

Web Studio 2019

6. Relationship

Contents

1. Query string
2. `primary_key`, `foreign_key`
3. Serialize, deserialize

Query string

URI와 Query string

1. URI에서 ? 를 기준으로 오른쪽을 query string이라 부름
2. &를 이용하여 구분함

주의 요약 | www.ssodam.com/content/649192?prev=1&prev_content=/today

sodam 커뮤니티 ▼ 커리어 ▼ 생활 ▼ 족보실 ▼ 전체글 인기글 내 페이지 로그아웃

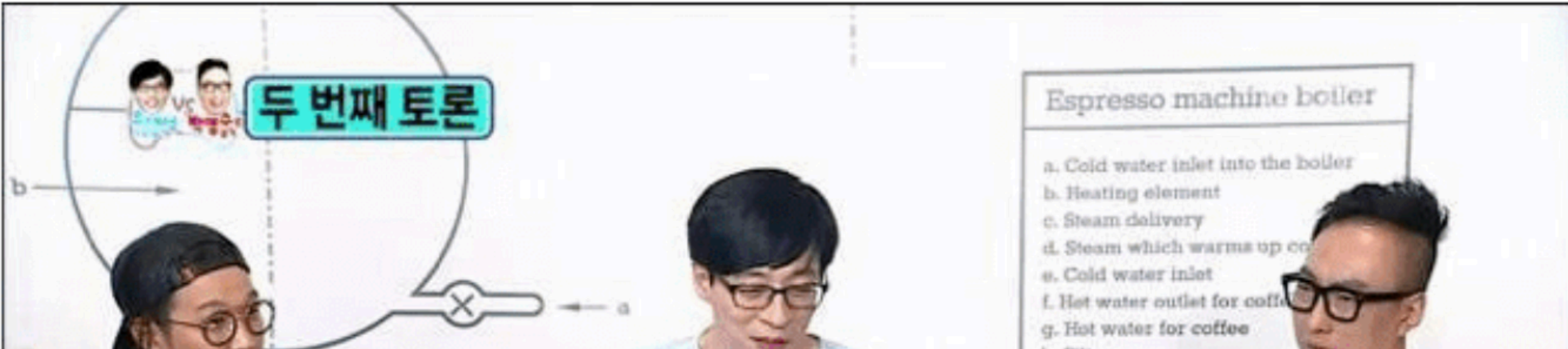
커뮤니티

자유게시판	+11
익게1	+18
익게2	+845
연애상담소	+49
학내 이슈	+3
졸업생	+3
헬스	+3
문의	+3
정치/핫이슈	+15
맛집	+4
다미챗	+2

팬들한테서 도망가도 욕 안 먹는 연예인

by 익명 | 익게2 | 조회 3010 | 2729

2019/04/06 20:52



Query string

쓰임새

1. http://0.0.0.0:5000/api/comments?article_id=3
: article_id가 3인 코멘트를 전부 가져와라!
2. http://0.0.0.0:5000/api/comments?article_id=3&user_id=1
: article_id가 3이며 user_id=1인 유저가 작성한 코멘트를 전부 가져와라!

Key, Relationship

Primary key, foreign key

1. Primary key는 row를 고유하게 식별하는 기본 키임
2. Foreign key는 다른 테이블의 기본 키를 가리키는 키임
3. Foreign key는 데이터의 참조 무결성을 확인하기 위해 사용됨
4. join연산의 최적화를 위해 사용됨

Key, Relationship

Primary key, foreign key

1. Primary key는 row를 고유하게 식별하는 기본 키임
2. Foreign key는 다른 테이블의 기본 키를 가리키는 키임
3. Foreign key는 데이터의 참조 무결성을 확인하기 위해 사용됨
4. join연산의 최적화를 위해 사용됨

Key, Relationship

Relationship

1. SQLAlchemy에선 relationship을 쉽게 사용할 수 있음
2. 양방향 바인딩이 됨

```
class Article(db.Model):
    __tablename__ = 'article'
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))
    title = db.Column(db.String(200))
    content = db.Column(db.Text)

    user = relationship('User', backref=backref('articles', order_by=id))

    def __init__(self, user_id, title, content):
        self.user_id = user_id
        self.title = title
        self.content = content

    def serialize(self):
        return json.dumps({
            'id': self.id,
            'user_id': self.user_id,
            'title': self.title,
            'content': self.content
        })
```

```
class Comment(db.Model):
    __tablename__ = 'comment'
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))
    article_id = db.Column(db.Integer, db.ForeignKey('article.id'))
    content = db.Column(db.Text)

    user = relationship('User', backref=backref('comments', order_by=id))
    article = relationship('Article', backref=backref('comments', order_by=id))

    def __init__(self, user_id, article_id, content):
        self.user_id = user_id
        self.article_id = article_id
        self.content = content

    def serialize(self):
        return json.dumps({
            'id': self.id,
            'user_id': self.user_id,
            'article_id': self.article_id,
            'content': self.content
        })
```


Key, Relationship

Relationship

1. 아래와 같이 접근 할 수 있음
2. nested하게 접근도 가능
3. Lazy evaluation

```
class ArticleList(Resource):
    def get_articles(self):
        articles = Article.query.all()
        return articles

    def get(self):
        articles = self.get_articles()
        for article in articles:
            print(article.comments)
        print(articles[0].comments[0].content)
        print(articles[0].comments[0].user.email)
        return serializer(articles)
```

```
127.0.0.1 - - [08/Apr/2019 18:29:57] "GET /api/articles?__debugger__=yes&
* Detected change in '/Users/sisobus/coding/WebStudio2019/6_relationship
* Restarting with stat
* Debugger is active!
* Debugger PIN: 189-506-823
[<Comment 1>, <Comment 4>]
[<Comment 2>]
[]
updated comment content1
sisobus2@vuno.co
127.0.0.1 - - [08/Apr/2019 18:30:06] "GET /api/articles HTTP/1.1" 200 -
```


Serialize, deserialize

Serialize

1. HTTP(S) 프로토콜로 통신할 때, 문자열 데이터를 주고 받음
2. 파이썬 객체를 문자열로 바꾸는 것 = serialize (직렬화)
3. 문자열을 파이썬 객체로 바꾸는 것 = deserialize (역직렬화^(?))

Serialize, deserialize

Serialize (json)

1. Serialize = `json.dumps`
2. Deserialize = `json.loads`

실습

블로그 api를 작성해보자

Q & A

Appendix

Bash 기본 명령어

1. ls : 현재 위치의 파일 리스트
 - ls -al : 숨겨진 파일까지 전부 보기
2. cd : 디렉토리 이동
 - .은 현재 디렉토리, ..은 바로 전 디렉토리를 의미함
 - 예를들어 cd .. 이 명령어는 바로 전 디렉토리로 이동
3. rm : 파일 지우기
 - rm -rf : 디렉토리도 지울 수 있음
4. vi <filename> : vim 에디터를 이용한 파일 작성하기
5. mkdir <directoryname> : 디렉토리(폴더) 만들기
6. pwd : 현재 위치 출력하기

Appendix

Vim

1. 최고의 Text editor
2. 이것만 잘써도 코딩 생산성이 상당히 많이 올라감
3. 대부분의 IDE에는 Vim Plugin이 존재함
4. Vim 쓰세요
5. 두번쓰세요
6. 세번쓰세요
7. 평생쓰세요

Appendix

Vim

1. 명령어 모드와 에디터 모드로 나뉨
2. 맨처음 들어갈 때에는 명령어 모드임
3. 에디터 모드에서 명령어 모드로 바꾸는 것은 esc로 함
4. 명령어 모드에서 에디터 모드로 바꾸는 것은 i, o, a로 함 (i만 알아도 상관 없습니다.)

Appendix

Vim 명령어 모드

1. i는 현재 커서에서 에디터모드로 전환
2. yy는 현재 줄 복사
3. p는 복사한 것을 붙여넣기
4. dd는 현재줄 삭제
5. u는 undo
6. ctrl+r는 redo
7. :wq 는 저장후 종료를 의미함
8. /<search string> 으로 검색할 수 있음
9. :%s/<stringA>/<stringB>/g 는 stringA를 stringB로 대체함
 - :%s/^/₩/₩//g : 맨 앞에 //를 붙임
 - :3,12s/ha/hi/g : 3번줄부터 12번째줄의 ha를 hi로 바꿈