2. 구글 스프레드시트를 DB로 활용하는 방법

날짜	@2023년 3월 15일
생성일	@2023년 3월 14일 오후 1:37
태그	@2023년 3월 15일 오전 9:09

SUMMARY

주제	구글 스프레드시트를 DB로 활용하는 방법
경로	\\Laypop-server-2\laypop\프로젝트\Homework\2023\박가은\02. index
관련프로젝트경로	없음
내용	구글 스프레드시트 데이터 가져오기, fetch API
어려웠던 점	데이터 가공
개선할 점	depth가 3개 이상인 경우 처리, 기타 다른 기능

(+ SpeechSynthesisUtterance 사용해보기)

https://docs.google.com/spreadsheets/d/1BWbGqSLJOY0ltwVAb1i9jFfxbALbXZWVa6LYIUkxUDE/edit#gid=0

내용

- **개발 환경**
  - Bootstrap 5.2.3 ↪️바로가기
  - jQuery 3.6.3
  - module, fetch, google spread sheet
  - 네트워크에서 실행
- **구글 스프레드시트 데이터 가져오기**
  - **사용된 구글 스프레드시트 설정**

    구글 스프레드 시트 생성 → 공유 → 링크가 있는 모든 사용자로 설정 → 완료

    스프레드시트의 스프레드시트 id

    `https://docs.google.com/spreadsheets/d/` **`spreadsheetId`** `/edit#gid=0`
  - **Google Visualization API 응용한 코드**

    Visualization API - 구글 시트의 데이터를 기반으로 구글 차트를 그려주는 API

    (차트가 데이터 기반이기 때문에 데이터를 얻을 수 있다.)

    스프레드시트를 데이터베이스로 변환하지 않고 Google 스프레드시트를 쿼리하고, 데이터 집합을 반환하는 빠르고 쉬운 방법을 제공한다.

    HTML 테이블 형식으로 쿼리 결과를 표시할 수 있다.

    사용자 지정 URL을 구성하기만 하면 됨

- **동기Synchronous, 비동기 Asynchronous**

  동기 - 응답을 받은 후 다음 동작 수행 가능

비동기 - 응답 상태 상관없이 다음 동작 수행 가능

`fetch` 활용, 순서 보장 위해 `callback` 함수 사용

이외 활용 가능 - `async / await` (`Promise`에 대한 이해 필요)

- **쿼리문**
  - `&tq=`

    데이터를 필터링하기 위한 SQL 쿼리문 입력

    (구글API의 쿼리 언어는 SQL과 거의 같다고함)
  - `Select *`

    테이블의 모든 내용을 선택한다.

- `encodeURIComponent()`

  URI로 데이터를 전달하기 위해 문자열을 인코딩

- etc
  - SpeechSynthesisUtterance 사용해보기

코드

- 규칙

  depth에 따른 sheetName 결정, depth는 _(언더바)으로 구분함

  sheetName에 따른 스프레드시트의 데이터가 있어야 함

  각 시트의 칼럼값은 같아야함

  tab의 정보는 현재 2depth까지 구성할 수 있음

- **info.js**

  탭 정보를 담은 `dataTab` 변수를 갖고 있는 파일

  각 main, sub는 txt, sheetName 속성 값을 가지고 있음

  (txt: 화면에 표시될 텍스트, sheetName: 구글 스프레드시트에서 시트 이름을 찾을 때 사용하는 값)

```
const dataTab = [
  {
      main: {txt: '메타카드', sheetName: 'meta'},
      sub: [
          { txt: '수학', sheetName: 'math'},
          ...
      ]
  }
]

// 탭 선택 예시) 메타카드 - 수학
// sheetName => meta_math
```

- **index.js**

`userSelect` 변수를 갖고 있음 (sheetname을 담은 변수)

초기 세팅을 진행함

```
// userSelect는 사용자가 탭을 선택할 때마다 변경
// 초기값: meta_math
let userSelect = `${dataTab[0].main.sheetName}_${dataTab[0].sub[0].sheetName}`;
getSheetData(userSelect, createTable);
```

- **googleSpreadSheet.js**

```
const sheetId = 'sheetId'; // 고정값

const getSheetData = (_sheetName, callback)=>{
    const base = `https://docs.google.com/spreadsheets/d/${sheetId}/gviz/tq?`;
    const sheetName = _sheetName;
    const query = encodeURIComponent('Select *');
    const url = `${base}&sheet=${sheetName}&tq=${query}`;

    fetch(url)
        .then(response => response.text())
        .then(str => JSON.parse(str.substring(47).slice(0, -2)))
        .then(data => callback(data))
}
```

- **table.js**

```
let col = true;

const createTable = (_data)=>{
    if(_data.status !== "ok"){ alert('구글 스프레드시트와 연결되지 않았습니다.') }
    const data = _data.table;

    if(col){
        createCols(data.cols);
        col = false;
    }

    createRows(data.rows);
}
```

참고

https://dabid.tistory.com/41

https://asbnotebook.com/fetch-google-spread-sheet-data-using-javascript/

https://developers.google.com/sheets/api/guides/concepts?hl=ko

구글 워크스페이스

https://coderwall.com/p/pluhsg/google-spreadsheet-json-api-sql-filtering

Google Visualization API 관련

Google Visualization API 쿼리 관련

구글 스프레드시트를 데이터베이스로 이용하기