웹 앱 개발을 위한 JavaScript 강의 노트

제 6회차

함수 생성과 사용

한국기술교육대학교 KOREA UNIVERSITY OF TECHNOLOGY & EDUCATION

▣ 학습목표

- 함수의 매개 변수와 리턴값에 대해 설명할 수 있다.
- 익명 함수와 선언적 함수를 선언하고 활용할 수 있다.
- 가변 인자 함수, 내부 함수, 자기 호출 함수, 콜백 함수, 클로저를 이해하고 사용할 수 있다.

▣ 학습내용

- 함수의 생성
- 다양한 함수의 형태

1. 함수의 생성

1) 함수란

- 함수

- 특정한 기능을 제공하기 위한 코드의 집합으로써, 호출을 통해 결과값을 얻게 됨
- 객체 자료형 중 하나
- 함수도 하나의 값처럼 취급되기 때문에, 변수에 할당 가능
- 함수 생성시 함수명은 선택사항으로, 함수명 없이 함수를 생성 할 수 있음

2) 익명함수

- 익명함수

- function() { }와 같이 함수의 형태이지만, 이름이 없는 함수
- 이름이 없기 때문에 변수에 넣어 사용
- 중괄호({ }) 안에 함수가 처리할 코드 나열
- 형태

  → 생성된 함수를 변수에 할당하여 함수를 생성

```
var 변수명 = function( ) { };
```

- 예

```
<script>
    var plusTen = function () {
                    var num1 = prompt('숫자1를 입력해 주세요', '');
                        alert(Number(num1) + 10);
                    };
    alert(typeof (plusTen));
    alert(plusTen);
    plusTen();

    var add = plusTen;
    add();

  </script>
```

1. 함수의 생성

2) 익명함수

- 익명함수

- 함수를 변수 plusTen에 할당 후 호출하기

```
<script>
      var plusTen = function () {
                        var num1 = prompt('숫자1를 입력해 주세요', '');
                              alert(Number(num1) + 10);
                          };
      alert(typeof (plusTen));
      alert(plusTen);
      plusTen();

      ① var add = plusTen;
      ② add();

   </script>
```

① 함수가 저장된 변수 plusTen을 add 변수에 할당

② plusTen = add( )

3) 선언적 함수

- 선언적 함수

- 일반적으로 함수를 만드는 방식으로, 함수명이 있음
- 익명 함수와 같이 중괄호({}) 안에 함수가 처리해야 할 코드를 기술
- 형태

```
function 함수명( ) { };
```

- 예 (10을 출력하는 기능을 지닌 선언적 함수 생성)

```
<script>
      function printTen() {
         alert(10);
      }
      printTen();
 </script>
```

1. 함수의 생성

3) 선언적 함수

- 함수의 재정의

- 이미 존재하는 함수명과 동일한 함수명을 가진 함수 정의
- 함수 호출 시 마지막으로 정의된 함수가 호출됨

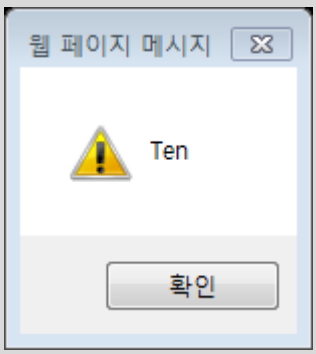
```
<script>
    function printTen() {
        alert(10);
    }

    function printTen() {
        alert('Ten');
    }

    ① printTen();
</script>
```

① 최종적으로 할당된 함수 실행

- 결과



1. 함수의 생성

4) 매개변수와 리턴값

- 매개변수

  - 함수를 호출하면서 함수의 입력으로 넘겨주는 것
  - 예 : 자판기의 동전과 버튼 누름

```
var output = prompt('input string', 'default string');
```

- 리턴 값

  - 호출한 함수가 돌려주는 값
  - 예 : 해당 음료수

```
var output = prompt('input string', 'default string');
```

- 항상 사용해야 하는 것은 아니며, 필요에 따른 선택사항
- 형식

```
function 함수명(매개변수, 매개변수) {
        //함수 코드
    return 리턴값;
}
```

- 예

```
<script>
    function printTen( x ) {
        return x+10;
    }

    alert(printTen( 3 ));
</script>
```

2. 다양한 함수의 형태

1) 가변 인자 함수

- 매개변수가 함수에서 정의된 것보다 많이 입력된 것만 사용
- 입력된 매개변수를 활용할 수 있도록 함수 구현 가능
- 매개변수의 개수가 변할 수 있는 함수
- 정의된 것과 달리 입력된 매개 변수를 모두 활용할 수 있는 함수
- 모든 함수는 매개 변수를 arguments 배열에 저장함
- Argument 매개변수의 저장

```
<script>
    function add() {
        var output = '';
        alert(typeof(arguments));
        alert(arguments.length);

        for (var i = 0; i < arguments.length; i++) {
            output += arguments[i] +'   ';
        }
        return output;
    }

    var out = add(1, 2, 3, 4, 5);
    alert(out);
</script>
```

2. 다양한 함수의 형태

2) 내부 함수

- 함수 코드 내부에 정의된 함수
- 형식

```
function 외부함수( ){
   function 내부함수1( ){
                  //실행할 문장
   }

   function 내부함수2( ){
                  //실행할 문장
   }

  // 실행할 문장
}
```

- 외부 함수 내에서 내부 함수가 다른 외부 함수보다 우선(변수와 동일)
- 내부 함수를 포함하는 외부 함수에서만 사용 가능
- 외부 함수의 변수에 접근이 가능

3) 자기 호출 함수

- 자기 호출 함수

- 생성과 동시에 실행되는 함수
- 함수를 다시 호출 할 수 없음
- 최초 한 번의 실행만을 필요로 하는 초기화 코드 부분 등에 사용
- 형식

```
(function ( ) {
     //실행할 코드
}) ( );
```

2. 다양한 함수의 형태

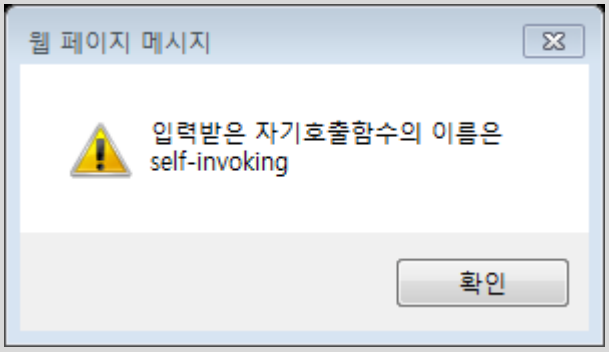
3) 자기 호출 함수

- 예

```
<script>
      (function (name) {
         ① alert('입력받은 자기호출함수의 이름은 ₩n' + name);
      })('self-invoking');
  </script>
```

① 함수가 저장된 변수 plusTen을 add 변수에 할당

- 결과



2. 다양한 함수의 형태

4) 콜백 함수

- 함수는 객체 자료형 중 하나이기 때문에 매개변수로 사용 가능
- 함수를 매개변수로 전달하는 함수

• 결과

```
<script>
    function check(message) {
        var i = 0;
        var sum = 0;

        while (1) {
            i++;
            sum += i;
            if (sum > 20) {
                message();
                break;
            }
        }
    }

    var message = function () {
        alert('합이 20을 넘었습니다.');
    }

     ① check(message);
</script>
```

① 함수를 매개 변수로 이용하여 check( ) 함수를 호출

2. 다양한 함수의 형태

5) 함수를 리턴하는 함수

- 함수를 매개 변수로 사용할 수 있다는 것 - 함수를 리턴값으로 사용할 수 있다는 것 의미

```
<script>
    function returnFunction() {
        return function ( ) {
                alert('return');
               };
    }

    returnFunction ① ( )( );

  </script>
```

① 호출 시 return을 출력하는 함수를 return으로 받기 때문에 괄호를 한 번 더 적음

- innerFunction()호출 시 내부 함수는 내부 함수를 포함하는 외부 함수에서만 사용할 수 있어 오류 발생

```
<script>
    function outerFunction() {
        var a = 10;
        var b = 20;

        function innerFunction() {
            var b = 30;

            alert(a);
            alert(b);
        }

        innerFunction();
        alert('outerFunction');
    }

    outerFunction();
    innerFunction();
  </script>
```

2. 다양한 함수의 형태

5) 함수를 리턴하는 함수

- 클로저(Closure)

- 종료된 외부 함수의 변수를 참조하는 함수
- 지역 변수를 남겨두는 현상
- 살아있는 a와 같은 지역변수
- 예

```
<script>
    function countVisit() {
      var count = 0;

      var plus = function () {
        return count += 1;
      }

      return plus;
    }

    var num = countVisit();
    num();
    num();
    num();
    alert(num( ));
  </script>
```

- 주의할 점

  → 반환되는 plus() 함수 호출 시 처음에는 count값이 0에서 시작하나 그 자체의 값이 변화되기 때문에 호출하는 횟수에 따라 달라진 count 값을 리턴하게 됨

▣ 정리하기

1. 함수의 생성

- 객체 자료형 중 하나로, 특정한 기능을 제공하기 위한 코드의 집합으로써, 호출을 통해 결과값을 얻음
- 리턴값 : 함수를 호출하면서 함수의 입력으로 넘겨주는 값을 매개 변수라 하며, return 키워드를 이용하여 돌려 받는 값
- 함수명이 없는 익명 함수와 함수명이 있는 선언적 함수로 생성할 수 있음

2. 다양한 함수의 형태

- 가변 인자 함수 : 매개변수의 개수가 변할 수 있으며, 입력된 매개 변수를 모두 활용할 수 있는 함수
- 내부 함수 : 함수 코드 내부에 정의된 함수
- 자기 호출 함수 : 생성과 동시에 실행되는 함수
- 콜백 함수 : 함수를 매개 변수로 전달하는 함수, 함수는 리턴값으로도 사용될 수 있음
- 클로저 : 종료된 외부 함수의 변수를 참조하는 함수 혹은 지역 변수를 남겨두는 현상, 남겨진 지역변수 그 자체