[11차시 교안]

<자바스크립트 함수, 객체>

1. 함수

① 입력을 받아 특정한 작업을 수행하여서 결과를 반환하는 블랙 박스

② 함수는 외부에서 함수가 호출되면 실행

③ 함수는 자바스크립트 코드의 어떤 부분에서도 호출 가능

```
function 함수 이름( )    // 함수 이름 낙타체
{                                  // 함수라는 것을 의미
        함수 몸체    // 호출되면 실행되는 코드
}
```

```
<!DOCTYPE html>
<html>
<head>
   <script>
      function showDialog() {
         alert("안녕하세요?");
      }
   </script>
</head>

<body>
   <button onclick="showDialog()">대화상자오픈</button>
</body>
</html>
```

2. 인수와 매개 변수

(1) 인수(argument)

① 함수 호출 시 함수로 전달하는 값

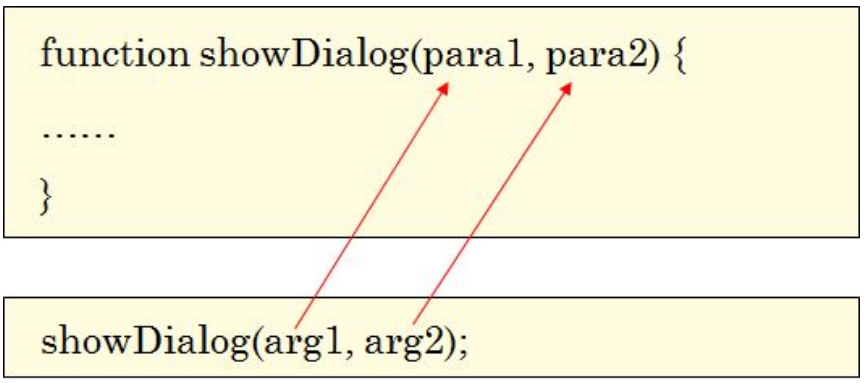
② 인자

③ 함수 내부에서 사용 가능

④ 콤마로 구분하여 여러 개의 인수 전달

(2) 매개변수(parameter) : 함수 선언 시 인수를 받을 변수

(3) 인수와 매개 변수는 선언된 순서대로 매칭



(4) 버튼이 눌려지면 경고창을 띄우는 프로그램

```
<html>
<head>
   <script>
      function greeting(name, position) {
         alert(name + " " + position + "님을 환영합니다.");
      }
   </script>
</head>
<body>
   <button onclick="greeting('홍길동','부장')">눌러보세요!</button>
</body>
</html>
```

3. 무명함수(anonymous function)

가. 함수에 이름을 주지 않고 만들어서 한 번만 사용하는 경우 유용

나. 자바스크립트에서 함수는 객체처럼 취급

① 변수에 저장되었다가 나중에 호출될 수 있음

다. 이벤트 처리함수 작성시 많이 사용

① 필요할 때 즉석에서 만들어서 사용

```
function showDialog( ) {
    alert("안녕하세요?");
}
```

```
var greeting = function ( ) {
    alert("안녕하세요?");
};
greeting( );
```

4. 함수의 반환값

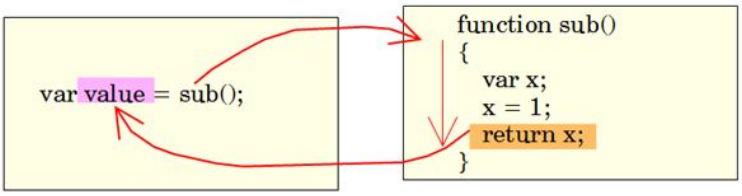
가. return 문장을 사용하여 외부로 값을 반환

나. return 문장

① 함수가 실행을 중지하고

② 지정된 값을 호출한 곳으로 반환하고

③ 함수 종료(단순 종료시에도 사용)




```
function divide(a, b) {
    if( b== 0) {
        return;
    }
    return a/b;
}
```

5. 지역 변수 & 전역 변수

가. 지역변수

① 함수 안에서 선언된 변수

② 함수 안에서만 사용 가능

③ 다른 함수에서도 똑같은 이름으로 변수 선언 가능(동일한 변수 아님)

④ 함수가 종료되면 자동적으로 소멸됨

나. 전역변수

① 함수 외부에서 선언된 변수

② 웹 페이지 상의 모든 스크립트와 모든 함수에서 사용 가능

③ 프로그램 종료 시 소멸

```
function add(a, b) {
    var sum = 0;    // add( )내부
    sum = a + b;
    return sum;
}
```

```
var sum = 0;      // 전역변수
    function add(a, b) {
    sum = a + b;
    return sum;
}
```

6. 자바스크립트 입출력

(1) alert( ) 함수

가. 자바스크립트는 웹 브라우저 안에서 실행됨

나. 자바스크립트의 입력과 출력은 모두 HTML 문서

다. alert( )

① 사용자에게 경고하는 윈도우를 화면에 띄우는 함수

② 사용자가 경고 윈도우를 제거할 때까지 다음 작업이 진행되지 않음

(2) confirm( ) 함수

① 사용자에게 어떤 사항을 알려주고 확인이나 취소를 요구하는 윈도우를 화면에 띄우는 함수

② 확인을 클릭하면 true반환, 그렇지 않으면 false 반환

(3) prompt( ) 함수

① 사용자에게 어떤 사항을 알려주고 사용자가 답변을 입력할 수 있는 윈도우를 화면에 띄우는 함수

② 사용자가 입력한 내용을 문자열로 반환

③ 입력 양식을 만들지 않고 간단히 사용할 경우 용이

(4) HTML 문서에 쓰기

가. document.write( )

① 자바스크립트에서 HTML 문서에 어떤 요소를 추가하기 위해 사용

② 큰따옴표 안의 내용이 HTML 문서로 출력

③ 주의할 점

a. 페이지가 적재된 후 document.write( )를 호출하면 전체 HTML 페이지가 다시 씌여짐

b. <body>요소 안에서 다른 요소들과 같이 실행되는 경우 사용하는 것이 좋음

```
<html>
<body>
   <h1 id="test">This is a heading.</h1>
   <script>
      function func() {
         document.write("페이지가 적재된 후에 write()를 사용하면 다시
                        씌여집니다.");
      }
   </script>
   <button type="button" onclick="func()">클릭하세요!</button>
</body>
</html>
```

7. 객체

가. 자바스크립트의 가장 기초적인 자료형(거의 모든 것이 객체)

문자열(String), 수치형(Number), 부울형(Boolean) 객체처럼 구현됨

나. 객체(object)

① 실제 세상에 존재하는 사물을 모델링 한 것

② 데이터와 동작으로 이루어짐

a. 데이터(변수) : 객체가 가지고 있는 특성값. 속성

자동차 : 메이커, 모델, 색상, 마력과 같은 속성

b. 동작(함수) : 객체가 수행할 수 있는 동작. 메서드

자동차 : 출발하기, 정지하기, 방향 전환하기 등의 동작도 가지고 있음

다. 객체 지향 프로그래밍

데이터와 동작을 하나로 합쳐서 프로그램을 작성하는 기법

8. 객체 생성 및 사용

(1) 객체의 종류

가. 내장 객체(bulit-in object)

① 생성자가 미리 작성되어 있음(생성자 정의 없이 사용 가능)

② Date, String, Array, document 등

나. 사용자 정의 객체(custom object): 사용자가 생성자를 정의

(2) 객체 생성 방법

가. 객체를 객체 상수로부터 직접 생성

나. 생성자 함수를 이용하여 객체를 정의하고, new를 통하여 객체의 인스턴스를

생성

(3) 자바스크립트에는 클래스 없음

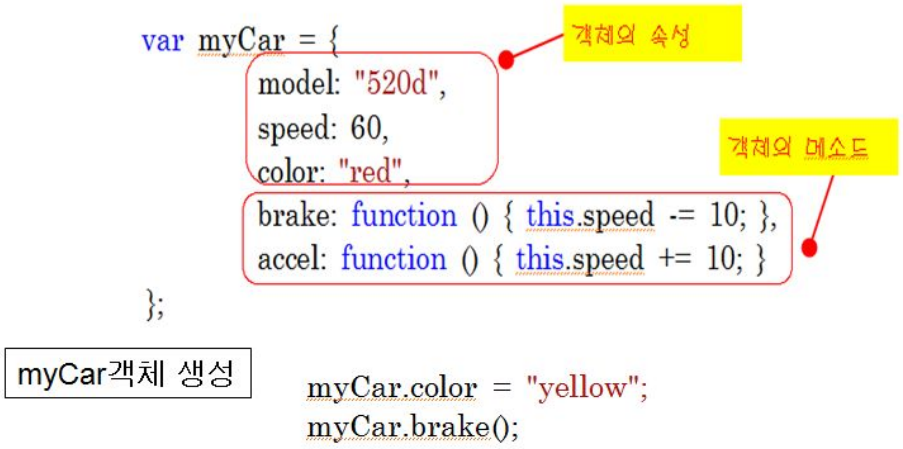
(4) 모든 것이 객체이고 생성자 함수가 클래스 역할을 흉내냄

(5) 객체 생성 방법

가. 객체 상수로부터 객체 생성(싱글톤)

① 객체를 하나만 생성 가능 – 싱글톤

② 추가로 객체를 생성하려면 동일한 코드를 반복해야 함



나. 생성자를 이용한 객체 생성

① 사용자 정의 객체

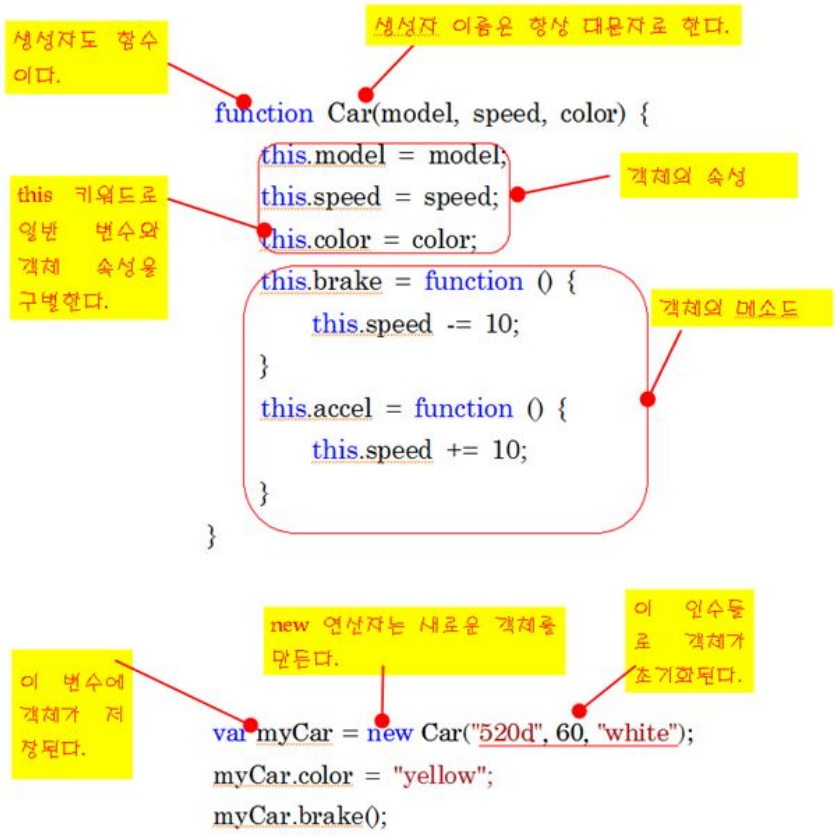
a. 개발자 자신만의 객체

b. 생성자 함수를 이용하여 생성 : 객체를 초기화 하는 역할

c. 객체 생성 연산자 new

d. 사용자가 원하는 개수만큼 객체 생성 가능

② this : 코드를 실행하는 현재 객체



9. 객체 멤버 사용하기

가. 객체 멤버 : 객체 안의 속성과 메서드

나. 객체 멤버 사용을 위해서 .(dot) 연산자 사용

① myCar라는 객체 안에 color 속성을 red로 변경하는 문장

a. myCar.color = "red" // 속성

b. myCar.break( ); // 메서드

```
<html>
<body>
  <script>
    function Car(model, speed, color)
    {
      this.model=model;
      this.speed=speed;
      this.color = color;
      this.brake = function () {
        this.speed -= 10;
      }
      this.accel = function () {
        this.speed += 10;
      }
    }
```

```
myCar = new Car("520d", 60, "red");

document.write("모델:" + myCar.model +" 속도:" + myCar.speed + "<br >");

myCar.accel();

document.write("모델:" + myCar.model + " 속도:" + myCar.speed + "<br >");

myCar.brake();

document.write("모델:" + myCar.model + " 속도:" + myCar.speed + "<br >");

  </script>

</body>

</html>
```

10. 기존 객체의 속성과 메서드 추가하기

가. 기존에 존재하고 있던 객체에도 속성 추가 가능

나. 생성자 변경 없이 단순히 값을 대입하는 문장을 적어주면 됨

다. 기존의 myCar 객체에 새로운 속성 turbo와 새로운 메서드 showModel( )을 추가하는 예

```
myCar.turbo = true;
myCar.showModel = function( ) {
    alert("모델은 " + this.model + "입니다.")
}
```

11. 프로토타입

가. 하나의 객체가 가지고 있는 속성과 메서드는 다른 객체가 전혀 공유할 수 없음

나. 속성이나 메서드를 여러 객체가 공유하는 것이 어떤 경우에는 필요

```
function Point(xpos, ypos) {
  this.x = xpos;
  this.y = ypos;
  this.getDistance = function () {
        return Math.sqrt(this.x * this.x + this.y * this.y);
  };
}
var p1 = new Point(10, 20);
var p2 = new Point(10, 30);
```

다. 만약 객체가 100개 만들어지면 getDistance( )메서드도 100개 생성

① 객체 마다 getDistance( ) 메서드를 가지고 있을 필요 없음

② 메모리 낭비, 비효율적

라. getDistance( )는 모든 객체가 공유하면 됨

마. 객체는 자신만의 데이터를 가져야 하지만 메서드는 가급적 서로 공유하는 것이 좋음

바. 객체들 사이에서 메서드 공유 방법

① 다른 객체 지향 언어 : 클래스 (객체에 대한 설계도)

② 자바스크립트에서는 프로토타입을 통하여 가능

사. 자바스크립트의 모든 객체들은 prototype이라는 숨겨진 객체를 가짐

아. 프로토타입 객체를 이용하여서 공유되는 메소드 작성 가능

```
function Point(xpos, ypos) {
   this.x = xpos;
   this.y = ypos;
}
Point.prototype.getDistance = function () {
   return Math.sqrt(this.x * this.x + this.y * this.y);
};
```

12. 프로토타입 체인

자바스크립트에서 속성이나 메소드를 참조하게 되면 검색하는 순서

① 객체 안에 속성이나 메소드가 정의되어 있는지 체크

② 객체 안에 정의되어 있지 않으면 객체의 prototype이 속성이나 메소드를 가지고 있는지 체크

③ 원하는 속성/메소드를 찾을 때까지 프로토타입 체인(chain)을 따라서 올라감

13. object 객체

가. 자바스크립트 객체의 부모가 되는 객체

자바스크립트의 모든 객체는 Object 객체를 기초로 하여 생성

나. Object 객체의 메서드는 하위 객체에서 재정의(오버라이딩)해서 사용 가능

속성/메서드	설명
constructor	속성으로 생성자 함수를 가리킴 Var d = new Date( ); d.Constructor는 Date( )와 같음
valueOf( )	객체를 숫자로 변환하는 메서드
toStirng( )	객체의 값을 문자열로 변환하는 메서드
hasOwnProperty( )	전달 인수로 주어진 속성을 가지고 있으면 true 반환
isPrototypeOf( )	현재 객체가 전달 인수로 주어진 객체의 프로토타입이면 true 반환

14. 자바스크립트 내장 객체

① 자바스크립트가 가지고 있는 객체

② 내장 객체만 가지고도 프로그램 개발이 충분함

(1) Date 객체

가. 날짜와 시간 작업을 하는데 사용되는 가장 기본적인 객체

나. 새로운 Date 객체 생성

다. Date 객체 생성자

① 다양한 형태의 지원 가능

② 월은 0부터 시작(1월이면 0)

```
new Date( );                  // 현재 날짜와 시간
new Date(millisecond);  // 1970년 1월1일 이후의 밀리초(1초를  1000으로 나눈 숫자)
New Date(dateString);   // 다양한 문자열
New Date(year, month, date[, hour[, minute[, seconds[,ms]]]]);
```

라. Date 객체의 메소드

- getDate() (1-31 반환)
- getDay() (0-6 반환)
- getFullYear() (4개의 숫자로 된 연도 반환)
- getHours() (0-23 반환)
- getMilliseconds()(0-999)
- getMinutes()(0-59)
- getMonth()(0-11)
- getSeconds()(0-59)

- setDate()
- setDay()
- setFullYear()
- setHours()
- setMilliseconds()
- setMinutes()
- setMonth()
- setSeconds()

① getXXX() 타입 메서드

a. 접근자

b. 객체의 속성을 추출하는데 사용

② setXXX() 타입 메서드

a. 설정자

b. 객체의 속성을 설정하는데 사용

15. 날짜 비교 예제

가. 두 개의 날짜를 비교하기 위해서

① 모든 날짜를 1970년 1월 1일 이후의 밀리초로 변환 : getTime( ) 이용

② 날짜 간격을 구하기 위해서 밀리초의 차이값을 (1000*60*60*24)로 나누어줌

나. 쇼핑몰에서 물건 교환을 위한 날짜 검사 프로그램

```
<html>
<head>
   <script>
      function checkDate() {
         var s = document.getElementById("pdate").value;
         var pdate = new Date(s);
         var today = new Date();
```

```
         var diff = today.getTime() - pdate.getTime();
         var days = Math.floor(diff / (1000 * 60 * 60 * 24));     // floor() : 내림함수
         if (days > 30) {
            alert("교환 기한이 지났습니다.");
         }
      }
   </script>
</head>
<body>
      구입날짜:
      <input type="date" id="pdate">
      <button onclick="checkDate()">검사</button>
</body>
</html>
```

16. 타이머 예제

```
<div id='remaining'></div>
<script>
   function datesUntilNewYear() {
      var now = new Date();
      var newYear = new Date('January 1, ' + (now.getFullYear() + 1));
      var diff = newYear - now;    // 밀리초 단위
      var milliseconds = Math.floor(diff % 1000);
      diff = diff / 1000;
      var seconds = Math.floor(diff % 60);
      diff = diff / 60;
      var minutes = Math.floor(diff % 60);
      diff = diff / 60;
```

```
      var hours = Math.floor(diff % 24);
      diff = diff / 24;
      var days = Math.floor(diff);

      var outStr = '내년도 신정까지 ' + days + '일, ' + hours + '시간, ' + minutes;
      outStr += '분, ' + seconds + '초' + ' 남았읍니다.';

      document.getElementById('remaining').innerHTML = outStr;
      // 1초가 지나면 다시 함수를 호출한다.
      setTimeout("datesUntilNewYear()", 1000);
   }
   // 타이머를 시작한다.
   datesUntilNewYear();
</script>
```

17. Number 객체

(1) Number 객체

① 수치형 값을 감싸서 객체로 만들어 주는 랩퍼(wrapper) 객체

랩퍼객체 : 수치값을 직접 사용할 수 없고 객체가 필요한 경우

② var num = new Number(7); // Number 객체 생성

(2) 메소드

① toFixed([digits]) // 고정 소수점 방식으로 반환

a. Var num = 123.456789;

b. Document.writeln(num.toFixed(1) + '
 '); // 123.5

② toPrecision([precision)) // 유효숫자 수 지정

a. Var num = 123.456789;

b. Document.writeln(num.toPrecision(1) + '
 '); // 1e+2

③ toString([radix]) // 주어진 진법으로 숫자 반환

18. String 객체

(1) 속성

가. length : 문자열의 길이

```
<html>
<head>
  <script>
    function checkID( ) {
      var s = document.getElementById("id").value;
      if (s.length < 6) {
        alert("아이디는 6글자 이상이어야 합니다.");
      }
    }
  </script>
</head>
<body>
    아이디: <input type="text" id="id" size=10>
    <button onclick="checkID( )">검사</button>
</body>
</html>
```

(2) 메소드

가. 대소문자 변환 : toUpperCase(), toLowerCase()

```
<script>
  var s = 'aBcDeF';
  var result1 = s.toLowerCase();
  var result2 = s.toUpperCase();
  document.writeln(result1); // 출력: abcdef
  document.writeln(result2); // 출력: ABCDEF
</script>
```

나. concat()

① 하나의 문자열을 다른 문자열과 합침

② 새로운 문자열 생성

③ +연산자와 동일한 결과

다. indexOf()

① 문자열 검색

② 문자열 안에서 주어진 텍스트가 처음 등장하는 위치 반환

③ 첫 번째 위치가 0

라. match()

① 문자열 매칭

② 문자열 안에서 일치하는 콘텐츠를 탐색하는데 사용

③ 정규식 사용 가능

마. replace()

① 문자열 대체

② 문자열 안에서 주어진 값을 다른 값으로 대체

③ 정규식 사용 가능

바. split(delimiter[, linit]))

① 첫 번째 인수를 분리자로 하여 주어진 문자열 분리한 후

② 각 항목을 가지고 있는 배열 반환

③ 문자열을 배열로 변환 가능

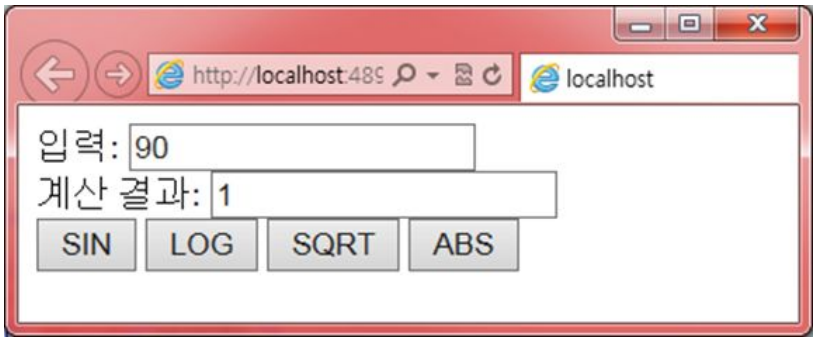
19. Math객체

수학적 작업을 위한 객체

생성자가 아니라 객체이므로 new를 통하여 객체를 생성할 필요없이 바로 사용

메소드	설명
abs(x)	절대값
acos(x), asin(x), atan(x)	아크 삼각함수
ceil(x), floor(x)	실수를 정수로 올림, 내림 함수
cos(x), sin(x), tan(x)	삼각함수
exp(x)	지수함수
log(x)	로그함수
max(x,y,z,...,n)	최대값
min(x,y,z,...,n)	최소값
pow(x,y)	지수함수 x^y
random()	0과 1 사이의 난수값 반환
round(x)	반올림
sqrt(x)	제곱근

20. 계산기 예제




```
<body>
    <form name="calculator">
        입력:              <input type="text" name="number1"><br />
        계산 결과:         <input type="text" name="total"><br />
        <input type="button" value="SIN" onclick="calc(1);">
        <input type="button" value="LOG" onclick="calc(2);">
        <input type="button" value="SQRT" onclick="calc(3);">
        <input type="button" value="ABS" onclick="calc(4);">
    </form>

</body>
</html>
```

```
<html>
<head>
   <script>
      function calc(type) {
         x = Number(document.calculator.number1.value);
         if (type == 1)
            y = Math.sin((x * Math.PI) / 180.0);
         else if (type == 2)   y = Math.log(x);
         else if (type == 3)   y = Math.sqrt(x);
         else if (type == 4)   y = Math.abs(x);
         document.calculator.total.value = y;
      }
   </script>
</head>
```