

2장 깃 설치 및 환경 설정

- 2.1 깃 설치
- 2.2 소스트리 설치
- 2.3 첫 번째 깃 실행
- 2.4 환경 설정
- 2.5 비주얼 스튜디오 코드
- 2.6 정리

1

2.1 깃 설치

2

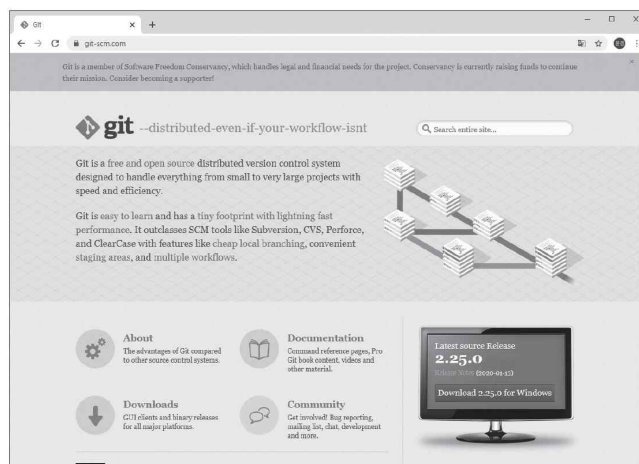
1. 깃 설치

➤ 깃 설치

- 윈도우, 리눅스, macOS 등 여러 운영 체제에서 깃을 설치하고 사용할 수 있음
- 깃은 오픈 소스이기 때문에 공개된 소스를 직접 내려받아 설치할 수 있지만, 컴파일 작업과 설치가 쉽지 않음
- 깃의 개발을 이끄는 단체에서는 일반 개발자도 쉽게 사용할 수 있도록 운영 체제별 설치 프로그램을 만들어 제공함
- 깃 공식 사이트(<https://git-scm.com>)에서 운영 체제별로 만든 배포판을 내려받아 설치할 수 있음

1. 깃 설치

▼ 그림 2-1 깃 공식 사이트



1. 깃 설치

➤ 깃 설치

- 깃의 기본 작업 환경은 터미널 모드임
- 리눅스나 macOS 사용자라면 콘솔창 환경에 익숙할 것임
- 윈도우 사용자라면 콘솔창을 처음 접하는 사람도 있을 수 있음
- 과거 MS-DOS 시절에는 콘솔 명령어를 주로 사용함
- 윈도우 그래픽 운영 체제가 널리 보급되면서 콘솔창에서 작업하는 빈도가 많이 줄었음
- 리눅스와 윈도우 계열의 콘솔 명령어는 조금 차이가 있음
- 깃은 배시(bash) 환경을 제공함
- 깃 배시는 윈도우, 리눅스, macOS 사용자 모두 동일한 콘솔 명령어로 작업할 수 있도록 통일된 환경을 제공함
- 터미널 작업:
콘솔창에서 직접 명령어를 입력하여 동작하게 하는 것임

1. 깃 설치

➤ 윈도우에서 설치

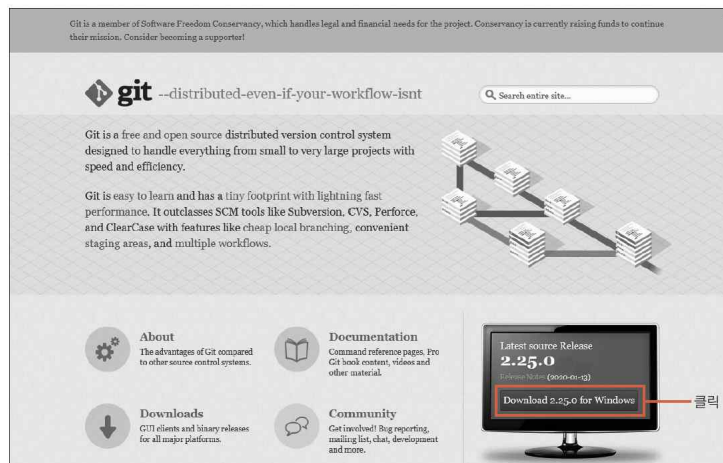
- 마이크로소프트의 윈도우는 전 세계적으로 가장 많이 사용하는 운영 체제임
- 윈도우 운영 체제에서 깃을 설치하는 방법을 알아보자

설치 파일 내려받기

- 깃 공식 사이트(<https://git-scm.com>)에 접속하여 윈도우용 설치 파일을 내려받음
- 웹 사이트 오른쪽 아래에 위치한 설치 파일을 클릭하여 내려받음
- 설치 파일 내려받기가 완료되면 윈도우의 다운로드 폴더에 저장됨

1. 깃 설치

▼ 그림 2-2 깃 설치 파일 내려받기



1. 깃 설치

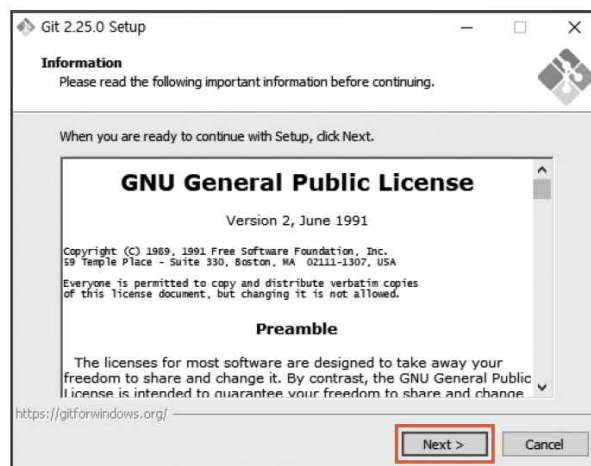
➤ 윈도우에서 설치

설치

- 내려받은 설치 파일을 더블클릭하여 실행함
- 다음과 같이 깃 설치 관리자 프로그램을 실행함
- 깃은 GNU 오픈 소스 라이선스 형태로 배포함
- 라이선스에 동의한다면 Next를 누름

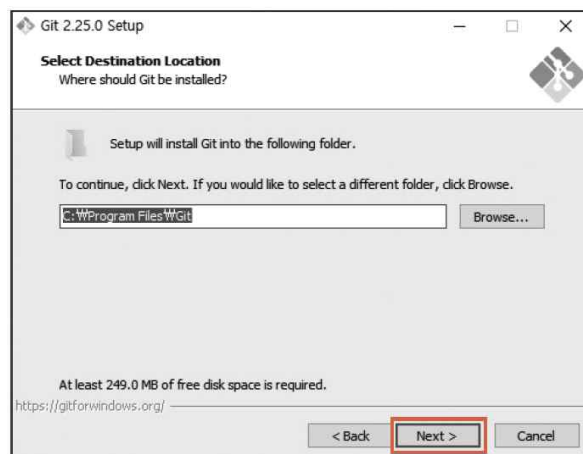
1. 깃 설치

▼ 그림 2-3 깃 설치 시작



1. 깃 설치

▼ 그림 2-4 설치 경로 확인



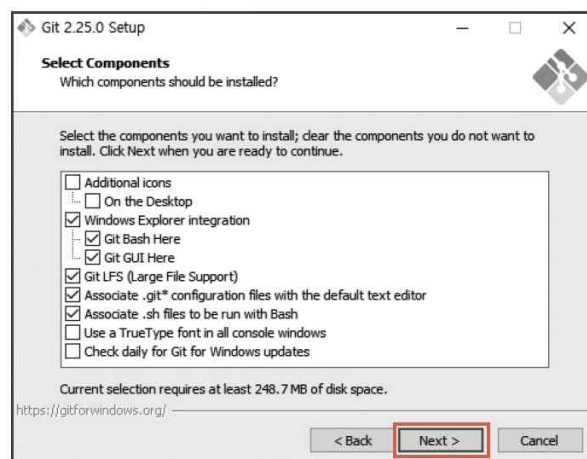
1. 깃 설치

➤ 윈도우에서 설치

- 깃의 설치 프로그램에는 다양한 컴포넌트가 있음
- 설치 과정에서 필요한 컴포넌트를 다음과 같이 선택 가능함
- 보통 기본 설정 값을 이용하면 무난하게 설치할 수 있음
- 책에서도 기본값을 사용함

1. 깃 설치

▼ 그림 2-5 컴포넌트 선택



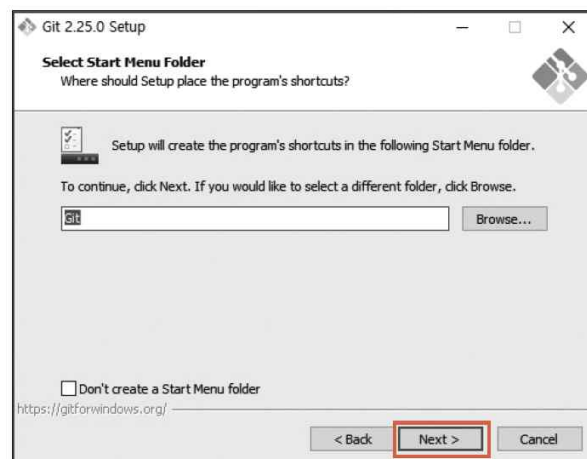
1. 깃 설치

➤ 윈도우에서 설치

- 윈도우에서는 유사한 응용 프로그램을 몇 개 묶어서 프로그램 메뉴 목록 형태로 관리함
- 깃 역시 관련 있는 실행 프로그램을 묶어서 등록함
- 등록 목록의 이름을 직접 지정할 수도 있지만, 'Git'이란 이름이 기본값으로 설정되어 있음
- 기본값으로 두고 Next를 누름

1. 깃 설치

▼ 그림 2-6 목록 이름 입력



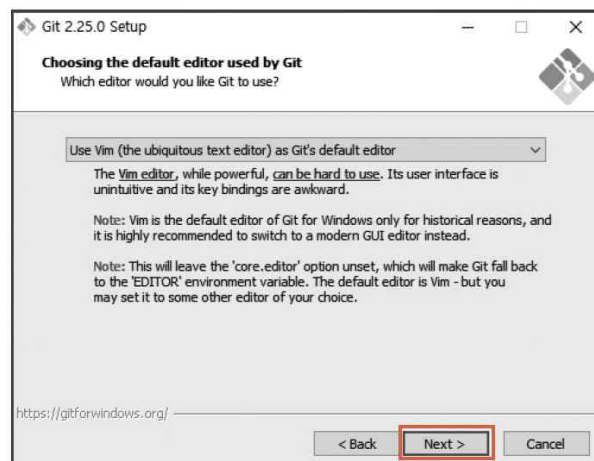
1. 깃 설치

➤ 윈도우에서 설치

- 깃은 커밋 등 작업을 처리할 때 메시지를 입력함
- 주석 문장을 작성하거나 편집할 때는 에디터가 필요한데 깃은 별도의 외부 에디터 프로그램을 사용함
- 기본값으로 Vim이 선택되어 있음
- 다른 외부 편집기를 사용하고 싶다면 직접 지정하면 됨
- 기본값인 Vim으로 두고 Next를 누름

1. 깃 설치

▼ 그림 2-7 에디터 선택



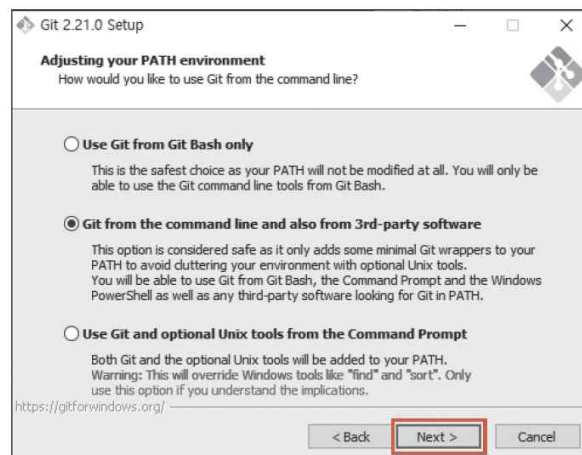
1. 깃 설치

➤ 윈도우에서 설치

- 다음은 깃 배시 명령어를 실행할 수 있는 경로를 설정하는 화면임
- 기본값으로 두고 Next를 누름

1. 깃 설치

▼ 그림 2-8 경로 설정



1. 깃 설치

➤ 윈도우에서 설치

- **Use Git from Git Bash only:** 깃 배시 터미널로만 깃을 사용할 수 있음
윈도의 환경 변수를 설정하지 않음
- **Git from the command line and also from 3rd-party software:** 기본 설정
값임
윈도용 cmd 창에서도 git 명령어를 사용할 수 있음
윈도의 환경 변수를 추가함
- **Use Git and optional Unix tools from the Command Prompt:** 윈도용 cmd
창에서 git과 유닉스 도구 명령어를 사용할 경우 선택함
유닉스 관련 도구 옵션을 추가함

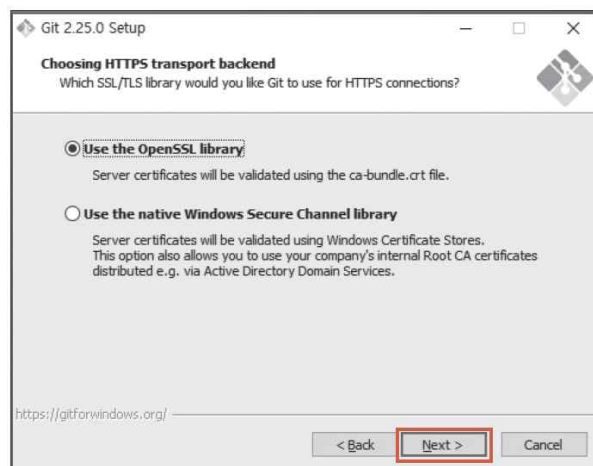
1. 깃 설치

➤ 윈도우에서 설치

- 깃은 SSH 프로토콜 통신을 이용하여 서버 간 코드 이력을 전송할 수 있음
- 다음은 SSH 프로토콜 통신과 관련된 설정을 하는 화면임
- 기본값으로 두고 Next를 누름

1. 깃 설치

▼ 그림 2-9 SSH 설정



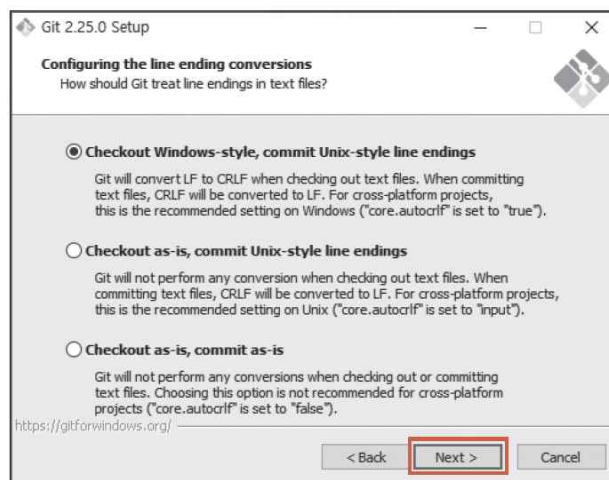
1. 깃 설치

➤ 윈도우에서 설치

- 파일의 엔딩 라인(ending line)을 처리하는 방법은 운영 체제별로 약간 차이가 있음
- 깃은 다양한 운영 체제와 엔딩 라인을 처리할 수 있도록 선택 화면을 제공함
- 기본값으로 두고 Next를 누름

1. 깃 설치

▼ 그림 2-10 엔딩 라인 처리 선택



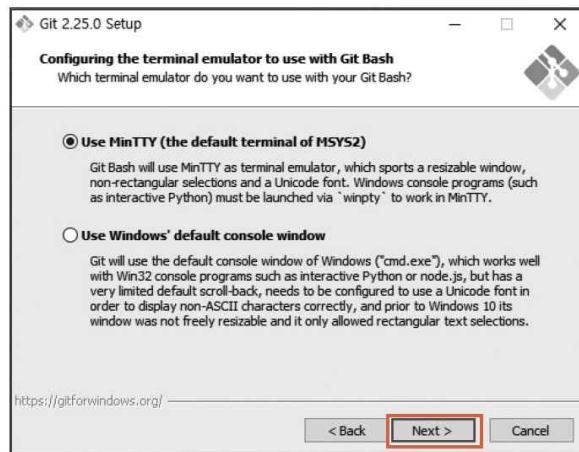
1. 깃 설치

➤ 윈도우에서 설치

- 다음은 터미널 에뮬레이터를 선택하는 화면임
- 기본값으로 두고 Next를 누름

1. 깃 설치

▼ 그림 2-11 터미널 에뮬레이터 선택



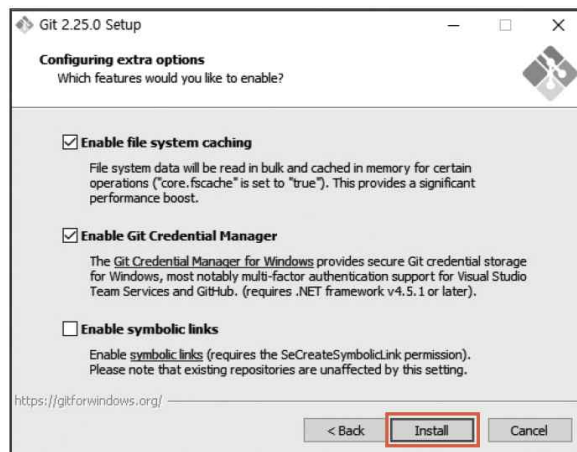
1. 깃 설치

➤ 윈도우에서 설치

- 다음은 추가 옵션 화면임
- 기본값으로 두고 Install을 누름

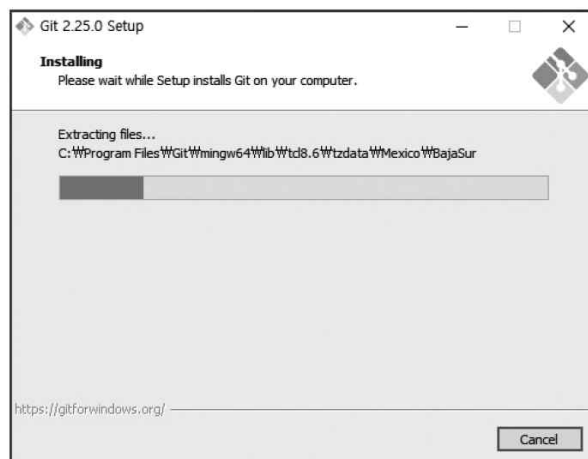
1. 깃 설치

▼ 그림 2-12 그 외 옵션들



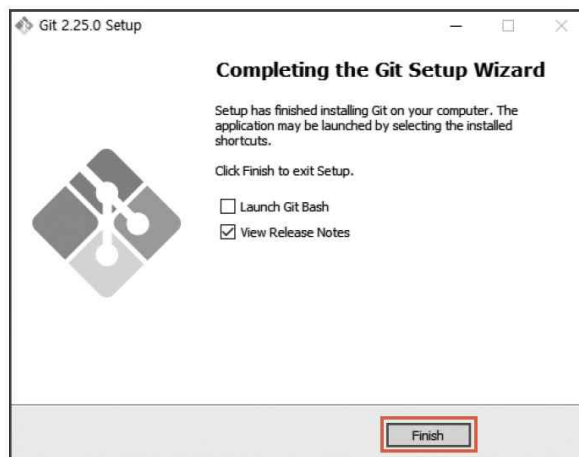
1. 깃 설치

▼ 그림 2-13 설치 진행



1. 깃 설치

▼ 그림 2-14 깃 설치 완료



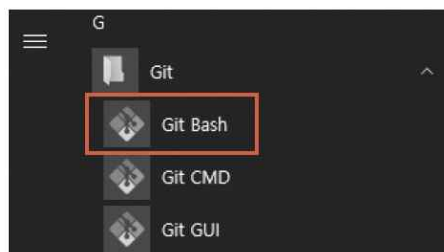
1. 깃 설치

➤ 윈도우에서 설치

설치 확인

- 윈도우 운영 체제에 깃이 정상적으로 설치되었는지 확인해 보자
- 윈도우 버튼을 클릭하고, 프로그램 목록에서 Git을 찾을
- 목록 폴더에 바로가기 아이콘 3개가 추가되어 있는지 확인함
- 윈도우용 깃은 리눅스와 동일한 명령어를 실행할 수 있도록 Git Bash(깃 배시)를 제공함

▼ 그림 2-15 윈도우 메뉴의 Git



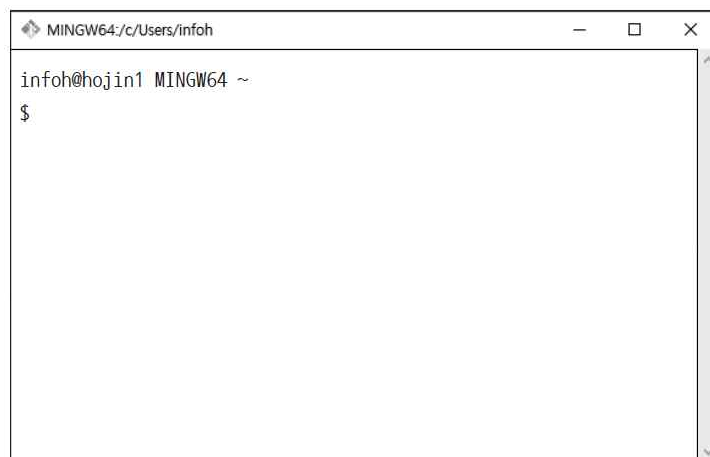
1. 깃 설치

➤ 윈도우에서 설치

- Git Bash를 선택하면 다음과 같이 터미널 콘솔창을 실행함
- 깃의 동작 명령어는 대부분 깃 배시 콘솔창에서 실행함

1. 깃 설치

▼ 그림 2-16 깃 배시 콘솔창



1. 깃 설치

➤ 윈도우에서 설치

- 콘솔창을 실행하면 다음과 같이 명령 프롬프트가 실행됨

컴퓨터이름 MINGW64 ~

\$

- 깃 배시는 현재의 작업 디렉터리 경로를 표시함
- 처음 깃 배시를 시작하면 경로 주소 이름이 물결(~)로 표시됨
- ~ 기호는 현재 자신의 계정 위치를 의미함
- 리눅스 계열의 운영 체제는 명령 프롬프트 기호로 달러(\$)를 사용함
- \$ 기호는 계정 아이디로 접속했을 때를 의미함
- 관리자 root로 접속하면 # 기호로 표시됨

1. 깃 설치

➤ 리눅스에서 설치

- 리눅스는 유닉스에서 파생된 운영 체제임
- 전 세계에서 가장 많이 운영하는 서버 환경임
- 클라우드와 서버용 환경을 같이 개발한다면 리눅스용 깃 환경이 필요할 것임
- 리눅스 운영 체제에서 깃을 설치하는 방법은 간단함
- 대부분의 리눅스 배포판은 관련된 응용 프로그램을 패키지 형태로 제공함
- 패키지는 리눅스 배포판마다 약간 차이가 있음
- 리눅스 패키지 관리 도구를 이용하면 깃 공식 사이트에서 설치 파일을 내려받지 않고도 설치할 수 있음
- 소스 코드를 직접 내려받아 복잡한 컴파일 작업을 하지 않아도 되므로 편리함

1. 깃 설치

➤ 리눅스에서 설치

- 깃을 설치하려면 먼저 자신의 리눅스 배포판과 환경을 알아야 함
- 다음과 같이 입력하여 자신의 환경을 확인함

```
root@hojin1:~# cat /etc/issue
```

```
Ubuntu 18.04.2 LTS \n \l
```

- 리눅스 배포판 서버에서 패키지를 내려받아 설치함
- 예를 들어 자신의 리눅스 배포판이 페도라(Fedora), CentOS 계열이라면 yum 명령어로 깃을 설치할 수 있음

예

```
$ sudo yum install git
```

1. 깃 설치

➤ 리눅스에서 설치

- 데비안 계열의 우분투 리눅스라면 apt 명령어를 사용함

예

```
$ sudo apt install git
```

- 최신 리눅스 배포판은 기본적으로 설치 패키지에 깃을 포함함
- 별도로 설치하지 않아도 깃을 사용할 수 있음
- 콘솔창에서 git --version 명령어로 깃이 설치되었는지 확인할 수 있음

```
root@hojin1:~# git --version
```

```
git version 2.17.1
```

1. 깃 설치

> 윈도우에서 설치

삭제 후 재설치

- 배포판에 설치된 깃이 오래된 버전이라면, 패키지를 삭제하고 다시 설치할 수도 있음
- remove 옵션을 사용하여 기존에 설치된 깃을 제거할 수 있음

예

```
root@hojin1:~# apt remove git
Reading package lists... Done
Building dependency tree
Reading state information... Done
생략...
Removing ubuntu-server (1.417.1) ...
Removing git (1:2.17.1-1ubuntu0.4) ...
```

1. 깃 설치

> 윈도우에서 설치

- 설치된 깃을 삭제했다면, 새로운 깃 버전을 패키지 관리자로 설치하면 됨

예

```
root@hojin1:~# apt install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
생략...
Preparing to unpack .../git_1%3a2.17.1-1ubuntu0.4_amd64.deb ...
Unpacking git (1:2.17.1-1ubuntu0.4) ...
Setting up git (1:2.17.1-1ubuntu0.4) ...
```

1. 깃 설치

➤ macOS에서 설치

- 최근 들어 macOS 운영 체제를 개발 환경으로 사용하는 사람이 많아짐
- macOS도 유닉스와 비슷한 NextStep을 기반으로 발전된 운영 체제이기 때문에 대부분의 사용법은 리눅스와 비슷함
- 리눅스와 마찬가지로 최근 macOS 시스템에는 대부분 깃이 기본으로 설치되어 있음
- 특정 버전의 깃을 설치하고 싶다면 깃 공식 사이트에 접속하여 macOS용 git 패키지를 내려받음
- dmg 파일을 내려받은 후 dmg 파일 안에 있는 pkg 파일을 실행하면 깃이 설치됨

2.2 소스트리 설치

2. 소스트리 설치

➤ 소스트리 설치

- 소스트리는 아틀라시안(Atlassian)에서 배포하는 GUI 깃 도구임
- 소스트리를 사용하려면 기본적으로 깃을 컴퓨터에 설치해야 함
- 소스트리는 깃을 GUI로 사용하는 외부 도구일 뿐임

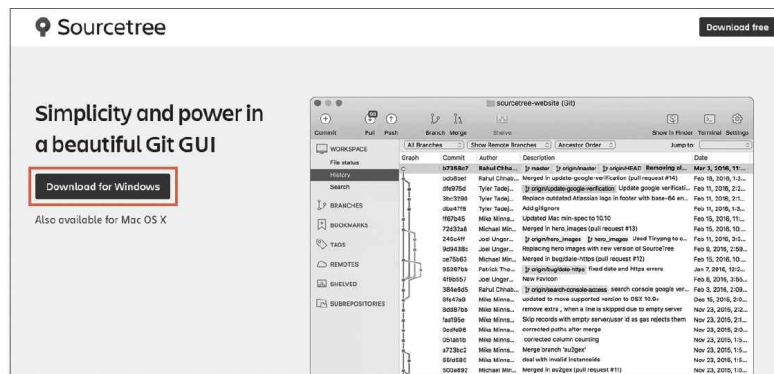
2. 소스트리 설치

➤ 설치 파일 내려받기

- 소스트리는 공식 사이트에서 무료로 내려받을 수 있음
- 먼저 소스트리 공식 사이트(<https://www.sourcetreeapp.com>)에 접속함
- 운영 체제에 맞는 다운로드 버튼을 클릭함

2. 소스트리 설치

▼ 그림 2-17 소스트리 공식



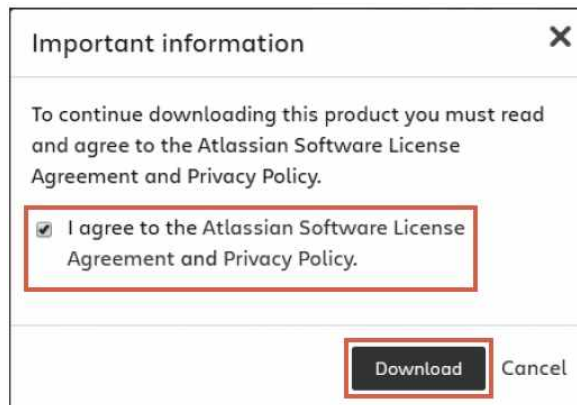
2. 소스트리 설치

➤ 설치 파일 내려받기

- 그림 2-18과 같은 화면이 나오면 라이선스에 동의한 후 Download를 눌러 설치 파일을 내려받음
- 소스트리는 윈도우와 macOS용 모두 제공하므로 자신의 컴퓨터 환경에 맞는 버전을 내려받으면 됨
- 책은 윈도우를 기반으로 설명함

2. 소스트리 설치

▼ 그림 2-18 라이선스 동의

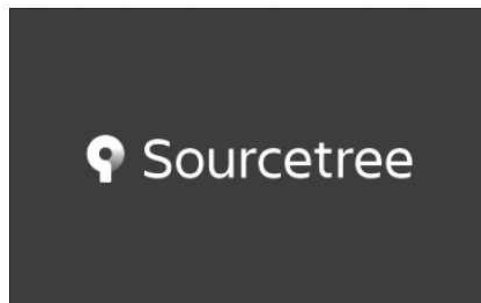


2. 소스트리 설치

➤ 설치

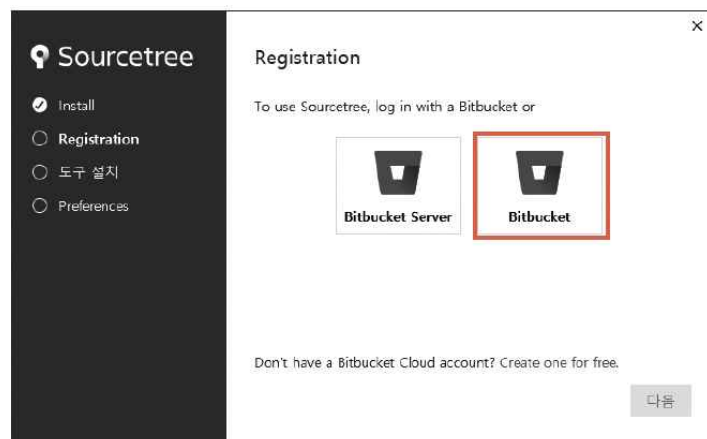
- 설치 파일을 내려받았다면 이제 소스트리를 설치함
- 먼저 내려받은 설치 파일을 실행함
- 잠시 기다리면 설치 화면이 나옴

▼ 그림 2-19 소스트리 설치 화면



2. 소스트리 설치

▼ 그림 2-20 [Bitbucket] 메뉴 선택

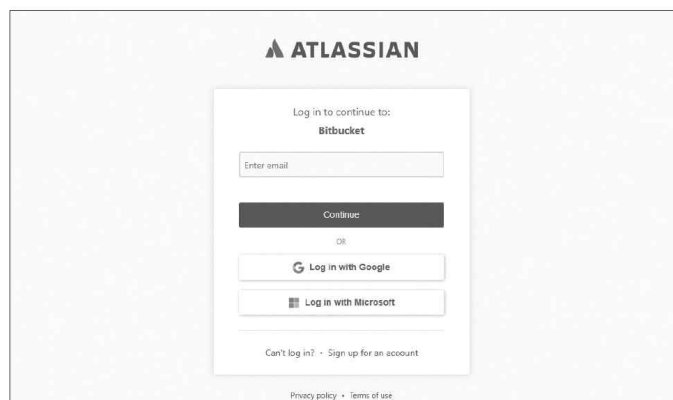


2. 소스트리 설치

➤ 설치

- 로그인 화면이 나오면 회원 가입을 하거나 로그인함

▼ 그림 2-21 아틀라시안 웹 사이트 로그인



2. 소스트리 설치

➤ 설치

- 정상적으로 회원 가입 및 로그인을 완료하면 등록 완료 화면으로 전환됨

▼ 그림 2-22 로그인 확인 중



2. 소스트리 설치

▼ 그림 2-23 등록 완료



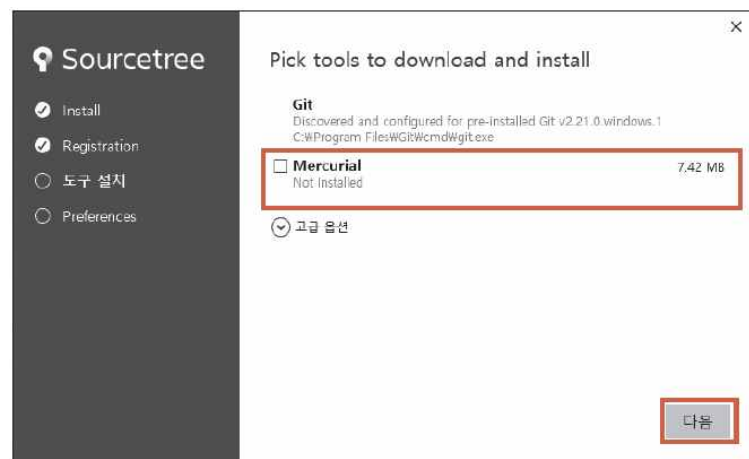
2. 소스트리 설치

➤ 설치

- 다음으로 본격적인 소스트리 설치 과정을 진행함
- 소스트리는 깃 외에 머큐리얼(Mercurial) 같은 버전 관리를 동시에 지원함
- 깃만 사용한다면 Mercurial 체크는 해제함
- 책에서는 Mercurial 체크를 해제한 후 다음을 누름

2. 소스트리 설치

▼ 그림 2-24 'Mercurial' 체크 해제



2. 소스트리 설치

➤ 설치

- 깃을 사용하려면 환경 설정을 해야 함
- 먼저 깃으로 이력을 관리할 때 사용할 이름과 이메일 주소를 설정함
- 첫 번째 항목에는 이름(영문)을 입력하고, 두 번째 항목에는 이메일 주소를 입력함
- 아틀라시안 계정이 기본값으로 입력되어 있으므로 그대로 사용해도 되고, 전혀 다르게 원하는 이름과 이메일 주소를 사용해도 됨
- 모두 입력했다면 다음을 누름

2. 소스트리 설치

▼ 그림 2-25 환경 설정용 이름과 이메일 주소

Sourcetree

- Install
- Registration
- 도구 설치
- Preferences

Preferences

Before we finish, take a moment to configure these settings.

☐ 모든 저장소에 이 세부 내용 활용

☒ Help improve Sourcetree by sending 사용 데이터

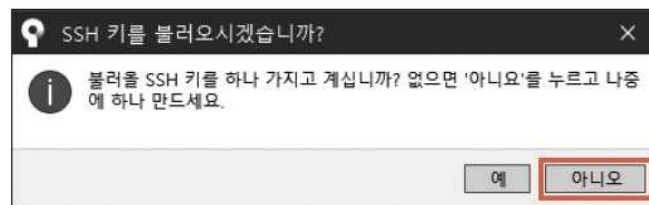
다음

2. 소스트리 설치

> 설치

- 소스트리는 깃허브, 비트버킷과 연동하여 원격 저장소(서버) 작업을 같이 수행할 수 있음
- 원격 저장소를 사용하려면 깃허브, 비트버킷에 로그인해야 함
- SSH키를 이용하여 로그인 할 수도 있음
- 아직 아무것도 준비되어 있지 않으니 다음 화면이 나오면 일단 아니오를 눌러 넘어감

▼ 그림 2-26 SSH키 로그인 설정



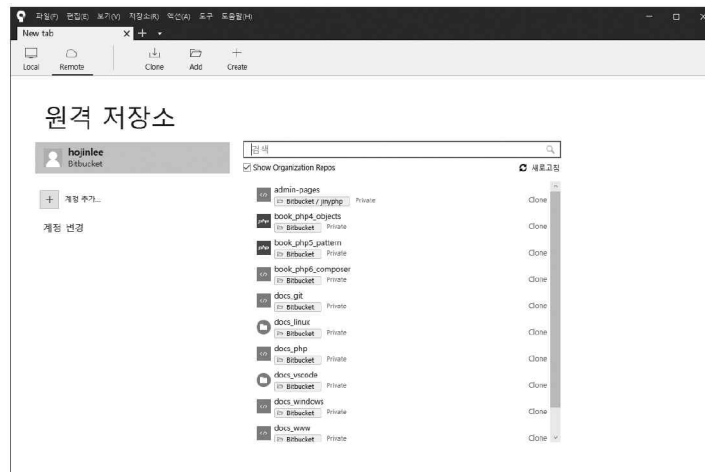
2. 소스트리 설치

> 설치

- 그림 2-27과 같이 소스트리 설치가 완료되면 프로그램을 실행함
- 소스트리는 비트버킷에 생성된 원격 저장소 목록을 같이 나열하여 표시함
- 깃허브의 원격 저장소도 같이 사용 할 수 있음
- 깃과 소스트리를 처음 설치했다면 아무 목록도 표시하지 않음

2. 소스트리 설치

▼ 그림 2-27 원격 저장소 목록



2.3 첫 번째 깃 실행

3. 첫 번째 깃 실행

➤ 첫 번째 깃 실행

- 깃의 기본 사용 환경은 터미널임

3. 첫 번째 깃 실행

➤ 터미널

- 터미널 환경:
예전의 MS-DOS 또는 셸(shell)처럼 텍스트 명령어를 입력하고 실행하는 환경을 의미함
- 터미널을 이용하면 깃 작업을 좀 더 다양하게 할 수 있음
- GUI는 많이 사용하는 기능 위주로 동작을 구현했기 때문에 터미널보다 정밀한 작업은 하기 어려움

3. 첫 번째 깃 실행

➤ 깃 명령어로 실행

- 깃 프로그램 이름은 git임
- 터미널에서 git과 명령어를 입력하면 깃이 동작함

```
$ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        <command> [<args>]
```

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)

clone	Clone a repository into a new directory
init	Create an empty Git repository or reinitialize an existing one

이하 생략

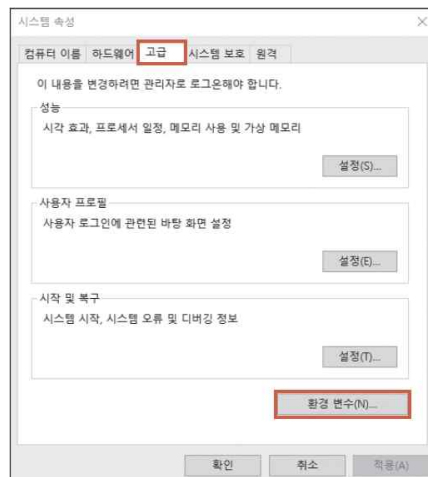
3. 첫 번째 깃 실행

➤ 깃 명령어로 실행

- git 명령어를 입력했는데도 제대로 실행되지 않았다면, 다음과 같이 윈도우의 환경 변수에 경로(PATH)를 추가해주어야 함
- 기본적으로 모든 디렉터리 경로에서 깃이 실행되도록 하고 싶을 때도 환경 변수에 경로를 설정해 줌

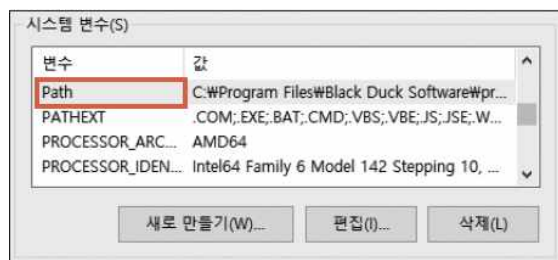
3. 첫 번째 깃 실행

▼ 그림 2-28 환경 변수 설정



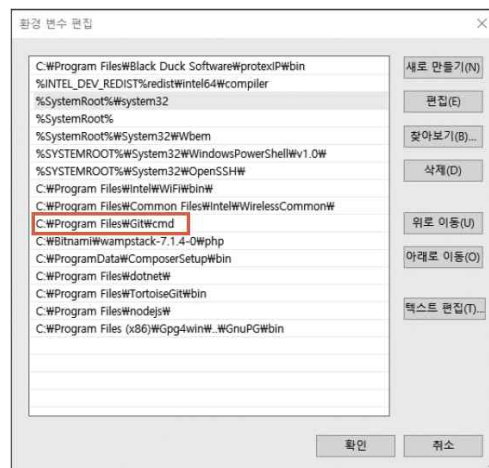
3. 첫 번째 깃 실행

▼ 그림 2-29 Path 수정



3. 첫 번째 깃 실행

▼ 그림 2-30 C:\Program Files\Git\cmd 경로 추가



3. 첫 번째 깃 실행

➤ 깃 명령어로 실행

- 깃은 명령어 하나로만 동작하는 방법과 옵션을 같이 사용하여 동작하는 명령어로 나뉨
- 기본적으로 깃은 독립된 명령어들로 구성되어 있음
- 독립적인 명령어들 외에 하위 명령어들을 같이 이용하면 동작 기능을 세분화할 수 있음

\$ git 명령어 또는 옵션

3. 첫 번째 깃 실행

➤ 깃 명령어로 실행

- 하위 명령어 외에 추가 옵션도 같이 지정할 수 있음
- 일부 옵션은 생략하기도 함
- 옵션은 짧은 옵션(-)과 긴 옵션(--)으로 구분함

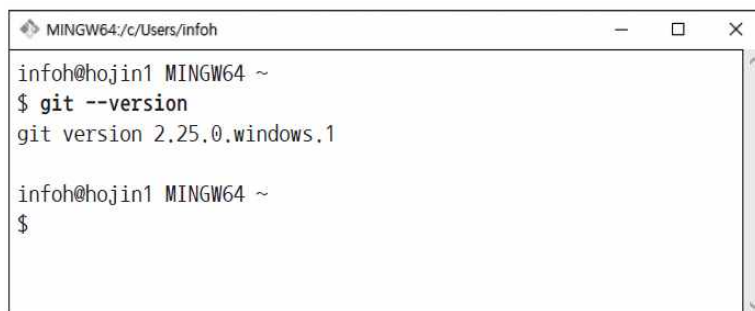
```
$ git help --all
```

- 이제 깃 배시를 실행하여 간단한 실습을 따라해 보자
- 다음과 같이 git --version 명령어를 입력하고 실행해 보자

```
$ git --version
```

3. 첫 번째 깃 실행

▼ 그림 2-31 git - -version 명령어 실행



```

MINGW64: c:/Users/infoh
infoh@hojin1 MINGW64 ~
$ git --version
git version 2.25.0.windows.1

infoh@hojin1 MINGW64 ~
$
  
```

3. 첫 번째 깃 실행

➤ 깃 명령어로 실행

- 깃 명령어는 명령어를 여러 개 묶어서 사용할 수도 있음
- 명령어를 묶어서 입력할 때는 세미콜론(;)으로 구분함

```
$ git tag; git branch
0.0.1
0.0.2
    footer
* master
```

----- 세미콜론으로 구분

3. 첫 번째 깃 실행

➤ 소스트리로 실행

- 소스트리를 같이 설치했다면 바탕화면에서 소스트리 아이콘을 확인할 수 있음

▼ 그림 2-32 소스트리 아이콘



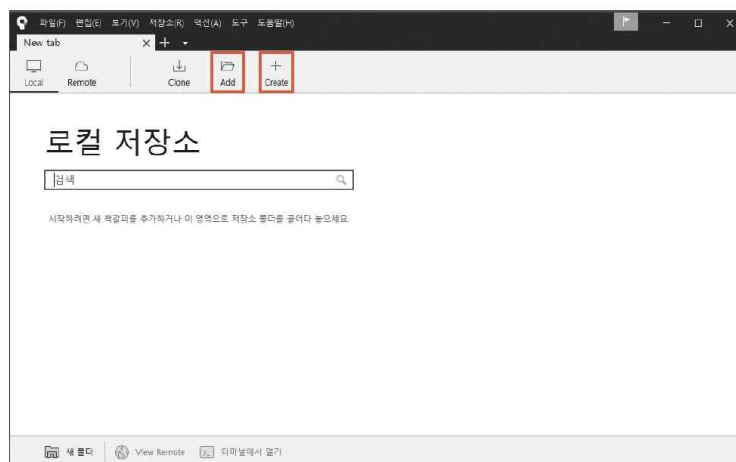
3. 첫 번째 깃 실행

➤ 소스트리로 실행

- 아이콘을 클릭하면 소스트리를 실행함
- 실행하면 다음과 같이 위쪽에 여러 버튼이 있음
- 여기서+ Create 버튼을 클릭하면 새로운 깃 저장소 폴더를 만들 수 있음
- Add 버튼을 클릭하면 로컬 PC에 만들어진 깃 저장소 폴더를 찾아 소스트리에 추가할 수 있음

3. 첫 번째 깃 실행

▼ 그림 2-33 소스트리 첫 실행 화면

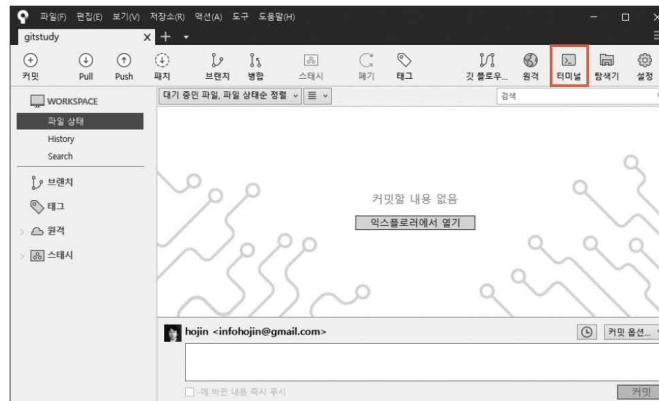


3. 첫 번째 깃 실행

➤ 소스트리로 실행

- 다음과 같이 로컬 PC의 깃 저장소 폴더와 소스트리의 저장소를 연결하여 사용하면 됨

▼ 그림 2-34 깃과 소스트리가 연결된 화면

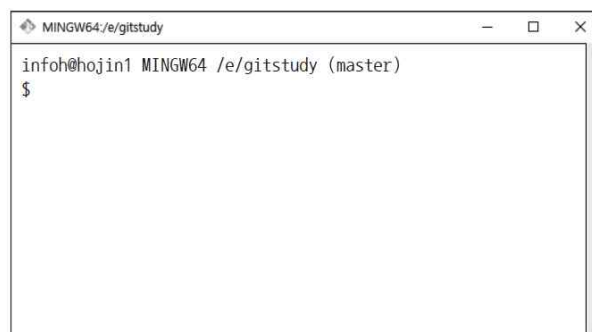


3. 첫 번째 깃 실행

➤ 소스트리로 실행

- 이 버튼을 클릭하면 소스트리에서 바로 깃 배시 터미널을 실행할 수 있음
- 깃 배시가 실행될때 현재 작업하는 저장소를 기준으로 터미널을 오픈함
- 명령어를 입력하여 경로를 이동할 필요가 없어 편리함

▼ 그림 2-35 소스트리의 터미널



2.4 환경 설정

75

4. 환경 설정

➤ 환경 설정

- 깃을 설치했다면 여러 가지 환경 설정도 해야 함
- 사용자 이름과 이메일 주소는 필수 항목이고, 그 외 나머지 항목은 옵션이므로 추가 항목은 가볍게 살펴보자

4. 환경 설정

➤ config 명령어

- 깃은 환경 설정을 위해 별도로 config 명령어를 제공함
- config 명령어는 환경 설정 파일을 직접 수정하지 않고도 환경 설정을 쉽게 할 수 있게 도와줌

```
$ git config 설정값
```

- 다음 명령어처럼 기존에 설정된 환경 파일을 삭제할 수도 있음
- --unset 옵션을 사용함

```
$ git config --unset 이메일주소
```

4. 환경 설정

➤ 로컬 사용자

- 깃은 여러 사람과 함께 개발할 수 있는 협업 도구임
- 프로젝트 하나를 다수의 개발자와 함께 작업할 때를 대비하여 각 개발자를 구분해야 함
- 깃은 각 개발자의 작업을 구분하려고 사용자를 등록하는 과정을 거침
- 로컬 저장소에서 사용자 등록은 별도의 웹 사이트에서 회원 가입을 하는 것이 아니라, 소스 코드의 변경 내역을 기록할 때 구분할 수 있는 사용자 설정 값만 등록하면 됨
- 사용자 등록은 최초로 깃을 사용하거나 커밋할 때 한 번만 함

4. 환경 설정

➤ 로컬 사용자

- 사용자 등록은 크게 두 가지로 구분함
- 선택한 로컬(local) 저장소에만 적용되는 로컬 사용자 설정 값과 모든 로컬 저장소에 공통으로 적용되는 글로벌 사용자 설정 값임
- 깃에서는 사용자를 구분하려고 '사용자 이름'과 '이메일 주소'를 사용하며, config 명령어로 환경 설정 파일에 등록함

4. 환경 설정

➤ 로컬 사용자

- 로컬 저장소에서는 다음 형태로 사용자를 등록함
- 사용자 이름과 이메일 주소는 한글로 입력하면 오류가 발생하므로 영문으로 작성해야 함

```
$ cd 저장소 폴더 ----- 깃 저장소 폴더
$ git config user.name "사용자이름"
$ git config user.email "이메일주소"
```

- 깃에서 사용자를 구분하는 데 쓰는 '사용자 이름'과 '이메일 주소' 중 이메일 주소는 깃이 개발자를 구별하는 고유의 키 값으로 사용함
- 자신의 저장소를 외부로 공개하면 등록된 이메일 주소도 외부에 공개되므로, 공개해도 무관한 이메일 주소를 사용하길 권장함

4. 환경 설정

➤ 글로벌 사용자(추천)

- 로컬 사용자 등록은 로컬 저장소를 생성할 때마다 설정해야 함
- 저장소마다 다르게 설정할 수 있음
- 저장소를 생성할 때마다 사용자 등록을 하는 것은 불편함
- 혼자서 사용하는 컴퓨터라면 글로벌(공통된) 사용자 등록을 하는 것이 편리함
- 글로벌 사용자 등록을 할 때는 다음과 같이 --global 옵션을 함께 사용함
- 설정된 모든 값은 글로벌(global)(전역) 영역에 설정됨

```
$ git config --global user.name "사용자이름"
$ git config --global user.email "이메일주소"
```

4. 환경 설정

➤ 글로벌 사용자(추천)

- 이 값들은 나중에 소스트리와 연동할 때도 필요하므로 잘 기록해두자

```
$ git config --global user.name "hojinlee" ----- 자신의 영문 이름(아무것이든 원하는 이름으로 입력)
$ git config --global user.email "infohojin@naver.com" ----- 자신의 이메일 주소(공개해도 괜찮은 주로 사용하는 이메일 주소 입력)
```

4. 환경 설정

➤ 환경 설정 파일 확인 및 직접 수정

- config 명령어로 만든 환경 설정 파일은 깃 저장소 안에 **.git/config** 파일 형태로 저장되어 있음
- 이 환경 설정 파일을 찾아 config 명령어로 등록한 내용이 제대로 적용되었는지 확인해보자

4. 환경 설정

➤ 환경 설정 파일 확인 및 직접 수정

- 환경 설정 파일이 어디에 있는지 찾기 위해서는 먼저 로컬 저장소를 생성함
- 해당 저장소로 이동하여 (로컬) 사용자 등록을 한 후 설정 파일을 찾음

예

```
$ mkdir gitstudy02 ----- gitstudy02 폴더 만들기
$ cd gitstudy02 ----- 만든 gitstudy02 폴더로 이동

infoh@hojin1 MINGW64 /e/gitstudy02
$ git init ----- 깃 초기화
Initialized empty Git repository in E:/gitstudy02/.git/

infoh@hojin1 MINGW64 /e/gitstudy02 (master)
$ git config user.name "hojinlee" ----- 로컬 사용자 이름 입력
$ git config user.email "infohojin@naver.com" ----- 로컬 이메일 주소 입력

infoh@hojin1 MINGW64 /e/gitstudy02 (master)
$ ls .git ----- 깃 목록 보기
config description FETCH_HEAD HEAD hooks info objects refs
```

4. 환경 설정

➤ 환경 설정 파일 확인 및 직접 수정

- 해당 위치로 이동하여 간단히 다음 명령어를 실행하면 환경 설정 파일이 있는지 확인할 수 있음

```
$ ls .git
```

- .git/config 파일은 로컬 저장소에서 직접 로컬 사용자 등록을 할 때만 찾을 수 있음
- 글로벌 사용자 등록을 했다면 저장소에 .git/config 파일은 존재하지 않음
- 그 대신 개인 계정 루트에 파일을 생성하므로 다음 명령어를 실행함

```
$ ls ~/.gitconfig ----- .gitconfig 폴더의 경로 보기
```

4. 환경 설정

➤ 환경 설정 파일 확인 및 직접 수정

- 컴퓨터에 글로벌 환경 설정 파일과 로컬 환경 설정 파일이 모두 있을 때도 있음
- 이때는 로컬 환경 설정 파일의 내용을 우선 적용함
- 환경 설정 파일을 찾았다면 간단히 cat 명령어로 내용을 확인할 수 있음

예

```
$ cat .git/config
```

```
[core]
```

```
    repositoryformatversion = 0
    filemode = false
    bare = false
    logallrefupdates = true
    symlinks = false
    ignorecase = true
```

```
[user]
```

```
    name = hojinlee
    email = infohojin@naver.com
```

4. 환경 설정

➤ 환경 설정 파일 확인 및 직접 수정

- 환경 설정 파일을 직접 열어 수정하고 싶다면 다음 명령어를 실행함
- VS Code 편집기를 사용함

예

\$ `code .git/config`

- 환경 설정 파일을 직접 수정하면 철자 오류 등 여러 가지 이유 때문에 정상적으로 동작하지 않을 수도 있으니, 가능하면 config 명령어를 사용하길 권장함

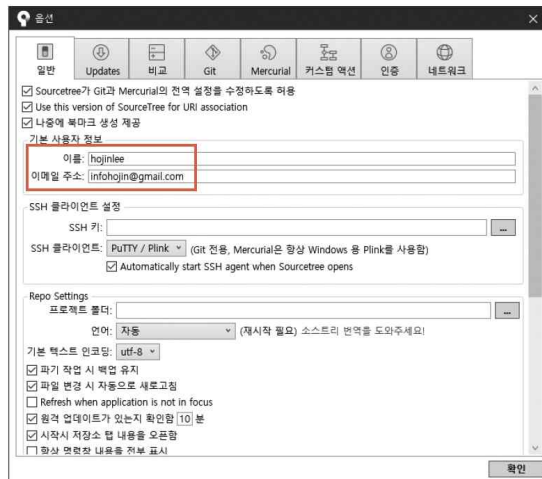
4. 환경 설정

➤ 소스트리의 환경 설정

- 소스트리에서도 환경 설정을 할 수 있음
- GUI 환경이라 좀 더 편리함
- 소스트리를 실행한 후 **도구 > 옵션** 메뉴를 선택하면 다음과 같이 환경 설정 화면이 나옴
- **이름**과 **이메일 주소** 항목은 필수임
- 일반적으로 소스트리를 설치할 때 입력했던 사용자 이름과 이메일 주소가 자동으로 입력되어 있음

4. 환경 설정

▼ 그림 2-36 소스트리 환경 설정



4. 환경 설정

➤ 별칭

- 좀 더 고급 환경 설정으로 별칭을 사용할 수 있음
- 별칭은 복잡한 git 명령어를 단순하게 닉네임 형태로 등록해 두는 기능임
- 예를 들어 다음과 같이 `log --graph --pretty=oneline` 명령어를 `show-graph`라는 별칭으로 등록해 두면 이후로는 이 별칭을 사용할 수 있음

예

```
$ git config -global alias.show-graph 'log --graph --pretty=oneline'
```

2.5 비주얼 스튜디오 코드

91

5. 비주얼 스튜디오 코드

➤ 비주얼 스튜디오 코드

- 코드를 작성하려면 코드 편집기가 필요함
- 책에서는 코드 편집용 툴로 비주얼 스튜디오 코드(VS Code, Visual Studio Code)를 사용함
- VS Code는 무료로 내려받아 설치할 수 있음
- 자신에게 익숙한 다른 편집기를 사용해도 됨
- VS Code는 다음과 같이 터미널에서 명령어로 쉽게 실행할 수 있음

```
$ code 파일이름
```

2.6 정리

93

6. 정리

➤ 정리

- 모든 학습이 그렇듯이 처음에 프로그램을 설치하고 환경 설정을 하는 데 많은 시간이 소요됨
- 책 내용을 천천히 따라가면서 자신만의 개발 환경을 설정하자