2. 자바스크립트 기본문법

1) 기본용어

(1) 식과 문

- 식은 표현식이라고도 하며 연산식이나 실제 할당하는 값, 함수를 실행하는 것도 식이라고 할 수 있다.
- 문은 명령이라고 생각할 수 있으며 ;(세미콜론)으로 구분한다. 조건문이나 제어문이 이에 해당하며 넓은 의미로 문은 값이나 식까지 포함하고 있다고 볼 수 있다

(2) 식별자

식별자란 개발자가 변수나 함수,속성 등 구분을 위해 붙이는 특정 단어를 의미하는데 작성시 몇가지 규칙이 있다.

- 키워드(예약어)를 사용할 수 없고 숫자나 공백을 허용하지 않는다.
- 특수문자는 _과 $만 허용하며 알파벳으로 작성하고 첫 글자는 숫자가 올 수 없다.
- 생성자 함수는 첫 문자를 대문자로 작성한다.

*자바스크립트 문서 제공 사이트 https://www.w3schools.com/jsref, http://devdocs.io/javascript

2) 변수

자바스크립트에서는 변수 선언 시 따로 자료형을 명시하지 않고 할당된 값에 따라 성질이 정해진다.

다만 변수 선언시 변수 앞에 변수의 범위와 성질을 결정하는 키워드를 붙일 수 있다.

1. var

```
var varString; // var 변수 선언 : 전역적 지역에서 유효한 범위를 갖는다.
   // var 변수에 초기 값을 지정하지 않는다면, 변수는 값이 설정될 때까지 undefined 값을 갖는다.
   // 자바스크립트에서는 var 변수를 스크립트 내 구문이 실행되기 전에 맨 먼저 선언을 하는데 이를
   // 호이스팅(hoisting)이라 한다.
```

2. let

```
let letString; // let 변수 선언 : 블록, 메소드, 구문지역 내에서 유효한 범위를 갖는다.
    // 구문 실행전 먼저 선언되는 hoisting 이 발생하지 않는다.
```

3. const

```
const CONSTSTRING = 1; // const 변수 선언 : let 과 같은 블록, 지역범위를 갖는다.
    // 상수값을 선언한다. 즉 선언 이후 다른 구문에서 재정의 할 수 없다. 보통 대문자로 작성한다.
```

3) 변수 자료형

자바스크립트에서는 변수형에 모든 타입의 자료형 데이터가 들어갈 수 있다. 배열이나 객체 함수 역시 가능하다.

ex)

```
var varString; // 자바스크립트에서는 변수 선언시 자료형을 지정하지않는다.
document.write('1.varString 타입 : ' + typeof (varString) + '- 값은 : ' + varString + '<br>');
varString = "\"홍길동\"";
document.write('2.varString 타입 : ' + typeof (varString) + '- 값은 : ' + varString + '<br>');
varString = 10;
document.write('3.varString 타입 : ' + typeof (varString) + '- 값은 : ' + varString + '<br>');
varString = false;
document.write('4.varString 타입 : ' + typeof (varString) + '- 값은 : ' + varString + '<br>');
varString = { name : '홍길동', age : 20, score : 3.5};
document.write('5.varString 타입 : ' + typeof (varString) + '- 값은 : ' + varString + '<br>');
varString = ["hello", 2, 10.5, false, { 'id': 'abcd', 'pw': '1111' }, [1, 2, 3, 4]];
document.write('6.varString 타입 : ' + typeof (varString) + '- 값은 : ' + varString + '<br>');
varString = function () {alert('함수 속');};
document.write('7.varString 타입 : ' + typeof (varString) + '- 값은 : ' + varString + '<br>');
```

결과)

1.varString 타입 : undefined- 값은 : undefined
2.varString 타입 : string- 값은 : "홍길동"
3.varString 타입 : number- 값은 : 10
4.varString 타입 : boolean- 값은 : false
5.varString 타입 : object- 값은 : [object Object]
6.varString 타입 : object- 값은 : hello,2,10.5,false,[object Object],1,2,3,4
7.varString 타입 : function- 값은 : function () {alert('함수 속');}

4) undefined, NaN, null, infinity

자바스크립트에서는 데이터 타입도 아니고 값도 아닌 4가지의 키워드가 존재한다.

(1) undefined : 정의되지 않은 값, 초기화 되지 않은 값이다.
boolean값으로는 false로 간주되며 Number로 변환하면 NaN으로 간주된다.

(2) Nan : 숫자가 아니라는 뜻의 값이다. 숫자가 와야하는 곳에 숫자가 아닌 다른 값이 오면 NaN으로 반환된다. boolean값으로는 false다. 연산자로는 값을 비교할수 없다는 특징이 있다.
* 해당 값이 NaN인지 구분하는 것은 **isNaN('비교할 값')**이라는 함수를 이용해야한다.

(3) null : 유효한 값이 아니라는 뜻의 값. 값 자체가 없는 게 아니라 null이라는 값이 있다.

(4) infinity : 무한대의 나타낼 수 있는 범위를 넘어서는 수를 나타내는 값이다.

- 키워드의 타입별 값

값	Boolean 문맥	Number 문맥	String 문맥
null	false	0	"null"
undefined	false	NaN	"undefined"
NaN	false	NaN	"NaN"
Infinity	true	Infinity	"Infinity"

5) 입력과 출력

자바스크립트에서 주로 사용하는 데이터 입력 창과 출력 창은 다음과 같다.

(1) **alert(“표시할 데이터”)**

알림 창을 생성하는 함수로 브라우저와는 별도의 새 창을 띄워 사용자에게 데이터를 전달한다
매개변수에 있는 내용을 창에 띄우며 확인버튼을 눌러야 다른 작업을 진행할 수 있다.

(2) **document.write(‘출력 데이터 값’)**

웹 페이지에 작성한 출력 데이터 값을 출력한다.

(3) **console.log(‘출력 데이터 값’)**

웹 페이지가 아닌 관리 도구 페이지의 콘솔 창에 출력 데이터 값이 출력된다. 사용자에게 보이지않는다.

(4) **confirm(‘표시할 데이터’)**

알림창과 비슷하게 브라우저와는 별도의 새 창을 띄워 데이터를 전달하고 확인과 취소 버튼이 있다.
확인을 누르면 true 취소를 누르면 false 값을 반환한다.

(5) **prompt(‘표시할 데이터’, ‘입력란 데이터’)**

표시할 데이터와 함께 새 창을 띄워 사용자에게 전달하며 데이터 입력 칸이 존재한다. 사용자가 입력한 데이터를 문자열 타입으로 반환한다. 생성시 입력란에 기본값을 넣을 수 있으며 생략해도 무방하다.

6) 자료형 변환

자바스크립트에서는 변수에 자료형을 명시하지 않으므로 변수에 특정 데이터를 넣으면 자바스크립트 내에서 자동 형 변환이 일어난다. 그래서 입력 받는 데이터를 원하는 형으로 변환하여 사용하는 경우가 많다.

- 숫자 형을 문자 형으로

(1) **String("변환할 값")** : 매개변수의 값을 명시적으로 문자형으로 변환해 준다.

(2) "문자" + "숫자" : 자바스크립트의 자동 형변환을 이용한것으로 문자와 숫자를 연산하면 문자형으로 형이 변환한다. 단, +를 제외한 연산은 모두 숫자 형으로 변환된다.

- 문자 형을 숫자 형으로

(1) **Number("변환할 값")** : 매개변수의 값을 명시적으로 숫자형으로 변환해 준다. 숫자형이 아닌 값이 들어오면 NaN을 반환한다.

(2) **parseInt("변환할 값")** : 역시 매개변수의 값을 숫자형으로 변환해 준다. 단 숫자가 포함된 문자열에서 숫자로 시작하는 경우에 숫자가 끝나는 지점까지만 형 변환을 하여 반환한다. 시작이 숫자가 아니면 Number()와 마찬가지로 NaN을 반환한다.

ex)

```
str = '100';
i = Number(str);
document.write('<h3>i = Number("100")</h3>');
document.write('i 의 타입은 ' + typeof (i) + ' - 값 :' + i + '<br>');
i = parseInt(str);
document.write('<h3>i = parseInt("100")</h3>');
document.write('i 의 타입은 ' + typeof (i) + ' - 값 :' + i + '<br>');
document.write('<hr>');

str = '100 점';
i = Number(str);
document.write('<h3>i = Number("100 점")</h3>');
document.write('i 의 타입은 ' + typeof (i) + ' - 값 :' + i + '<br>');
i = parseInt(str)
document.write('<h3>i = parseInt("100 점")</h3>');
document.write('i 의 타입은 ' + typeof (i) + ' - 값 :' + i + '<br>');
document.write('<hr>');
```

결과)

i = Number("100")

i의 타입은 number - 값 :100

i = parseInt("100")

i의 타입은 number - 값 :100

---

i = Number("100점")

i의 타입은 number - 값 :NaN

i = parseInt("100점")

i의 타입은 number - 값 :100
