Git과 GitHub의 차이

Git

Git 은 프로젝트의 버전 관리를 위한 도구(저장공간)입니다.

1. 버전 관리를 할 수 있습니다.

Git 을 사용하면 무슨 작업을 했는지 히스토리 파악이 가능

2. 작업 단위 나누기

에러가 난 부분을 쉽가 파악 가능

3. 협업해서 하나의 프로젝트를 만드는데 유용

누가, 언제, 어떤 부분을 수정했는지를 한 눈에 파악

GitHub

Git 원격 저장소 + Git 으로 할 수 있는 커뮤니티 기능 서비스

1. 온라인 상의 프로젝트 저장소

2. 개발자들의 커뮤니티

다른 사람 공개 프로젝트 구경하고, 내가 관심있는 주제/ 프로젝트 소식 받아볼 수 있습니다.

3. 프로젝트 함께 만드는데 참여

'프로젝트에 기여하기(contribution)' 하기 위한 여러 기능도 제공합니다.

Git - 버전관리 도구

Git 에서는 '누가, 언제, 현재 프로젝트의 상태가 어떤지(현재 파일 내용들)' 세 가지 정보를 포함해 작업내역을 관리합니다.

이렇게 현재 프로젝트 상태를 저장한 것을 commit(커밋) 이라고 표현합니다.

commit 을 하는 순간 현재 프로젝트의 파일 내용, 언제, 누가 저장했는지 정보가 남습니다.

commit 들은 언제 했는지 정보도 포함하고 있어서 순서대로 보면 그 자체가 히스토리가 됩니다.

- \*\*버전관리를 한다는 것은\*\*

프로젝트 상태가 변경되는 정보를 알고 있다는 것입니다. Git 은 commit 을 사용해서 버전이 달라지는 것을 관리합니다.

- 컴퓨터에 있는 프로젝트를 Git 이 관리하는 프로젝트로 만들 수 있습니다. 앞으로 Git 으로 관리할꺼야! 하고 설정해주면 됩니다.

이 작업을 \*\*git 초기화(git initialize)\*\*한다고 표현합니다.

- 현재 프로젝트의 상태를 저장하는 것을 \*\*commit\*\* 이라고 합니다.

- commit 에는 아래를 포함합니다.

- 누가(author), 언제 commit 했는지의 정보와 프로젝트 변경 내용

- 작업내역이 어떤 것인지 알아볼 수 있게 적는 메시지를 'commit 메시지'라고 합니다.

- commit 에 반영할지 안할지는 파일 단위로 선택할 수 있습니다. commit 에 반영할 파일을 선택하는 것을

\*\*add (혹은 staging, 스테이징)\*\* 이라고 합니다.

- commit 한 기록은 \*\*history\*\* 로 볼 수 있습니다.

- 지금까지 우리가 한 작업은 'git 초기화하기(initialize) - add(staging) - commit' 입니다.

- git 초기화는 처음에 단 한번만 해 주면 됩니다. 작업 내역을 저장하기 위해서는 \*\*add - commit\*\* 만 하면 됩니다.

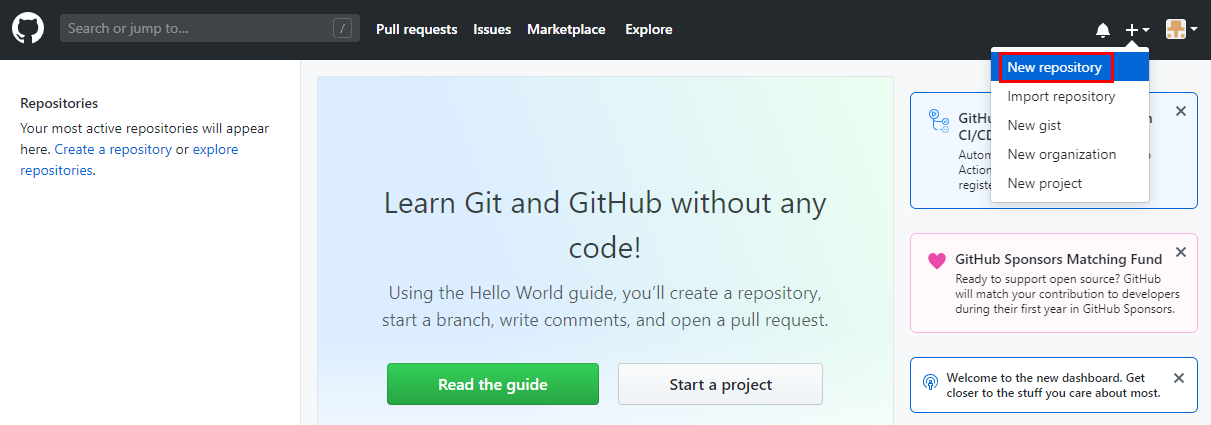
Repository 생성하기

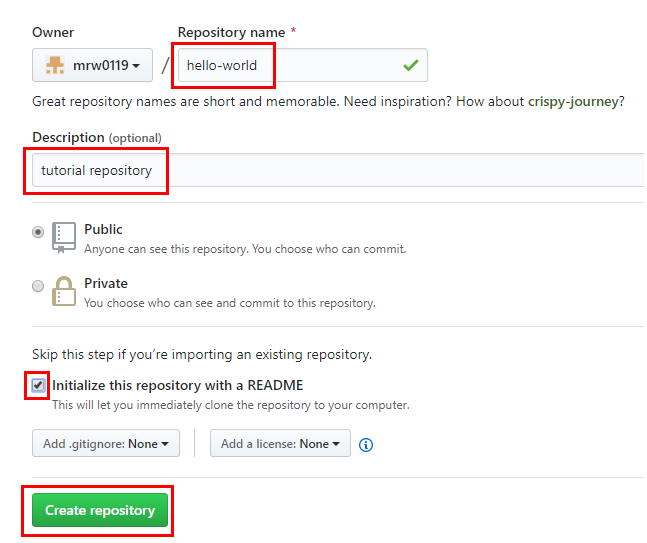
하나의 저장소는 보통 하나의 프로젝트를 만들기 위해 사용된다.

저장소는 폴더와 파일, 이미지, 영상, 데이터셋 등 어떤 것이든 저장할 수 있다.

또는 아이디어, 리소스 다른 사람들과 공유하거나 토론하는 것들도 저장할 수 있다.

1. 우상당의 ‘+’버튼을 클릭하고 New repository를 선택



1. 이름과 repository에 대한 설명을 적는다.
2. Initialize this repository with a README를 체크하고, Create repository 버튼을 클릭한다. 

브랜치 생성하기

브랜칭은 하나의 저장소에서 서로 다른 버전을 동시에 작업하는 방법이다.

기본적으로 저장소는 ‘main’라는 이름의 메인 브랜치를 가진다.

우리는 ‘main’ 외의 다른 브랜치를 만들어서 ‘main’에 커밋하기 전에 수정하고 테스트한다.

새로운 브랜치에서 작업하는 동안 누군가 ‘main’브랜치를 수정한다면 그 업데이트들을 풀(pull)받을 수 있다.

1. 좌상단의 main을 클릭한다.
2. 브랜치의 이름을 작성하고 ‘Create branch : (브랜치 이름) from ‘main’을 클릭한다

