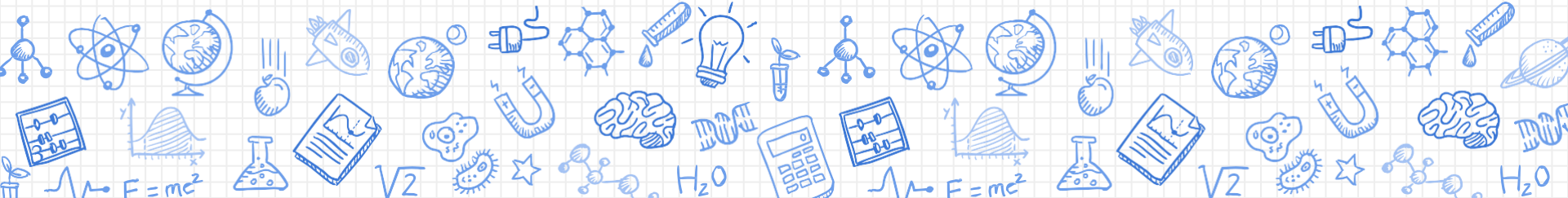
컴퓨터정보공학과 20202254 박현지

졸업 작품 포트폴리오

Contents

깃허브



Git

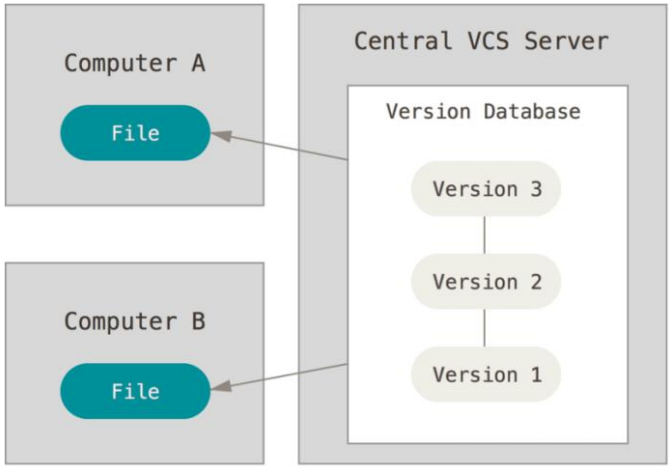
✘ Git이란? - 버전관리, 협업, 백업

- ✓ 컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간 해당 파일들의 작업을 조율하기 위한
  분산 버전 관리 시스템

- ✓ 소프트웨어개발에서 소스 코드 관리에 주로 사용되지만 어떠한 집합의 파일의 변경사항을 지속적으로 추적하기 위해 사용 가능

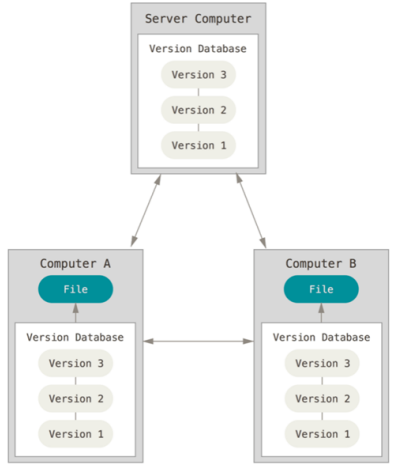
중앙집중관리 vs 분산버전 관리

✘ 중앙집중관리



복구 불가능

✘ 분산버전 관리



분산된 저장소에 의해 복구가 가능

저장소와 작업디렉토리 확인

- git add
- git init
- git status
- git commit -am
- git log -p [파일]

git add & git init

✗ git add

- Untracked에서 Index로
- Modified에서 Index로

✗ git init

새로운 저장소를 만들거나
다른 저장소의 URL을 이용해
저장소를 복사

$ git init
$ git init .
현 폴더에 저장소 생성

$ git init [project-name]
지정된 이름의 폴더를 만들면서 저장소 생성

git status & git commit

✗ git status

- Git 상태 정보 보이기
- 로컬 저장소에 추가할 준비가 된 모든 수정된 파일을 나열
- on branch master : 현재 master 브랜치라는 뜻
- no commits yet : 아직 커밋한 파일이 없음
- nothing to commit : 현재 커밋할 파일이 없음
- untracked files : 한 번도 (버전관리)하지 않은 파일이 존재함
- Changes to be committed : 커밋 가능
- Changes not staged for commit
- 커밋을 위해 준비되지 않은 수정 사항
- 변경된 파일이 아직 스테이지에 올라가지 않았음

✗ git commit

Index에서 깃 리포지토리(저장소)로
$ git commit –m “메시지”
$ git commit
메시지를 입력하기 위한 디폴트 에디터에서 입력

작업디렉토리에서 index를 거치지 않고 바로 깃 레포지토리(저장소)로
$ git commit –am “메시지”
$ git commit –am
메시지를 입력하기 위한 디폴트 에디터에서 입력
$ git commit –a -m “메시지”
$ git commit –a -m
메시지를 입력하기 위한 디폴트 에디터에서 입력

git log

✗ git log

- 커밋 기록(history) 보기
- $ git log

- 특정 파일의 커밋 기록 보기
- $ git log –p [file]

- Commit ID: 해쉬코드
- "ec72682f07e02f85648bfb7afb82fd097655f08f" 값
- 유일한 값
- commit 할 때 SHA-1 해시를 사용하여 만들어진 체크섬(hash) 데이터
- 40자 길이의 16진수

$ git show [commit]

✘ 특정 커밋에 포함된 변경 사항과 메타데이터를 표시

$ git show: 현재 브랜치의 가장 최근 커밋 정보를 확인함

$ git show : 커밋해시값
특정 커밋 정보를 확인함

$ git show HEAD: HEAD 포인터가 가리키는 커밋 정보를 확인함

$ git show 커밋해시값 또는 HEAD^
^표시 한 개면 한 개전 두 개면 두 개전, 갯수로 얼마나 이전 값인지 알 수 있음

$ git show 커밋해시값 또는 HEAD~숫자
~숫자 는 명시적으로 몇개 전인지 표시

$ git diff

- 파일의 어떤 내용이 변경되었는지 차이점을 비교
- Working Directory와 Staging Area간의 비교도 가능
- commit간의 비교, branch간의 비교도 가능

❖ $ git diff :commit된 파일 상태와 현재 수정 중인 상태 비교

❖ $ git diff –staged:commit된 파일 상태와 add된 파일 상태 비교

❖ $ git diff [비교할commit해쉬1] [비교할commit해쉬2]
commit간의 상태 비교하기 - commit hash 이용

❖ $ git diff HEAD HEAD^
-commit간의 상태 비교하기 - HEAD 이용
-가장 최근의 커밋과 그 전의 커밋을 비교한다

$ git config –global alias.새명령어 ‘원명령어’

명령어를 다른 이름으로 지정

```
$ git config --global alias.last 'log -1‘
$ git last

$ git config --global alias.st 'status‘
$ git st
```

Git log

commit history 확인

Author 영역의 이름과 이메일 주소는 git config 명령을 통해 세팅했던 user.name / user.email 값이 표기

옵션

-p : 변경사항 확인
--oneline : 커밋 메세지만 한 줄씩 표시
--all : 모든 브랜치 로그 표시
--graph : 브랜치 트리 그래프 표시
-n : 최근 n개 보이기

$ git commit --amend -m "new message"
$ git checkout [file]
$ git revert

$ git commit --amend -m "new message"
새로운 커밋 메시지로 수정

$ git checkout [file]
파일을 수정 전의 파일로 복구
    $ git restore [file]

$ git revert
현재 HEAD를 특정 시점의 commit 상태로 변경하는 commit을 추가로 수행
    과거의 모든 커밋은 그대로 유지 관리되며
    새로운 커밋을 추가해 측정 시점으로 이동

Git의 4가지 영역

1. Working Directory(작업영역)
   - 프로젝트 디렉토리이며, 개발자가 직접 코드를 수정하는 공간을 의미
   - .git을 제외한 모든 영역에 해당
2. Index (Staging Area)
   - Working Directory 에서 Repository로 정보가 저장되기 전 준비 영역
   - .git/index 파일로 관리
3. Repository(저장소)
   - 파일이나 폴더를 변경 이력 별로 저장해두는 영역
   - .git 디렉토리 내에 존재
   - Local, Remote Repository로 구분
4. Stash(임시영역)
   - 임시적으로 작업 사항을 저장해두고, 나중에 꺼내올 수 있는 영역

$ git revert <commitid>

- ✓ 현재 HEAD를 특정 시점 commit 이전 상태로 변경
- ✓ <commitid> 바로 이전으로 이동commit을 추가로 수행
- ✓ 과거의 모든 커밋은 유지 관리되며 새로운 커밋을 추가해 특정 시점으로 이동

$ git reset

✗ 과거 커밋 지점으로 이동하고, 이동된 이후의 커밋은 삭제하는 명령어

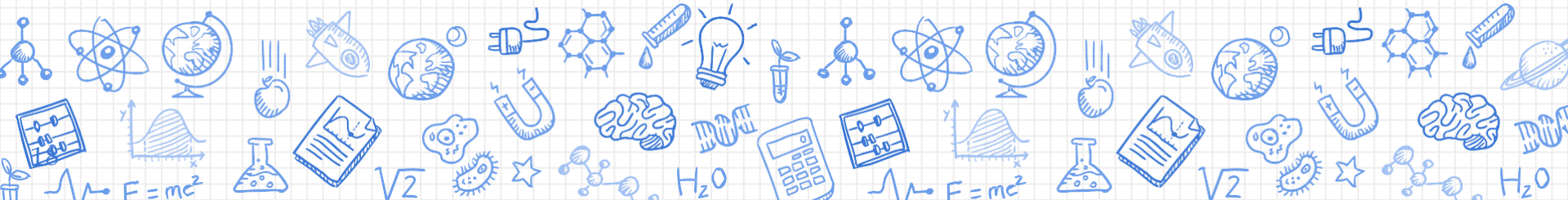
※ git reset –hard
- 해당 커밋ID의 상태로 이동하
- Working Directory와 Index영역 모두 초기화

※ git reset –mixed
- 해당 커밋ID의 상태로 이동하고,
- Index영역은 초기화되고
- Working Directory는 변경되지 않습니다.

※ git reset –soft
- 해당 커밋ID의 상태로 이동하고,
- Index영역과 Working Directory 모두 변경되지 않고

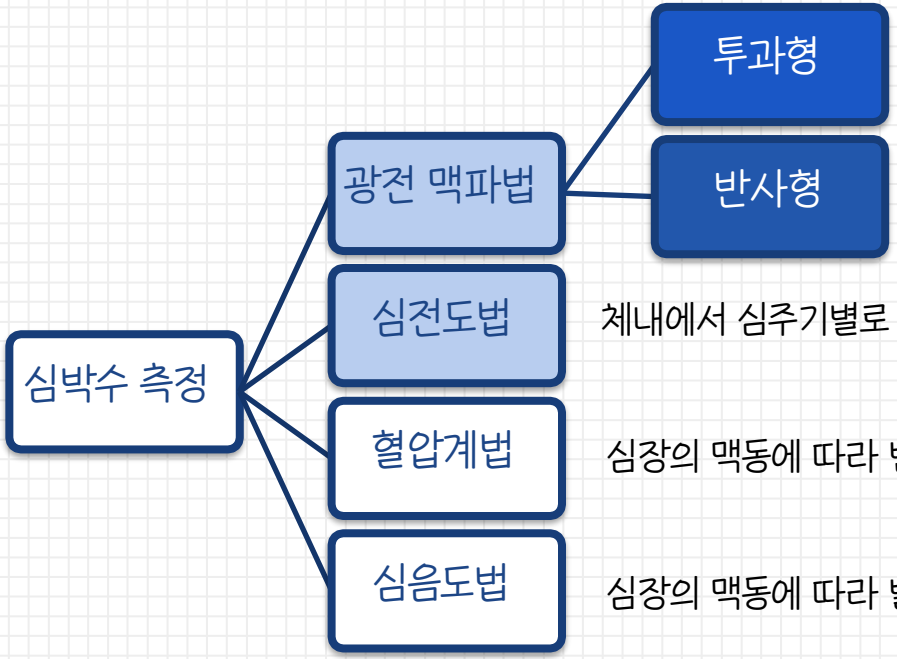
프로젝트 수행 중 학습 내용

-아두이노



ECG센서/맥파센서

ECG센서/맥파센서



체내에서 심주기별로 생성되는 전기 펄스를 계측하는 방법

심장의 맥동에 따라 변하는 혈관압의 변화를 계측하는 방법

심장의 맥동에 따라 발생하는 소리를 계측하는 방법

ECG센서/맥파센서

ECG센서

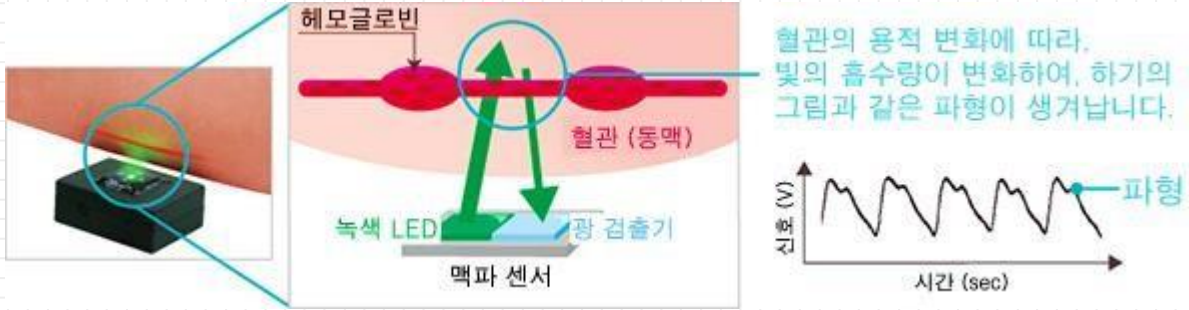
투과형 맥파센서

반사형 맥파센서

ECG센서/ 맥파센서 – 반사형 맥파센서

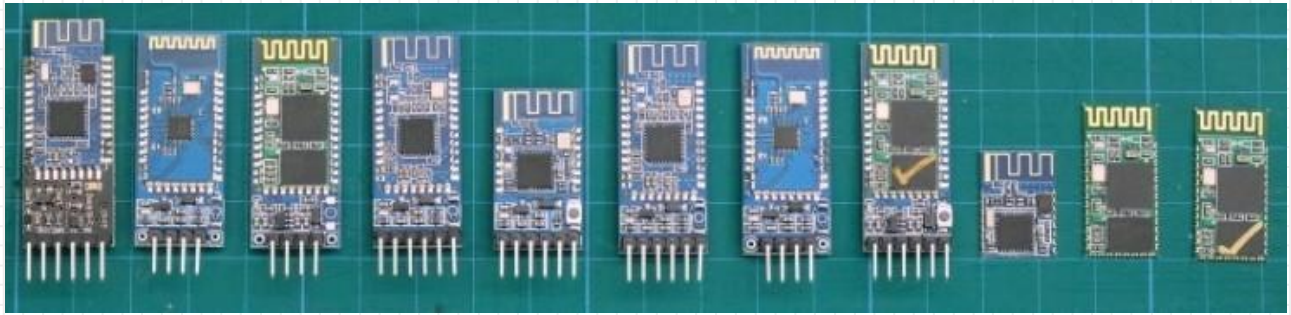
✘ 반사형 맥파 센서

- 적외선 및 적색광,녹색 파장의 빛을 생체에 조사
- 포토 다이오드 또는 포토 트랜지스터를 사용하여 생체 내부에서 반사된 빛을 계측
- 동맥의 혈액의 산화 헤모글로빈이 입사광을 흡수하는 특성을 이용해
- 심장의 맥동에 따라 변화하는 혈류량을 측정해 맥파 신호를 계측
- 반사광의 계측이므로, 투과형과 반대로 측정 부분에 한정 X



블루투스 모듈

블루투스 모듈의 종류



Bluetooth 2.0/2.1 EDR Modules

- HC-06 (ZG-B23090W)
- HC-05 (ZG-B23090W)
- SPP-C HC-06 / BT06 HC-06
- HC-06 zs-040 hc01.com v2.0
- HC-05 zs-040 hc01.com V2.1
- HC-05 FC-114 and HC-06 FC-114
- HC-05 and HC-06 zs-040 Bluetooth modules

Bluetooth 4 / BLE Modules

- HM-10
- HM-11
- BT05-A mini BLE Bluetooth V4.0 iBeacon
- AT-09 Bluetooth V4.0 CC2541

HC-06/HC-05 비교

	HC-06	HC-05
AT모드	NO line ending(CR LF 해제) 상태에서 AT모드가 자체 활성화	Default로 AT 모드가 아니라 34번 핀에 High를 주어야AT모드로
기본AT모드 보율 설정	9600	38400
	마스터 및 슬레이브 모드 전환 X	마스터 및 슬레이브 모드 전환 O
외부 버튼 유무	없음	있음

Credits

참고문헌:

✘ 깃허브

https://juyoungit.tistory.com/391

✘ ECG센서

https://imotions.co m/guides/electro cardigraphy-ecg/

✘ 블루투스

https://m.blog.naver.com/damtaja/222001696734

HC Serial Bluetooth Products User Instructional Manual

THANK YOU

27