Seoul Software ACademy

웹 개발자 부트캠프 과정

SeSAC x CODINGOn

With. 팀 뤼쳐드

JavaScript 객체

JavaScript 자료형

Primitive 자료형

Object 자료형

Primitive 자료형

- Boolean
  - 참(true), 거짓(false) 둘 중 하나의 값을 갖는 요소
- Number
  - 숫자형으로 정수와 부동 소수점, 무한대 및 NaN(숫자 아님)
- String
  - 문자 데이터를 나타낼 때 사용
- Null
  - 빈 값을 뜻하는 null 타입
  - 타입은 존재하지만 값 존재 X
- Undefined
  - 값 X 타입 X

Object 자료형

- 자바스크립트를 이루고 있는 거의 모든 것 객체
- 원시 타입을 제외한 나머지 값(함수, 배열, 클래스 등)은 모두 객체!
- 객체 타입(Object type) or 참조 타입 (Reference type)

- 키(key)와 값(value)로 구성된 프로퍼티(property) 집합
- 프로퍼티 값으로 함수를 쓸 수 있음! == 메소드 (method)

- 객체
  - 프로퍼티(property): 데이터를 의미
  - 메소드(method): 동작을 의미 -> 프로퍼티 값이 함수 일 경우!

객체

실생활에서 우리가 인식할 수 있는 사물

객체 : 고양이 그 자체

속성 :

이름 – 나비

나이 – 1살

메소드 :

mew() – 울다

객체

실생활에서 우리가 인식할 수 있는 사물

```
const cat = {
        name: "나비",
        age : 1,
        mew : function() {
                return "냐옹";
        }
};
```

Object 자료형 - 배열

숫자형과 문자열과 마찬가지로 일반적인 스크립트와 동일

[]나 new Array()를 이용해 생성

```
const arr1 = [1, 2, 3, '안녕', '반가워'];
const arr2 = new Array(1, 2, 3, '안녕', '반가워');
```

Object 자료형 - 배열

- 변수명.length
- 변수명.push(추가할 값)
- 변수명.pop()
- 변수명.unshift(추가할 값)
- 변수명.shift()
- 변수명.indexOf(찾을 값)
- …

Object 자료형 - 딕셔너리

```
const me = {
  name: 'John',
  birth: '0707'
};

me['gender'] = 'F';
me.age = 30;
conslle.log(me);
```

키-값 형태로 저장

기본 자료형 vs 객체 자료형

- 기본 자료형
  - 다른 변수에 값을 할당하거나 함수 인자를 넘길 때 값을 복사해 전달
  - → Pass by value

- 객체 자료형
  - 값을 복사해 전달하는 것이 아닌 메모리 주소를 참조값(address)을 저장
  - → Pass by reference
  - 즉, 같은 객체를 참조할 뿐

JavaScript 표준객체

JavaScript 표준 객체

- 자바스크립트가 **기본적으로 가지고 있는 객체들**
- 프로그래밍을 하는데 기본적으로 필요한 도구들
- String, Number, Array, Date, Math .....

Date 객체

- Javascript 에서 매 순간 바뀌는 시간과 날짜에 관한 정보를 얻기 위해 사용하는 객체

- 초기화

```
new Date()
new Date(밀리초)
    -> new Date(8000000); // 1970년 1월 1일 0시부터 밀리초만큼 지난 날짜
new Date(년, 월, 일, 시, 분, 초, 밀리초)
    -> new Date(16,5,25); // 1916년 5월 25일 00:0:00
    -> new Date(2016,5,25,15,40) // 2016년 5월 25일 15:40:00
```

Date 객체 - 함수

- Date.now()
- Date.prototype getter 메소드 (var date = new Date();)
  - date.getFullYear()
  - date.getDate()
  - date.getDay()
  - date.getTime()
  - date.getHour()
  - date.getMinutes()
  - date.getSeconds()

Math 객체

- 수학에서 자주 사용하는 상수와 함수들을 미리 구현해 놓은 Javascript 표준 내장 객체

- 웹 브라우저마다 다른 결과를 얻을 가능성이 있기에 정확한 결과를 얻어야 할 경우에는 Math 메소드를 사용하지 않는 것이 좋다.

Math 객체 – 함수

- Math.**PI**
- Math.**E**

- Math.**min**()
- Math.**max**()
- Math.**random**()
- Math.**round**()
- Math.**floor**()
- Math.**ceil**()

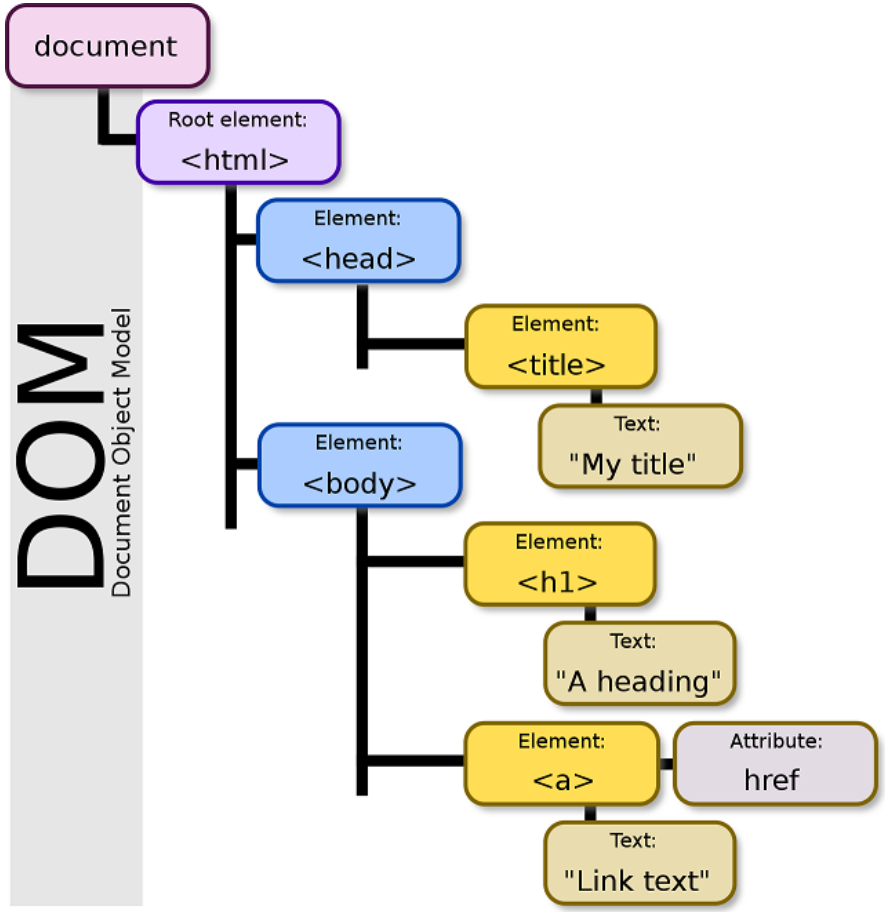
JavaScript DOM

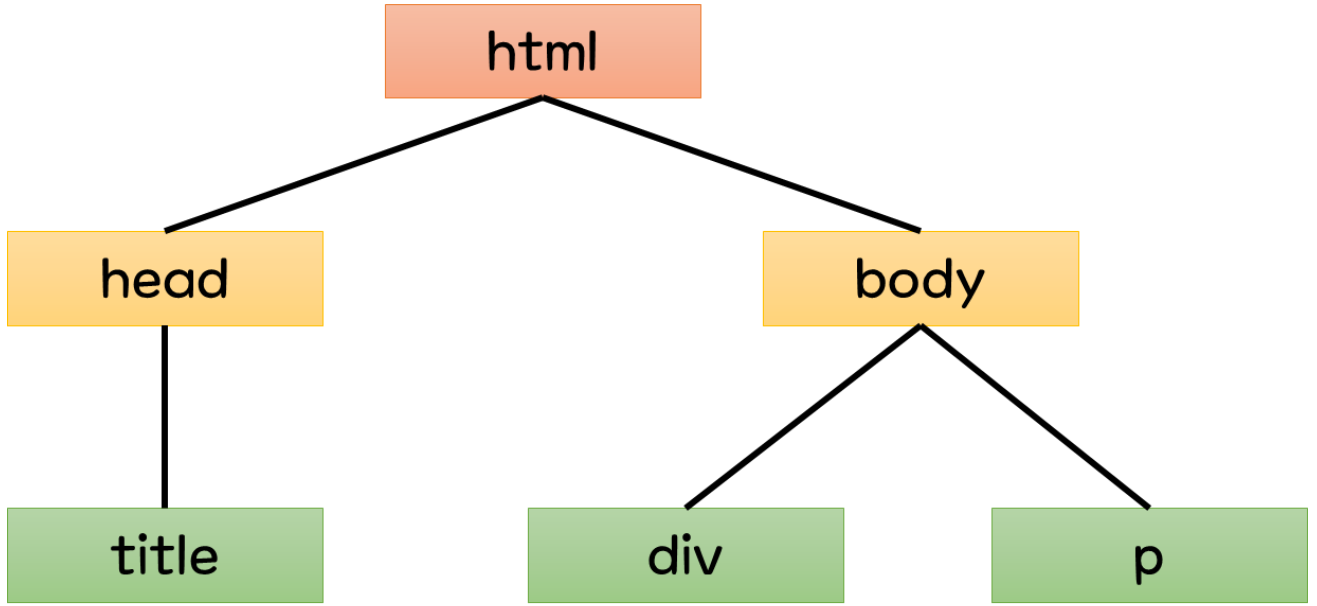
DOM

- **Document Object Model (문서 객체 모델)**

- XML 이나 HTML 문서에 접근하기 위한 일종의 인터페이스로 문서 내의 모든 요소를 정의하고, 각각의 요소에 접근하는 방법을 제공

DOM


document
Root element:
<html>
Element:
<head>
Element:
<title>
Text:
"My title"
Element:
<body>
Element:
<h1>
Text:
"A heading"
Element:
<a>
Attribute:
href
Text:
"Link text"
DOM
Document Object Model


html
head
body
title
div
p

DOM 다음과 같은 일을 할 수 있음!!!

1. 새로운 HTML 요소나 속성 추가
2. 존재하는 HTML 요소나 속성 제거
3. HTML 문서의 모든 HTML 요소 변경
4. HTML 문서의 모든 HTML 속성 변경
5. HTML 문서의 모든 CSS 스타일 변경
6. HTML 문서에 새로운 HTML 이벤트 추가
7. HTML 문서의 모든 HTML 이벤트에 반응

JavaScript Document

Document

- 웹 페이지에 존재하는 **HTML 요소에 접근**하여 행동을 하고자 할 때 사용하는 객체

Document - 속성

- document.documentElement
- document.head
- document.title
- document.body

- document.URL
- document.domain

Document – 요소 선택

- document.getElementById(아이디 속성값)
- document.getElementsByClassName(클래스 속성값)
- document.getElementsByTagName(태그 이름)
- document.getElementsByName(name 속성값)

- document.querySelector(CSS 선택자)
- document.querySelectorAll(CSS 선택자)

querySelector("요소 선택자")

- 요소 선택자를 사용해서 자신이 가져오고 싶어하는 요소를 가져오는 메소드
- 문서에서 만나는 제일 첫번째 요소를 반환 합니다!

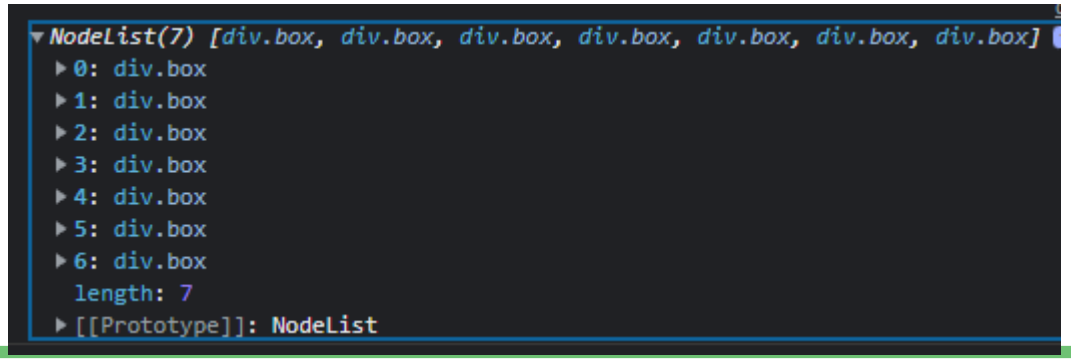
```
let boxEl = document.querySelector(".box");

console.log(boxEl);
```

querySelectorAll("요소 선택자")

- 문서에 존재하는 모든 요소를 찾아주는 메소드
- 모든 요소를 가져와서 배열(같은) 데이터로 만들어 줍니다!

```
<body>
    <div class="box">1</div>
    <div class="box">2</div>
    <div class="box">3</div>
    <div class="box">4</div>
    <div class="box">5</div>
    <div class="box">6</div>
    <div class="box">7</div>
</body>
```

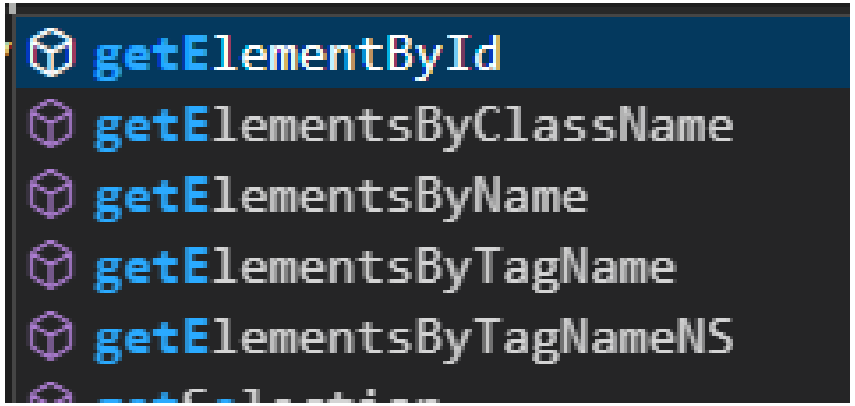
```
let boxEls = document.querySelectorAll(".box");
console.log(boxEls);
```



getElementById("ID이름")

- 해당 ID를 가지는 요소를 불러오는 메소드

```
const inputEl = document.getElementById("input");
```



Document – 요소 다루기

- document.createElement(html요소)
- document.write(텍스트)

- [].appendChild();
- [].removeChild();

- [].append();
- [].remove();

- [].innerText = 내용;
- [].className = 클래스 이름;

.textContent .innerText .innerHTML

- **태그 내에 들어갈 문자열을 지정**

  요소.textContent=“hi”;

  선택된 요소에 내부의 문자열로 hi가 들어가게

  됩니다.

classList.~

- 선택 요소에 class 를 더하거나, 빼거나, 클래스가 존재하는지 체크하는 메소드
- 해당 기능과 CSS 를 잘 활용하면 액티브한 웹페이지 구성이 가능
  - 요소.classList.add()
  - 요소.lassList.remove()
  - 요소.classList.contains()
  - 요소.classList.toggle()

setAttribute, html 요소 속성 추가

- 선택한 요소의 속성 값을 직접 지정할 수 있는 메소드
- 요소.setAttribute("속성명", "지정할 속성")

```
searchInputEl.setAttribute("placeholder", "통합검색");
```

다른 노드에 접근하기

- 특정 노드를 선택한 후, 그 노드의 형제, 부모, 자식 노드에 접근하는 방법

  - 요소.children / 요소.children[0]
  - 요소.parentNode
  - 요소.previousElementSiblings
  - 요소.nextElementSiblings

createElement('html 요소')

- html 의 특정 노드를 생성
- 괄호안에는 html의 요소인 태그명을 넣어주시면 됩니다!

```
let p = document.createElement('p');
```

요소를 만들었으면 추가해야겠죠?

- 요소.append() /요소.appendChild()
  - 선택된 요소의 맨 뒤의 자식 요소로 추가됨
- 요소.prepend()
  - 선택된 요소의 맨 앞쪽인 자식 요소로 추가됨
- 요소.before()
  - 선택된 요소의 앞에 있는 형제 요소로 추가됨
- 요소.after()
  - 선택된 요소의 바로 뒤인 형제 요소로 추가됨

요소 삭제, remove() removeChild()

- 요소.remove();
  - 선택된 요소가 삭제 됩니다.
- 요소.removeChild('요소의 자식요소');
  - 선택된 요소의 자식 요소가 삭제 됩니다.

Document – 요소 다루기

append()와 appendChild()
차이점은?

Document – 요소 다루기

remove()와 removeChild()
차이점은?