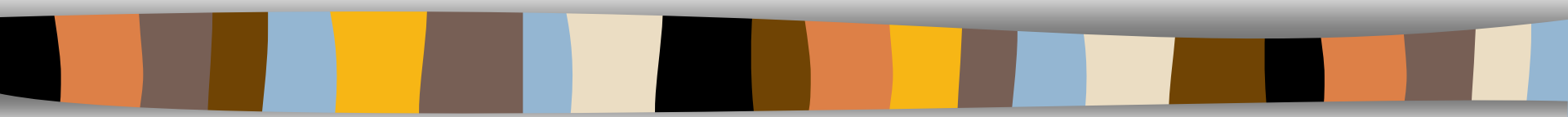
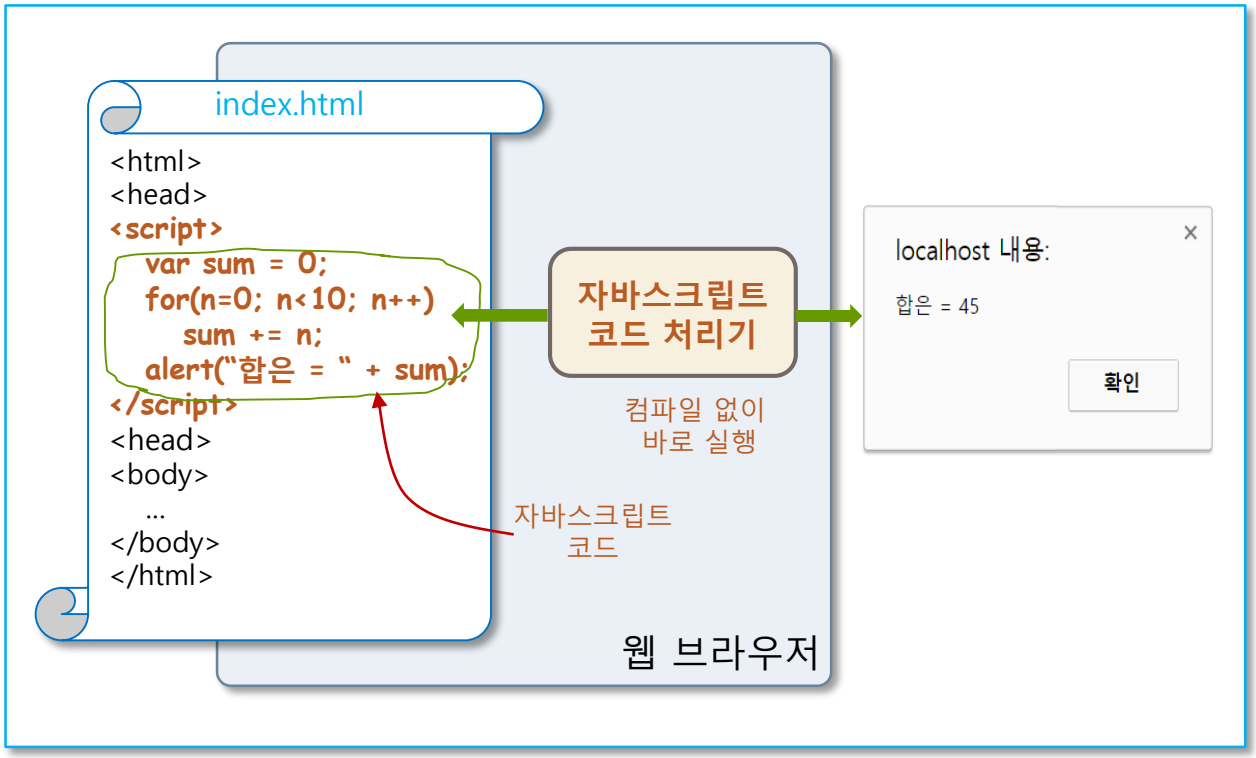
Javascript Basics



자바스크립트 언어

- Javascript
  - 1995년 넷스케이프 개발
  - Netscape Navigator 2.0 브라우저에 최초 탑재
  - 웹 프로그래밍 개념 창시
- 특징
  - HTML 문서에 내장
    - 조각 소스 코드
  - 스크립트 언어
    - 인터프리터 실행
    - 컴파일 필요 없음
  - 단순
    - C언어 구조 차용
    - 배우기 쉬움



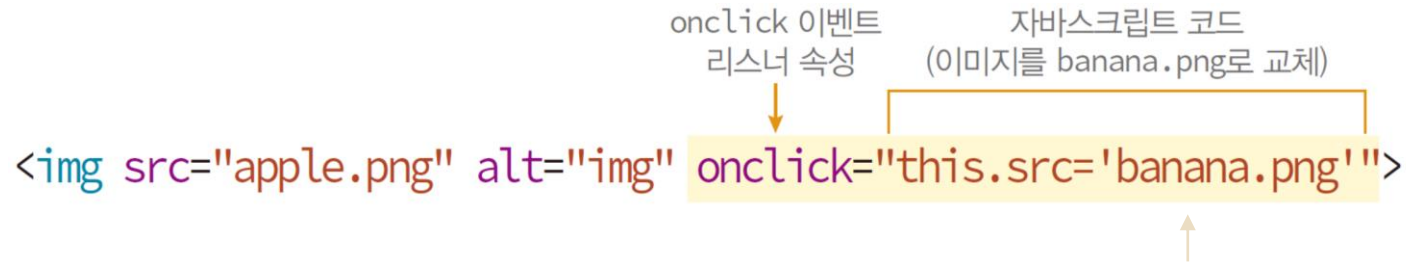
웹 페이지에서 자바스크립트의 역할

- 사용자의 입력 데이터 연산
- 웹 페이지 내용 및 모양의 동적 제어
  - HTML 태그의 속성, 콘텐츠, CSS 프로퍼티 값 동적 변경
- 브라우저 제어
  - 브라우저 윈도우 크기와 모양 제어
  - 새 윈도우 열기/닫기
  - 다른 웹 사이트 접속
  - 히스토리 제어
- 웹 서버와의 통신
- 웹 애플리케이션 작성
  - 캔버스 그래픽, 로컬/세션 스토리지 저장, 위치정보서비스 등

자바스크립트 코드의 위치

- 자바스크립트 코드 작성이 가능한 위치
  1. HTML 태그의 이벤트 리스너 속성에 작성
  2. <script></script> 태그에 작성
  3. 자바스크립트 파일에 작성
  4. URL 부분에 작성

방법1. HTML 태그의 이벤트 리스너에 자바스크립트 코드 작성



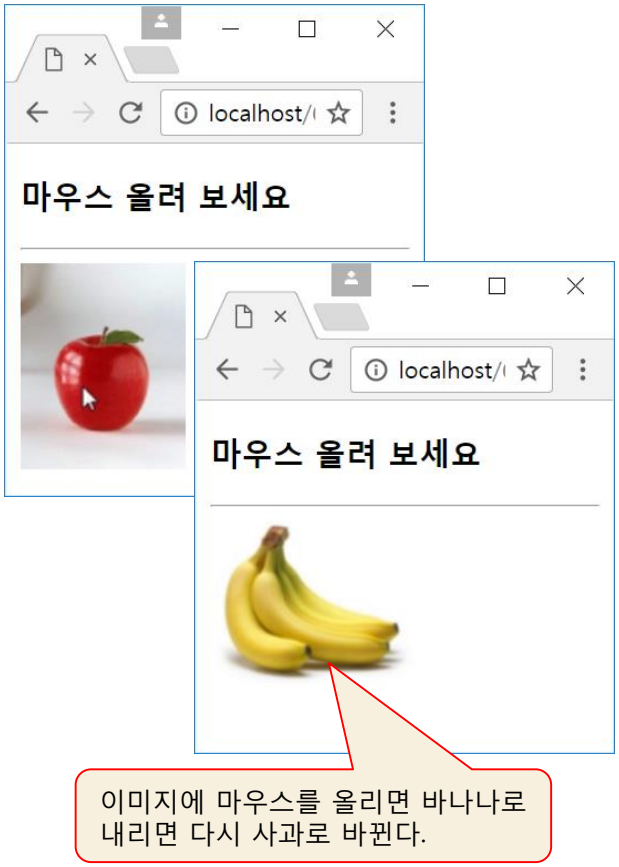
예제: 이벤트 리스너 속성에 작성


```
<!DOCTYPE html>
<html>
<head>
<title>이벤트 리스너 속성에 자바스크립트 코드</title>
</head>
<body>
<h3>마우스 올려 보세요</h3>
<hr>
<img src="media/apple.png" alt="이미지"
        onmouseover="this.src='media/banana.png'"
        onmouseout="this.src='media/apple.png'">
</body>
</html>
```

이벤트 리스너 속성

this는 현재 img 태그를 가리키는 자바스크립트 키워드

자바스크립트 코드



이미지에 마우스를 올리면 바나나로 내리면 다시 사과로 바뀐다.

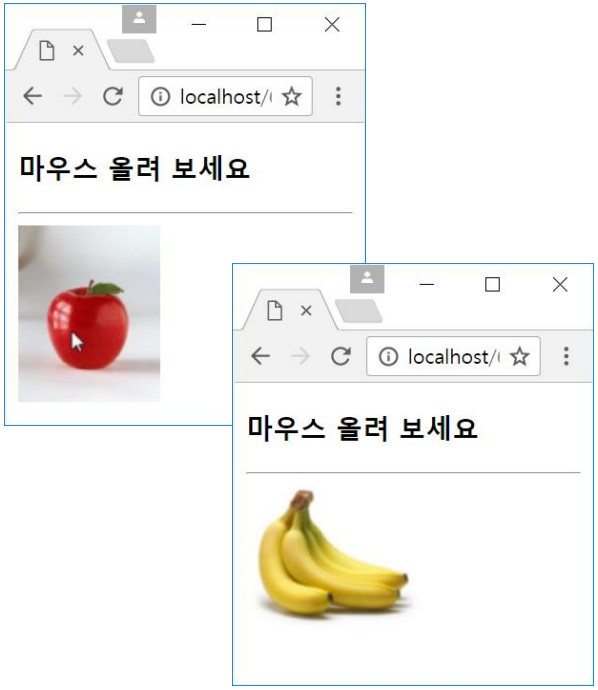
방법2. <script></script> 태그에 작성

- 특징
  - <head></head>나 <body></body> 내 모두 가능
  - 웹 페이지 내에 여러 번 삽입 가능

obj는 전달받은 img 태그를 가리킴

```
<script>
function over(obj) {
   obj.src="media/banana.png";
}
function out(obj) {
   obj.src="media/apple.png";
}
</script>
</head>
<body>
<h3>마우스 올려 보세요</h3>
<hr>
<img src="media/apple.png" alt="이미지"
        onmouseover="over(this)"
        onmouseout="out(this)">
</body>
</html>
```

this는 현재 img 태그를 가리키는 자바스크립트키워드



방법3: 별도 파일에 작성

- 자바스크립트 코드 파일 저장
  - 확장자 .js 파일에 저장
  - <script> 태그 없이 자바스크립트 코드만 저장
- 여러 웹 페이지에서 불러 사용
  - 웹 페이지마다 자바스크립트 코드 작성 중복 불필요
  - <script> 태그의 src 속성으로 파일을 불러 사용

```
<script src="파일이름.js">
        // 이곳에 자바스크립트 코드 추가 작성 가능
</script>
```

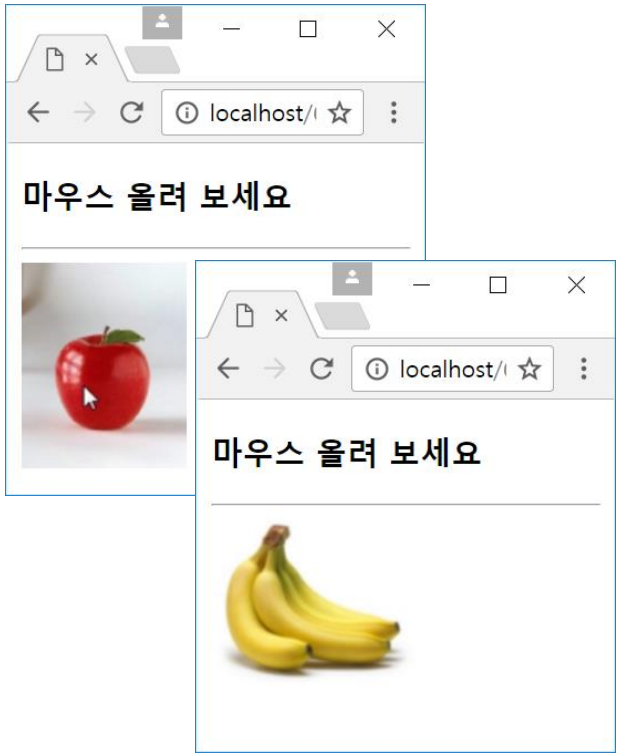
예제: 별도 파일로 작성

lib.js

```
/* 자바스크립트 파일 lib.js */
function over(obj) {
    obj.src="media/banana.png";
}
function out(obj) {
    obj.src="media/apple.png";
}
```

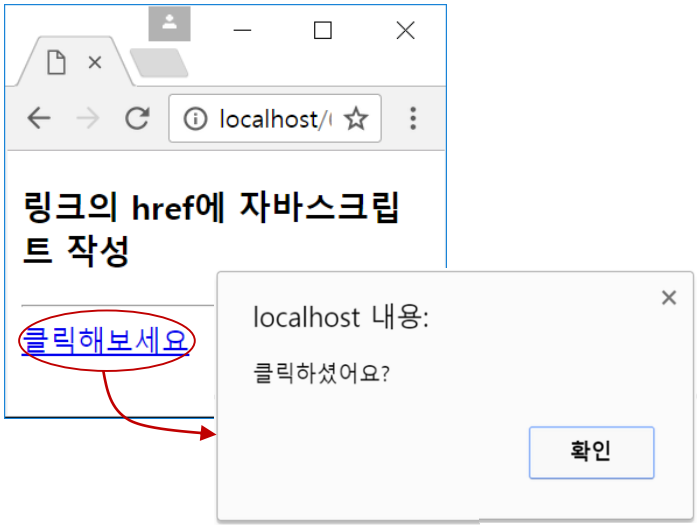
lib.js
불러오기

```
<!DOCTYPE html>
<html>
<head><title>외부 파일에 자바스크립트 작성</title>
<script src="lib.js">
</script>
</head>
<body>
<h3>마우스 올려 보세요</h3>
<hr>
<img src="media/apple.png" alt="이미지"
      onmouseover="over(this)"
      onmouseout="out(this)">
</body>
</html>
```



방법4: href에 작성


```
<!DOCTYPE html>
<html>
<head><title>URL에 자바스크립트 작성</title>
</head>
<body>
<h3>링크의 href에 자바스크립트 작성</h3>
<hr>
<a href="javascript:alert('클릭하셨어요?')">
  클릭해보세요</a>
</body>
</html>
```



자바스크립트로 출력

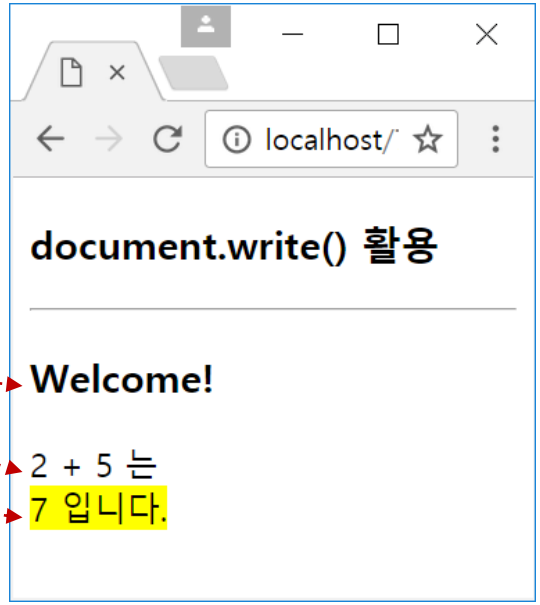
- 자바스크립트로 HTML 콘텐츠를웹 페이지에 직접 삽입
  - 바로 브라우저 윈도우에 출력
  - document.write()

    예) document.write("<h3>Welcome!</h3>");

  - document.writeln()
    - writeln()은 텍스트에 '₩n ' 을 덧붙여 출력
    - '₩n'을 덧붙이는 것은 고작해야 빈칸 하나 출력
    - 다음 줄로 넘어가는 것은 아님

예제: document.write()로 웹 페이지 출력


```
<!DOCTYPE html>
<html>
<head><title>document.write() 활용</title>
</head>
<body>
<h3>document.write() 활용</h3>
<hr>
<script>
   document.write("<h3>Welcome!</h3>");
   document.write("2 + 5 는 <br>");
   document.write("<mark>7 입니다.</mark>");
</script>
</body>
</html>
```



입출력 대화상자

- 종류
  - 프롬프트 대화상자
    - prompt("메시지", "디폴트 입력값")
    - 사용자로부터 문자열을 입력 받아 리턴
  - 확인 대화상자
    - confirm("메시지")
    - “메시지”를 출력하고 ‘확인/최소(OK/CANCEL)’버튼을 가진 다이얼로그 출력
    - ‘확인’ 버튼을 누르면 true, '취소' 버튼이나 강제로 다이얼로그를 닫으면 false 리턴
  - 경고 대화상자
    - alert("메시지") 함수
    - 메시지’와 '확인' 버튼을 가진 다이얼로그 출력, 메시지 전달

데이터 타입

- 자바스크립트 언어에서 다루는 데이터 종류
  - 숫자 타입 : 정수, 실수(예: 42, 3.14)
  - 논리 타입 : 참, 거짓(예: true, false)
  - 문자열 타입(예: ‘좋은 세상’, "a", "365", "2+4")
  - 객체 레퍼런스 타입 : 객체를 가리킴. C 언어의 포인터와 유사
  - null : 값이 없음을 표시하는 특수 키워드.

  - 자바스크립트에는 문자 타입 없음. 문자열로 표현

- 변수
  - 스크립트 언어로 변수 선언 없음
  - [var] var_nam e [= value][;]

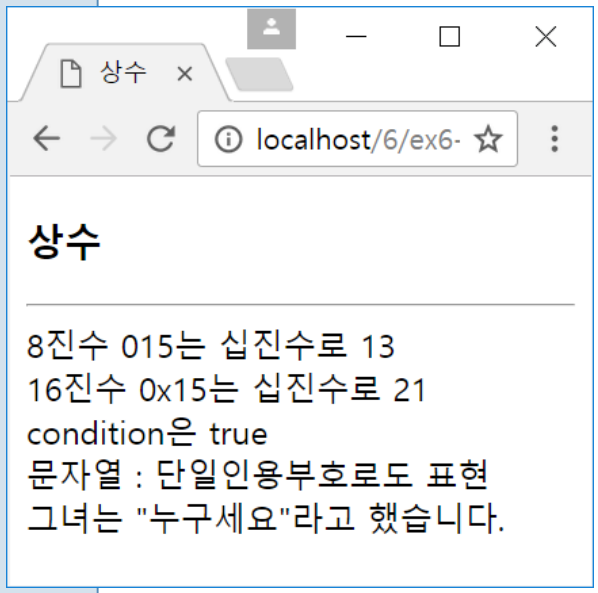
상수

- 상수(literal)
  - 데이터 값 그 자체
- 상수 종류

상수의 종류		특징	예
정수	8진수	0으로 시작	var n = 015; // 8진수 15. 10진수로 13
	10진수		var n = 15; // 10진수 15
	16진수	0x로 시작	var n = 0x15; // 16 진수 15. 10진수로 21
실수	소수형		var height = 0.1234;
	지수형		var height = 1234E-4; // $1234 \times 10^{-4} = 0.1234$
논리	참	true	var condition = true;
	거짓	false	var condition = false;
문자열		""로 묶음	var hello = "안녕하세요";
		''로 묶음	var name = 'kitae';
기타	null	값이 없음을 뜻함	var ret = null;
	NaN	수가 아님을 뜻함	var n = parseInt("abc"); // 이때 parseInt()는 NaN을 리턴

예제: 상수


```
<!DOCTYPE html>
<html>
<head><title>상수</title></head>
<body>
<h3>상수</h3>
<hr>
<script>
   var oct = 015; // 015는 8진수. 10진수로 13
   var hex = 0x15; // 0x14는 16진수. 10진수로 21
   var condition = true; // True로 하면 안됨

   document.write("8진수 015는 십진수로 " + oct + "<br>");
   document.write("16진수 0x15는 십진수로 " + hex + "<br>");
   document.write("condition은 " + condition + "<br>");
   document.write('문자열 : 단일인용부호로도 표현' + "<br>");
   document.write("그녀는 ₩"누구세요₩"라고 했습니다.");
</script>
</body>
</html>
```



구문 규칙

- Statements: can use the C syntax
- Operator
  - Arithmetic: +, -, *, /, %
  - Relational: >, >=, <, <=, !=, ==
  - Logical: ||, &&, !
  - Assign: =, +=, -=, *=, /=, %= , ++, --
- Control Statement
  - Condition: If, switch
  - Iterator: While, do ~ while, for

예제 문자열 연산

- 문자열 연결
  - + , + =

```
"abc" + "de"        // "abcde"
"abc" + 23          // "abc23"
23 + "abc"          // "23abc"
23 + "35"           // "2335"
23 + 35             // 58, 정수 더하기
```

  - 순서에 유의

```
23 + 35 + "abc";    // 23 + 35 -> 58로 먼저 계산, 58 + "abc" -> "58abc"
"abc" + 23 + 35;    // "abc" + 23 -> "abc23"로 먼저 계산, "abc23" + 35 -> "abc2335"
```

- 문자열 비교
  - 비교 연산자(!=, ==, > , <, <=, >=) 사용
  - 사전 순으로 비교 결과 리턴

```
var name = "knu";
var res = (name == "knu"); // 비교 결과 true, res = true
var res = (name > "park"); // name이 "park"보다 사전순으로 앞에 나오므로 res = false
```

사용자 정의 함수

- 선언

  function *func_name*([*arg1*, …*argn*]) {

  statements; [return expression;]

  }

- 호출

  func_name([arg1, …argn]);

  or

  Varname= *func_name*([arg1, …argn]);

```
<HTML>
<HEAD>
<TITLE> Example: Function 1 </TITLE>
<script>
        function big(x, y) {
                if(x>y) return x;
                else return y;
        }
</script> </HEAD>
<BODY>
        <script> document.write(big(1, 2)); </script> </BODY>
</HTML>
```

```
<TITLE> Example: func2 </TITLE> <script>
function small(x, y) {
        if(x<y) return x;
        else return y;
}
function common(x,y) {
        var res = 1;
        var s = small(x,y);
        for(var i = 1; i<=s; i++) {
            if(!(x % i) && !(y % i)) {
                res = i;
            }
        }
        return res;
} </script> </HEAD>
<BODY> <h3> 함수에 대한 예제</h3><hr>
<script>
        x = prompt("양수를입력하시오", 1);
        x = parseInt(x); var y = prompt("양수를입력하시오", 1);
        y = parseInt(y);
        document.write("x=", x, ", y=", y, "GCM:  common(x, y));
</script>
```

```
<title>간단한계산기제작</title>
<script >
function cal() {
      exp=prompt('계산할수식을입력하세요.','23-5')
      result=eval(exp)
      alert(result)
}
</script></HEAD>
<BODY>
<form>
      <input type=button value="계산하기" onClick="cal()">
</form>
</BODY>
```

전역 함수

- 대표적인 자바스크립트 함수
  - eval() 함수

    예) var res = eval("2*3+ 4*6"); // res는 32
  - parseInt() 함수

    예) var l = parseInt("32"); // "32"를 10진수로 변환, 정수 32 리턴

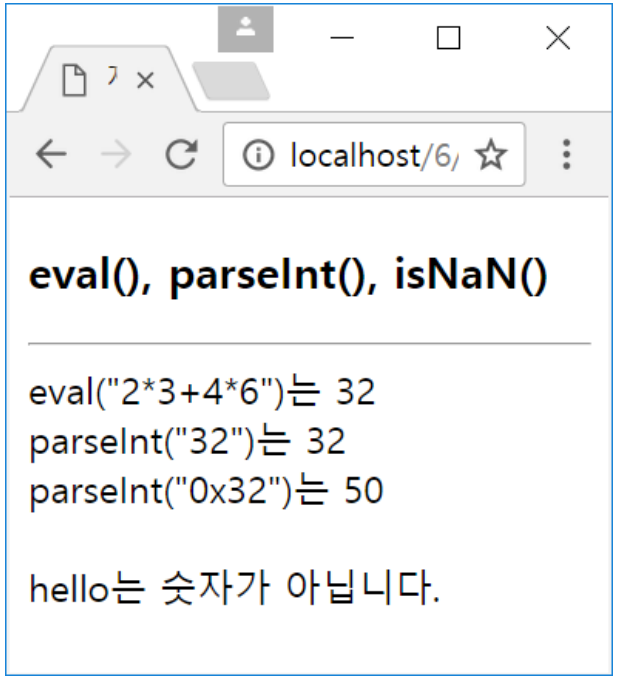
    var n = parseInt("0x32"); // "0x32"를 16진수로 해석, 정수 50 리턴
  - isNaN() 함수

    예) isNaN(32) // false 리턴

전역 함수명	설명
eval(exp)	exp의 자바스크립트 식을 계산하고 결과 리턴
parseInt(str)	str 문자열을 10진 정수로 변환하여 리턴
parseInt(str, radix)	str 문자열을 radix 진수로 해석하고, 10진 정수로 바꾸어 리턴
parseFloat(str)	str 문자열을 실수로 바꾸어 리턴
isFinite(value)	value가 숫자이면 true 리턴
isNaN(value)	value가 숫자가 아니면 true 리턴

예제: eval(), parseInt(), isNaN()

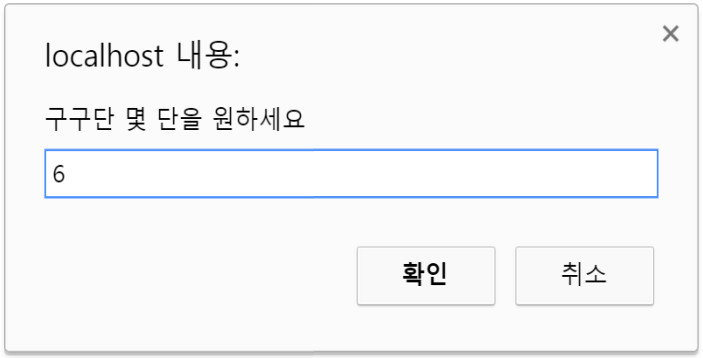

```
<!DOCTYPE html>
<html>
<head>
<title>자바스크립트 전역함수</title>
<script>
function evalParseIntIsNaN() {
    var res = eval("2*3+4*6"); // res는 32
    document.write("eval(₩"2*3+4*6₩")는 32<br>");
    var m = parseInt("32");
    document.write("parseInt(₩"32₩")는 " + m + "<br>");
    var n = parseInt("0x32");
    document.write("parseInt(₩"0x32₩")는 " + n + "<br><br>");

    // "hello"는 정수로 변환할 수 없으므로 parseInt("hello")는 NaN 리턴
    n = parseInt("hello");
    if(isNaN(n)) // true
        document.write("hello는 숫자가 아닙니다.");
}
</script>
</head>
<body>
<h3>eval(), parseInt(), isNaN()</h3>
<hr>
<script>
    evalParseIntIsNaN();
</script>
</body>
</html>
```

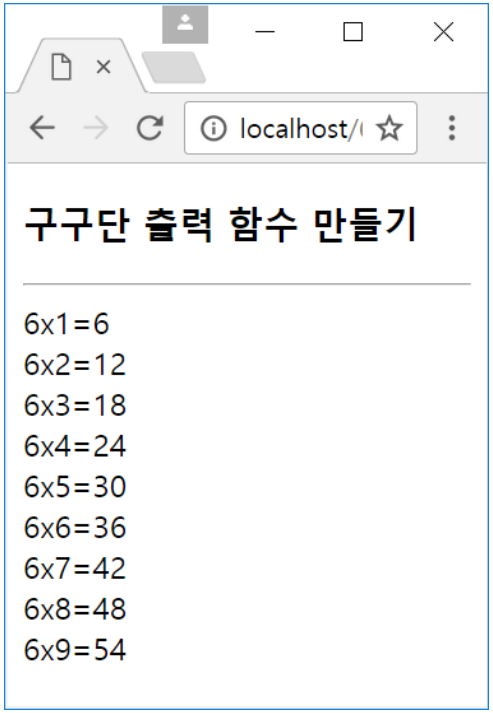


예제: 구구단 출력 함수 만들기


```
<!DOCTYPE html>
<html><head><title>함수 만들기</title>
<script>
function gugudan(n) { // 함수 작성
   var m = parseInt(n); // 문자열 n을 숫자로 바꿈
   if(isNaN(m) || m < 1 || m > 9) {
      alert("잘못입력하셨습니다.");
      return;
   }
   for(var i=1; i<=9; i++) { // i는 1~9까지 반복
      document.write(m + "x" + i + "=" + m*i + "<br>");
   }
}
</script>
</head>
<body>
<h3>구구단 출력 함수 만들기</h3>
<hr>
<script>
   var n = prompt("구구단 몇 단을 원하세요", ""); // n은 문자열
   gugudan(n); // 함수 호출
</script>
</body>
</html>
```

n이 1~9사이의 숫자가 아닌 경우 처리





배열

- 배열 생성 방법
  - []로 배열 만들기

```
var week = [“월”, “화”, “수”, “목”, “금”, “토”, “일”];
var plots = [-20, -5, 0, 15, 20];
```

  - Array 객체로 배열 만들기

```
var week = new Array(“월”, “화”, “수”, “목”, “금”, “토”, “일”);
var week = new Array(7);
```

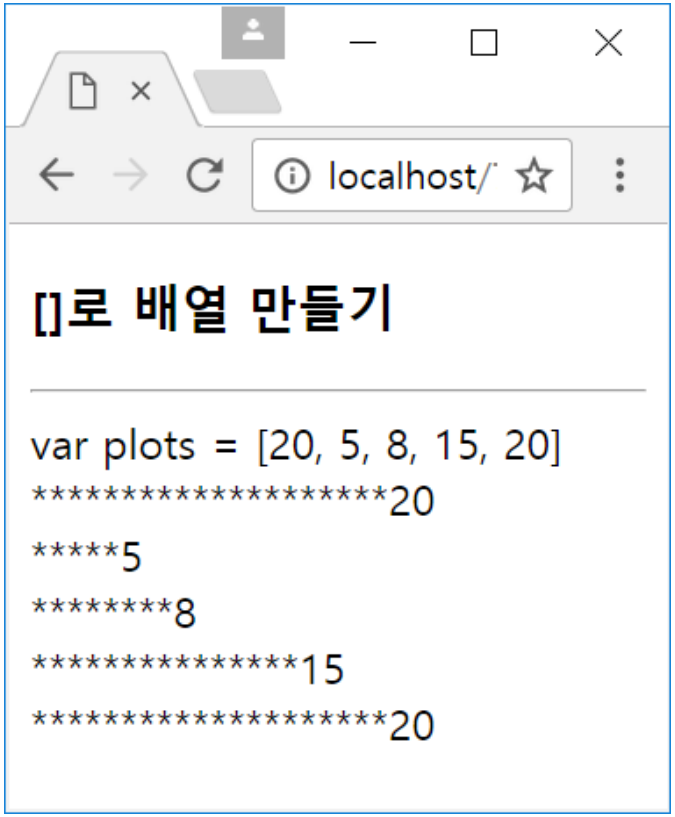
- 배열 사용
  - 배열의 크기는 고정되지 않고, 마지막 원소 추가 시 늘어남

```
plots[5] = 33; // plots 배열에 6번째 원소 추가. 배열 크기는 6이 됨
plots[6] = 22; // plots 배열에 7번째 원소 추가. 배열 크기는 7이 됨
```

  - length 프로퍼티 사용
    - var n = **week.length**;
  - 서로 다른 타입의 자료도 가능
  - 첨자는 0부터

예제1


```
<!DOCTYPE html>
<html><head><title>[]로 배열 만들기</title></head>
<body>
<h3>[]로 배열 만들기</h3>
<hr>
<script>
   var plots = [20, 5, 8, 15, 20]; // 원소 5개의 배열 생성
   document.write("var plots = [20, 5, 8, 15, 20]<br>");

   for(i=0; i<5; i++) {
      var size = plots[i]; // plots 배열의 i번째 원소
      while(size>0) {
         document.write("*");
         size--;
      }
      document.write(plots[i] + "<br>");
   }
</script>
</body>
</html>
```

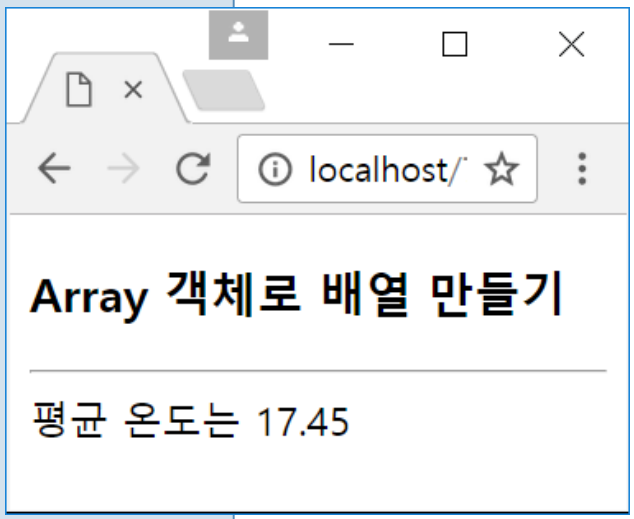


예제2


```
<!DOCTYPE html>
<html><head><title>Array 객체로 배열 만들기</title></head>
<body>
<h3>Array 객체로 배열 만들기</h3>
<hr>
<script>
   var degrees = new Array(); // 빈 배열 생성
   degrees[0] = 15.1;
   degrees[1] = 15.4;
   degrees[2] = 16.1;
   degrees[3] = 17.5;
   degrees[4] = 19.2;
   degrees[5] = 21.4;

   var sum = 0;
   for(i=0; i<degrees.length; i++)
      sum += degrees[i];

   document.write("평균 온도는 " + sum/degrees.length + "<br>");
</script>
</body>
</html>
```

배열 크기만큼 루프

배열 degrees의 크기, 6



예제3: 객체의 메소드 활용


```
<!DOCTYPE html>
<html><head><title>Array 객체의 메소드 활용</title>
<script>
    function pr(msg, arr) { document.write(msg + arr.toString() + "<br>"); }
</script>
</head>
<body>
<h3>Array 객체의 메소드 활용</h3>
<hr>
<script>
    var a = new Array("황", "김", "이");
    var b = new Array("박");
    var c;

    pr("배열 a = ", a);
    pr("배열 b = ", b);
    document.write("<hr>");

    c = a.concat(b); // c는 a와 b를 연결한 새 배열
    pr("c = a.concat(b)후 c = ", c);
    pr("c = a.concat(b)후 a = ", a);

    c = a.join("##"); // c는 배열 a를 연결한 문자열
    pr("c = a.join() 후 c = ", c);
    pr("c = a.join() 후 a = ", a);

    c = a.reverse(); // a.reverse()로 a 자체 변경. c는 배열
    pr("c= a.reverse() 후 c = ", c);
    pr("c= a.reverse() 후 a = ", a);

    c = a.slice(1, 2); // c는 새 배열
    pr("c= a.slice(1, 2) 후 c = ", c);
    pr("c= a.slice(1, 2) 후 a = ", a);

    c = a.sort(); // a.sort()는 a 자체 변경. c는 배열
    pr("c= a.sort() 후 c = ", c);
    pr("c= a.sort() 후 a = ", a);

    c = a.toString(); // toString()은 원소 사이에 ","를 넣어  문자열 생성
    document.write("a.toString() : " + c); // c 는 문자열
</script></body></html>
```