

A thin orange rectangular frame is centered on the page, surrounding the text.

Git & Github



git

vs



GitHub



깃(Git)

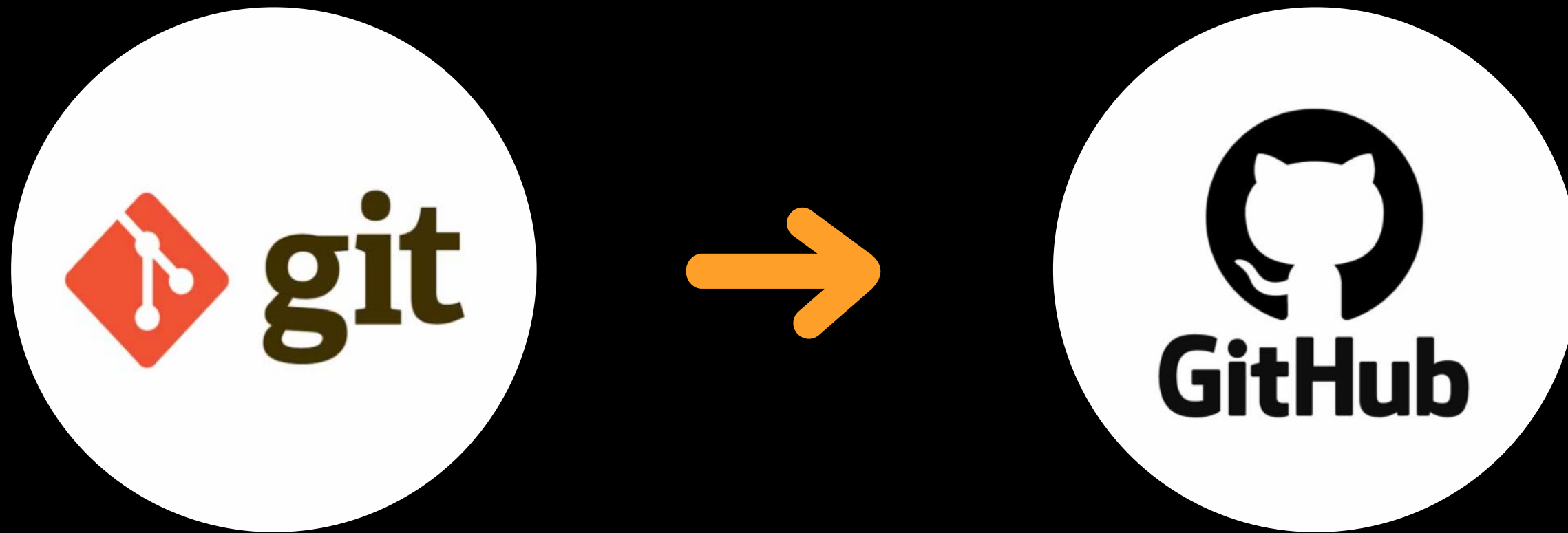
컴퓨터 파일의 변경사항을 추적하고
여러 명의 사용자들 간에 해당 파일들의 작업을
조율하기 위한 분산 버전 관리 시스템
또는 이러한 명령어



깃허브(GitHub)

분산 버전 관리 툴인 깃저장소 호스팅을
지원하는 웹 서비스

버전 관리가 필요한 이유?



개발자 간의 협업을 위해

깃의 장점

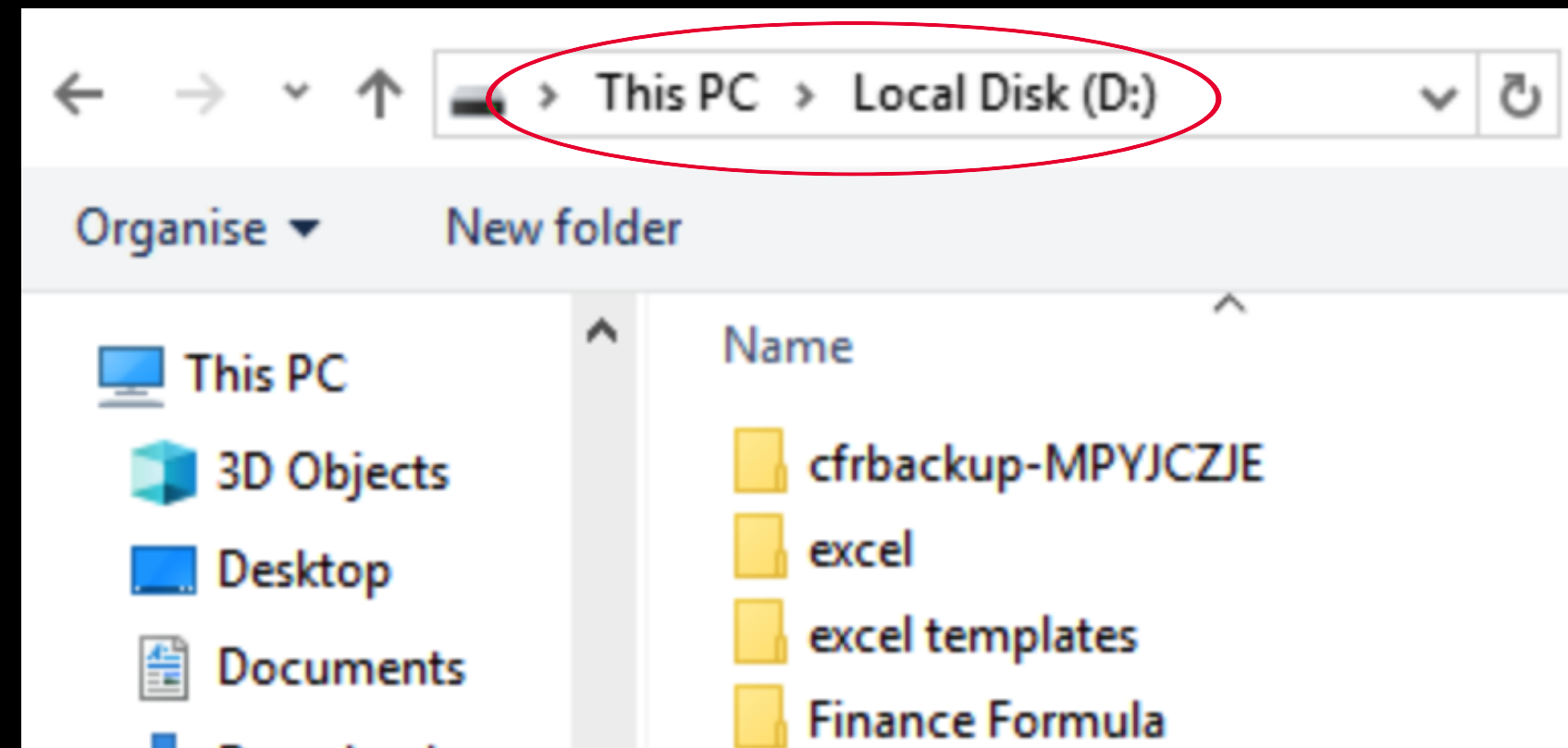
빠른 협업 환경 조성 가능

누가 언제 무엇을 수정했는지 확인 가능

이슈 트래킹 가능

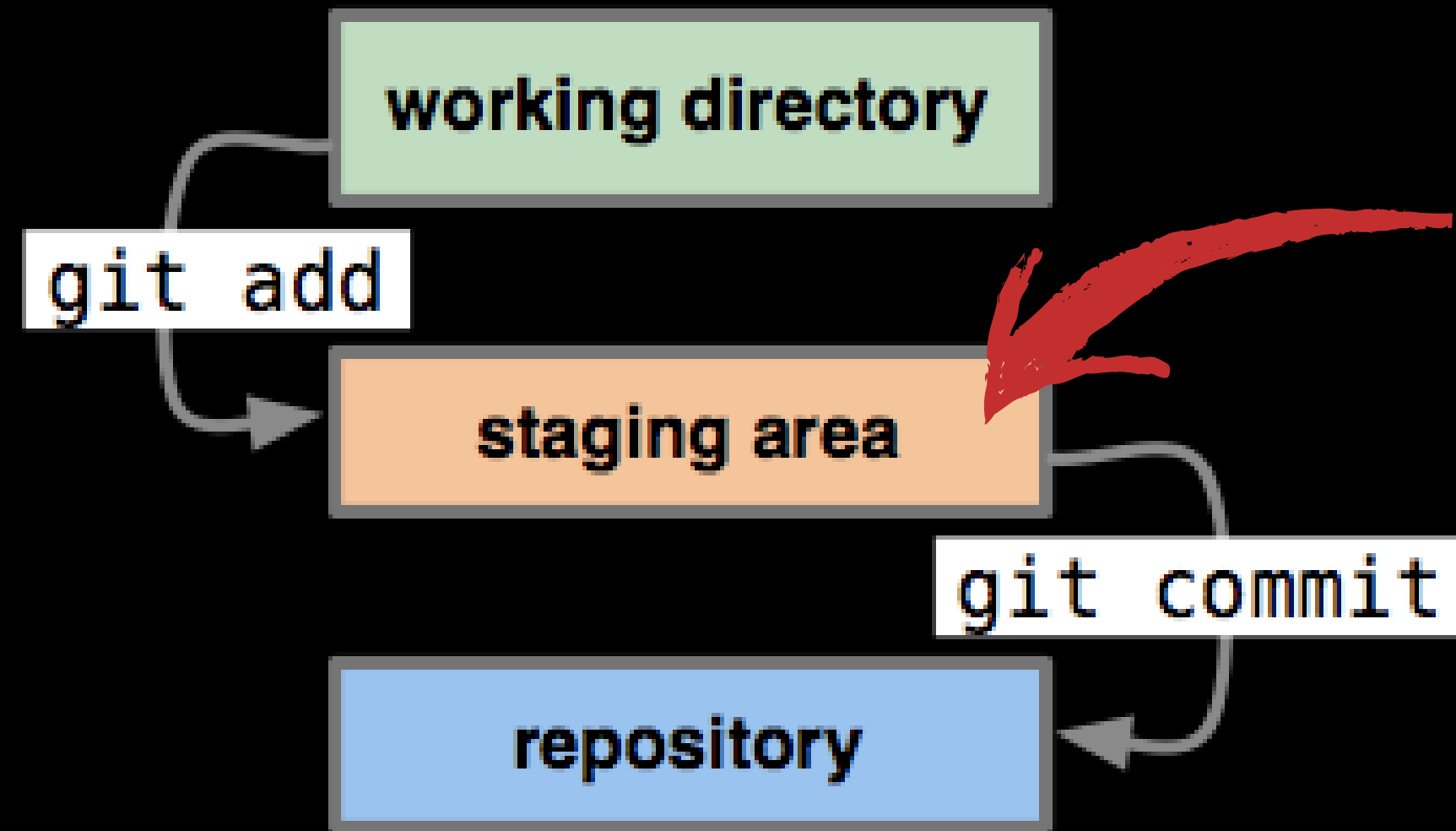
대부분의 IDE에서 GIT 연동 제공

Working Directory 작업 디렉토리



계층화된 파일 시스템의 디렉토리
보통 현재 위치의 디렉토리를 가리킴

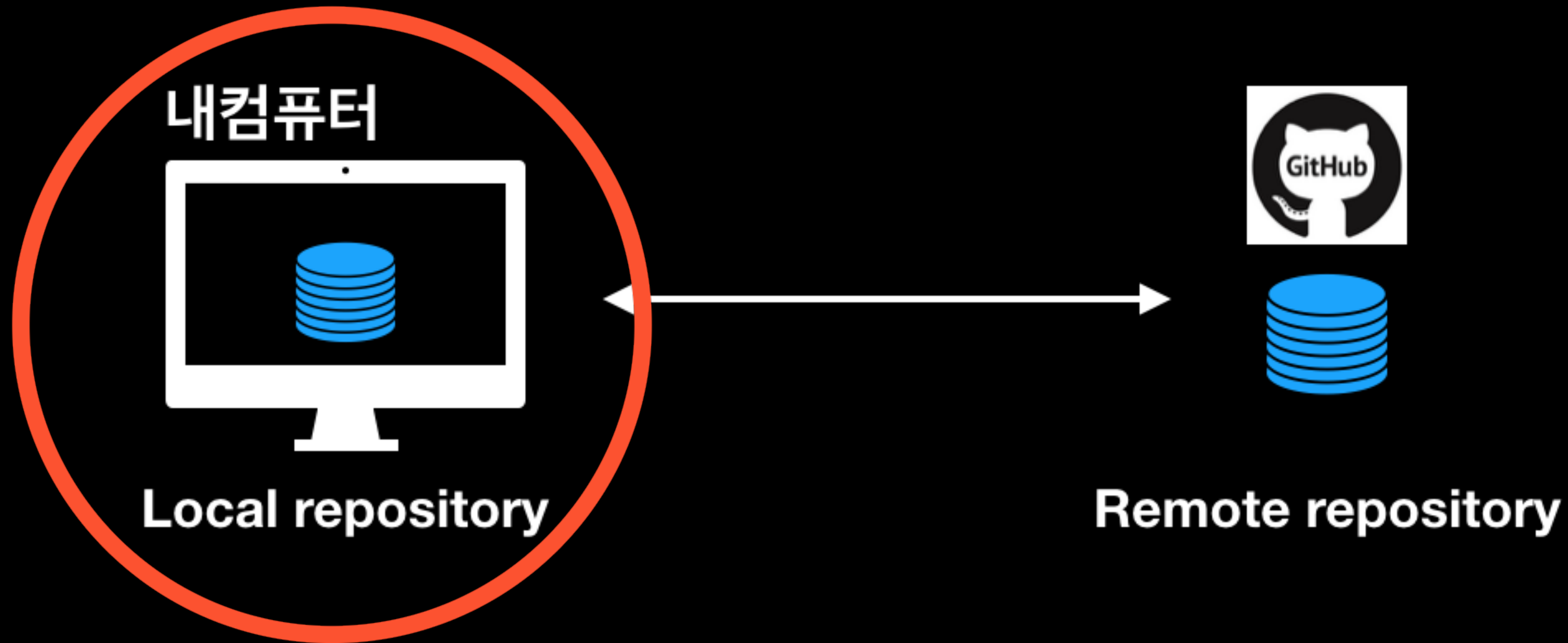
Staging Area 준비영역 or 랜딩영역



어떤 변경사항이 저장소에 커밋되기 전에
반드시 거쳐야만 하는 중간단계

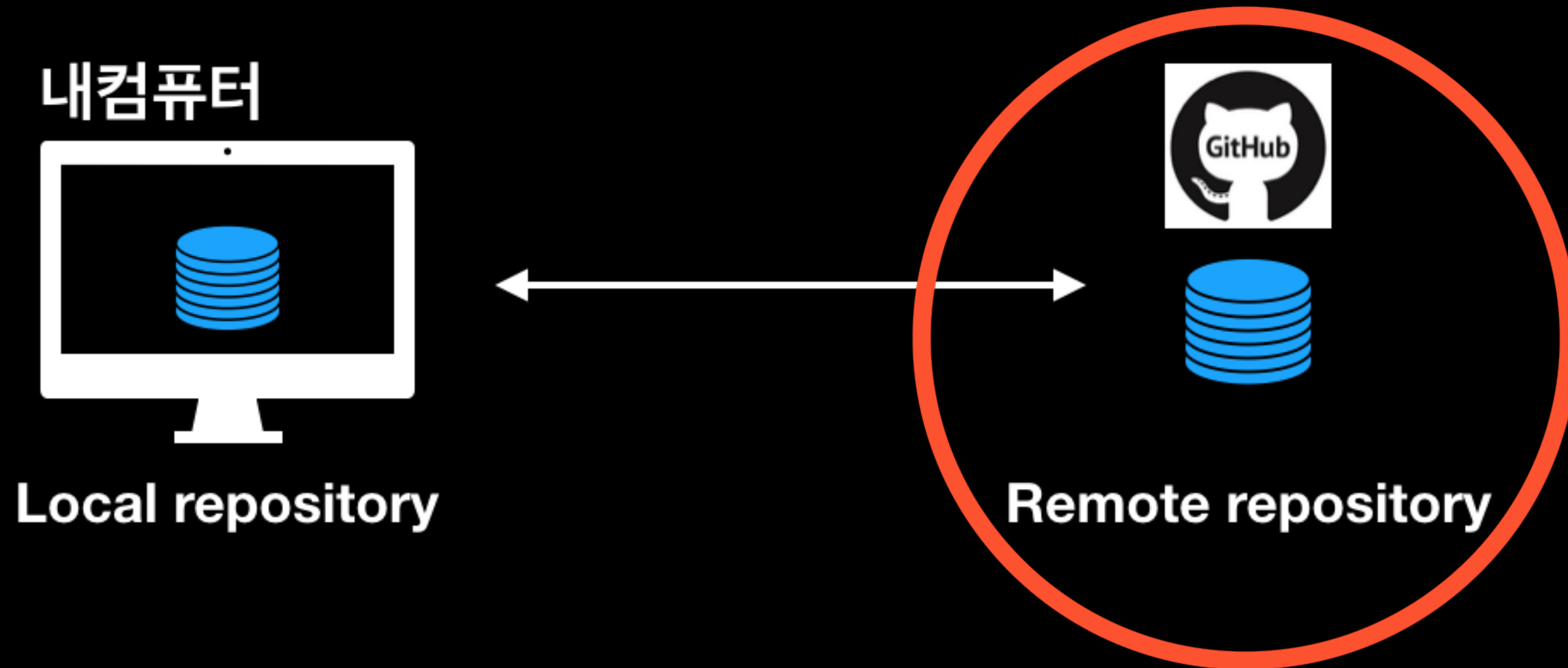
Local Repository

로컬 저장소

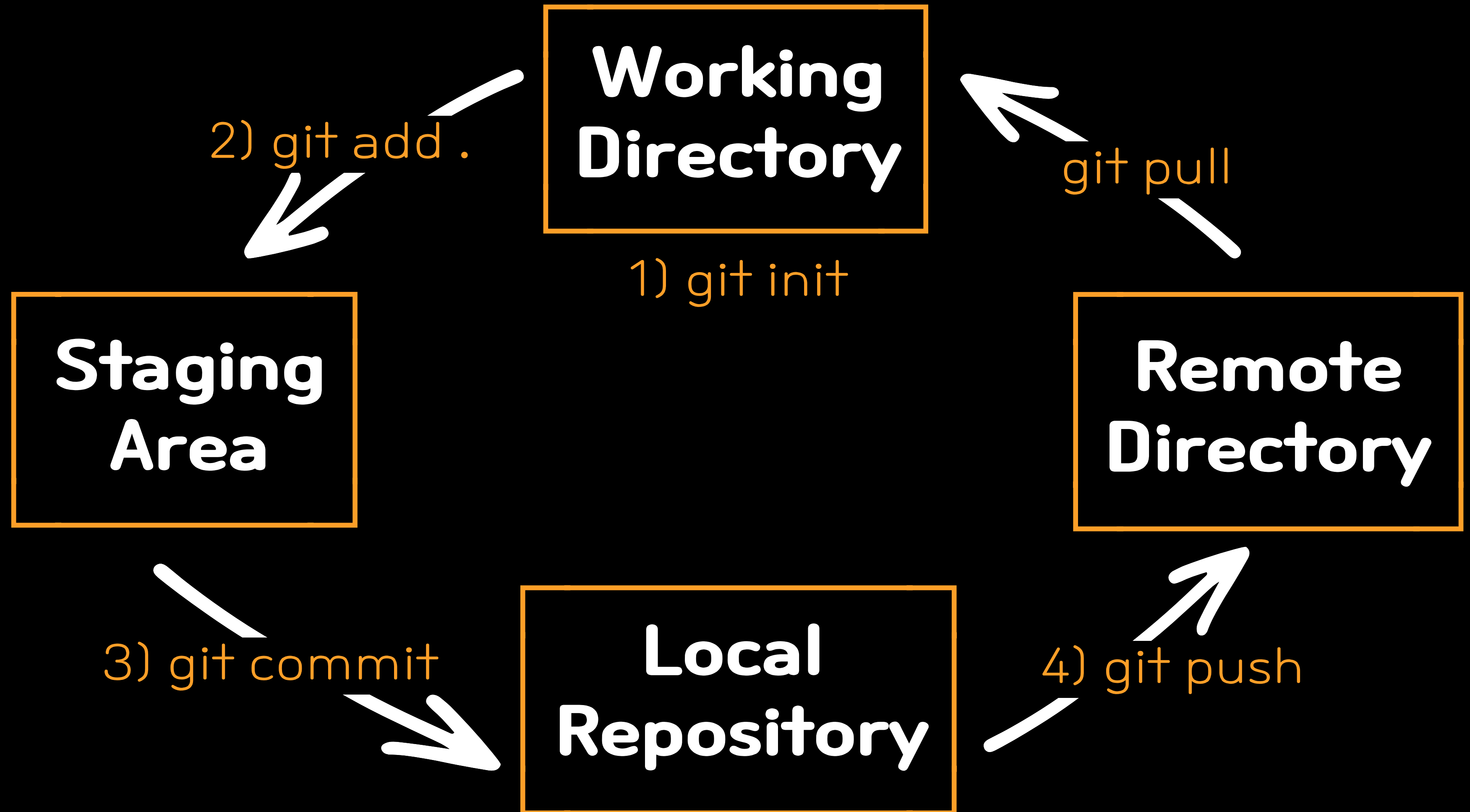


내 컴퓨터에 파일이 저장되는
개인 전용 저장소

Remote Directory 원격 저장소



네트워크 상의 서버에 있는 저장소



PPT를 보면서
함께 실습해보아요

질문은 **교육팀 운영진**분들께
바로바로 물어봐주시면 됩니다

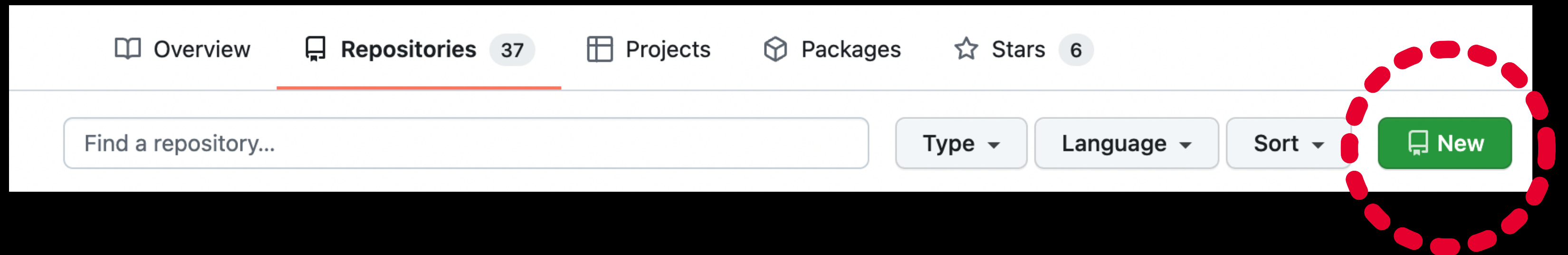


깃과 깃허브 연결해보기

1) 깃허브 레포지토리 만들기

<https://github.com/>

깃과 깃허브 연결해보기




깃과 깃허브 연결해보기

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 *

 HYERINI ▾

/


Git&Github ✓


Great repository names

Your new repository will be created as **Git-Github**. [How about **animated-engine**?](#)

Description (optional)

깃과 깃허브에 대한 레포지토리입니다.

☒  **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** **README.md 파일 생성**
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](#).

This will set  master as the default branch. Change the default name in your [settings](#).

Create repository

레포지토리 이름

간단 설명

공개

혹은 비공개

README.md 파일 생성

README.md

▶ 해당 레포지토리에 대한 설명을 담은 파일

깃과 깃허브 연결해보기

The screenshot shows a GitHub repository interface. At the top, the branch 'master' is selected, with a red circle and the Korean label '브랜치' (branch). Next to it, '1 branch' is also circled in red with the label '현재 브랜치 : master' (current branch : master). The repository name is 'HYERINI Initial commit'. A file 'README.md' is listed with the note 'Initial commit'. The main content area shows the title 'Git-Github' and the description '깃과 깃허브에 대한 레포지토리입니다.' (This is a repository about Git and GitHub).

On the right, the 'Clone' dropdown menu is open, showing options: 'HTTPS', 'SSH', and 'GitHub CLI'. The 'HTTPS' option is selected, and the URL 'https://github.com/HYERINI/Git-Github.' is displayed in a text box with a copy icon. This entire clone section is highlighted with a red rectangle. Below the URL, it says 'Use Git or checkout with SVN using the web URL.' Other options in the menu include 'Open with GitHub Desktop', 'Open with Visual Studio', and 'Download ZIP'.

Annotations in Korean:

- 브랜치 (branch) - pointing to the 'master' branch selector.
- 현재 브랜치 : master (current branch : master) - pointing to the '1 branch' indicator.
- 깃허브 레포지토리 주소 (GitHub repository address) - pointing to the clone URL.

깃과 깃허브 연결해보기 (초기설정)

2) 깃 계정 정보 설정하기

```
git config --global user.name ""  
git config --global user.email ""
```

깃과 깃허브 연결해보기 (초기설정)

3) 깃과 연결할 폴더 생성하기

```
cd "폴더 경로"
```

깃과 깃허브 연결해보기 (초기설정)

4) 깃 저장소 만들기

```
git init
```

깃과 깃허브 연결해보기 (초기설정)

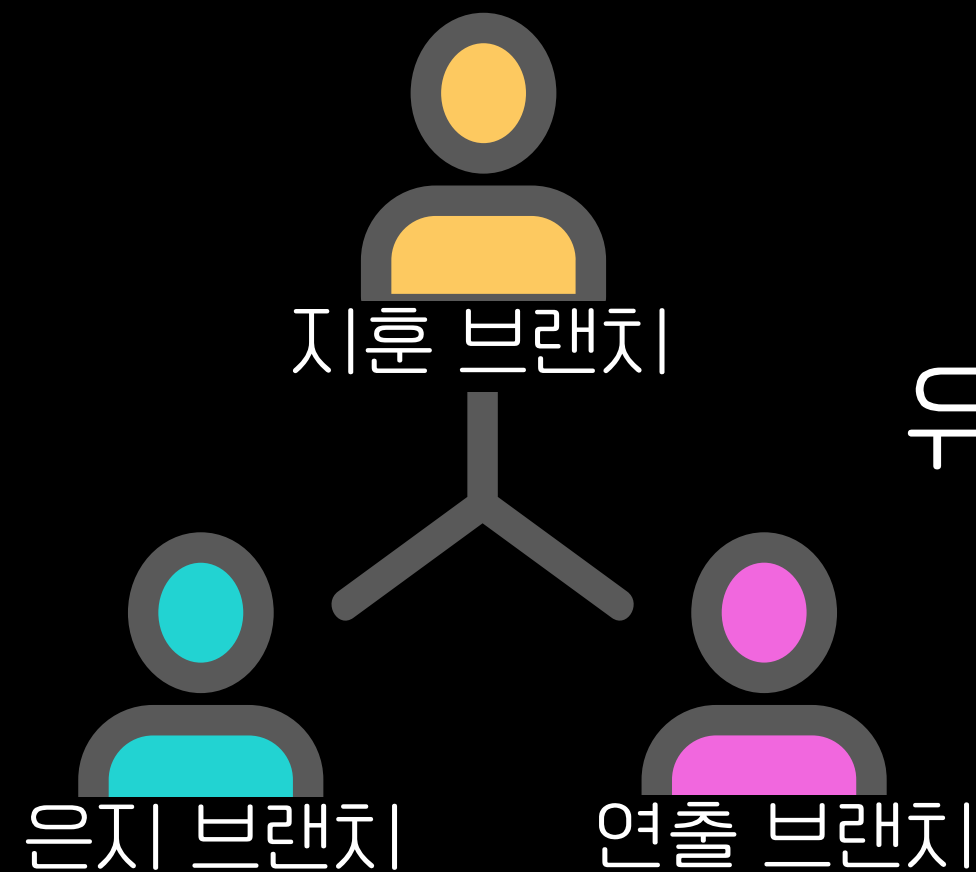
5) 폴더와 레포지토리 연결하기

```
git remote add origin  
"레포지토리 주소"
```

깃과 깃허브 연결해보기 (초기설정)

브랜치란 ?

로컬 저장소 내에서 별도의 저장 공간을 만드는 것



기존에 저장한 파일은 main 브랜치에
유지하면서 독립적으로 기존 내용을 수정하거나
스테이징, 추가할 수 있다.

깃과 깃허브 연결해보기 (초기설정)

6) 브랜치 생성하고, 해당 브랜치로 옮기기

git branch // 현재 브랜치 확인

git branch <브랜치명> // 새로운 브랜치 생성

git checkout <브랜치명> // 새로운 브랜치로 이동

깃과 깃허브 연결해보기 (초기설정)

7) 브랜치 생성하고, 해당 브랜치로 옮기기

```
git checkout -b <브랜치명>
```

// <브랜치명>이라는 새로운 브랜치를
생성하고, 해당 브랜치로 이동

깃과 깃허브 연결해보기 (초기설정)

8) 파일을 staged 상태로 만들기

```
git add .
```


깃과 깃허브 연결해보기 (초기설정)

9) 업로드할 파일에 커밋 달아주기

```
git commit -m "1주차과제"
```

깃과 깃허브 연결해보기 (초기설정)

10) 폴더의 파일을 레포지토리에 push

```
git push origin <브랜치명>
```

2주차 과제 제출 방법

1

한 폴더에 2주차 과제를 완료 후 저장한다.

2

트랙별로 생성되어 있는 레포지토리와 자신의 폴더를 연결한 후 자신의 이름으로 된 브랜치를 생성한다.

3

해당 브랜치에 2주차 과제를 "2주차과제 [자신의 이름]" 커밋과 함께 push해 제출한다.

초기 설정 후 작업 코드

```
git pull origin <브랜치명>
```

```
git add .
```

```
git commit -m "커밋메세지"
```

```
git push origin <브랜치명>
```

三