

JavaScript

◆ JSON 데이터 통신

정수아

Contents

01 JSON이란?

02 Fetch API

03 Fetch API - GET

04 Fetch API - POST



01

JSON이란?

JSON

❖ JSON(JavaScript Object Notation)

- 클라이언트와 서버 간의 HTTP 통신을 위한 텍스트 데이터 포맷
- 대부분의 프로그래밍 언어에서 사용 가능
- 표기 방식
 - 키와 값으로 구성

```
{  
  "name" : "Soo",  
  "age" : 20,  
  "alive" : true,  
  "hobby" : ["traveling", "piano"]  
}
```

JSON

❖ **JSON.stringify()**

- 객체 또는 배열을 JSON 포맷의 문자열로 변환
- 직렬화(serializing)
 - 클라이언트가 서버로 객체를 전송하려면 객체를 문자열화해야 하는데, 이를 직렬화라고 함

JSON

❖ JSON.stringify()

- 객체 → JSON

```
const obj = {  
  name: "Soo",  
  age: 20,  
  alive: true,  
  hobby: ["traveling", "piano"]  
};  
  
const json = JSON.stringify(obj);  
console.log(json);
```

- 실행 결과

```
{"name":"Soo","age":20,"alive":true,"hobby":["traveling","piano"]}
```

JSON

❖ JSON.stringify()

- 배열 → JSON

```
const person = [  
  { id: 1, name: "Soo", age: 20 },  
  { id: 2, name: "Kim", age: 30 },  
  { id: 3, name: "Lee", age: 40 }  
];  
  
const json = JSON.stringify(person);  
console.log(json);
```

- 실행 결과

```
[{"id":1,"name":"Soo","age":20},{"id":2,"name":"Kim","age":30},  
{"id":3,"name":"Lee","age":40}]
```

JSON

❖ **JSON.parse()**

- JSON 포맷의 문자열을 객체로 변환
- 역직렬화(deserializing)
 - 서버로부터 클라이언트에게 전송된 JSON 데이터는 문자열 타입
 - 이 문자열을 객체로 사용하려면 JSON 포맷의 문자열을 객체화 해야 하는데, 이를 역직렬화라고 함

JSON

❖ JSON.parse()

- JSON 포맷 문자열 → 객체

```
const obj = {  
  name: "Soo",  
  age: 20,  
  alive: true,  
  hobby: ["traveling", "piano"]  
};  
  
// 직렬화  
const json = JSON.stringify(obj);  
  
// 역직렬화  
const parsed = JSON.parse(json);  
console.log(parsed);
```

- 실행 결과

```
{ name: 'Soo', age: 20, alive: true, hobby: ['traveling', 'piano'] }
```

JSON

❖ JSON.parse()

- JSON 포맷 문자열 → 배열

```
const person = [  
  { id: 1, name: "Soo", age: 20 },  
  { id: 2, name: "Kim", age: 30 },  
  { id: 3, name: "Lee", age: 40 }  
];  
  
// 직렬화  
const json = JSON.stringify(person);  
  
// 역직렬화  
const parsed = JSON.parse(json);  
console.log(parsed);
```

- 실행 결과

```
[ { id: 1, name: 'Soo', age: 20 },  
  { id: 2, name: 'Kim', age: 30 },  
  { id: 3, name: 'Lee', age: 40 } ]
```



02

Fetch API

Fetch API

❖ Fetch API

- 요청과 응답 등의 요소를 자바스크립트에서 접근하고 조작할 수 있는 인터페이스를 제공
- `fetch()` 메서드로 네트워크의 리소스를 쉽게 비동기적으로 가져올 수 있음

❖ `fetch()` 메서드

```
fetch(resource, options)
```

- 예시

```
fetch(url, options)
  .then(response => console.log(response))
  .catch(error => console.log(error));
```

[실습] fetch() 메서드로 데이터 읽어오기

❖ fetch.html

- <body> 태그 안에 fetch.js 파일 추가

```
<body>  
  <script src="fetch.js"></script>  
</body>
```

[실습] fetch() 메서드로 데이터 읽어오기

❖ fetch.js

```
fetch(https://jsonplaceholder.typicode.com/todos/1)
  .then(response => console.log(response))
  .catch(error => console.log(error));
```

- response 객체



```
▼ Response {type: 'cors', url: 'https://jsonplaceholder.typicode.com/todos/1',
  e, ...} ⓘ
  body: (...)
  bodyUsed: false
  headers: Headers {}
  ok: true
  redirected: false
  status: 200
  statusText: ""
  type: "cors"
  url: "https://jsonplaceholder.typicode.com/todos/1"
  ► [[Prototype]]: Response
```

[실습] fetch() 메서드로 데이터 읽어오기

❖ response 객체

- HTTP 응답 상태(status), HTTP 응답 헤더(headers), HTTP 응답 본문(body) 등을 포함
- JSON 문자열 형태로 변환을 해줘야 처리 가능
- response → JSON 형태로 파싱

```
fetch(https://jsonplaceholder.typicode.com/todos/1)
  .then(response => response.json())
  .catch(error => console.log(error));
```

[실습] fetch() 메서드로 데이터 읽어오기

❖ response 객체 메서드

메서드	설명
response.text()	응답을 텍스트 형태로 반환
response.json()	응답을 JSON 형태로 반환
response.formData()	응답을 FormData 객체 형태로 반환

[실습] fetch() 메서드로 데이터 읽어오기

❖ 데이터 처리

```
fetch(https://jsonplaceholder.typicode.com/todos/1)
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.log(error));
```

• 실행 결과

```
{ userId: 1,
  id: 1,
  title: 'delectus aut autem',
  completed: false
}
```



03

Fetch API - GET

json-server

❖ json-server

- json 파일을 사용하여 간단한 테스트를 위한 REST API server를 구축할 수 있는 패키지
- <https://www.npmjs.com/package/json-server>

❖ 설치

- node.js가 설치되어 있어야 함

```
$ npm install -g json-server
```

❖ 실행

```
$ json-server --watch 파일명 --port 포트번호
```

Fetch API - GET

❖ **fetch() 메서드를 이용한 GET 방식 호출**

- fetch() 메서드는 디폴트로 GET 방식으로 작동
- 메서드 호출 시, 별도의 옵션을 추가하지 않아도 됨

로컬 JSON 파일 읽어오기

❖ bookList.json

```
{
  "books": [
    {
      "id": 1,
      "title": "HTML+CSS+자바스크립트",
      "publisher": "이지스퍼블리싱",
      "price": "30,000"
    },
    {
      "id": 2,
      "title": "리액트 프로그래밍 정석",
      "publisher": "이지스퍼블리싱",
      "price": "36,000"
    },
    // 생략
  ]
}
```

로컬 JSON 파일 읽어오기

❖ 데이터 읽기 및 처리

- bookList.json 파일의 모든 데이터 읽기

```
fetch("http://localhost:4000/books")  
  .then((response) => response.json())  
  .then((data) => console.log(data))  
  .catch((error) => console.log(error));
```

- 실행 결과

```
▼ (4) [{...}, {...}, {...}, {...}] ⓘ  
  ▶ 0: {id: 1, title: 'HTML+CSS+자바스크립트', publisher: '이지스퍼블리싱', price: '30,000'}  
  ▶ 1: {id: 2, title: '리액트 프로그래밍 정석', publisher: '이지스퍼블리싱', price: '36,000'}  
  ▶ 2: {id: 3, title: '트렌드 코리아 2023', publisher: '미래의창', price: '17,100'}  
  ▶ 3: {id: 4, title: '아버지의 해방일지', publisher: '창비', price: '13,500'}  
    length: 4  
  ▶ [[Prototype]]: Array(0)
```

로컬 JSON 파일 읽어오기

❖ 데이터 읽기 및 처리

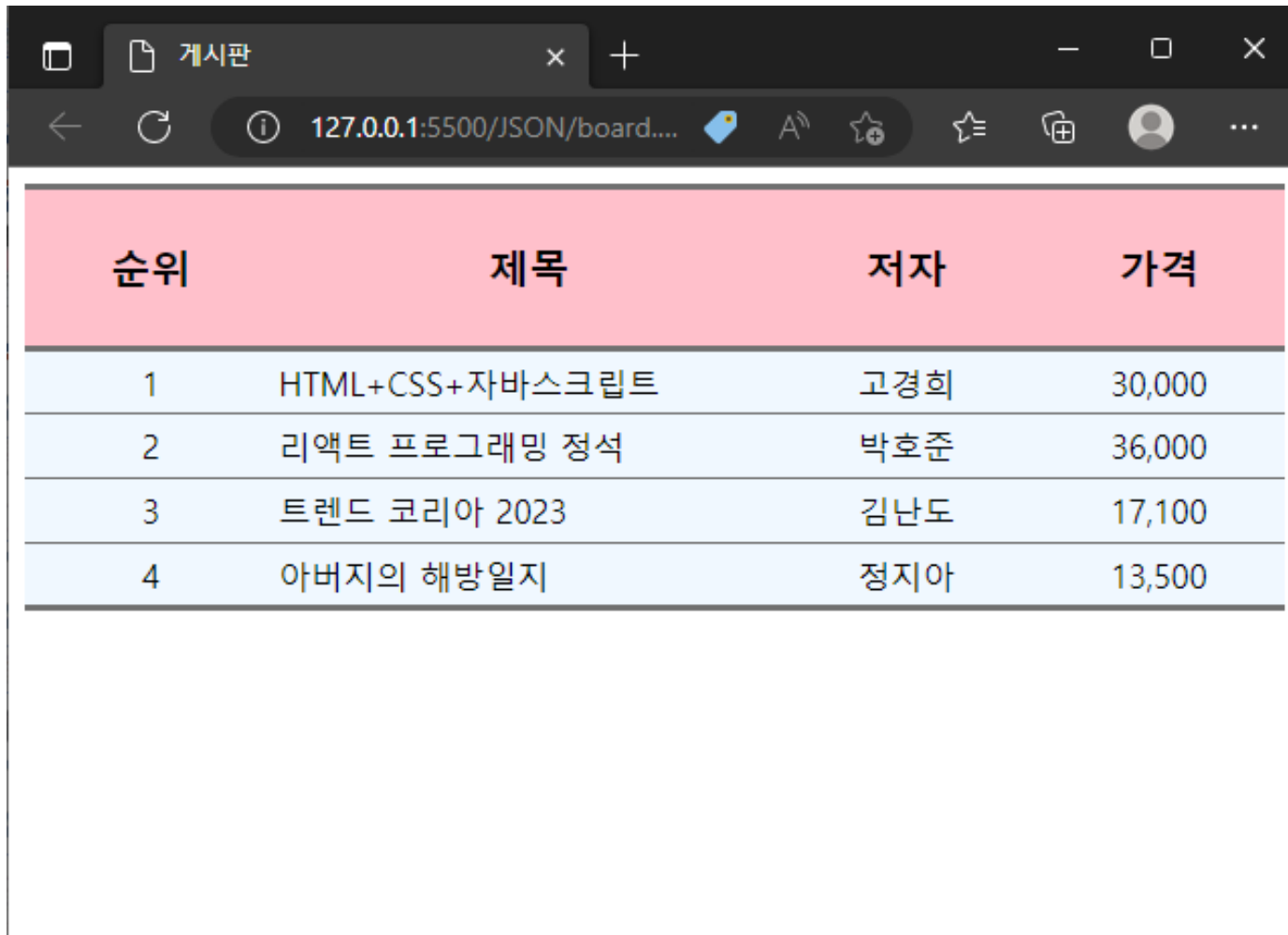
- bookList.json 파일의 일부 데이터만 읽기

```
fetch("http://localhost:4000/books")  
  .then((response) => response.json())  
  .then((data) => console.log(data[0]))  
  .catch((error) => console.log(error));
```

- 실행 결과

```
{ id: 1,  
  title: 'HTML+CSS+자바스크립트',  
  publisher: '이지스퍼블리싱',  
  price: '30,000'  
}
```

[실습] bookList 게시판 만들기



A screenshot of a web browser window displaying a book list. The browser's address bar shows the URL `127.0.0.1:5500/JSON/board....`. The page title is "게시판". The table has four columns: "순위" (Rank), "제목" (Title), "저자" (Author), and "가격" (Price). The table contains four rows of book data.

순위	제목	저자	가격
1	HTML+CSS+자바스크립트	고경희	30,000
2	리액트 프로그래밍 정석	박호준	36,000
3	트렌드 코리아 2023	김난도	17,100
4	아버지의 해방일지	정지아	13,500



04

Fetch API - POST

Fetch API - POST

❖ fetch() 메서드를 이용한 POST 방식 호출

- fetch() 메서드의 두 번째 매개변수 options 이용
- options
 - <https://developer.mozilla.org/en-US/docs/Web/API/fetch>

속성	설명
method	데이터 전송 방식
headers	HTTP 요청 헤더
body	HTTP 요청 본문

[실습] 도서 추가 기능 만들기

게시판

127.0.0.1:5500/JSON/board....

순위	제목	저자	가격
1	HTML+CSS+자바스크립트	고경희	30,000
2	리액트 프로그래밍 정석	박호준	36,000
3	트렌드 코리아 2023	김난도	17,100
4	아버지의 해방일지	정지아	13,500

도서 추가

제목 :

저자 :

가격 :

[실습] 도서 추가 기능 만들기

❖ board.js

```
// form 태그 찾기
const form = document.getElementById("save");

// submit 이벤트 처리
form.addEventListener("submit", (e) => {
  // 디폴트 행동 제거
  e.preventDefault();

  // FormData 객체 생성
  const formData = new FormData(form);

  // URLSearchParams 객체 생성
  const bookInfo = new URLSearchParams(formData);
});
```

[실습] 도서 추가 기능 만들기

❖ board.js

```
form.addEventListener("submit", (e) => {  
  
  // 생략  
  
  // 서버에 데이터 전송  
  fetch("http://localhost:4000/books", {  
    method: "POST",  
    body: bookInfo,  
  })  
    .then((response) => response.json())  
    .then((data) => console.log(data))  
    .catch((error) => console.log(error));  
});
```

THANK 😊 YOU