CRUD

#01. CRUD 개요.

> 출처: 위키백과 (https://ko.wikipedia.org/wiki/CRUD)

1) 정의

대부분의 컴퓨터 소프트웨어가 가지는 기본적인 데이터 처리 기능인 Create(생성), Read(읽기), Update(갱신), Delete(삭제)를 묶어서 일컫는 말.

사용자 인터페이스가 갖추어야 할 기능(정보의 참조/검색/갱신)을 가리키는 용어로서도 사용됨.

> CRUD 용어를 최초로 사용한 논문으로는 Kilov, H(1990)의 논문이며, 그 개념은 Kilov(1998)에도 자세히 서술되어 있다.

2) 프로그램 동작 구분

이름	조작	Method	SQL
Create	생성	POST	INSERT
Read	읽기	GET	SELECT
Update	갱신	PUT	UPDATE
Delete	삭제	DELETE	DELETE

유저 인터페이스 CRUD는, 여러 응용 프로그램의 사용자 인터페이스에도 들어맞는다. 예를 들어, 주소록나 전화번호부 소프트웨어를 생각해볼 수 있다. 여기서 기본적인 기록 단위는 각 개인의 연락처이다. 다음과 같은 기능들은 가장 간단한 것이면서도 필수적이다.

- 새로운 연락처 정보를 추가할 수 있다.
- 기존의 연락처 정보를 검색할 수 있다.
- 기존의 연락처 정보를 편집할 수 있다.
- 기존의 연락처 정보를 삭제할 수 있다.

이러한 4개의 조작을 모두 할 수 없다면 그 소프트웨어는 완전하다고 할 수 없다. 이들 기능은 매우 기본적이기 때문에, 한 묶음으로 설명되는 경우가 많다.

#02. Restful API

1) 개요

CRUD에 입각하여 어떠한 주제 하나를 놓고 입력, 수정, 삭제, 조회가 가능하도록 구성되어진 백엔드 시스템.

Java(Spring), Node.js(Express), PHP, Python 등을 활용하여 구현되고 실질적인 데이터는 DATABASE에 저장된다.

Ajax를 통해 백엔드와 통신한다는 것은 데이터베이스에 저장되어진 내용에 대한 입력, 수정, 삭제, 조회를 백엔드에 요청하는 것을 의미하는데 이를 위해 구축된 시스템이 Restful API이다.

2) GET 방식

QueryString

URL 뒤에 ?를 기준으로 하여 변수명=값&변수명=&값... 형식으로 요청변수를 나열하는 방법.

변수가 URL에 그대로 노출되기 때문에 보안이 필요한 경우 사용해서는 안된다.

전송 가능한 데이터의 용량에 한계가 있다. URL의 총 길이는 1024byte를 넘을 수 없기 때문에 포함 가능한 변수의 내용도 그 안에서 결정되어야 한다.

```
http://localhost/myschool/department/view.html?deptno=101&category=el
```

변수를 URL에 포함하므로 웹 브라우저로 조작이 가능하다.

Path

QueryString 방식에서 변수 명은 미리 정해진 순서로 약속하고 변수의 값만 전달하는 방식.

URL구성이 마치 폴더간의 연결처럼 보이기 때문에 QueryString방식보다는 상대적으로 보안에 유리하지만 QueryString의 모든 한계는 그대로 다 갖고 있다.

```
http://localhost/myschool/department/view/101/el
```

변수를 URL에 포함하므로 웹 브라우저로 조작이 가능하다.

3) POST, PUT, DELETE 방식

요청 변수는 URL이 아닌 HTTP Header에 포함하여 전송하는 방식.

서버(백엔드)의 설정에 따라 전송 가능 용량을 설정할 수 있다.(일반적으로 20M 내외)

파일 업로드가 가능하다.

개발자 도구를 사용할 수 있다면 요청정보를 열람할 수 있지만 일반 사용자에게는 요청 변수가 노출되지 않는다는 점에서 상대적으로 보안에 유리하다고 볼 수 있다.

웹 브라우저를 통한 조작이 불가능하다.

2) JSON Server

json 파일을 데이터베이스로 삼아 Restful API를 간단하게 가동할 수 있게 해 주는 오픈소스 프로그램.

3) Insomnia

Restful API와 연동하는 Ajax 기능 구현 전, 백엔드 시스템이 어떻게 가동하는지 테스트할 수 있는 프로그램.

https://insomnia.rest/

#03. 예제 폴더 구조

아래 구조는 일반적인 퍼블리싱 과정에서의 폴더 구조이다.

assets와 inc외에 필요하다면 페이지를 그룹단위로 관리하기 위한 폴더를 임의로 추가할 수 있다.

특정 폴더 안의 `index.html`은 그 폴더의 대문 페이지로서 URL상에 파일 이름이 지정되지 않고 폴더까지의 경로만 지정되면 `index.html`이 우선적으로 표시된다.

```
.
├── assets                : 리소스를 모아 놓는 폴더
│   ├── css               : CSS 파일 폴더
│   │   └── style.css
│   ├── fonts             : 웹폰트 구현을 위한 폰트파일 폴더
│   ├── img               : 이미지 파일 폴더
│   ├── js                : Javascript 파일 폴더
│   │   └── include.js    : (예제를 위한) 컴포넌트 참조 기능 구현
│   ├── scss              : SCSS 파일 폴더
│   └── plugins
│   │   └── something     : 다운받은 플러그인을 폴더 단위로 구분
├── inc                   : 공용 컴포넌트 폴더
│   ├── footer.html       : (예제를 위한) 페이지 하단 컴포넌트
│   └── header.html       : (예제를 위한) 페이지 상단 컴포넌트
├── account               : (폴더예시) 개인정보 관리
│   └── *.html
├── basket                : (폴더예시) 장바구니
│   └── *.html
├── order                 : (폴더예시) 주문관리
│   └── *.html
├── add.html              : (예제를 위한) 추가
├── edit.html             : (예제를 위한) 수정
├── index.html            : (예제를 위한) 목록(첫 페이지)
├── view.html             : (예제를 위한) 상세 내용 조회
└── README.md             : 프로젝트 개요 markdown 파일
```