교과목 포트폴리오

20212189_양원근

목차 A table of contents

자기주도학습(책 요약형식)

챕터1, 강의계획서

챕터2, GUI에서의 GIT

목차 A table of contents

자기주도학습(책 요약형식)

챕터3, CLI에서의 GIT

챕터4, 실전활용 및 소감

Chapter 1,

강의계획서

동양미래대학교 DONGYANG MIRAE UNIVERSITY

강 의 계 획 서

아시아 직업교육 허브대학

2022 학년도 1학기	전공	컴퓨터정보공학과	학부	컴퓨터공학부
과 목 명	시스템분석설계(종합설계)(2016015-PB)			

강의실 과 강의시간	화:5(3-215),6(3-215),7(3-215),8(3-215)	학점	4
교과분류	실습	시수	4

담당 교수	강환수 + 연구실 : 2호관-706 + 전 화 : 02-2610-1941 + E-MAIL : hskang@dongyang.ac.kr + 면담가능기간 : 화 9시~ 종일, 그외 아무때나
학과 교육목표	
과목 개요	본 교과목은 학과 교과과정을 통하여 습득한 다양한 기술을 종합적으로 활용할 수 있는 능력을 배양하기 위하여 팀 프로젝트방식으로 강의를 진행하는 교과목으로 2학기의 전공필수 과목인 졸업작품 교과목을 수강하기 위해서는 반드시 이수해야하는 선수과목이다. 본 교과목은 프로젝트 팀별로 시스템 개발을 추진하는 것으로, 각 팀은 개발시스템 선정, 팀별 업무분장, 일정계획, 개발방법 및 범위 등을포함하여 시스템 개발을 위한 제반사항을 직접 수행하게 되며, 관련 필요기술에 대한 세미나, 시스템분석기법 및 설계기법, 개발에 필요한샘플 예제소스분석 등도 학습하고 익히게 된다. 각 팀은 시스템개발의 진행과정을 다른 팀에게 수시로 보고, 발표하고 토의하면서 개발시스템에 대한 의견을 수렴한다. -프로젝트팀 구성, 팀명결정 -프로젝트 관련 샘플예제 소스분석(웹,네트워크) -프로젝트 개발계획서 작성 -프로젝트 개발계획서의 일정계획에 의거 프로젝트를 수행 -프로젝트 개발계획서 제출 -프로젝트 최종보고서 제출 및 발표
학습목표 및 성취수준	대학 교육목표와 학과 교육목표를 달성하기 위하여 학과 교과과정을 통해 습득한 기술을 활용한 개발 프로젝트를 수행하여 기술활용 능력배양, 프로젝트 수행 실무능력 향상 등을 도모하며 졸업작품 개발을 위한 각종 설계서, 보고서 및 프로토타입을 개발한다. 최종적으로 매년 열리는 전산학부 학술대회에 개발된 작품을 출품하는 것을 목표로 한다.

	도서명	저자	출판사	비고
주교재	유인물	-	-	

수업시 사용도구	학과 기자재 및 대학 서버시스템
평가방법	팀평가: 팀과제평가 및 보고서 등50%, 개별평가: 출석20%, 중간고사10%, 개별발표 및 참여도20%
수강안내	-2학기의 졸업작품 과목을 이수하기 위한 선수과목임 -제출자료는 한글 또는 워드, 발표자료는 파워포인트로 작성해서 제출함

책 설명

시스템분석설계 과목을 효율적으로 듣고
더 자세하게 GIT과 GitHub를 탐구하기 위해
'팀개발을 위한 Git GitHub 시작하기'라는 책을
구매해 병행하였습니다.
이 교과목 포트폴리오는 제가 이책으로 자기주도학습
을 하면서 정리한점들을 모았습니다.

동양미래대학교 DONGYANG MIRAE UNIVERSITY

강 의 계 획 서

아시아 직업교육 허브대학

	-중간고사 전까지는 팀과제 학습, 이후는 팀 과제 선정, 개발계획서 제출, 과제 개발 순서로 진행됨
1 주차	**[개강일(3/2), 수강정정(3/7~3/8)]**
학습주제	강의소개
목표및 내용	-강의목표소개 -강의계획 및 일정소개 -수업 진행방법 안내 -팀 구성 요청
미리읽어오기	
과제,시험,기타	
2 주차	**[2주, 보강(3/9 -> 6/18)]**
학습주제	프로젝트팀 구성 및 개발계획서 작성방법 설명
목표및 내용	-프로젝트 팀구성 및 팀명 결정 -팀별 프로젝트 내용 토의
미리읽어오기	
과제,시험,기타	-프로젝트 개발계획서 양식 배포 -학과대표는 프로젝트 팀구성 내역과 팀원정보를 메일로 담당교수에게 전달
3 주차	**[3주]**
학습주제	프로젝트 작품주제와 관련된 분석 및 내용논의
목표및 내용	-전년도 졸업작품 사례연구 -개발 프로젝트 방향 결정 -개발 프로젝트 관련 시장 동향 논의
미리읽어오기	
과제,시험,기타	
4 주차	**[4주]**
학습주제	프로젝트관련 네트워크 프로그램 개발사례분석1
목표및 내용	-웹프로그래밍에 대한 이해 -프로젝트를 위한 클라이언트/서버 개념학습 -웹사이트구축을 위한 프로그램소스분석 및 토의 -PHP를 이용한 DB 연동 연습 -회원관리 연습
미리읽어오기	
과제,시험,기타	강의후반에 과제수행후 팀별 발표
5 주차	**[5주]**
학습주제	프로젝트 관련 네트워크프로그램 개발 사례분석2
목표및 내용	-웹사이트구축을 위한 프로그램소스분석 및 토의 -PHP를 이용한 DB 연동 연습 -회원관리를 활용한 게시판 만들기
미리읽어오기	

1주차

강의를 듣고 졸작프로젝트 기획제안 개인과제를 작성하였습니다.

2주차

강의를 듣고 책의 챕터 0과 챕터 1을 공부하였습니다. 팀원을 모으느라 바쁜 시기였습니다.

3주차

강의에서 가르쳐준 소프트웨어 공학 개요와 소프트웨어 개발의 생명 주기, 소프트웨어 개발모델에 대해 공부했습니다.

4,5주차

책의 챕터 2, 챕터3, 챕터4 에 대해 공부했습니다.

동양미래대학교 DONGYANG MIRAE UNIVERSITY

강 의 계 획 서

아시아 직업교육 허브대학

과제,시험,기타	강의후반에 과제수행후 팀별발표
6 주차	**[6주]**
학습주제	프로젝트관련 웹사이트개발 사례분석1
목표및 내용	-웹사이트구축을 위한 프로그램소스분석 및 토의 -PHP를 이용한 웹사이트 전체 구조 및 개별 페이지 분석
미리읽어오기	
과제,시험,기타	강의후반에 과제수행후 팀별발표
7 주차	**[7주]**
학습주제	프로젝트관련 웹사이트개발 사례분석2
목표및 내용	-회원관리와 게시판 통합 및 게시판 수정 -회원관리와 게시판 통합 완성 및 renewal -상용 사이트처럼 멋있게 만들기
미리읽어오기	
과제,시험,기타	강의후반에 과제수행후 팀별발표
8 주차	**[중간고사 : 4/21(목)~27(수)]**
학습주제	중간고사
목표및 내용	중간고사
미리읽어오기	
과제,시험,기타	-개인별 프로젝트 주제 레포트 제출 -팀별 프로젝트 주제선정
9 주차	**[9주]**
학습주제	프로젝트 설계서 및 개발계획서(기본 계획서) 제출
목표및 내용	-프로젝트 설계서 작성(전체 구조 및 UI 등) -프로젝트 개발계획서 협의, 방향결정 -개발계획서 작성, 제출 -프로젝트명, 팀원 업무분장, 프로젝트 개요, 개발방법, 일정계획 등
미리읽어오기	
과제,시험,기타	-프로젝트 설계서 제출 -개발계획서 제출
10 주차	**[10주, 보강(5/5 -> 6/9)]**
학습주제	팀별 프로젝트 설계서 및 개발계획서 발표
목표및 내용	-팀별 프로젝트 개발계획서 발표 -개발계획서 논의 -개발계획서 수정 보완 -개발계획서 확정
미리읽어오기	
과제,시험,기타	-수정보완된 프로젝트설계서 제출 -수정보완된 개발계획서 제출

6주차

분석설계서 초안을 작성했습니다.
그리고 책의 챕터 5에 대해 공부했습니다.

7주차

중간고사를 위해 지금까지 공부한 내용을 정리하고 복습했습니다.

9,10주차

책의 챕터 6과 챕터7에 대해 공부했습니다.

동양미래대학교 DONGYANG MIRAE UNIVERSITY

강 의 계 획 서

아시아 직업교육 허브대학

11 주차	[11주]
학습주제	팀별 프로젝트 개발 추진
목표및 내용	-계획서 일정계획에 따라 시스템 개발추진 -팀원 업무분장 -각 모듈별 개발진도 확인 -시스템 상세기능 협의, 개발 -유사제품에 대한 시장조사 -팀별 사용 요소기술 세미나: 개발기술, 분석기법, 개발기법 등
미리읽어오기	
과제,시험,기타	
12 주차	**[12주, 보강(5/20 -> 6/10)]**
학습주제	팀별 프로젝트 개발 추진
목표및 내용	-계획서 일정계획에 따라 시스템 개발추진 -팀원 업무 분장 -시스템 상세기능 협의, 개발 -유사제품에 대한 시장조사 -팀별 사용 요소기술 세미나: 개발기술, 분석기법, 개발기법 등
미리읽어오기	
과제,시험,기타	
13 주차	**[13주, 보강(6/1 -> 6/14)]**
학습주제	팀별 프로젝트 개발추진
목표및 내용	-계획서 일정계획에 따라 시스템 개발추진 -팀원 업무분장 -시스템 상세기능 협의, 개발 -유사제품에 대한 시장조사 -팀별 사용요소기술 세미나: 개발기술, 분석기법, 개발기법 등 -발표자료 작성
미리읽어오기	
과제,시험,기타	-수정된 계획서 제출 회의록 포함
14 주차	**[14주, 보강(6/6 -> 6/13)]**
학습주제	프로젝트 최종보고서 작성
목표및 내용	-계획서 일정계획에 따라 시스템 개발추진
미리읽어오기	
과제,시험,기타	-최종보고서 제출 및 발표자료 검토

11주차

책의 챕터 8에 대해 공부하고,
교과목 포트폴리오 발표를 준비했습니다.

12주차

책의 챕터 9에 대해 공부했습니다.

Chapter 2,

GUI에서의 버전관리 시스템

Part 2, Git, GitHub 감 익히기

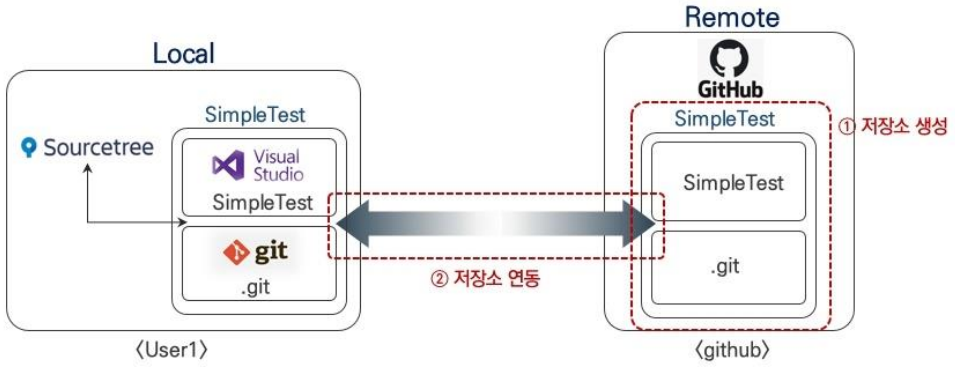
GIT이란?

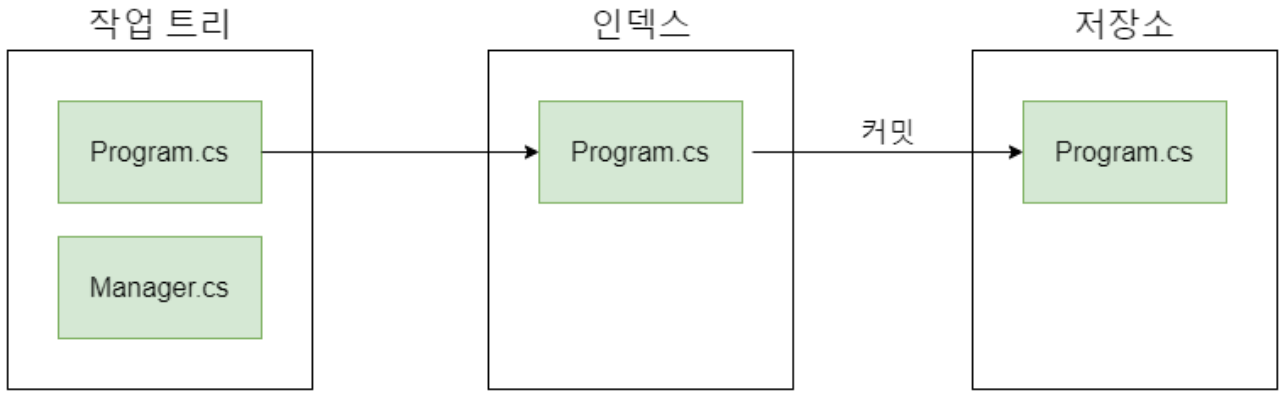
GIT이란 소스코드를 효과적으로 관리하기 위해 개발된 '분산형 버전 관리 시스템'입니다. GIT에서는 소스코드가 변경된 이력을 쉽게 확인할 수 있고, 특정 시점에 저장된 버전과 비교하거나 특정 시점으로 되돌아갈 수도 있습니다.

Git 호스팅 사이트	모기업	특징
GitHub.com	GitHub Inc (Microsoft 에서 인수)	사용자 2,800만 명. 세계 최대 규모의 Git 호스팅 사이트
GitLab.com	GitLab Inc	GitHub에 뒤지지 않는다. NASA, Sony 등 10만 개 이상의 조직이 사용하고 있다. GitLab 프로젝트 자체가 오픈소스여서 직접 서비스 발전에 기여할 수 있다.
BitButcket.org	Atlassian	사용자 600만 명. 이슈 관리 시스템인 지라(Jira)를 만든 Atlassian이 모기업이어서 지라와 연동이 쉽다.

표 0-1 클라우드 서버 서비스

Part 2, Git, GitHub 감 익히기

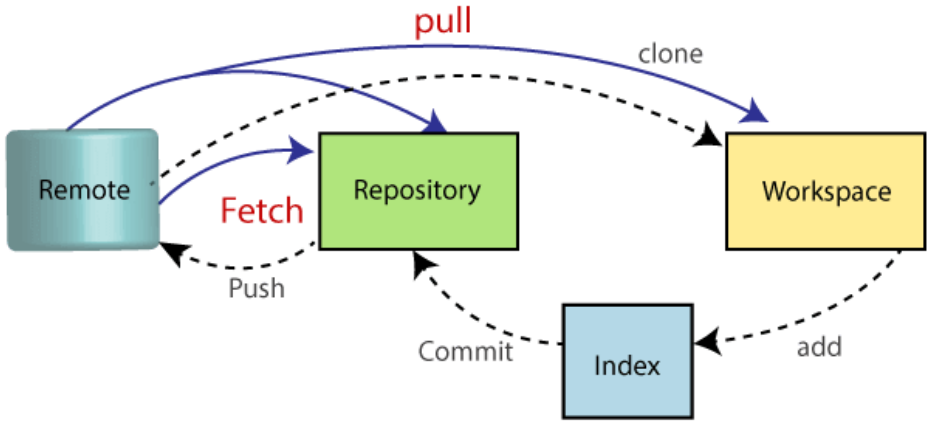




저장소는 파일이나 폴더를 저장해 두는 곳입니다. 2종류의 저장소가 있는데, 원격 저장소는 파일이 원격 저장소 전용 서버에서 관리되며, 여러 사람들과 함께 공유하기 위한 저장소입니다. 그리고 로컬저장소는 내 컴퓨터에 저장되는 개인 전용 저장소입니다.

작업 트리 : 파일 수정, 저장등의 작업을 하는 디렉토리(평소 폴더라고 부름)
스테이지 : 버전으로 만들 파일이 대기하는 곳, 스테이징, 언스테이징
저장소 : 스테이지에서 대기하고 있던 파일들을 버전으로 만들어 저장하는 곳,(리퍼지토리)

Part 2, Git, GitHub 감 익히기



원격저장소(GitHub)에 변경된 파일을 업로드하는 것을 Push라고 합니다. Push를 실행하면, 원격 저장소와 로컬 저장소가 동일한 상태가 됩니다. 그리고 clone은 원격저장소의 파일을 그대로 복제해오는 곳이고, pull을 통해 이를 계속 갱신할 수 있습니다.

저장소에 추가/변경사항을 기록하려면 '커밋'을 해야합니다. 먼저 인덱스는 커밋하기 전에 커밋할 파일을 등록하는 공간인데, 저장소에 커밋하려면 파일을 인덱스에 추가해야만 합니다. 인덱스에 파일을 추가하는 것을 '스테이징'이라고 합니다. 또한 커밋을 하려면 커밋 메시지가 필수로 필요합니다.

Part 2, Git, GitHub 감 익히기

1. 로컬저장소 내에서 커밋하기

아래 코드는 간단하게 로컬저장소 내에서 커밋을 할수 있게 하는 코드들입니다.

$ git

// git의 기본적인 명령어가 나옵니다.

$ git init

// 프로젝트 폴더에서 git 초기화

$ git config –global user.email “”

$ git config –global user.name “”

// 버전 관리를 위해 내 정보 기입

$ git add 파일이름

// 해당 파일을 커밋합니다.

$ git commit -m “설명”

// 해당 커밋에 설명을 붙입니다.

2. 다른 커밋으로 시간 이동하기

아래 코드는 간단하게 로컬저장소를 만드는 코드들입니다.

$ git log

// 지금까지 만든 커밋 확인

$ git checkout 번호

// 해당 커밋으로 코드를 되돌리기

Part 2, Git, GitHub 감 익히기

3. GitHub 원격저장소에 커밋 올리기

원격저장소를 만들고 관리하는 방법입니다.
먼저 GitHub 레포지토리를 만들어야합니다.

$ git remote add origin “”

// 로컬저장소에 원격저장소 주소를 알립니다.

$ git push origin master

// push명령어로 로컬저장소의 커밋을 원격저장소에 올립니다.

4. GitHub 원격저장소의 커밋을 내려받기

원격저장소의 코드와 버전 전체를 내려받는 것을 **클론**이라고 합니다.

$ git clone 원격저장소 주소 .

// 원격저장소의 코드를 내려받습니다.

$ git add 파일이름

// 해당 파일을 커밋합니다.

$ git commit -m “설명”

// 해당 커밋에 설명을 붙입니다.

$ git push origin master

// 원격저장소에 푸시합니다.

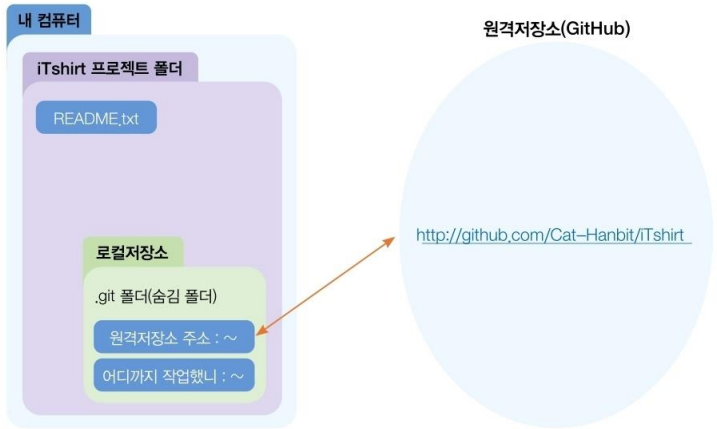
$ git pull origin master

// 원격저장소의 새로운 커밋을 로컬저장소에 갱신합니다.

Part 3, 혼자서 git으로 버전 관리

1. 로컬저장소의 정체는 .git 폴더

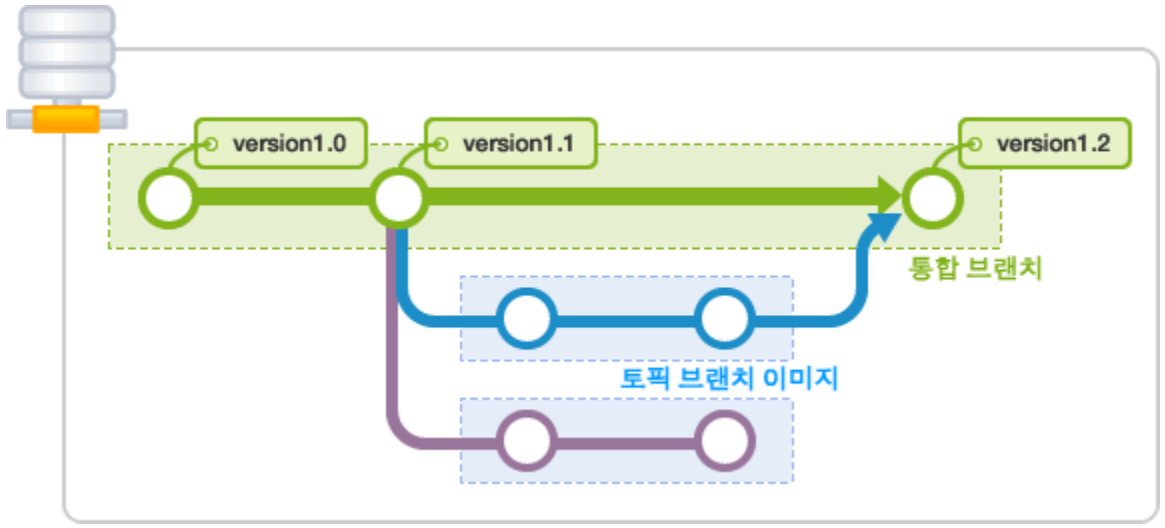
앞서 실습했던 **$ git init** 은 .git폴더를 만들고,
해당 폴더에는 버전 관리한 데이터와
이를 올릴 원격저장소의 주소 등 필요한
정보를 저장합니다.

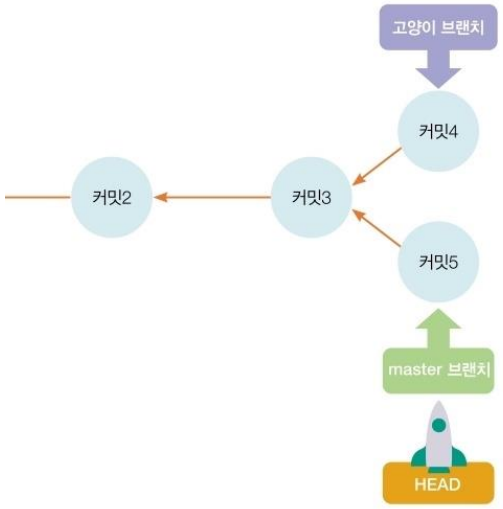


2. 커밋은 차이점이 아니라 스냅사진

Git과 기존 버전관리시스템의 차이로는 Git이
커밋에 차이점뿐만 아니라 **전체 코드를**
저장한다는 점입니다.
얼핏보면 더 비효율적으로 보이지만,
차이점만 저장하는 방식은 맨처음부터 시작해
서 바뀐점을 모조리 계산해야합니다.

Part 4, 여러명이 함께 협업하기.





브랜치란?

특정 기준에서 줄기를 나누어 작업할 수 있는 기능을 "브랜치(Branch)"라고 합니다.
브랜치를 만들지 않고 낡은 코드에 푸시를 하면 오류가 납니다. 브랜치는 포인터로써, 커밋을 가리킵니다..
만약 브랜치가 갈라진다면 어떻게 넘나들수 있을까요? 바로 HEAD라는 포인터로 가능합니다. 또한 기본 브랜치는 'master'입니다.
만약 개발을 한다면 master에서 새로운 브랜치를 만들어서 각자의 브랜치에서 개발하다가 개발이 완료되면 브랜치 작업물을 합치면 됩니다..
다음장에서 합치는 법을 정리했습니다.

Part 4, 여러명이 함께 협업하기.

1. Merge(병합)

병합 커밋 – 커밋 2개를 병합
빨리 감기 병합 – 새로 만들 필요 X 상태만 바꾸면 되는 것
충돌 상태 – 두 커밋이 충돌해서 병합이 안됨

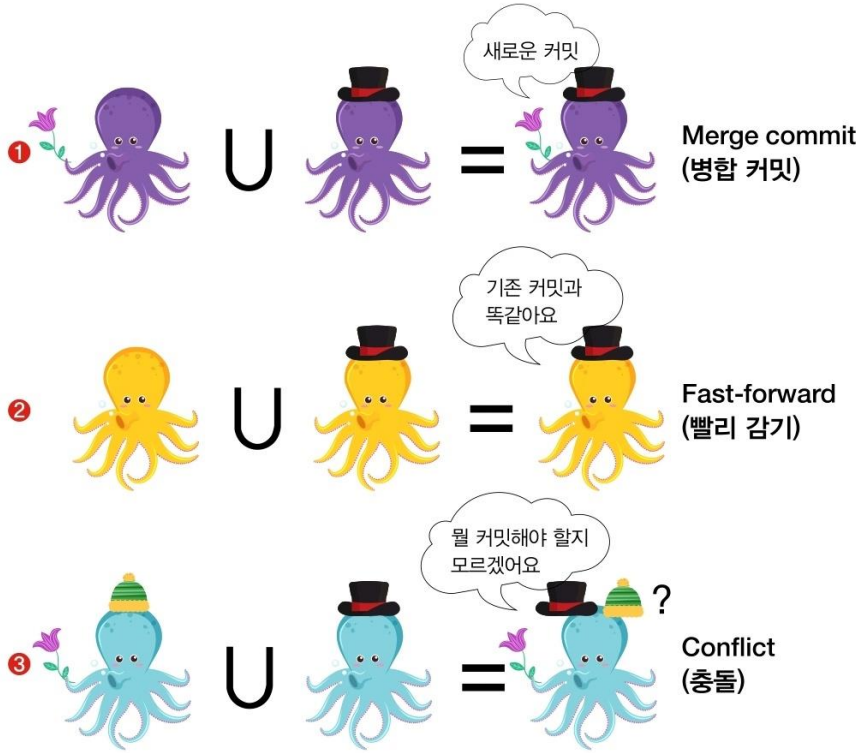
$ git checkout 브랜치 이름

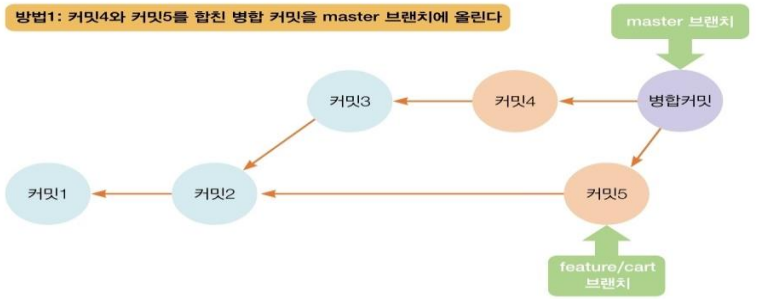
// 현재 브랜치를 떠나 새로운 브랜치로 이동

충돌 상태는 소스트리에서 직접 수동으로 편집하여 해결 가능함. 그 후 병합하고 문제가 없다면 병합 커밋을 master 브랜치에 반영함.

이 병합을 예의바르게 하는 것이 풀 리퀘스트

Create pull request
Merge pull request





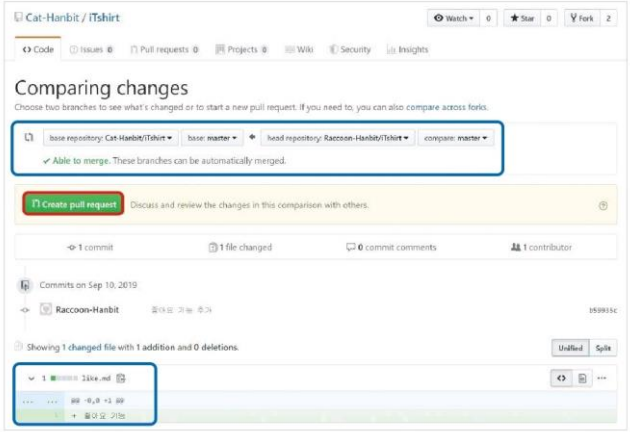
Part 4, 여러명이 함께 협업하기.

1. 포크

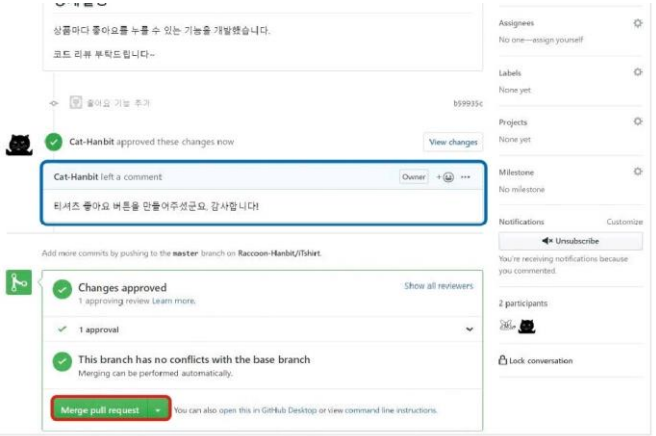
원본저장소의 소유자 입장에서는 협력자가 늘어날수록 원본저장소를 관리하기 힘듭니다.
따라서 협력자가 많을경우 "풀 리퀘스트"를 활용합니다.(브랜치와 동일)

포크는 브랜치를 포함한 원본저장소의 모든 커밋 이력을 새로운 원격저장소로 통째로 복사하는 것을 말합니다.

	의의	편리한 점	불편한 점
브랜치	하나의 원본저장소에서 분기를 나눈다.	하나의 원본저장소에서 코드 커밋 이력을 편하게 볼 수 있다.	다수의 사용자가 다수의 브랜치를 만들면 관리하기 힘들다.
포크	여러 원격저장소를 만들어 분기를 나눈다.	원본저장소에 영향을 미치지 않으므로 원격저장소에서 마음껏 코드를 수정할 수 있다.	원본저장소의 이력을 보려면 따로 주소를 추가해야 한다



Create pull request

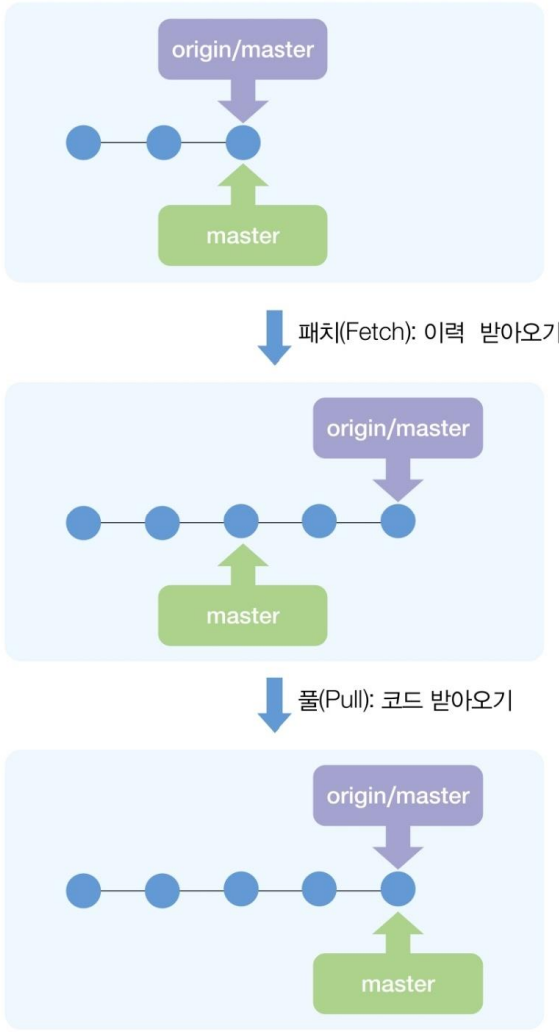


Merge pull request

Part 4, 여러명이 함께 협업하기.

2. 패치

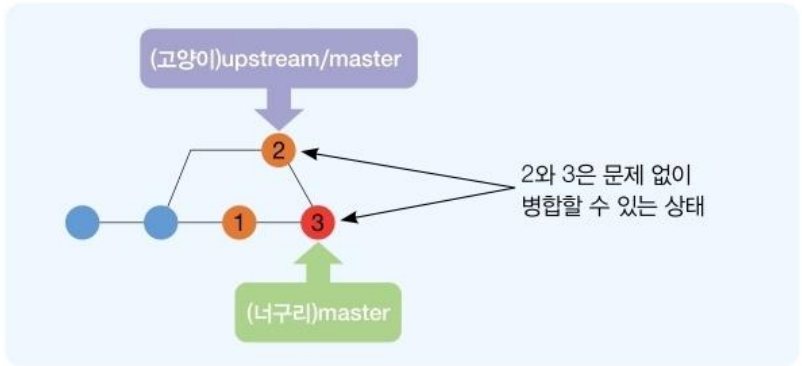
Fetch는 간단히 말하자면 '새로고침' 기능입니다. 패치를 하면 원본저장소 이력을 업데이트 합니다. 이력만 가져오기때문에 내 코드에는 아무런 영향이 없습니다. 최신 코드를 내 코드에 반영하는 Pull과는 다릅니다.



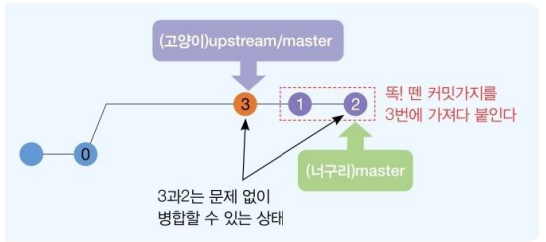
Part 4, 여러명이 함께 협업하기.

Rebase란?

- 커밋의 베이스를 똑 떼서 다른 곳에 붙이는 것을 "리베이스(rebase)라고 합니다. 이 방법은 병합 커밋이 생기는 앞에서 설명한 방식과는 달리 깔끔하게 풀 리퀘스트를 보낼 수 있습니다.



단, 강제 푸시를 사용해야합니다.



Part 5, 실무사례와 함께 git 다루기

1. Amend

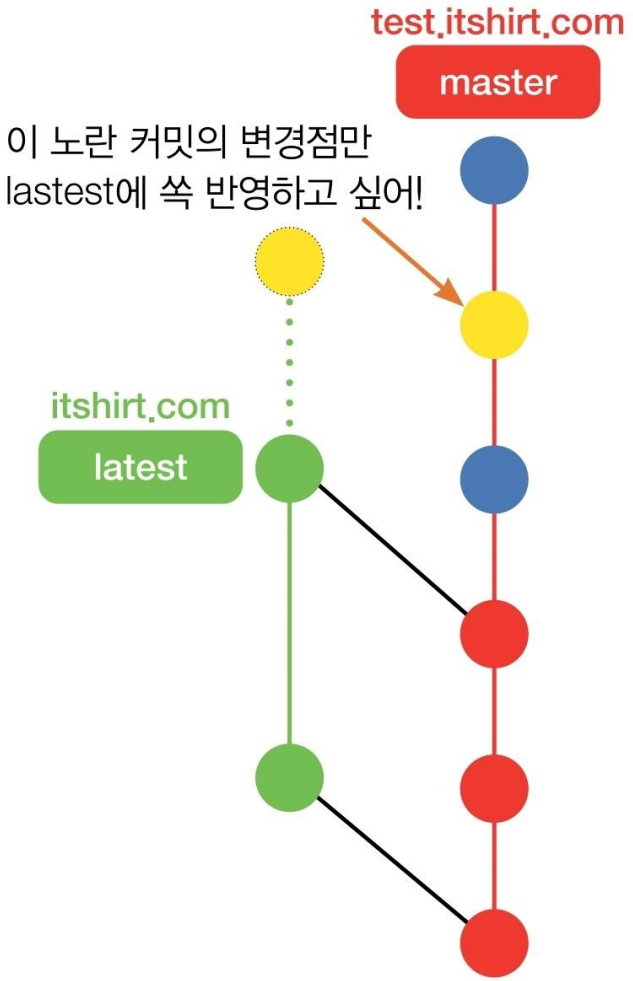
커밋을 이미 만들었는데, 추가할 파일이 또 있다는 것을 뒤늦게 알았을 때 유용합니다. 커밋 메시지와 파일 내용 변경에서도 유용하게 사용 가능합니다.

커밋 옵션 - 마지막 커밋 정정 클릭

2. Cherry-Pick

브랜치와 브랜치 사이에서 특정 커밋만 반영할때 매우 유용합니다.

커밋 – 체리 픽



Part 5, 실무사례와 함께 git 다루기

3. Reset

옛날 커밋으로 간단히 상태만 돌릴 수 있는 매우 자주쓰이는 기능입니다.

'이 커밋까지 현재 브랜치를 초기화'
[HARD], [SOFT], [MIXED]

HARD는 깔끔, SOFT와 MIXED는
돌리긴하지만 변경사항이 남음
자세히 파고들면 SOFT는 변경사항이 스테이지 위로, MIXED는 아래에 있습니다.

4. Revert

Reset과 거의 같지만 '변경사항 되돌리기'라는 새로운 커밋을 추가합니다.

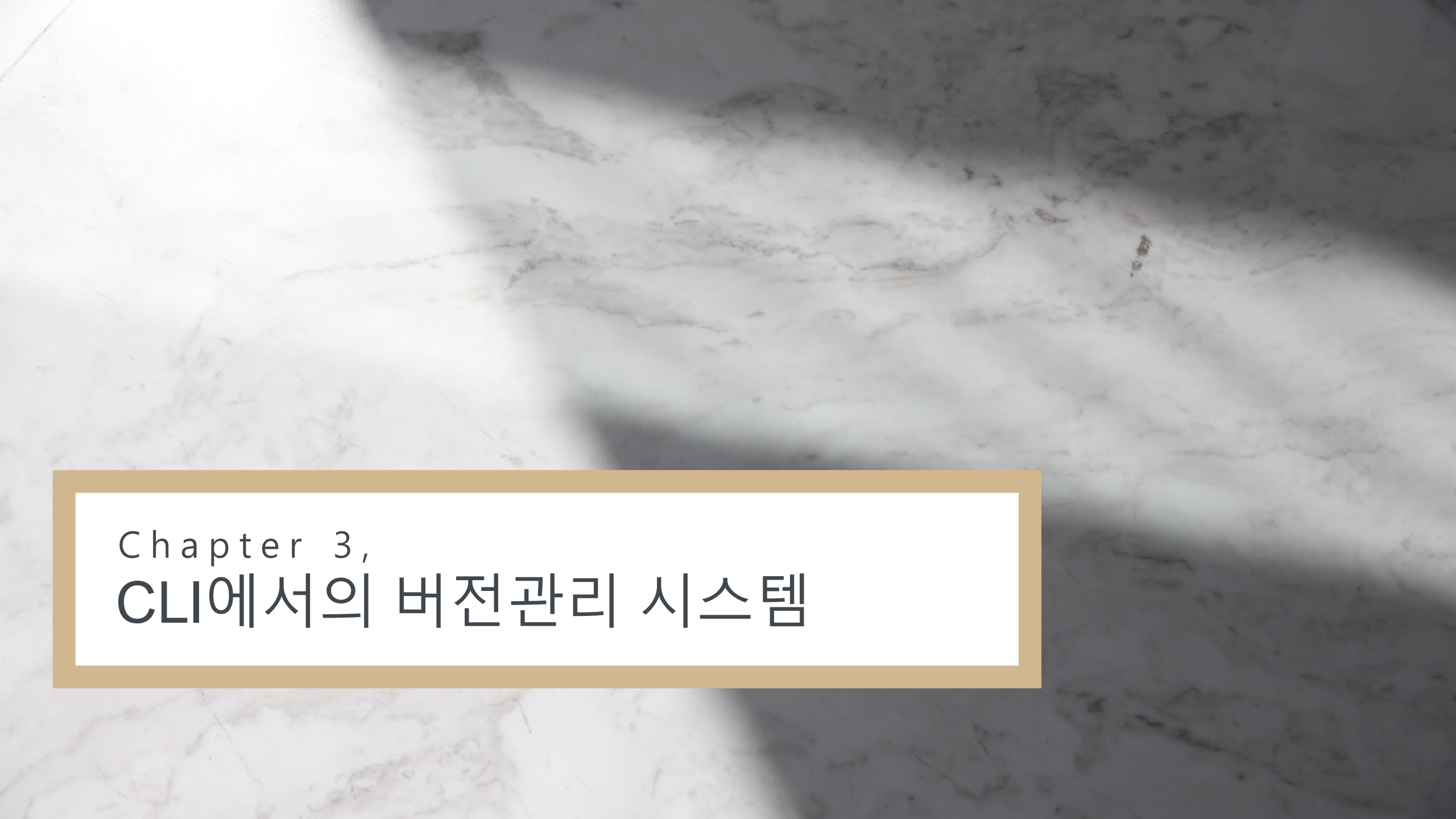
커밋 - 커밋 되돌리기

5. Stash

아직 커밋하지 않은 변경사항이 있을때,
커밋하기 애매하면 이 변경사항을 잠깐 저장하고 다음에 다시 꺼내쓸 수 있는 기능입니다.

상단메뉴 – 스태시(tracked상태만)

스태시 – 스태시 적용



Chapter 3,

CLI에서의 버전관리 시스템

Part 6, 1에서 수행했던 기본 명령어

명령어	설명
pwd	현재 폴더의 위치를 확인합니다.
ls -a	현재 폴더의 파일 목록을 확인합니다. -a 옵션을 이용해 숨김 파일도 볼 수 있습니다.
cd	홈 폴더로 이동합니다. 홈 폴더는 사용자 이름과 폴더명이 같고 내 문서 폴더의 상위 폴더입니다.
cd <폴더이름>	특정 위치의 디렉토리로 이동합니다.
cd ../	현재 폴더의 상위 폴더로 이동합니다.
mkdir <새폴더이름>	현재 폴더의 아래에 새로운 폴더를 만듭니다.
echo "Hello Git"	메아리라는 뜻
	화면에 " " 안의 문장인 "Hello Git"을 표시합니다.

표 6-1 Git Bash 기본 명령어

옵션의 종류

시스템 환경 옵션은 pc전체의 사용자를 위한 옵션, 지역 옵션은 현재 Git저장소에서만 유효한 옵션

로컬저장소를 위한 새 폴더 만들기

```
$ mkdir hello-git-cli
// hello-git-cli 폴더 생성
$ git status
// Git 저장소의 상태를 알립니다.
$ git init
// Git 저장소 생성(로컬저장소)
$ git config –global <옵션명>
// 지정한 전역 옵션의 내용을 살펴봄
$ git config –global <옵션명> <새로운값>
// 지정한 전역옵션의 값 새로 설정
$ git config –global --unset <옵션명>
// 지정한 전역 옵션 삭제
$ git config --list
// 모든 옵션을 살펴보기
```

Part 6, 1에서 수행했던 기본 명령어

기본적인 git 명령어들

$ git add 파일1 파일2 ...
// 파일들을 스테이지에 추가
$ git commit
// 스테이지에 있는 파일들 커밋.
$ git push [원격저장소] [브랜치이름]
// 커밋들을 원격저장소에 업로드
$ git pull
// 원격저장소의 변경사항을 반영
$ git fetch [원격저장소] [브랜치이름]
// 원격저장소의 브랜치와 커밋을 동기화
$ git merge 브랜치이름
// 지정된 브랜치의 커밋들을 반영
$ git reset [파일명]
// 스테이징 취소

원격저장소 관련 CLI 명령어

$ git remote add <원격저장소>
// 원격저장소를 등록합니다.
$ git push –u origin master
// push와 동시에 업스트림 지정
$ git clone <저장소주소>
// 프로젝트를 복제해 옵니다.
$ git pull
// 원격저장소의 변경사항을 반영

Part 7, 브랜치 생성 및 조작하기

브랜치에 대해 꼭 기억해야하는 내용

1.커밋하면 커밋 객체가 생깁니다. 커밋 객체에는 부모 커밋에 대한 참조와 실제 커밋을 구성하는 파일 객체가 들어 있습니다.

2. 브랜치는 논리적으로는 어떤 커밋과 그 조상들을 묶어서 뜻하지만, 사실은 단순히 커밋 객체 하나를 가리킬 뿐입니다.

브랜치 관리 명령어 정리

$ git branch
// 로컬저장소의 브랜치 목록을 봄

$ git branch <브랜치이름>
// 새로운 브랜치 생성

$ git branch -r
// 원격저장소의 브랜치 목록을 봄.

$ git checkout <브랜치이름>
// 특정 브랜치로 체크아웃할때 사용

$ git merge <대상 브랜치>
// 현재 브랜치와 대상 브랜치를 통합

$ git rebase <대상 브랜치>
// 내 브랜치의 커밋을 대상 브랜치에 재배치

$ git branch –d <브랜치이름>
// 브랜치 삭제

Part 7, 브랜치 생성 및 조작하기

Reset --hard로 브랜치 되돌리기

$ git reset –hard <이동할 커밋 체크섬>

// 현재 브랜치를 지정한 커밋으로 옮김

커밋 체크섬은 git log를 통해 확인 가능, 번거롭다면 HEAD~<숫자>같은 약칭을 사용가능

$ git reset –hard HEAD~2

// 2단계 위의 커밋으로 되돌립니다.

$ git reset –hard <이동할 커밋 체크섬>

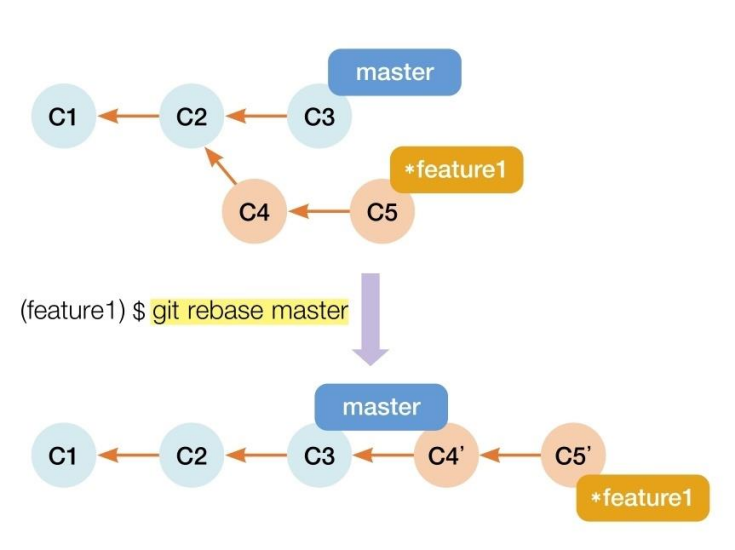
// 위에있는 2번째 커밋으로 되돌립니다.

	3-way 병합	rebase
특징	머지 커밋 생성	현재 커밋들을 수정하면서 대상 브랜치 위로 재배치함
장점	한 번만 충돌 발생	깔끔한 히스토리
단점	트리가 약간 지저분해짐	여러 번 충돌이 발생할 수 있음

표 7-2 3-way 병합과 rebase

CLI로 3-way 병합하기

직접 비주얼스튜디오 코드에서 수정합니다.
다만 이렇게 해결할시 병합커밋이 생성되기때문에 코드가 더러워지는데, 이때 트리를 깔끔하게 하고 싶으면 rebase를 사용합니다.
먼저 rebase한후 빨리감기 병합을 사용합니다.



Part 8, git 내부 동작 원리

정리

1. Git add 명령을 수행하면 워킹트리의 내용을 스테이지에 추가한다.
2. Git commit 명령을 수행하면 스테이지의 내용으로 새로운 커밋을 만든다.
3. 커밋 이후 git status 명령을 내리면 clean한 상태임을 표시해주는데, 이 상태는 워킹트리, 스테이지, HEAD 커밋들이 모두 동일한 내용을 담고 있다는 뜻이다.
4. 커밋 객체는 트리 객체와 blob객체의 조합으로 이루어져 있다.
5. 커밋 객체는 부모 커밋에 대한 참조를 가지고 있다.
6. 브랜치를 생성하면 단순히 브랜치 파일 하나를 추가한다.
7. 브랜치를 체크아웃하면 HEAD를 해당 브랜치로 변경하고 브랜치가 참조하는 커밋의 내용으로 스테이지와 워킹트리의 내용을 변경한다.

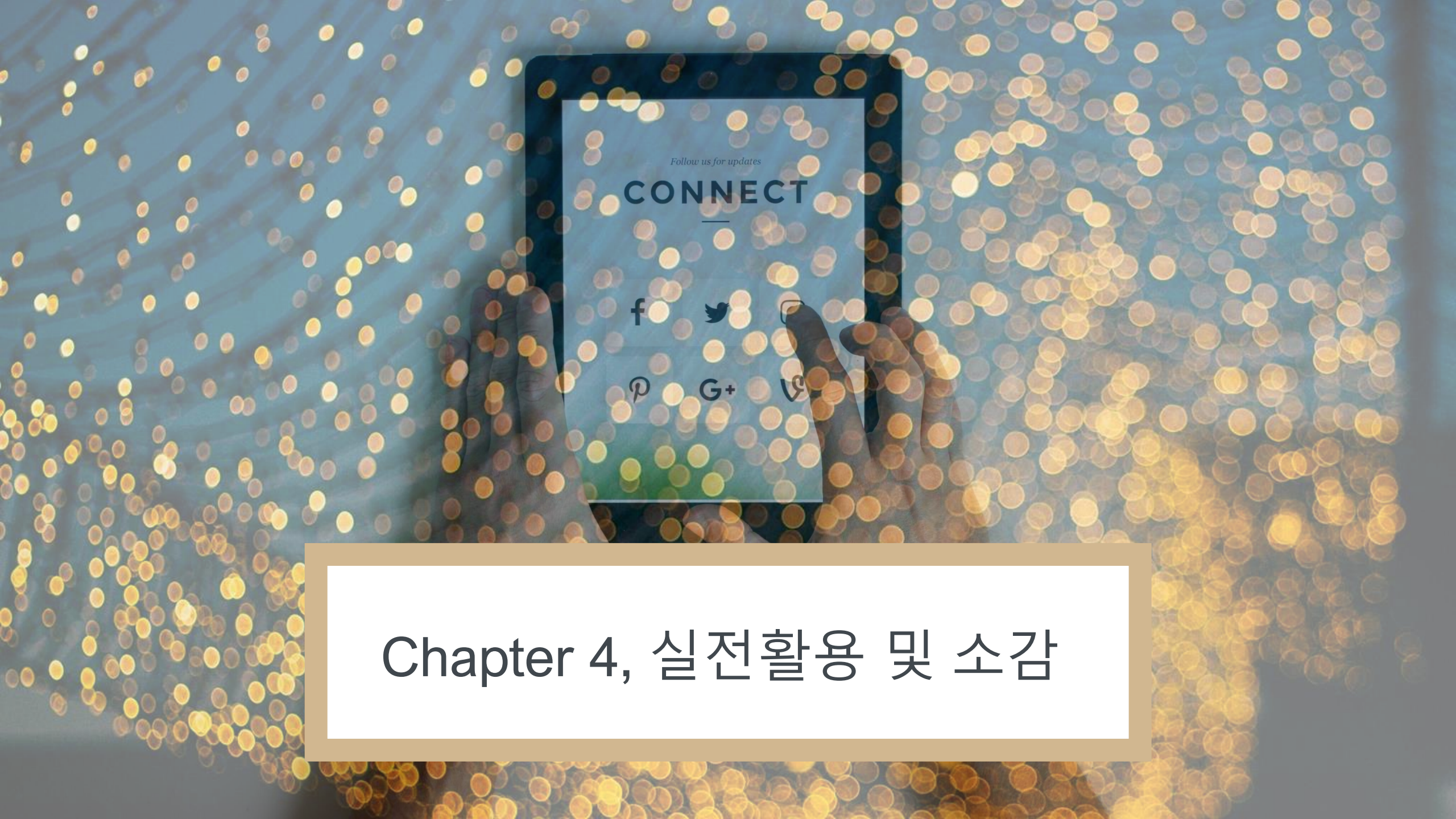
Part 9, 인증 기능 살펴보기

정리

1. 소스트리에서 인증하기

2. 자격증명 관리자 – WINDOWS 자격 증명에 GITHUB의 아이디와 패스워드가 저장됩니다. 따라서 새로운 계정을 적용하려면 자격증명 관리자에서 기존 자격 증명을 제거하고,
$ git push 를 사용합니다.

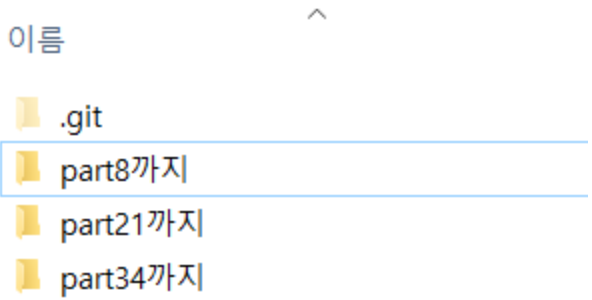
3. SSH 프로토콜은 1995년에 만들어졌는데 매우 안전합니다. SSH는 공개키/비밀키 방식을 사용하는데, $ssh-keygen으로 SSH 키를 만들 수 있습니다. 그리고 이렇게 생성된 키를 GitHub에 등록해야 사용할 수 있습니다. https://github.com을 git@github.com:으로 바꾸면 됩니다.



Chapter 4, 실전활용 및 소감

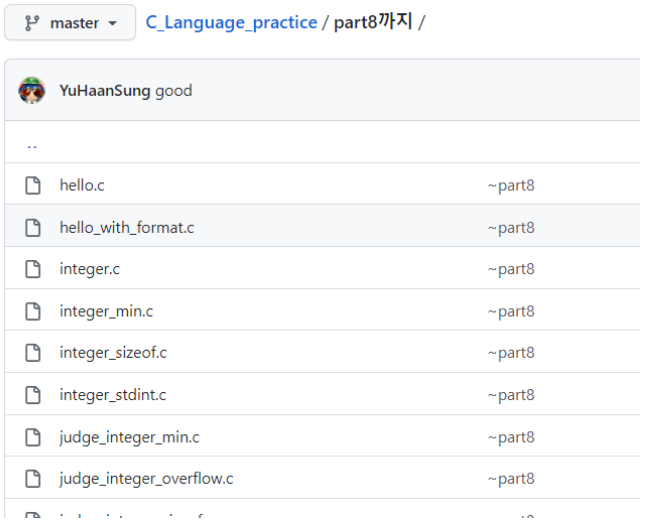
Part 10

실전 활용



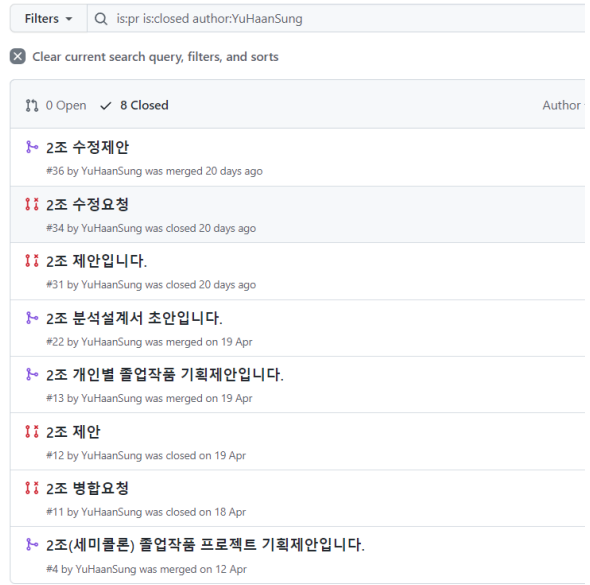
로컬저장소

개인적인 C언어 연습 코드를
로컬저장소로



원격저장소

이를 Git으로 조작하여
로컬저장소와 연동함.



과목에서 활용

이렇게 배운 Git으로
수업을 수월하게 진행

소감

이책으로 기본적인 CLI명령들을 살펴보았습니다. 처음 이 책을 만났을때는 생각보다 책 분량이 많아서 당황스러웠지만, 먼저 GUI(소스트리)로 공부하고 CLI방식을 공부하니까 확실히 이해가 편했습니다.

제가 교과목 포트폴리오로 GIT을 공부하게 된 이유는 이 과목이 시작하고 충돌이나 브랜치 실수가 많아 저희 세미콜론조원들에게 피해를 끼칠 수 있다는 생각에서 였습니다.

이 책과 강의를 함께 병행하고, 개인적인 C언어 연습에서도 GIT을 활용하여 GIT에 대한 이해가 많이 쌓였으니 졸업작품 프로젝트에서 안드로이드 스튜디오와 GitHub를 연동하여 팀원들과 체계적으로 개발할 것입니다.

Thank you