바닐라 자바스크립트(Vanilla JS)란 프레임워크 또는 라이브러리가 적용되지 않은 순수한 자바스크립트를 뜻함

자바스크립트란?

javaScript 는 웹 브라우저에서 사용하기 위하여 만들어진 프로그래밍 언어이고 90년대부터 주로 웹 브라우저 상에서 UI 를 동적으로 보여주기 위하여 사용해 왔으며 기존에는 브라우저에서만 사용해왔던 언어인데, 이제는 단순히 웹페이지에서만 국한되지 않고 Node.js 런타임을 통하여 서버 쪽에서도 사용을 할 수 있게 되었습니다

자바스크립트의 역사

자바스크립트는 1995년에 넷스케이프(Netscape)의 브렌던 아이크(Brendan Eich)에 의해 만들어졌습니다.
처음에는 모카(Mocha)라는 이름으로 개발되었으나, 그 후에 라이브스크립트(LiveScript), 최종적으로는 자바스크립트(JavaScript)라는 이름으로 변경됩니다.

자바스크립트의 특징

1. 자바스크립트는 객체 기반의 스크립트 언어입니다.
2. 자바스크립트는 동적이며, 타입을 명시할 필요가 없는 인터프리터 언어입니다.
3. 자바스크립트는 객체 지향형 프로그래밍과 함수형 프로그래밍을 모두 표현할 수 있습니다.
4. 절차적 언어
5. 태그로 DOM요소를 검색할 시, 제이쿼리보다 425배 빠르다고 함
6. 프론트에서의 리액트, 뷰, 엥귤러가 자바스크립트의 es6의 문법을 사용하면서 자바스크립트에 대한 중요성이 높아짐

자바스크립트 표준

1996년에 넷스케이프(Netscape)는 자바스크립트를 국제 표준안으로 만들기 위해 ECMA(European Computer Manufacturers Association)에 제출합니다.
그 결과 ECMA는 ECMAScript라는 새로운 표준을 제정하였고, 그 첫 번째 버전인 ECMA-262를 1997년에 공표합니다.
ECMAScript는 자바스크립트뿐만 아니라 마이크로소프트의 JScript나 어도비의 액션스크립트도 따르는 국제 표준이 됩니다.
현재 자바스크립트의 최신 표준은 2015년에 발표된 ECMAScript 6입니다.

구조 - HTML / 디자인 - CSS
행동 - javascript & jQuery 화면단 바닐라 자바스크립트(Vanilla JS)
화면단 : 1. 메뉴 / 2. 비주얼 배너 / 3. 마우스 휠 (원페이지) / 4. 모바일 메뉴 / 5. 쇼핑몰 배너 / 6. 전체메뉴 / 7. 모달 창 , 팝업 배너 광고 등
코딩 프로그램 : 비주얼스튜디오코드 ,이클립스, 코모도, 웹스톰, 아톰 등

	형식	설명
선언	자바스크립트 작성법 1. head문 , body문 어디든 사용가능 2. 주로 body문 맨 끝에 사용(처리속도) 자바스크립트 문법 주의사항 1.대소문자 구분 2.실행문 종료시 ; 을 사용 3.한줄에 한문장만 작성하는 것이 가독성이 좋음 4.{},()짝이 맞아야 한다 5.“ ”,‘ ’ 따옴표 겹침주의 6.주로 작은따옴표안에 작성 7.외부 스크립트 선언 (분리) <script src="경로"></script> 8.주석(설명) : // 한줄 주석 , /* 여러줄 주석 */	<script> 자바스크립트 실행문 ; // 한줄 주석문 /* 여러줄 주석문 - 설명 */ </script> - 외부 스크립트 선언 <script src="경로"></script> <script src="js/main.js"></script> - main.js 스크립트 내용 작성
작성법	document.write(“ ”또는‘ ’),document.write(“‘ ’” 또는 ‘“ ”’) 1. 문자 또는 숫자를 출력 2. 태그를 출력 (화면에 태그인식) alert() 경고메세지 출력 , 개발자(퍼블리셔)가 값체크 할 때 console.log() 1. F12 - console 부분에 나타남 2. console 부분에 에러메세지 출력 , 중간값 체크	<script> document.write(‘문자열 출력’);/document.write(‘태그출력 ’); document.write(‘<img src="이미지주소" alt="대체텍스트">’); window.alert('경고 메시지‘); / alert('경고 메서지 출력‘); console.log(‘메세지’); </script> <script src="js/main.js"></script> js폴더안에 main.js document.write(‘문자열 출력’); document.write(‘태그출력 ’);
변수	변수란 - 변하는 데이터값을 저장할수 있는 메모리 공간 (그릇)	let 선언 (variable) let 변수명=“사용할 문자나 숫자” let 변수명=숫자; 또는Number("숫자");

	- 주고 받기를 하는 과정에서 사용되는 데이터를 일시적으로 보존해주는 그릇 - 변수에는 문자형,숫자형,논리형(true/false)를 저장할수 있다 - 특정 이름에 특정 값을 담을 때 사용 종류 var (variable) : 추천하지 않는다, 예전방식 let : 재할당가능 유효범위를 관리할수 있다 const : 재할당 불가 # 스코프 (Scope) 란? (유효성 범위) Scope란 말그대로 코드의 영역이라는 뜻 { } Function 단위로 scope 가 이루어짐 변수 선언규칙 (식별자) 1. 첫글자는 숫자사용불가, $, _, 영문자만 올 수 있다 2. 영문자, 숫자, _, $ 혼용해서 사용가능 3. 의미를 부여해서 지정한다 4. 대소문자를 구분한다 5. 한글 예약어 , 특수문자는 사용할수 없다 6. 의미에 맞는 영문으로 사용한다 단어를 조합할 때 규칙 - 스네이크 표기법(Snake case) : 소문자만 사용하고 각각의 사이를 언더바(_)를 넣어서 color_top , cat_dog - 카멜 표기법(Camel case) : colorTop - 두번째 단어의 첫 글자를 대문자로 사용 catDog - 파스칼 표기법(Pascal case) : 카멜과 비슷하지만 첫글자도 대문자로시작 CatDog	let 변수명=true or false; 또는 Boolean(데이터); # 리터널 변수나 상수에 저장되는 값 자체를 말한다 코드상에서 데이터를 표현하는 방식 변수리터널 , 상수리터널 , 배열리터널 키워드 : 자바스크립트에서 사용하는 단어 식별자 : 사용자가 임의로 사용하는 단어 변수에 저장할 수 있는 데이터 타입 - 문자형(String) , 숫자형(Number), 논리형(Boolean) , 빈(Null) 변수로 사용할수 없는 이름 - 예약어 (키워드) break, case, catch, continue, default, delete, do, else, finally, for, function, if, instanceof, new, return, switch, this, throw, try, typeof, var, void, while, with etc 변수 이름을 지을 때 사용하는 일반적인 규칙 카멜표기법이나 언어스코어 표기법을 주로 사용 상수는 대문자로 표현 논리값은 변수 이름 앞에 is를 사용 (상태변수) 루프 카운터는 i, j, k를 사용 (반복문) 카멜 : 변수 / 스네이크: 상수명 / 파스칼 : 클래스(생성자)명 개발에서 비공개임을 표시할 때 : _변수명을 사용한다 이스케이프 \' 따옴표 , \" 쌍따옴표, \\ 역슬래시 , \b 백스페이스 , \r 캐리지 리턴 , \n 줄바꿈 , \s 스페이스 , \t 수평탭 ,\v 수직탭 ,\f 폼피드,\0 null

변수의 유형	데이터형
	기본형 : 값을 변수에 대입하는 방법
	숫자(Number): 숫자에는 정수와 실수
	문자열(String) : 문자열은 문자
	논리값(Boolean) : 논리값은 참과 거짓
	특수값(null) : 비어있다
	특수값(undefined) : 정의되지 않았다 (미지정)
	자료형(typeof) : 변수에 저장된 자료형을 알아내기
	
	참조형 : 참조값(값을 실제로보관하고 있는 메모리)를 보관
	
	- 배열(Array) : 데이터 집합
	- 객체(Object) : 데이터 + 함수 집합
	- 함수(function) : 구문의 집합
	
	추가 : 변수에는 숫자,문자열,함수,클래스,클래스의 인스턴스도 넣을수있다
	
	리터널
	let num = 10 => num은 변수이고 10은 숫자 리터널
	변수나 상수에 저장되는 값 자체를 말한다
	코드상에서 데이터를 표현하는 방식
	숫자리터널 / 문자열 리터널 / 템플릿 리터널 / 배열리터널 / 객체리터널 /함수리터널
		let str ='javascript'
		let a = '200'
		let tag =' 나는제목 '
		let num = Number('200') ==> 200
		let t1 = true
		let t2 = 10 >= 100 ==> false
		let k = Boolean('hello') ==> true
		let s; ==> undefined
		let t = null
		
		console.log(typeof str)
		타입 확인
		typeof str ==> string
		typeof num ==> number
		
		>배열
		let array = [273, 'String', true, function () { }, {}, [273, 103]];
		alert(array);
		
		>객체
		let book ={title:'javascript',publish:'기술'}
		alert(book.title)

연산자

산술 연산자	설명
+	왼쪽 피연산자의 값에 오른쪽 피연산자의 값을 더함.
-	왼쪽 피연산자의 값에서 오른쪽 피연산자의 값을 뺌.
*	왼쪽 피연산자의 값에 오른쪽 피연산자의 값을 곱함.
/	왼쪽 피연산자의 값을 오른쪽 피연산자의 값으로 나눔.
%	왼쪽 피연산자의 값을 오른쪽 피연산자의 값으로 나눈 후, 그 나머지를 반환함.

대입 연산자	설명
=	왼쪽 피연산자에 오른쪽 피연산자의 값을 대입함.
+=	왼쪽 피연산자의 값에 오른쪽 피연산자의 값을 더한 후, 그 결괏값을 왼쪽 피연산자에 대입함.
-=	왼쪽 피연산자의 값에서 오른쪽 피연산자의 값을 뺀 후, 그 결괏값을 왼쪽 피연산자에 대입함.
*=	왼쪽 피연산자의 값에 오른쪽 피연산자의 값을 곱한 후, 그 결괏값을 왼쪽 피연산자에 대입함.
/=	왼쪽 피연산자의 값을 오른쪽 피연산자의 값으로 나눈 후, 그 결괏값을 왼쪽 피연산자에 대입함.
%=	왼쪽 피연산자의 값을 오른쪽 피연산자의 값으로 나눈 후, 그 나머지를 왼쪽 피연산자에 대입함.

비교 연산자	설명
==	왼쪽 피연산자와 오른쪽 피연산자의 값이 같으면 참을 반환함.
===	왼쪽 피연산자와 오른쪽 피연산자의 값이 같고, 같은 타입이면 참을 반환함.
!=	왼쪽 피연산자와 오른쪽 피연산자의 값이 같지 않으면 참을 반환함.
!==	왼쪽 피연산자와 오른쪽 피연산자의 값이 같지 않거나, 타입이 다르면 참을 반환함.
>	왼쪽 피연산자의 값이 오른쪽 피연산자의 값보다 크면 참을 반환함.
>=	왼쪽 피연산자의 값이 오른쪽 피연산자의 값보다 크거나 같으면 참을 반환함.
<	왼쪽 피연산자의 값이 오른쪽 피연산자의 값보다 작으면 참을 반환함.
<=	왼쪽 피연산자의 값이 오른쪽 피연산자의 값보다 작거나 같으면 참을 반환함.

논리 연산자	설명
&&	논리식이 모두 참이면 참을 반환함. (논리 AND 연산)
||	논리식 중에서 하나라도 참이면 참을 반환함. (논리 OR 연산)
!	논리식의 결과가 참이면 거짓을, 거짓이면 참을 반환함. (논리 NOT 연산)

A	B	A && B	A || B	!A
true	true	true	true	false
true	false	false	true	false
false	true	false	true	true
false	false	false	false	true

증감 연산자	설명
++x	먼저 피연산자의 값을 1 증가시킨 후에 해당 연산을 진행함.
x++	먼저 해당 연산을 수행하고 나서, 피연산자의 값을 1 증가시킴.
--x	먼저 피연산자의 값을 1 감소시킨 후에 해당 연산을 진행함.
x--	먼저 해당 연산을 수행하고 나서, 피연산자의 값을 1 감소시킴.

삼항연산자
조건식 ? true실행문 : false실행문

```
let x = 10, y = 20 , z ;
z = ( x > y ) ? 'x값이크다' : 'y값이 크다'
```