개요

2022년 10월 8일 토요일 오후 6:08

- RESTfull 웹 서비스는 REST 규정에 맞게 만든 ROA를 따르는 웹 서비스 디자인 표준
- 자원을 다루는 방법과 특정 웹 페이지로 접근하는 방법을 비슷한 형태로 구성 의미
- RESTfull 웹서비스의 구조

	컬렉션(배열) /collection	요소 /conllection/id
GET	컬렉션을 조회	컬렉션의 특정 요소 조회
POST	컬렉션에 새로운 데이터 추가	사용하지 않음
PUT	컬렉션 전체를 한번에 변경	컬렉션의 특정 요소를 수정
DELETE	컬렉션 전체를 삭제	컬렉션의 특정 요소를 삭제

- 강제하지 않기 때문에 서비스를 제공하는 기업의 정책에 따라 다를 수 있다
- 예 : 사용자 관련 정보 형태
  - GET / user: 사용자 전체를 조회
  - GET / user/ 273: 273번 사용자를 조회한다.
  - POST/ user : 사용자 추가
  - DELETE /user/273 : 273 사용자 삭제
- RESTfull 웹 서비스 예

메소드	경로	설명
GET	/user	모든 사용자 정보 조회
POST	/user	사용자 추가
GET	/user/:id	특정 사용자 정보 조회
PUT	/user/:id	특정 사용자 정보 수정
DELETE	/user/:id	특정 사용자 정보 삭제

- GET 이외의 요청은 모두 X-WWW-url-encoded 방식으로 데이터 전달한다
- 따라서 body-parser를 사용해서 클라이언트 데이터 분석해야한다

코드 구성

2022년 10월 12일 수요일 오후 9:37

1. 사용자 데이터

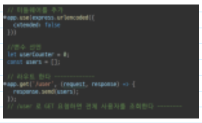
- 데이터를 어떻게 저장할 것인지부터 생각해야한다
- 현재 웹 서비스는 중심이 되는 데이터는 사용자
  {
  Id: 사용자_id,
  Name: 사용자_이름,
  Region: 거주_지역
  }

- 위의 데이터를 어떻게 저장할 것인지 생각
- 일반적인 웹서비스는 데이터베이스를 사용해서 데이터를 저장한다(자바코드)
- 자바스크립트 배열을 사용해 데이터 저장하기
  - Const users = []
- 데이터가 저장되면
  - Const users = [{
    Id: 0,
    Name : '홍길동',
    Region: '서울 강서구'
    }, {
    Id: 1,
    Name : '유관순',
    Region: '인천 연수구'
    }]

- 자바 스크립트 배열이므로
  1) 데이터 조회
     Users // 모든 데이터 조회
     Users[0] //0번 인덱스에 있는 데이터 조회
  2) 데이터 추가
     Users.push({
     Id: 10,
     Name : '이름',
     Region: '지역'
     })
  3) 데이터 수정
     Users[0].name = '새로운 이름'
     Users[0].region= '새로운 지역'
  4) 데이터 제거 순서 가능
     users.splice(0,1) // 0qjsWo epdlxj 제거

2. GET/ user

- Get 요청으로 /user에 요청을 걸면 사용자 전체 리스트를 응답해줘야한다



3. GET/ user /:id

- 변수 선언에서 id 값을 추출
- 필터함수로 id가 일치하는 사용자 찾는다
- 해당 id를 가진 사용자가 있다면 요소 하나를 가지는 배열이 리턴 된다.
- Id를 가진 사용자가 없다면 빈 배열 리턴

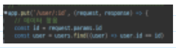
4. Post/user

- Post 요청으로 /user에 요청을 걸면, user 배열에 데이터를 추가해야 한다 .
  1) 사용자 요청 유효성 확인
     - 사용자 요청에 name 속성과 region속성이 포함되었는지 확인
     - 하나라도 포함되어 있지 않으면 오류와 함께 오류원인 전달
     - Name 속성을 전달하지 않으면, body.name이 undefined로 나옴
     - 따라서 !body.undefined가 true로 평가되며 조건문 안으로 들어간다
  2) 데이터를 users 배열에 추가
     - Body에서 name 속성과 region속성을 꺼내고 이를 활용해 데이터 조합
     - 최종적으로 users.push() 메소드로 배열에 요소 추가
     - userCounter 라는 정수 변수를 활용해 자동으로 지정

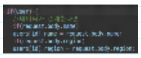
  3) 사용자에게 데이터를 추가가 성공했다고 응답
     - 최종적으로 send()메소드로 응답
     - 데이터가 어떻게 추가 되었는지 요청자전달을 위해 data 리턴

5. PUT/user

- Users 배열의 특정 id를 가진 사용자의 정보를 수정
  1) 사용자 요청을 기반으로 사용자 찾는다.



  2) 데이터를 수정

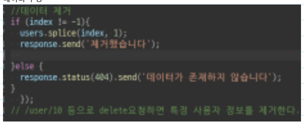


     - 사용자가 있을 경우에만 user를 수정
     - 이때 사용자가 name 속성과 region 속성 전달
  3) 응답

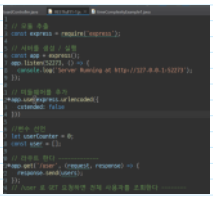
6. DEL/user/:id

  1) 데이터 찾기

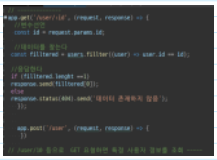
     - Find 메소드 활용해서 특정 id가진 사용자 찾을 수 있음
  2) 데이터 수정



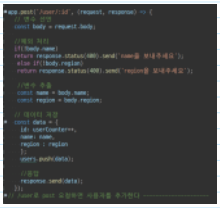
     - findIndex() 메소드는 요소를 찾으면 해당 인덱스를 리턴 한다
     - 만약 요소를 찾지 못하면 -1을 리턴
     - 요소 제거는 splice()메소드를 사용



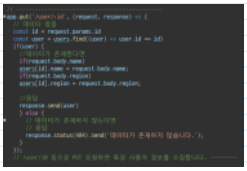
화면 캡처: 2022-10-13 오후 12:01



화면 캡처: 2022-10-13 오후 12:01



화면 캡처: 2022-10-13 오후 12:02



화면 캡처: 2022-10-13 오후 12:02

화면 캡처: 2022-10-13 오후 12:02