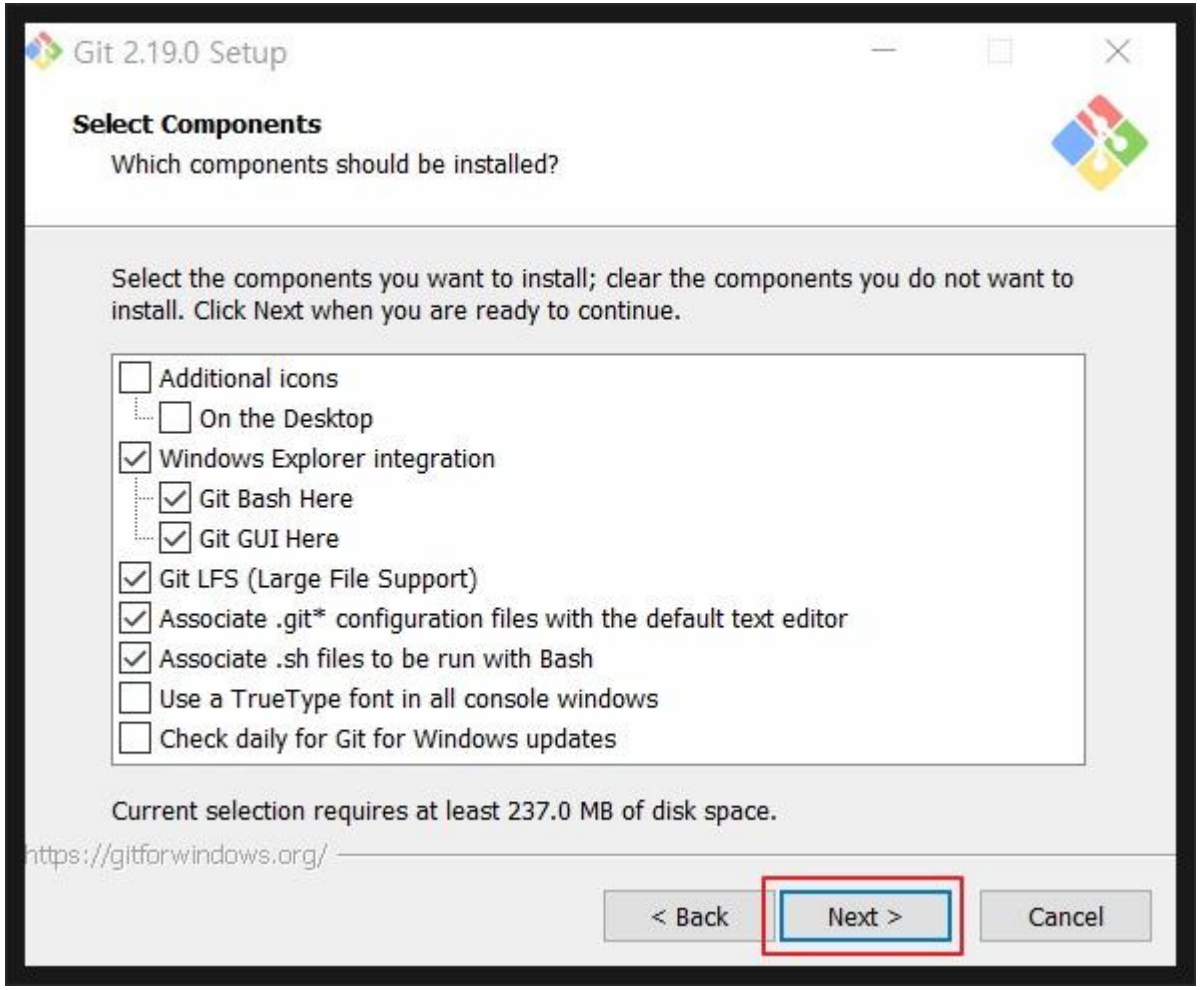
Git bash 설치


Git 2.19.0 Setup
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
https://gitforwindows.org/
Next >
Cancel

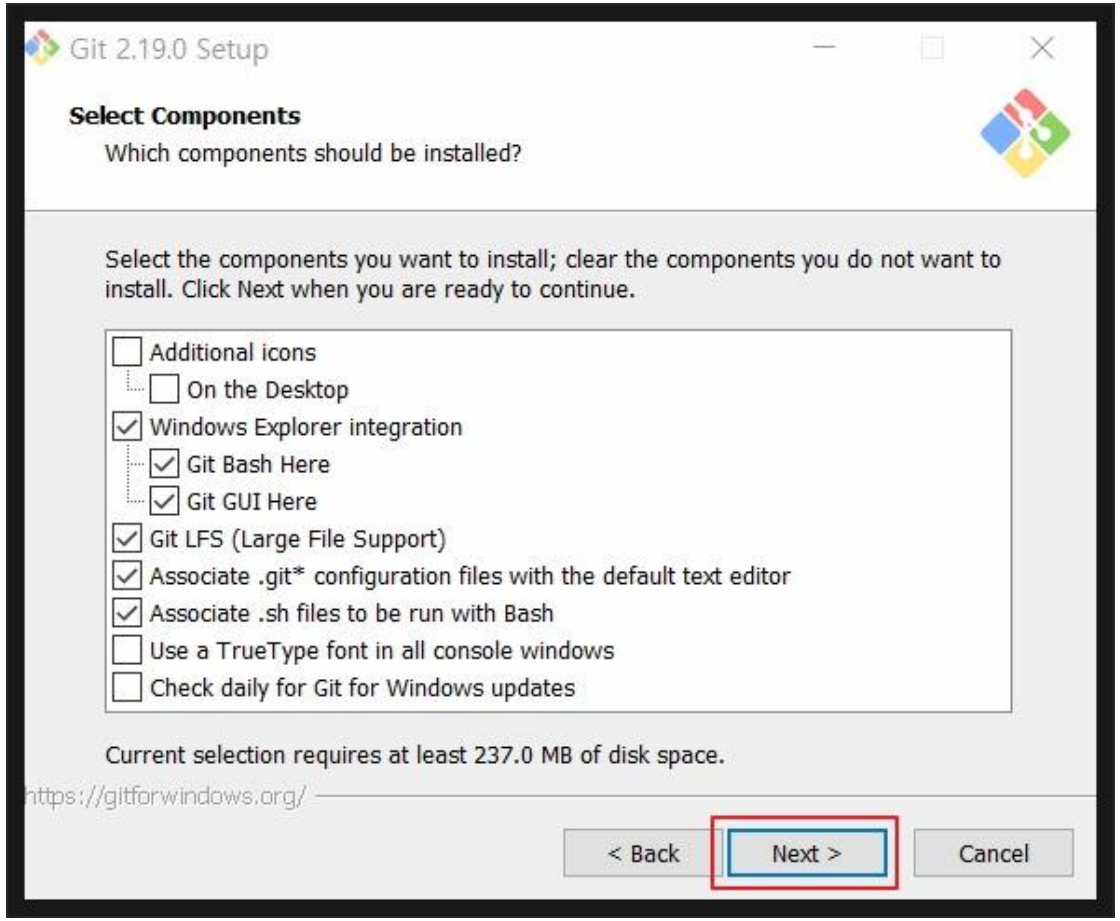

Git 2.19.0 Setup
Select Destination Location
Where should Git be installed?
Setup will install Git into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:₩Program Files₩Git
Browse...
At least 237.4 MB of free disk space is required.
https://gitforwindows.org/
< Back
Next >
Cancel

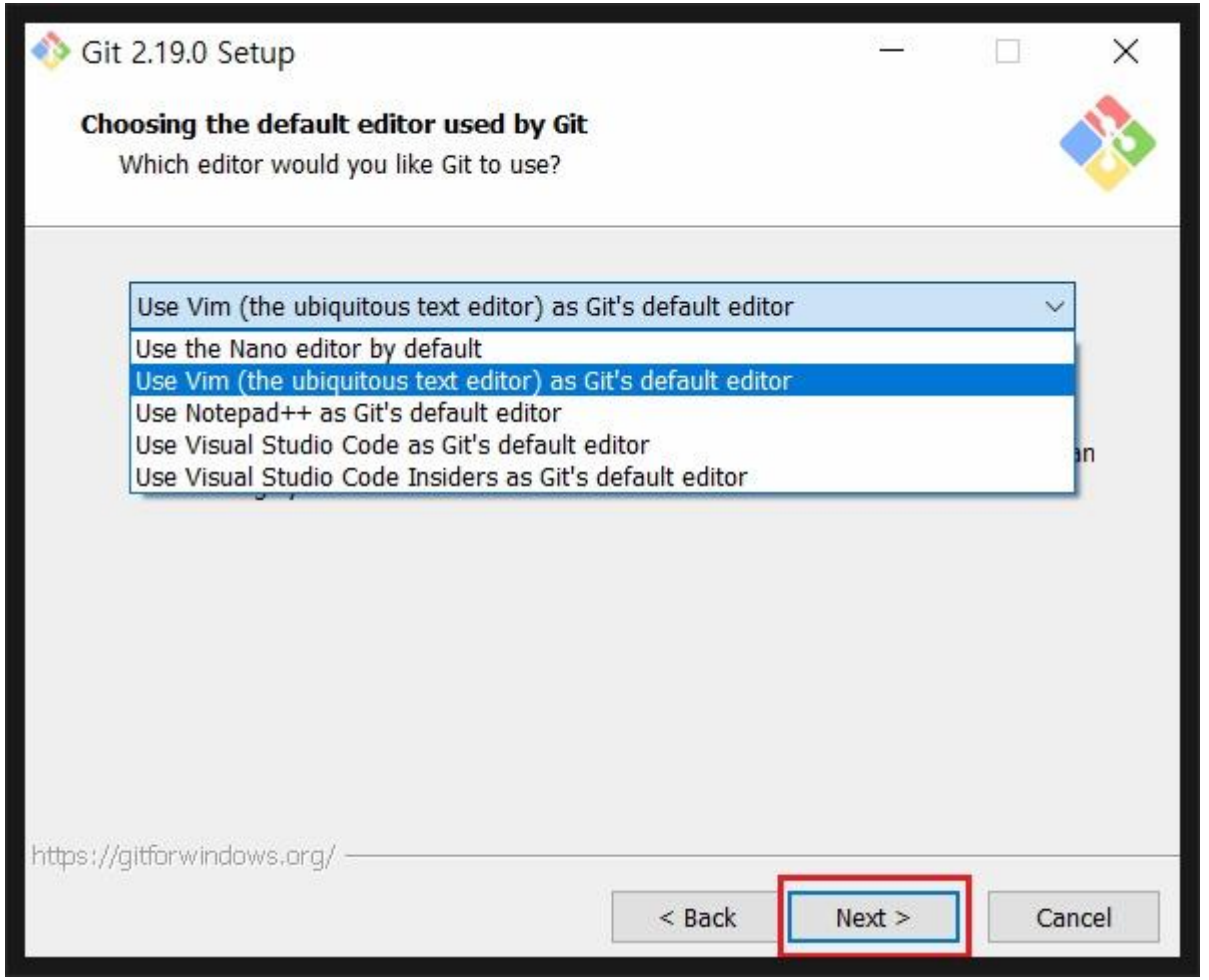
Git bash 설치


Git 2.19.0 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 237.4 MB of free disk space is required.
https://gitforwindows.org/
< Back
Next >
Cancel

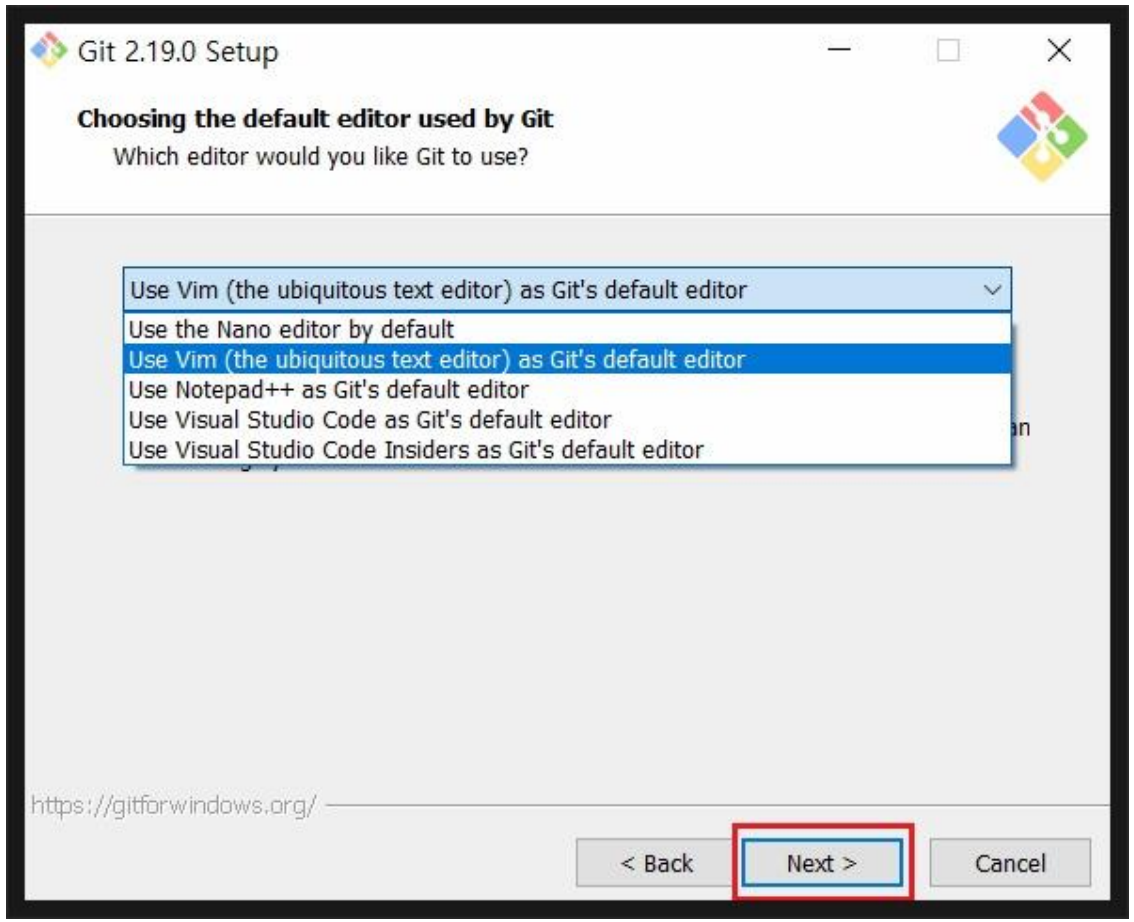

Git 2.19.0 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
Windows Explorer integration
Git Bash Here
Git GUI Here
Git LFS (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 237.0 MB of disk space.
https://gitforwindows.org/
< Back
Next >
Cancel

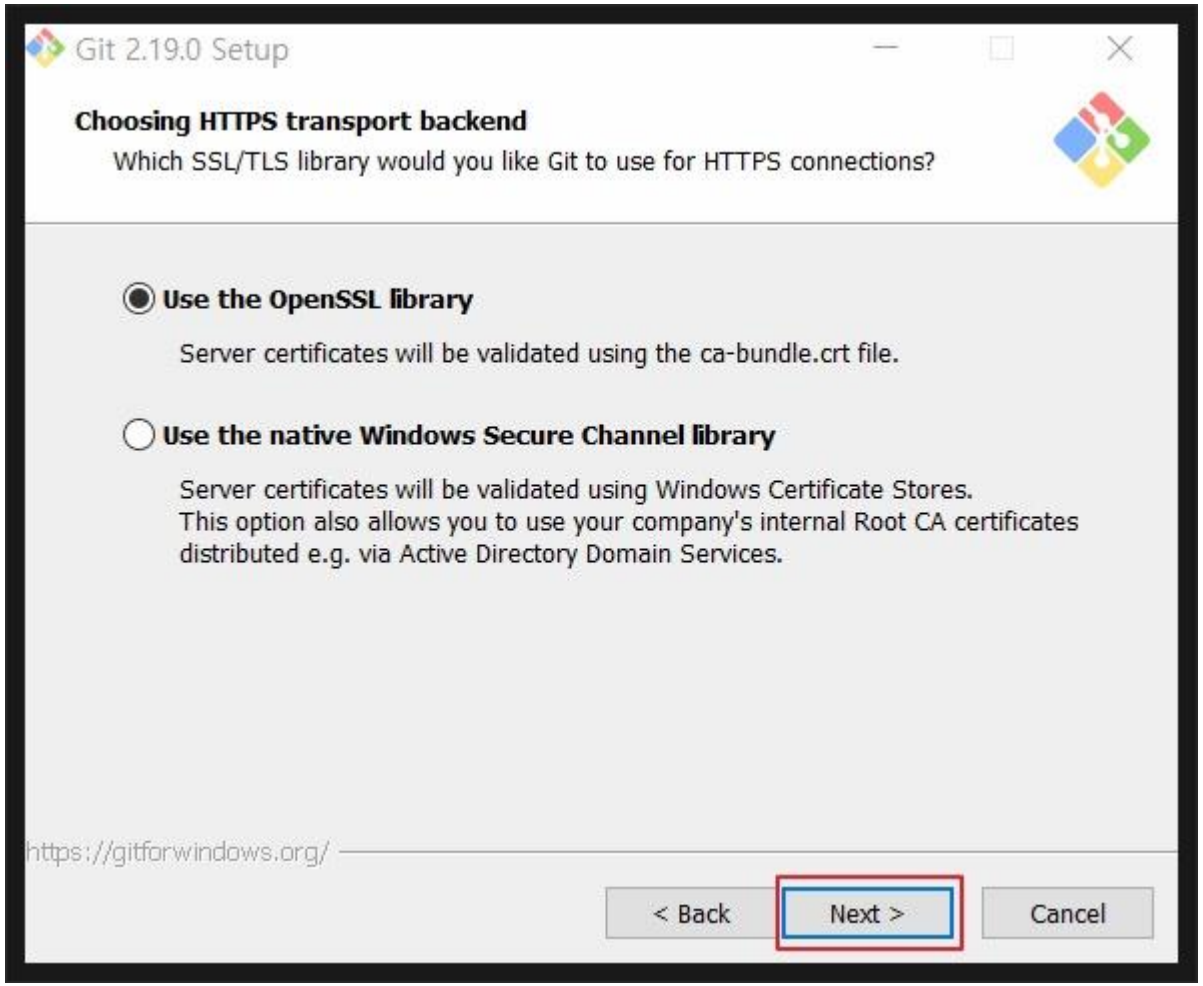
Git bash 설치


Git 2.19.0 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 237.0 MB of disk space.
https://gitforwindows.org/
< Back
Next >
Cancel

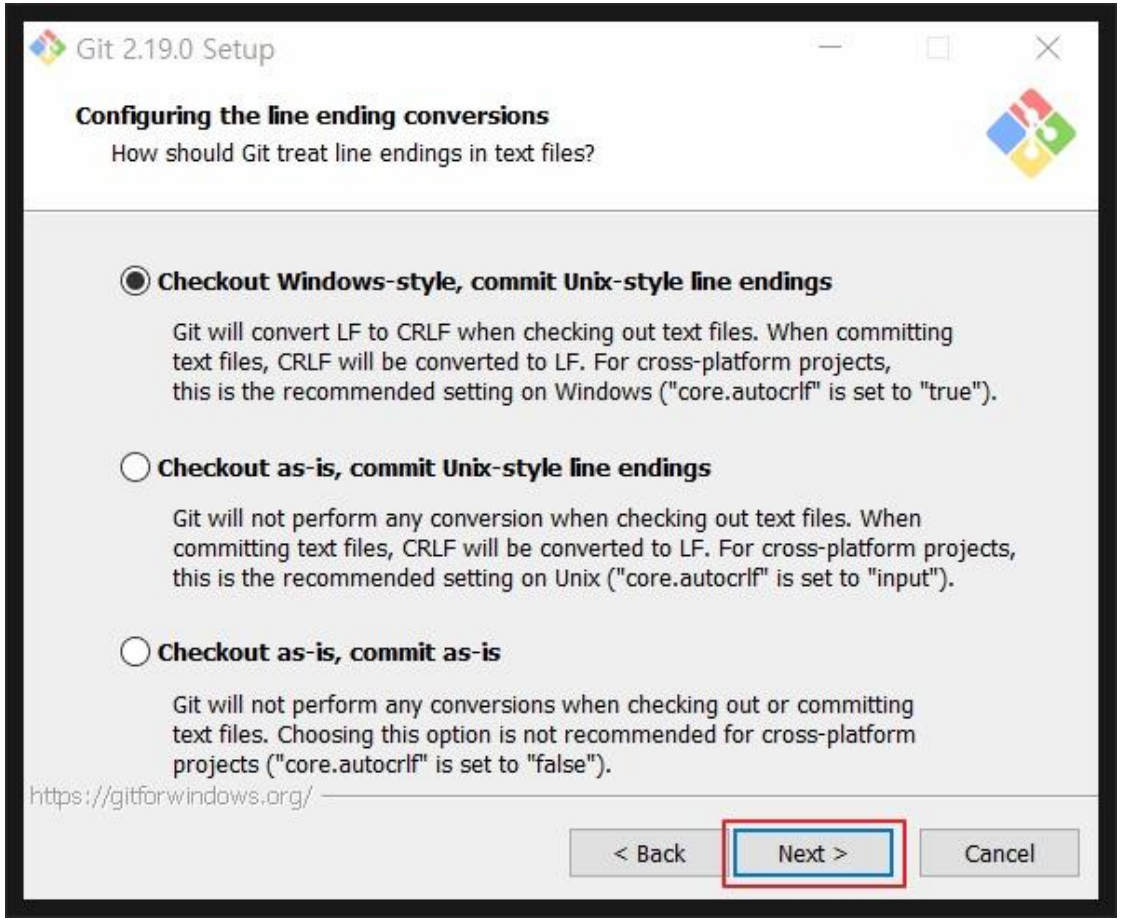

Git 2.19.0 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use Vim (the ubiquitous text editor) as Git's default editor
Use the Nano editor by default
Use Vim (the ubiquitous text editor) as Git's default editor
Use Notepad++ as Git's default editor
Use Visual Studio Code as Git's default editor
Use Visual Studio Code Insiders as Git's default editor
https://gitforwindows.org/
< Back
Next >
Cancel

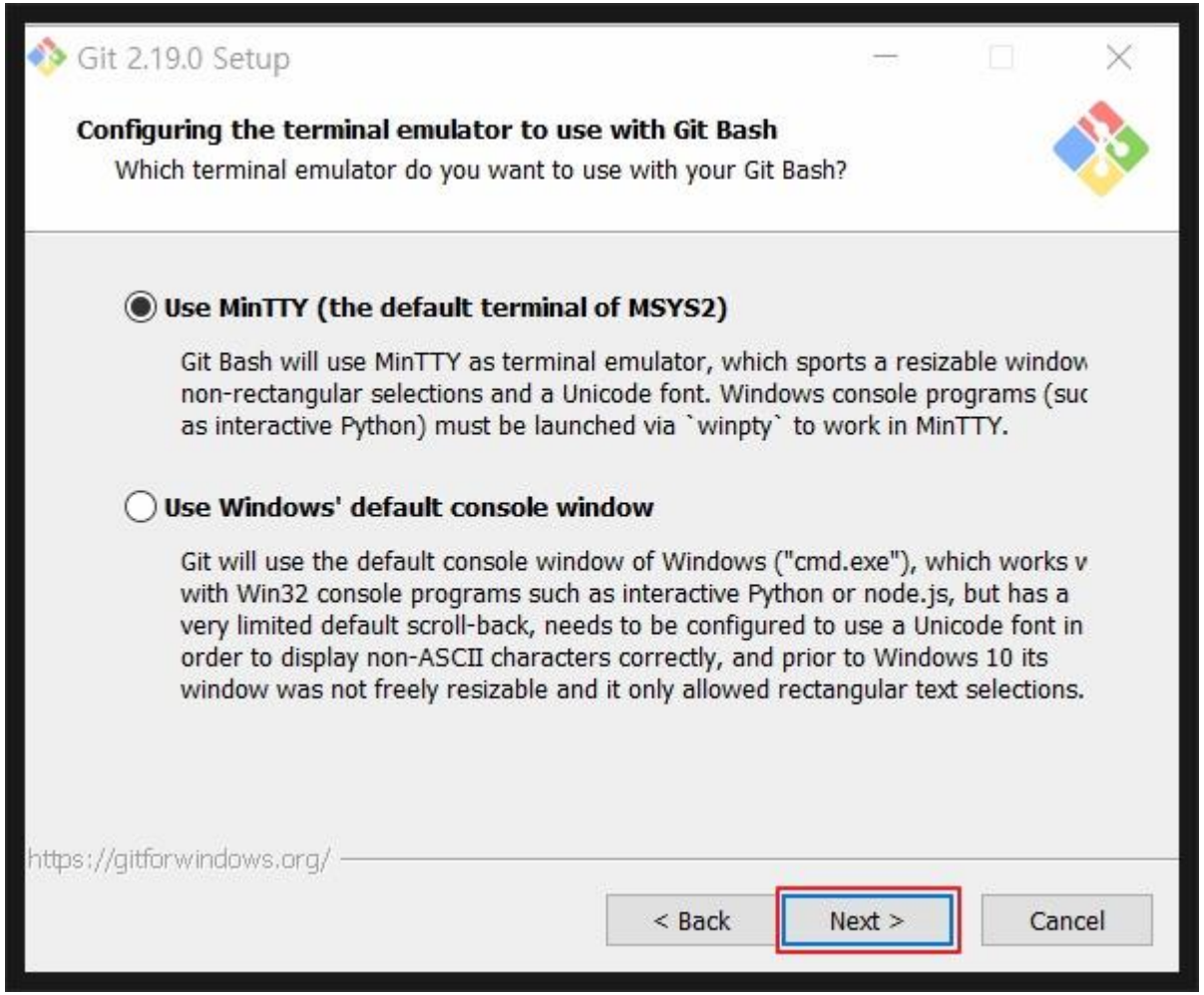
Git bash 설치


Git 2.19.0 Setup
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
Use the Nano editor by default
Use Vim (the ubiquitous text editor) as Git's default editor
Use Notepad++ as Git's default editor
Use Visual Studio Code as Git's default editor
Use Visual Studio Code Insiders as Git's default editor
https://gitforwindows.org/
< Back
Next >
Cancel

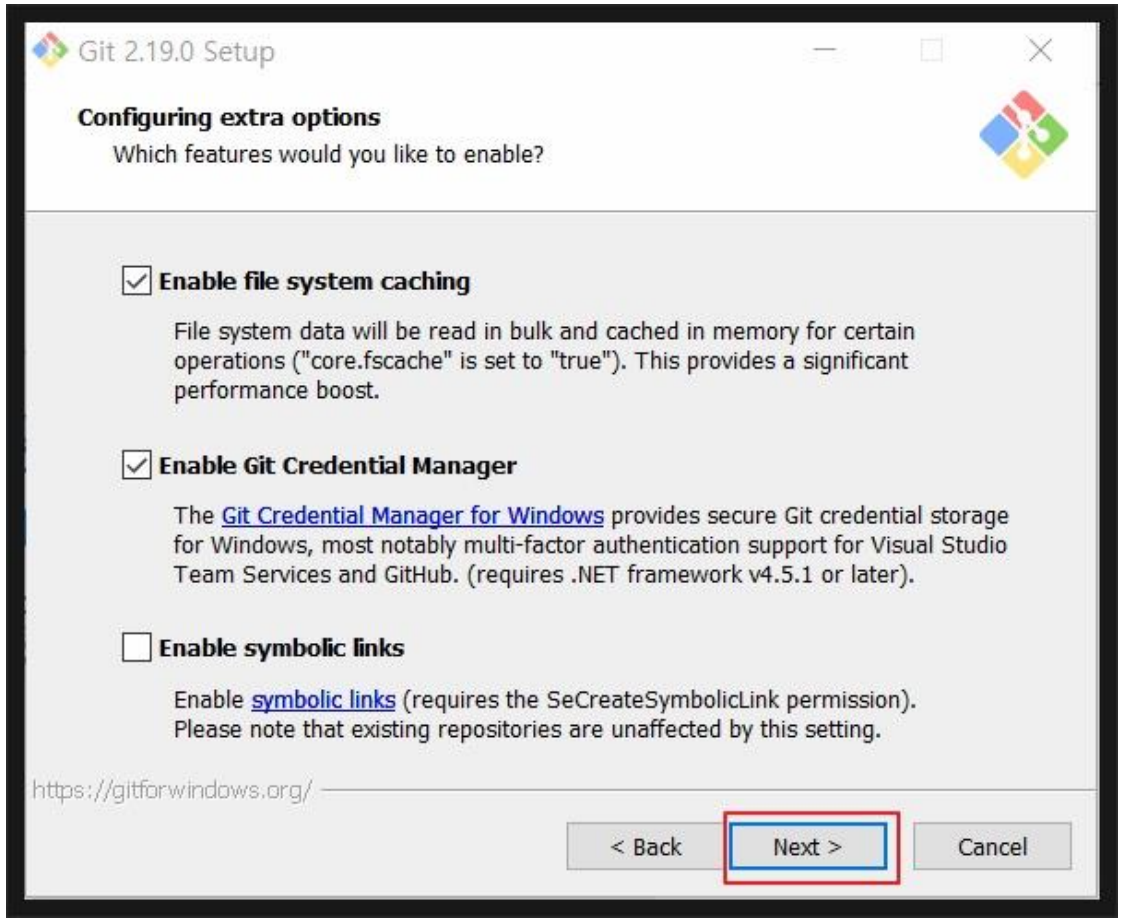

Git 2.19.0 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using 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

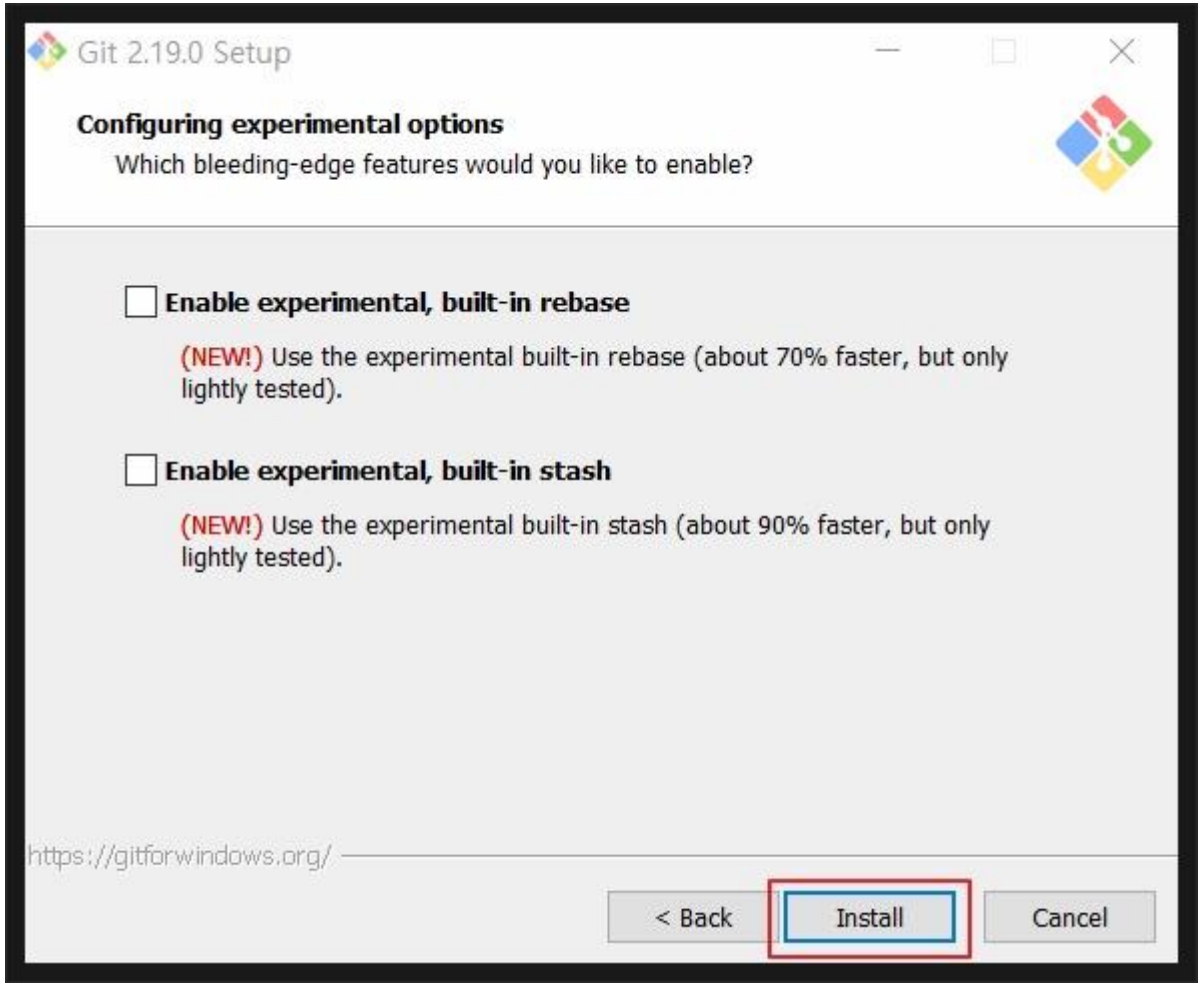
Git bash 설치


Git 2.19.0 Setup
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

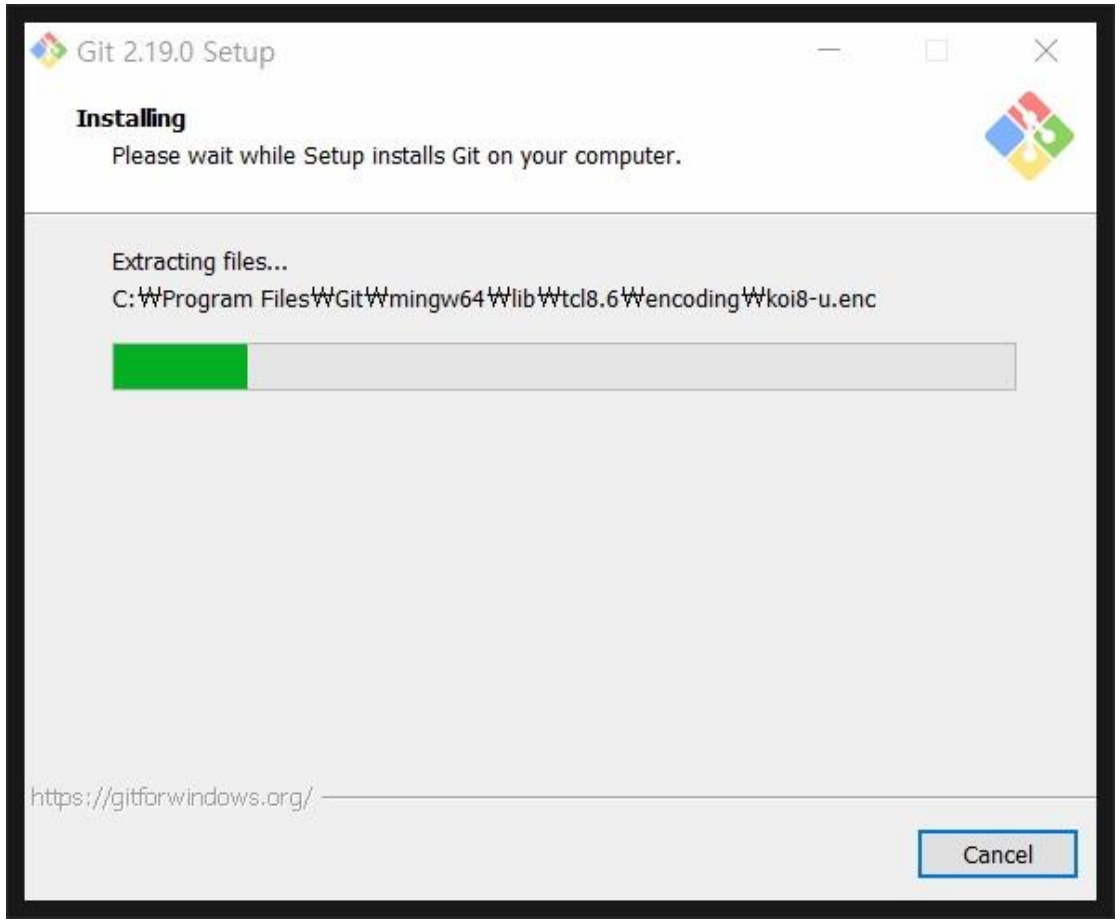

Git 2.19.0 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable windov non-rectangular selections and a Unicode font. Windows console programs (suc 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 v 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

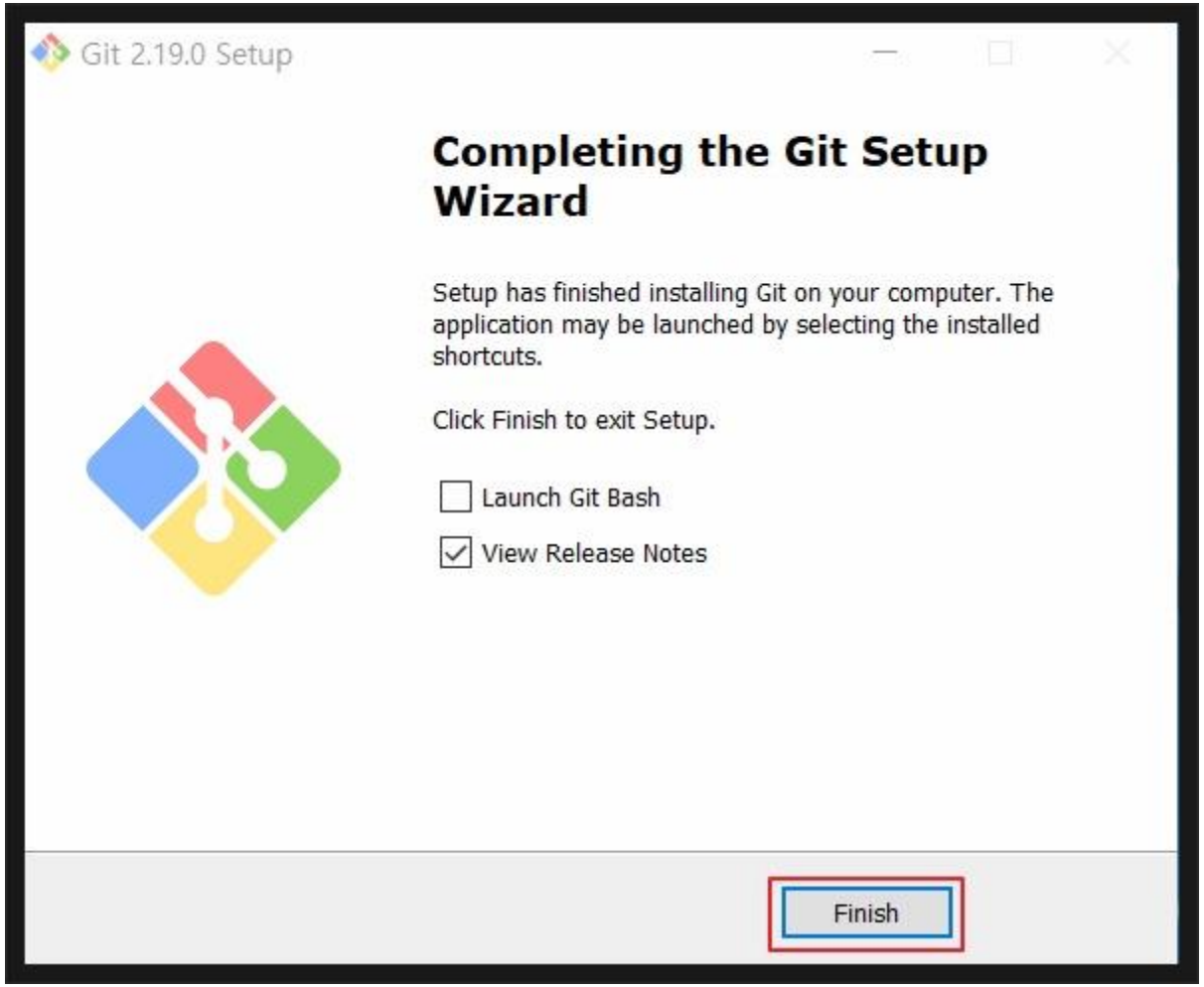
Git bash 설치


Git 2.19.0 Setup
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 later).
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


Git 2.19.0 Setup
Configuring experimental options
Which bleeding-edge features would you like to enable?
Enable experimental, built-in rebase
(NEW!) Use the experimental built-in rebase (about 70% faster, but only lightly tested).
Enable experimental, built-in stash
(NEW!) Use the experimental built-in stash (about 90% faster, but only lightly tested).
https://gitforwindows.org/
< Back
Install
Cancel

Git bash 설치





출처 https://coding-factory.tistory.com/245

Git 환경 설정

1. git init

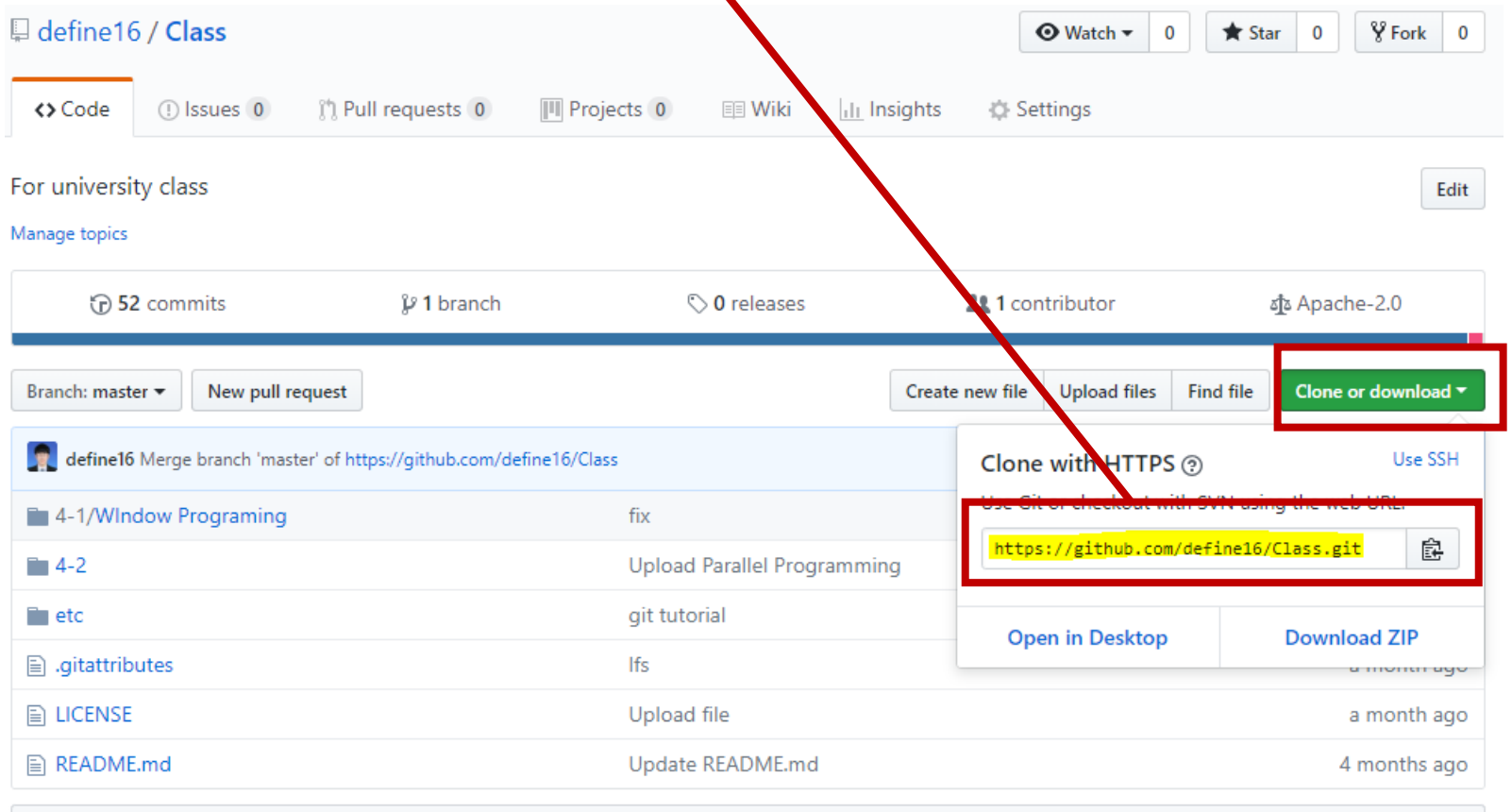
```
// 로컬저장소로 설정할 프로젝트 위치로 이동한다.
cd C:/dev/workspace/eom2017

// 로컬저장소로 설정한다.
// (master) 브랜치로 보이면 성공한 것이다.
git init

// 만약 init을 취소하려면 아래의 명령어를 입력한다.
rm -r .git
```

Git 다운로드 과정

1. git clone “git 저장소 주소”



Git 업로드 과정

1. git status

```
sujin@eom-PC MINGW64 /c/dev/workspace/eom2017 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .classpath
        .project
        .settings/
        .springBeans
        pom.xml
        src/
        target/

nothing added to commit but untracked files present (use "git add" to track)
```

Git 업로드 과정

- git romote

  로컬 저장소와 원격 저장소를 연결하는 과정, 연결이 되지 않았다면 해야함.

  // Github 원격저장소와 연결한다.
  git remote add origin [자신의 Github 원격저장소 주소]

  // 연결된 원격저장소 확인한다.
  remote -v

Git 업로드 과정

2. git add

```
// a.html 파일만 추가
git add a.html

// 워킹 디렉터리 내 모든 파일을 추가
git add .

// 명령 프롬프트에서 상호작용하면서 추가 (나갈땐 q를 입력)
git add -i

// 내 모든 파일을 추가
git add --all

// 진행중인 파일일 경우, Staging Area에서 워킹 디렉터리로 옮겨온다.
$git rm --cached a.html
$git rm -r --cached .
```

Git 업로드 과정

3. git commit

```
// 에디터가 출력되고, 에디터에서 커밋 메시지 입력 후 저장하면 커밋됨
git commit

// 간단한 커밋 메시지를 입력후 커밋
git commit -m "커밋 메시지"

// Staging Area에 들어간 파일에 대해서만 (워킹 디렉터리는 적용 X)
git commit -a -m "커밋 메시지"
cs
```

Git 업로드 과정

4. git push

```
// 원격저장소에 저장한다.
git push -u origin master

// 에러 - ! [rejected] master -> master (fetch first)
// 이미 변경된 파일이 원격저장소에 있을경우 발생
git pull origin master

// 에러 - ! [rejected] master -> master (non-fast-forward)
git push origin +master
```

출처 http://gbsb.tistory.com/10

Git 명령어

git init
깃 저장소를 초기화한다. 저장소나 디렉토리 안에서 이 명령을 실행하기 전까지는 그냥 일반 폴더이다. 이것을 입력한 후에야 추가적인 깃 명령어들을 줄 수 있다.

git config
"configure"의 준말, 처음에 깃을 설정할 때 가장 유용하다.

git help
명령어를 잊어버렸다? 커맨드 라인에 이걸 타이핑하면 21개의 가장 많이 사용하는 깃 명령어들이 나타난다. 좀 더 자세하게 "git help init"이나 다른 용어를 타이핑하여 특정 깃 명령어를 사용하고 설정하는 법을 이해할 수도 있다.

git status
저장소 상태를 체크. 어떤 화일이 저장소 안에 있는지, 커밋이 필요한 변경사항이 있는지, 현재 저장소의 어떤 브랜치에서 작업하고 있는지 등을 볼 수 있다.

git add
이 명령이 저장소에 새 화일들을 추가하진 않는다. 대신, 깃이 새 화일들을 지켜보게 한다. 화일을 추가하면, 깃의 저장소 "스냅샷"에 포함된다.

Git 명령어

git commit
깃의 가장 중요한 명령어. 어떤 변경사항이라도 만든 후, 저장소의 "스냅샷"을 찍기 위해 이것을 입력한다. 보통 "git commit -m "Message hear." 형식으로 사용한다. -m은 명령어의 그 다음 부분을 메시지로 읽어야 한다는 것을 말한다.

git branch
여러 협업자와 작업하고 자신만의 변경을 원한다? 이 명령어는 새로운 브랜치를 만들고, 자신만의 변경사항과 화일 추가 등의 커밋 타임라인을 만든다. 당신의 제목이 명령어 다음에 온다. 새 브랜치를 "cats"로 부르고 싶으면, git branch cats를 타이핑한다.

git checkout
글자 그대로, 현재 위치하고 있지 않은 저장소를 "체크아웃"할 수 있다. 이것은 체크하길 원하는 저장소로 옮겨가게 해주는 탐색 명령이다. master 브랜치를 들여다 보고 싶으면, git checkout master를 사용할 수 있고, git checkout cats로 또 다른 브랜치를 들여다 볼 수 있다.

git merge
브랜치에서 작업을 끝내고, 모든 협업자가 볼 수 있는 master 브랜치로 병합할 수 있다. git merge cats는 "cats" 브랜치에서 만든 모든 변경사항을 master로 추가한다.

Git 명령어

git push
로컬 컴퓨터에서 작업하고 당신의 커밋을 깃허브에서 온라인으로도 볼 수 있기를 원한다면, 이 명령어로 깃허브에 변경사항을 “push”한다.

git pull
로컬 컴퓨터에서 작업할 때, 작업하고 있는 저장소의 최신 버전을 원하면, 이 명령어로 깃허브로부터 변경사항을 다운로드한다(“pull”).

깃 저장 : status -> add -> commit -> push

깃 불러오기 : pull

Git 대용량 파일 업로드

- Git은 100MB이상 업로드시 경고 메시지를 보게 된다.

```
$ git push
Counting objects: 3086, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2980/2980), done.
Writing objects: 100% (3086/3086), 363.25 MiB | 935.00 KiB/s, done.
Total 3086 (delta 1236), reused 111 (delta 57)
remote: error: GH001: Large files detected. You may want to try Git
Large File Storage - https://git-lfs.github.com.
remote: error: Trace: ***
remote: error: See http://git.io/iEPt8g for more information.
remote: error: File *** is 120.94 MB; this exceeds GitHub's file
size limit of 100.00 MB
To git@github.com:***
 ! [remote rejected] master -> master (pre-receive hook declined)
 ! [remote rejected] *** -> *** (pre-receive hook declined)
error: failed to push some refs to 'git@github.com:***'
```

Git 대용량 파일 업로드

- 그래서 Commit 과정에서 지정한 파일을 작게 조각내주는 Git extensio인 git-lfs — *Git Large File Storage을 설치해야한다.*
- *https://git-lfs.github.com/* — 를 로컬에 설치한 뒤, 적용하려는 Repository 경로에서 다음 명령을 실행해야 한다.

```
$ git lfs install
Updated pre-push hook.
Git LFS initialized.
```

Git 대용량 파일 업로드

- 그다음 용량이 큰 파일을 git-lfs의 관리 대상으로 등록해준다. 다음 예시는 120MB 정도의 exe 파일을 Stage에 추가한 상황에서, 확장자가 exe인 모든 파일을 git-lfs의 관리 대상으로 지정하고 Commit을 수행한 모습이다.

```
$ git lfs track "*.exe"
Tracking *.exe

$ git commit -m "Large file included"
[master (root-commit) dd2b715] Large file included
(...)
```

이제 하단에 있는 3번 과정대로 Github에 push를 시도하면 된다.

Git 대용량 파일 업로드

- 그런데 기존에 100MB 이상의 파일을 Commit한 적이 있다면 여전히 100MB 이상의 파일을 올릴 수 없다는 경고 메시지를 보게 된다. 그럴 땐 다음 해당 과정을 적용해야 한다.
- **BFG Repo-Cleaner 적용** BFG다운로드 클릭
- 기존 Commit에서 100MB보다 큰 파일의 로그를 강제로 없애줘야 한다. BFG Repo-Cleaner — BFG Repo-Cleaner https://rtyley.github.io/bfg-repo-cleaner/ — 를 이용하면 그 작업을 손쉽게 적용할 수 있다.

Git 대용량 파일 업로드

- 공식 사이트에서 bfq-x.x.x.jar — x.x.x는 버전 — 를 받고, 대상이 되는 Repository에서 다음과 같이 그동안의 Commit에 포함된 100MB 이상의 파일을 정리하는 명령을 실행한다.

```
$ java -jar bfg-x.x.x.jar --strip-blobs-bigger-than 100M
(...)
Deleted files
    - - - - - - -
    Filename Git id
    - - - - - - - - - - - - - - - -
    ***.exe | c304fcfb (120.9 MB)
(...)
```

Git 대용량 파일 업로드

- 간혹 다음과 같은 오류가 나타날 수 있다.

```
$ java -jar bfg-x.x.x.jar --strip-blobs-bigger-than 100M

Using repo : C:\***\.git

Scanning packfile for large blobs: 132
Scanning packfile for large blobs completed in 13 ms.
Warning : no large blobs matching criteria found in packfiles - does
the repo need to be packed?
Please specify tasks for The BFG :
bfg x.x.x
(...)
```

Git 대용량 파일 업로드

- 그럴 땐 아래 명령을 먼저 수행하고 다시 위의 bfg-x.x.x.jar에 의한 명령을 실행한다.

```
$ git repack && git gc
Counting objects: 3002, done.
(...)
```

- 이후 git push 재시도

출처 https://medium.com/@stargt/github%EC%97%90-100mb-%EC%9D%B4%EC%83%81%EC%9D%98-%ED%8C%8C%EC%9D%BC%EC%9D%84-%EC%98%AC%EB%A6%AC%EB%8A%94-%EB%B0%A9%EB%B2%95-9d9e6e3b94ef