JavaScript

Part. 2

2024년 2월

출석 확인하셨나요?

목차

1. 매개변수를 갖는 함수

1. 일반적인 함수
   function 함수명 () {실행문;}함수를 호출할때 함수의 앞, 뒤에서 호출가능한다

2. 익명함수
   변수에 함수 데이터를 저장하여 변수를 마치 함수처럼 사용할 수 있도록 만들어준다 이벤트 핸들러 작성 시 주로 사용하며 변수 선언 이후에 호출해야 한다

   함수명 = function(){실행문;}

3. 스스로 실행하는 함수
   (function(){실행문;})();
   선언과 동시에 함수가 실행되며 함수명이 없기 때문에 재호출할 수 없다

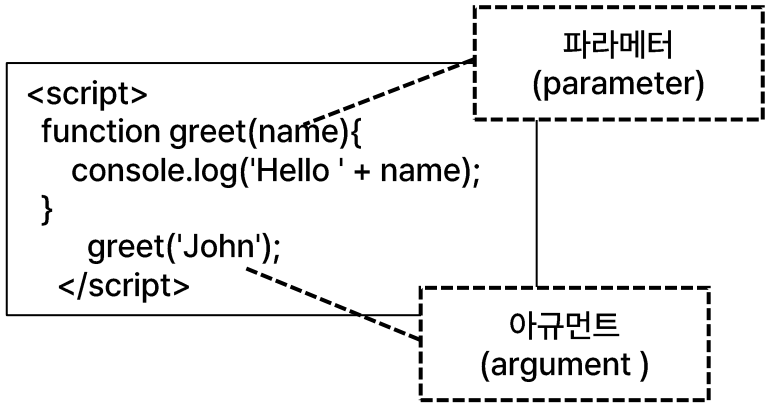
2. Scope (Local vs Global)

- Scope (스코프)는 variable 또는 constant 가 결정한다.
- 변수선언에 사용되는 var 가 있다 오랫동안 사용해 왔던거지만 여러가지 문제로 인해 let, const 가 추가되었다

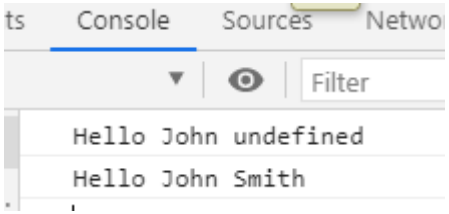
1. var선언되면 window객체로 접근가능하다
2. this 키워드

3. 함수2

- Arguments
- Default parameter




```
function greet(name, lastName){
    console.log('Hello ' + name + ' '+
lastName);
 }
     greet('John');
     greet('John','Smith');
```



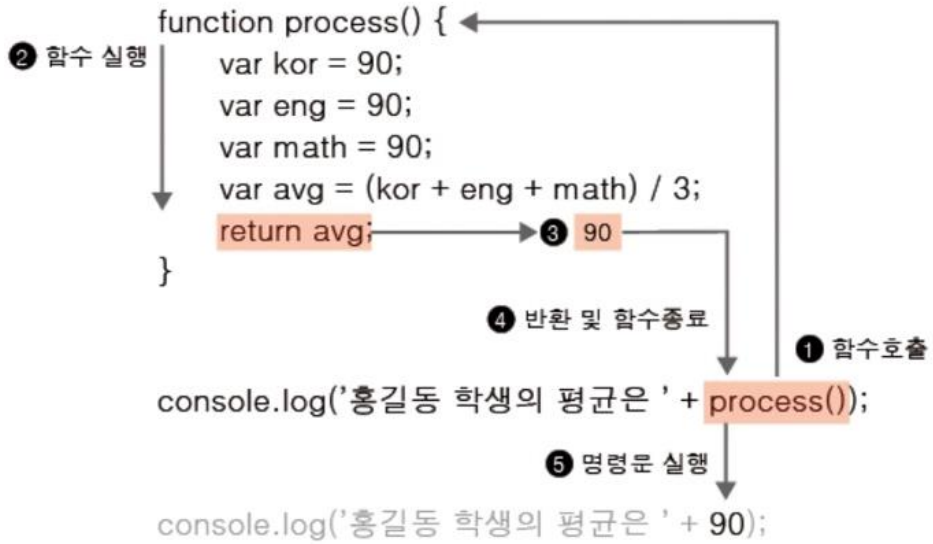

```
function square(number) {
    return number * number;
}
let number = square(2);
console.log(number);
```

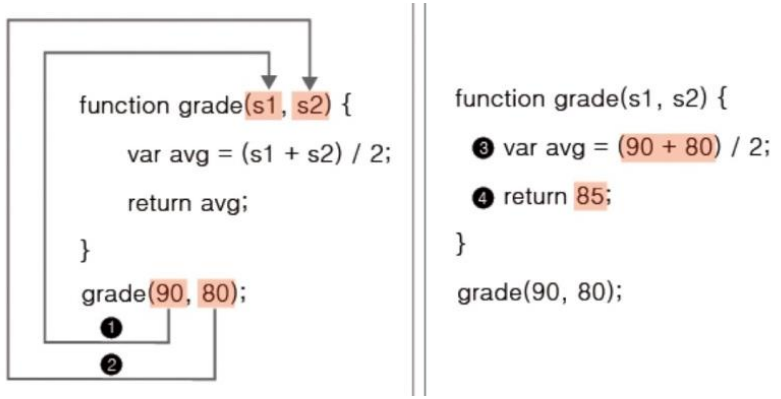
```
function sum(a,b){
// console.log(arguments)
   return a+b
}
console.log(sum(1,2))
console.log(sum(1) )
// 1+ undefined = NaN
console.log(sum(1,2,3,4) )// 1,2 만
// 모든 함수에는 arguments 라는 특별한
// 오브젝트를 갖고 있다. --> 옆의코드 해결
```

```
function sum(a,b){
 let total=0;
 for( let value of arguments)
     total += value
   return total;
}
console.log(sum(1,2,3,4))

//파라메터없이도가능,배열X,오브젝트다
// function sum()(OK)
```

Return 문





- Default 지정

사용자의 부주의로 입력이 제대로 주어지지 않을때 default 파라메터를 지정할 수 있다

```
function bank( deposit, rate, year){
    return deposit *rate /100 *year;
}
console.log(bank(1000, 3.5, 5))
```

```
function withdraw2( deposit, rate=3, year=5){
    return deposit *rate /100 *year;
}
```

- 자동실행함수 (immediately invoked function 표현)
  함수의 호출은 일반적인 경우 함수의 이름을 호출함으로 실행하지만 자동실행함수는 선언과 동시에 실행

  새 js파일에서 실행한다

4. JavaScript 객체

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";
        }
};
```

객체 (Object)

- 자바스크립트를 이루고 있는 거의 모든 것이 객체라고 할 수 있다.
- Object 타입에는 object, array, function 속한다.
- primitive type 과 달리 다양한 데이터를 담을 수 있다 키로 구분된 데이터 집합이나 복잡한 개체를 저장할 수 있다
- 중괄호 { } 를 사용하고 키 (key) : 값 (value) 쌍으로 구성된 프로퍼티 (property) 를 여러개 가질 수 있다 .
- key (문자형), value (모든 자료형이 허용), 앞의 key 를 ' 프로퍼티 이름 ' 이라고 부른다

a) 객체를 만드는 방법

b) 객체를 읽어오는 방법

c) 객체의 프로퍼티 추가, 삭제

d. Const 로 선언된 객체는 수정될 수 있다.

e. 변수를 키로 사용할 수 있다

f. 계산된 프로퍼티 (computed 프로퍼티)

g. 단축 프로퍼티

h. 메서드 : 객체안에 능력을 부여해 줄 수 있다 . 함수표현식으로 함수를 만듭니다 . 객체 프로퍼티에 할 당된 함수를 메서드 (method) 라고 부른다

i. 메서드 단축구문

j. this의미

k. 화살표 연산자

a. 객체를 만드는 방법

1. Object() 객체를 new 키워드와 함께 사용
2. 리터럴방식을 이용
3. 함수를 통해 객체를 생성

b. 객체를 읽어오는 방법

1. Dot 표기법
2. [대괄호] 표기법

c. 객체의 프로퍼티 추가, 삭제

JavaScript

```
let user = {
  name: "John",
  age: 30,
  "likes birds": true // 복수의단어는 따옴표로
};
// get
alert(user["likes birds"]); // true
// delete
delete user["likes birds"];
```

d. Const 로 선언된 객체는 수정될수 있다

JavaScript

```javascript
const user = {
  name: "John"
};
user.name = "Pete"; // (*)
alert(user.name); // Pete
```

e. 변수를 키로 사용할 수 있다

JavaScript

```
let key = "likes birds";
let user = {
  name: "John",
  age: 30,
};

// user["likes birds"] = true; 와 같습니다.
user[key] = true;
```

f. 계산된 (computed) 프로퍼티

- 객체를 만들때 리터럴안의 프로퍼티 키가 대괄호로 둘러싸여 있는 경우, 이를 계산된 프로퍼티(computed property)라고 부른다

```
let fruit = prompt("어떤 과일을 구매하시겠습니까?", "apple");
let bag = {};
bag[fruit] = 5;
```

```
let fruit = 'apple';
let bag = {  [fruit + 'Computers']: 5 };
// bag.appleComputers = 5
```

g. 단축 프로퍼티

```
function makeUser(name, age) {
  return {
    name: name,  --> name,
    age: age,    --> age,
    // ...등등
  };
}

let user = makeUser("John", 30);
alert(user.name); // John
```

키와 value의 값이 같은 경우 축약해서 사용할 수 있다
아래의 예제는 옆의 예제와 동일한 결과를 출력한다

```
function makeUser(name, age) {
    return {
    name,
    age,
        };
    }
```

h. 메서드

- 객체안에 능력을 부여해 줄 수 있다 . 함수표현식으로 함수를 만듭니다 . 객체 프로퍼티에 할 당된 함수를 메서드 (method) 라고 부른다
- 메서드는 함수로 만든다.

```
let user = {
  name: "John",
  age: 30
};
```

```
user.sayHi = function() {
  alert("안녕하세요!");
};

user.sayHi(); // 안녕하세요!
```

i. 메서드 단축구문

```
//es5에서
var obj = {
name :'Lee',
sayHi : function(){
 console.log('Hi es5 ~ ' + this.name)
}
}
obj.sayHi();
```

```
// es6에서 function 키워드 생략한 표현을
사용할 수 있다.
var obj2 = {
 name : 'Lee',
 sayHi(){
   console.log('Hello es6 ~' + this.name)
    }
}
obj2.sayHi();
```

j. this 키워드

- 자바스크립트의 모든 함수에서 this 를 사용할 수 있는데 this 의 값은 런타임에 결정된다 . 예기치 않은 결과를 얻기도 한다 . 그래서 규칙은 간단하다 . obj.f() 를 호출했다면 this 는 f 를 호출하는 동안의 obj 이다

```
let user = {
  name: "John",
  age: 30,

  sayHi() {
    // 'this'는 '현재 객체'를 나타냅니다.
    alert(this.name);
   }

};

user.sayHi(); // John
```

전역객체 (global object) 는 코드 실행 이전단계에서 자바스크립트 엔진에 의해 어떤 객체보다 먼저 생성되는 특수한 객체로 this를 정확히 지정하지 않으면 전역객체가 반환된다

ECMAScript 2020에서 혼란을 없애기 위해 기존의 전역객체 this 를 globalThis로 명명하기로 했다.
globalThis === this
globalThis === window
globalThis === self

k. 화살표 연산자

- es6에 추가된 내용으로 화살표를 이용하여 함수를 간결하게 표현할 수 있다.

```JavaScript
let func = (arg1, arg2, ..., argN) => 표현식;
let func = function(arg1, arg2, ..., argN) {
 return expression;
};
```

5. Javascript 표준객체

자바스크립트가 기본적으로 가지고 있는 객체들

가. Date()
나. Math()

가. Date 객체 - 함수

- Javascript 에서 매 순간 바뀌는 시간과 날짜에 관한 정보를 얻기 위해 사용하는 객체
- 이미 만들어진 객체로 호출해서 사용하되 new 키워드를 사용해야 한다.

- Date.now()
- 일반적인 사용 형태 var date = new Date();)
  - date.getFullYear()
  - date.getDate()
  - date.getDay()
  - date.getTime()
  - date.getHour()
  - date.getMinutes()
  - date.getSeconds()

사용 예)
const now = new Date();
const date1 = new Date("April 11 2018 09:00");// 기준 1970.1.1
const date2= new Date(2018,3,11,9,0);
console.log(now, date1 ,date2) now.setFullYear(2017) // 년도 바꾸기

브라우저 콘솔창에서 다음의 내용을 실행
>now.toDateString()
>now.toTimeString()
>now.tolSOString()

나. Math 객체 - 함수

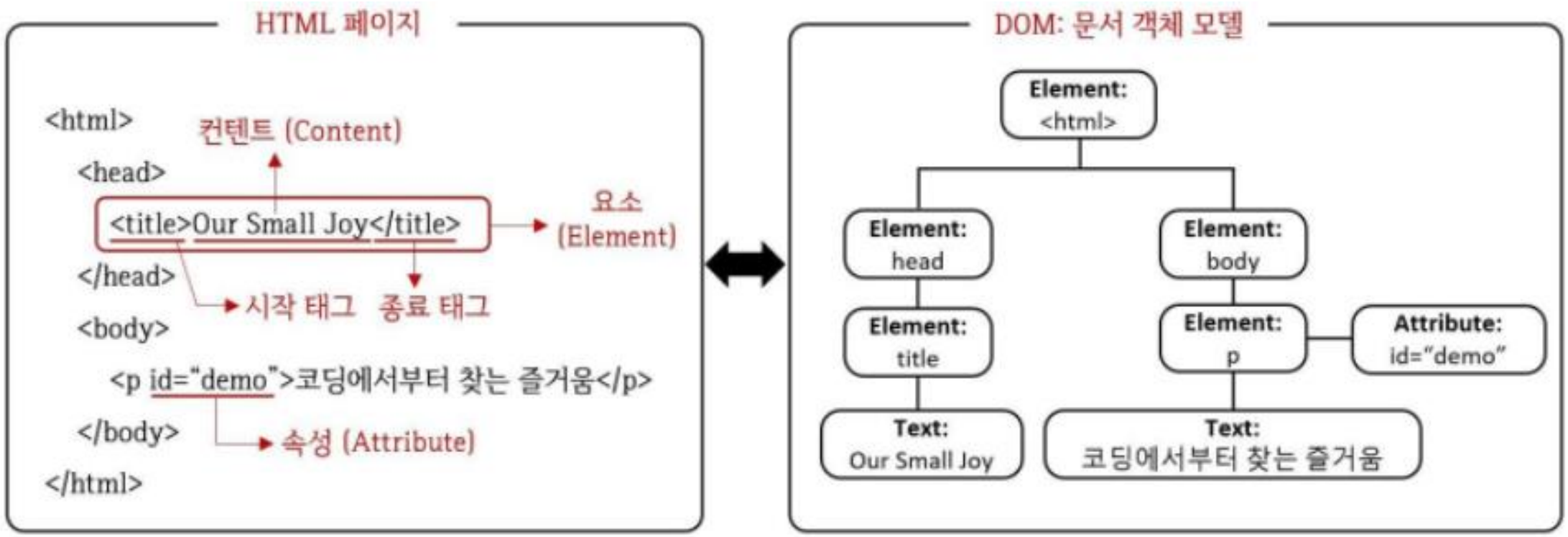
- 수학에서 자주 사용하는 상수와 함수들을 미리 구현해 놓은 Javascript 표준 내장 객체
- 웹 브라우저마다 다른 결과를 얻을 가능성이 있기에 정확한 결과를 얻어야 할 경우에는 Math 메소드를 사용하지 않는 것이 좋다.

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

- Math.PI
- Math.E

6. JavaScript DOM

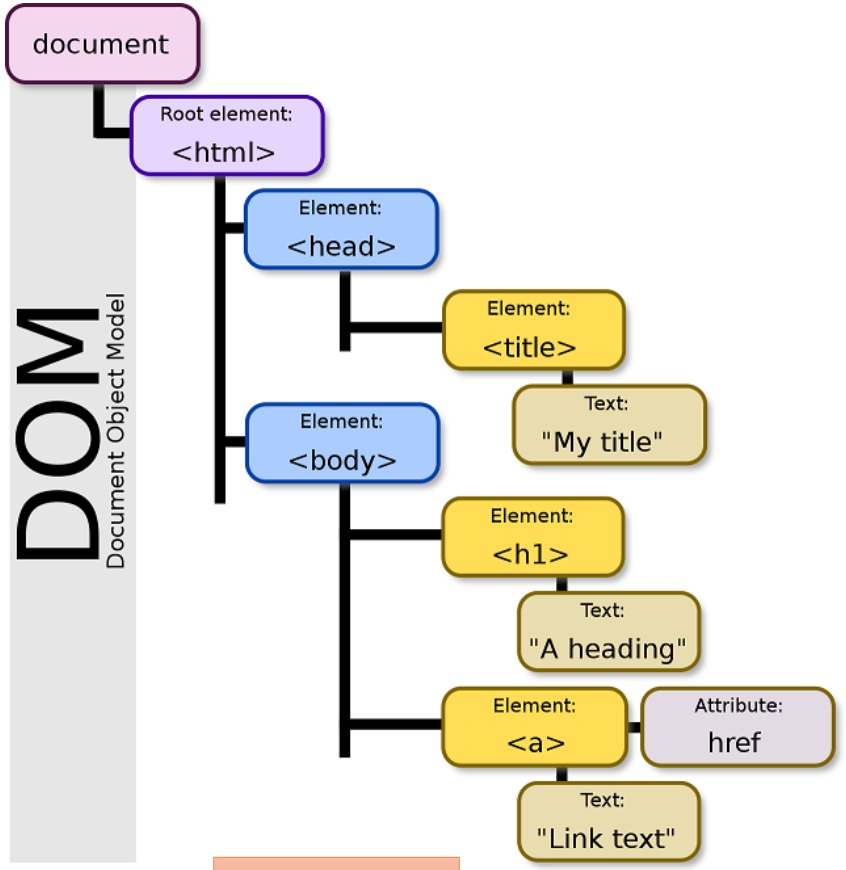
DOM은 도큐먼트 객체 모달(DOM: Document Object Model) 이다. Document는 문서이고 Object는 객체로 번역이 된다. 그리고 Model은 말 그데로 모델 이다 문서 객체란 이나 같은 html문서의 태그들을 JavaScript가 이용할 수 있는 객체(object)로 만들면 그것을 문서 객체라고 한다. 여기서는 문서 객체를 '인식하는 방식'이라고 해석할 수 있다 DOM을 보게 되면 웹 브라우저가 html 페이지를 인식하는 방식을 이야기 하거나 문서 객체(document object)와 관련된 객체의 집합에 관한 이야기라는 것을 쉽게 추측할 수 있을것이다.

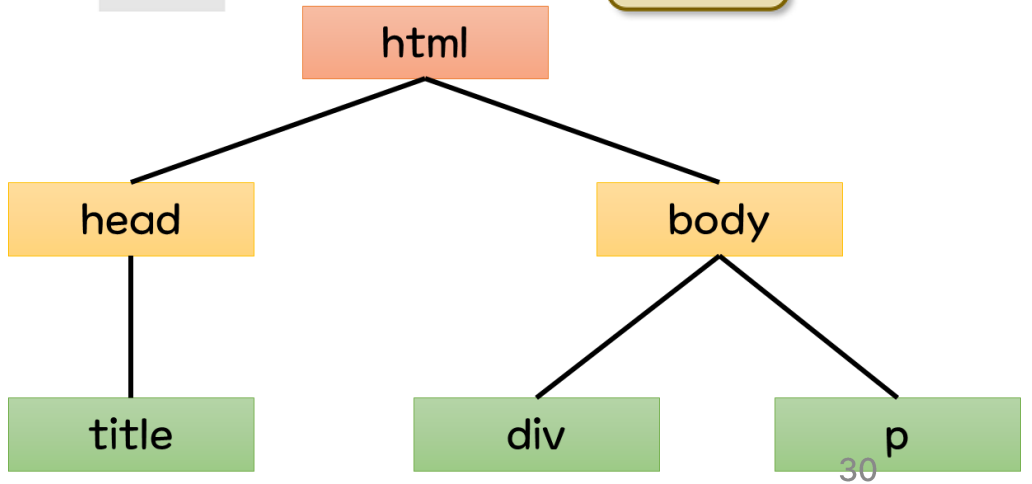

HTML 페이지
<html>
<head>
<title>Our Small Joy</title>
</head>
<body>
코딩에서부터 찾는 즐거움
</body>
</html>
컨텐트 (Content)
요소 (Element)
시작 태그
종료 태그
속성 (Attribute)
DOM: 문서 객체 모델
Element: <html>
Element: head
Element: body
Element: title
Element: p
Attribute: id="demo"
Text: Our Small Joy
Text: 코딩에서부터 찾는 즐거움

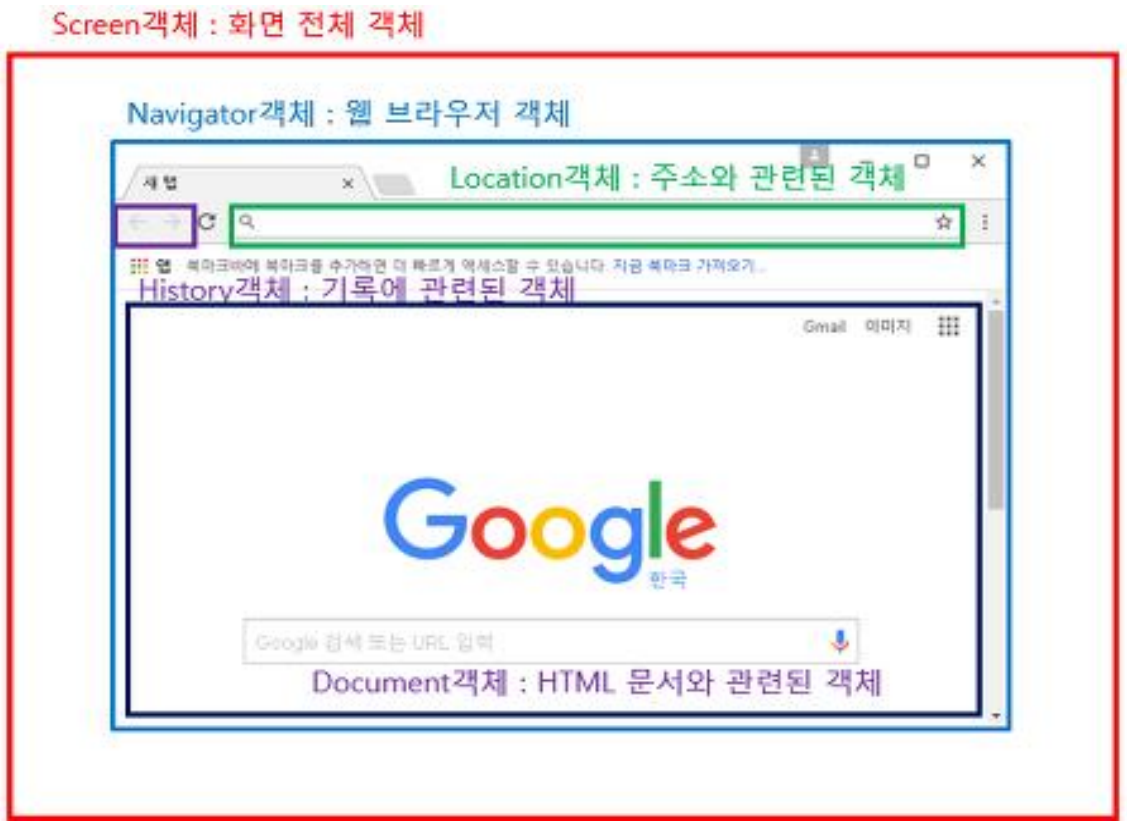
DOM

- Document Object Model (문서 객체 모델)

- XML 이나 HTML 문서에 접근하기 위한 일종의 인터페이스로 문서 내의 모든 요소를 정의하고, 각각의 요소에 접근하는 방법을 제공
- DOM은 계층적으로 구성된 HTML문서를 도식화하면 나무모양과 비슷하다고 해서 DOM트리 하고 부른다







- DOM은 자식(child),형제 (sibling), 부모 (parent)과 같은 표현을 이용하여 DOM트리에 접근하여 구조를 재구성하거나 내용을 바꾼다.
- 크롬 브라우저 디버깅 > 콘솔창 > $0의 의미
  - 현재 선택된 엘리먼트 하나를 0이라는 임시변수에 할당하고 이것을 $0으로 하여 접근할 수 있다.

상위에 있는 window 객체는 전역객체로써 하위객체인 document 를 갖는다
window.document 또는 document 이다
(window 는 생략해도 된다)

- 문서객체 (DOM)
- 브라우저 객체 모델(BOM)
  1. History객체
  2. Location 객체
  3. Screen 객체
  4. Navigator 객체

DOM이 할 수 있는 일

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

DOM을 다루는 명령어들

	output	HTML 엘리먼트 찾기	HTML 엘리먼트 변경	엘리먼트 추가 & 삭제
1	innerHTML	document.getElementById(*id*)	*element*.innerHTML = *new html content*	document.createElement(*element*)
2	document.write()	document.getElementsByTagName(*name*)	*element.attribute = new value*	document.removeChild(*element*)
3	window.alert()	document.getElementsByClassName(*name*)	*element*.style.*property = new style*	document.appendChild(*element*)
4	console.log()		*element*.setAttribute*(attribute, value)*	document.replaceChild(*new, old*)
		querySelector()	*element*.getAttribute(*attributeName*)	document.write(*text*)
		querySelectorAll()		

이벤트란

- 이벤트(Event)는 어떤 사건을 의미합니다. 브라우저에서의 사건이란 사용자가 "클릭을 했을 때", "스크롤을 했을 때", "무언가 입력했을 때" 등의 상호작용으로 인해 일어나는 사건을 의미하는데, DOM 요소와 관련이 있습니다

이벤트 리스너(Event Listener)

이벤트가 발생했을 때 그 처리를 담당하는 함수를 이벤트 리스너 라 한다

Events Handlers
document.getElementById(*id*).onclick = function(){*code*}
document.getElementById("myBtn").addEventListener("click", displayDate);

element.addEventListener("mouseover", myFunction);
element.addEventListener("click", mySecondFunction);
element.addEventListener("mouseout", myThirdFunction);

element.removeEventListener("mousemove", myFunction); window.addEventListener("resize", function(){ });

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)

Document – 요소 다루기

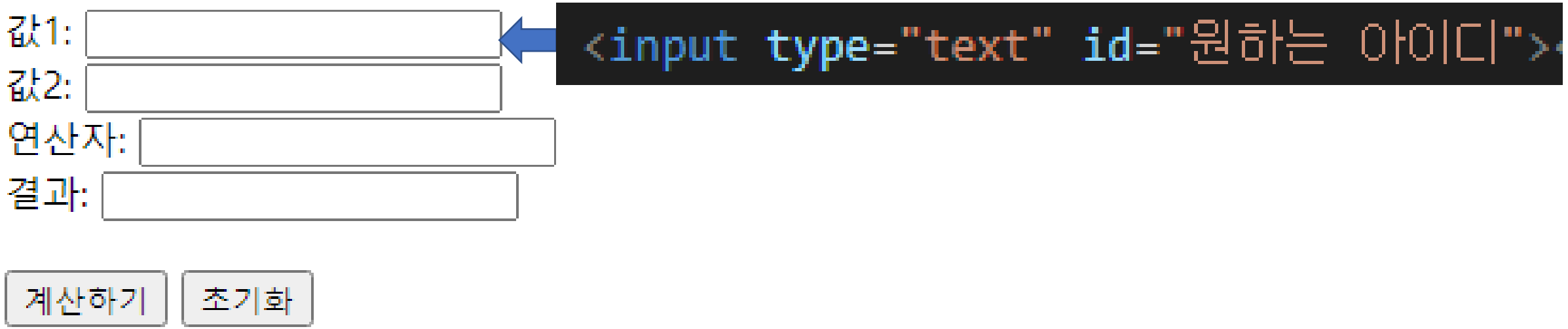
- document.createElement(html요소)
- document.write(텍스트)

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

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

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

실습. 간단 계산기 만들기



힌트 1) id 속성이 “value1”인 input 값 가져와 변수에 넣기

변수 = document.getElementById(“value1”).value;

힌트 2) id 속성이 “value1”인 input에 값 넣기

document.getElementById(“value1").value = 값;

실습. 방명록 만들기

방명록입니다.

작성자
내용

작성

번호	작성자	내용	작성일
1	김규리	여러분 반가워요	2022-7-2 15 : 3
2	김규리	오늘 수업도 열심히 해봐요	2022-7-2 15 : 3
3	홍길동	안뇽!	2022-7-2 15 : 4

작성자와 내용을 쓰고 “작성” 을 누르면 아래 table 에 추가

이때, 작성일은 작성한 시간이 되어야 한다.