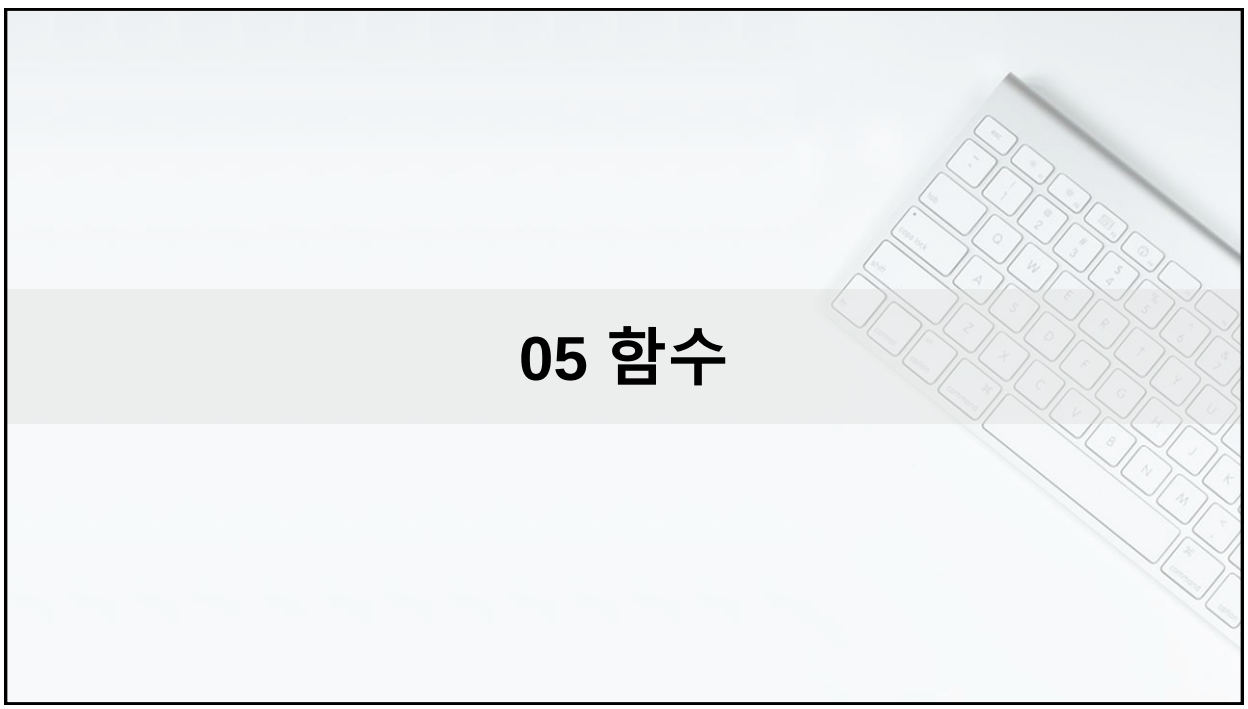
JAVASCRIPT

Creative by GT



05 함수

학습목표

✓함수의 기본 형태를 만들 수 있다.

✓매개변수와 리턴값을 이해하고 사용할 수 있다.

✓콜백 함수를 이해하고 사용할 수 있다.

CONTENTS

+ 함수 생성 방법
+ 함수의 기본 형태
+ 매개 변수
+ 콜백 함수
+ 표준 내장 함수

선언적 함수

- 일반적인 함수의 형식
  - '선언적 함수'라 표현

```
function 함수명 ( ) { }
```

  - 같은 기능을 수행하는 함수

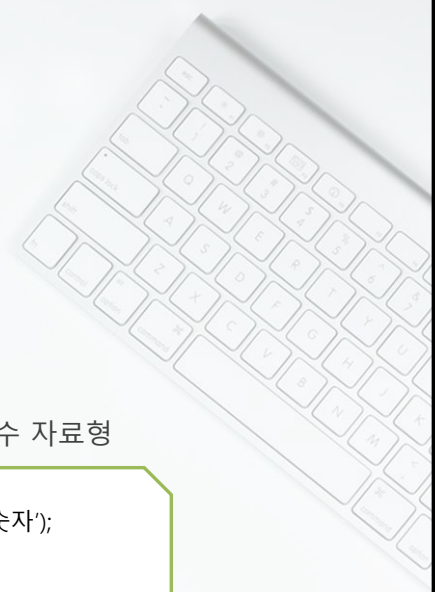
```
var 함수 = function ( ) { };
```

  - 익명 함수와 마찬가지의 방법으로 만들고 사용

함수

- 함수function
  - 코드의 집합
  - [예] alert(), prompt() ...
- 함수의 형태
  - var 함수 = function ( ) { };
    - 괄호 내부에 코드를 넣음
  - [예] 두 문장을 포함하는 함수를 생성하고 출력
    - 문자열처럼 보일 수 있지만 typeof 연산자를 사용하면 함수 자료형

```
var 함수 = function() {
        var output = prompt('숫자 입력 : ', '숫자');
        alert(output);
};
alert(함수);
```



함수 개요

- 익명 함수
  - 이름을 가지고 있지 않은 함수
  - 이름이 없으므로 **변수에 넣어 사용**해야 함

```
function ( ) { }
```

- 내장 함수의 출력
  - 모든 브라우저는 내장하고 있는 함수의 소스를 볼 수 없게 막아놓음
  - '선언적 함수'
    - 이름을 가지고 있는 함수
- 함수 호출
  - 함수의 실행을 함수 호출이라 함
  - 함수는 자료형이지만 뒤에 괄호를 열고 닫음으로써 코드 실행

NOTE

Arrow Function (ECMAScript6)

- **'하나의 표현식을 리턴하는 함수' 생성시 중괄호({ }) 생략 가능**
- **기본형**

```
( ) => { }
```

```
let 함수 = ( ) => {
        let output = prompt('숫자 입력 : ', '숫자');
        alert(output);
};
alert(함수);
```

매개 변수와 리턴값

- 매개 변수Parameter
  - 함수를 호출할 때 괄호 안에 적는 것
- 리턴 값
  - 함수를 호출하고 함수가 변환되는 값
    - prompt() 함수를 사용하면 사용자가 입력한 문자열로 변환

```
function 함수이름(매개 변수, 매개 변수, 매개 변수) {
        // 함수 코드
        return 리턴 값;
}
```

매개 변수

- 매개 변수Parameter
  - 함수를 호출하는 쪽과 호출된 함수를 연결하는 매개가 되는 변수
  - 자바스크립트는 함수를 생성할 때 지정한 매개 변수보다 많거나 적은 매개 변수의 사용이 허용됨
    - 원래 함수에서 선언된 매개 변수보다 많게 사용하면
      - » 추가된 매개 변수는 무시
    - 원래 함수에서 선언된 매개 변수보다 적게 사용하면
      - » 지정하지 않는 매개 변수는 undefined

가변 인자 함수

- 가변 인자 함수란?
  - 매개 변수의 개수가 변할 수 있는 함수
  - 좁은 의미의 가변 인자 함수
    - 매개 변수의 선언된 형태와 다르게 사용했을 때에도 매개 변수를 모두 활용하는 함수
    - Array() 함수의 매개 변수에 따른 차이

함수 형태	설명
Array()	빈 배열 생성
Array(number)	Number 크기를 갖는 배열 생성
Array(num1, num2, ...)	매개 변수를 배열로 생성

가변 인자 함수

- sumAll() 함수
  - 매개 변수로 입력된 숫자를 모두 더하는 함수
  - 자바스크립트의 모든 함수는 내부에 자동으로 변수 arguments 를 포함
  - arguments 객체의 자료형과 배열의 길이 출력
    - 함수를 호출할 때 아홉 개의 매개 변수 입력
      - » arguments 배열의 length 속성은 9

```
function sumAll() {
    alert(typeof (arguments) + ': ' + arguments.length);
}
// 함수를 호출합니다.
sumAll(1, 2, 3, 4, 5, 6, 7, 8, 9);
```

가변 인자 함수

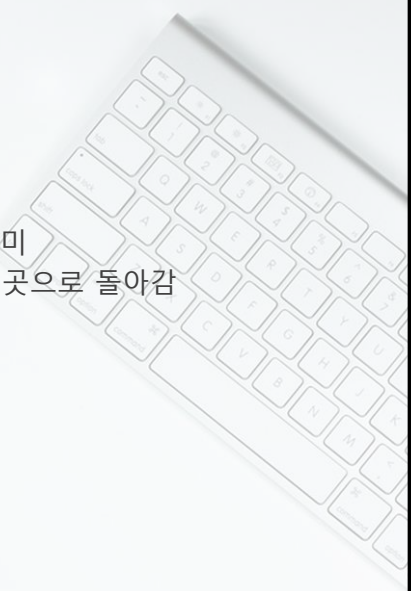
- 함수의 매개 변수 숫자가 다를 때 처리하는 예
  - 배열 arguments의 요소 개수에 따라 조건 설정

```
function 함수( ) {
        // 매개 변수의 개수를 구함
        var length = arguments.length;

                // 조건
                if (length = 0) {
                        // 매개 변수가 없을 때
                } else if (length = 1) {
                        // 매개 변수가 한 개일 때
                } else {
                        // 매개 변수가 두 개일 때
                }
}
```

리턴 값

- 리턴 값 활용
  - return 키워드를 사용해 함수를 호출한 곳으로 값 넘김
  - return 키워드의 의미
    - 함수가 실행되는 도중 함수를 호출한 곳으로 돌아가라는 의미
    - return 키워드 사용시 값을 지정하지 않아도 함수를 호출한 곳으로 돌아감



내부 함수

- 내부 함수란?
  - 함수 내부에 선언한 함수

```
function 외부 함수( ) {
        function 내부 함수1( ) {
                // 함수 코드
        }
        function 내부 함수2( ) {
                // 함수 코드
        }

        //함수 코드
}
```

내부 함수

- 함수 이름이 충돌하는 경우
  - 내부 함수를 사용하는 경우
    - 외부에 이름이 같은 함수가 있어도 내부 함수 우선
  - 내부 함수는 내부 함수가 포함되는 함수에서만 사용 가능
    - 내부 함수는 외부에서 사용 불가

콜백 함수

- 함수의 매개 변수로 전달되는 함수

```
function theCall ( callback ) {
        for ( let i = 0; i < 5; i++ ) {
                // 매개변수로 전달된 함수 호출
                callback();
        }
}

theCall( function () {
        document.write('콜백함수 호출');
} );
```

함수를 리턴하는 함수와 클로저

- 클로저closure
  - 지역 변수를 남겨두는 현상
  - 함수 outerFunction()로 인해 생성된 공간
    - 함수 outerFunction() 내부의 변수들이 살아있음
  - 리턴되는 함수 자체
  - 살아 남은 지역 변수 output

- 클로저의 사용
  - 리턴 된 클로저 함수를 사용해서만 지역 변수 output 사용 가능
  - 클로저 함수로 인해 남는 지역 변수는 각각의 클로저의 고유한 변수

자바스크립트 내장 함수 (1)

- 내장 함수
  - 자바스크립트에서 자체 제공하는 기본 내장 함수
    - 인코딩 – 문자를 저장하거나 통신에 사용할 목적으로 부호화
    - 디코딩 - 부호화된 문자를 원래대로 되돌리는 것
    - 자바스크립트의 인코딩/디코딩 관련 내장 함수

함수 이름	설명
escape()	적절하게 인코딩
unescape()	적절하게 디코딩
encodeURI(uri)	최소한의 문자만 인코딩
decodeURI(encodedURI)	최소한의 문자만 디코딩
encodeURIComponent(uriComponent)	대부분의 문자를 인코딩
decodeURIComponent(encodedURI)	대부분의 문자를 디코딩

자바스크립트 내장 함수 (1)

- 내장 함수 간 비교
  - escape()
    - 영문 알파벳, 숫자, 일부 특수 문자(@, *, -, _, +, ., /)를 제외한 모든 문자
    - 1바이트 문자는 **%**XX의 형태로, 2바이트 문자는 **%u**XXXX의 형태로 변환
  - encodeURI()
    - escape() 함수에서 인터넷 주소에 사용되는 일부 특수 문자(:, ;, /, =, ?, &)는 변환하지 않음
  - encodeURIComponent()
    - 알파벳과 숫자를 제외한 모든 문자 인코딩
    - UTF-8 인코딩과 같음

자바스크립트 내장 함수 (2)

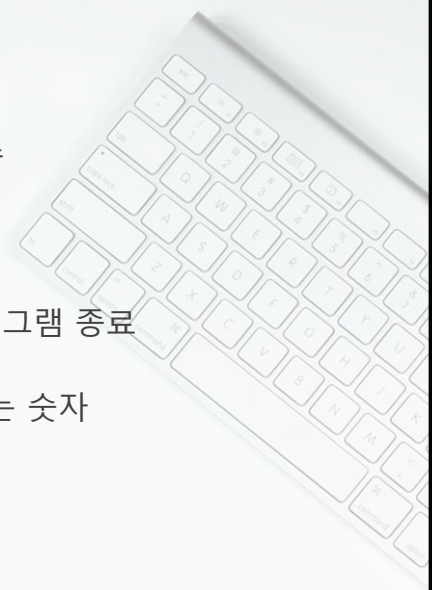
- 자바스크립트 기본 내장 함수

함수 이름	설명
eval(string)	string을 자바스크립트 코드로 실행
isFinite(number)	number가 유한한 값인지 확인
isNaN(number)	number가 NaN인지 확인
parseInt(string)	string 중 숫자 부분만 정수로 변환
parseFloat(string)	string 중 숫자 부분만 실수로 변환

자바스크립트 내장 함수 (2)

- eval() 함수
  - 문자열을 자바스크립트 코드로 변환해 실행하는 함수

- isFinite() 함수 와 isNaN() 함수
  - 0으로 숫자를 나누면 자동으로 오류가 발생하며 프로그램 종료
    - 자바스크립트는 0으로 숫자를 나누면 infinity
  - NaN(Not a Number)- 자바스크립트가 표현할 수 없는 숫자



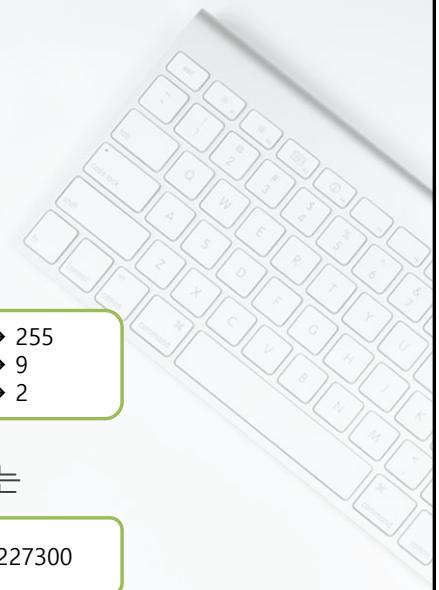
자바스크립트 내장 함수 (2)

- isFinite() 함수
  - isFinite(유한한 수)면 true 리턴
- isNaN() 함수
  - isNaN이면 true 리턴
  - 자바스크립트에는 Infinity, NaN이 변수로 존재
- **number1** == **Infinity** ?
  - 무한대의 수인지 구분할 때는 꼭 isFinite() 함수 사용
  - 음수를 0으로 나누면 –Infinity가 되므로 비교 불가

자바스크립트 내장 함수 (2)

- **NaN**의 비교
  - NaN은 스스로를 비교할 수 없어 불가
  - 자바스크립트는 NaN == NaN을 거짓으로 인식
    - NaN을 확인할 때에는 무조건 isNaN() 함수 사용
- parseInt() 함수와 parseFloat() 함수
  - 두 함수 모두 문자열을 숫자로 변경하는 함수
  - [참고] Number() 함수
    - 숫자로 바꿀 수 없으면 무조건 NaN 리턴

자바스크립트 내장 함수 (2)



- parseInt(), parseFloat() 함수 사용시 주의점
  - 0으로 시작하면 8진수, 0x로 시작하면 16진수로 인식
    - 10진수로 자동 변환
  - parseInt() 함수의 두 번째 매개 변수에 진법 입력
    - 앞의 수를 해당 진법의 수로 인식하고 10진수로 출력

```
parseInt('273')      → 273
parseInt('0273')     → 187
parseInt('0x273')    → 627
```

```
parseInt('FF', 16)   → 255
parseInt('11', 8)    → 9
parseInt('10', 2)    → 2
```

  - parseFloat() 함수 - 중간에 e가 들어가면 e뒤의 숫자는 10의 지수로 인식

```
parseFloat('52.273e5')      → 5227300
```