

# 컴퓨터 네트워크

[실습 6]

컴퓨터공학과

201702042 우정균

## Restful API 시나리오 수행 결과

```
woo-jk >...  
tion/json'; echo  
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo  
  
echo -n 'PUT'; curl -X 'PUT' 'http://localhost:8080/paste/1' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "First PUT for 0"}'; echo  
echo -n 'PUT'; curl -X 'PUT' 'http://localhost:8080/paste/2' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second PUT for 1"}'; echo  
  
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo  
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo  
  
echo -n 'DELETE'; curl -X 'DELETE' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo  
  
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo  
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo  
  
/{ "message": "Hello World" }  
GET{"paste_id":1,"paste":null}  
GET{"paste_id":2,"paste":null}  
PUT{"paste_id":1,"paste":null}  
PUT{"paste_id":2,"paste":null}  
DELETE{"paste_id":1,"paste":null}  
DELETE{"paste_id":2,"paste":null}  
POST{"paste_id":1,"paste":{"content":"First post"}}  
POST{"paste_id":2,"paste":{"content":"Second post"}}  
GET{"paste_id":1,"paste":{"content":"First post"}}  
GET{"paste_id":2,"paste":{"content":"Second post"}}  
PUT{"paste_id":1,"paste":{"content":"First PUT for 0"}}  
PUT{"paste_id":2,"paste":{"content":"Second PUT for 1"}}  
GET{"paste_id":1,"paste":{"content":"First PUT for 0"}}  
GET{"paste_id":2,"paste":{"content":"Second PUT for 1"}}  
DELETE{"paste_id":1,"paste":{"content":"First PUT for 0"}}  
GET{"paste_id":1,"paste":null}  
GET{"paste_id":2,"paste":{"content":"Second PUT for 1"}}  
woo-jk
```

실습 자료의 시나리오 예제를 터미널에 그대로 복사 붙여넣기 한 결과이다. 자료 21페이지의 결과와 똑같다.

## 멱등성 증명

```
woo-jk ~
> echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo
echo -n 'PUT'; curl -X 'PUT' 'http://localhost:8080/paste/2' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second PUT for 1"}'; echo
echo -n 'PUT'; curl -X 'PUT' 'http://localhost:8080/paste/2' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second PUT for 1"}'; echo
echo -n 'PUT'; curl -X 'PUT' 'http://localhost:8080/paste/2' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second PUT for 1"}'; echo
echo -n 'DELETE'; curl -X 'DELETE' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo
echo -n 'DELETE'; curl -X 'DELETE' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo
echo -n 'DELETE'; curl -X 'DELETE' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo
GET{"paste_id":2,"paste":{"content":"Second PUT for 1"}}
GET{"paste_id":2,"paste":{"content":"Second PUT for 1"}}
GET{"paste_id":2,"paste":{"content":"Second PUT for 1"}}
PUT{"paste_id":2,"paste":{"content":"Second PUT for 1"}}
PUT{"paste_id":2,"paste":{"content":"Second PUT for 1"}}
PUT{"paste_id":2,"paste":{"content":"Second PUT for 1"}}
DELETE{"paste_id":1,"paste":null}
DELETE{"paste_id":1,"paste":null}
DELETE{"paste_id":1,"paste":null}
```

GET, PUT, DELETE 메소드의 경우 같은 요청을 여러 번 날려도 서버의 상태는 유지된다.

```
woo-jk ~
> echo -n 'POST'; curl -X 'POST' 'http://localhost:8080/paste/' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second post"}'; echo
echo -n 'POST'; curl -X 'POST' 'http://localhost:8080/paste/' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second post"}'; echo
echo -n 'POST'; curl -X 'POST' 'http://localhost:8080/paste/' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second post"}'; echo
echo -n 'POST'; curl -X 'POST' 'http://localhost:8080/paste/' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second post"}'; echo
POST{"paste_id":4,"paste":{"content":"Second post"}}
POST{"paste_id":5,"paste":{"content":"Second post"}}
POST{"paste_id":6,"paste":{"content":"Second post"}}
```

POST 메소드의 경우 같은 요청을 여러 번 날리면 서버에 데이터가 계속해서 추가된다. 사진의 경우 paste\_id가 계속 올라간다.

## SQLite3 활용 증명

\*SQL 테이블을 생성하는 코드와 GET 요청 메소드의 SQLite 코드는 실습자료에 포함되어 있으므로 생략한다.

### POST 메소드

```
@app.post('/paste/')
def post_paste(paste: Paste):
    res = cur.execute('''SELECT id, content
                        FROM Paste
                        ORDER BY id DESC''')
    data = res.fetchone()
    if data is not None:
        paste_id = data[0] + 1
    else:
        paste_id = 1
    cur.execute('''INSERT INTO Paste (id, content) VALUES (?, ?)''',
                (paste_id, paste.content,))
    conn.commit()
    return {'paste_id': paste_id,
            'paste': paste}
```

테이블의 행 중 id가 가장 큰 행을 가져와서 data 변수에 저장한다.

데이터가 있다면 행의 id + 1을 저장할 paste의 id로 저장하고 데이터가 없다면 paste의 id를 1로 설정한다.

\*AUTOINCREMENT 기능을 사용해서 id를 지정할 경우, DELETE했을 때 id가 줄어들지 않는 문제가 있기 때문에 AUTOINCREMENT 기능을 사용하지 않고 위와 같은 방식으로 id를 지정해준다.

excute()함수를 통해 INSERT 문을 실행해준다.

commit() 함수를 통해 db를 저장하고, 저장한 id와 paste를 반환한다.

## PUT 메소드

```
@app.put('/paste/{paste_id}')
def put_paste(paste_id: int, paste: Paste):
    res = cur.execute(''SELECT id, content
                        FROM Paste
                        WHERE id = ?'', (paste_id,))
    data = res.fetchone()
    if data is not None:
        cur.execute(''UPDATE Paste
                    SET content = ?
                    WHERE id = ?'', (paste.content, paste_id,))
        conn.commit()
        return {'paste_id': paste_id,
                'paste': paste}
    else:
        return {'paste_id': paste_id,
                'paste': None}
```

SELECT문을 통해 paste\_id에 해당하는 행을 가져온다.

데이터가 있다면 UPDATE문을 실행하고 COMMIT 해준 뒤 저장한 값들을 반환한다.

데이터가 없다면 요청한 id값과 paste에는 None을 반환해준다.

## DELETE 메소드

```
@app.delete('/paste/{paste_id}')
def delete_paste(paste_id:int):
    res = cur.execute(''SELECT id, content
                        FROM Paste
                        WHERE id = ?'', (paste_id,))
    data = res.fetchone()
    if data is not None:
        cur.execute(''DELETE FROM Paste
                    WHERE id = ?'', ( paste_id,))
        conn.commit()
        paste = Paste(content=data[1])
        return {'paste_id': paste_id,
                'paste': paste}
    else:
        return {'paste_id': paste_id,
                'paste': None}
```

SELECT문을 통해 paste\_id에 해당하는 행을 가져온다.

데이터가 있다면 DELETE문을 실행해서 행을 삭제하고 COMMIT 해준다. 요청한 id와 저장했던 data 변수의 paste를 함께 반환한다.

데이터가 없다면 요청한 id값과 paste에는 None을 반환해준다.

## 서버 종료 후 재실행 사진

```
uvicorn main:app --reload --host 0.0.0.0 --port 8080

INFO: 127.0.0.1:50835 - "PUT /paste/2 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50836 - "DELETE /paste/1 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50837 - "DELETE /paste/2 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50838 - "POST /paste/ HTTP/1.1" 200 OK
INFO: 127.0.0.1:50839 - "POST /paste/ HTTP/1.1" 200 OK
INFO: 127.0.0.1:50840 - "GET /paste/1 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50841 - "GET /paste/2 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50842 - "PUT /paste/1 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50843 - "PUT /paste/2 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50844 - "GET /paste/2 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50845 - "GET /paste/2 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50846 - "DELETE /paste/1 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50847 - "GET /paste/1 HTTP/1.1" 200 OK
INFO: 127.0.0.1:50848 - "GET /paste/2 HTTP/1.1" 200 OK
WARNING: StatReloader detected changes in 'main.py'. Reloading...
INFO: Shutting down
INFO: Waiting for application shutdown.
INFO: Application shutdown complete.
INFO: Finished server process [9415]
INFO: Started server process [10201]
INFO: Waiting for application startup.
INFO: Application startup complete.
^CINFO: Shutting down
INFO: Waiting for application shutdown.
INFO: Application shutdown complete.
INFO: Finished server process [10201]
INFO: Stopping reloader process [9413]
wo-jk ➤ ~/Project/CN6/fastapi
➤ uvicorn main:app --reload --host 0.0.0.0 --port 8080
INFO: Will watch for changes in these directories: ['/Users/woo-jungkyun/Project/CN6/fastapi']
INFO: Uvicorn running on http://0.0.0.0:8080 (Press CTRL+C to quit)
INFO: Started reloader process [10865] using StatReloader
INFO: Started server process [10867]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:59506 - "GET /paste/2 HTTP/1.1" 200 OK

/.....
tion/json'; echo
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo

echo -n 'PUT'; curl -X 'PUT' 'http://localhost:8080/paste/1' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "First PUT for 0"}'; echo
echo -n 'PUT'; curl -X 'PUT' 'http://localhost:8080/paste/2' -H 'accept: application/json' -H 'Content-Type: application/json' -d '{"content": "Second PUT for 1"}'; echo

echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo

echo -n 'DELETE'; curl -X 'DELETE' 'http://localhost:8080/paste/1' -H 'accept: application/json';
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/1' -H 'accept: application/json'; echo
echo -n 'GET'; curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'; echo

/{"message": "Hello World"}
GET{"paste_id":1,"paste":null}
GET{"paste_id":2,"paste":null}
PUT{"paste_id":1,"paste":null}
PUT{"paste_id":2,"paste":null}
DELETE{"paste_id":1,"paste":null}
DELETE{"paste_id":2,"paste":null}
POST{"paste_id":1,"paste":{"content": "First post"}}
POST{"paste_id":2,"paste":{"content": "Second post"}}
GET{"paste_id":1,"paste":{"content": "First post"}}
GET{"paste_id":2,"paste":{"content": "Second post"}}
PUT{"paste_id":1,"paste":{"content": "First PUT for 0"}}
PUT{"paste_id":2,"paste":{"content": "Second PUT for 1"}}
GET{"paste_id":1,"paste":{"content": "First PUT for 0"}}
GET{"paste_id":2,"paste":{"content": "Second PUT for 1"}}
DELETE{"paste_id":1,"paste":{"content": "First PUT for 0"}}
GET{"paste_id":1,"paste":null}
GET{"paste_id":2,"paste":{"content": "Second PUT for 1"}}
wo-jk ➤ curl -X 'GET' 'http://localhost:8080/paste/2' -H 'accept: application/json'
{"paste_id":2,"paste":{"content": "Second PUT for 1"}}
wo-jk ➤
```

좌측이 fastapi 서버를 돌리는 터미널이고 우측이 메소드 요청을 하는 터미널이다.

서버를 종료하고 재실행을 한 뒤 GET 메소드 요청을 했을 때, 데이터가 사라지지 않고 보존되어 있는 모습을 볼 수 있다.