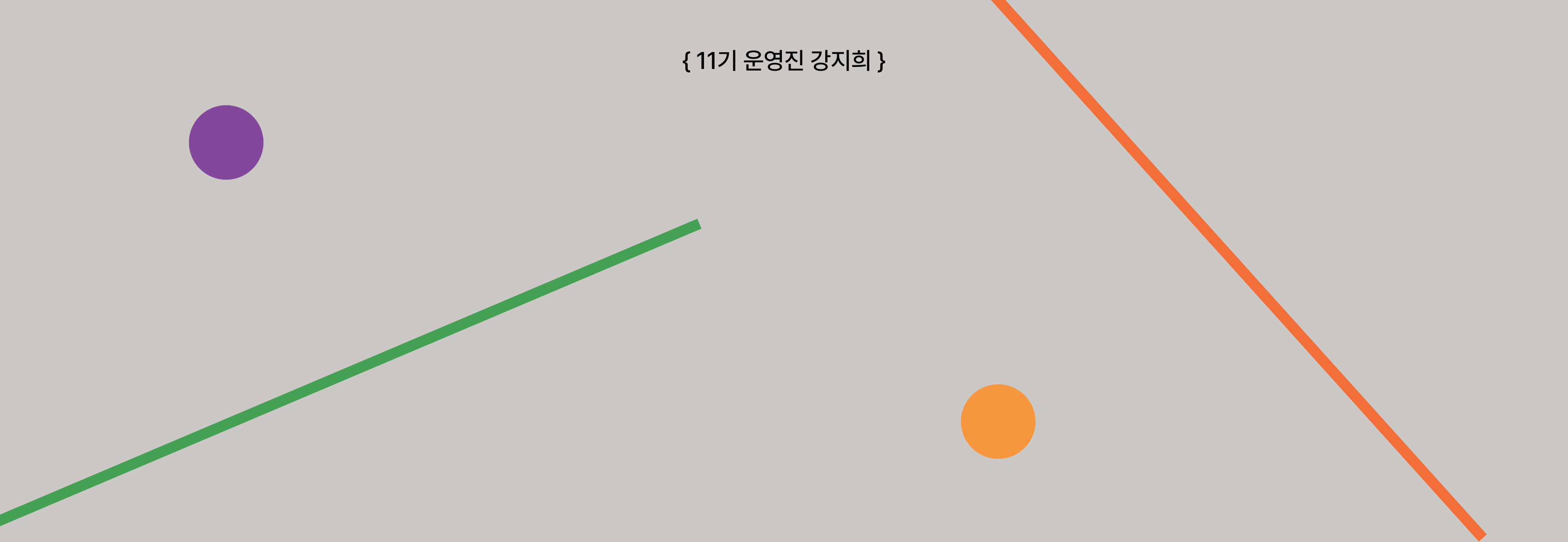


{ 11기 운영진 강지희 }



(GIT & FIGMA)

ABOUT & WORKS

BRADING & GRAPHIC

1) Homebrew & git 설치 ●

¹ 터미널을 실행합니다.

² 아래의 코드를 입력해 homebrew를 설치합니다.

- M1 노트북 :

1) /bin/bash -c "\$(curl -fsSL https://gist.githubusercontent.com/nrubin29/bea5aa83e8dfa91370fe83b62dad6dfa/raw/48f48f7fef21abb308e129a80b3214c2538fc611/homebrew_m1.sh)"

2) eval \$(/opt/homebrew/bin/brew shellenv)

3) brew --version

*3)번 명령어 입력 후 버전이 제대로 나온다면 설치성공

- 다른 맥 :

1) /bin/bash -c "\$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

2) brew --version

*2)번 명령어 입력 후 버전이 제대로 나온다면 설치성공

³ 아래의 코드를 입력해 git을 설치합니다

brew install git

1) Homebrew & git 설치 예시 ●

```
gimga-eun@gimga-eun-ui-MacBookAir ~ % /bin/bash -c "$(curl -fsSL https://gist.githubusercontent.com/nrubin29/bea5aa83e8dfa91370fe83b62dad6dfa/raw/48f48f7fef21abb308e129a80b3214c2538fc611/homebrew_m1.sh)"
Password: * 비밀번호 입력 시 화면에 입력한 비밀번호가 안보임

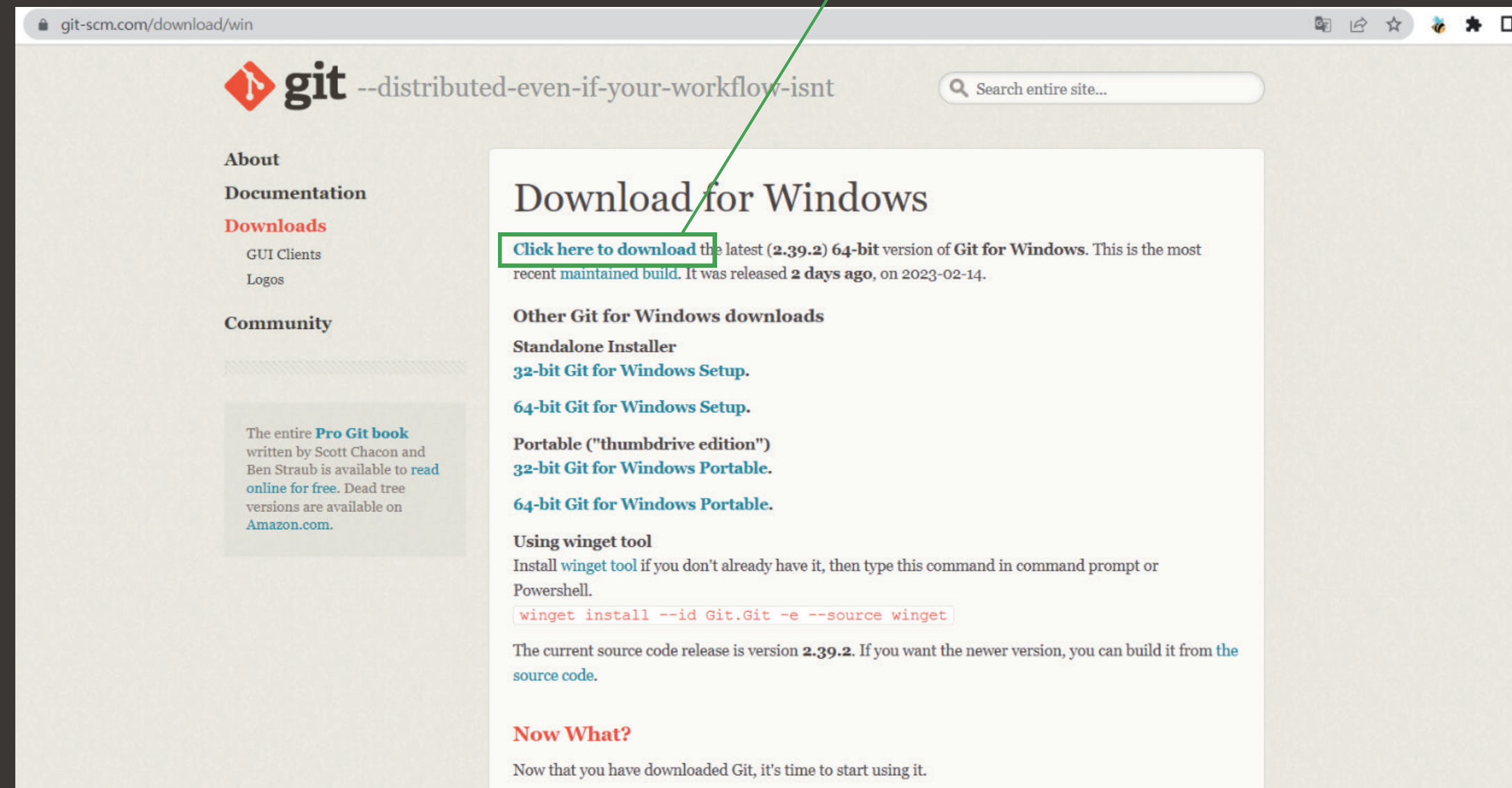
gimga-eun@gimga-eun-ui-MacBookAir ~ % eval $(/opt/homebrew/bin/brew shellenv)
[gimga-eun@gimga-eun-ui-MacBookAir ~ % brew install git
[hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /opt/homebrew/.git/
```


(01.) GIT 설치(WINDOWS)

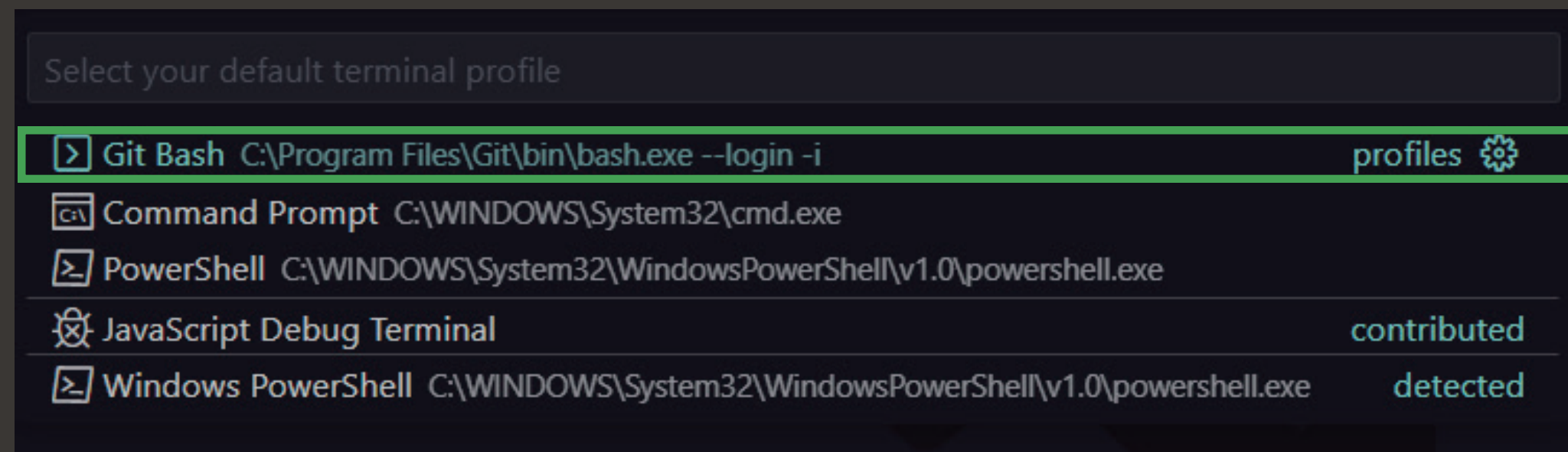
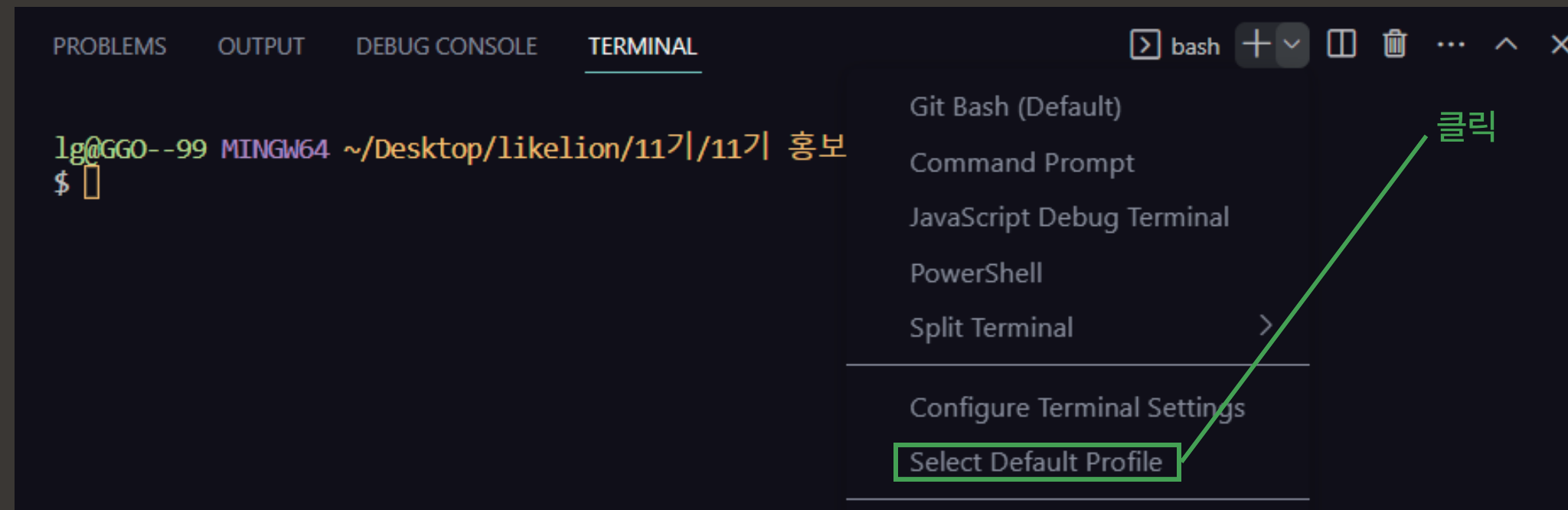
1) 아래 링크에서 다운로드 ●

<https://git-scm.com/download/win>

클릭



(01.) GIT 설치(WINDOWS)



Git은 소규모 프로젝트부터 대규모 프로젝트까지 모든 것을
빠르고 효율적으로 처리하도록 설계된 무료 오픈 소스
분산 버전 제어 시스템●

버전관리 시스템

버전관리 시스템은 파일 변화를 시간에 따라 기록했다가
나중에 특정 시점의 버전을 다시 꺼내올 수 있는 시스템

로컬 버전 관리

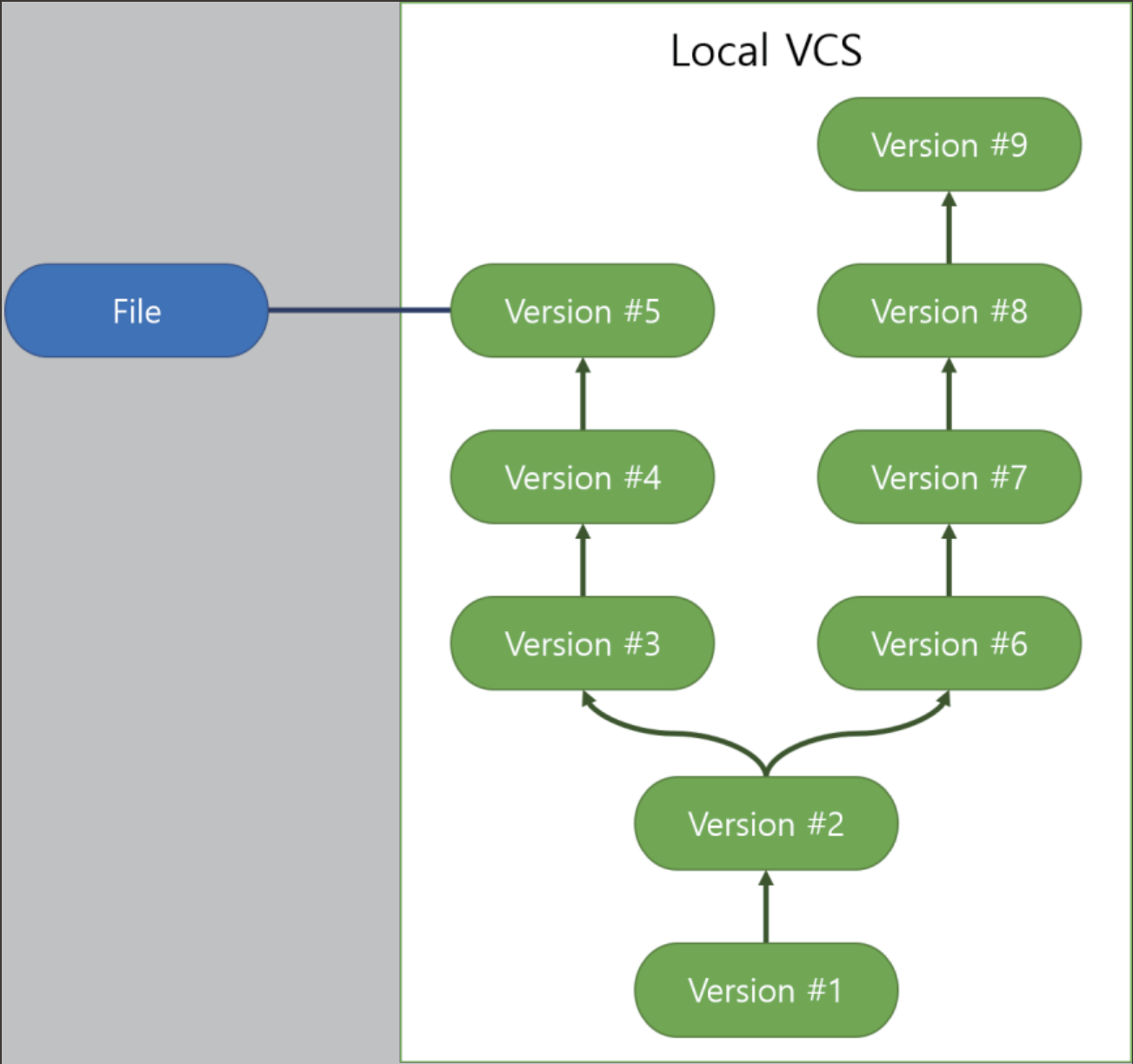
중앙집중식 버전 관리

분산 버전 관리

로컬 버전 관리

로컬에 간단한 데이터베이스를 이용해 파일의 이력(변경 정보)를 관리하는 시스템

단점 : 프로젝트 진행 시 협력 불가

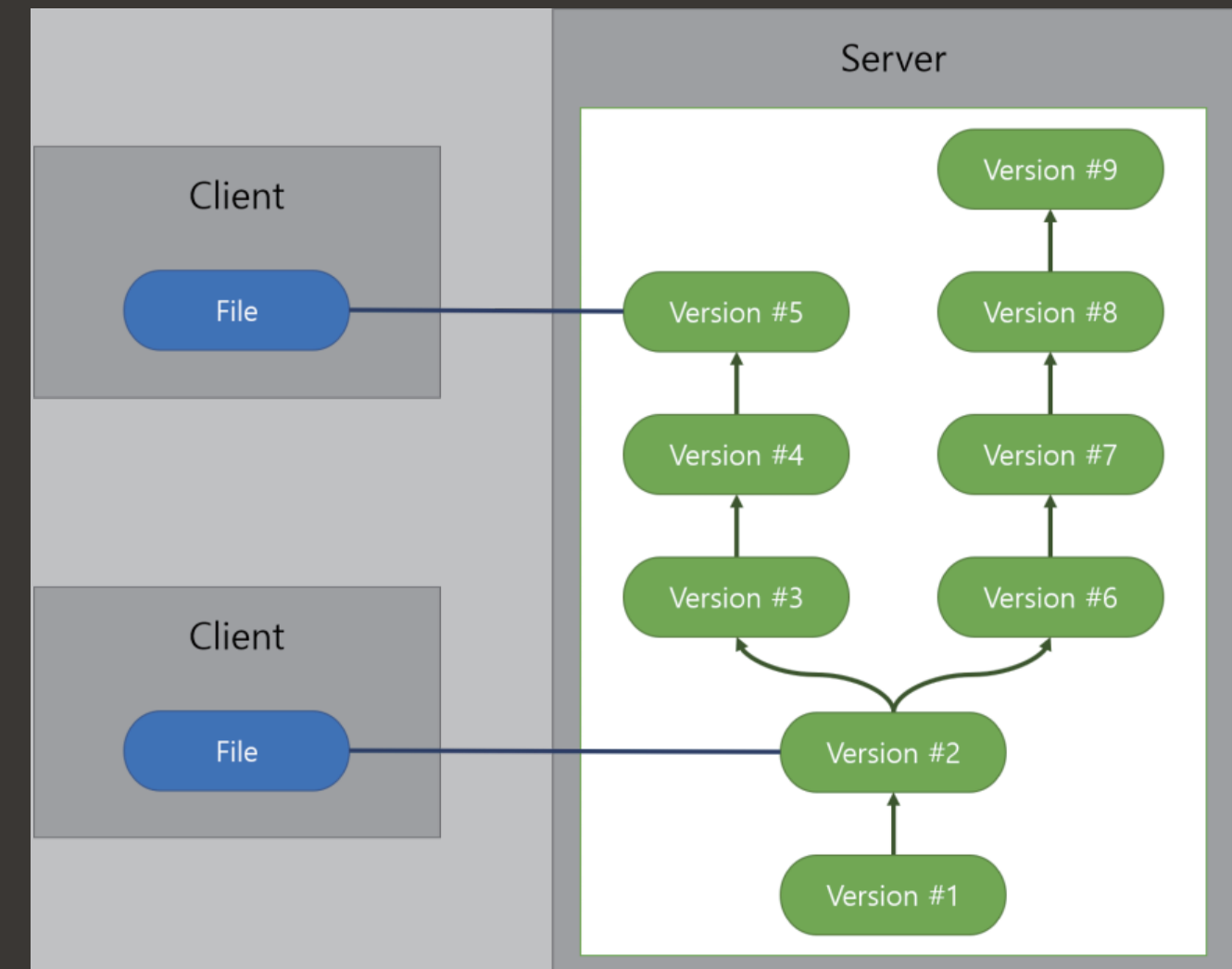


중앙집중식 버전 관리

중앙의 서버가 파일들과 이들의 변경 이력을 관리하고
각 클라이언트가 중앙 서버에서 파일을 받아서 사용

단점

- 중앙 서버에서 문제가 생기면 다른 사람과 협력할 수 없으며 백업도 불가
- 중앙 데이터베이스가 있는 하드디스크에 문제가 생기면 프로젝트의 모든 히스토리를 잃음

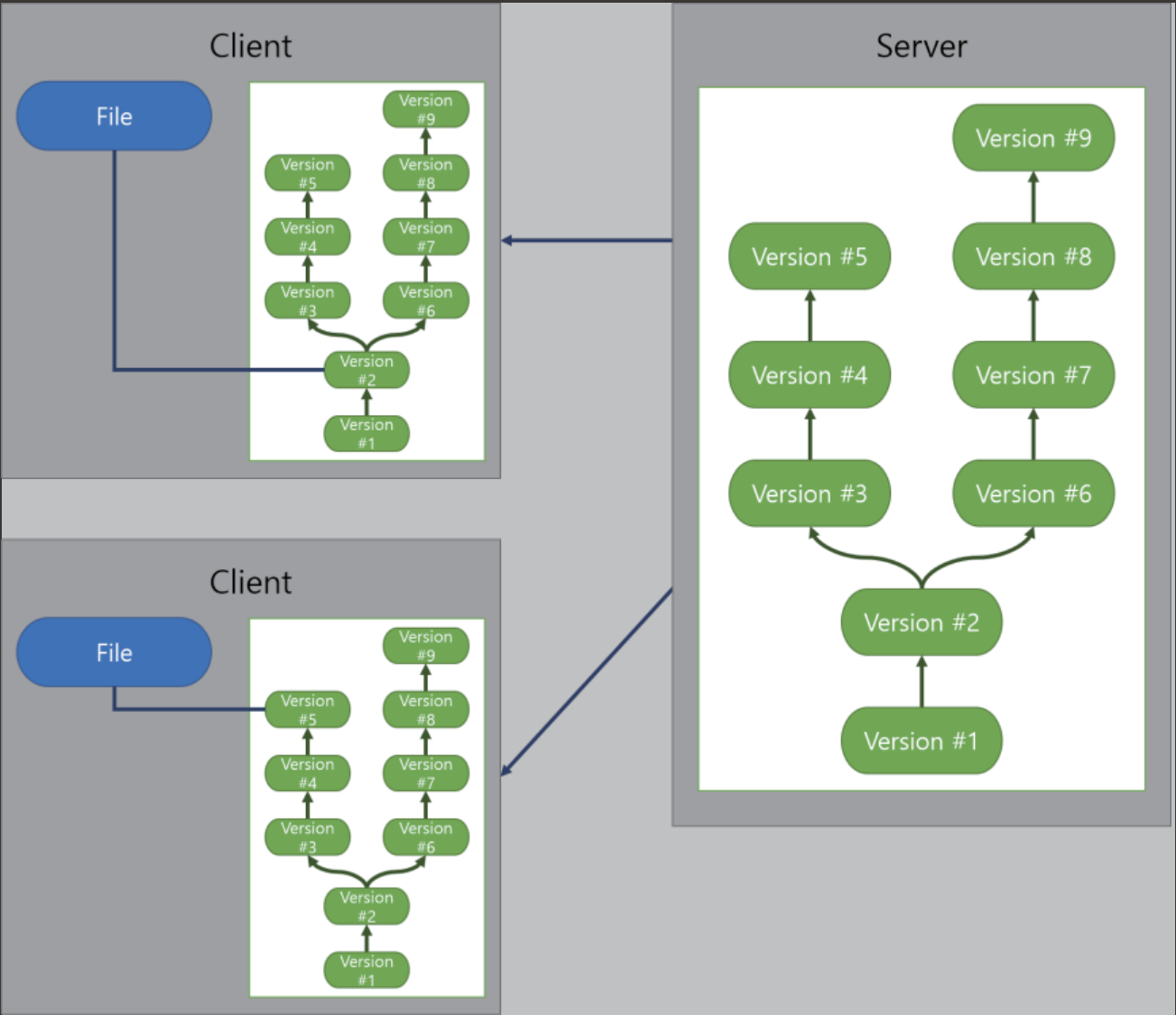


분산 버전 관리

클라이언트가 단순히 파일만을 받아오는 것이 아니라
저장소를 히스토리와 더불어 전부 복제 및 관리가 가능

리모트 저장소가 존재해 동시에 다양한 그룹 및 사람들과
협업 가능

* 리모트 저장소 : 인터넷 혹은 네트워크 어딘가에 존재하는 저장소



Git branch

그 시점 별로 존재하는 버전
브랜치 생성 시, 그 시점 내용을 복제하여 새로운 버전을
만들 수 있음.



Git branch 관련 명령어

branch 생성 : git branch 브랜치명

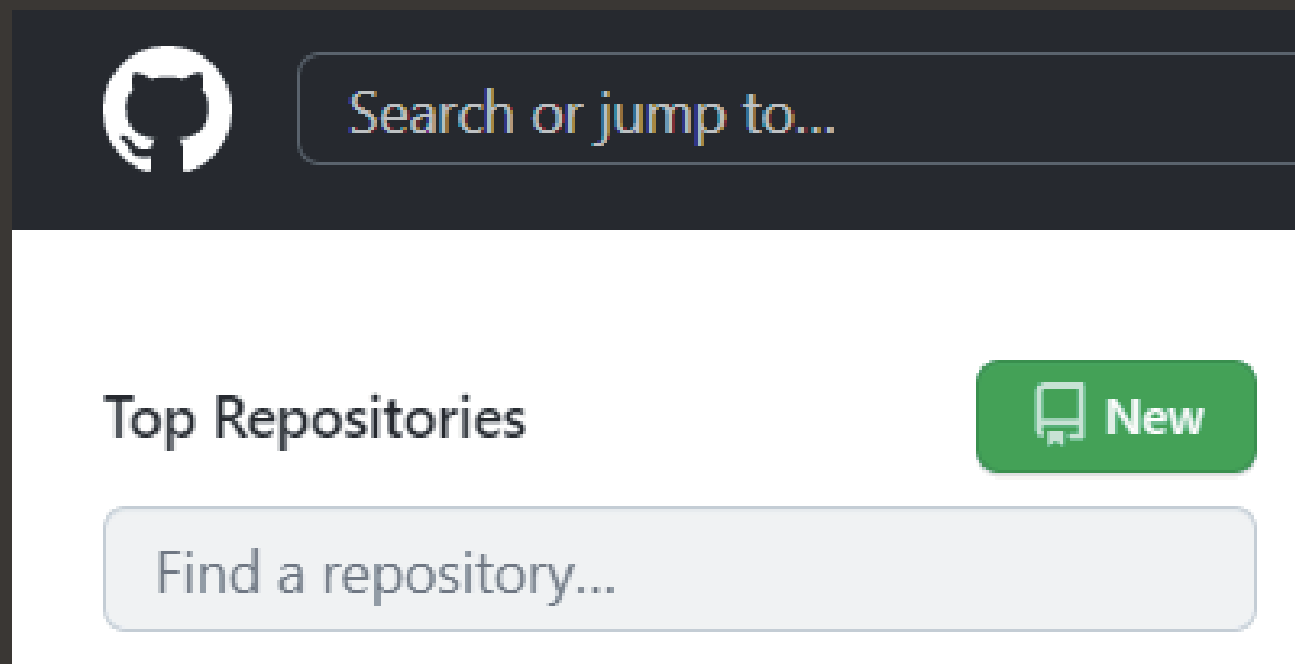
branch 이동 : git checkout 브랜치명

branch 삭제 : git branch -D 브랜치명

1. 깃허브 로그인, 없다면 회원가입 진행

<https://github.com>

2. new 버튼 클릭해 새로운 Repositories 생성

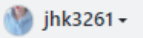


3. Repository 이름 작성

Create a new repository


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


Owner * Repository name *

 jkh3261 /

Great repository names are short and memorable. Need inspiration? How about [fluffy-octo-waffle?](#)

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.](#)

Add .gitignore
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.](#)

📘 You are creating a public repository in your personal account.

[Create repository](#)

 **jkh3261 / gitsession1** Public

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

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

[Set up in Desktop](#) or [HTTPS](#) [SSH](#) [Copy](#)

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

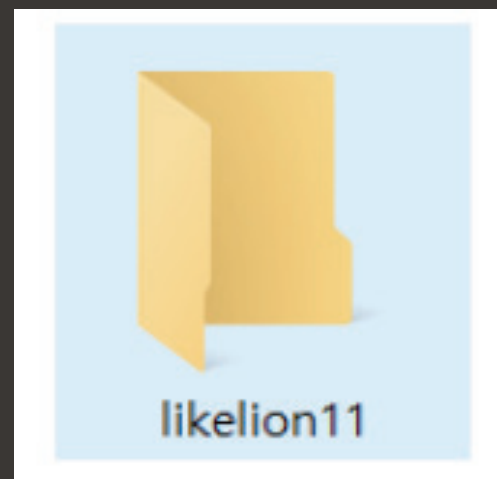
...or create a new repository on the command line

```
echo "# gitsession1" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/jkh3261/gitsession1.git
git push -u origin main
```

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

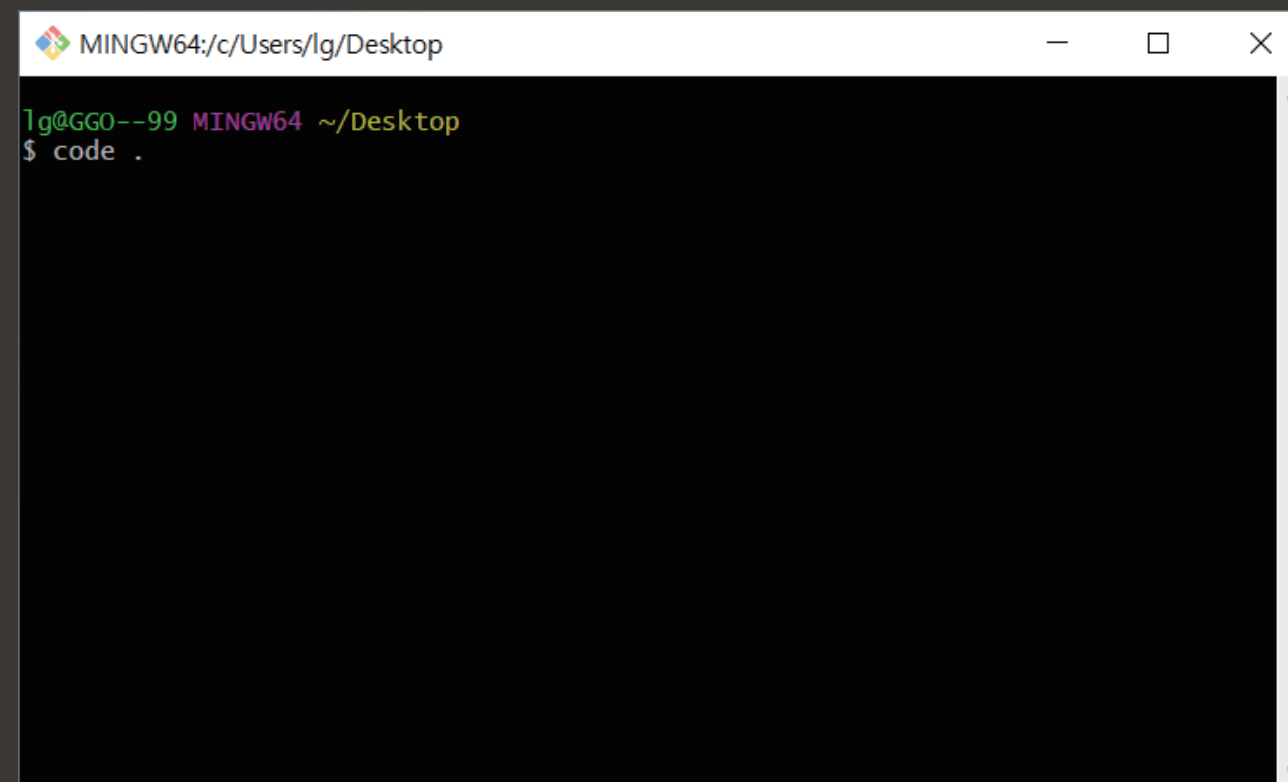
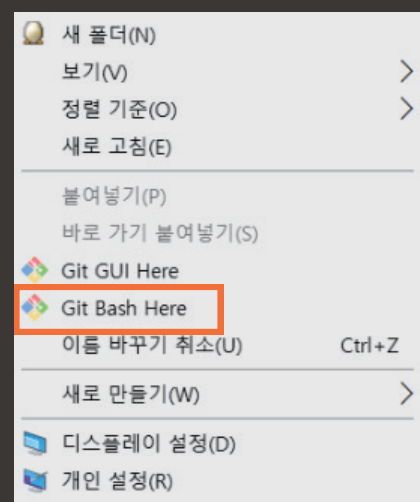
```
git remote add origin https://github.com/jkh3261/gitsession1.git
git branch -M main
git push -u origin main
```


4. 원하는 곳에 새로운 폴더 생성



Vscode 실행

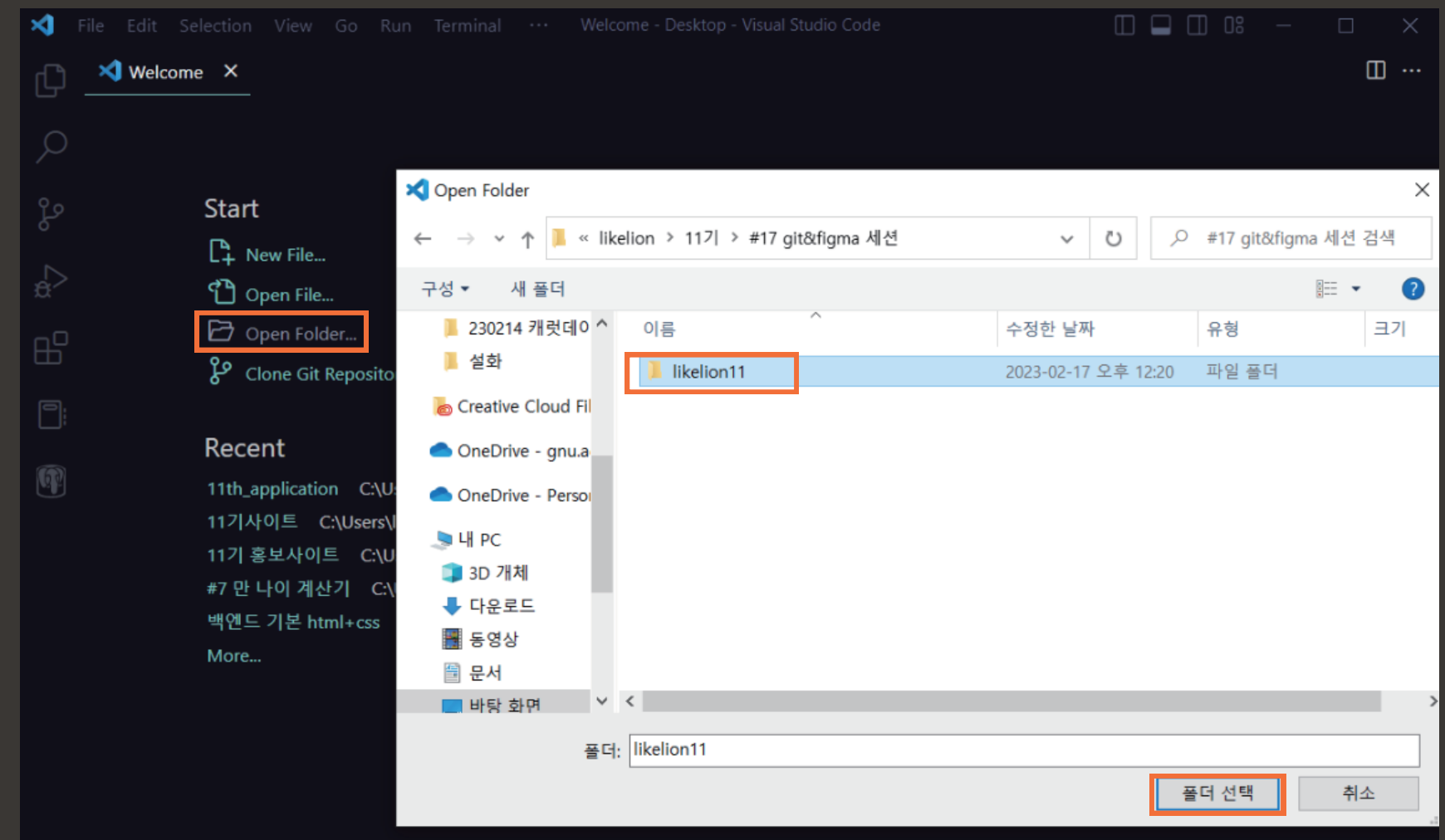
1) 윈도우 경우 -> 오른쪽 마우스 클릭 후 git bash here 클릭 -> code . 명령어 입력해서 실행



2) 맥북은 vscode 앱 실행

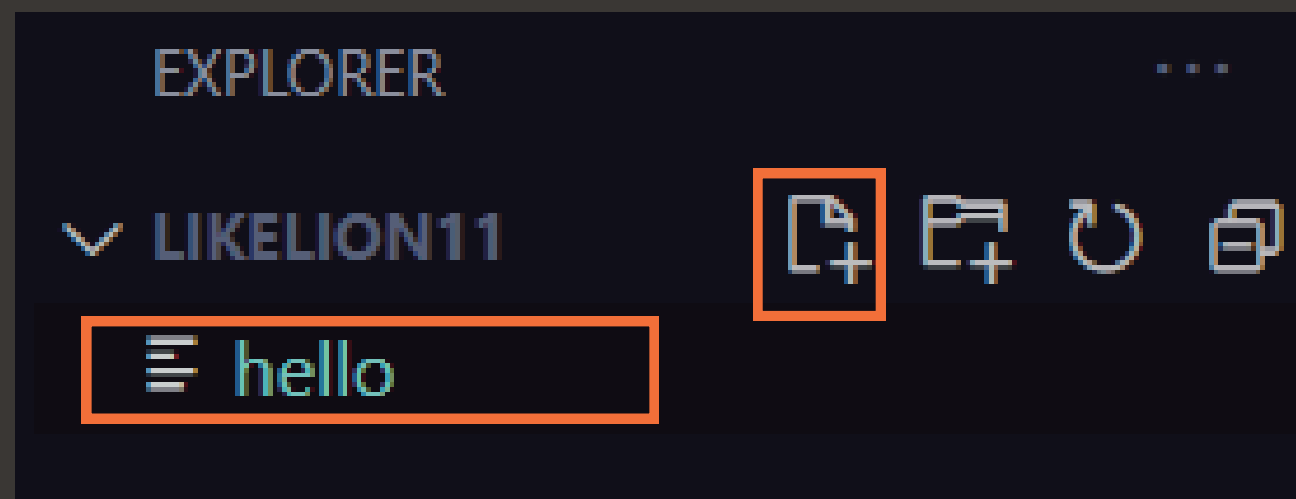
Vscode 실행

1. 만약 상단에 신뢰할 수 없다는 경고창이 뜬다? -> '관리' 버튼 클릭
2. '신뢰할 수 있는 창에서' 영역에 있는 '신뢰하기' 버튼 클릭
3. 폴더 열기 클릭
4. 새로 만들었던 폴더를 선택해 열기 버튼 클릭

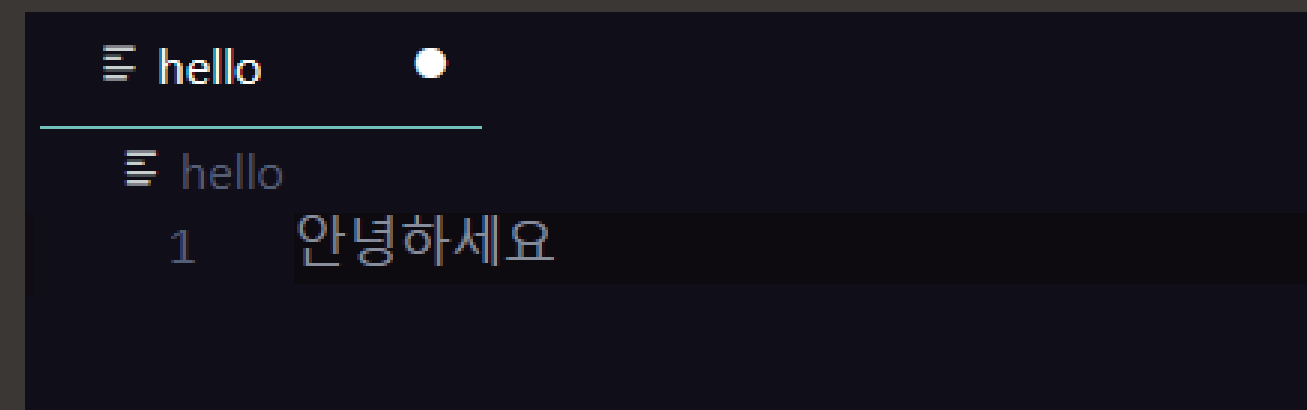


Vscode 실행

5. 새 파일 생성



6. 파일 내용 변경 후 저장



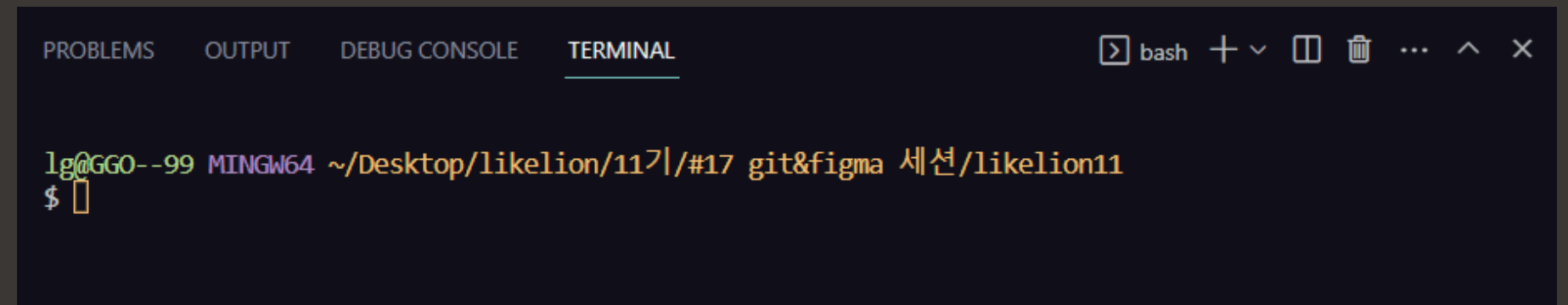
Vscode 실행

7. 터미널을 폴더 경로로 이동

-윈도우 경우 git bash로 vscode를 열었기 때문에 자동으로 경로가 설정됨.

* vscode 내에서 터미널 여는 법 : ctrl + `

-맥북 경우 폴더를 터미널로 끌고오면 경로가 자동으로 적힘

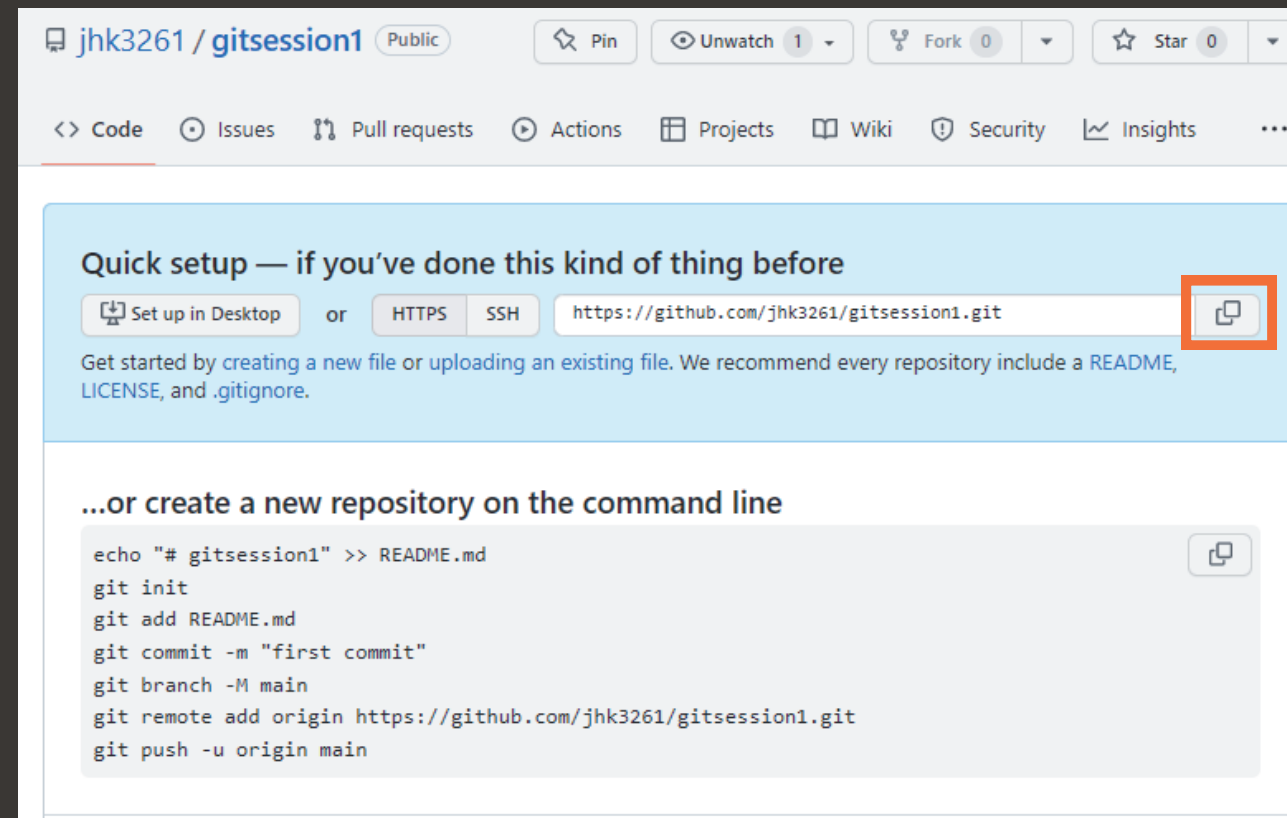


The screenshot shows a VS Code terminal window with the 'TERMINAL' tab selected. The terminal title bar indicates it's a 'bash' session. The prompt shows the user 'lg' is in a 'MINGW64' environment, with the current directory set to '~/Desktop/likelion/11기/#17 git&figma 세션/likelion11'. The prompt is '\$ ' followed by a cursor.

```
lg@GGO--99 MINGW64 ~/Desktop/likelion/11기/#17 git&figma 세션/likelion11
$
```


Vscode 실행

8. 아래 명령어를 순서대로 입력

`git init``git remote origin 주소`
`add`

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  bash + - [ ] [X] ... ^ X

lg@GGO--99 MINGW64 ~/Desktop/likelion/11기/#17 git&figma 세션/likelion11
$ git init
Initialized empty Git repository in C:/Users/lg/Desktop/likelion/11기/#17 git&figma 세션/likelion11/.git/

lg@GGO--99 MINGW64 ~/Desktop/likelion/11기/#17 git&figma 세션/likelion11 (master)
$ git remote add origin https://github.com/jhk3261/gitsession1.git
```

Vscode 실행

8. 아래 명령어를 순서대로 입력

git add .

git commit -m "커밋내용"

git push origin master

- * commit(커밋)이란?
- 파일 및 폴더의 추가/변경 사항을 저장소에 기록하는 것

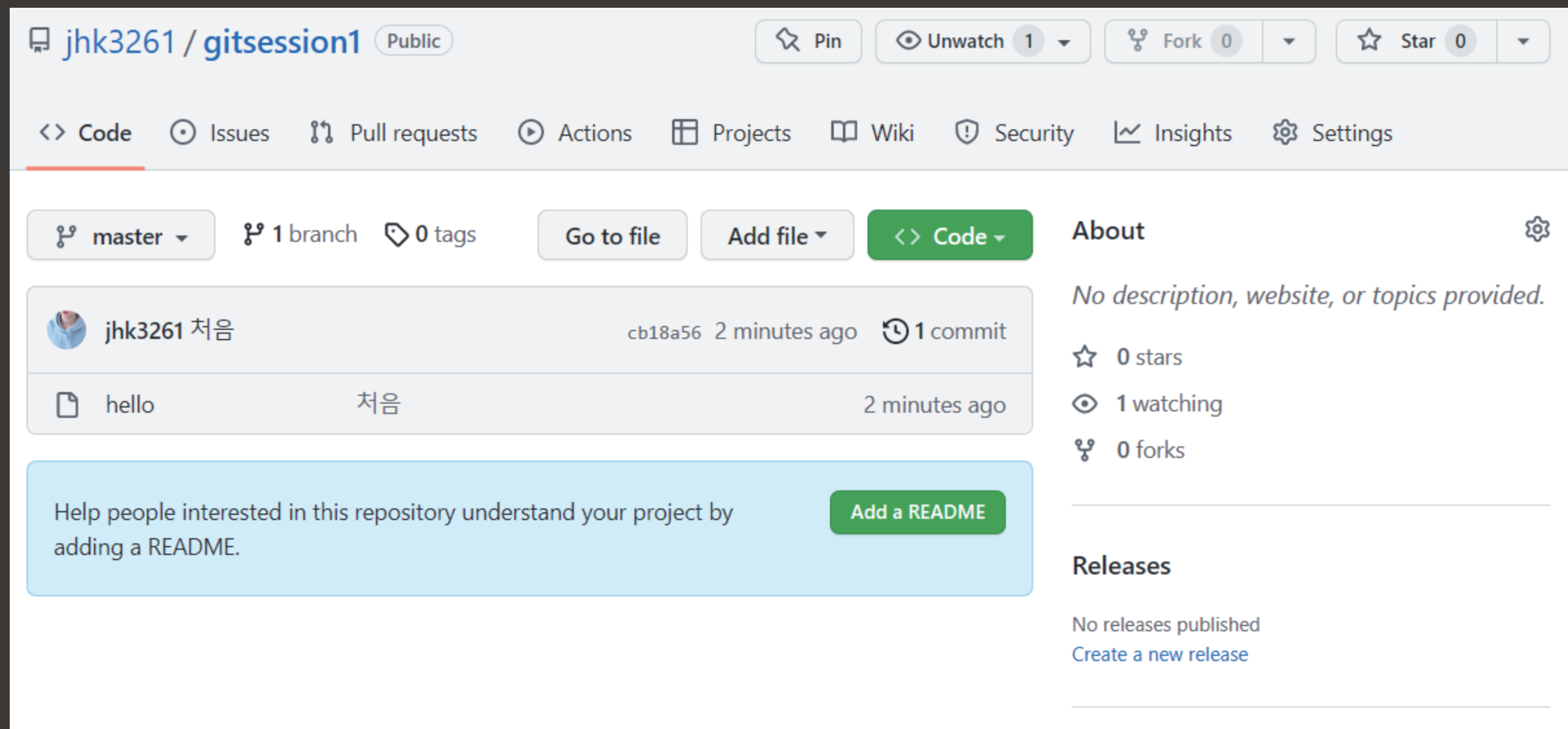
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  bash + - [ ] [X] ... ^ X

lg@GGO--99 MINGW64 ~/Desktop/likelion/11기/#17 git&figma 세션/likelion11 (master)
$ git add .

lg@GGO--99 MINGW64 ~/Desktop/likelion/11기/#17 git&figma 세션/likelion11 (master)
$ git commit -m "처음"
[master (root-commit) cb18a56] 처음
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 hello

lg@GGO--99 MINGW64 ~/Desktop/likelion/11기/#17 git&figma 세션/likelion11 (master)
$ git push origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 206 bytes | 25.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/jhk3261/gitsession1.git
 * [new branch]      master -> master
```

Github 페이지에 올라간 것을 확인 가능



Github 페이지 내에서 수정 후 변경된 내용 받아오기

명령어 입력

git pull origin master



```
hello x
hello
1 안녕하세요 (수정합니다)
2

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
lg@GGO--99 MINGW64 ~/Desktop/likelion/11기/#17 git&figma 세션/likelion11 (master)
$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 648 bytes | 19.00 KiB/s, done.
From https://github.com/jhk3261/gitsession1
* branch          master      -> FETCH_HEAD
   7595ef6..fb6dd7e master      -> origin/master
Updating 7595ef6..fb6dd7e
Fast-forward
 hello | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

(04.) FIGMA



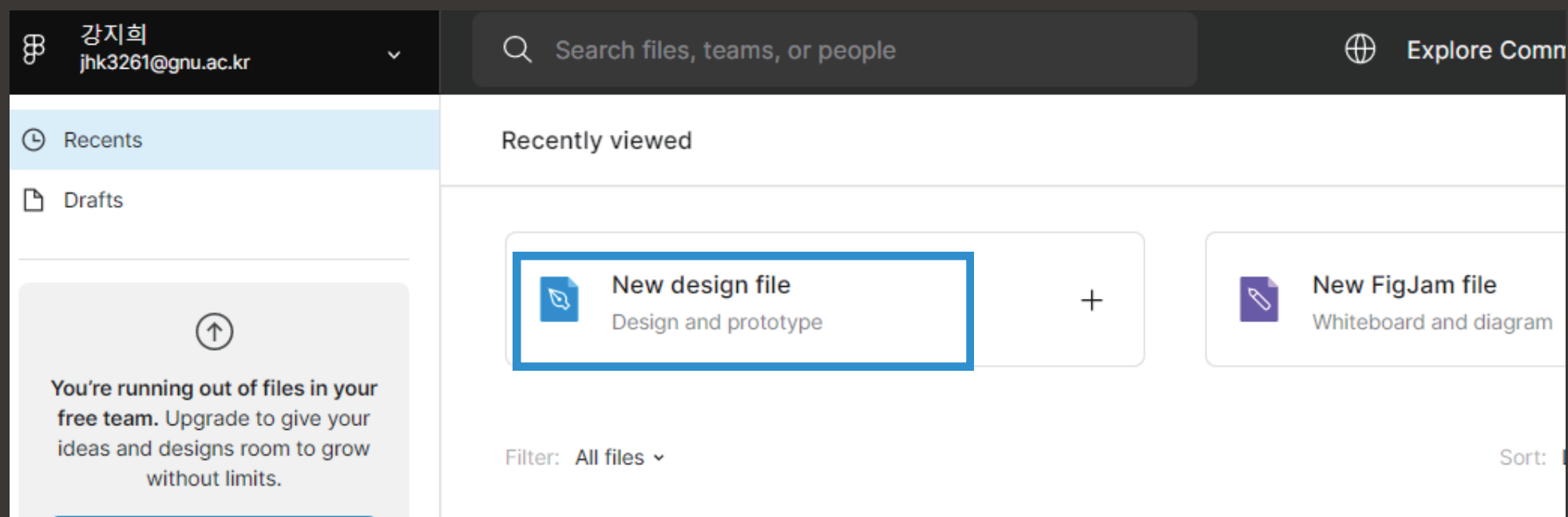
Figma는 웹(web) 브라우저 기반의
팀 단위 실시간 협업이 가능한 툴

1. 피그마 로그인, 없다면 회원가입 진행

<https://www.figma.com>

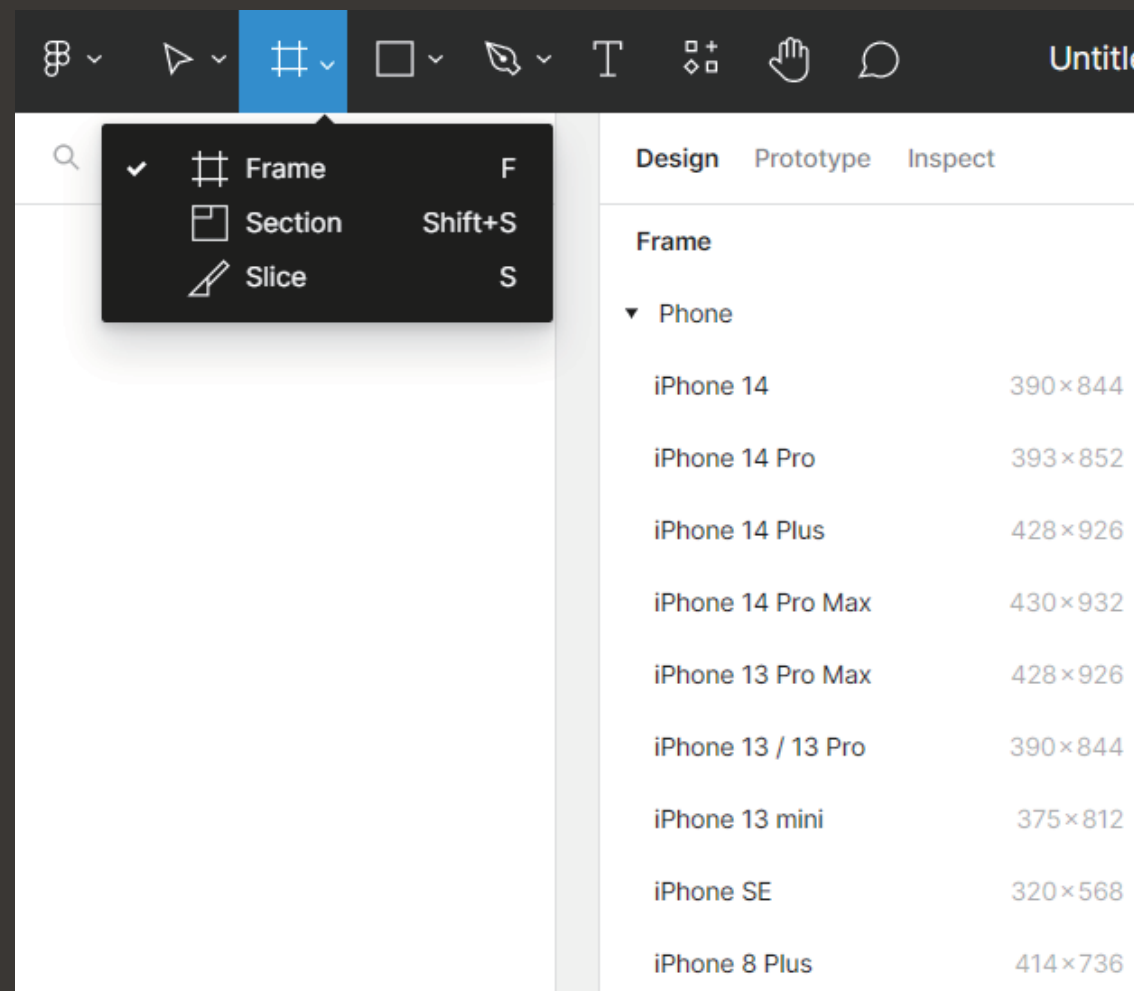


2. New design file 클릭해 파일 생성

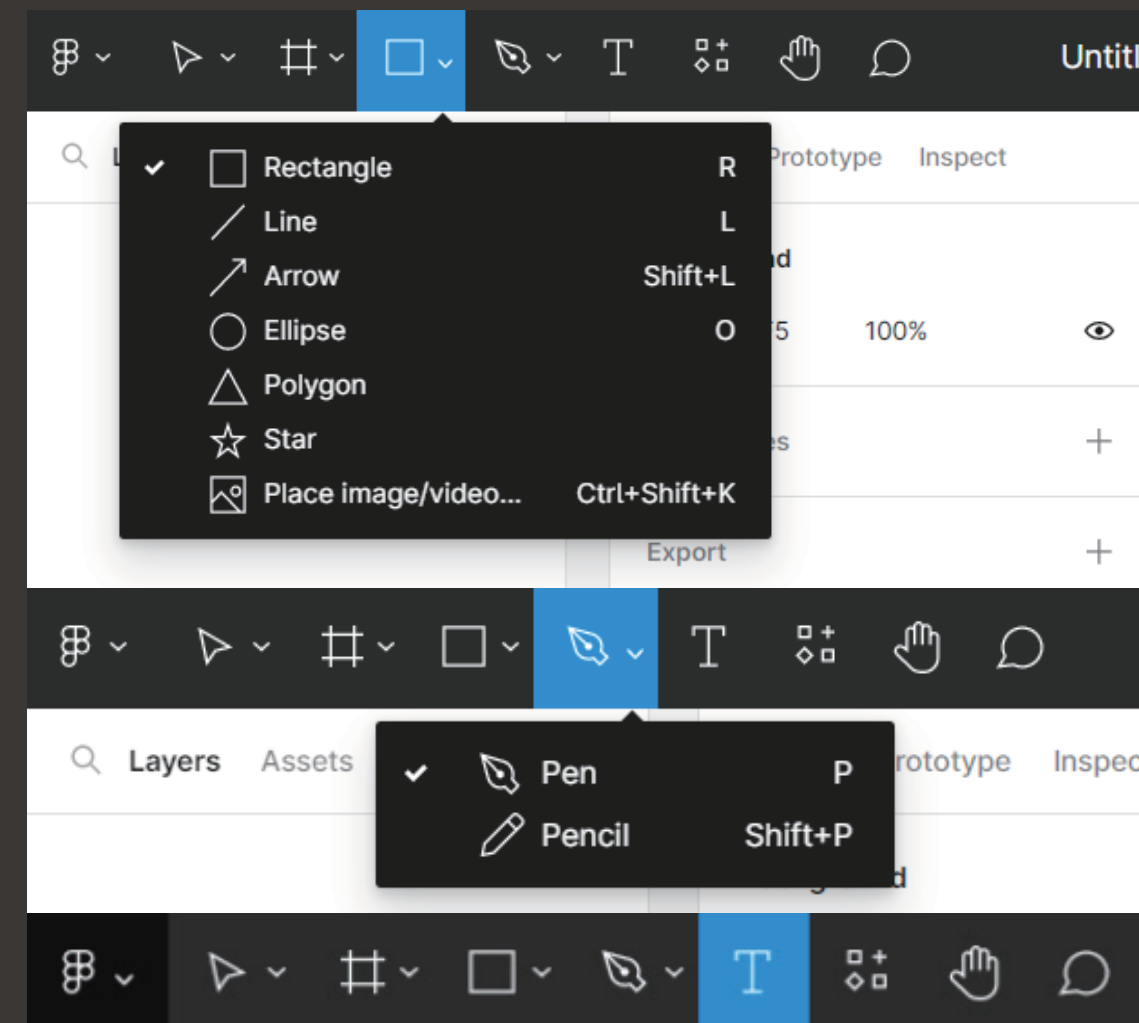


1. 요소 만들기

1) frame 선택 가능

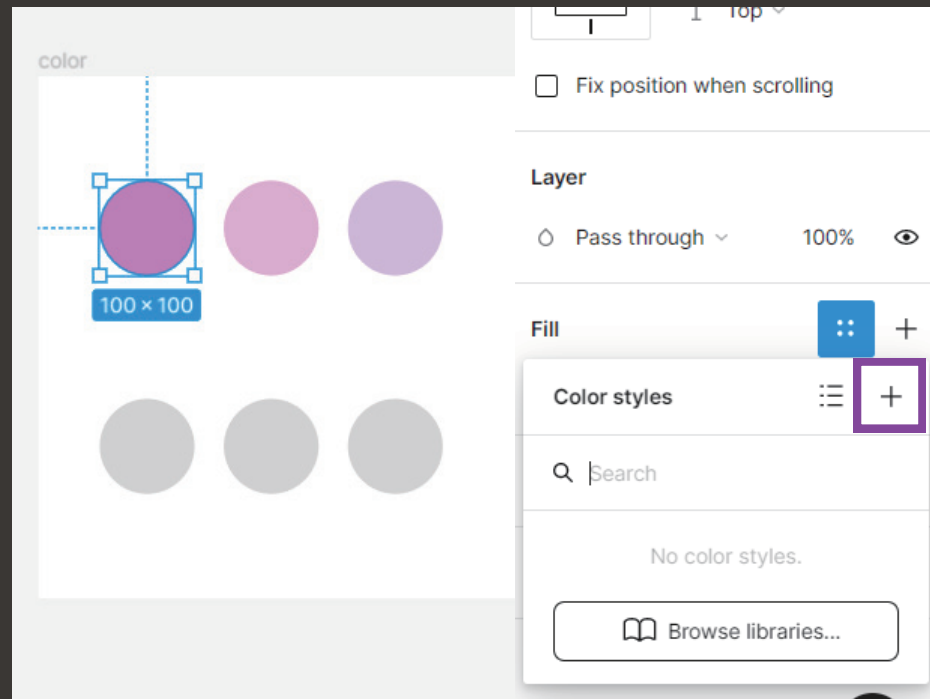


2) 여러가지 도형, 선, text 생성

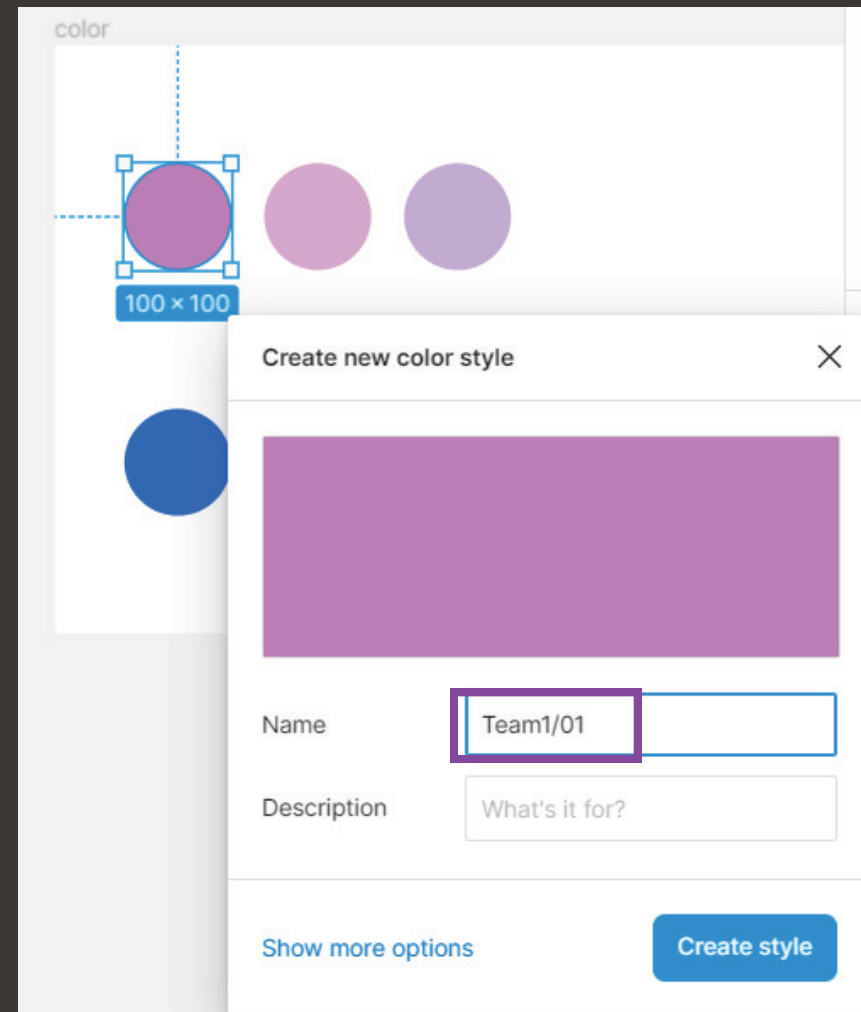


2. Color 설정

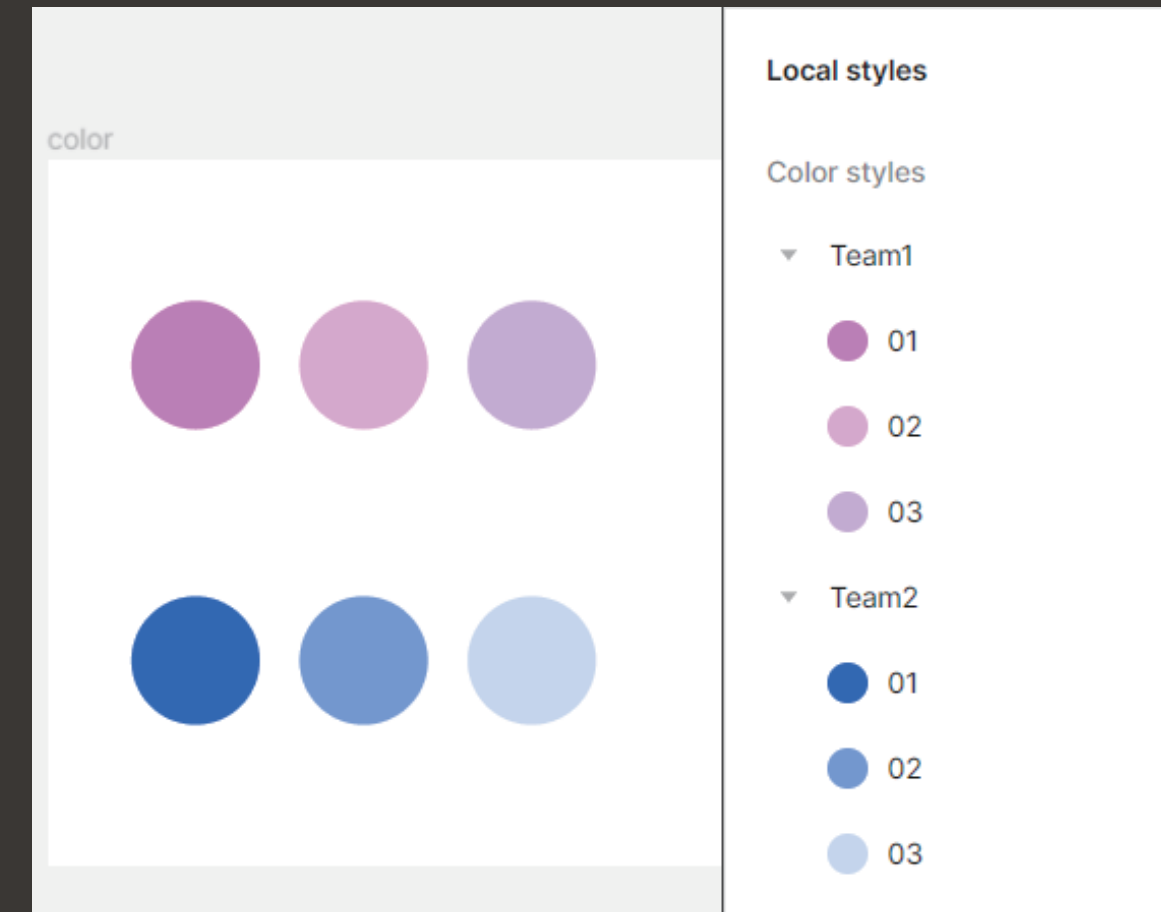
1) 요소 선택후 Fill창 클릭
-> + 버튼 클릭



2) 팀이름/색이름 입력

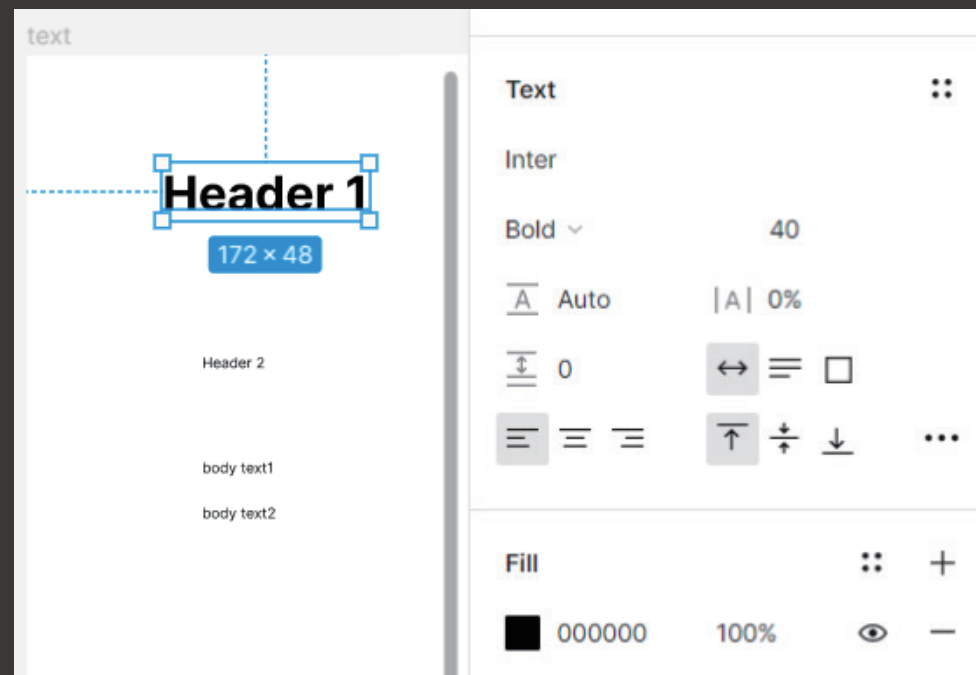


3) 팀별 색상 적용 완료

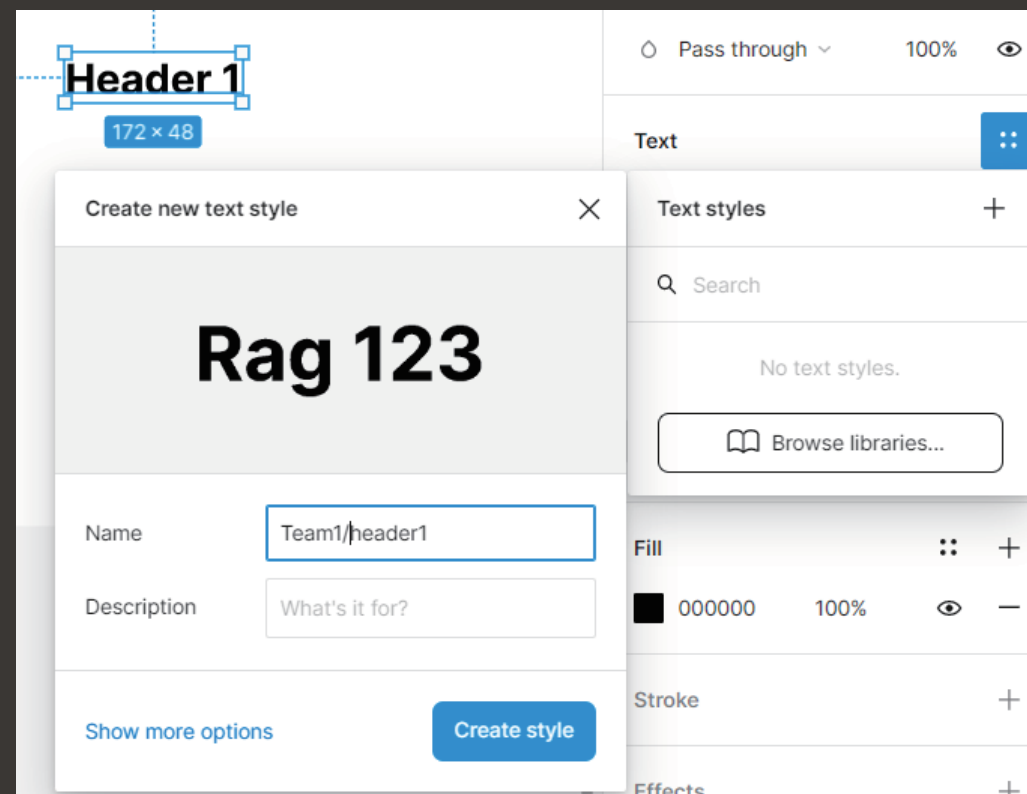


3. Text Style 설정

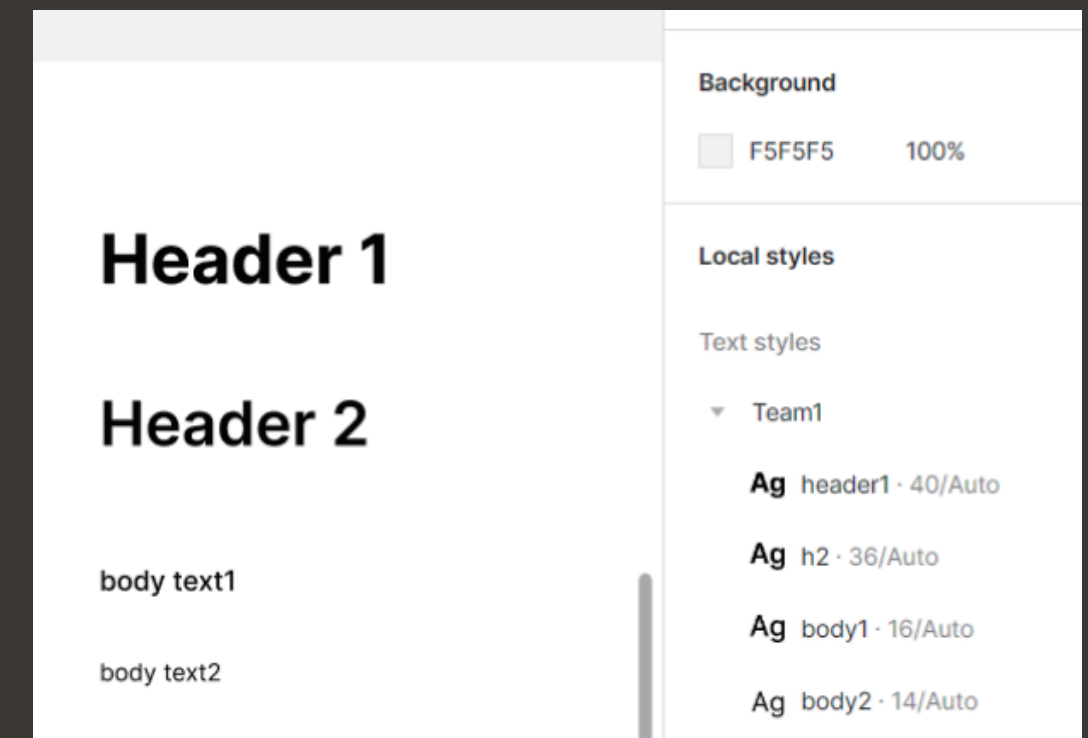
1) 요소 선택후 Text창 클릭
-> 점 네 개 버튼 클릭



2) 팀이름/스타일이름 입력

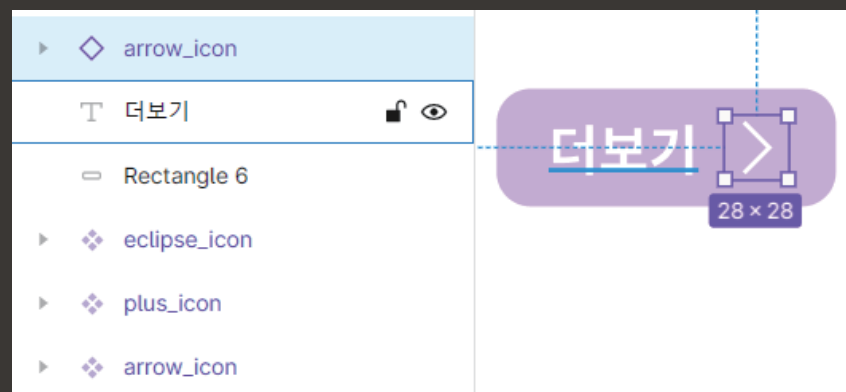
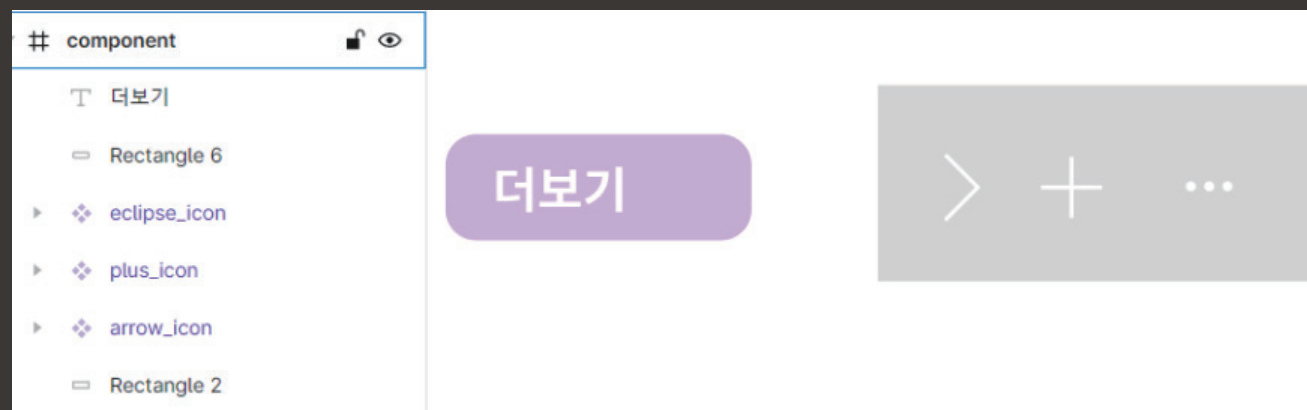


3) 팀별 text 스타일 적용 완료

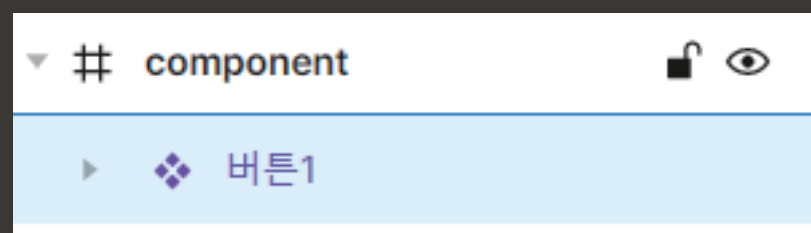
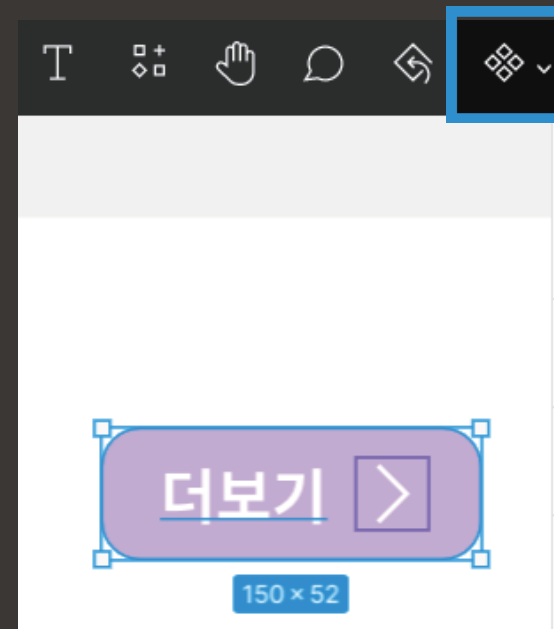


4. Component 설정

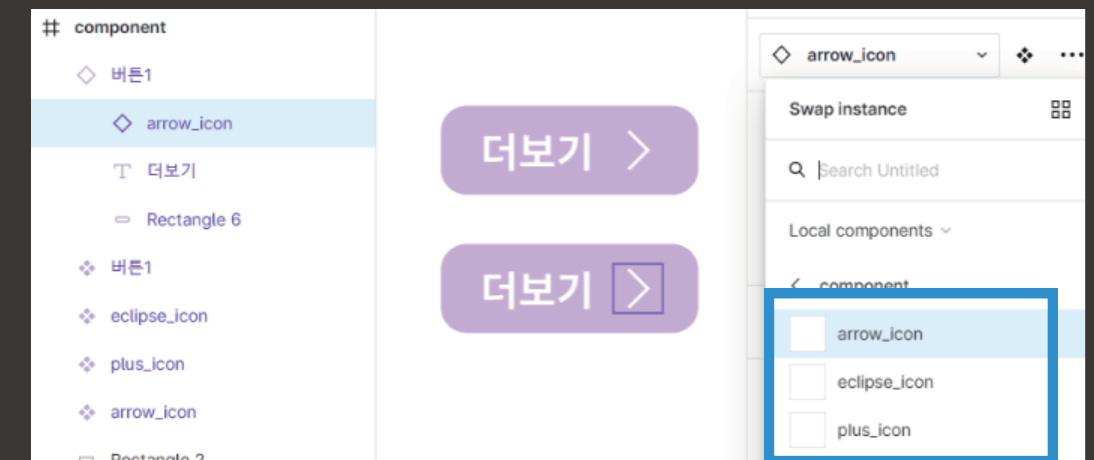
1) 버튼 만들고 안에 컴포넌트 넣기



2) 모두 선택해서 위에 다이아 버튼 클릭 -> 컴포넌트 생성

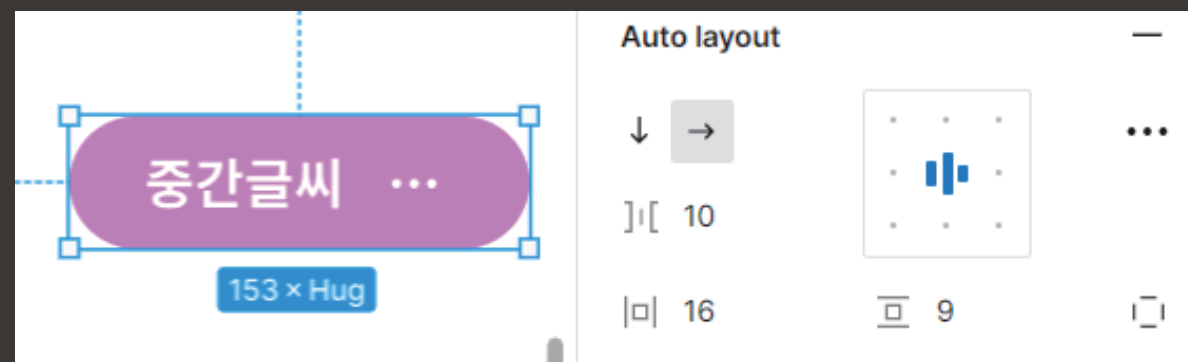
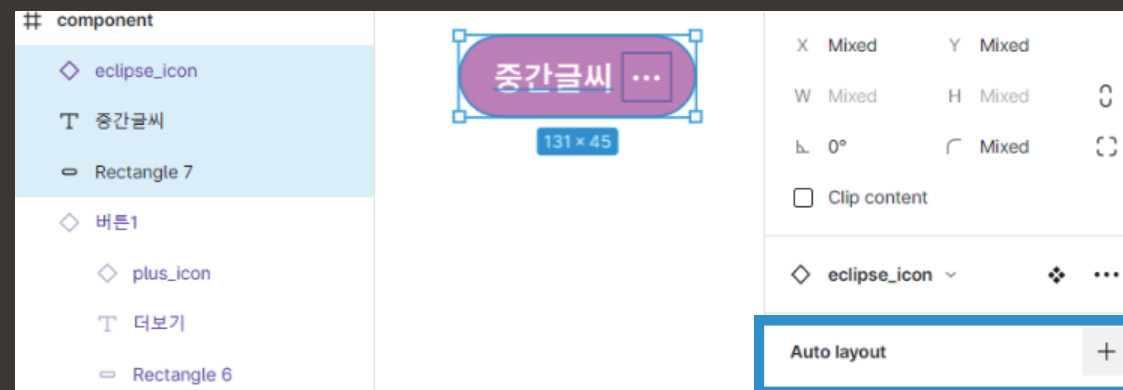


3) icon 컴포넌트 변경 및 버튼 색상 변경



4. Component 설정 - Auto layout

1) 요소 여러 개 선택 후 Auto layout창 클릭 -> 버튼 레이아웃 설정 가능

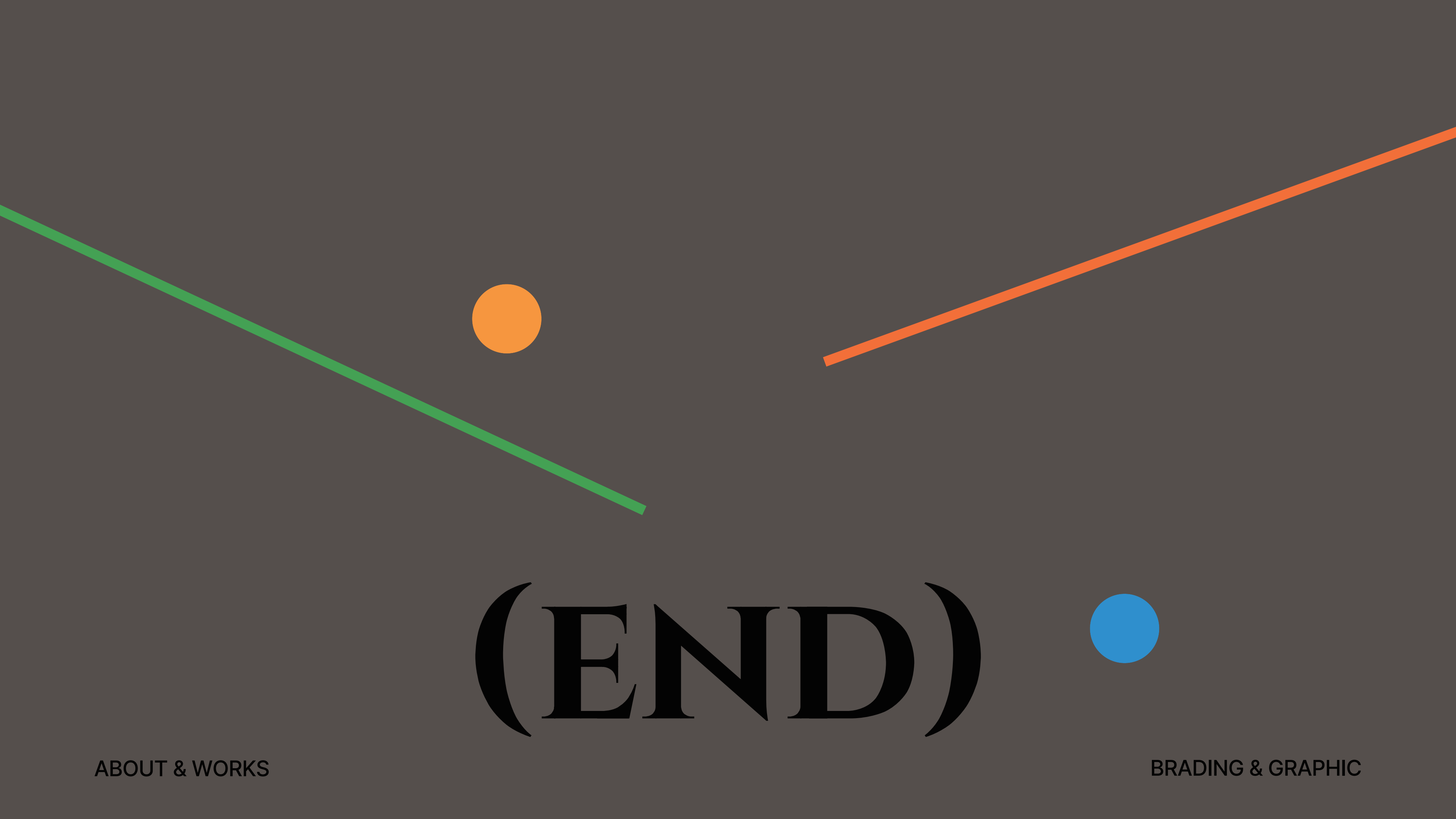


2) 컴포넌트 만드는 버튼 두번 클릭



3) 모형은 같고 색상/구성이 다른 컴포넌트들 설정 가능





(END)

ABOUT & WORKS

BRADING & GRAPHIC