Python BackEnd

Web Development By Django & Flask

1. Python 심화

- 가상환경 설정

```
$ python3 -m venv myvenv # 원하는 가상환경 이름
$ source ./myvenv/bin/activate # 가상환경 활성화
```

- 가상환경 폴더 안에서 작업 진행

1. 할당과 복사

- 파이썬에서는 데이터도 객체다.
  - 변수에 데이터가 저장된다. (x)
  - 변수가 데이터를 가리킨다. (o)
    - ex) 리스트의 얕은, 깊은 복사 (`.copy()`)

2. 매개변수

- 위치 가변 매개변수 (`*args`, Positional Variable Length Parameter)
- 키워드 가변 매개변수 (`**kwargs`, Keyword Variable Length Parameter)
  - 개수가 정해지지 않은 매개변수
  - *args는 튜플형, **kwargs는 딕셔너리형

```
def print_fruits(*args):
    for arg in args:
        print(arg)

def print_dict(**kwargs):
    for key, value in kwargs.item():
        print(f"{key}:{value}")
```

3. 람다함수

- 이름을 지을 필요도 없는 간단한 형태의 함수
- 다른 함수의 인자(argument)로 넣을 수 있다.
- 코드가 간결해지고, 메모리가 절약된다.

```
lambda 매개변수: 결과

(lambda a: a-1)(10)
```

```
or

minus_one = lambda a: a-1
minus_one(10)
```

4. map, filter

```
# map
map(함수, 순서가 있는 자료형)
map(int, ['3', '4', '5'])
# map object를 반환, 따라서 list함수에 넣어서 list로 반환
list(map(int, ['3', '4', '5']))


# filter
filter(함수, 순서가 있는 자료형)
map(func, [-1, 0, 1, 2])
# func의 return이 true인 인자만 filter object로 반환
```

5. class 심화

- 클래스와 객체
- 여러가지 속성 (인스턴스 속성, 클래스 속성, 비공개 속성)
- 여러가지 메서드 (인스턴스 메서드, 클래스 메서드, 정적 메서드, 매직 메서드)
- 상속 (오버라이딩, 추상클래스)

1. 절차지향 vs 객체지향

- 절차지향:
  - 기능들을 어떤 순서로 처리할 것 인가에 초점을 맞춤
- 객체지향:
  - 객체가 중심이 되고, 객체를 정의하고, 객체간 상호작용에 초점을 맞춤

2. 클래스 vs 객체

- 클래스: 객체를 만들기 위한 설계도
  - 속성과 메서드를 가지고 있음
- 객체: 셀계도로부터 만들어낸 제품

3. 클래스 속성

- 인스턴스 속성 (instance attribute)
  - 객체마다 다르게 가지는 속성
  - 클래스 안: self.속성명
  - 클래스 밖: 객체명.속성명
- 클래스 속성 (class attribute)
  - 모든 객체가 공유하는 속성

    - 생성자 밖에 선언
    - 클래스이름.속성명
- 비공개 속성 (private attribute)
    - 클래스 안에서만 접근 가능한 속성
    - self.__속성명
    - self.속성명 -> 이건 public

4. 클래스 메서드

- 인스턴스 메서드 (instance method)
    - 인스턴스 속성에 접근 할 수 있는 메서드
    - 항상 첫번째 파라미터로 self를 갖는다
- 클래스 메서드 (class method)
    - 클래스 속성에 접근하기 위해서 사용함
    - 클래스를 의미하는 cls를 파라미터로 받는다

```
class Unit:
  count = 0
  ...
  @classmethod
  def print_count(cls):
    print(f"전체 유닛 개수: {cls.count}")
```

- 정적 메서드 (static method)
    - 인스턴스를 만들 필요가 없는 메서드
    - self를 받지 않는다
    - 메서드가 인스턴스 유무와 관계없이 독립적으로 사용 될 때

```
class Math:
  @staticmethod
  def add(x, y):
    return x+y
```

- 매직 메서드 (magic method)
    - 클래스 안에서 정의할 수 있는 스페셜 메서드
    - 특별한 상황에서 호출된다.
    - __이름__의 형태로 되어있다.
        - ex) __init__: 클래스에서 객체를 생성할 때 호출
        - ex) __str__: 객체를 출력할 때 호출
        - dir() 함수로 메서드를 확인 할 수 있음

5. 상속

- 클래스의 공통된 속성과 메서드를 뽑아내서 부모 클래스를 만든다.
- 이를 자식 클래스에서 상속받아서 사용한다.

- 코드의 중복을 제거할 수 있다.
- 유지보수가 편리해진다.
- 추상클래스 (abstract class)
    - 추상메서드 (abstract method)를 가짐
    - 상속받는 자식클래스에서 구현을 강제하도록 만드는 것

6. is-a, has-a 링크

- is-a:
    - is-a는 추상화(형식이나 클래스와 같은)들 사이의 포함 관계를 의미하며, 한 클래스 A가 다른 클래스 B의 서브클래스(파생클래스)임을 이야기합니다. 다른 말로, 타입 A는 타입 B의 명세(specification)를 암시한다는 점에서 타입 B의 서브타입이라고도 할 수 있습니다.
    - is-a 관계는 타입 또는 클래스 간의 has-a 관계와는 대조됩니다. has-a 및 is-a 관계들 간의 혼동은 실세계 관계 모델에 대한 설계에 있어 자주 발견되는 에러입니다. is-a 관계는 또한 객체 또는 타입 간의 instance-of 관계와도 대조됩니다.
- has-a:
    - has-a는 구성 관계를 의미하며 한 오브젝트(구성된 객체, 또는 부분/멤버 객체라고도 부릅니다)가 다른 오브젝트(composite type이라고 부릅니다)에 "속한다(belongs to)"를 말합니다. 단순히 말해, has-a 관계는 객체의 멤버 필드라고 불리는 객체를 말하며, Multiple has-a 관계는 소유 계층구조를 형성하기 위해 결합하는 경우를 말합니다.

6. 데이터베이스

- 파이썬에서 SQLite 사용방법
- DDL(CREATE, ALTER, DROP)
- DML(INSERT, SELECT, UPDATE, DELETE)

1. 데이터베이스란?

- 데이터베이스의 개념
    - 구조화된 데이터의 집합 (like 엑셀)
- 데이터베이스 구성요소
    - 데이터베이스 (database): 테이블(table)의 집합
    - 테이블 (table): 행(row)의 집합
    - 행 (row): 한 단위의 데이터 기록(record)
    - 열 (column): 데이터의 항목 (field)
- DBMS(DataBase Management System)
    - 데이터베이스를 관리해주는 시스템
        - 클라이언트가 SQL을 이용하여 서버에 명령을 내리면 서버가 클라이언트에 응답을 함
        - ex) MySQL, Oracle, SQLite (파이썬 내장)
- SQL이란 무엇인가?
    - Structured Query Language

    - 데이터베이스를 관리하기 위해 사용되는 언어
    - SQL의 종류
        1. DDL (Data Definition Language) 데이터 정의 언어 (CREATE, ALTER, DROP)
        2. DML (Data Manipulation Language) 데이터 조작 언어 (INSERT, SELECT, UPDATE, DELETE)

- DB Browser for SQLite 설치 공홈 홈브류를 통해 설치

```
$ brew install --cask db-browser-for-sqlite
```

2. DDL

- SQLite 데이터 타입

데이터타입	설명
integer	정수
real	실수
text	문자열
null	데이터 없음

- SQL CREATE
    - 테이블 생성 명령(쿼리)

```
CREATE TABLE 테이블명 (컬럼명1 데이터타입, 컬럼명2 데이터타입);
CREATE TABLE post (id integer primary key, title text not null default
'제목없음', content text default '내용없음');
CREATE TABLE user (id integer primary key autoincrement, nickname text
unique);
```

    - 제약조건

    1. primary key: 기본키, 레코드를 구분 짓는 값
    2. not null: 데이터가 비어있을 수 없다.
    3. default: 기본값
    4. unique: 중복 불가

- SQL DROP

- Sqlite3 browser 설치 및 사용법

7. 정규표현식

- 문자열에서 특정 패턴을 찾고 싶을 때 사용
- 문자열 추출, 유효성 검사에서 유용하게 쓰일 수 있다.
- 거의 모든 언어에서 지원한다. 따라서, 범용성이 높다.
- 하지만, 가독성이 좋지 못하고, 유지보수가 어렵다.

8. 동시성과 병렬성

1. 프로그램
    - 작업을 수행하는 명령어 집합
2. 프로세스
    - 실행중인 프로그램
3. 스레드
    - 프로세스에서 실행되는 작업
    - 프로세스는 기본적으로 하나의 스레드로 구성
    - 경우에 따라서 여러개의 스레드로 구성이 가능하다. (멀티스레딩)
4. 동시성 프로그래밍
    - 멀티스레딩, 하나의 프로세스에서 여러개의 스레드를 빠르게 스위칭하며 작업
    - 동시에 실행되는 것 처럼 보이지만 스레드 여러개를 번갈아 가며 실행
5. 병렬성 프로그래밍
    - 멀티코어, 프로세스가 여러개 존재
    - 실제로 작업이 동시에 실행됨, 프로세스를 여러개 만들어서 동시에 실행