

# Open Source SW Utilization

(524820-2)

송영상(Youngsang Song)

sw.yssong@dankook.ac.kr

# Outline

---

- Github
  - Github 연결
  - Github로 소스 관리
  - 협업(collaboration)
- 명령어는 녹색으로 표시

## ● Github

- 깃허브(Github)는 분산 버전 관리 툴인 깃(Git)를 사용하는 프로젝트를 지원하는 웹호스팅 서비스
- github는 **버전 관리와 협업**을 위한 코드 **웹 호스팅 플랫폼**으로, 언제, 어디서나 협업 프로젝트를 쉽게 진행할 수 있도록 돕는 역할



## ● Github 용어

Git	코드 버전 관리 프로그램
리포 (repo, repository)	코드 저장소
브랜치 (branch)	코드의 버전 이름
마스터/메인 (master/main)	메인/디폴트 브랜치
커밋 (commit)	코드 변경사항 제출
PR (pull request)	코드 리뷰/검토 신청
머지 (merge)	변경사항을 메인 브랜치에 반영



# GitHub

- 환경 셋팅

- Git 설치

- Git configuration Setting

- VS code 설치

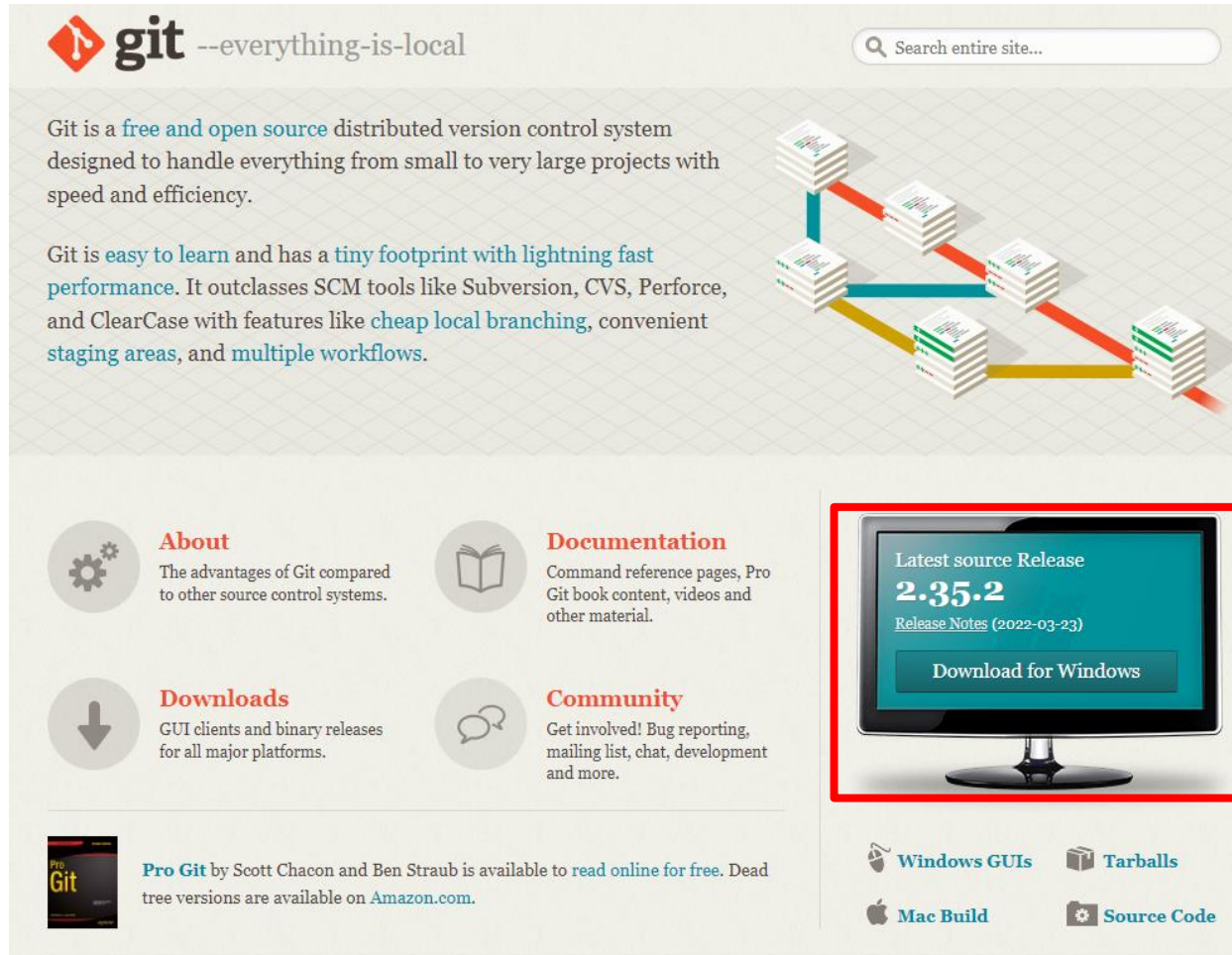
- Shell Setting

- Github 가입

- Github가입
    - 소스 올리기

## ● Git 설치

- <http://git-scm.com/>



The screenshot shows the Git website homepage. At the top, the Git logo is followed by the tagline "--everything-is-local". A search bar is located in the top right corner. The main content area features two paragraphs describing Git as a free and open source distributed version control system, designed for handling projects of various sizes with speed and efficiency. It also mentions that Git is easy to learn, has a tiny footprint, and lightning fast performance, outclassing other SCM tools like Subversion, CVS, Perforce, and ClearCase. To the right of the text is a diagram illustrating Git's distributed nature with multiple repositories connected by a network. Below the main text are four sections: "About" (advantages of Git), "Documentation" (command reference, Pro Git book, videos), "Downloads" (GUI clients, binary releases), and "Community" (bug reporting, mailing list, chat). A red box highlights a monitor displaying the "Latest source Release 2.35.2" with a "Download for Windows" button. At the bottom, there are links for "Pro Git" (available online for free), "Windows GUIs", "Tarballs", "Mac Build", and "Source Code".

git --everything-is-local

Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

**About**  
The advantages of Git compared to other source control systems.

**Documentation**  
Command reference pages, Pro Git book content, videos and other material.

**Downloads**  
GUI clients and binary releases for all major platforms.

**Community**  
Get involved! Bug reporting, mailing list, chat, development and more.

**Latest source Release**  
**2.35.2**  
[Release Notes \(2022-03-23\)](#)  
[Download for Windows](#)

**Pro Git** by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

[Windows GUIs](#) [Tarballs](#)  
[Mac Build](#) [Source Code](#)

## ● Terminal & Shell & Bash

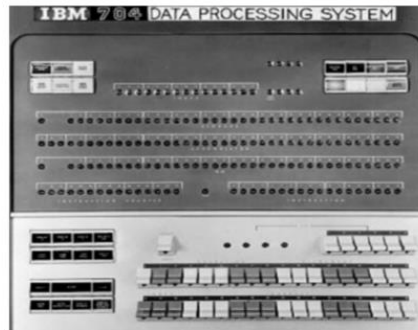
- The **terminal** is the **GUI window** that you see on the screen. It takes commands and shows output
- The **shell** is the **software** that interprets and executes the various commands that we type in the terminal.
- **Bash** is a **particular shell**.

1. **Terminal** physical input/output connected to a



<http://www.istockphoto.com>

2. **Console** physical port linked to a



<http://www.ibm.com>

3. **Mainframe** hardware controlled by



<https://en.wikipedia.org>

5. **Shell** software for input/output

**bash** one of the "modern (1989)" shells  
(Like Chrome is a modern browser)

4. **Kernel** backend software accessed by

## ● Git 설치 확인

### ■ Window key + R : **Git Bash** or **PowerShell** or **cmd**

#### ● 버전확인 : **git --version**

```
MINGW64:/c/Users/ysson  
  
yssong@DESKTOP-GT35BCL MINGW64 ~  
$ git --version  
git version 2.35.1.windows.2  
  
yssong@DESKTOP-GT35BCL MINGW64 ~  
$ |
```

```
Windows PowerShell  
PS C:\Users\ysson> 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  
  
work on the current change (see also: git help everyday)  
  add    Add file contents to the index  
  mv     Move or rename a file, a directory, or a symlink  
  restore Restore working tree files  
  rm     Remove files from the working tree and from the index  
  
examine the history and state (see also: git help revisions)  
  bisect Use binary search to find the commit that introduced a bug  
  diff   Show changes between commits, commit and working tree, etc  
  grep   Print lines matching a pattern  
  log    Show commit logs  
  show   Show various types of objects  
  status Show the working tree status  
  
grow, mark and tweak your common history  
  branch List, create, or delete branches  
  commit Record changes to the repository  
  merge  Join two or more development histories together  
  rebase Reapply commits on top of another base tip  
  reset  Reset current HEAD to the specified state  
  switch Switch branches  
  tag    Create, list, delete or verify a tag object signed with GPG  
  
collaborate (see also: git help workflows)  
  fetch  Download objects and refs from another repository  
  pull   Fetch from and integrate with another repository or a local branch  
  push   Update remote refs along with associated objects  
  
'git help -a' and 'git help -g' list available subcommands and some  
concept guides. See 'git help <command>' or 'git help <concept>'  
to read about a specific subcommand or concept.  
See 'git help git' for an overview of the system.  
PS C:\Users\ysson>
```



## ● Git 초기 설정

- 사용자명 등록 `$ git config --global user.name "XXXXXX"`
- 메일 주소 등록 `$ git config --global user.email "XX@xx.xx"`

```
Windows PowerShell
PS C:\Users\ysson> git config --global user.name "syscrytpo"
PS C:\Users\ysson> git config --global user.email "sw.yssong@dankook.ac.kr"
PS C:\Users\ysson>
```

## ● 설정 확인 : VS code Terminal

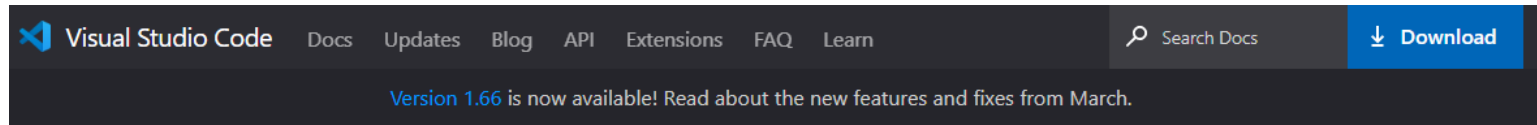
- `git config --list`
  - 빠져나오기 :q

```
MINGW64:/c/Users/ysson
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=true
pull.rebase=false
credential.helper=manager-core
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
filter.lfs.clean=git-lfs clean -- %f
user.name=syscrytpo
user.email=sw.yssong@dankook.ac.kr

ysson@DESKTOP-GT35BCL MINGW64 ~
$
```

## ● Visual Studio Code

- <https://code.visualstudio.com/download>



## Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



### ↓ Windows

Windows 7, 8, 10, 11

User Installer [64 bit](#) [32 bit](#) [ARM](#)  
System Installer [64 bit](#) [32 bit](#) [ARM](#)  
.zip [64 bit](#) [32 bit](#) [ARM](#)



### ↓ .deb

Debian, Ubuntu

### ↓ .rpm

Red Hat, Fedora, SUSE

.deb [64 bit](#) [ARM](#) [ARM 64](#)  
.rpm [64 bit](#) [ARM](#) [ARM 64](#)  
.tar.gz [64 bit](#) [ARM](#) [ARM 64](#)

[Snap Store](#)

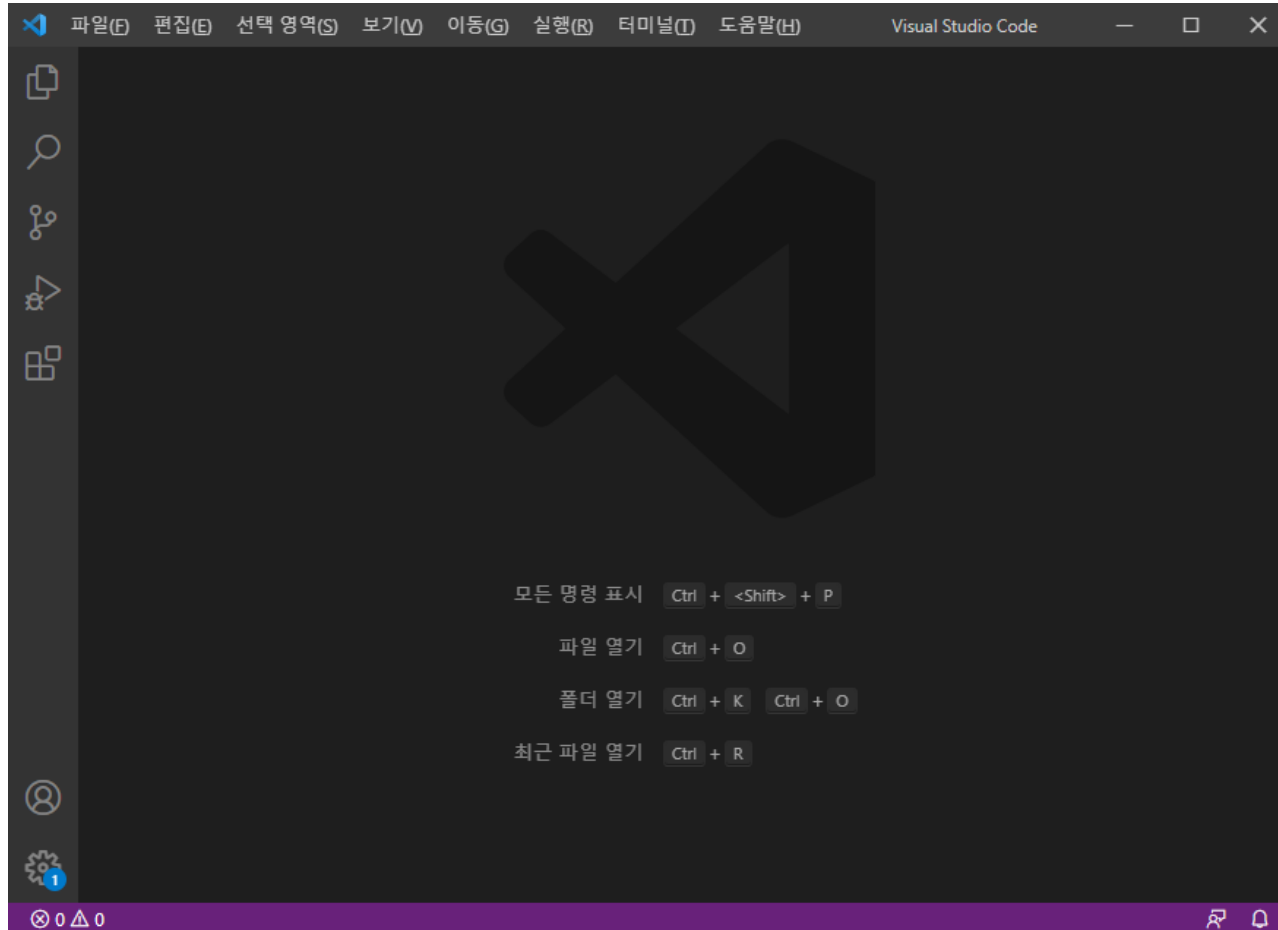


### ↓ Mac

macOS 10.11+

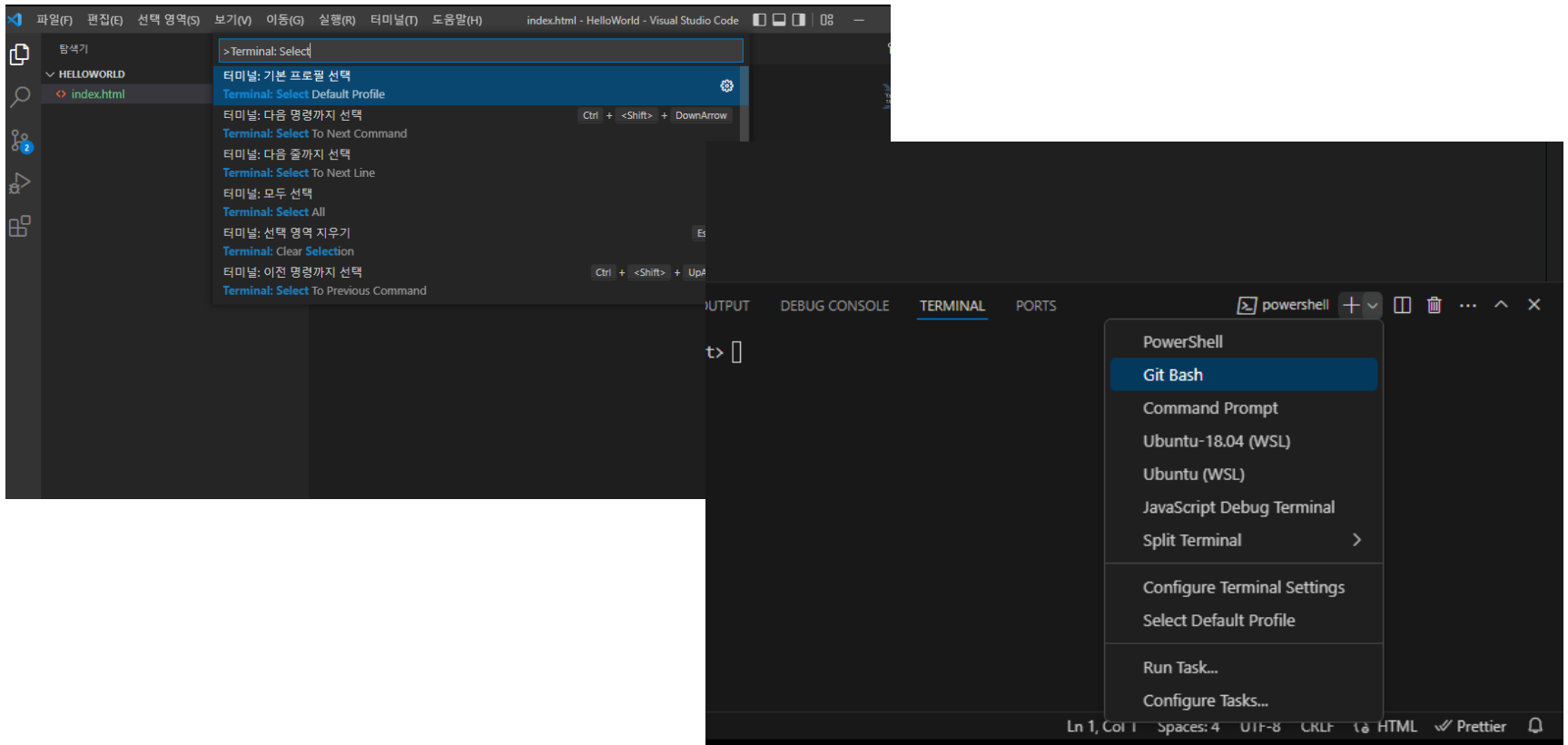
.zip [Universal](#) [Intel Chip](#) [Apple Silicon](#)

- Visual Studio Code 실행
  - CMD창에서 : `code .`



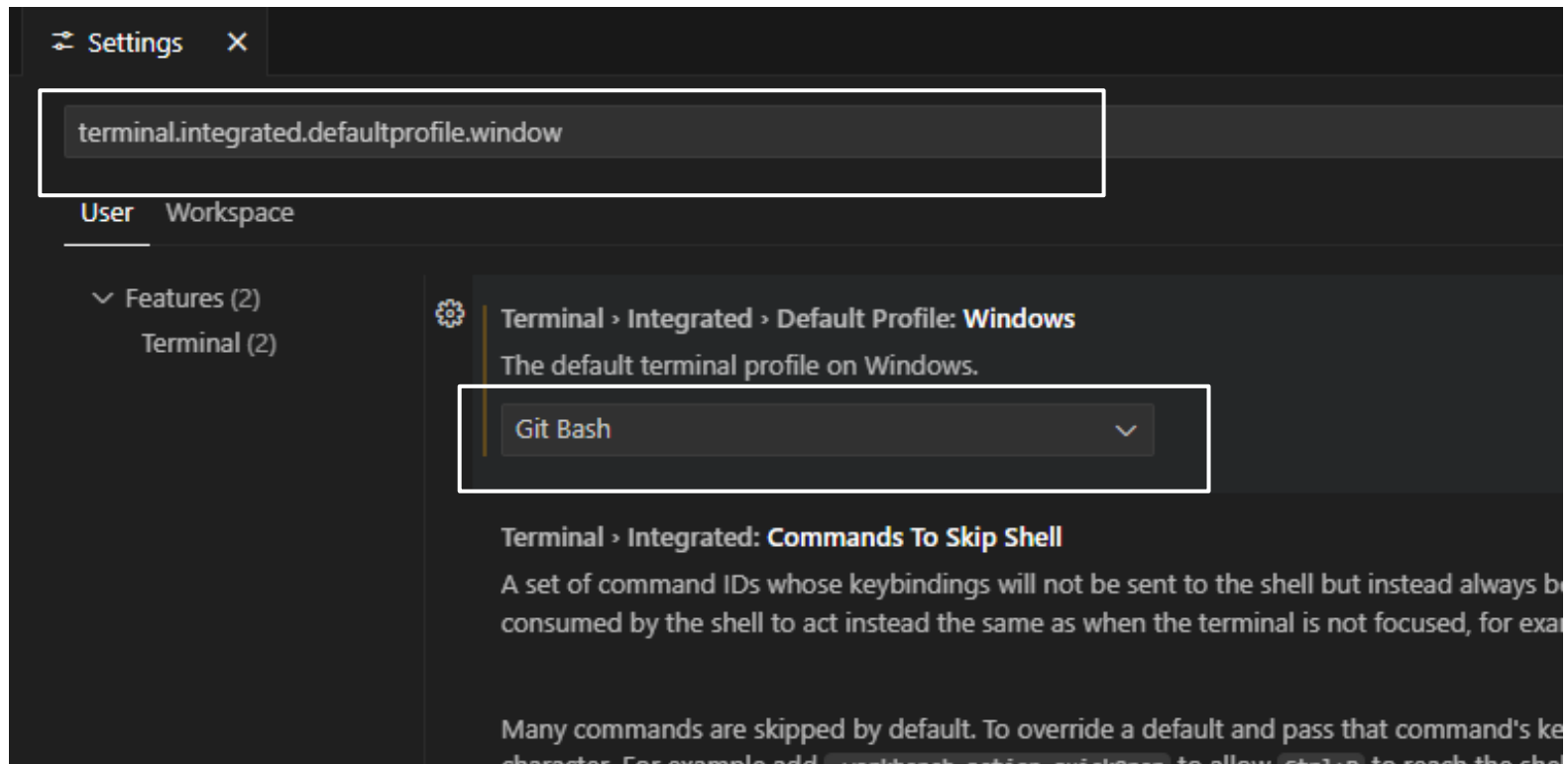
## ● Terminal Shell 변경

- Ctrl + Shift + P
- Command Palette -> Terminal: Select Default Profile



## ● Terminal Shell 설정

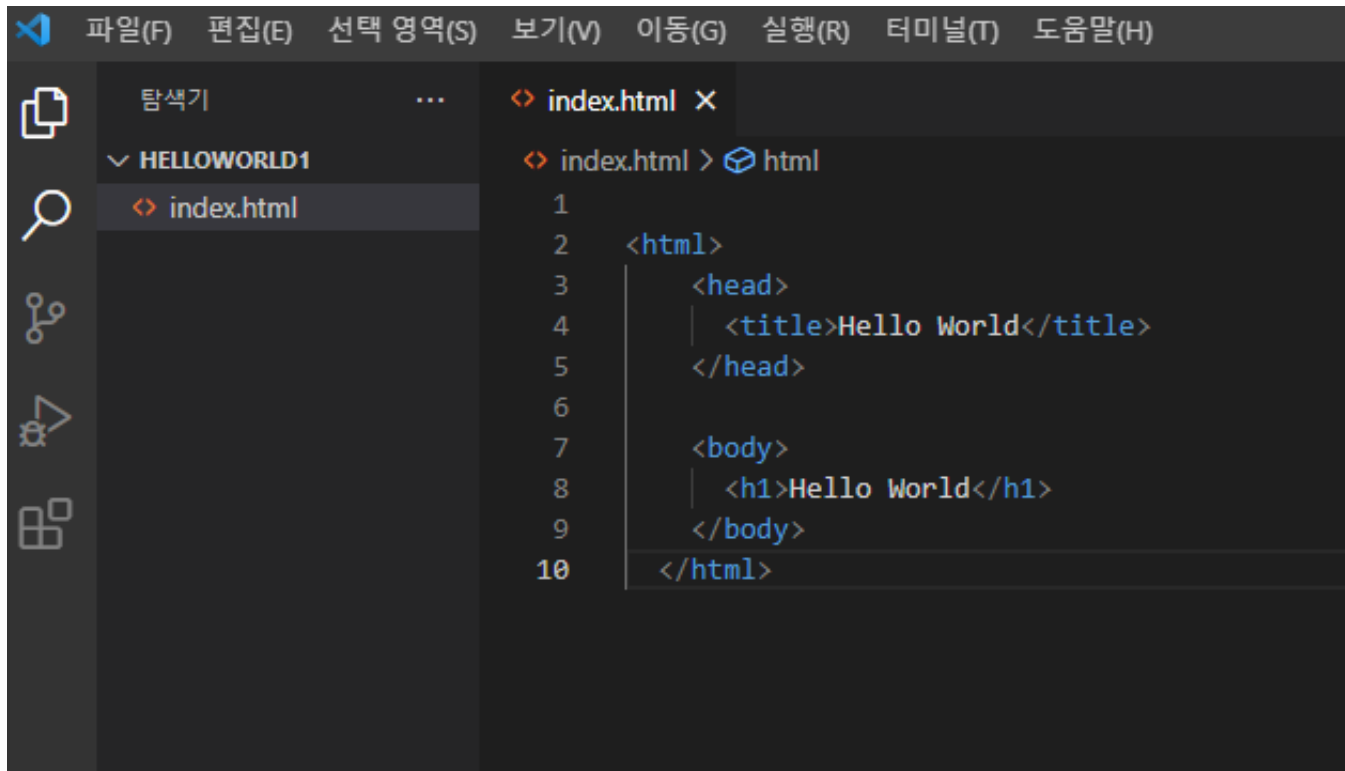
- Ctrl + ,
- terminal.integrated.defaultprofile.windows
- Git Bash 선택



## ● HelloWorld Project

- Project 실행을 위한 폴더 생성
- VS code 파일 – 폴더 열기 : 생성한 폴더 지정
- 파일 만들기 : index.html

HelloWorld



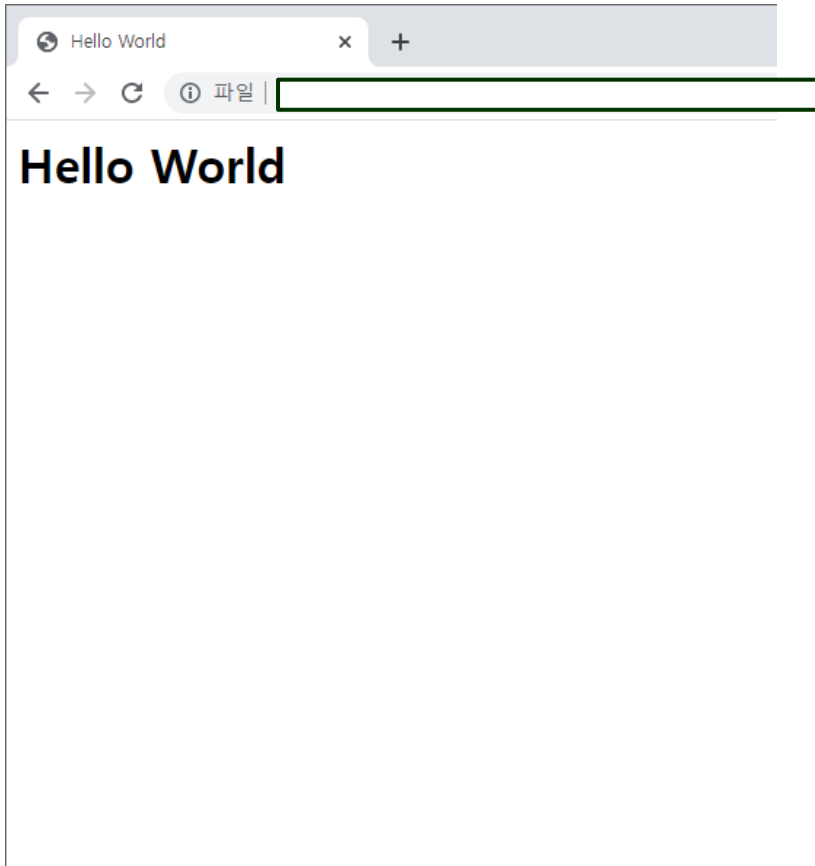
The screenshot shows the VS Code interface with the following details:

- File Explorer (Left):** Shows a folder named 'HELLOWORLD1' containing a file named 'index.html'.
- Editor (Right):** Displays the content of 'index.html' with the following HTML code:

```
1  
2 <html>  
3   <head>  
4     <title>Hello World</title>  
5   </head>  
6  
7   <body>  
8     <h1>Hello World</h1>  
9   </body>  
10  </html>
```

## ● HelloWorld 실행

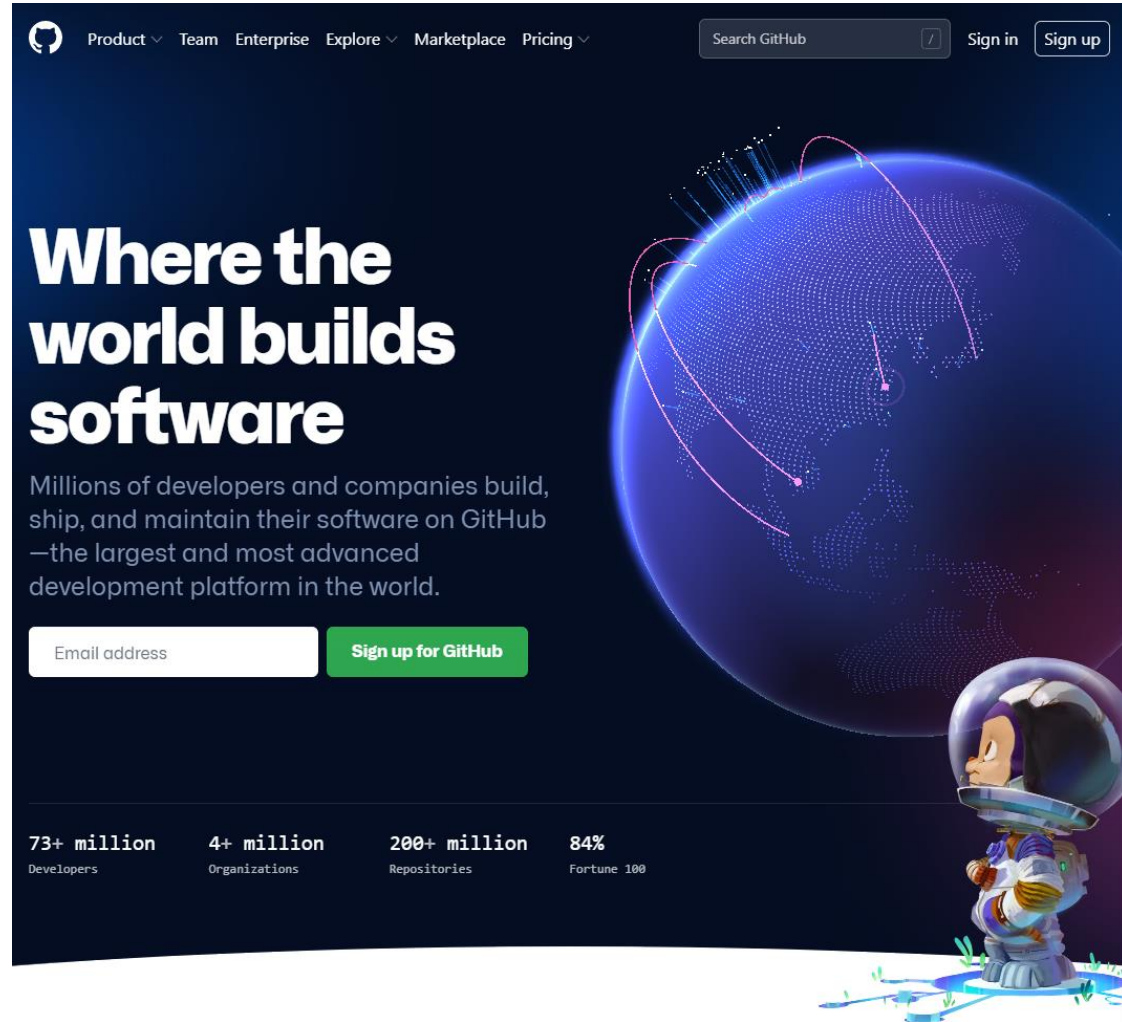
- 실행 – 디버깅없이 실행 (Ctrl + F5)
- Chrome 선택



Run 단축키 :

- start debugging (F5)
- 디버깅없이 실행 (Ctrl + F5)
- Stop debugging(shift + F5)
- Restart debugging (ctrl + shift + F5)

## ● Github가입



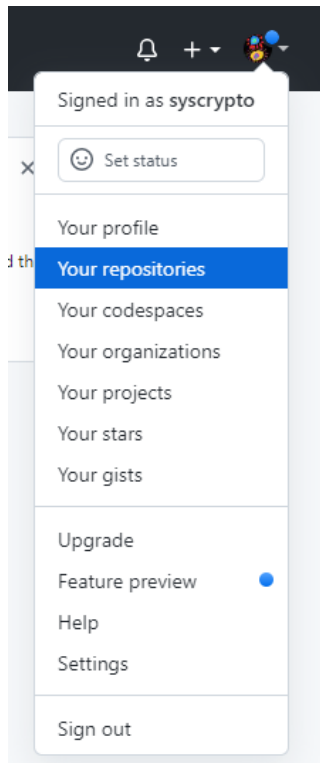


## ● Repository 생성

### ■ Your repositories

### ■ New

### ■ Create repository



## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

### Repository template

Start your repository with a template repository's contents.

No template ▾

Owner \*

syscrypto ▾

Repository name \*

HelloWorld1 ✓

Great repository names are short and memorable. Need inspiration? How about [miniature-disco?](#)

Description (optional)



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.](#)

## ● 첫 번째 Repository 화면

Quick setup — if you've done this kind of thing before

 Set up in Desktop

or

HTTPS

SSH

<https://github.com/syscrypto/HelloWorld1.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# HelloWorld1" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/syscrypto/HelloWorld1.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/syscrypto/HelloWorld1.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

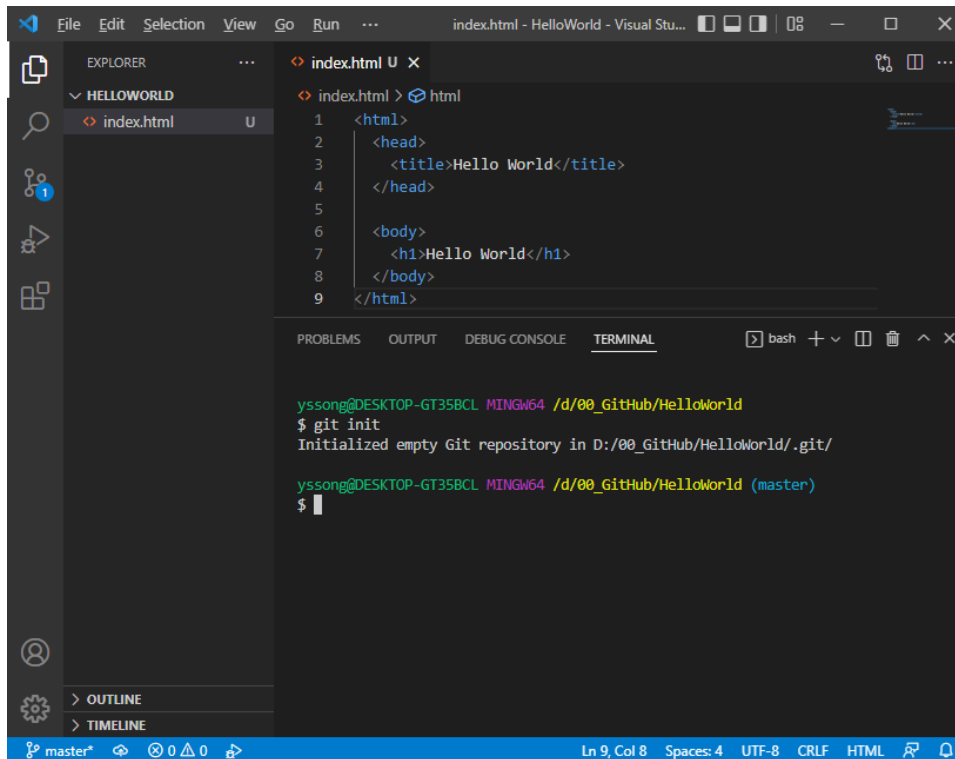
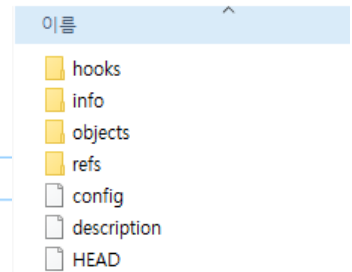
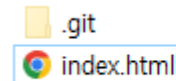
## ● 초기화

### ■ 소스코드 폴더를 버전 관리를 위해 initialize함

#### ● git init

### ■ Project하위 폴더에 .git폴더 생성 확인

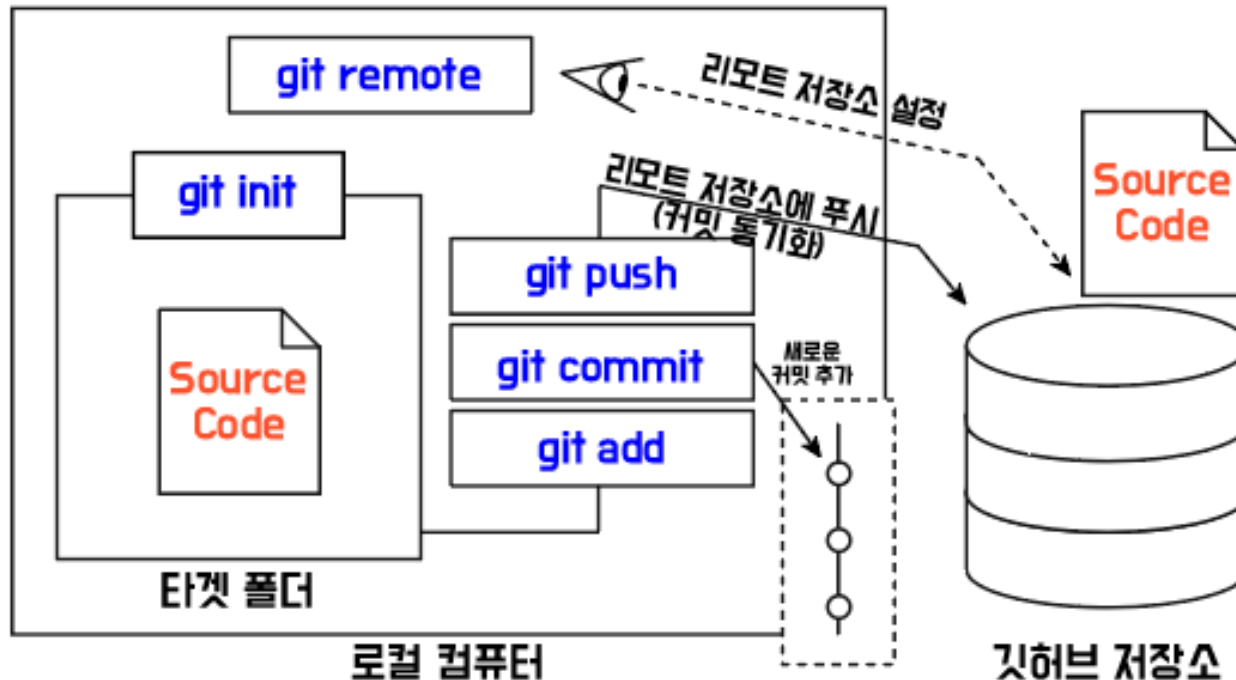
#### ● 버전관리 정보가 저장



## ● Github 소스코드 올리기

### ■ add – commit –(github와 연결) push

- add 를 통해 폴더에서 모든 변경사항을 깃이 체크
- commit 을 통해 변경사항을 새로운 commit으로 저장
- push를 통해 github 저장소와 동기화(update 된 commit)



## ● Github 소스코드 올리기

### ■ `git add .` : 모든 파일 올리기

#### ● `git status` : 상태 확인

### ■ `git commit` : history 만들기

#### ● `git commit -m "history 내용"`

### ■ Github와 연결 -> **Next Page**

### ■ `git push origin main` : 파일 보내기

```
$ git add .  
$ git commit -m "커밋에 대한 간단한 설명 메세지"  
$ git push origin master
```

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1  
$ git add .  
  
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1  
$ git status  
On branch master  
  
No commits yet  
  
Changes to be committed:  
  (use "git rm --cached <file>..." to unstage)  
    new file:   index.html
```

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1  
$ git commit -m "first commit"  
[master (root-commit) 3174985] first commit  
1 file changed, 9 insertions(+)  
create mode 100644 index.html
```

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)  
$ git push origin master  
Enumerating objects: 3, done.  
Counting objects: 100% (3/3), done.  
Delta compression using up to 12 threads  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 285 bytes | 285.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/syscrypto/HelloWorld1.git  
* [new branch]      master -> master
```

## ● Github저장소와 연결

- 폴더와 github 저장소의 remote 주소를 연동
  - `git remote add origin https://github.com/ ...`(복사한 주소를 붙여넣기)
  - `git remote -v` : 연결 확인
- origin : github저장소에 업로드, 다른 주소로도 등록 가능
  - 변경 : `git remote set-url origin <git url>`

The image shows a screenshot of the GitHub 'Quick setup' interface and a terminal window. In the 'Quick setup' section, the 'HTTPS' option is selected, and the URL 'https://github.com/syscrypto/HelloWorld1.git' is entered in the text box. Below this, a section titled '...or create a new repository on the command line' provides a list of commands. A red box highlights the last three commands in this list: 'git branch -M main', 'git remote add origin https://github.com/syscrypto/HelloWorld1.git', and 'git push -u origin main'. To the right, a terminal window shows the execution of these commands. The first command 'git remote add origin https://github.com/syscrypto/HelloWorld1.git' is highlighted with a red box, and the output of 'git remote -v' is shown below it, confirming the remote setup.

```
Quick setup — if you've done this kind of thing before
```

☒ Set up in Desktop or ☒ HTTPS ☐ SSH

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# HelloWorld1" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/syscrypto/HelloWorld1.git
git push -u origin main
```

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git remote add origin https://github.com/syscrypto/HelloWorld1.git

yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git remote -v
origin https://github.com/syscrypto/HelloWorld1.git (fetch)
origin https://github.com/syscrypto/HelloWorld1.git (push)
```

# Error Message

## 오류

```
$ git push origin main
remote: Permission to <git주소> denied to <git hub 계정1>
fatal: unable to access '<git주소>/' : The requested URL returned error: 403
```

- Window – 자격증명관리자
- Github 사용자 계정 확인 및 변경

### 자격 증명 관리

웹 사이트, 연결된 응용 프로그램 및 네트워크에 대해 저장된 로그인 정보를 보고 삭제합니다.



웹 자격 증명



Windows 자격 증명

```
swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (master)
$ git status
On branch master
nothing to commit, working tree clean
```

```
swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (main)
$ git status
On branch main
nothing to commit, working tree clean
```

## ● Github Repository 확인

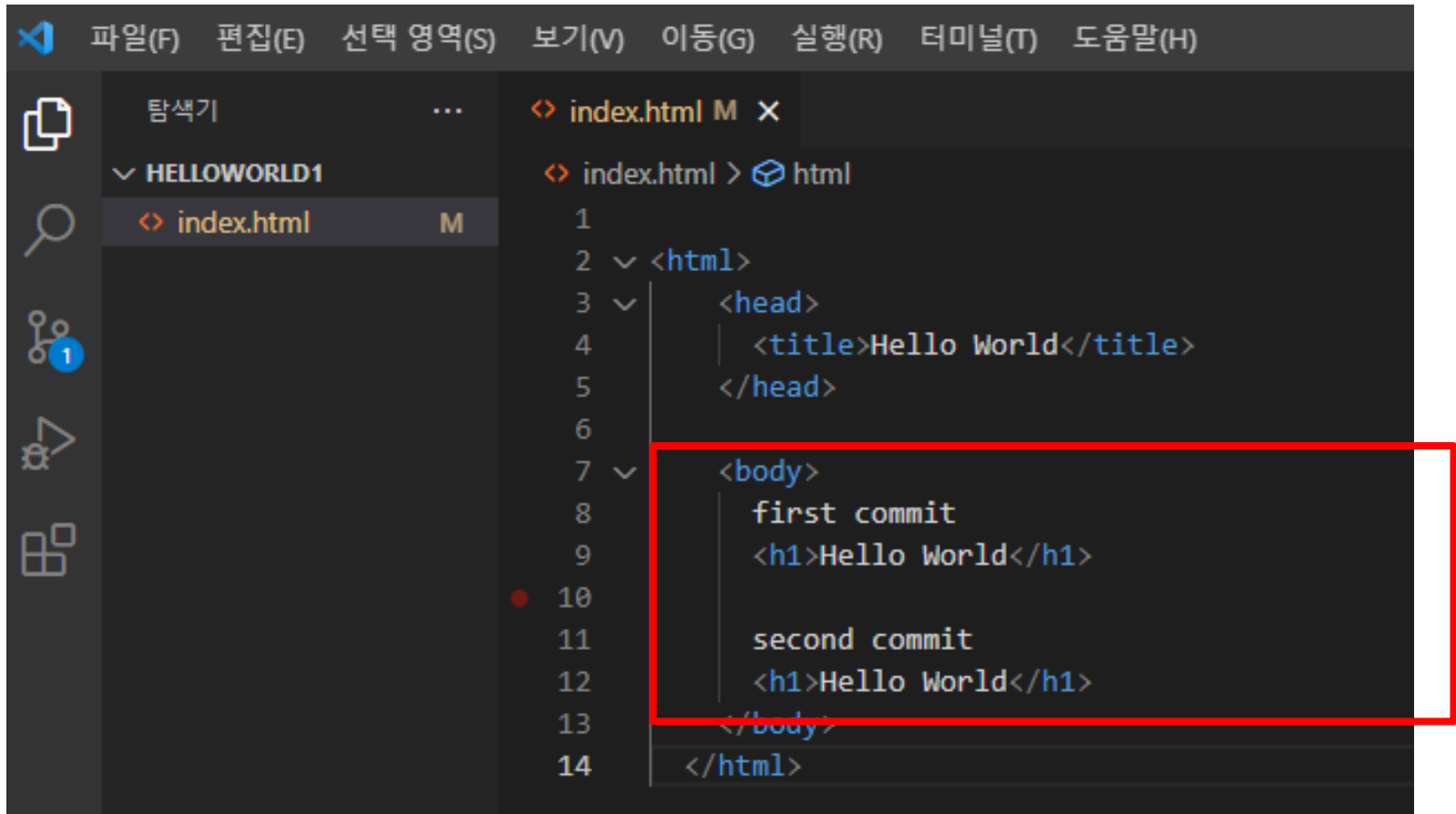
The screenshot shows the GitHub interface for a repository named 'syscrypto / HelloWorld1'. The repository is public and has a 'master' branch with 1 branch and 0 tags. The commit history shows a single commit by 'syscrypto' titled 'first commit' with hash '2bf270b' made 4 minutes ago. The file 'index.html' is highlighted, showing its first commit. The file content is displayed as follows:

```
10 lines (8 sloc) | 130 Bytes
1  <html>
2    <head>
3      <title>Hello World</title>
4    </head>
5    <body>
6      <h1>Hello World</h1>
7    </body>
8  </html>
```

An arrow points from the 'index.html' file in the commit history to the file view below.



## ● 파일 수정



The screenshot shows the Visual Studio Code editor interface. The top menu bar includes options like '파일(F)', '편집(E)', '선택 영역(S)', '보기(V)', '이동(G)', '실행(R)', '터미널(T)', and '도움말(H)'. The left sidebar shows a file explorer with a folder named 'HELLOWORLD1' containing a file 'index.html'. The main editor area displays the content of 'index.html', which is an HTML document. The code is as follows:

```
<? index.html M X
<? index.html > html
1
2 <html>
3   <head>
4     <title>Hello World</title>
5   </head>
6
7   <body>
8     first commit
9     <h1>Hello World</h1>
10
11     second commit
12     <h1>Hello World</h1>
13   </body>
14 </html>
```

A red rectangular box highlights the content within the `<body>` tags, specifically the text 'first commit', the first `<h1>Hello World</h1>` tag, 'second commit', and the second `<h1>Hello World</h1>` tag.

- 수정 파일 적용

- git add .

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git add .

yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.html
```

- git commit -m "second commit"

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git commit -m "second commit"
[master f6bbbf0] second commit
1 file changed, 4 insertions(+)
```

- git push origin master

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 321 bytes | 321.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/syscrypto/HelloWorld1.git
   2bf270b..f6bbbf0  master -> master
```

## ● 적용 확인

master HelloWorld1 / index.html

syscrypto first commit

1 contributor

10 lines (8 sloc) | 130 Bytes

```
1
2 <html>
3   <head>
4     <title>Hello World</title>
5   </head>
6
7   <body>
8     <h1>Hello World</h1>
9   </body>
10  </html>
```

master 1 branch 0 tags

syscrypto second commit

index.html second commit

master HelloWorld1 / index.html

syscrypto second commit

1 contributor

14 lines (11 sloc) | 204 Bytes


```
1
2 <html>
3   <head>
4     <title>Hello World</title>
5   </head>
6
7   <body>
8     first commit
9     <h1>Hello World</h1>
10
11    second commit
12    <h1>Hello World</h1>
13  </body>
14  </html>
```

## ● 변경사항 표시

### ■ second commit click -> Split/Unified

second commit

master

 syscrypto committed 7 minutes ago

1 parent [2bf270b](#)    commit [f6bbb0c9e40dab6c3ae0788e7c88776a7285860](#)

[Browse files](#)

Showing 1 changed file with 4 additions and 0 deletions.

4 index.html

@@ -5,6 +5,10 @@

5

</head>

6

7

<body>


8

<h1>Hello World</h1>

9

</body>

10

</html> 

5

</head>

6

7

<body>

8

+ first commit

9

<h1>Hello World</h1>

10

+ second commit

11

+ <h1>Hello World</h1>


12

+ </body>

13

</body>

14

</html> 

## ● Github에서 소스 수정

<> Edit file

Preview changes

1

2 <html>

3 <head>

4 <title>Hello World</title>

5 </head>

6

7 <body>

8 first commit

9 <h1>Hello World</h1>

10

11 second commit

12 <h1>Hello World</h1>

13

14 Modify line

15 <h1>Hello World</h1>

16

17 </body>

18 </html>

master 1 branch 0 tags


Go to file

Add file

Code

syscrypto Update index.html 840fdb0 now 3 commits

index.html Update index.html now



Commit changes

Update index.html

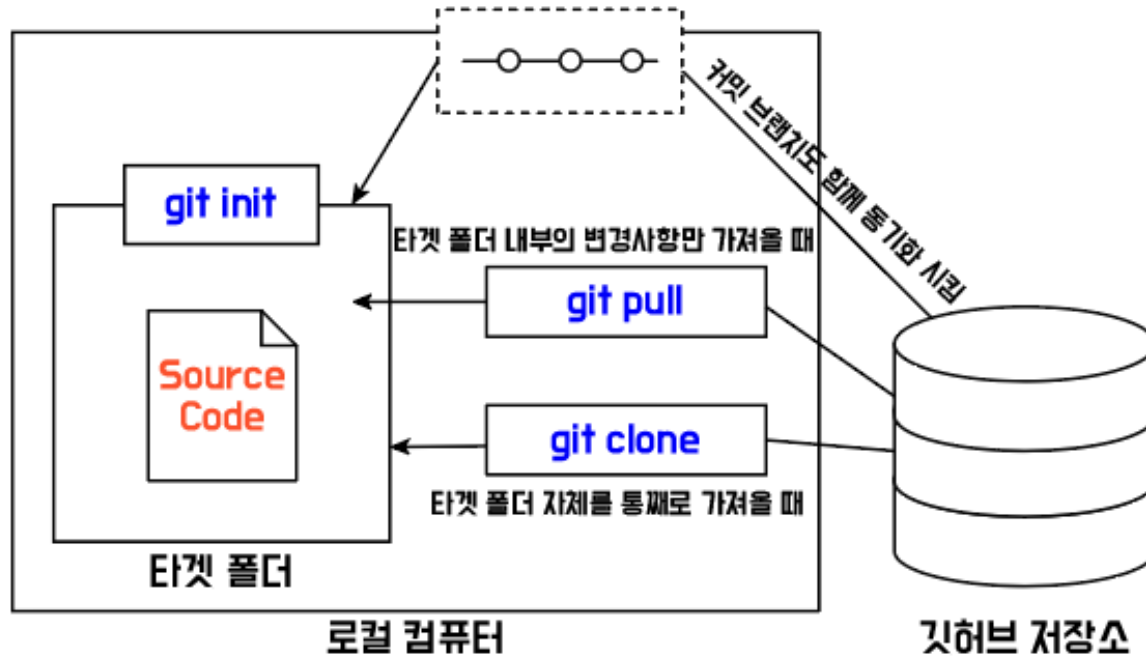
Add an optional extended description...

☒ Commit directly to the master branch.  
☐ Create a new branch for this commit and start a pull request.

Commit changes Cancel

## ● 소스 가져오기

- clone : 모든 소스를 가져오기
  - git clone 주소(remote 주소)
- pull : 수정된 사항만 가져오기
  - git pull origin master



- 수정된 파일 갖고 오기
  - `git pull origin master`

```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 698 bytes | 6.00 KiB/s, done.
From https://github.com/syscrypto/HelloWorld1
* branch            master      -> FETCH_HEAD
   f6bbbf0..840fdb0  master      -> origin/master
Updating f6bbbf0..840fdb0
Fast-forward
 index.html | 6 +++++-
 1 file changed, 5 insertions(+), 1 deletion(-)
```

```
<> index.html > ...
1
2  ∨ <html>
3  ∨   <head>
4      |   <title>Hello World</title>
5      | </head>
6
7  ∨   <body>
8      |   first commit
9      |   <h1>Hello World</h1>
10
11      |   second commit
12      |   <h1>Hello World</h1>
13
14      |   Modify line
15      |   <h1>Hello World</h1>
16
17      | </body>
18      | </html>
19
```


## ● Third commit


```
<> index.html > html > body
1
2 <html>
3   <head>
4     <title>Hello World</title>
5   </head>
6
7   <body>
8     first commit
9     <h1>Hello World</h1>
10
11    second commit
12    <h1>Hello World</h1>
13
14    Modify line
15    <h1>Hello World</h1>
16
17    third commit
18    <h1>Hello World</h1>
19
20  </body>
21 </html>
```


```
yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git add .
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.html


yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git commit -m "third commit"
[master f1cb1f0] third commit
1 file changed, 3 insertions(+)


yssong@DESKTOP-GT35BCL MINGW64 /d/00_GitHub/HelloWorld1 (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 287 bytes | 287.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/syscrypto/HelloWorld1.git
   840fdb0..f1cb1f0  master -> master
```


 main ▾

 1 Branch

 0 Tags



 **syscrypto** Third commit

 index.html

Third commit



# Checking

- 변경된 내용 없이 commit 하는 경우

```
SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (main)
$ git add .
```

```
SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (main)
$ git commit -m "third commit"
On branch main
nothing to commit, working tree clean
```

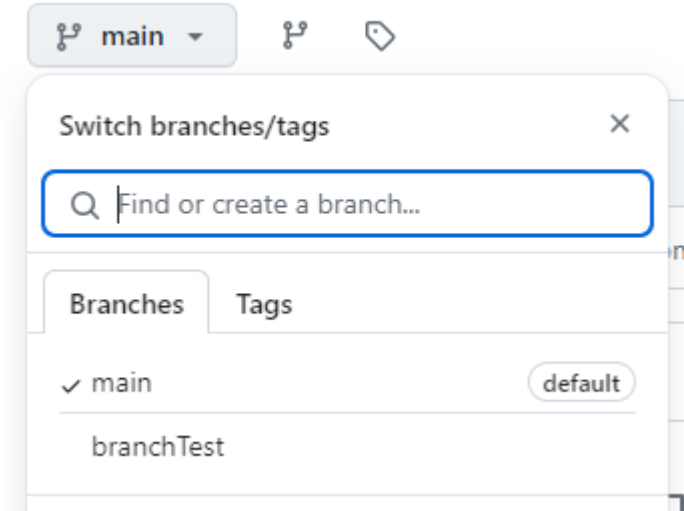
```
SWCU_sys@DESKTOP-3S6F5MB MIN $ git add .
SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (main)
$ git push origin main
Everything up-to-date
```

```
SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (main)
$ git commit -m "fourth commit"
[main c7787eb] fourth commit
1 file changed, 2 insertions(+)
```

```
SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 286 bytes | 286.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/syscrypto/HelloWorld.git
dad28af..c7787eb main -> main
```

# Branch 생성 및 활용

- Branch 생성
  - `git branch` 브랜치이름
- Branch로 전환
  - `git checkout` 브랜치이름
- Github에 반영
  - `git push origin` 브랜치이름
- `git checkout -b` 브랜치이름
- 수정 파일 브랜치에 추가



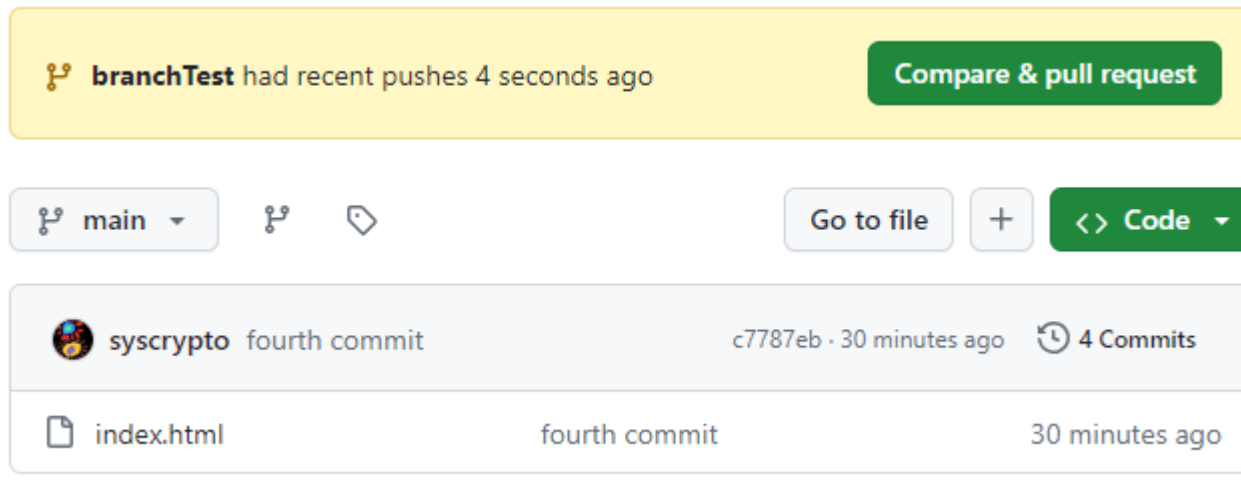
```
SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (branchTest)
$ git add .

SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (branchTest)
$ git commit -m "branch 1 commit"
[branchTest 67d202b] branch 1 commit
1 file changed, 4 insertions(+)

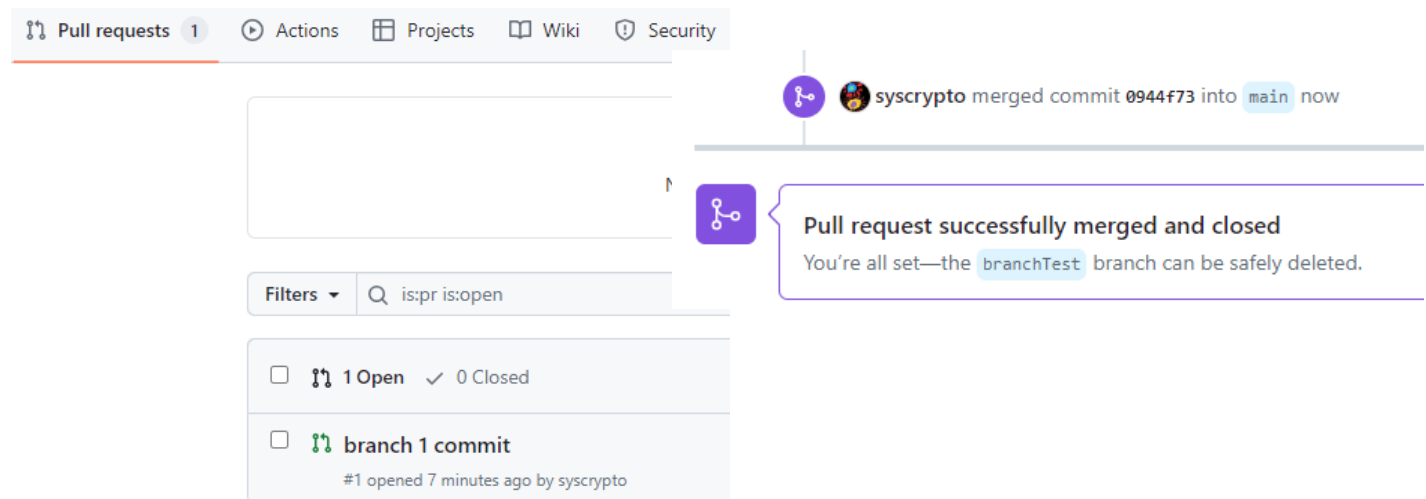
SWCU_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest (branchTest)
$ git push origin branchTest
Enumerating objects: 5, done.
```

# Branch 생성 및 활용

## ● Github branch compare & pull request



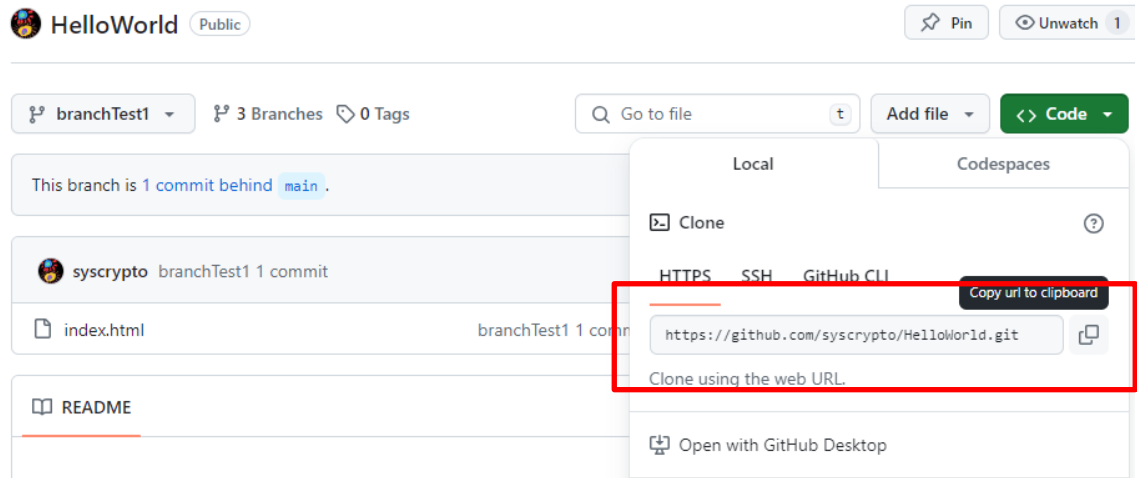
## ● Merge



# Clone

- Vscode 새로운 창열기
- Terminal 열기
- 위치 이동

- `cd / -> cd 폴더명`



- Clone

- `git clone [github clone url] [받아올 디렉토리]`
- Ex. : `git clone https://github.com/XXXXX/HelloWorld.git cloneTest`
- 받은 파일을 수정 후 main push

- Clone branch 생성

- `git checkout -b cloneTester`
- 현재 branch 확인 : `git branch`

# Clone

## ● Branch 확인 : git branch

```
swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest/cloneTest (cloneTester)
$ git branch
* cloneTester
main
```

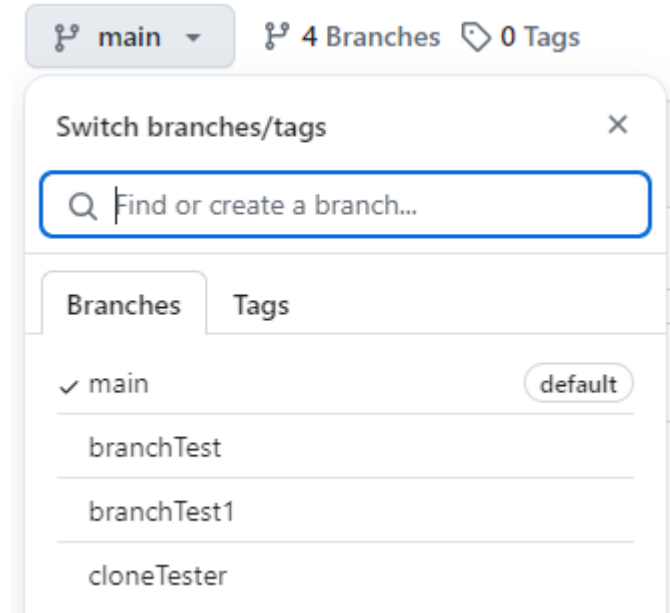
## ● 파일 수정 후 commit

- Git add .
- Git commit -m "cloneTester commit"
- Git push origin cloneTester

```
swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest/cloneTest (cloneTester)
$ git add .

swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest/cloneTest (cloneTester)
$ git commit -m "cloneTester branch"
[cloneTester 2e851ba] cloneTester branch
1 file changed, 4 insertions(+)

swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/gitTest/cloneTest (cloneTester)
$ git push origin cloneTester
```



# Github Collaboration

---

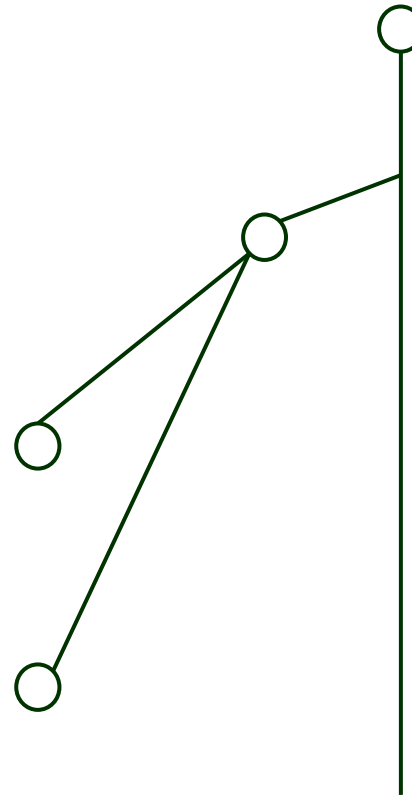
**Project Manager**  
(branch : main)

**Main copy branch**  
(branch : develop)

**Programmer1**  
(branch : function-a)

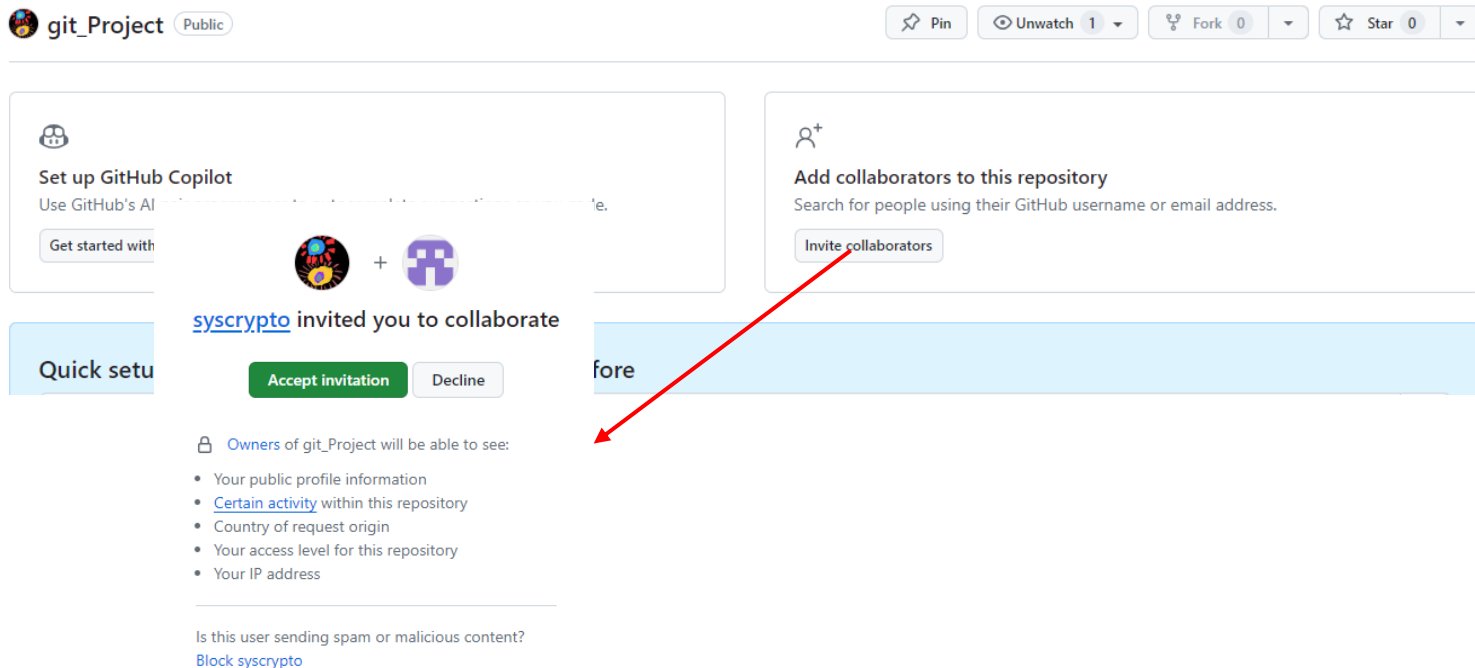
**Programmer2**  
(branch : function-a)

**Github project menu** 활용



# Repository 생성


- Repository name : git\_Project
- Invite collaborators
  - Email or username으로 초대
  - 초대 수락 : Accept invitation



# Repository 생성

- Repository name : git\_Project
- 초기 셋팅

Quick setup — if you've done this kind of thing before

 Set up in Desktop

or

HTTPS

SSH

[https://github.com/syscrypto/git\\_Project.git](https://github.com/syscrypto/git_Project.git)

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#)

...or create a new repository on the command line

```
echo "# git_Project" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/syscrypto/git_Project.git
git push -u origin main
```

- Main code commit
  - git add .
  - git commit -m "main second commit"
  - git push origin main



# Main Branch 복사본 만들기

---

- Develop branch

- Main branch는 서비스용 파일
- 복사본을 만들어 main에 merge하기 전에 테스트 및 검증용으로 사용

- develop branch 생성

- `git checkout -b develop`
- `git push`
- `git push --set-upstream origin develop`
- Github에서 branch 확인

- 참고 사항

- Branch 변경 : `git switch branch명` 또는 `git checkout branch명`
- Branch 확인 : `git branch`

# Main branch Protection

## ● Settings – Branches – Add branch protection rule

- Branch name pattern : main
- Protect matching branches
  - Require a pull request before merging
  - Lock branch

The screenshot shows the GitHub repository settings page for a branch protection rule. On the left is a sidebar with navigation links: Branches (selected), Tags, Rules, Actions, Webhooks, Environments, Codespaces, and Pages. Below these are links for Security, Code security and analysis, Deploy keys, and Secrets and variables. The main content area has a header note: "Your GitHub Free plan can only enforce rules on its public repositories, like this one." Below this are two sections. The first section, "Branch name pattern \*", contains a text input field with the value "main". The second section, "Protect matching branches", contains two checked checkboxes: "Require a pull request before merging" and "Require approvals". Each checkbox has a descriptive text block explaining its function.

[Your GitHub Free plan](#) can only enforce rules on its public repositories, like this one.

**Branch name pattern \***

main

**Protect matching branches**

- ☒ **Require a pull request before merging**  
When enabled, all commits must be made to a non-protected branch and submitted via a pull request before they can be merged into a branch that matches this rule.
- ☒ **Require approvals**  
When enabled, pull requests targeting a matching branch require a number of approvals and no changes requested before they can be merged.

# Protect Test

- Developer가 수정한 코드를 직접 main에 올릴 경우 error

```
swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (develop)
$ git add .

swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (develop)
$ git commit -m "dev 3 commit"
[develop 981d7ad] dev 3 commit
1 file changed, 2 insertions(+)

swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (develop)
$ git push origin main
To https://github.com/syscrypto/git_Project.git
! [rejected]        main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/syscrypto/git_Project.git'
hint: Updates were rejected because a pushed branch tip is behind its remote
hint: counterpart. If you want to integrate the remote changes, use 'git pull'
hint: before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (develop)
$ git push origin develop
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 319 bytes | 159.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/syscrypto/git_Project.git
36ed580..981d7ad develop -> develop
```

# Main branch 에 merge

- Commit된 파일 확인
- Contribute에서 open pull request
- Create pull request

The screenshot displays a GitHub repository interface for a branch named 'develop'. At the top, there are navigation links for 'Go to file', a '+' icon, and a 'Code' button. Below this, a status bar indicates 'This branch is 1 commit ahead of, 1 commit behind main'. A 'Contribute' button is visible on the right. The main content area shows a file list with 'index.html' and 'README' (highlighted with a red underline). A modal window is open, displaying the message 'This branch is 1 commit ahead of main' and 'Open a pull request to contribute your changes upstream.' Below this message are 'Compare' and 'Open pull request' buttons. At the bottom right, there is a 'Create pull request' button. The interface also shows a 'syscrypto dev 4 commit' header and a 'Markdown is supported' notice at the bottom.

# Main branch 에 merge

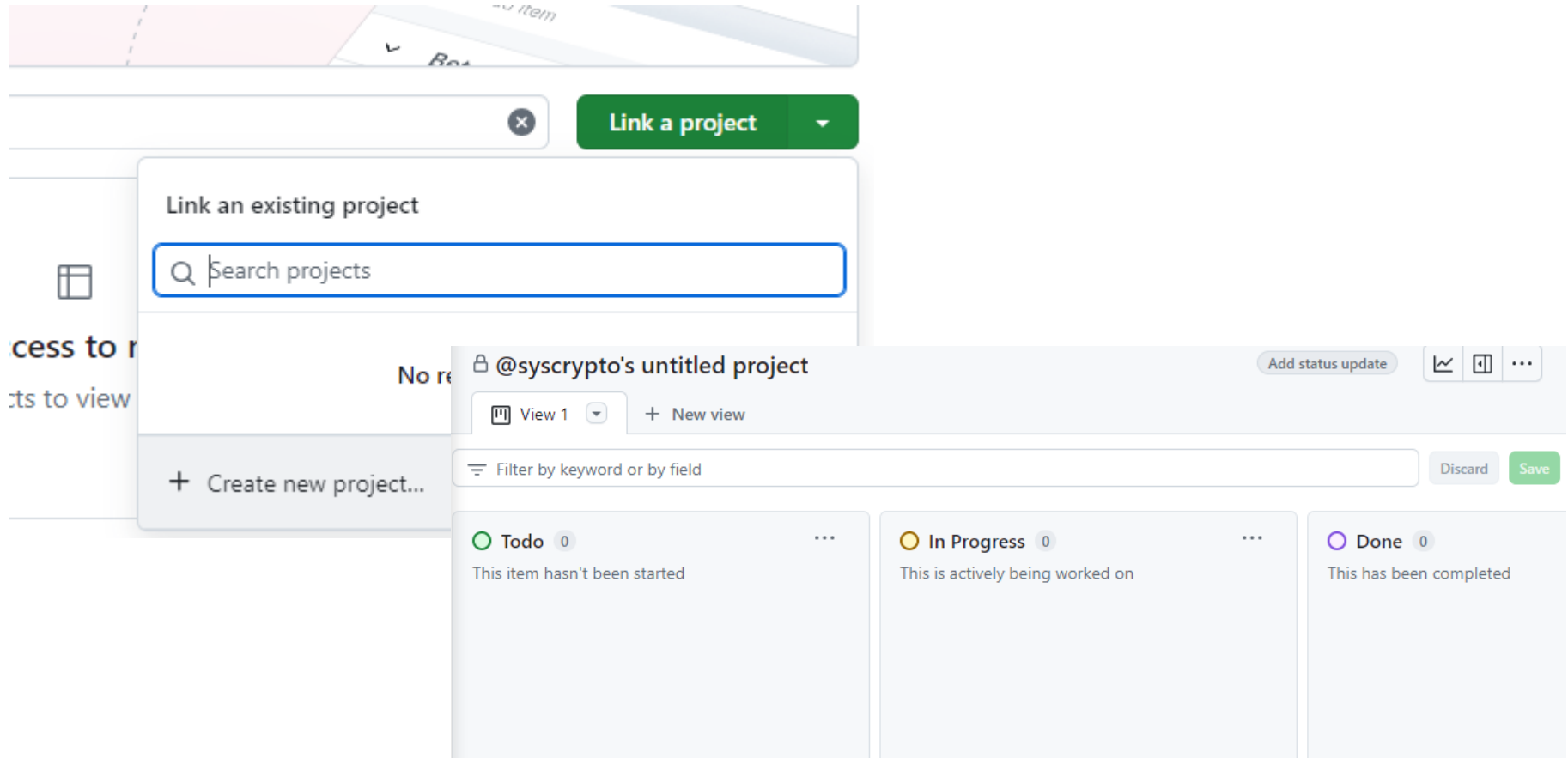
- 현재 main branch는 protect되어 있음
- Checkbox 선택 후 Merge pull request 선택
- Confirm merge

The screenshot shows the GitHub pull request merge interface. It features several error messages and a success message, with blue arrows indicating the correct workflow:

- Review required** (top left): At least 1 approving review is required by reviewers with write access. [Learn more about pull request reviews.](#)
- Merging is blocked** (middle left): Merging can be performed after you have resolved conflicts.
- Merge without waiting for review** (checkbox): This checkbox is highlighted with a blue arrow.
- Merge pull request** (button): This button is highlighted with a blue arrow.
- Review required** (middle right): At least 1 approving review is required by reviewers with write access. [Learn more about pull request reviews.](#)
- Merging is blocked** (bottom middle): Merging can be performed after you have resolved conflicts.
- Merge pull request** (button): This button is highlighted with a blue arrow.
- Merge pull request #4 from syscrypto/develop** (summary): dev 4 commit. This commit will be authored by 35800198+syscrypto@users.noreply.
- Confirm merge** (button): This button is highlighted with a blue arrow.
- Cancel** (button): This button is highlighted with a blue arrow.
- Pull request successfully merged and closed** (bottom left): You're all set—the `develop` branch can be safely deleted. [Delete branch](#)

# Project Management

- Projects – Link a project – Create new project – Board – Create project



# Project Management

- Add item – Function A, Function B – A, B중에 선택 – convert to issue – Create a branch – Branch Source(develop)

The screenshot displays the GitHub interface for creating a branch for an issue. On the left, a 'Todo' issue titled 'Function A' is shown, with a draft comment and a history of updates. The main panel, titled 'Create a branch for this issue', contains the following elements:

- Branch name:** A text input field containing 'function-a'.
- Repository destination:** A dropdown menu showing 'syscrypto/git\_Project'.
- Branch source:** A dropdown menu showing 'develop', which is highlighted with a red box.
- What's next?** Three radio button options: 'Open in codespace' (disabled), 'Checkout locally' (selected), and 'Open branch with GitHub Desktop'.
- Create branch:** A large green button at the bottom right.

Below the 'What's next?' section, a terminal window shows the following commands:

```
git fetch origin
git checkout function-a
```

On the left side of the main panel, there are sections for 'Assignees', 'Labels', 'Projects', 'Milestones', and 'Notifications'. A red box highlights the 'Development' section, which contains a link to 'Create a branch for this issue or request'.

# Project Management

- Branch 정보 copy
- Vscode terminal에 paste -> branch 생성

Checkout in your local repository



Run the following commands in your local clone.

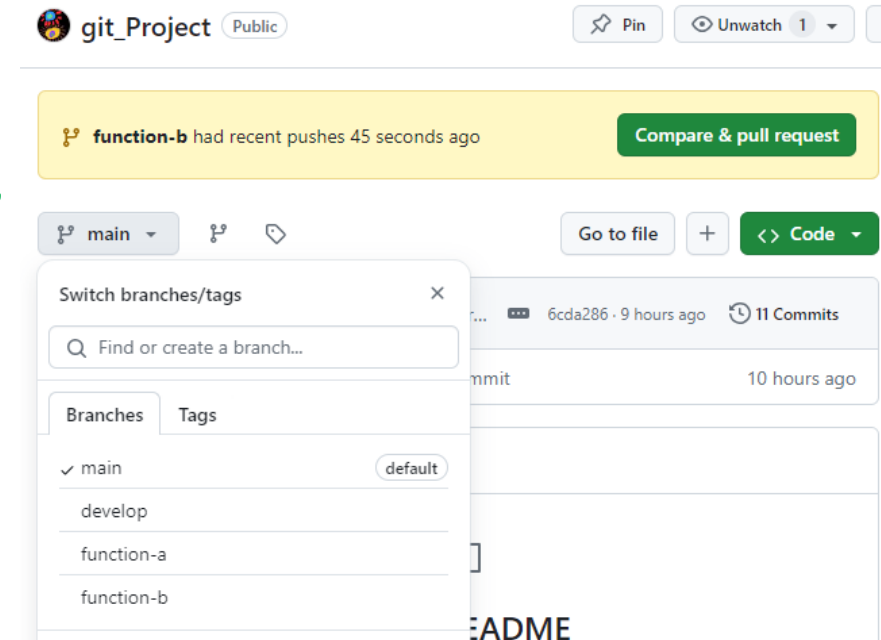
```
git fetch origin  
git checkout function-a
```

```
swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (develop)  
$ git fetch origin  
remote: Enumerating objects: 2, done.  
remote: Counting objects: 100% (2/2), done.  
remote: Compressing objects: 100% (2/2), done.  
remote: Total 2 (delta 0), reused 0 (delta 0), pack-reused 0  
Unpacking objects: 100% (2/2), 1.75 KiB | 162.00 KiB/s, done.  
From https://github.com/syscrypto/git_Project  
* [new branch]      function-a -> origin/function-a  
3560444..6cda286  main      -> origin/main  
  
swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (develop)  
$ git checkout function-a  
branch 'function-a' set up to track 'origin/function-a'.  
Switched to a new branch 'function-a'
```



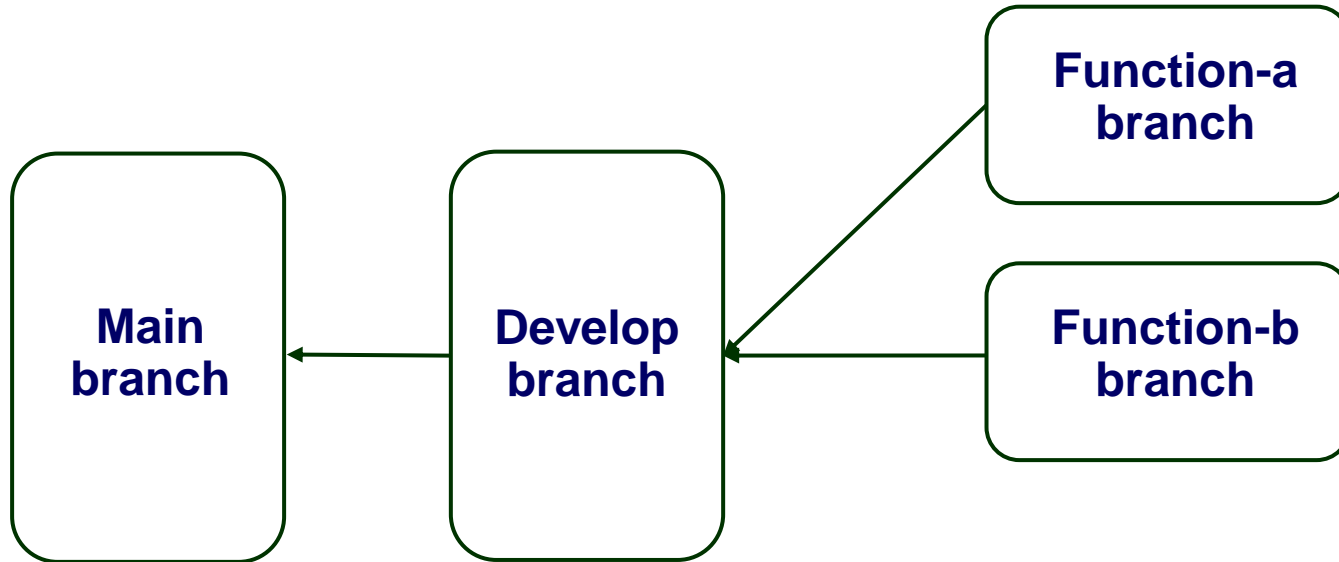
# Project clone

- Programmer10이 git\_Project 에 참여
  - 프로젝트 시작하려는 위치에서 clone
  - `git clone project url directory name`
    - 현재 작업하는 디렉토리 내에 만들면 안됨
- Github project에서 Function-b branch 생성 및 assignment
- 코드 수정 후 commit
  - `git add .`
  - `git commit -m "pro1 first commit"`
  - `git push origin function-b`



# Project management

- Programmer1이 작업한 파일을 function-b로 commit하고
- 이를 develop branch로 이동




## ● Pull request


- Function-b branch에서 develop branch로


The screenshot shows a GitHub pull request interface. At the top, there is a header bar with a red border. It contains a dropdown menu set to 'base: develop', a dropdown menu set to 'compare: function-b', and a green checkmark with the text 'Able to merge. These branches can be automatically merged.' Below this header, there is a section for adding a title and description. The title field contains the text 'pro1 first commit'. The description field is empty and labeled 'Add a description'.

# Pull request


## ● Checking & Merge






**Continuous integration has not been set up**  
[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.




**This branch has no conflicts with the base branch**  
Merging can be performed automatically.


**Merge pull request** ▾  
You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

 **develop** had recent pushes 36 seconds ago **Compare & pull request**

 develop ▾   **Go to file** **+** **<> Code** ▾

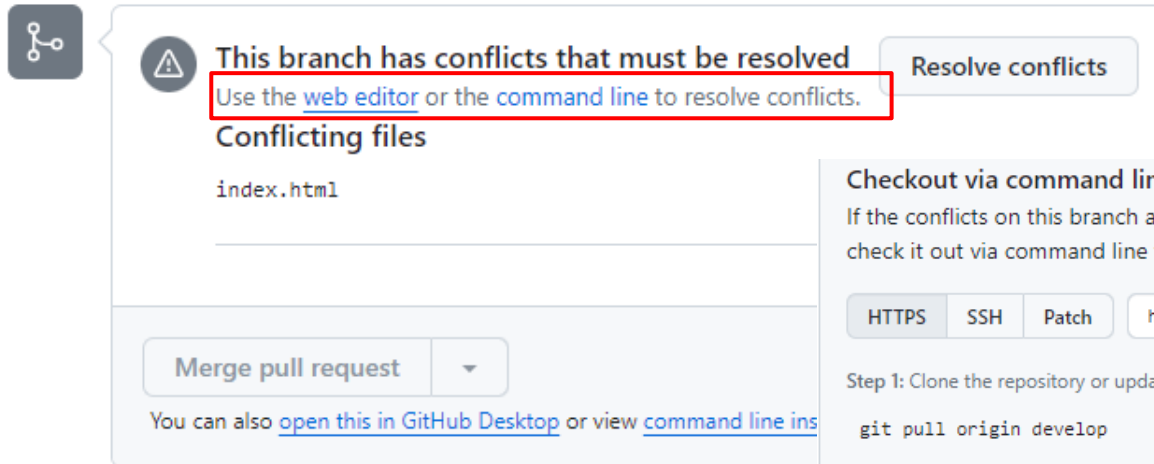
This branch is **2 commits ahead of**, **2 commits behind** **main** . **Contribute** ▾

 **syscrypto** Merge pull request #7 from sys...  a826303 · 1 minute ago  11 Commits

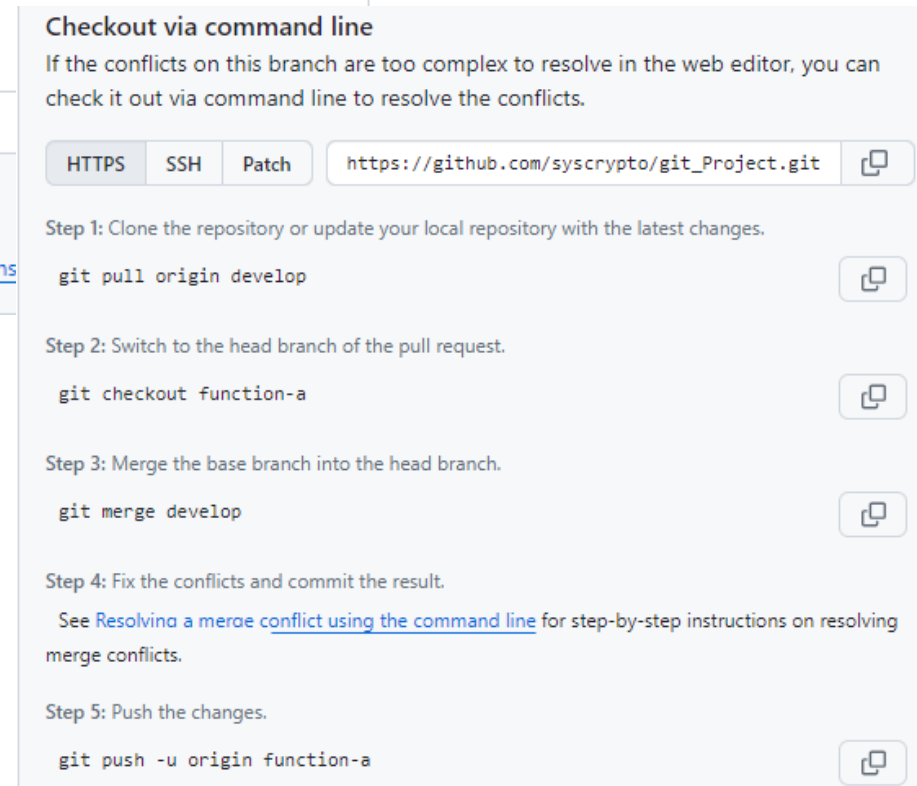
 index.html **pro1 first commit** 14 minutes ago

# Conflict 해결

- PM이 작성한 수정 코드를 develop branch에 commit하고 pull request한 후 Merge 전에 conflict 발생



- Command line click
  - git switch develop 변경 후
  - 순차적으로 수행



# Merge – conflict 해결

```
17      <h1> developer가 추가 코드를 main에 merge 예제 </h1>
18
19      Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
20      <<<<<<< HEAD (Current Change)
21      <h1> PMO이 Function-a 작업한 코드를 develop에 반영 </h1>
22      =====
23      <h1> programmer1은 function-b를 개발 </h1>
24      >>>>>>> develop (Incoming Change)
25      </body>
</html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS




```
Updating 93a0b20..a826303
Fast-forward
 index.html | 2 ++
 1 file changed, 2 insertions(+)


swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (develop)
$ git checkout function-a
Switched to branch 'function-a'
Your branch is up to date with 'origin/function-a'.


swcu_sys@DESKTOP-3S6F5MB MINGW64 /c/git_Project (function-a)
$ git merge develop
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```


# 최종 파일 올리기

- Develop에서 main으로 최종코드 반영


 base: main  compare: develop  **Able to merge.** These branches can be automatically merged.



 **Review required**  
At least 1 approving review is required by reviewers with write access.  
[Learn more about pull request reviews.](#)

 **Merging is blocked**  
Merging can be performed automatically

☐ **Merge without waiting for requirements**


Merge pull request 

This commit will be authored by 35800198+syscrypto@users.noreply.!

Merge pull request #9 from syscrypto/develop

Develop

**Confirm merge** **Cancel**



**Pull request successfully merged and closed**  
You're all set—the `develop` branch can be safely deleted.

Delete branch


524820-2, S'22



SW-Centric University Project

- 54 -




# 최종 배포 파일



---

 main ▾

[Go to file](#) [+](#) [Code ▾](#)

 **syscrypto** Merge pull request #9 from syscrypto/d...  02fe108 · now  17 Commits

 programmer1	pm edit commit	23 minutes ago
 index.html	solve problem	10 minutes ago

# Reference

---

- <https://docs.github.com/en/get-started/quickstart/hello-world>
- <https://medium.com/@kjunha77/%EA%B0%95%EC%9D%98%EB%85%B8%ED%8A%B8-11-%EB%B2%88%EC%99%B8-%EA%B9%83%ED%97%88%EB%B8%8C-%EC%82%AC%EC%9A%A9%EB%B2%95-d8d57f794f5>
- <https://medium.com/@krish.raghuram/terminal-shell-and-bash-3e76218c8865>
- <https://www.youtube.com/watch?v=lelVripbt2M>
- <https://gist.github.com/ihoneymon/652be052a0727ad59601>
- <https://bskyvision.com/1140>
- <https://blog.gaerae.com/2015/01/bash-hello-world.html>
- <https://about.gitlab.com/>
- <https://aws.amazon.com/ko/devops/what-is-devops/>
- <https://ko.wikipedia.org/wiki/%EB%8D%B0%EB%B8%8C%EC%98%B5%EC%8A%A4>
- <https://inpa.tistory.com/entry/GIT-%E2%9A%A1%EF%B8%8F-%EA%B0%9C%EB%85%90-%EC%9B%90%EB%A6%AC-%EC%89%BD%EA%B2%8C%EC%9D%B4%ED%95%B4>
- <https://www.youtube.com/watch?v=tkkbYCajCjM>



# 중간고사 대체 과제

---

## ● <Github 활용>

- 'Mid-Term-Project' Repository를 생성해서 작업
- 최소 15회이상 Commit
- Github Page 또는 서버구축하여 webpage 운영
  - URL 제출

## ● <내용>

- 자기 소개를 위한 개인 홈페이지 작성
  - Template 사용가능
- 최소 이미지 2개 이상
- 옵션사항
  - 동영상 또는 동영상 링크 추가