Introduction to Git & Github

조형기

https://johg15.github.io

전북대학교 전자공학부

2022-1

What is Git & Github?

Version

흔한 대학생의 삶..

HW_조형기_v1.cpp
HW_조형기_v2.cpp
HW_조형기_v2(최종).cpp
HW_조형기_v2(진짜최종).cpp
…
HW_조형기_final.cpp
HW_조형기_final_final.cpp

Version

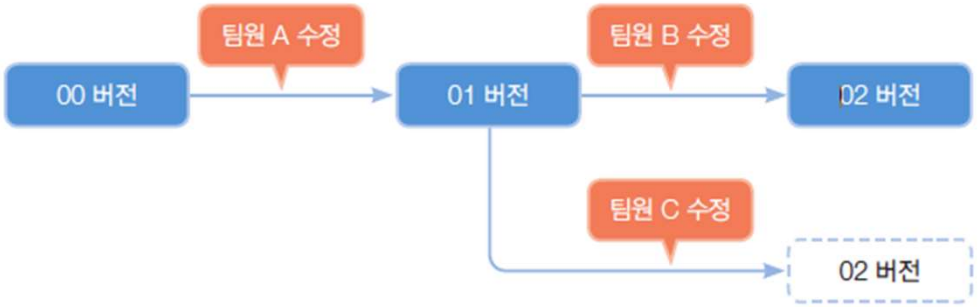
- 내가 원하는 시점, 원하는 버전에 자유롭게 이동 가능
- 디자인, 웹 개발 등등 어떤 프로젝트든지 가능

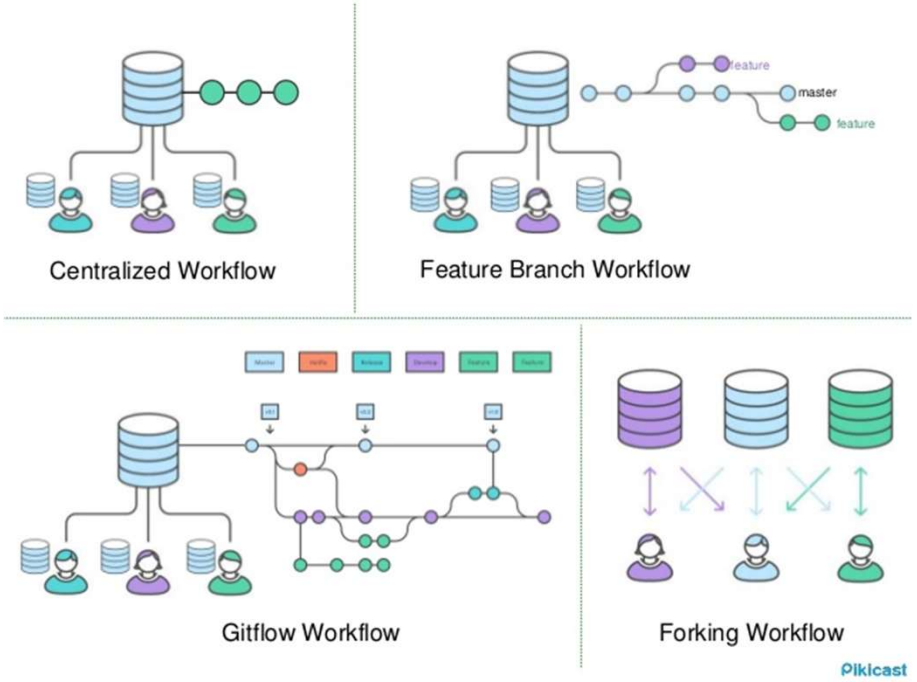
Version

여러 명이 같이 하는 Project라면..?

Project_조형기_v1.cpp
Project_조형기_v2.cpp
Project_조형기_v2(최종).cpp
Project_조형기_v2(진짜최종).cpp
Project_김북대_v2(최종).cpp
Project_팀1_합본.cpp
…
Project_조형기_final.cpp
Project_조형기_final_final.cpp
Project_팀1_합본_최종.cpp

Version





- 여럿이 함께 작업하는 협업 프로젝트에서 더욱 강력함
- 따로 조금씩 작업하다가 원할 때 합치기 + 백업도 가능
- 기존 버전과 현재 버전을 비교하여 바뀌어 있는 부분 확인 가능

Powerful Git

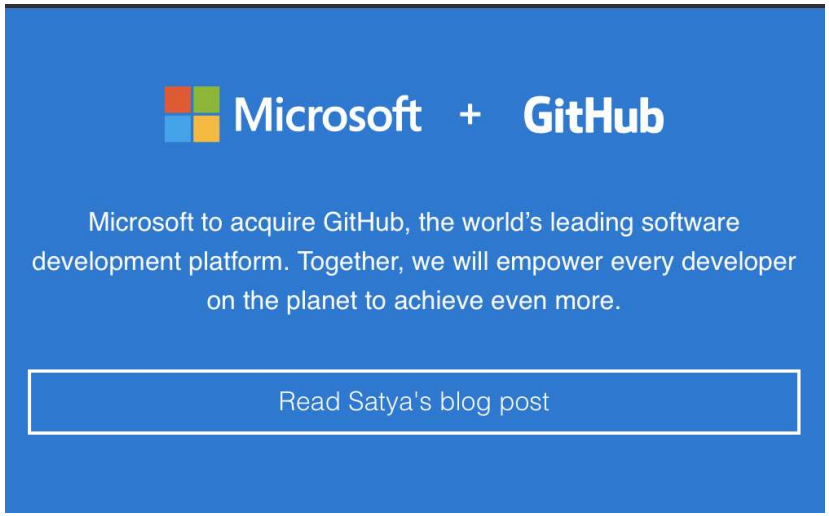
1.
코딩할 때 (코딩에만 한정되어 있지 않음)
단순히 ctrl+z를 눌러 이전상태로 되돌리는 것이 아니라,
원하는 시점마다 깃발을 꽂고(버전을 만들고)
이들 간에 **자유롭게 돌아다닐 수** 있다.

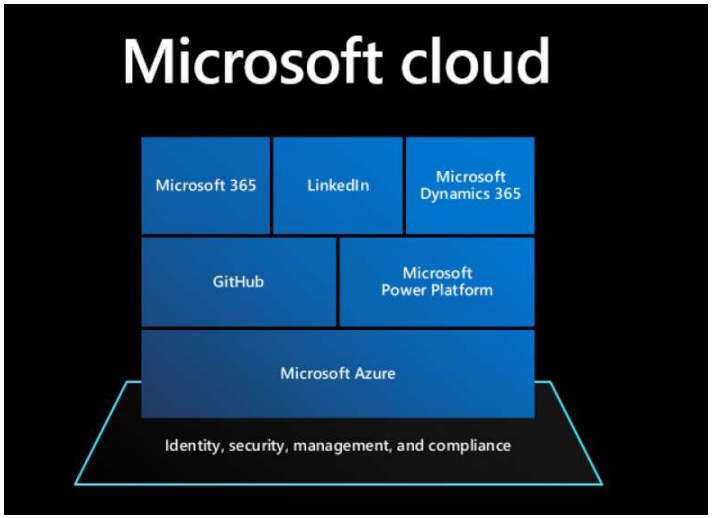
2.
내가 만든 버전 뿐만 아니라
동료가 만든 버전으로 이동할 수 있고,
동료와 내 버전을 **비교**해서
최신본으로 코드를 업데이트 할 수 있다.

Git

➢Git(버전이 관리되는 작업공간)은 어디에나 저장 가능함

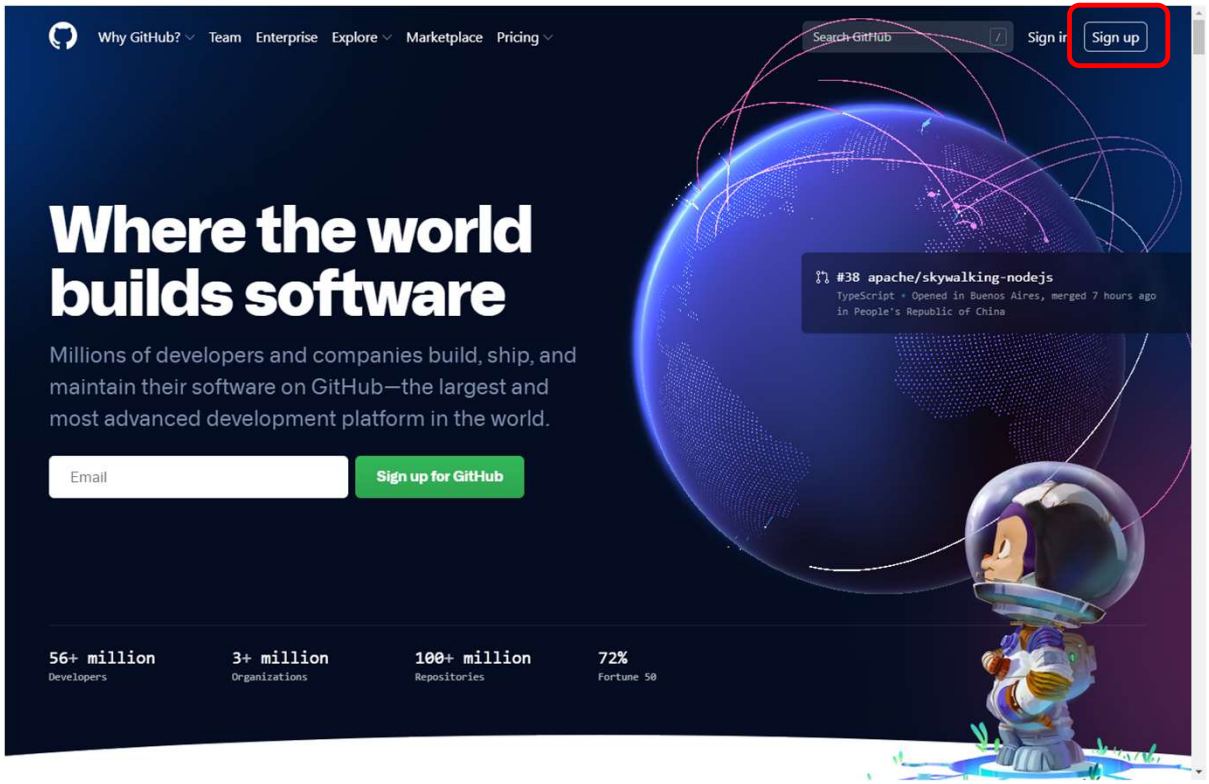
- USB
- PC
- 특정 서버
- 클라우드
- 웹 (**GitHub**, GitLab, BitButcket)





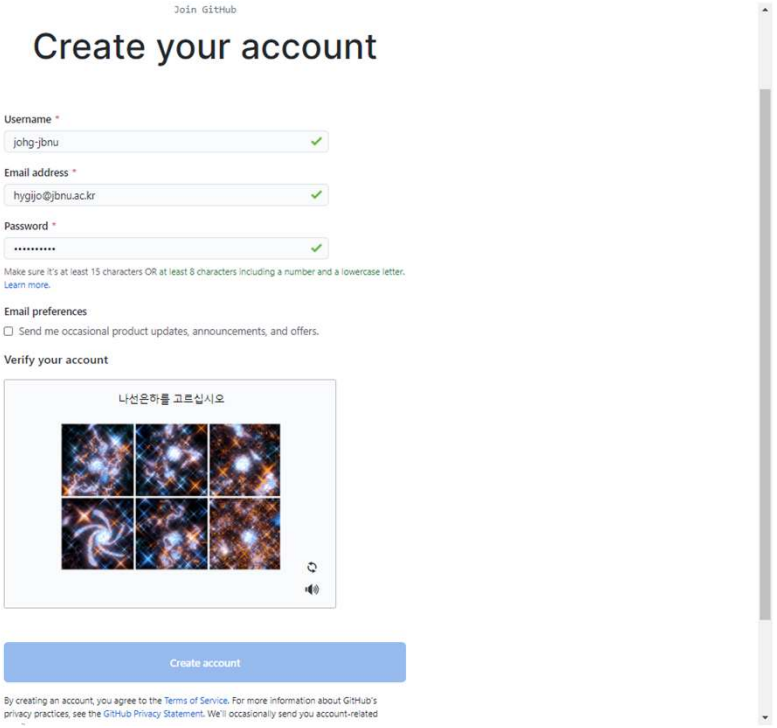
Sign up (GitHub)

1. GitHub 사이트 가입하기 https://github.com

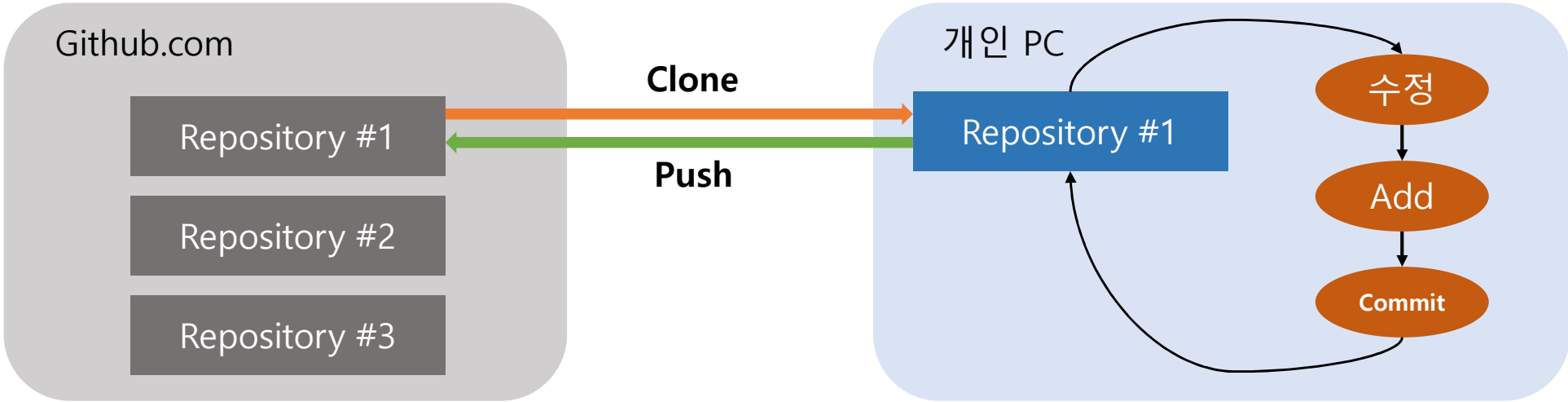


Sign up (GitHub)

2. Account 생성
   - **Email address, Username** – 본명(영문)으로
   - Email Verification 완료하기



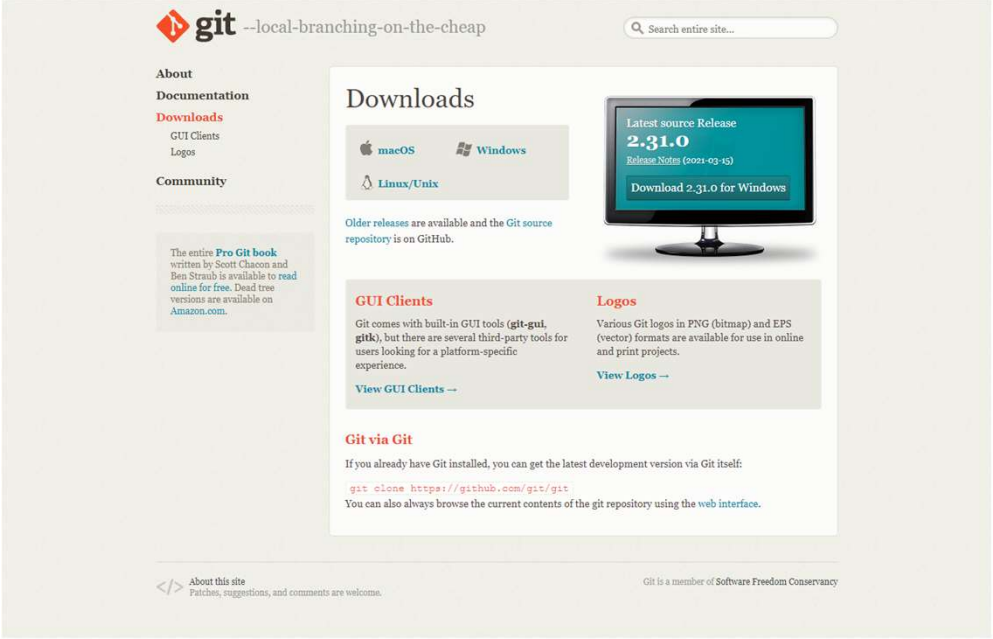
Git, GitHub


Github.com
Repository #1
Repository #2
Repository #3
Clone
Push
개인 PC
Repository #1
수정
Add
Commit

로컬에서 Git으로 관리하기

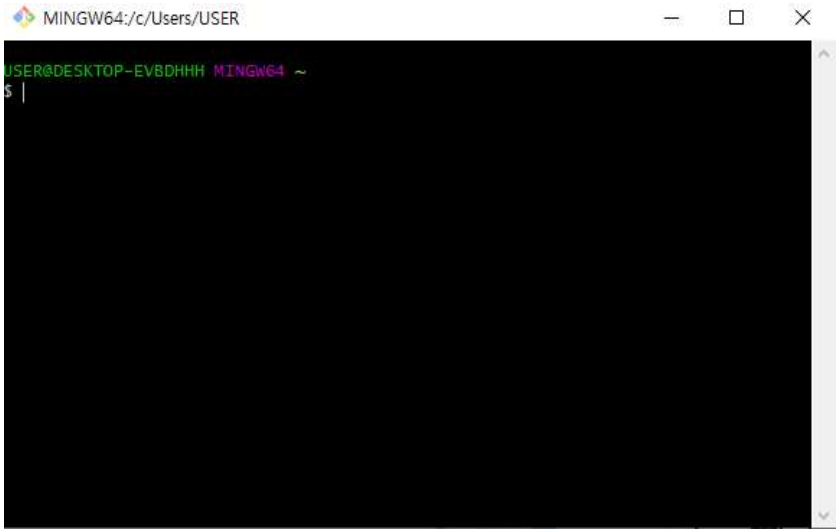
Install

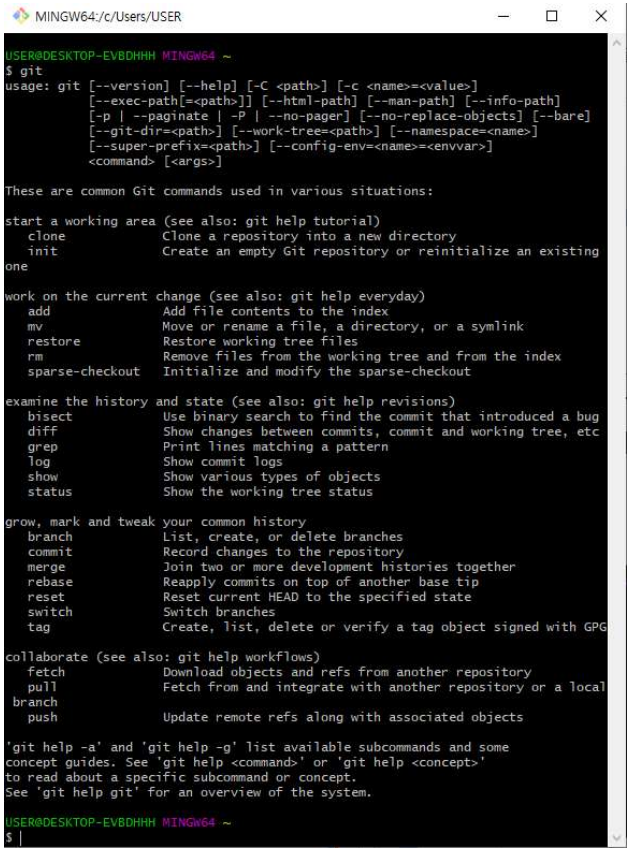
1. Git 다운로드 (윈도우 버전) https://git-scm.com/downloads
2. Git 설치하기
   - 설정 바꾸지 않고 Next, Install 진행



Install

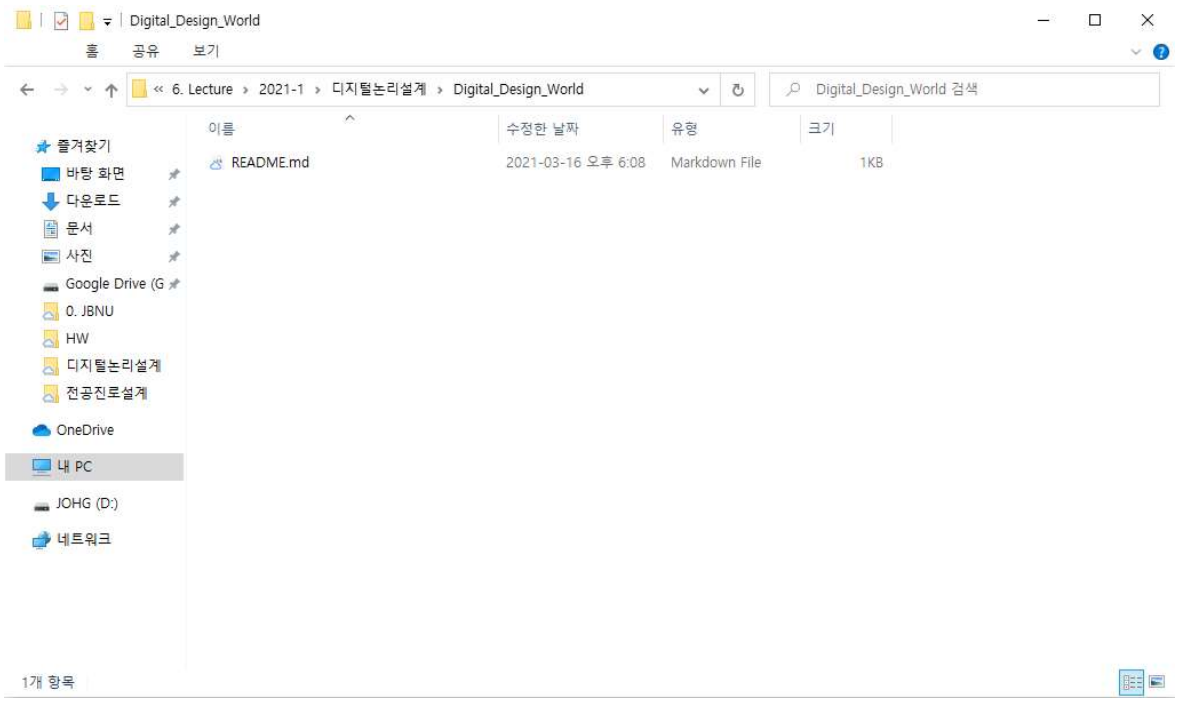
1. 윈도우 시작 버튼 옆 돋보기 버튼
2. “Git Bash” 입력
3. “$git” 입력

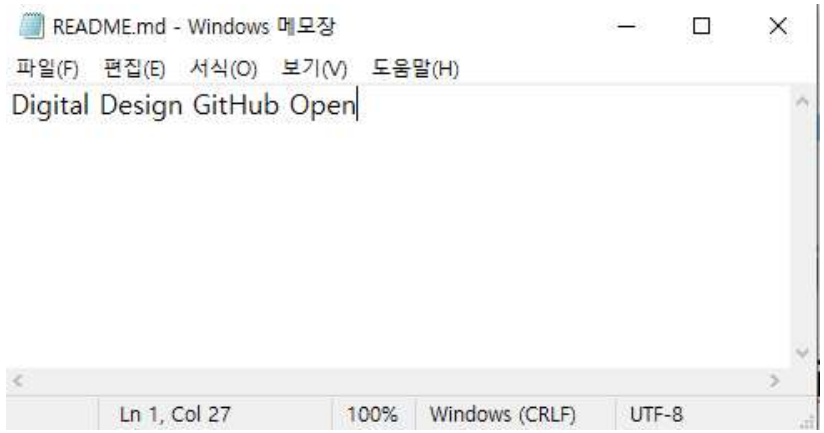




Sample Project - Init

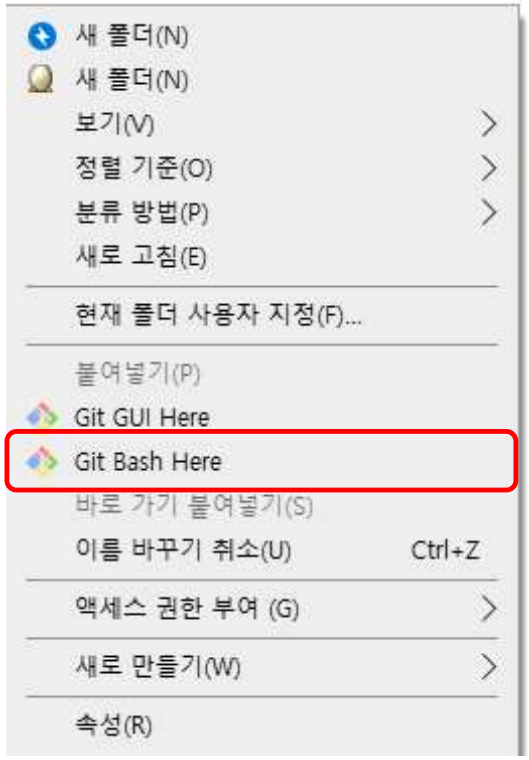
1. 작업 폴더 생성
2. 메모장으로 README.md 파일 생성 (아무내용이나)

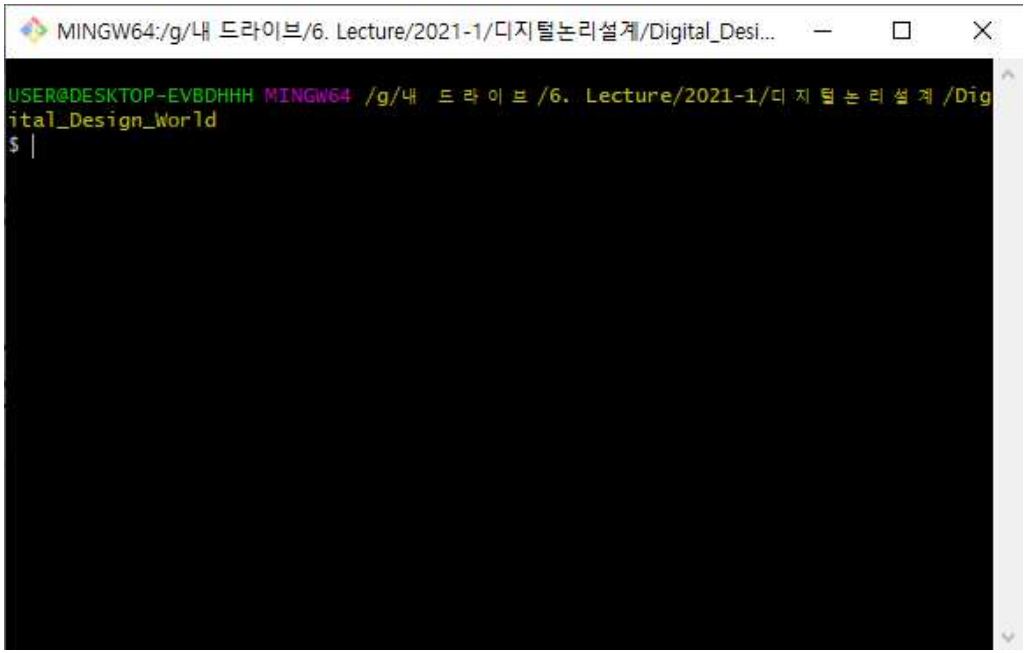




Sample Project - Init

3. 작업 폴더 안에서 오른쪽 클릭
4. 해당 폴더 경로의 터미널이 열림



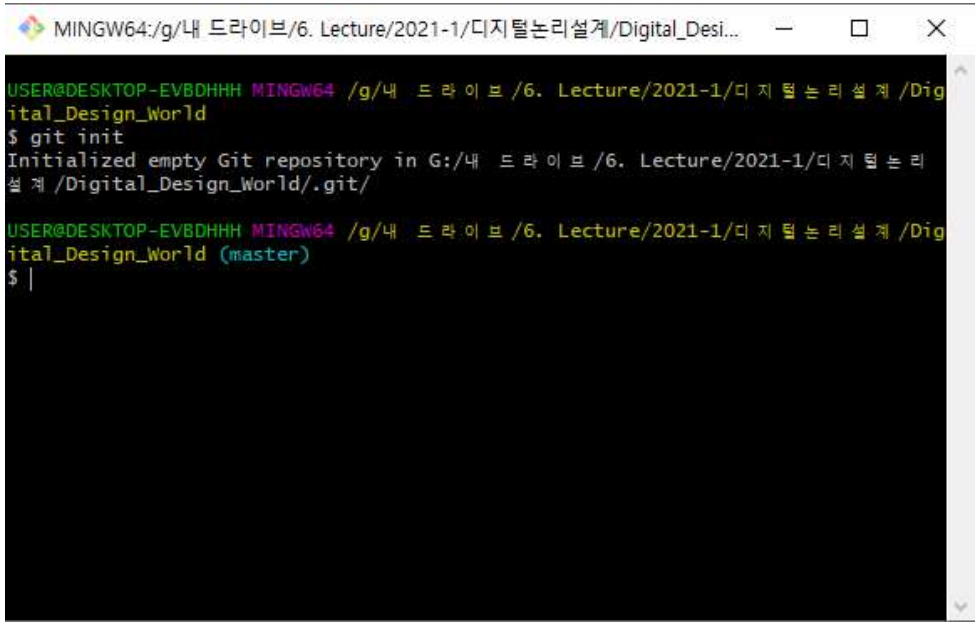


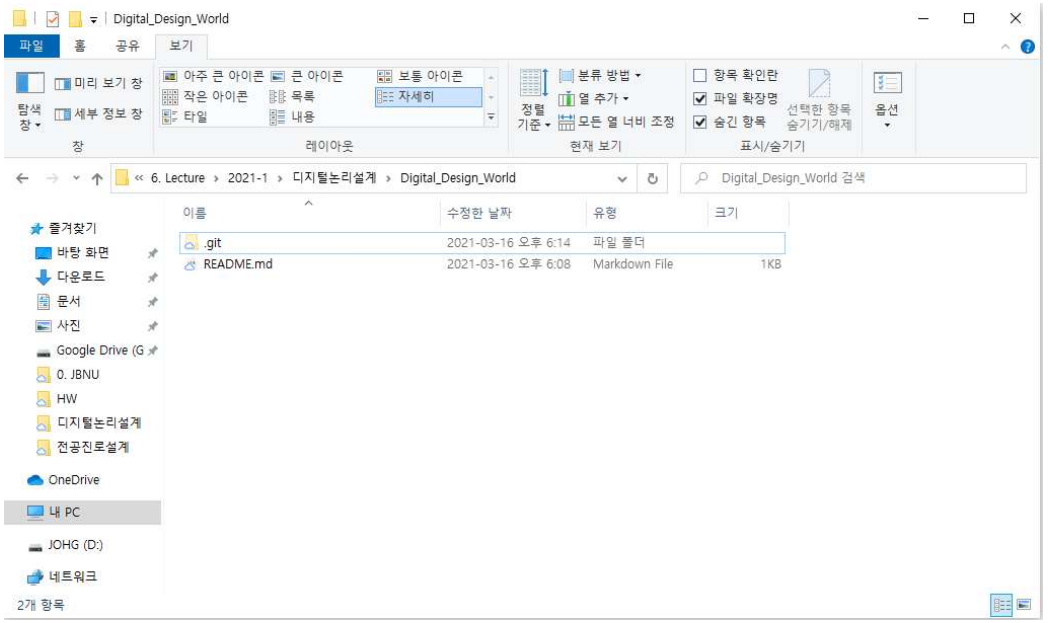
Sample Project - Init

5. Git 초기화

$ git init

Git Repository 생성됨 (.git 폴더가 생성)





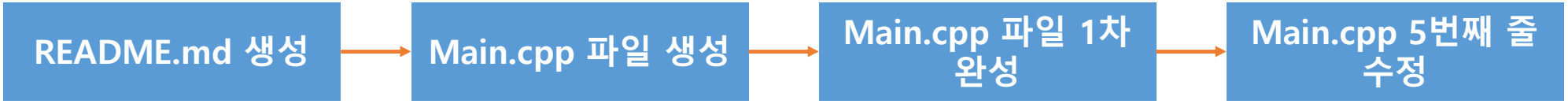
Commit 커밋

➢커밋이란?

- 하나의 변경 사항 단위
- Ex) xxx_yyy.cpp 5번째 줄 수정
- Ex) README.md 파일 생성

➢꾸준히 쌓이는 커밋

- 어떤 의미있는 변동사항이 있을 때마다 커밋을 하면 된다.
- $ git commit –m "어떤 변동사항이 있었는지 메시지 간단하게"



Sample Project

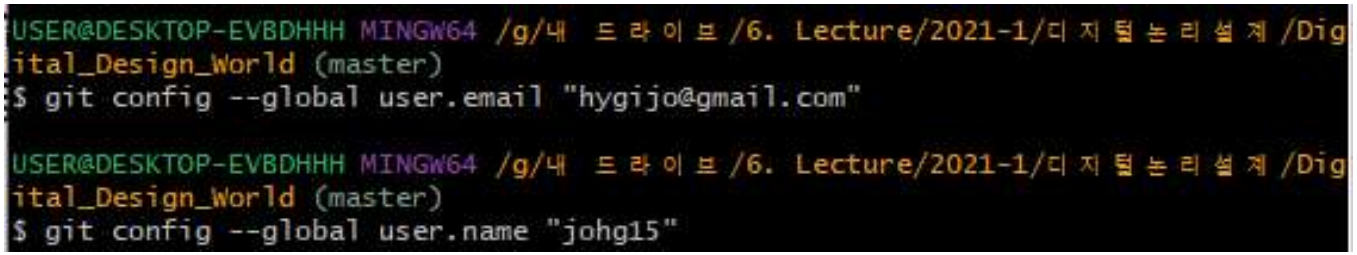
6. 커밋 Commit

- README.md 파일을 하나의 버전으로 저장
- RPG 게임에서 중간 저장이라고 생각

6-1. 계정 등록

- Email address, Username

$ git config --global user.email “your_email@gmail.com”
$ git config --global user.name “username”



Sample Project

6-2. 커밋할 파일 선택

- 특정 파일 선택 또는 전체를 지정할 수 있음

$ git add README.txt

$ git add .

```
USER@DESKTOP-EVBDHHH MINGW64 /g/내 드라이브/6. Lecture/2021-1/디지털논리설계/Dig
ital_Design_World (master)
$ git add .
```

6-3. 커밋하기

- 어떤 커밋인지 설명 메시지 간략하게 넣어서 커밋하기

$ git commit -m “readme 파일 생성”

```
USER@DESKTOP-EVBDHHH MINGW64 /g/내 드라이브/6. Lecture/2021-1/디지털논리설계/Dig
ital_Design_World (master)
$ git commit -m "readme 파일 생성"
[master (root-commit) 003de72] readme 파일 생성
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
```

Sample Project

6-4. 두 번째 커밋하기

- 파일 추가하고 커밋해보기

$ git add .

$ git commit –m “second file added”

```
USER@DESKTOP-EVBDHHH MINGW64 /g/내  드 라 이 브 /6. Lecture/2021-1/디 지 털 논 리 설 계 /Dig
ital_Design_World (master)
$ git add .

USER@DESKTOP-EVBDHHH MINGW64 /g/내  드 라 이 브 /6. Lecture/2021-1/디 지 털 논 리 설 계 /Dig
ital_Design_World (master)
$ git commit -m "second file add"
[master 8e67e13] second file add
 1 file changed, 76 insertions(+)
 create mode 100644 AND_gate.circ
```

Sample Project

6-5. 시간 여행

- 지금까지 커밋 확인
- 첫 번째 커밋 ID 앞 7자리 확인

$ git log

```
USER@DESKTOP-EVBDHHH MINGW64 /g/내 드라이브/6. Lecture/2021-1/디지털논리설계/Dig
ital_Design_World (master)
$ git log
commit 8e67e137cb53b7133060be4a43d91fe127df99ba (HEAD -> master)
Author: johg15 <hygijo@gmail.com>
Date:   Tue Mar 16 18:46:41 2021 +0900

    second file add

commit 003de7203452c108cf764b7e57aa710943923ce6
Author: johg15 <hygijo@gmail.com>
Date:   Tue Mar 16 18:41:04 2021 +0900

    readme 파일생성
```

Sample Project

6-6. 시간 여행

- 첫 번째 커밋으로 돌아가기

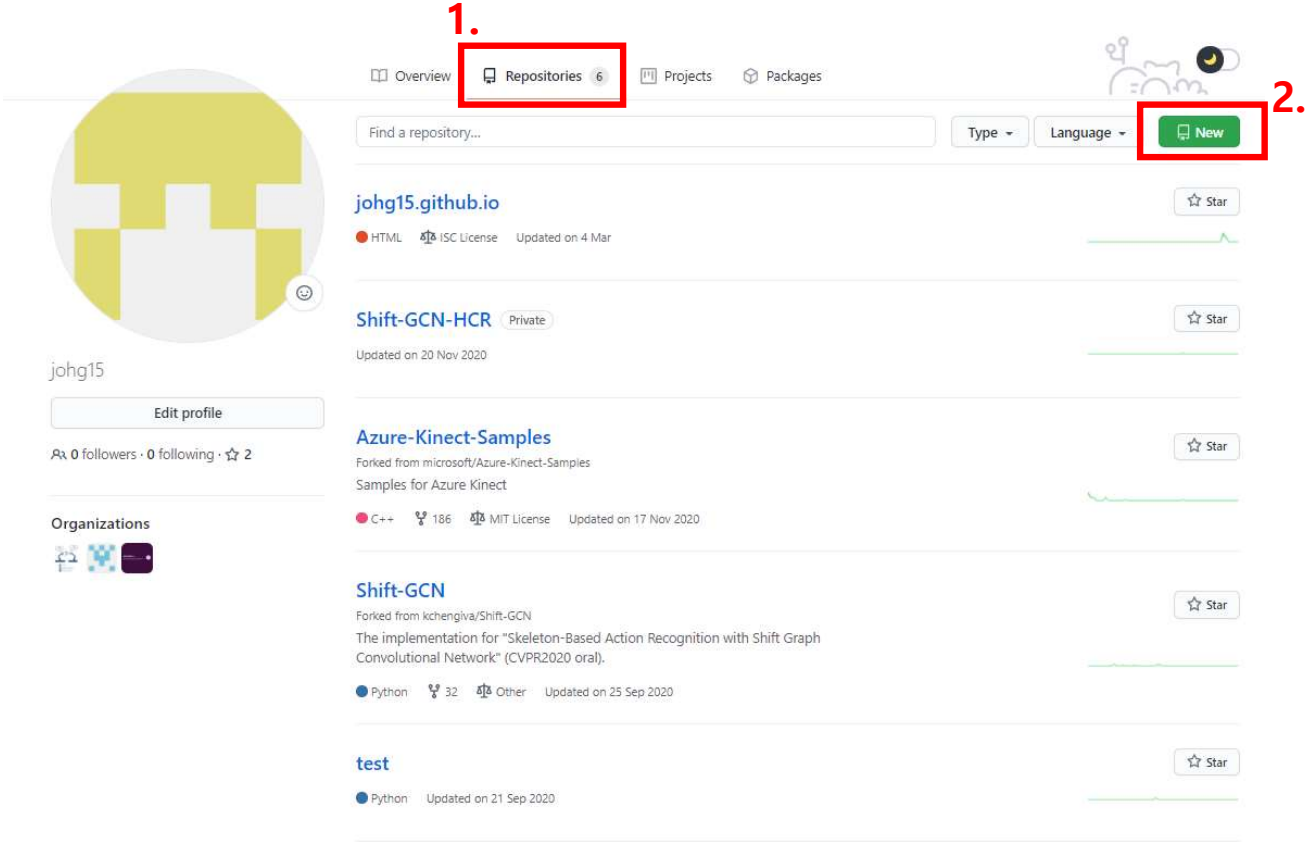
$ git checkout 003de720

```
USER@DESKTOP-EVBDHHH MINGW64 /g/내 드라이브/6. Lecture/2021-1/디지털논리설계/Dig
ital_Design_World (master)
$ git checkout 003de720
Note: switching to '003de720'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 003de72 readme 파일생성
```

GitHub 원격저장소

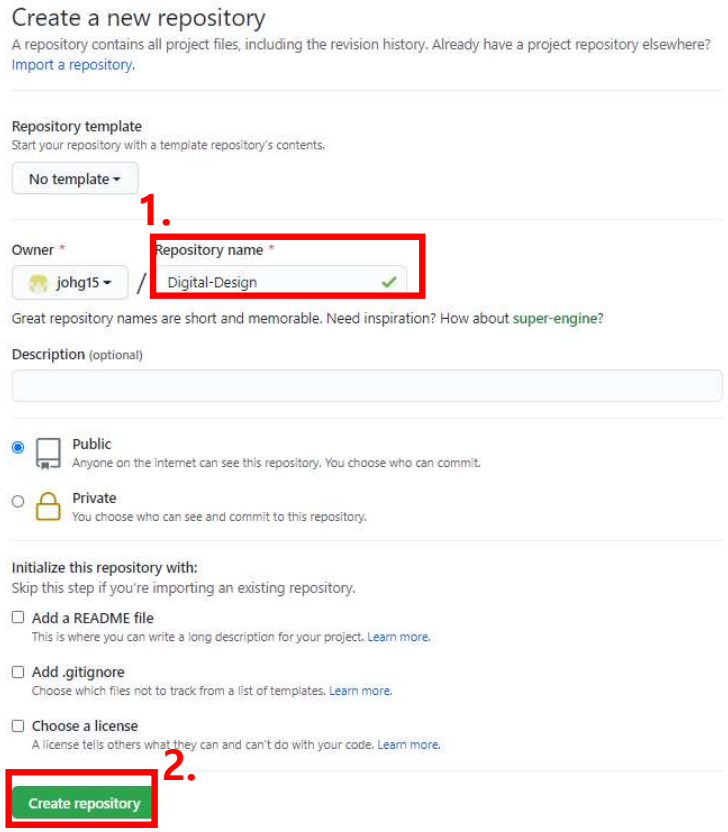
GitHub 원격저장소

➢Repository 생성



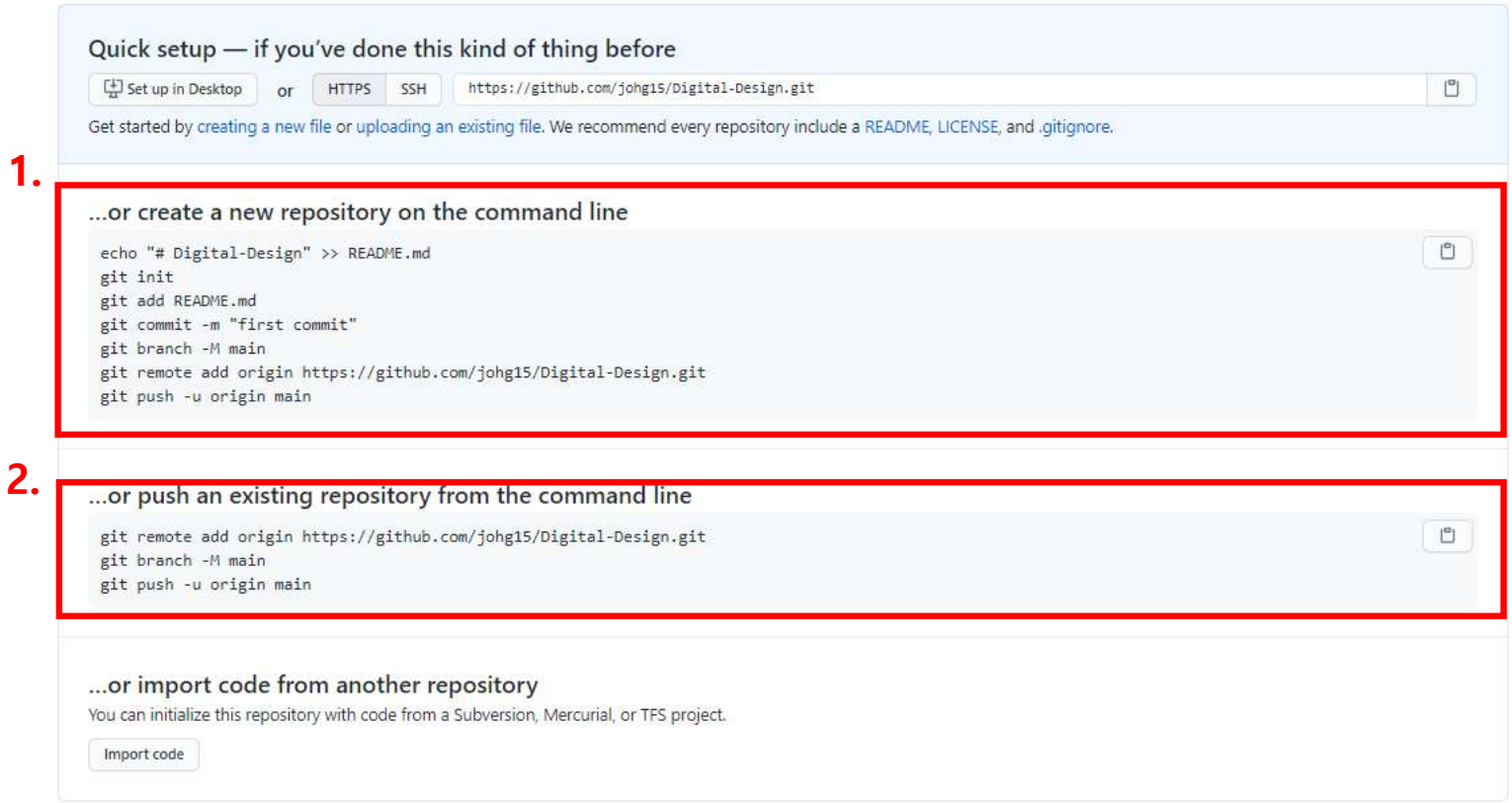
GitHub 원격저장소

➢Repository 이름



GitHub 원격저장소

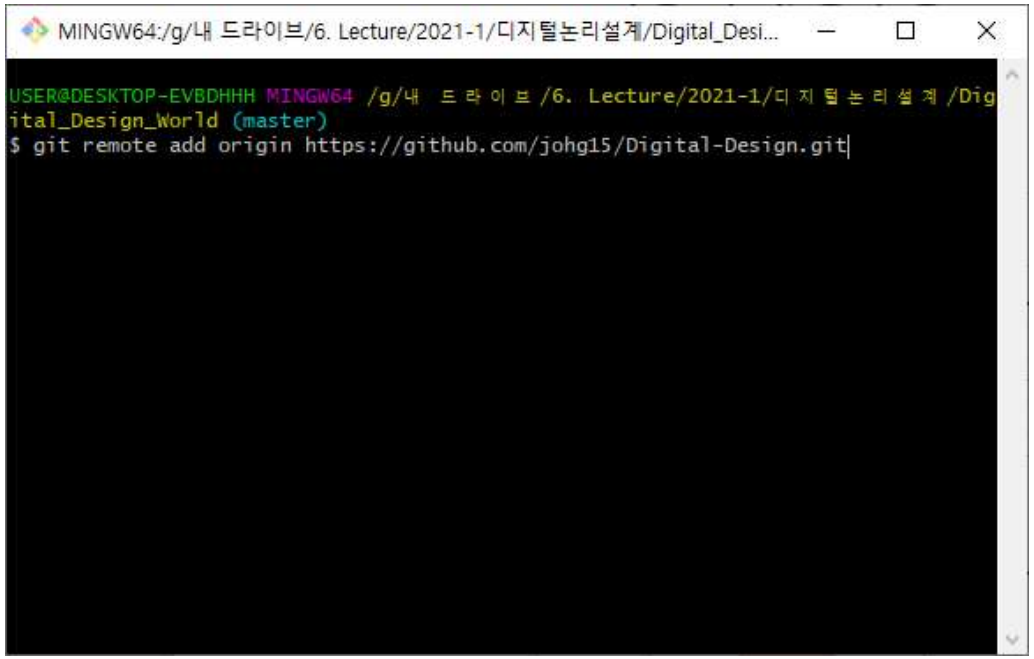
➢Quick Setup



GitHub 원격저장소

➢Quick Setup

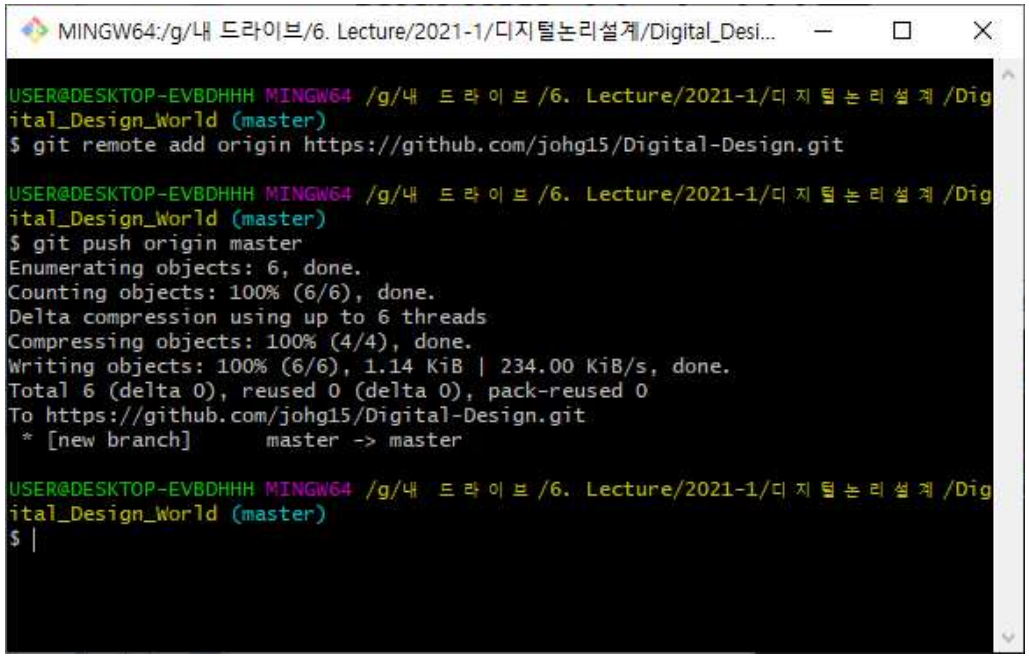
- 로컬과 원격저장소 연결
- $git remote add origin https://github.com/(your_username)/(Repository_name).git



GitHub 원격저장소

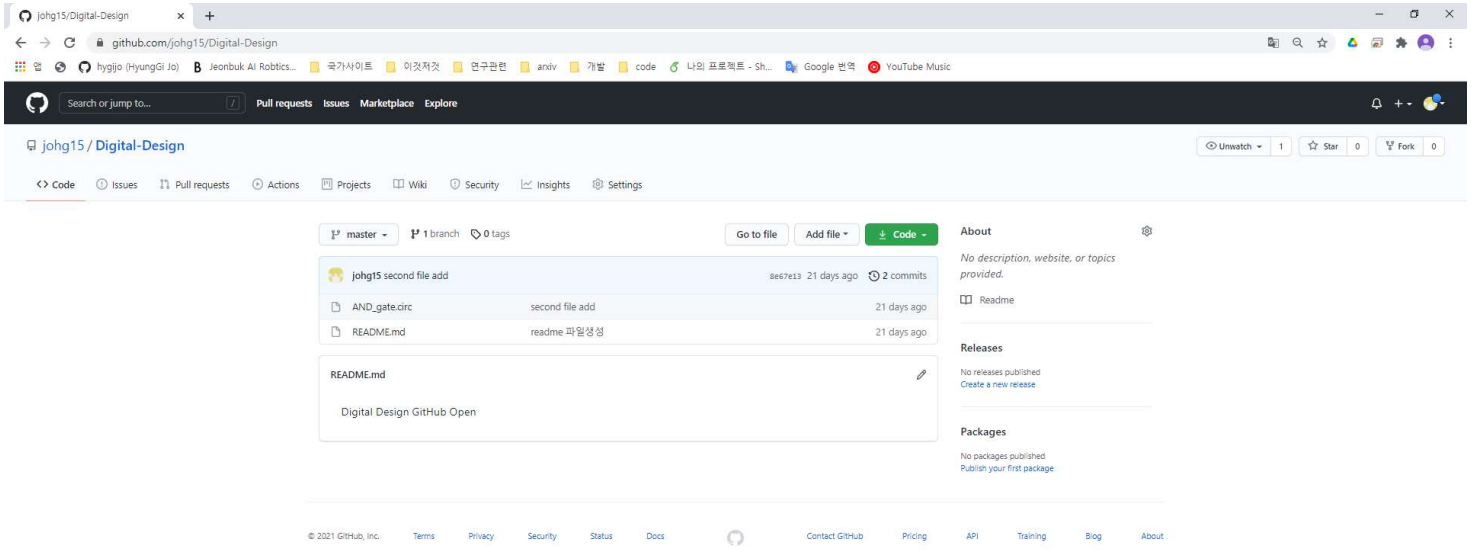
➢Quick Setup

- 로컬 저장소의 커밋을 push 명령어로 원격저장소에 올리기
- $git push origin master



GitHub 원격저장소

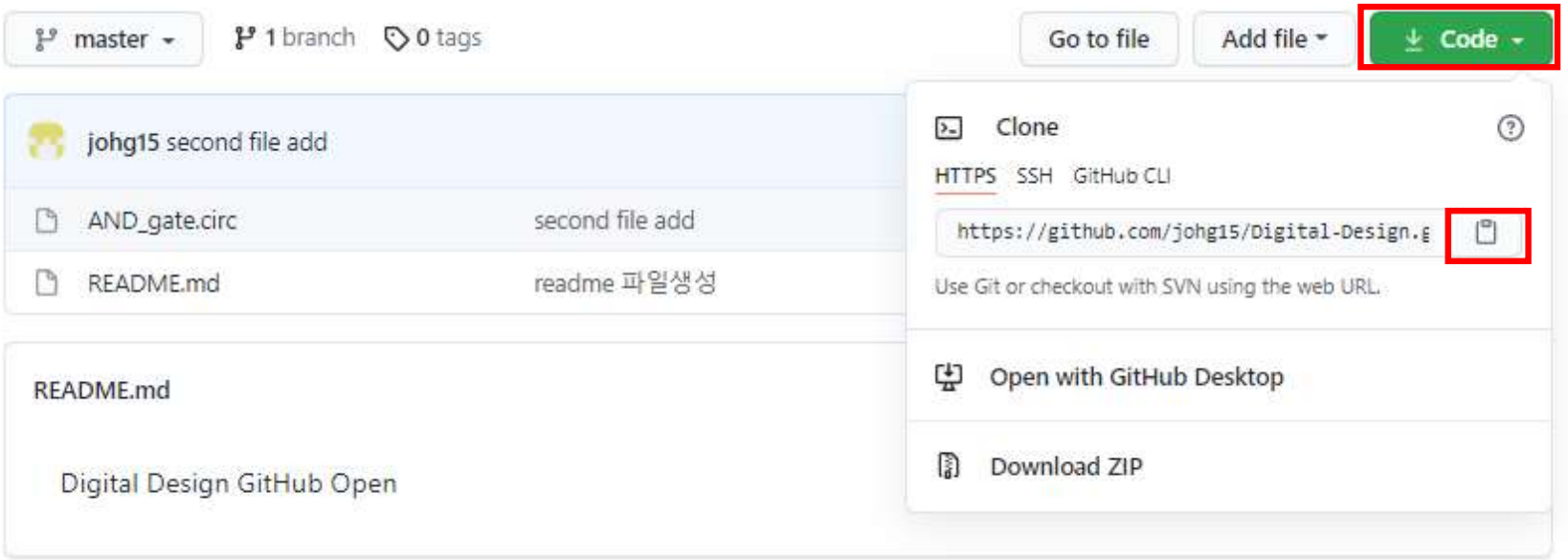
- Quick Setup
  - GitHub에서 잘 올라왔는지 확인



GitHub 내려받기

➢원격저장소에서 내려받기 (Clone)

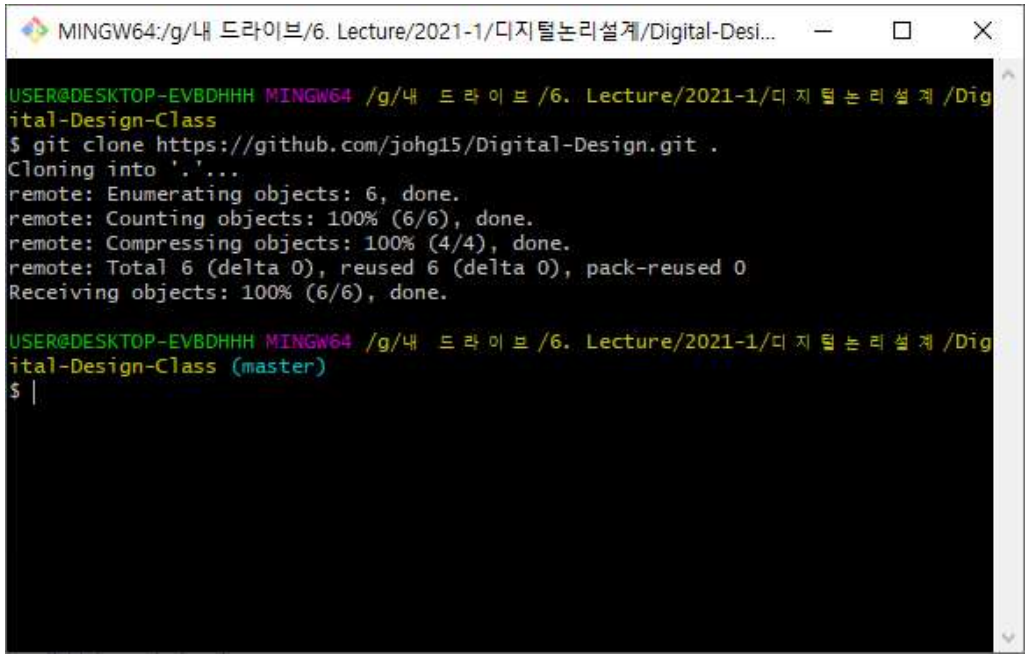
- 다른 새로운 폴더 생성 (Creative-Eng-Class) → Git Bash Here



GitHub 내려받기

➢원격저장소에서 내려받기 (Clone)

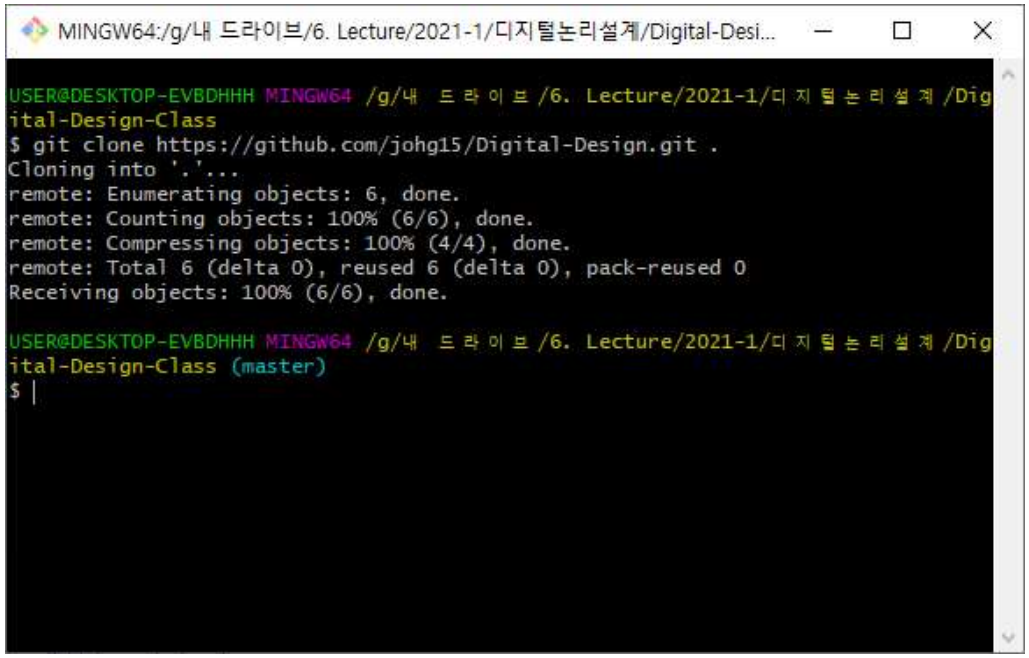
- 다른 새로운 폴더 생성 (Creative-Eng-Class) → Git Bash Here



GitHub 내려받기

➢원격저장소에서 내려받기 (Clone)

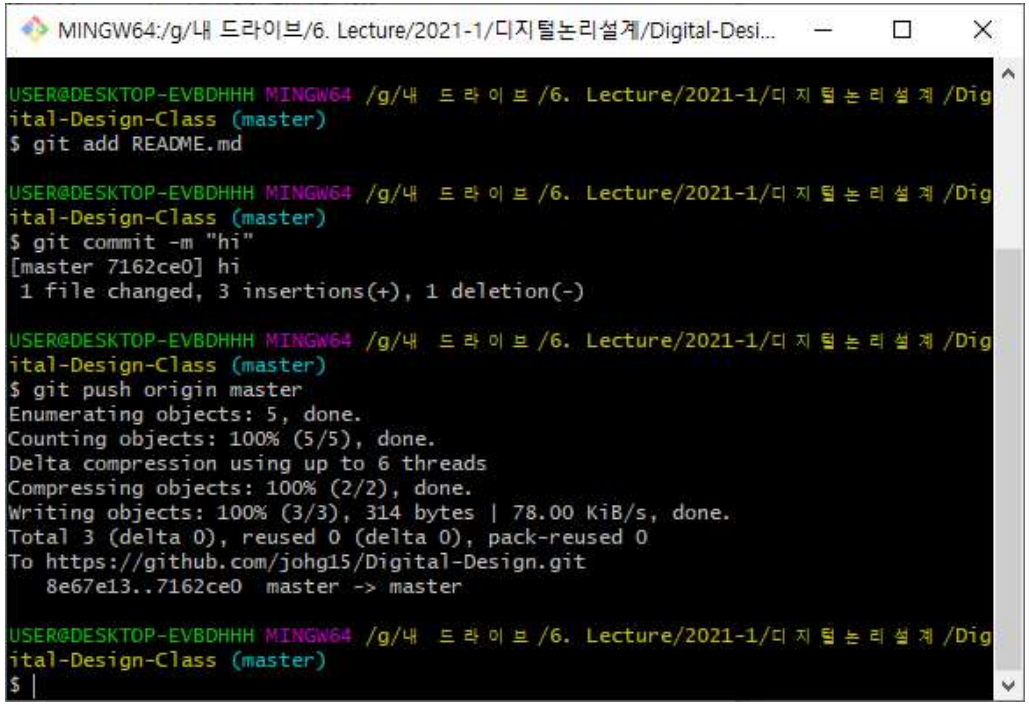
- $git clone (복사한 주소) .
- 내려받기 확인



GitHub 올리기

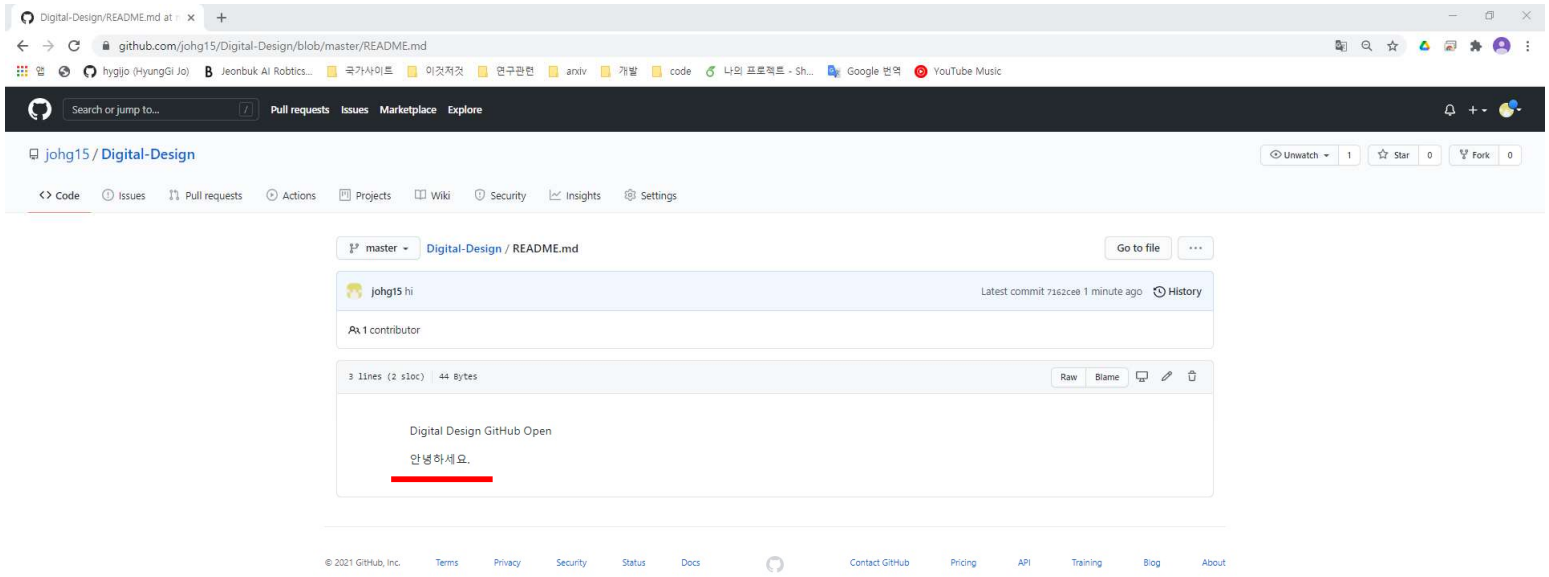
➢로컬저장소에서 수정 후 올리기

- $git add README.md
- $git commit –m “hi”
- $git push origin master



GitHub 올리기

➢로컬저장소에서 수정 후 올리기



GitHub 다시 내려받기

➢수정된 사항 다시 내려받기

- 작업 폴더에서 다시 내려받기
- $git pull origin master