기본용어, 출력

2022년 10월 4일 화요일 오후 7:07

[키워드]

키워드는 제어문의 시작과 끝, 특정한 조작 목적 등으로 쓰임
이런 기능들이 정의되어 있기 때문에 식별자나 프로퍼티 이름으로 사용 할 수 없다.

Break	Else	Instanceof	true
Case	False	New	Try
Catch	Finally	Null	typeof
Continue	For	Return	Var
Default	Function	Switch	Void
Delete	If	This	While
Do	In	Throw	With
Let	const	debugger	

[식별자]

이름을 붙일 때 사용하는 단어
변수와 함 수 이름 등으로 사용, 규칙에 맞춰서 만들어야 한다

- 키워드를 사용하지 않음
- 특수 문자는 _와 $만 허용
- 숫자로 시작하지 않음
- 공백 입력하지 않는다
- 클래스의 이름은 항상 대문자로 시작
- 변수, 함수, 속성, 메소드의 이름은 항상 소문자
- 여러 단어로 된 식별자는 각 단어의 첫글자를 대문자로 함
  Will out -> willOut

식별자의 종류

구분	단독으로 사용	다른 식별자와 사용
식별자 뒤에 괄호 없음	변수 또는 상수	속성
식별자 뒤에 괄호 있음	함수	메소드

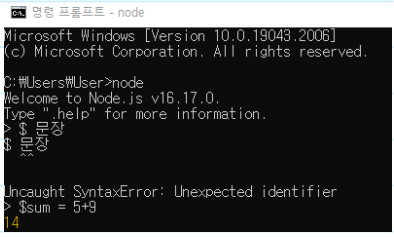
- Alert('Hello World') : 함수
- Array.length : 속성
- Input : 변수 또는 상수
- Prompt('Message', 'Defstr') : 함수
- Math.PI : 속성
- Math.abs(-237) :메소드

[출력 메소드]

Console.log("문자열")

[REPL을 이용한 출력]

Node로 REPL환경 바로 접속

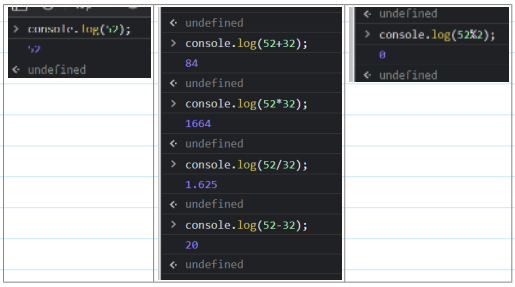


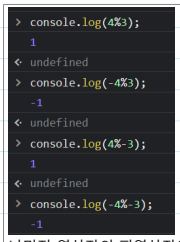
화면 캡처: 2022-10-04 오후 7:24

기본자료형

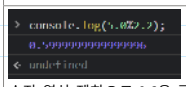
2022년 10월 4일 화요일 오후 7:21

[숫자]





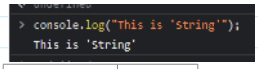
나머지 연산자의 피연산자로 음수가 올 때 프로그래밍 언어마다 처리방식 다름
모두 양수로 변화해서 사용하는 것이 좋음



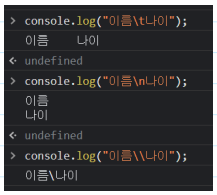
숫자 연산 제한으로 0.6을 구하지 못함
프로그래밍언어 문제
소수점이 있는 숫자에 나머지 연산자 적용 ㄴ

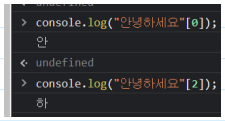
[문자열]

1) 기본문자열 : ' ' / " " 둘다 사용



이스케이프 문자	설명
\t	수평 탭
\n	줄바꿈
\'	작은따옴표
\"	큰따옴표
\\	역슬래시

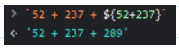




문자열 선택 가능

2) 템플릿 문자열

같은 취급 받음
템플릿 문자열은 생성할 때 내부에 ${표현식}을 사용 할 수 있다.
내부에 ${표현식}을 입력하면 표현식이 계산되어 문자열에 들어간다.



3) 비교연산자

연산자	설명
==	같다
!=	다르다
>	왼쪽이 크다
<	오른쪽이 크다
>=	왼쪽이 크거나 같다
<=	오른쪽이 크거나 같다

4) 논리 연산자

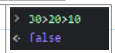
연산자	설명
!	논리 부정 연산자
||	논리합 연산자
&&	논리곱 연산자

4-1) 논리합 연산자 (|| / OR 둘중 하나가 TRUE면 TRUE)

왼쪽	오른쪽	결과
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

4-2) 논리곱 연산자(&& 2개 모두 TRUE여야 TRUE)

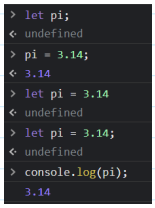
왼쪽	오른쪽	결과
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

** 주의 30 > 20 > 10 (30 > 20) > 10 True > 10 1> 10 false	

[변수 선언]

Let 식별자 ;

:: 변수에 값을 할당한다라고 표현



화면 캡처: 2022-10-04 오후 7:50

연산자

2022년 10월 4일 화요일 오후 7:49

[복합 대입 연산자]

A += 10
=> a= a + 10

연산자	설명
+=	숫자 덧셈 후 대입 연산자
	문자열 연결 후 대입 연산자
-+	숫자 뺄셈 후 대입 연산자
*=	숫자 곱셈 후 대입 연산
/+	숫자 나눗셈 후 대입 연산

```
undefined
> let output = "hello ";
undefined
> output += "world";
'hello world'
> output += "!";
'hello world!'
> console.log(output);
hello world!
```

화면 캡처: 2022-10-04 오후 7:54

[증감 연산자]

연산자	설명
변수++	기존값에 1을 더한다 (후위)
++변수	기존 값에 1을 더한다(전위)
번수--	기존값에 1뺀다 (후위)
--변수	기존값에 1뺀다 (전위)

[일치 연산자]

연산자	설명
===	자료형과 값이 같은지 비교
!==	자료형과 값이 다른지 비교

예 :

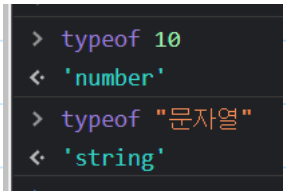
0 과 ""(빈문열)을 비교하면 둘다 불로 변환하면 false 가, 되니까 Ture라는 결과가 나옴

자료형

2022년 10월 4일 화요일 오후 7:57

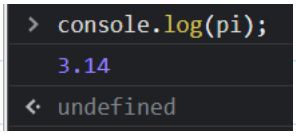
[자료형 검사]
자바스크립트는 하나의 변수에 여러 자료형 넣을 수 있다.
원치 않은 자료형이 변수에 들어갈 수 있기 때문에 자료형 검사해야한다.

Typeof : 해상 변수의 자료형을 추출



화면 캡처: 2022-10-04 오후 7:58

[undefined 자료형]
초기화되지 않은 것 혹은 선언하지 않은 것을 나타냄



[강제 자료형 변환]

함수	설명
Number()	숫자로 자료형 변환
String()	문자열로 자료형 변환
Boolean()	불로 자료형 변환

1. Number()

```
> console.log(Number("52"));
  52
< undefined
> console.log(Number(true));
  1
< undefined
> console.log(Number("dkssud");
✖ Uncaught SyntaxError: missing ) after argument list
> console.log(Number("안녕");
✖ Uncaught SyntaxError: missing ) after argument list
```

문자열은 숫자자료형이 될 수 없어 NaN으로 나타냄

NaN인지 확인하려면 isNaN()활용 할 수있음

```
> let nan = Number("안녕");
< undefined
> console.log(nan == nan)
  false
< undefined
> console.log(nan != nan)
  true
< undefined
> console.log(isNaN(nan));
  true
```

2. Boolean()함수

- False
  - 0
  - NaN
  - "" (빈문자열)
  - Null
  - Undefined
- 위 5가지 외에는 전부 ture

[자동 자료형 변환]
+ 를 제외한 연산자를 적용하면
문자열이 숫자로 변환된 뒤에 숫자 연산이 이루어진다

! 연산자를 사용할 때는 불 자료형으로 자동변환된다

상수

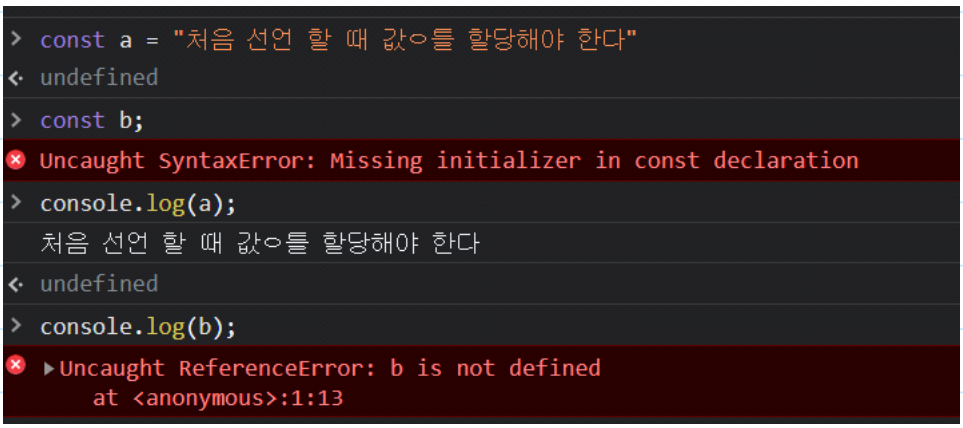
2022년 10월 4일 화요일 오후 8:10

[상수]

고정 된 숫자와 같은 변경하지 않을 대상 예를 들어 pi 3.14149...

Const 이름 = 값;

프로그램을 더 빠르고 안정적으로 만들기 위해 사용
자바스크립트가 내부적으로 간단한 형태로 저장한다.



화면 캡처: 2022-10-04 오후 8:16

일단 상수(const)를 사용하고 오류가 발생하면 변수(let)을 교환한다.