Javascript 기초 다지기

Javascript

- 웹 페이지 상에서 요소들의 동작을 제어하는 스크립트 언어
- 현대의 최신 브라우저에서 지배적으로 사용하고 있음
- 과거에는 클라이언트 언어, 최근에는 풀스택 언어로 사용
- 자바와 자바스크립트는 전혀 다른 언어
    - 자바스크립트의 초기 이름은 Mocha -> LiveScript 였으나 자바의 인기에 편승하기 위해 변경

- Javascript 예시

```
function clickButton() {
    alert("you clicked it");
}
```

```
<button onclick="clickButton()">click me</button>
```

Javascript History

- 1995, Netscape (Mozilla) 사의 엔지니어가 브라우저 단의 작은 단위 일을 처리하기 위해 제작
- Mocha -> LiveScript -> Javasript 순으로 명명
- 90년대 중후반에 첫 표준 제정 : ECMA (European Computer Manufacturers Association)
- 90년대 말까지 ECMA 2 -> ECMA 3 -> ECMA 4 지정
- 여러 오픈소스 프로젝트가 진행되면서 2005년 Ajax 등장하고, 또 다른 오픈소스 프로젝트 (jQuery, ...) 들로 이어짐
- 2009, ECMA 5
- 2015, ECMA 6

Javascript **변수**

- 값을 저장하는 저장소, 공간, 용기

```
var bread = "1300 won";
var milk = "1800 won";
var jam = 4000;
```

Javascript 조건문

- 값이 특정 조건에 이르렀을 때 원하는 행동을 취할 수 있음
  - ex) 미세먼지 농도가 최악 (150 이상) 이면 경고 발생

```
var dust = 200;

if (dust > 150) {
  alert("나가지 마세요");
} else {
  alert("죽을 정도는 아니네요");
}
```

Javascript **반복문**

- 동일한 연산을 여러번 수행할 때 반복문을 사용

```
var sum = 0;
sum = sum + 1;
sum = sum + 2;
sum = sum + 3;
...

// 반복문 사용
for (var i = 0; i < 4; i++) {
  sum += i;
  // sum = sum + i;
}
console.log(sum); // 6
```

Javascript 함수

- 특정 목적을 가지는 코드의 집합
- 코드를 더 가지런히 정리하기 위한 목적으로, 함수 마다 기능을 부여하여 작성한다

```
// 함수의 기본 형태
function functionName() {
  ...
}
functionName(); // 함수 호출
```

- 실제 코드에 함수 적용해보기

```
// 정리되지 않은 코드
var loadDOM = "completed";
...
var a = 10;
var b = 20;
var sum = a + b;
console.log(sum);
```

```
// 덧셈 함수로 정리한 코드
function sumNumber() {
  var a = 10;
  var b = 20;
  var sum = a + b;
  return sum;
}

var loadDOM = "completed";
...
sumNumber();
```

Javascript 객체

- 객체란 현실 세계의 사물
  - ex) 나, 너, 책상, 컴퓨터, 패스트캠퍼스 학원..

```
var my = {}; // var my = new Object();
my.name = "josh";
my.height = "177";
my.address = "pangyo";

console.log(my); // Object {name: "josh", height: "177", address: "pangyo"}
```

```
▶ Object {name: "josh", height: "177", address: "pangyo"}
```

Javascript **배열**

- 하나의 변수에 여러 가지 값을 담는 또 하나의 방법
- 피팅룸의 번호처럼, 각 값은 배열 안에서 순서를 갖는다.

```
var clothes = [];
clothes[0] = "shirt";
clothes[1] = "jean";
clothes[2] = "socks";

console.log(clothes); // ["shirt", "jean", "socks"]
```

```
▶ ["shirt", "jean", "socks"]
```

AirBnb JS Guide

Object

- 객체 생성은 `{}` 이용

```
// bad
var obj = new Object();

// good
var obj = {};
```

- 예약어는 사용 금지

```
// bad
var bread = { default: 1000 };

// good
var bread = { defaults: 1000 };
```

Arrays

- 배열 생성도 `[]` 사용

```
// bad
var arr = new Array();

// good
var arr = [];
```

- 배열 아이템 추가는 `.push()` 활용

```
var arr = [];

// bad
arr[0] = "first item";

// good
arr.push("first item");
```

- 배열 아이템 분할은 `.slice()` 활용

```
var len = items.length;
var itemsCopy = [];
var i;

// bad
for (i = 0; i < len; i++) {
  itemsCopy[i] = items[i];
}

// good
itemsCopy = items.slice();
```

> 브라우저 성능 관점에서는 내장 API 사용이 더 도움 된다.

String

- `""` 보다는 `''` 을 사용
- 문자열이 길어질 때는 `\` 보다는 `+` 사용
- IE 에서 성능 향상을 위해서 `+` 보다 `Array.join()` 활용 (연산이 많을 경우 유용)

```
// bad
var str = "hello" + "world" + "!";

// good
var arr = ["hello", "world", "!"];
var result = [];
for (var i = 0; i < arr.length; i++) {
  result[i] = arr[i];
}
result.join('');
```

Function

- 함수 선언식 보다는 함수 표현식 활용

```
// bad
function functionDeclaration() {
  //
}

// good
var functionExpression = function () {
  //
};
```

Variables

- 변수 선언은 항상 `var` 사용
- 여러개 변수 선언시에도 `;` 사용

```
// bad
var items = getItems(),
    goSportsTeam = true,
    dragonball;

// good
var items = getItems();
var goSportsTeam = true;
var dragonball; // 값이 없는 변수는 가급적 뒤에 위치
```

기타

- `,` 는 라인의 끝에
- String to Number 타입 변환은 가급적 `parseInt()`, `Number()` 활용

```
var strValue = '4';
var numValue = Number(strValue);
var intValue = parseInt(strValue);
```

Naming Convention

- 변수, 함수 등 모든 이름에는 각 역할에 맞춰 자세한 네이밍 필요

```
// bad
var i = 0;

// good
var index = 0;
```

- 변수나 함수 이름은 camelCase 기법을 권고

```
var sumNumbers = function () {};
var boardObject = {};
```

최신 트렌드

- TypeScript
- Reactive Programming

참고

- JS History - W3C
- Brief History - Auth0
- JS Convention Guide - AirBnb
- JS Tutorial - Codecademy

끝