JSON

- JavaScript Object Notation (자바스크립트 객체 표기법)

#01. Object(객체)의 임시 정의

하나의 변수 안에 다른 하위 변수를 내장하는 특수한 형태의 변수.

하위 데이터에 접근하기 위해서는 점(.)으로 구분한다.

```
객체변수이름.하위변수1이름 = 123;
객체변수이름.하위변수2이름 = 456;
객체변수이름.하위변수n이름 = 999;
```

즉, JSON은 어떤 변수 안에 하위 변수를 내장시키기 위한 Javascript의 표기법이다.

#02. JSON 표기법으로 객체 정의하기

1) 변수들의 그룹으로서의 JSON

하나의 변수에 하위 정보들이 포함되어 있는 변수들의 그룹으로 이해할 수 있다.

```
const 객체이름 = { 이름1: 값1, 이름2: 값2, ..., 이름n: 값n };
```

- key 이름을 지정하고 콜론(:)으로 구분지은 후 값을 명시한다.
  - key(이름)와 value(값)의 쌍으로 이루어진 자료구조
- 두 개 이상의 데이터는 콤마(,)로 구분한다.
- 원칙적으로 key 이름은 따옴표로 감싸는 것이 맞지만 key 이름에 띄어쓰기나 대시(–)가 없는 경우는 따옴표 처리를 생략해도 무관하다.

2) 배열을 포함하는 JSON

JSON의 값에는 지금까지 다룬 모든 형태의 데이터 타입이 지정될 수 있다.

```
const 객체이름 = { 이름1: 숫자, 이름2: 문자열, ..., 이름n: [값1, 값2, 값3 ... 값n] };

console.log(객체이름.이름n[0]);
```

계층 구조

JSON 표기법의 장점은 복잡한 정보 구조를 계층화 하여 표현할 수 있다는 것이다.

```
const json1 = { ... };
const json2 = { ... };

const json3 = {
    data1: json1,
    data2: json2
};
```

단일 형태의 JSON 구조를 별도로 참조하는 것이 아니라 직접 정의하는 패턴

```
const myjson = {
    data1: {
        ...
    },
    data2: {
        ...
    }
};
```

목록 구조

JSON의 value가 배열로 정의되어 있으면서, 각 배열의 원소가 또 다른 JSON 형식인 경우.

단, 이 경우 배열의 원소로 포함되는 JSON 끼리는 동일한 구조를 갖는다.

```
const 객체이름 = {
    key : [ {
        key: value, key: value...
    }, {
        key: value, key: value...
    }, {
        key: value, key: value...
    }, {
        key: value, key: value...
    } ]
}
```

#03. JSON 구조의 확장

1) 존재하지 않는 key를 사용하는 경우

존재하지 않는 key에 대해 출력하거나 다른 변수에 대입할 경우 `undefined`로 처리된다.

2) 존재하지 않는 key에 대한 대입

그 즉시 객체가 확장된다.

빈 객체 확장하기

아무런 key도 정의되지 않은 빈 json 객체에 점진적으로 내용을 덧붙여 나갈 수 있다.

통신프로토콜 : 데이터 통신 규격