02. Rest API

컴퓨터공학부 201911228 홍지우

List

- URL
- Rest API [= Restful API] + 예시
- Rest API Method
  - GET
  - PUT
  - POST
  - DELETE
  - PATCH
- Rest API[Http] Response
  - 200 OK
  - 201 Created
  - 204 No Content
  - 400 Bad Request
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 409 Conflict
  - 500 Internal Server Error

01. URL

: 웹에서 접근 가능한 자원의 주소를 일관적으로 표현하는 형식

[URL의 구성]

- 프로토콜 : 서버와 클라이언트 간에 자원을 접근하는 방법 (ex: 웹브라우저-https/http, 이메일 주소 지정-mailto, 파일-ftp)
- 호스트 : 웹 서버의 위치를 지정. IP 또는 도메인명 / 포트 : 웹서버에서 자원을 접근하기 위해 사용하는 관문. HTTP 포트는 80, HTTPS 는 443. 표준 포트를 사용한다면 포트번호는 보통 생략한다.
- 매개변수 : 서버에서 보내는 추가 파라미터. 키와 값으로 짝을 이룬다. 여러 개의 파라미터를 사용할 경우 “&”으로 구분한다.
- 부분 식별자: 자원 안에서 어떤 세부 부분을 가리키는 역할. (ex- 위키백과의 책갈피로 특정 항목으로 바로 이동)

* ASCII 문자만 허용. 이외의 문자는 url 인코딩을 활용. 공백문자는 “+” 으로 사용
https://www.urlencoder.org/

02. Rest API Representational State Transfer API

: HTTP 통신에서 어떤 자원에 대한 CRUD 요청을 Resource와 Method (Get, Post) 으로 표현하여 특정한 형태로 전달하는 방식

[Rest API의 구성요소]

- Resource(자원) : 서버의 유니크한 ID를 가지는 Resource 으로, URI(Uniform Resource Identifier)라고도 함.
- Method(행위) : 서버에 요청을 보내기 위한 방식
- Representation of Resource(표현) : 클라이언트와 서버가 데이터를 주고 받는 형태. json, xml, text, rss 등이 있음.

02. Rest API Method

: 서버에 요청을 보낼 때 사용

[GET] - 보기

- 서버에게 리소스를 보내 달라고 요청
- 서버 혹은 DB의 리소스는 클라이언트로 전달만 될 뿐 변경되지 않는다.
- 웹브라우저 주소창에 주소를 입력하면 이 신호는 항상 get으로 요청된다.
- URL에 데이터를 포함시켜 요청하여 보안에 취약

```
HTTP GET  http://localhost:8080/boardList?name=good&page=5
```

[POST] - 생성

- 서버에게 리소스를 보내면서 생성해달라고 요청. 또는 특정한 명령을 실행할 때 사용
- 데이터를 body에 포함시켜 요청
- ex) 회원가입 버튼을 눌렀을 경우 -〉 이 정보들을 DB에 등록해줘

```
HTTP POST /dogs { "name":"goodboy", "age": 5}
```

02. Rest API Method

: 서버에 요청을 보낼 때 사용

[PUT] - 전체 수정

- 서버에게 리소스를 업데이트하거나, 리소스가 없다면 새롭게 생성해달라고 요청
- ex) 게시물을 수정할 때
- PATCH와 비교해서 전체 데이터를 교체하는 차이점이 있다.

```
HTTP PUT /dogs/3 { “name”:”goodboy”, “age”: 5}
```

[PATCH] – 부분 수정

- 서버에게 리소스의 업데이트를 요청
- PUT과 비교해서 부분 데이터를 업데이트하는 차이점이 있다.

```
{id: 1, username: 'admin', email:'email@example.org'}

PATCH /users/123

[
        {
                "op": "replace",
                "path": "/email",
                "value": "new.email@example.org"
        }
]
```

02. Rest API Method

: 서버에 요청을 보낼 때 사용

[DELETE] - 삭제

- 서버에게 리소스의 삭제를 요청
- 이를 수행할 경우, cache의 상태가 변하므로 cache를 삭제해야 한다.

```
HTTP DELETE /dogs/1
```

03. Rest API[Http] Response

: 서버에 요청에 대한 응답을 할 때 사용

[성공]

- 200 : OK (요청이 성공적으로 수행되었음)
- 201 : Created (PUT 메소드에 의해 서버에 파일이 생성됨)
- 204 : No Content (PUT, POST, DELETE 요청의 경우 성공은 했지만, 전송할 데이터가 없는 경우)

[클라이언트 요청 에러]

- 400 : Bad Request (사용자의 잘못된 요청으로 처리할 수 없음)
- 401 : Unauthorized (인증이 필요한 페이지를 요청한 경우)
- 403 : Forbidden (접근 금지. 관리자 페이지 접근 차단)
- 404 : Not Found (요청한 페이지가 없음)
- 409 : Conflict

[서버 에러]

- 500 : Internal Server Error (내부 서버 오류)

03. Rest API[Http] Response

: 예시

HTTP GET http://localhost:8080/users

- 리소스가 발견되었을 경우 → 응답 내용 + 200 (OK)
- 리소스가 발견되지 못했을 경우 → 404 (Not Found)
- 요청 형식이 적합하지 않은 경우 → 400 (Bad Request)

HTTP POST http://localhost:8080/users

- 서버에 리소스가 정상적으로 생성된 경우 → 201 (Create)
- 특정 명령 실행이 성공한 경우 → 200 (OK)
- 특정 명령 실행이 실패한 경우 → 204 (No Content)

HTTP PUT http://localhost:8080/users/1

- 정상적으로 서버에 업데이트 된 경우 → 200 (OK)
- 실패한 경우 → 204 (No content)

HTTP DELETE http://localhost:8080/users/1

- 성공적으로 리소스를 삭제했을 경우 → 200 (OK)
- 그렇지 않은 경우 → 204 (No Content)
- 해당 리소스가 없을 경우 → 404 (Not Found)

참고 사이트

https://mangkyu.tistory.com/46
https://sabarada.tistory.com/29