



## **GITHub & GITHub\_Desktop 의 비교 및 조사**

	<b>EON</b>
	<b>1주차 보고서</b>
<b>제 출 자</b>	201912512 원종찬 (전자공학과)
<b>제 출 일</b>	2022/06/14

## 목차

1. GitHub
2. GitHub Desktop
3. ChatGPT를 이용한 조사
4. 느낀점

### [GitHub]

#### Git이란?

위키백과에 의하면 컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 스냅샷 스트림 기반의 분산 버전 관리 시스템이다.

여기서 분산 버전 관리 시스템이란 무엇일까?

분산은 여러 개발자가 하나의 소프트웨어개발 프로젝트를 한다고 할 때, 그 업무를 분리하여 관리 및 개발을 할 수 있다는 의미이고,

버전은 개발된 파일들에 대해 버전관리를 할 수 있게 해준다는 의미이다.  
이것의 장점은 무엇일까?

위의 변경사항 추적에 주목을 할 필요가 있다.

변경사항이 변경되고 이전의 사항이 자동 폐기된다고 생각해보자. 이건 마냥 좋은건 아니다.

개발 과정에서 유용할수 있었지만, 프로젝트에 맞지 않아 폐기되는 코드들이 많다.

즉, 다른 과정에서도 유용할 수 있는 코드들이 많을 것이고,

해당 프로젝트의 기록이 있다면 프로젝트가 빌드된 과정또한 볼 수 있으며,

프로젝트의 변경사항이 많다면 해당 기록들을 보면서

프로젝트 멤버 또한 변경사항에 대해 빠르게 파악이 가능하다.

이는 실로 유용하다고 볼 수 있다.

코드가 변경될 때, 수없이 많은 코드중 바뀐 코드를 파악하기란 실로 쉽지 않고, 이를 단시간에 파악후, 코멘트를 통해 변경사항에 대한 구체적인 설명이 달린다는 개념은 가히 혁신적이라고 볼 수 있다.

이와같이 협업에 있어서 깃의 영향력은 엄청나다.

정리된 깃의 장점이 있어 가져왔다.

<https://yanacoding.tistory.com/4>

인터넷 연결이 되지 않은 곳에서도 개발을 진행할 수 있으며, 분산 버전 관리이기 때문에 중앙저장소가 삭제되어도 원상 복구가 가능하다.

각각의 개발자가 Branch에서 개발 한 뒤, 본 프로그램에 합치는 merge 방식을 통한 병렬 개발가능.

놀랍지 않은가?

(난 개인적으로 놀랍다. 사실 매일 깃허브만 사용하고 깃은 사용안해봤는데 오프라인에서도 사용할수 있다는 사실이 굉장히 신선하다.)

이에 비교해서 깃허브란 뭘지 확인해보겠다.

### GitHub??

깃을 사용하는 프로젝트를 지원하는 웹 호스팅 서비스.

위의 링크에서는 애니팡에 비교했다. 좀 올드하긴 하지만,

깃 : 휴대폰 속 애니팡(오프라인에서 플레이가 가능)

깃허브 : 애니팡 서버(온라인 서버에서 각 유저가 interaction이 가능하게...!)

사실 깃허브를 쓰는편이 더 간편하다.

깃의 경우는 명령어를 사용하지만, 깃허브는 UI가 있기 때문에 보기에 편리하고

영어를 잘 못하는 나조차도 접근도 못할정도로 어렵지 않다는게 큰 장점으로 다가온다...!

흥미로운 사실!

<https://namu.wiki/w/GitHub>

원래는 공개 프로젝트만 무료였고 비공개 프로젝트는 결제를 해야 했으나, MS에 인수된 이후 2019년 1월 초부터 비공개 저장소를 무료로 제공하기 시작했다

private저장소 사용이 원래 금액을 지불해야 했다고 한다.

### [GitHub Desktop]

사실 이 GitHub Desktop이라는 원래 안친했다.

GitHub에서 클론하지 않으면 깃헙 데스크톱에 내 repos가 뜨질 않았어서

처음에 사용했을 때는 굉장히 당황했었다. 그래서 자꾸 안쓰는 버릇이 들었었다.

지금은 뭐 능수 능란하게 사용이 가능하다.

깃허브 code(녹색버튼)을 누르고 URL을 복사한 뒤

깃허브 데스크톱에서 add 누르고 URL입력 창에서 URL을 붙여주고

fetch origin 이나 pull origin을 누르면 깃허브에 있는 repos를 clone하고, 또한 내가 local하게 변경한것도 내 github에 반영해준다.

### [ChatGPT를 이용한 조사]

위의 내용을 지금도 여전히 핫한 ChatGPT로 내가 직접 물어보았다.

물어본 질문은 2개이다.

\* What is Github and Github Desktop?

\* how to use them?



What is Github and Github Desktop?



GitHub is a web-based platform for version control and collaboration that allows developers to store and manage their code repositories. It provides a centralized location for developers to share and collaborate on their projects, track changes to their code, and manage their workflow.

GitHub Desktop is a graphical user interface (GUI) application for GitHub that allows users to manage their repositories using a desktop application rather than the command line. It provides an easy-to-use interface for performing common Git commands, such as committing changes, creating branches, and merging code. With GitHub Desktop, developers can easily synchronize their local repositories with their online GitHub accounts and collaborate with others on their projects.



how to use them?



how to use them?



To use GitHub and GitHub Desktop, follow these basic steps:

Using GitHub:

1. Create a GitHub account: Go to the GitHub website and sign up for a free account.
2. Create a new repository: Once you have signed in to your account, you can create a new repository to store your code. You can create a new repository by clicking the "+" sign in the top right corner of the GitHub page and selecting "New repository".
3. Add files to the repository: Once you have created a repository, you can add files to it by uploading them directly to the GitHub website or by using Git commands in a command-line interface.
4. Collaborate with others: You can share your repository with others and collaborate on code together. You can add collaborators to your repository by going to the "Settings" page of your repository and selecting "Manage Access".

### Using GitHub Desktop:

1. Install GitHub Desktop: Download and install GitHub Desktop on your computer from the GitHub Desktop website.
2. Clone a repository: Open GitHub Desktop and clone a repository by selecting "Clone Repository" from the "File" menu and entering the repository URL.
3. Make changes to the code: Once you have cloned the repository, you can make changes to the code using your preferred code editor.
4. Commit and push changes: Once you have made changes to the code, you can commit the changes and push them to the GitHub repository using GitHub Desktop. Select the changes you want to commit and enter a commit message, then click "Commit" to save your changes locally. Finally, click "Push" to upload your changes to the GitHub repository.

These are just the basic steps, and there are many more advanced features available in both GitHub and GitHub Desktop. It's recommended that you read the documentation and tutorials provided by GitHub to learn more about the platform and how to use it effectively.



이건 진짜 미쳤다.. pico second마다 정말 미치도록 놀랍다.

#### \* What is Github and Github Desktop?

GitHub는 개발자가 코드 리포지토리를 저장하고 관리할 수 있는 버전 제어 및 협업을 위한 웹 기반 플랫폼입니다. 개발자가 프로젝트를 공유 및 공동 작업하고, 코드 변경 사항을 추적하고, 워크플로를 관리할 수 있는 중앙 집중식 위치를 제공합니다.

GitHub Desktop은 사용자가 명령줄이 아닌 데스크톱 애플리케이션을 사용하여 리포지토리를 관리할 수 있는 GitHub용 그래픽 사용자 인터페이스(GUI) 애플리케이션입니다. 변경 사항 커밋, 분기 생성 및 코드 병합과 같은 일반적인 Git 명령을 수행하기 위한 사용하기 쉬운 인터페이스를 제공합니다. GitHub Desktop을 사용하여 개발자는 로컬 리포지토리를 온라인 GitHub 계정과 쉽게 동기화하고 프로젝트에서 다른 사람과 협업할 수 있습니다.

\* how to use them?

GitHub 및 GitHub Desktop을 사용하려면 다음 기본 단계를 따르십시오.

GitHub 사용:

GitHub 계정 만들기: GitHub 웹사이트로 이동하여 무료 계정에 가입합니다.

새 리포지토리 생성: 계정에 로그인하면 코드를 저장할 새 리포지토리를 생성할 수 있습니다.

GitHub 페이지의 오른쪽 상단 모서리에 있는 "+" 기호를 클릭하고 "새 리포지토리"를 선택하여 새 리포지토리를 만들 수 있습니다.

리포지토리에 파일 추가: 리포지토리를 만든 후에는 파일을 GitHub 웹 사이트에 직접 업로드하거나 명령줄 인터페이스에서 Git 명령을 사용하여 파일을 추가할 수 있습니다.

다른 사람과 협업: 리포지토리를 다른 사람과 공유하고 함께 코드에 대해 협업할 수 있습니다. 리포지토리의 "설정" 페이지로 이동하고 "액세스 관리"를 선택하여 리포지토리에 공동 작업자를 추가할 수 있습니다.

GitHub 데스크톱 사용:

GitHub Desktop 설치: GitHub Desktop 웹 사이트에서 컴퓨터에 GitHub Desktop을 다운로드하여 설치합니다.

리포지토리 복제: GitHub Desktop을 열고 "파일" 메뉴에서 "리포지토리 복제"를 선택하고 리포지토리 URL을 입력하여 리포지토리를 복제합니다.

코드 변경: 리포지토리를 복제한 후에는 선호하는 코드 편집기를 사용하여 코드를 변경할 수 있습니다.

변경 사항 커밋 및 푸시: 코드를 변경한 후에는 변경 사항을 커밋하고 GitHub Desktop을 사용하여 GitHub 리포지토리로 푸시할 수 있습니다. 커밋할 변경 사항을 선택하고 커밋 메시지를 입력한 다음 "커밋"을 클릭하여 변경 사항을 로컬에 저장합니다. 마지막으로 "푸시"를 클릭하여 변경 사항을 GitHub 리포지토리에 업로드합니다.

이는 기본 단계일 뿐이며 GitHub 및 GitHub Desktop에서 사용할 수 있는 더 많은 고급 기능이 있습니다. 플랫폼과 이를 효과적으로 사용하는 방법에 대해 자세히 알아보려면 GitHub에서 제공하는 설명서 및 자습서를 읽는 것이 좋습니다.

[느낀점]

위와같이 GitHub와 GitHub desktop에 대해 알아보았다. 또한 ChatGPT도 사용해봤는데 정말 놀랍다.

아직 답변을 직접 생각하는 것은 아니지만, 구글 서칭을 이정도 고속으로 할 수 있는 것도 놀랍다. 이번 과제의 기회를 통해 내가 아직 완벽하게 숙지하진 않았었던 GitHub 및 GitHub Desktop에 대해서 잘 알 수 있는 기회가 되었다. clone이 되게 어려운줄 알았는데 원래 생각했던 것보다 너무 쉬웠다. 솔직히 아직 협업을 할 기회는 많지 않아 자주 쓰진 않을 것

같지만, 앞으로 알아두면 이런 정보들을 알았다는 사실에 기쁘고, 추가적인 깃허브 툴과 그 사용법에 대해 시간날 때마다 찾아보고 공부해보는 것도 좋을 것 같다.