



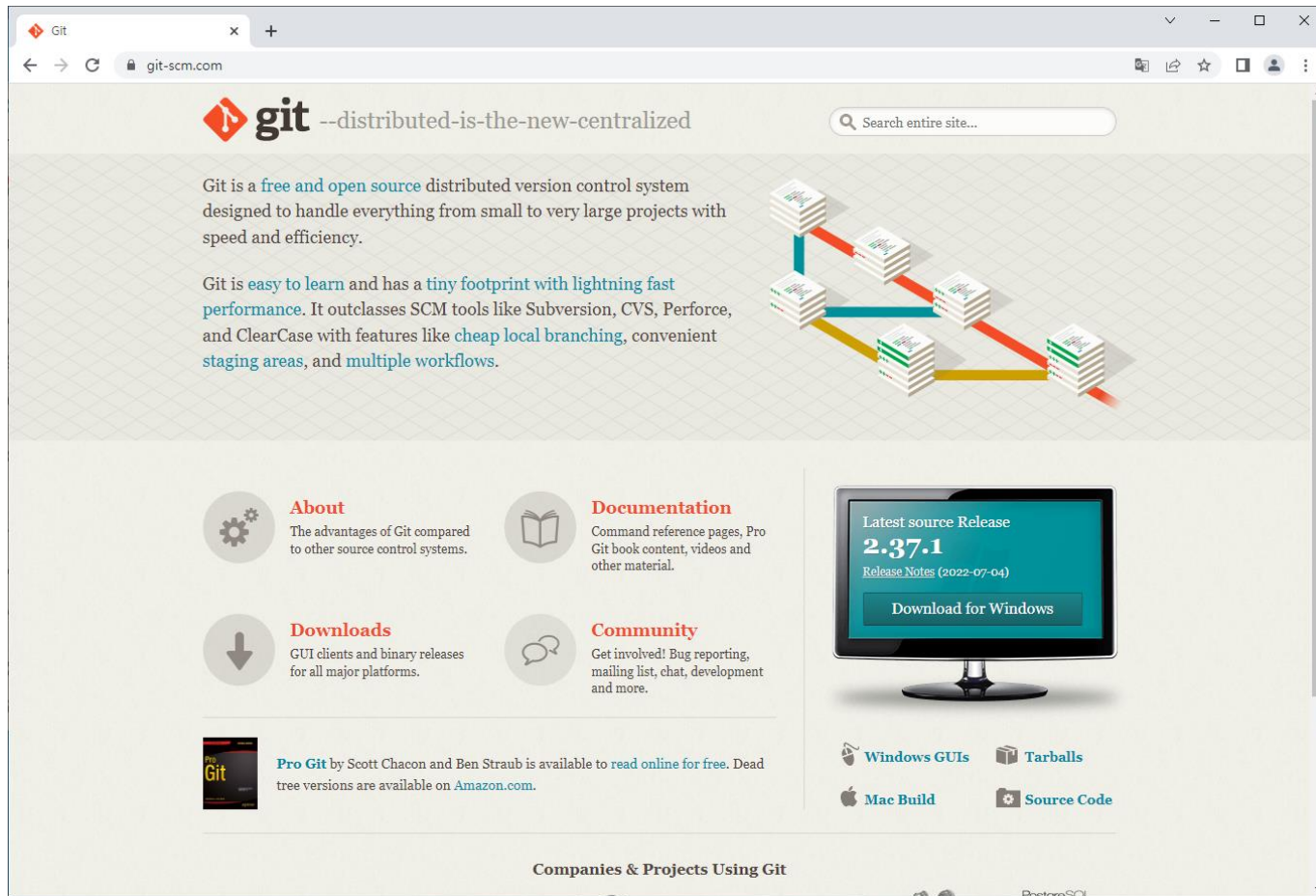
git

Git

- 2005년 리누스 토르발스(Linus Torvalds)가 개발
- 버전 관리(Version Control)
 - ✓ 어떤 파일이 변경되었는가?
 - ✓ 언제 변경되었는가?
- 백업(Backup)
 - ✓ 드롭 박스나 구글 드라이브와 같은 온라인 저장소를 제공
- 협업(Collaboration)
 - ✓ 여러 사람이 github에 저장된 파일을 가져와서 수정하거나 재 업로드 가능
 - ✓ 각자 자신의 브랜치에서만 작업하기 때문에 자신의 작업이 다른 사람의 작업에 영향을 미치지 않음

Git 다운로드

- 다운로드 링크 : <https://git-scm.com/>



Git 설치

Git 2.37.1 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
License is intended to guarantee your freedom to share and change

<https://gitforwindows.org/>

NextCancel

Git 2.37.1 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\GitBrowse...

At least 265.9 MB of free disk space is required.

<https://gitforwindows.org/>

BackNextCancel

Git 2.37.1 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.

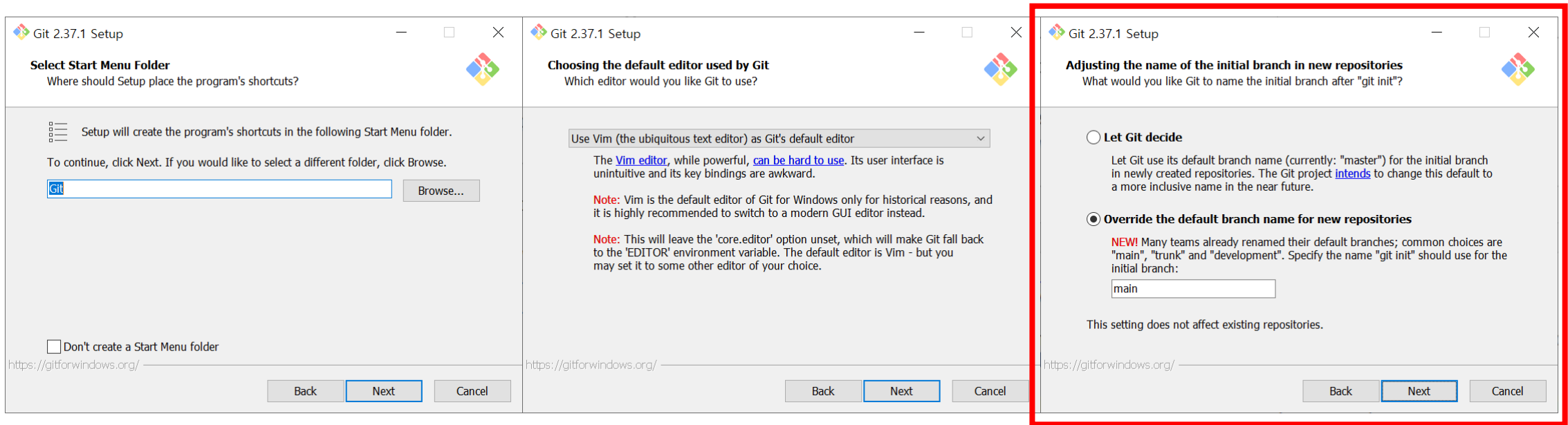
☐ Additional icons
☐ On the Desktop
☒ Windows Explorer integration
☒ Git Bash Here
☒ Git GUI Here
☒ Git LFS (Large File Support)
☒ Associate .git* configuration files with the default text editor
☒ Associate .sh files to be run with Bash
☐ Check daily for Git for Windows updates
☐ (NEW!) Add a Git Bash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

Current selection requires at least 265.9 MB of disk space.

<https://gitforwindows.org/>

BackNextCancel

Git 설치



기본 편집기는 Vim
Notepad++, VSCode 등으로
수정 가능

default branch를 main으로 지정

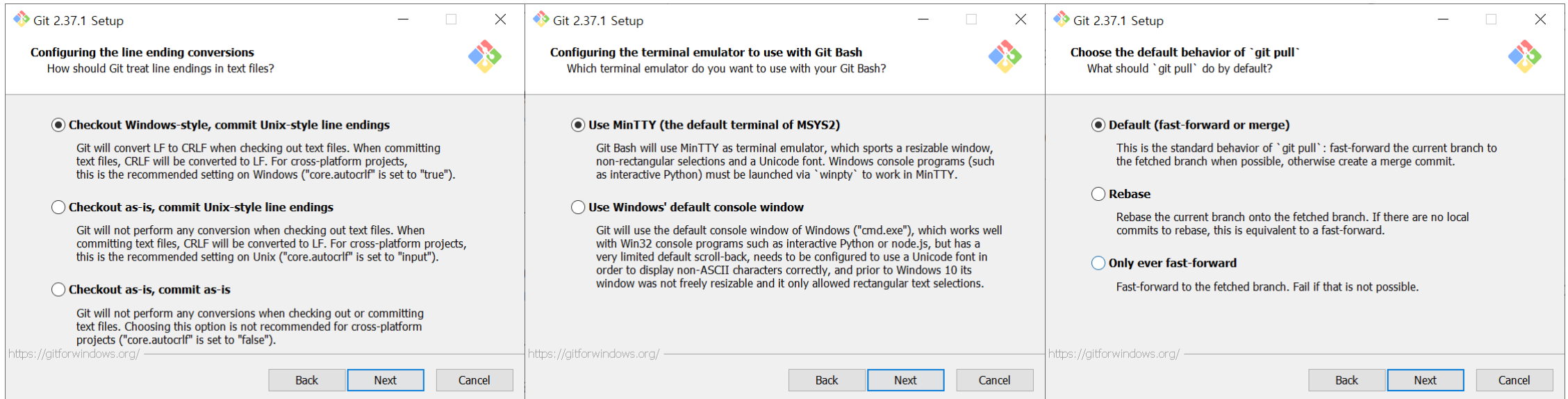
Git 설치

Git 2.37.1 Setup	Git 2.37.1 Setup	Git 2.37.1 Setup
Adjusting your PATH environment How would you like to use Git from the command line?	Choosing the SSH executable Which Secure Shell client program would you like Git to use?	Choosing HTTPS transport backend Which SSL/TLS library would you like Git to use for HTTPS connections?
<p><input type="radio"/> 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.</p> <p><input checked="" type="radio"/> 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.</p> <p><input type="radio"/> 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.</p> <p>https://gitforwindows.org/</p> <p>Back Next Cancel</p>	<p><input checked="" type="radio"/> Use bundled OpenSSH This uses ssh.exe that comes with Git.</p> <p><input type="radio"/> Use (Tortoise)Plink To use PuTTY, specify the path to an existing copy of (Tortoise)Plink.exe: <input type="text"/> ... <input type="checkbox"/> Set ssh.variant for Tortoise Plink</p> <p><input type="radio"/> 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.</p> <p>https://gitforwindows.org/</p> <p>Back Next Cancel</p>	<p><input checked="" type="radio"/> Use the OpenSSL library Server certificates will be validated using the ca-bundle.crt file.</p> <p><input type="radio"/> 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.</p> <p>https://gitforwindows.org/</p> <p>Back Next Cancel</p>

보안 프로그램 OpenSSH

HTTPS 접속 OpenSSL

Git 설치



텍스트 줄 바꿈 처리 방식

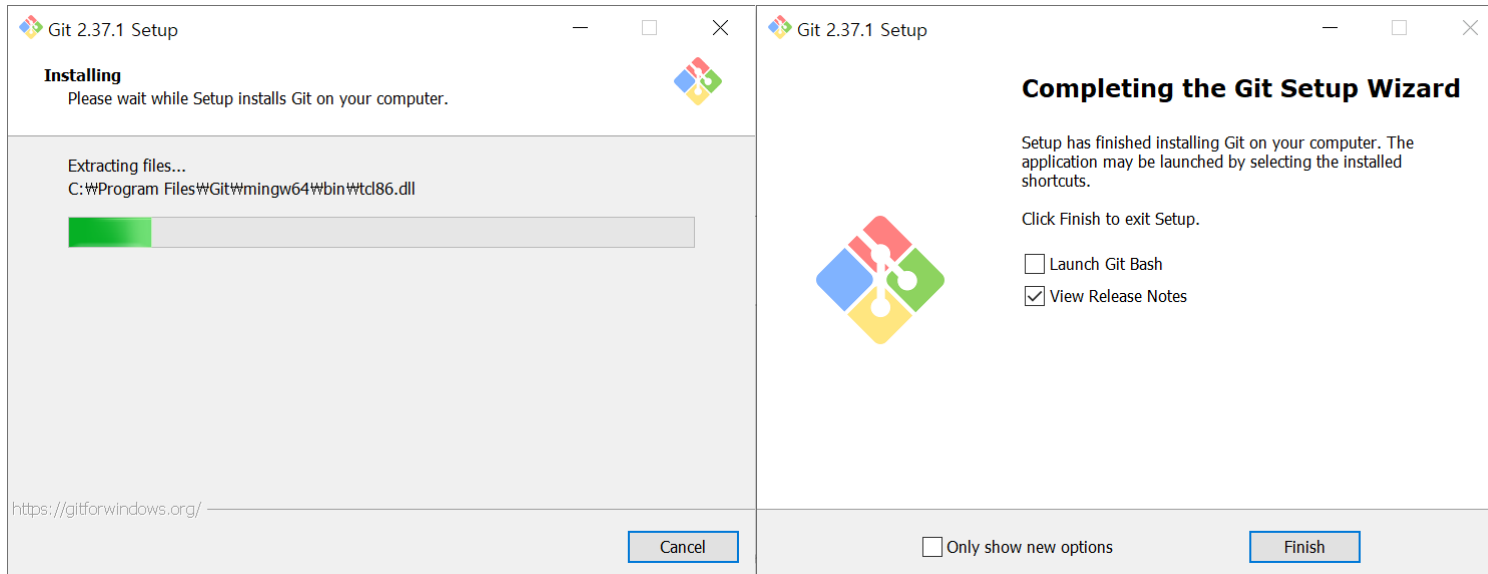
터미널 에뮬레이터 MinTTY

git pull 명령의 기본 동작

Git 설치

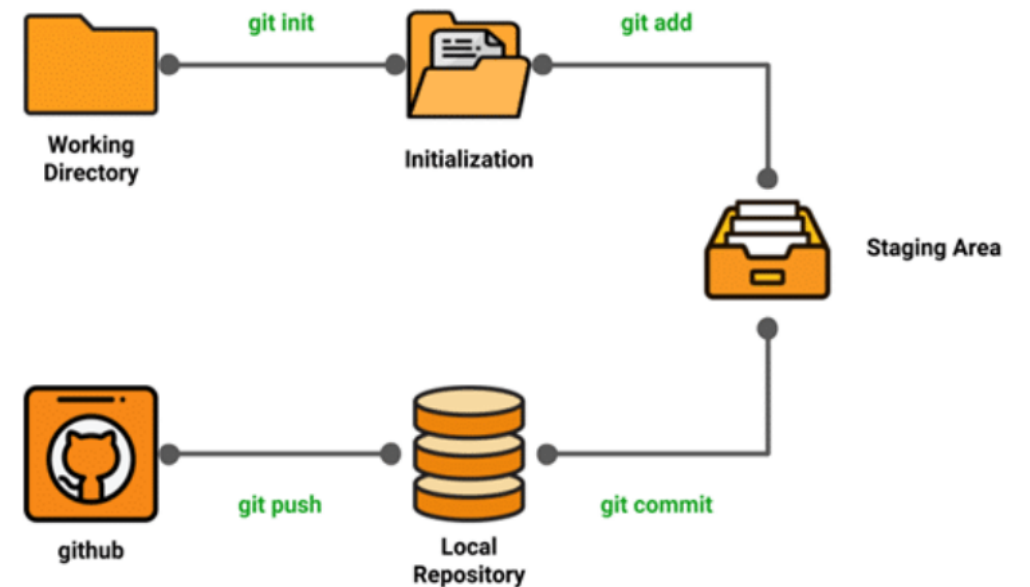
Git 2.37.1 Setup	Git 2.37.1 Setup	Git 2.37.1 Setup
Choose a credential helper Which credential helper should be configured?	Configuring extra options Which features would you like to enable?	Configuring experimental options These features are developed actively. Would you like to try them?
<p><input checked="" type="radio"/> Git Credential Manager Use the cross-platform Git Credential Manager. See more information about the future of Git Credential Manager here.</p> <p><input type="radio"/> None Do not use a credential helper.</p>	<p><input checked="" type="checkbox"/> 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.</p> <p><input type="checkbox"/> Enable symbolic links Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.</p>	<p><input type="checkbox"/> 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.</p> <p><input type="checkbox"/> 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.</p>
<p>https://gitforwindows.org/</p> <p>Back Next Cancel</p>	<p>https://gitforwindows.org/</p> <p>Back Next Cancel</p>	<p>https://gitforwindows.org/</p> <p>Back Install Cancel</p>

Git 설치



Repository

- 레파지토리(Repository) = 저장소
- 레파지토리(Repository)는 파일이나 디렉터리의 변경 내역(commit)을 관리하는 장소
- 로컬 저장소(Local Repository)와 원격 저장소(Remote Repository)로 구분
- 로컬 저장소(Local Repository)
 - ✓ 개인 PC의 작업 디렉터리에 있는 저장소
 - ✓ 사용자 개인의 변경 이력을 관리
 - ✓ 작업 디렉터리의 .git이라는 숨겨진 디렉터리가 저장소의 실체
 - ✓ 작업 디렉터리에서 git init 명령으로 생성
- 원격 저장소(Remote Repository)
 - ✓ 서버에 있는 저장소
 - ✓ 여러 사용자가 변경 내용을 공유하는 저장소
 - ✓ github.com에서 생성



Git Directory / Working Tree / Staging Area

- Git Directory

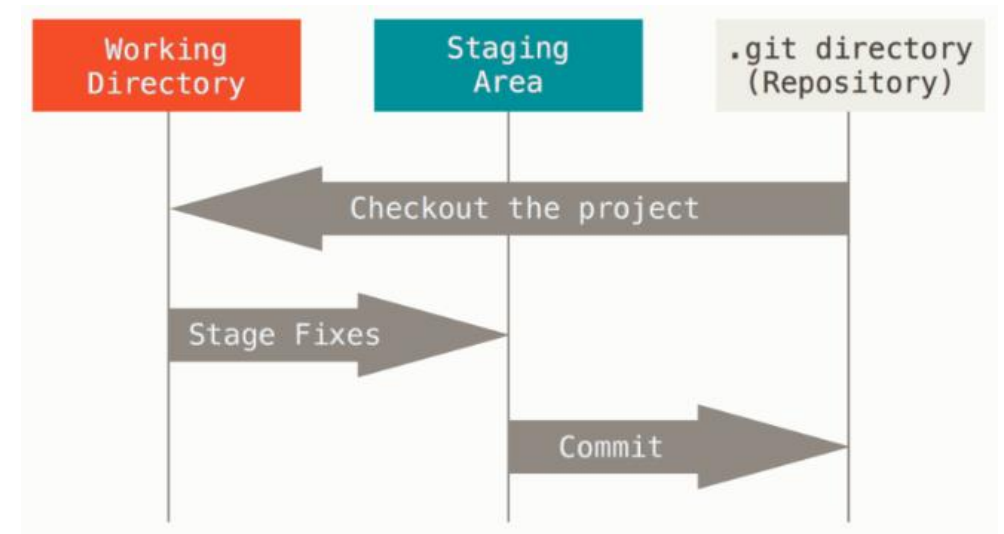
- ✓ Git이 프로젝트의 메타데이터와 객체 데이터베이스를 저장하는 장소
- ✓ 다른 저장소를 복제(clone) 할 때 Git Directory가 생성

- Working Tree

- ✓ 프로젝트의 특정 버전을 체크아웃(Checkout) 한 것
- ✓ Git Directory 안에 압축된 데이터베이스에서 파일을 가져와서 생성

- Staging Area

- ✓ Git Directory 내에 존재
- ✓ 단순 파일로 곧 커밋(commit)할 파일에 대한 정보를 저장

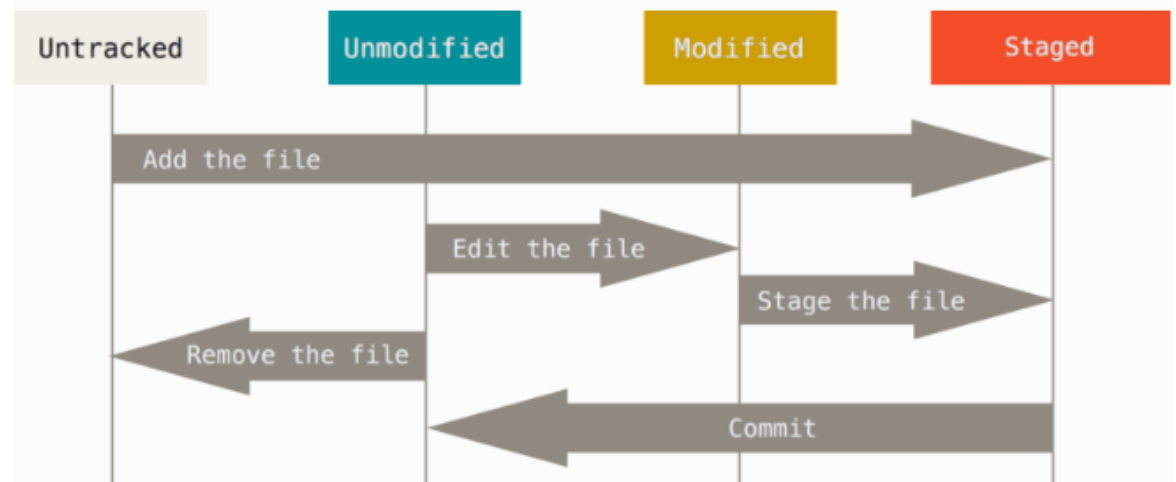


- Git의 기본 동작

- ✓ Working Tree에서 파일을 수정
- ✓ Staging Area에 파일을 Stage해서 커밋할 스냅샷을 생성(모든 파일을 추가할 수도 있고 선택해서 추가할 수도 있음)
- ✓ Staging Area에 있는 파일들을 커밋(commit)해서 Git Directory에 영구적인 스냅샷으로 저장
- ✓ 즉, Git Directory에 있는 파일들은 모두 Committed 상태

Git staging / commit

- Git 저장소의 커밋 과정
 1. Working Tree에서 파일 변경 작업
 2. 이 중 커밋할 내용을 stage해서 Index에 추가
 3. Index의 내용이 커밋
- staging
 - ✓ git이 관리하는 대상(Tracked) 중에서 커밋할 대상을 stage 영역에 넣음
 - ✓ git add 명령으로 파일을 지정하고 Index를 생성
- commit
 - ✓ 변경 이력을 로컬 저장소에 등록하는 것
 - ✓ git commit 명령으로 수행
 - ✓ 반드시 커밋메시지를 작성해야 함

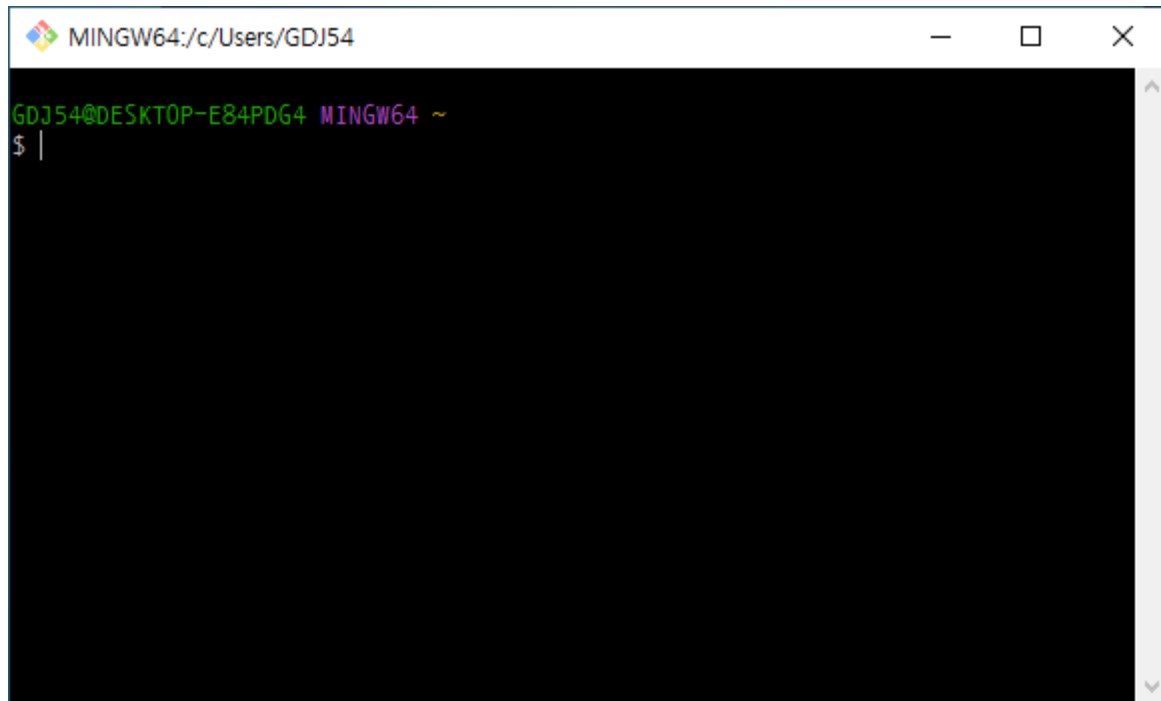


Git 최초 설정

- /etc/gitconfig
 - ✓ 시스템의 모든 사용자와 모든 저장소에 적용되는 설정 파일
 - ✓ git config --system 옵션으로 읽기 및 쓰기 가능
- ~/.gitconfig, ~/.config/git/config
 - ✓ 특정 사용자(즉 현재 사용자)에게 적용되는 설정 파일
 - ✓ git config --global 옵션으로 읽기 및 쓰기 가능
- .git/config
 - ✓ 특정 저장소나 현재 작업 중인 프로젝트에만 적용되는 설정 파일
- Git은 커밋할때마다 사용자 이름과 이메일 정보를 사용하기 때문에 이 정보를 등록해 두어야 함
 - ✓ 사용자 이름 등록 : git config --global user.name 사용자명
 - ✓ 사용자 메일 등록 : git config --global user.email 이메일
 - ✓ --global 옵션을 이용하면 딱 한 번만 설정하면 됨
 - ✓ 프로젝트마다 다른 이름과 이메일 정보를 사용하려면 --global 옵션을 빼면 됨

Git Bash

- Git Bash
 - ✓ 이름 그대로 Bash를 사용할 때 사용
 - ✓ Bash란 Linux에서 사용되는 셸(Shell)을 의미
 - ✓ 당연히 Git Bash에서는 Linux 명령어를 사용



```
MINGW64: c:/Users/GDJ54
GDJ54@DESKTOP-E84PDG4 MINGW64 ~
$ |
```

Directory

- 홈 디렉터리(Home Directory)
 - ✓ 컴퓨터에 로그인 한 사용자의 디렉터리
 - ✓ [Windows] C:\Users\사용자명
 - ✓ [Mac] /Users/사용자명
 - ✓ ~(물결표, Tilt)가 홈 디렉터를 의미함
- 상위 디렉터리
 - ✓ ..(마침표 2개)가 상위 디렉터를 의미함
- 현재 디렉터리
 - ✓ .(마침표 1개)가 현재 디렉터를 의미함

Linux 명령

- 디렉터리 만들기
\$ **mkdir testdir**
- 현재 디렉터리에 있는 testdir 디렉터리로 이동
\$ **cd testdir**
- C: 드라이브 아래 testdir 디렉터리로 이동
\$ **cd c:/testdir**
- 상위 디렉터리로 이동
\$ **cd ..**
- 디렉터리 삭제
\$ **rmdir testdir**

Linux 명령

- 현재 작업 중인 디렉터리 확인
\$ **pwd**
- 현재 디렉터리에 있는 파일 및 디렉터리 목록 확인
\$ **ls**
\$ **ls -a**
- 파일 생성
\$ **touch test.txt**
- 파일에 데이터 추가
\$ **echo '텍스트' >> test.txt**

Linux 명령

- 파일의 내용 표시
\$ **cat test.txt**
- 파일의 마지막 1줄 출력
\$ **tail -n 1 test.txt**
- 파일 삭제
\$ **rm test.txt**
- 비어 있는 디렉터리 삭제
\$ **rm -r testdir**
- 디렉터리에 파일이 있어도 강제로 디렉터리 삭제
\$ **rm -rf testdir**

Git 명령

- 로컬 저장소 생성
\$ **git init**
- 특정 파일 스테이징(Staging)
\$ **git add test.txt**
- 전체 파일 스테이징(Staging)
\$ **git add .**
- 커밋
\$ **git commit**
\$ **git commit -m '커밋메시지'**

Git 명령

- 스테이징 + 커밋 (최초 커밋에서는 git add + git commit 분리 사용해야 함)
\$ **git commit -am '커밋메시지'**
- git add 취소(커밋 이력이 없는 경우)
\$ **git rm --cached test.txt**
- git add 취소(커밋 이력이 있는 경우)
\$ **git restore --staged test.txt**
- git commit 취소
\$ **git reset --hard 이전커밋아이디**

Git 명령

- git 상태 확인
\$ **git status**
- git 로그 확인
\$ **git log**
\$ **git log --oneline**
- git 현재 상태 확인
\$ **git show**
- git 차이 확인
\$ **git diff --word-diff 커밋아이디1 커밋아이디2**

Git 명령

- develop 브랜치 만들기
\$ **git branch develop**
- develop 브랜치로 전환
\$ **git checkout develop**
- develop 브랜치 만들기 + 전환하기
\$ **git checkout -b develop**
- main 브랜치에 develop 브랜치 병합하기(main 브랜치로 이동한 뒤 작업해야 함)
\$ **git checkout main**
\$ **git merge develop**
- develop 브랜치 삭제
\$ **git branch -d develop**

Git 명령

- 원격 저장소 등록(origin)
\$ **git remote add origin https://github.com/home/GDJ.git**
- 원격 저장소 삭제(origin)
\$ **git remote remove origin**
- 원격 저장소 주소 변경(origin)
\$ **git remote set-url origin https://github.com/edu/GDJ.git**
- 변경된 주소와 동기화(origin)
\$ **git remote update origin --prune**
- 원격 저장소 이름 바꾸기(origin → origin2)
\$ **git remote rename origin origin2**

Git 명령

- 원격 저장소 복제하기
\$ **git clone https://github.com/goodee/GDJ.git**
- 원격 저장소 → 로컬 저장소 내려 받기
\$ **git pull origin main**
- 원격 저장소 → 로컬 저장소 히스토리 없어도 내려 받기
\$ **git pull origin main --allow-unrelated-histories**
- 로컬 저장소 → 원격 저장소 올리기
\$ **git push origin main**
- 로컬 저장소 → 원격 저장소 강제로 올리기
\$ **git push -f origin main**

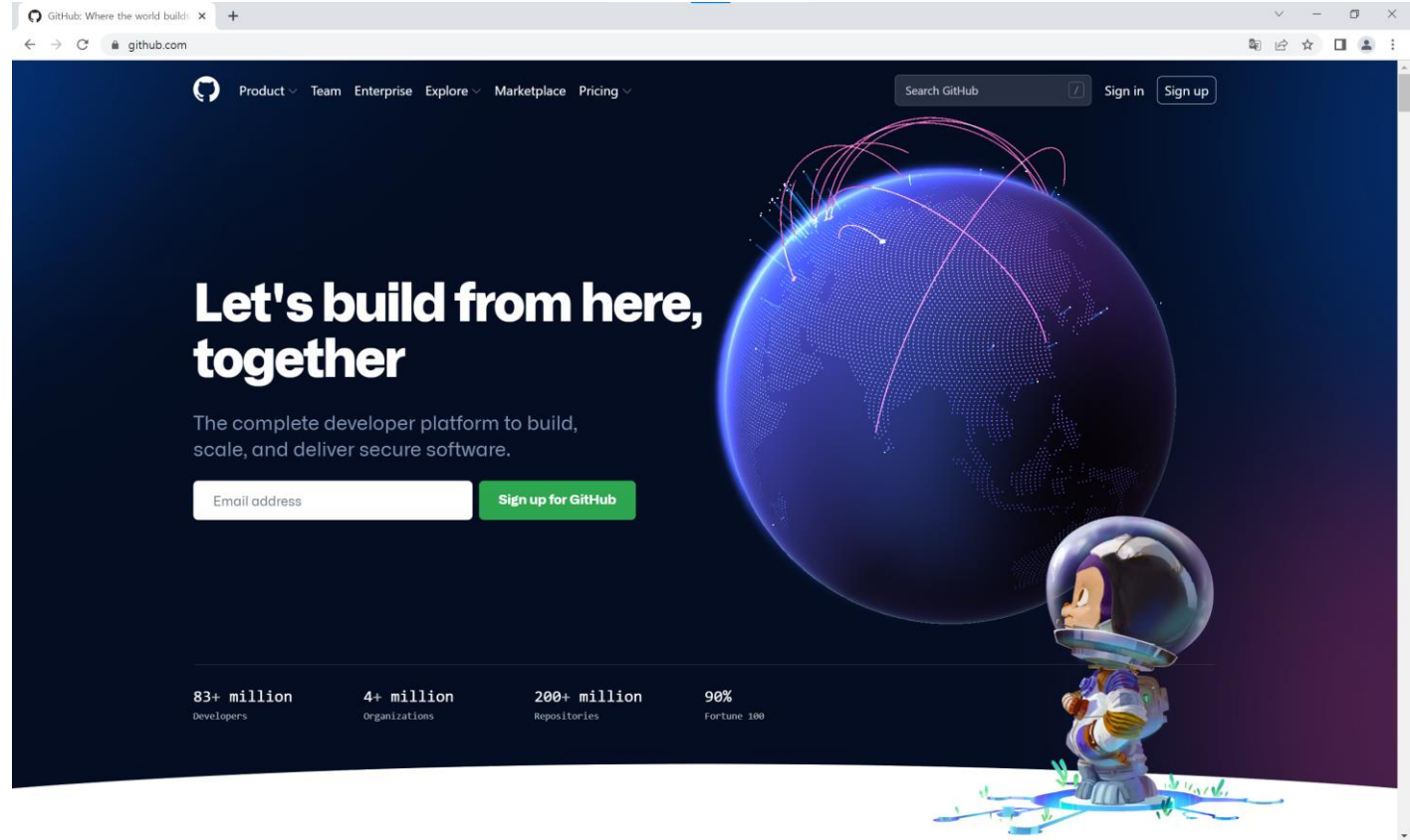


github

github

- 깃허브(github)

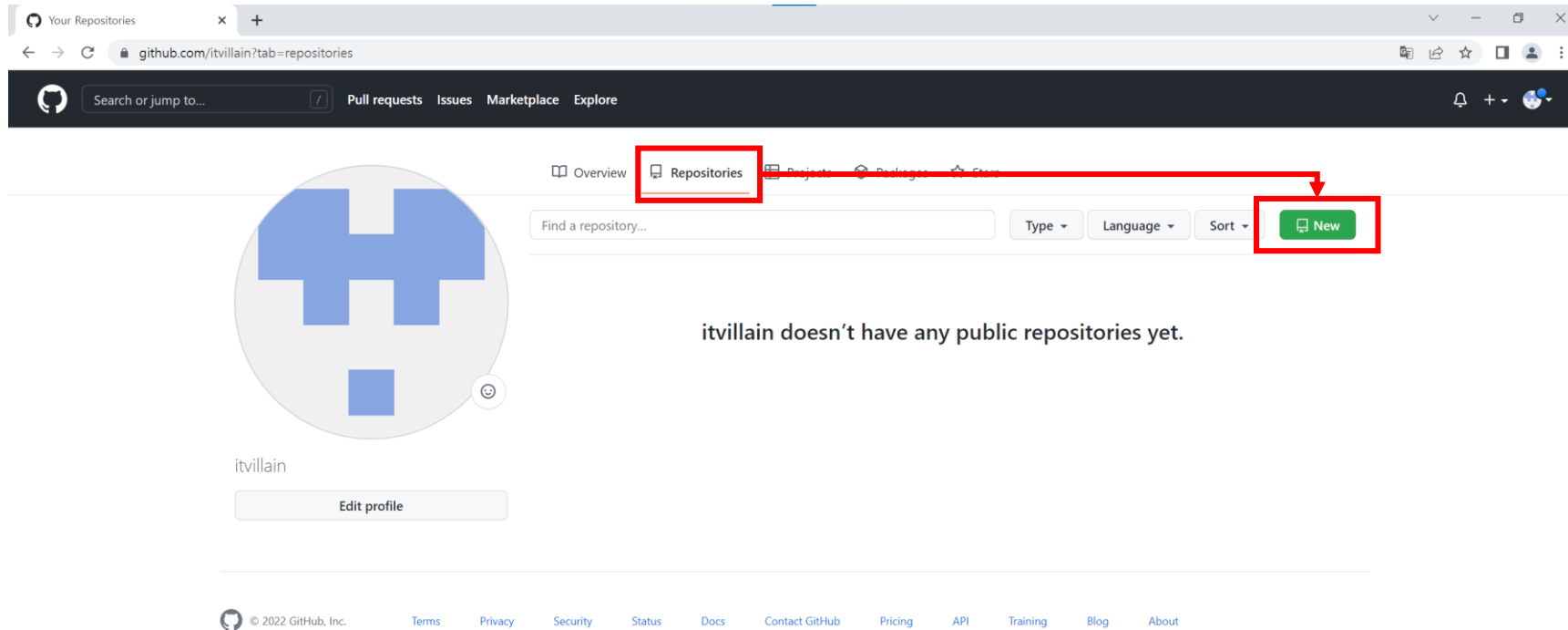
- ✓ <https://github.com/>
- ✓ Git 저장소 호스팅을 지원하는 웹 서비스
- ✓ 원격 저장소(Remote Repository)를 생성할 수 있음
- ✓ 회원 가입 후 사용 가능



github 메인 화면



원격 저장소 생성



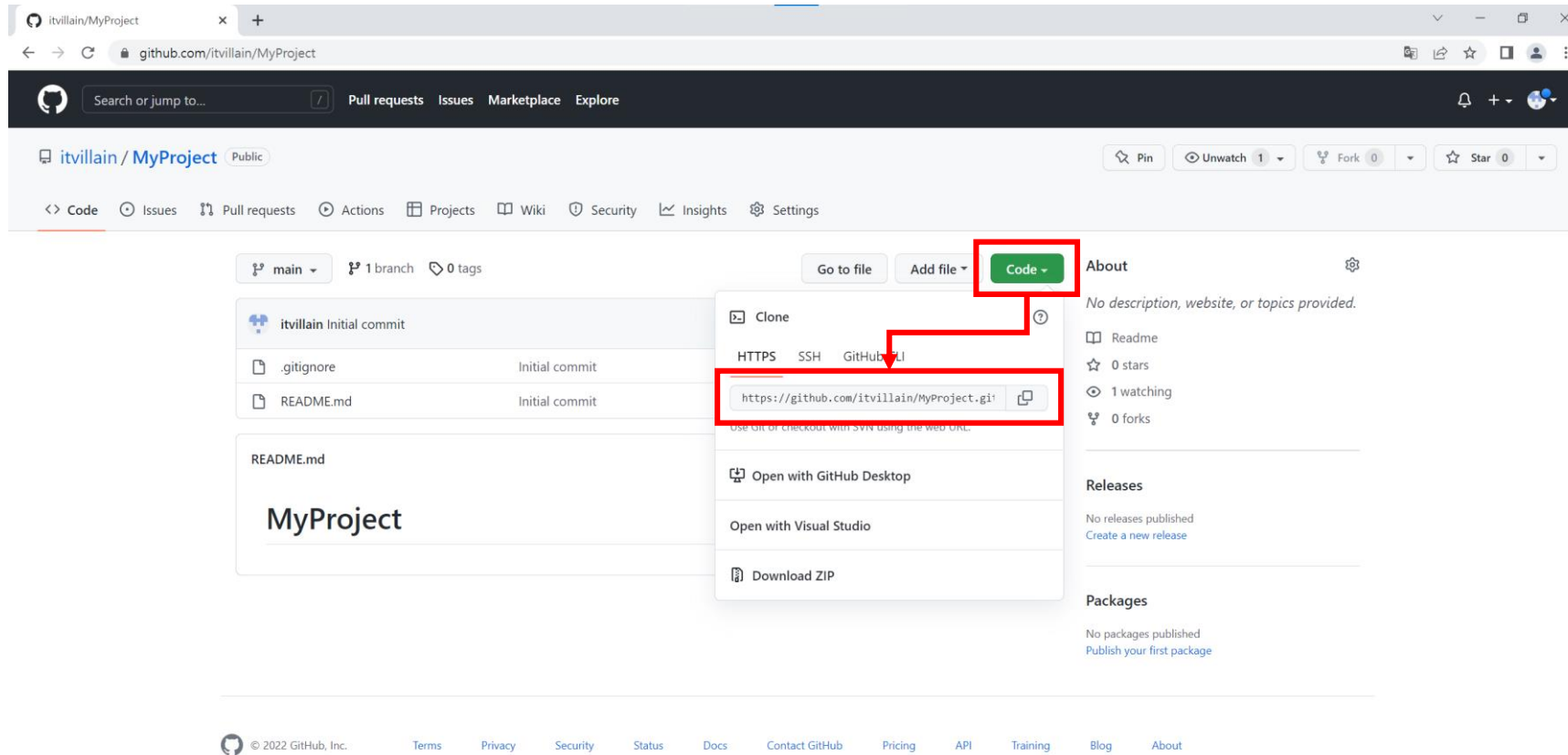
원격 저장소 생성

The screenshot shows the GitHub 'Create a new repository' page. Several elements are highlighted with red boxes:

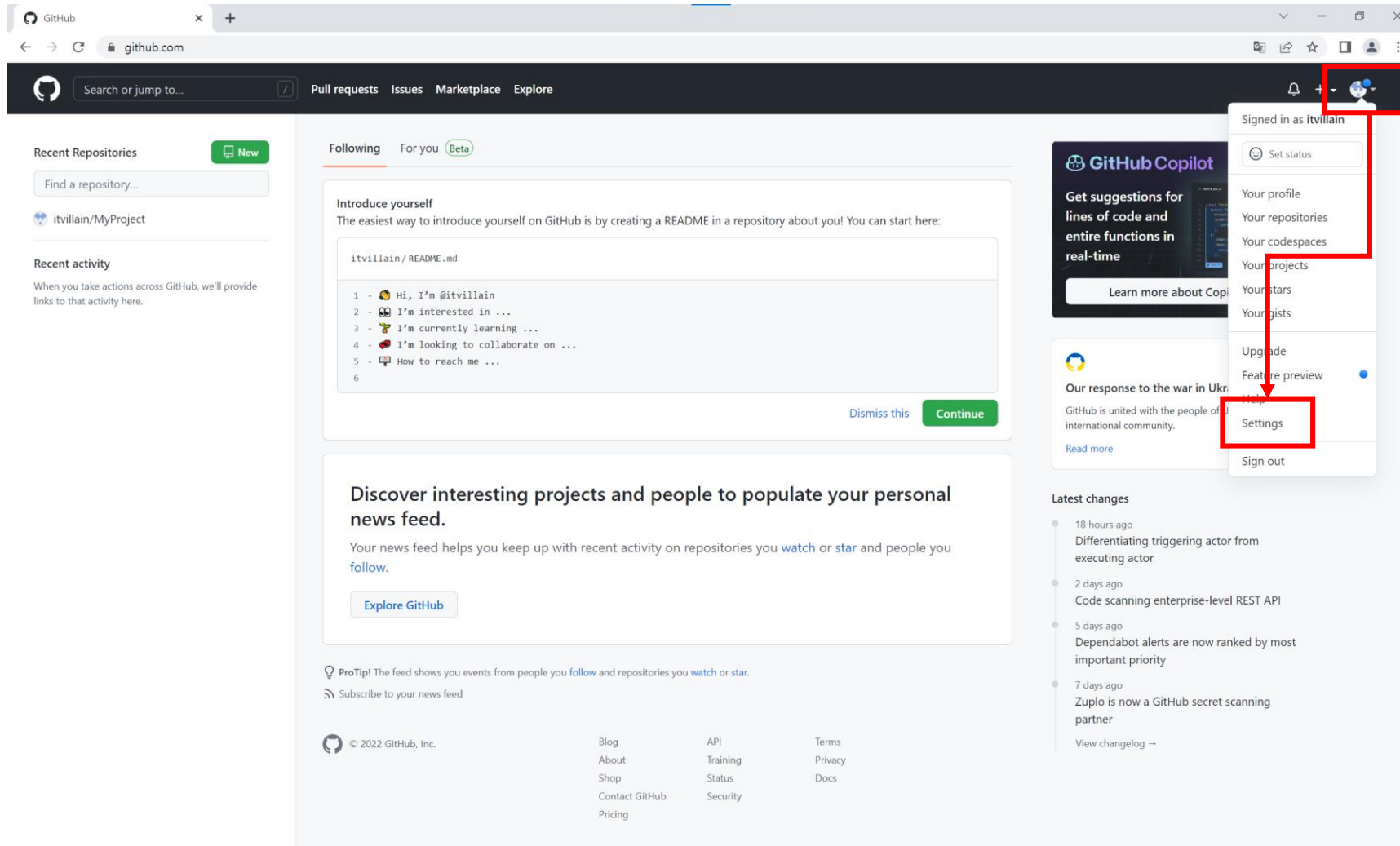
- Owner and Repository name:** A box around the 'Owner' dropdown (showing 'itvillain') and the 'Repository name' input field (containing 'MyProject' with a green checkmark). A red text overlay to the right says '로컬 저장소와 같은 이름 사용' (Use the same name as the local repository).
- Visibility:** A box around the 'Public' radio button, which is selected. A red text overlay to the right says '누구나 볼 수 있는 public' (Public, visible to everyone).
- Initialize this repository with:** A box around the 'Add a README file' checkbox, which is checked. A red text overlay to the right says 'README.md 파일 추가' (Add README.md file).
- Add .gitignore:** A box around the 'Add .gitignore' section, which includes a dropdown menu showing '.gitignore template: Java'. A red text overlay to the right says '.gitignore 파일 추가 후 내용 변경' (Change content after adding .gitignore file) and provides a URL: 'https://www.toptal.com/developers/gitignore/'.
- Create repository button:** A box around the green 'Create repository' button at the bottom.

Other visible text on the page includes: 'Create a new repository', 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.', 'Description (optional)', 'Great repository names are short and memorable. Need inspiration? How about **studious-disco**?', 'Public: Anyone on the internet can see this repository. You choose who can commit.', 'Private: You choose who can see and commit to this repository.', 'Initialize this repository with: Skip this step if you're importing an existing repository.', 'Add a README file: This is where you can write a long description for your project. Learn more.', 'Choose which files not to track from a list of templates. Learn more.', 'Choose a license: A license tells others what they can and can't do with your code. Learn more.', 'License: None', 'This will set **main** as the default branch. Change the default name in your settings.', and 'You are creating a public repository in your personal account.'

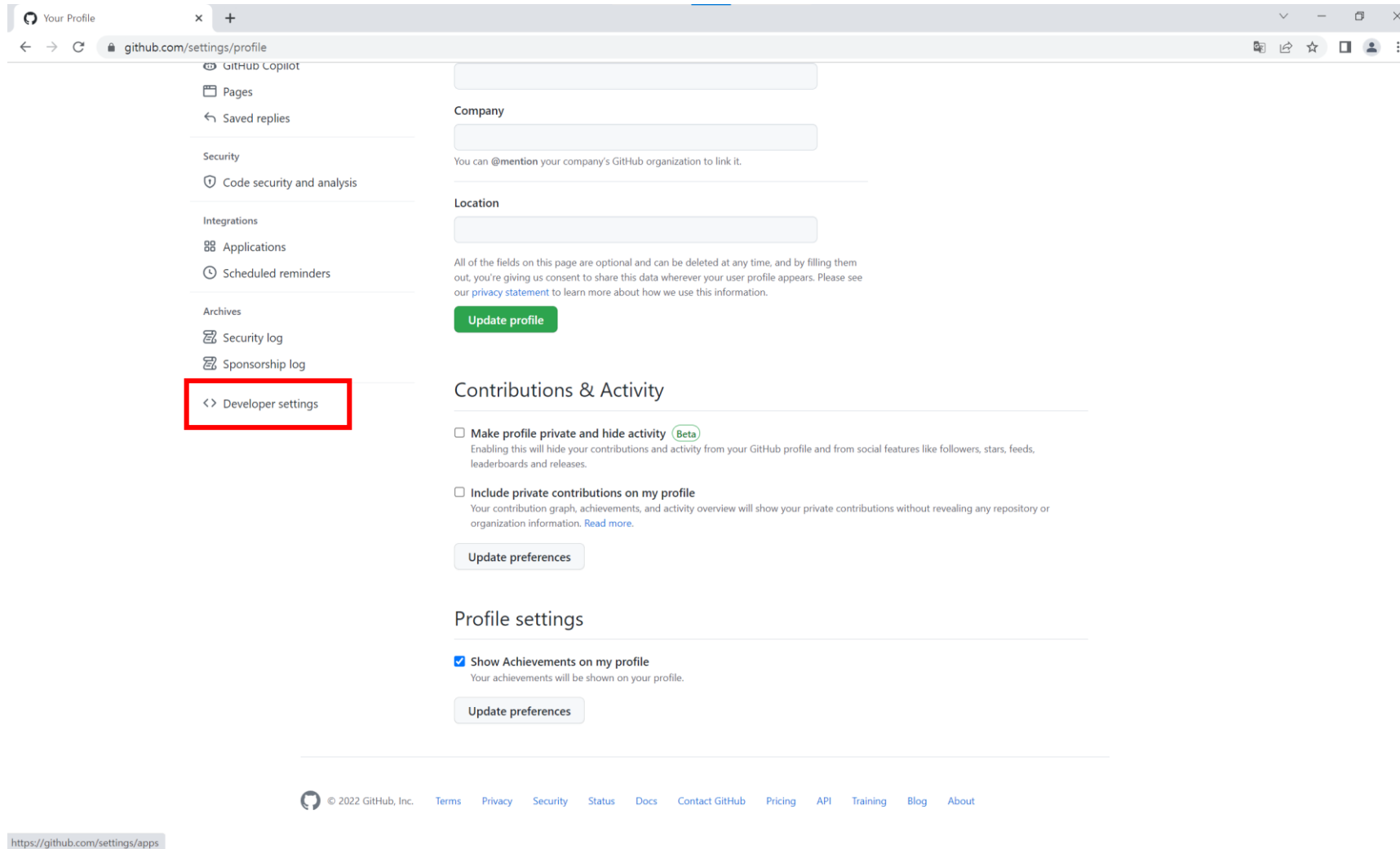
원격 저장소 주소 확인



Personal access token 생성



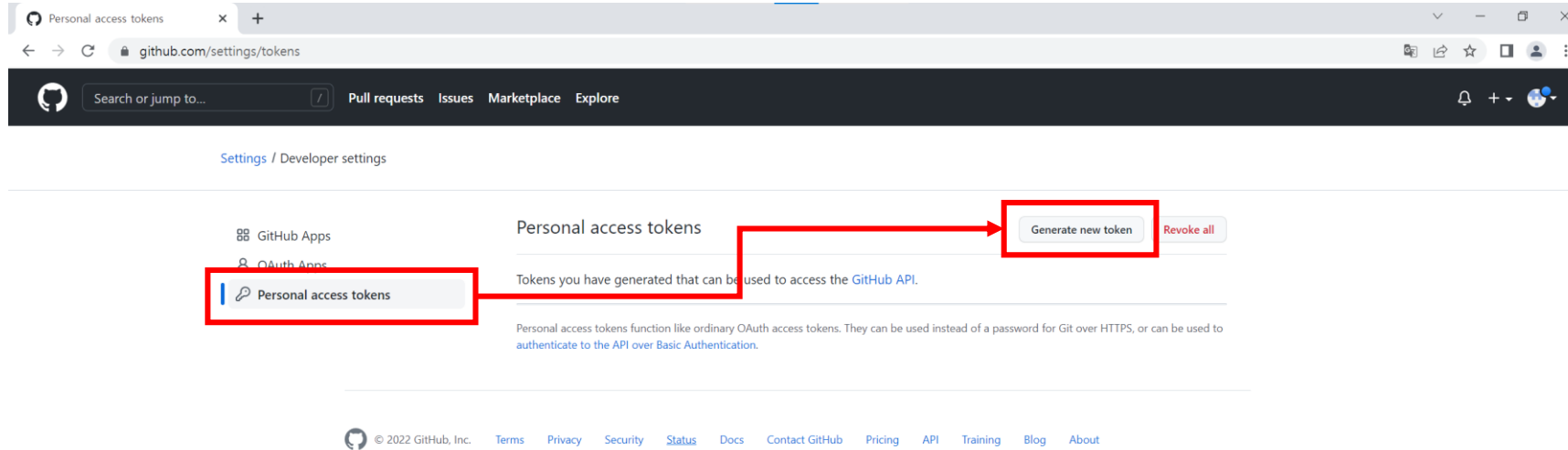
Personal access token 생성



The screenshot shows the GitHub 'Your Profile' settings page. The left sidebar contains a list of settings categories: GitHub Copilot, Pages, Saved replies, Security (with sub-items 'Code security and analysis'), Integrations (with sub-items 'Applications' and 'Scheduled reminders'), Archives (with sub-items 'Security log' and 'Sponsorship log'), and 'Developer settings' which is highlighted with a red rectangle. The main content area is divided into three sections: 1. Profile information fields for Name, Company, and Location, followed by a disclaimer and an 'Update profile' button. 2. 'Contributions & Activity' section with two toggle options: 'Make profile private and hide activity' (marked as Beta) and 'Include private contributions on my profile', each with a description and an 'Update preferences' button. 3. 'Profile settings' section with a checked option 'Show Achievements on my profile' and its description, also with an 'Update preferences' button. The footer includes the GitHub logo, copyright notice for 2022, and a list of links: Terms, Privacy, Security, Status, Docs, Contact GitHub, Pricing, API, Training, Blog, and About. A small text box at the bottom left shows the URL 'https://github.com/settings/apps'.

https://github.com/settings/apps

Personal access token 생성



Personal access token 생성

The screenshot shows the GitHub 'New Personal Access Token' page. The left sidebar contains links for 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens'. The main content area is titled 'New personal access token' and includes a description of how these tokens function. Four red boxes highlight specific fields: the 'Note' field with the value 'itvillain', the 'Expiration' dropdown set to '30 days', the 'repo' scope which is checked, and the 'Generate token' button. Red Korean text annotations are placed to the right of each highlighted field.

Settings / Developer settings

GitHub Apps
OAuth Apps
Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note
itvillain

Expiration *
30 days The token will expire on Fri, Aug 19 2022

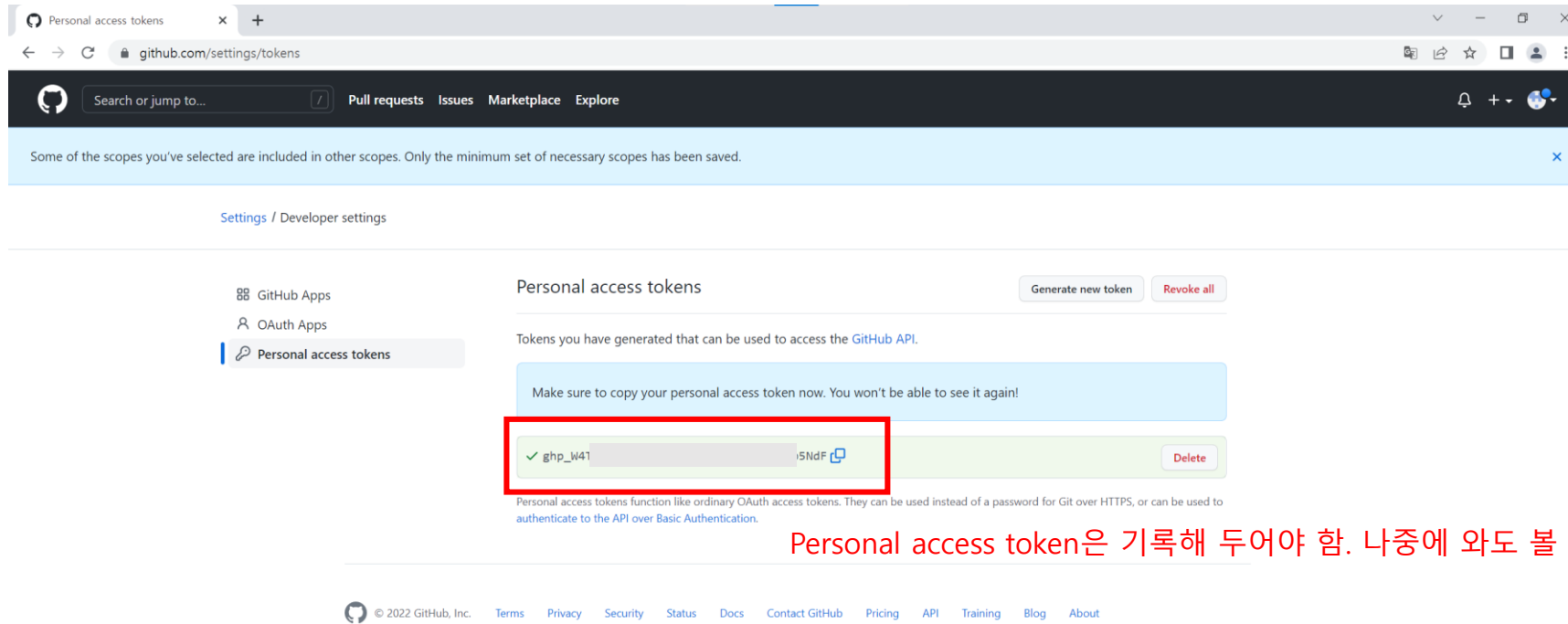
Select scopes
Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input type="checkbox"/> write:gpg_key	Write public user GPG keys
<input type="checkbox"/> read:gpg_key	Read public user GPG keys

Generate token Cancel

© 2022 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

Personal access token 생성



Personal access tokens

Generate new token Revoke all

Tokens you have generated that can be used to access the GitHub API.

Make sure to copy your personal access token now. You won't be able to see it again!

✓ ghp_w41 i5NdF Delete

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

© 2022 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

Personal access token은 기록해 두어야 함. 나중에 봐도 볼 수 없음.