JavaScript

JavaScript

자바스트립트(JavaScript)는 웹 브라우저에서 많이 사용하는
인터프리트 방식의 객체지향 프로그래밍 언어로
ECMA(European Computer Manufacturers Association) 스크립트 표준을
따르는 대표적인 웹 기술

▶ 자바스크립트 선언

HTML에서 제공하는 <script></script>태그를 사용하며
자바스크립트 작성 영역을 설정하고 그 사이에 자바스크립트 코드 작성
type속성은 브라우저 호환성을 위해 사용되나 default값으로 생략 가능

```
<script type="text/javascript">
        자바스크립트 내용
</script>
```

* language속성과 charset속성이 있었지만 language속성은 폐기되고
  charset속성은 meta태그가 적용되기 때문에 사용할 필요 없음

▶ 자바스크립트 위치

<script></script>는 <head>, <body> 안 어느 영역에나 작성 가능

특히 <html>태그 영역 밖에서 작성도 가능하지만 웹 표준과 웹 접근성을 고려해 <head>나 <body>안에 작성함

```
<script>자바스크립트 내용</script>
<html>
        <head><script>자바스크립트 내용</script></head>
        <body><script>자바스크립트 내용</script></body>
</html>
<script>자바스크립트 내용</script>
```

▶ 자바스크립트 작성 방식

KH정보교육원

- inline 방식 : 자바스크립트 양이 한 두 줄 정도로 소량일 때 사용
  태그에 이벤트 핸들러 속성을 이용하여 직접 실행 코드 작성
- internal 방식 : 가장 일반적인 방식으로 html파일 내
  <head>나 <body>안에 자바스크립트 소스 작성
- external 방식 : 자바스크립트 양이 많을 경우 자바스크립트 코드 부분을
  외부 파일로 저장하여 작성
  <script src="경로">태그를 이용해 내용 삽입 후 사용

자바스크립트 작성 방식

✓ inline 방식

html 태그의 on이벤트 속성을 이용하여 내장 메소드를 호출하거나
개발자가 선언한 사용자 정의 함수를 호출할 때 사용

<태그 명 on이벤트 = "함수명();">

✓ internal 방식

자바스크립트 코드 작성, 함수 단위로 소스코드를 작성하고
html태그에서 이벤트 핸들러를 통해 함수를 실행시키는 방식

<script>실행할 소스코드 작성</script>

▶ 자바스크립트 작성 방식

✓ external 방식

외부에 별도의 자바스크립트 소스파일(*.js)을 작성하고
html에서 <script src="경로.js">태그를 이용하여 해당 파일을 불러와
사용하는 방식으로 여러 개 html파일에서 공통적으로 사용하는
기능일 경우 많이 씀

<script src="경로"></script>

▶ 자바스크립트 실행 방식

인터프리터 방식은 웹 브라우저에 내장되어 있는 자바스크립트 파서가
소스코드를 한 줄씩 읽고 해석함
전체를 해석해 놓은 컴파일 언어와는 차이가 있음
자바스크립트 실행은 작성된 html문서를 브라우저에서 읽으면
바로 실행을 할 수 있음

* 자바스크립트를 지원하지 않는 브라우저를 대비해 출력문구를 <noscript>에 작성

```
<noscript>
        지원하지 않을 경우 출력문구
</noscript>
```

▶ 주석 처리

주석 종류		설명
자바스크립트 주석	한 줄 주석	// 로 시작
	여러 줄 주석	/*로 시작하고 */로 끝남
HTML주석		<!--로 시작하고 -->로 끝남 자바스크립트나 스타일시트 태그 이외의 HTML태그에서는 이 주석 태그를 사용해서 주석 처리를 해주어야 함

데이터 입·출력

▶ 데이터 출력

코드	설명
document.write(내용);	브라우저 화면 상의 페이지에 값 출력
window.alert(내용);	내용을 메시지 창에 출력 * window객체는 모두 적용되는 것으로 생략 가능
innerHTML = 내용;	태그 엘리먼트의 내용을 변경하여 출력
console.log(내용);	개발자 도구 화면의 콘솔에 출력

▶ 데이터 입력

자바스크립트 내장객체인 window 객체가 제공하는

confirm(), prompt()메소드를 사용하여 입력 받을 수도 있고

HTML태그에 접근해 대상의 값을 읽거나

HTML form태그의 input 입력 양식을 통해 값을 입력 받을 수도 있음

▶ 데이터 입력

✓ window.confirm()

어떤 질문에 대해 "예/아니오"의 결과를 얻을 때 사용

대화 창에 메시지와 확인, 취소 버튼 표시 * 리턴 값 : 확인(true), 취소(false)

var 변수 = [window.]confirm("질문내용");

✓ window.prompt()

텍스트 필드와 확인/취소버튼이 있는 대화 창 출력 * 리턴 값 : 입력한 내용

var 변수 = [window.]prompt("메시지");

기본 문법

▶ 변수

✓ 변수 선언

- 변수 종류 : 멤버 변수와 지역 변수
- 멤버 변수 : 전역 변수, 기본적으로 window객체의 멤버 변수

변수에 대한 자료형은 있으나 선언하지는 않음

형식	설명
변수명 = 값;	window.변수 명 또는 this.변수명과 같은 의미 선언 시 변수 명에 var를 붙이지 않으면 전역변수로 간주
var 변수명 = 값;	변수 선언 시 변수명 앞에 var를 붙이면 지역변수

▶ 변수

✓ 명명 규칙

- 영어 대/소문자, 숫자, _, $ 사용 가능
- 첫 글자 숫자 사용 불가
- 예약어 사용 불가
- 한글 사용 가능
- 생성자 함수는 항상 대문자로 시작
- 변수, 인스턴스, 함수, 메소드는 항상 소문자로 시작
- 이름에 의미 있는 단어의 조합 권장
- 두 단어 결합 시 낙타봉 표기법 권장

▶ 변수

✓ typeof()

값의 자료형을 확인하는 연산자

* 선언 시 자료형을 지정하지 않아 변수 명을 보고 데이터 확인 불가
  자료형 확인 시 자주 사용

typeof("문자열값") 또는 typeof('문자열값') → string

typeof(숫자) → number

typeof(참/거짓) → boolean

typeof(객체) → object

typeof(초기값이 없는 변수) → undefined

typeof(function) → function

▶ 자료형

✓ 문자열(String)

"", ''로 묶여있는 리터럴

내장 객체로 String객체, 기본적인 메소드 존재

메소드	내용
toUpperCase()	모든 문자 대문자로 변환
toLowerCase()	모든 문자 소문자로 변환
length	글자 개수 조회용 멤버변수
indexOf()	찾는 문자의 순번(위치) 리턴
lastIndexOf()	뒤에서부터 찾는 문자의 순번 리턴
charAt()	찾는 위치의 문자 리턴
substring()	값을 일부분만 리턴
split()	토큰 문자로 분리한 문자열 배열 리턴

▶ 자료형

✓ 숫자(number)

정수형 숫자와 부동소수점 숫자로 구분

내장 객체로 Math객체 제공, 기본 메소드 존재

메소드	내용
Math.abs()	절대값 리턴
Math.random()	임의의 난수발생 리턴(소수점)
Math.round()	반올림처리 후 리턴
Math.floor()	부동소수점 숫자를 정수로 리턴(소수점 자리버림)
Math.ceil()	소수점 자리에서 무조건 올림

* NaN(Not a Number) : 숫자가 아닌 데이터를 숫자처럼 사용할 때 출력

▶ 자료형

✓ 기타 자료형

- 논리값(Boolean)

  true, false 두 가지 값을 가짐

- 객체(Object)

  new로 선언된 사용자 객체와 자바스크립트 내장 객체

- undefined

  변수 명이나 함수 명으로 선언되지 않은 식별자일 때 지정

- 함수(function)

  함수(메소드)를 가지는 자료형

▶ 데이터 형변환

✓ 숫자 → 문자열

숫자와 문자를 +연산하게 되면 문자가 우선되어 숫자를 문자로 변환

강제 형변환 : String()함수 이용

✓ 문자열 → 숫자

숫자, 문자 + 이외의 사칙 연산 시 숫자가 우선돼 문자를 숫자로 변환

강제 형변환 : Number(), parseInt(), parseFloat()함수 이용

* parseInt()함수는 인자가 2개로 문자열, radix(해당진수)를 선택할 수 있음

▶ 연산자

KH정보교육원

연산자 종류	연산자
최우선 연산자	(), [], .
단항 연산자	++, --, + sign, - sign
산술 연산자	+, -, *, /, %
관계 연산자	>, <, >=, <=, ==, !=, ===, !==
논리 연산자	&&, ||
대입 연산자	=
복합대입연산자	+=, -=, *=, /=, %=
삼항 연산자	? : ;

삼중 : 자료형까지 비교

자료형에 상관없이 비교

* 연산자 우선순위 : 최우선 > 단항 > 산술 > 관계 > 논리 > 삼항 > 대입

▶ 제어문

✓ 조건문

if, if~else, if~else if~else, switch문, 짧은 조건문(&&, ||)

✓ 반복문

for, while, do~while, for in문

✓ 분기문

continue, break문

HTML태그 접근

메소드

메소드	설명
getElementById("아이디명")	태그의 id 속성 값을 이용해 태그 엘리먼트 객체의 정보를 가져 옴
getElementsByName("이름")	태그의 name속성 값을 이용해 태그 엘리먼트의 객체 정보를 배열에 담아 가져 옴 같은 이름의 태그가 여러 개 존재할 수 있기 때문에 기본적으로 배열로 리턴
getElementsByTagName("태그명")	태그 명을 이용하여 해당 태그들의 객체 정보를 배열에 담아 가져옴

document.getElementById()

HTML태그의 id속성 값은 페이지에서 유일한 식별자 역할을 하도록 권장

* 리턴 값 : 단일 값(id는 중복 허용 안 함)

var 변수 = document.getElementById("아이디명");

* 변수는 객체를 의미하는 레퍼런스 변수

document.getElementsByName()

HTML태그의 name속성 값으로 객체 정보를 가져올 때 사용

* 리턴 값 : 배열(name은 중복 가능)

var 변수 = document.getElelemtsByName("이름");

* 변수는 배열이 됨

document.getElementsByTagName()

HTML태그의 태그 명을 이용해 태그들을 한꺼번에 가져와 순서대로 반환

* 리턴 값 : 배열(태그 중복 가능)

var 변수 = document.getElementsByTagName("태그 명");

* 변수는 배열이 됨