(슬라이드 1) 안녕하십니까 오늘 깃 허브에 대해 발표하게 된 소프트웨어학과 22학번 이이슬입니다.

(슬라이드 2) 깃 허브를 보기 전 먼저 깃을 보자면, 깃은 버전관리 도구입니다. 이와 같이 프로젝트 파일들이 저장되어 있으면 뭐가 진짜 최종본 인지 찾기가 힘든데 이러한 여러 버전들의 변경된 내용만 딱딱 관리하는 도구입니다. 그래서 깃을 이용하게 되면 코드를 언제 누가 무엇을 변경했는지 알 수 있고, 코드를 합치거나 이전 버전으로 돌아가거나 할 수 있습니다.

깃 허브는 코드 저장소인데요. 구글 드라이브, 구글 포토, 아이 클라우드, 네이버 마이 박스 같은 클라우드들에다가 사진이나 파일들을 올리시죠?? 근데 이제 코드를 올려놓을 때는 주로 깃 허브에 많이 올려놓습니다. 코드계의 드라이브 느낌으로 생각하시면 좋을 것 같고요.

(슬라이드 3) 깃 허브에 대해 좀 더 자세히 설명 드리겠습니다. 깃 허브는 소프트웨어 개발 프로젝트를 위한 소스코드 관리서비스(원격 저장소)인데요. 소스코드 열람 및 간단한 버그정리, 버전관리 그리고 sns기능도 있는 호스팅 플랫폼 서비스입니다. 따라서 함께 개발한 소스코드를 공유하고 수정할 때 굉장히 유용하게 쓰입니다.
그리고 깃 허브는 깃을 웹에서 보다 편하게 쓸 수 있도록 만든 도구인데요. 즉, 깃을 활용해서 짠 코드를 공유할 수 있는 공간이라 생각하시면 됩니다.
요약하자면 깃 허브는 소스코드를 저장하고 공유하며 협업하기 좋은 공간입니다.

(슬라이드 4) 다음으로는 Branch인데요 브랜치는 원래 나뭇가지라는 뜻인데요. 버전 관리 시스템에서는 나무가 가지에서 새 줄기를 뻗듯이 여러 갈래로 퍼지는 데이터 흐름을 가리키는 말로 사용합니다. 그리고 브랜치는 독립적으로 어떤 작업을 진행하기 위한 것인데요. 필요에 의해 만들어지는 각각의 브랜치는 다른 브랜치의 영향을 받지 않기 때문에, 여러 작업을 동시에 진행할 수 있습니다. 즉, 자유롭게 일할 수 있고 협업을 쉽고 편리하게 할 수 있도록 도와줍니다.

이를 좀 더 쉽게 생각한다면 작업공간, 연습장의 개념으로 볼 수 있을 것 같습니다. 예를 들어본다면 제가 팀원들과 프로젝트를 하는데 이 프로젝트의 찐 최종 결과의 작업물이 master에 있는데 이 최종 작업물에 다른 사람들이 코드를 막 올려서 충돌이 일어나거나 프로그램에 오류가 생길 수 있기 때문에 안전하게 develop이라는 브랜치를 생성해 여기를 우리의 연습장으로 쓰자!! 하고 만들고 또 develop에서 가지를 뻗어 개개인의 작업공간, 연습장을 만들어 준다고 보시면 될 거 같습니다.

(슬라이드 5) 그 다음에 개개인이 작성한 코드를 복사본 develop브랜치에 올려서 팀원들의 코드와 합치고 최종본에 올려도 된다고 생각이 되면 master브랜치에 합쳐야겠죠?? 이렇게 합치는 것을 병합 즉, merge라고 합니다.

(슬라이드 6) 이제 실습을 해볼 건데, 프로젝트를 할 때 github사용하는데 있어 도움이 되고자 같은 주제로 발표하는 친구와 함께 프로젝트 팀장과 팀원1, 팀원2 역할로 나눠 실습을 진행하였는데, 저는 프로젝트 팀장과 팀원1 역할을 맡아 프로젝트 저장소를 생성하고 소스코드 틀을 생성 후 업로드 한 뒤 팀원의 역할로 돌아가 clone을 하여 프로젝트 코드를 로컬파일에 복사해 p태그

안에 이름 학번 작성 후 merge까지 하도록 하겠습니다.

(슬라이드 7) 실습 순서 이해를 돕기 위해 그림으로 준비해봤는데요 처음에 저장소를 만들고 소스코드 틀을 만들어 업로드 해줍니다. 최종본을 보호하기 위해 복사본인 develop브랜치를 만들어 주고, 제 개인 작업공간도 만들어 주겠습니다. 그 뒤 제 공간에서 p태그를 수정하여 merge하는 이 과정으로 실습할 예정이고
제가 명령어들은 사용할 때 조금씩 설명 드리고, ppt에는 과정별로 명령어 의미까지 있으니 참고해 주시면 좋을 거 같습니다.

(슬라이드 8) 처음에 깃을 설치하시면 환경설정을 해야 하는데요. 깃은 앞에 발표에서 설치했으니 바로 환경설정 하도록 하겠습니다.

- Git bash를 검색해 클릭해줍니다.
- git config --global user.name "your_name" 입력해 주시는데 큰 따옴표 안에는 자기 이름 입력해 주시면 됩니다.
- 다음은 똑같은데 user.name부분만 email로 바꿔 입력해 주시고 큰 따옴표 안에 깃허브 가입시 사용한 이메일을 입력하면 됩니다.
    - git config --global user.email "your_email"
- git config --list 를 입력하여 user.name, user.emil에 입력한 이름과 메일이 들어가있는지만 확인해 주시면 됩니다.

(슬라이드 9) 이제 프로젝트 저장소를 만들어 본건데요.

- 먼저 GitHub에 들어가 로그인을 해줍니다.
- New클릭 -> 이름 작성 -> 만들기 해 주시면 됩니다.
- 이제 팀원도 초대할 건데요
- Invite collaborator 클릭 -> add people 클릭 -> 팀원 초대를 하면 됩니다.
- 아직 팀원이 수락 전일 때는 pending invite가 뜨는데
- 혹시 수락하셨나요...? 네~ 수락하면 뜨지 않습니다.

(슬라이드 10) 다음으로 소스코드를 만들어 업로드 하는 것인데요.

- 배경화면에 소스코드가 저장될 폴더 하나를 만들겠습니다.
- 다음으로 VScode를 열어서 아까 만든 폴더를 열어줍니다.
- 제가 소스코드 틀을 만들어야 하기 때문에 index.html 파일을 만들어 주고,
- 터미널을 열어 주시고 +버튼 화살표를 내려 git bash를 선택해줍니다.
- git init : 초기화 명령어로, 맨 처음에 프로젝트를 올릴 때는 해줘야 합니다.

- git add . : 폴더를 추가하는 명령어로 .은 프로젝트에 있는 모든 파일을 추가하겠다는 명령어 이고
- git status는 선택 사항이고, 이를 입력하면 상태를 알 수 있습니다.
- git commit -m "first commit" : 큰따옴표 안에는 커밋 메시지 적으시면 되고 변경사항에 대해 적어 주시면 됩니다. 이 커밋 명령어는 히스토리를 만들어 주는 건데 변경된 과정을 만드는 것으로 보시면 좋을 것 같습니다.
- 다음은 깃허브로 돌아가 git remote add origin git@github.com:22seul/test.git 이 명령어를 복사해 붙여 주시면 됩니다. 이는 지금 로컬 프로젝트랑 깃허브 저장소와의 연결고리가 없기 때문에 연결고리를 만들어 주는 과정입니다.
- git remote -v 도 선택 사항이고 연결고리가 잘 만들어 졌는지 확인하는 명령어로 복사한 주소가 나오면 성공입니다.
- git push origin master : 코드를 업로드 하는 명령어로 master자리에는 브랜치 이름을 적어주시면 됩니다.

자 이제 깃 허브를 가게 되면 index.html이 업로드 되었고, commit메시지가 제가 적은 메시지로 된 것을 확인할 수 있습니다.

(슬라이드 11) 다음으로는 복사본 develop브랜치를 만들겠습니다.

- git checkout -b develop : 브랜치 만들기
- 현재 브랜치가 깃 허브에는 만들어 지지 않았고, 로컬에서 만들어 졌는데요
- Push 명령어를 통해 깃 허브에 등록하겠습니다.
- git push -> (git push 하고 나오는 명령어 복붙)git push --set-upstream origin develop
- 이제 깃 허브에 들어가 새로 고침을 해보면 develop브랜치가 생긴 것을 볼 수 있죠??

다음으로는 master 브랜치 밑으로 branch생성과 merge를 못하도록 보호하겠습니다.

- 깃허브에 저장소 들어가서 setting들어가신 뒤에 branches클릭 -> 빈 공간에 보호할 브랜치 이름을 적어줍니다. 그 뒤 lock branch랑 Require a pull request before merging 체크해주고 만들어줍니다.

(슬라이드 12) 팀원들과 협업하기 위해 프로젝트 보드를 만들어 주겠습니다.

- Project 클릭 -> link a project -> create new project -> 타입 선택 후 보드를 만들어 주면 됩니다.
- Add item 후에 -> memberA, memberB로 팀원이 해야 할 todo를 만들어 줍니다.
- 그 후 팀원이 프로젝트를 볼 수 있게 ...들어가서 manage access 들어가서 invite collaborators에서 팀원을 추가해 줍니다.

이제 팀장의 역할을 끝났고, 팀원역할로써 소스코드 다운로드를 해줄 텐데요

- 먼저 배경화면에 소스코드를 다운로드 할 폴더를 만들어 주고,
- 폴더에서 마우스 우클릭 해서 터미널을 열어줍니다.
- 다음으로는 깃 허브 저장소 주소를 복사하고,
- 터미널에 git clone 주소 명령어 입력해주면
- 깃 허브에서 소스코드 다운로드가 되어 이렇게 폴더에 코드가 복사가 됩니다.

(슬라이드 13) 다음으로 제가 저의 작업공간인 Member A 브랜치를 만들겠습니다.

- Project 들어가서 -> Member A클릭 -> convert to issue를 클릭해 이슈를 만들어줍니다.
- -> create a branch클릭 후, change branch source에서 develop을 선택 후 브랜치 만들어 주시면 됩니다.
- 그 후 나오는 코드를 복사하여 터미널에 붙여 넣어 줍니다.

이제 코드를 수정하고 업로드 후 develop 브랜치에 머지까지 하겠습니다.

- p태그에 이렇게 학번 이름을 써주고 저장해줍니다.
- git add . -> git commit -m "이름 작성" -> git push origin memberA 아까 설명드린 명령어 들을 입력하여 제 작업공간에 소스코드를 업로드 합니다.
- 머지를 하기 위해 깃허브에 가서 pull requests 클릭 -> new pull request
  -> 제 작업환경에서 develop으로 선택해주고 -> create pull request해서 만들어 주고,
  -> 이와 같이 merging을 클릭해주면 develop에 병합됩니다.

(슬라이드 14) 이상으로 발표를 마치겠습니다. 감사합니다.