HTML5 웹 프로그래밍 입문(교수용)

8장. 자바스크립트 프로그래밍기초

목차

8.1 자바스크립트 시작하기

8.1.1 자바스크립트 개요와 특징

8.1.2 자바스크립트 실행 및 디버깅

자바스크립트 개요

- 개요 및 특징
  - 동적인 웹 문서 제작과 웹 응용프로그램 개발을 위한 사용자 인터페이스 개발을 위해서 필수적으로 사용됨
    - 자바 애플릿, CGI 스크립트 대체 가능
  - C/C++이나 자바 언어 등에 비해서 작성 및 실행이 매우 간편함
  - 인터프리터 (interpreter) 방식
- 자바스크립트 연혁
  - 라이브스크립트라는 이름으로 넷스케이프사에서 개발 시작
  - 1995년에 썬 (Sun, 현재 오라클)사와의 공동 개발로 자바스크립트(JavaScript)라는 이름을 가지게 됨
  - 현재는 ECMA(European Computer Manufacturers Association)에서 ECMA-262 혹은 ISO 16262로 표준 제정
    - 따라서, ECMAScript라고도 불리움

Nothing to do with Java

객체 기반의 자바스크립트

- 자바 언어의 영향을 받아서 문법적으로 비슷한 형태를 가지는 공통점이 있으나 자바 언어와는 다음과 같이 차이점을 가진다

	자바스크립트	자바 언어
실행 방식	웹 브라우저에서 바로 자바스크립트 코드를 해석하고 바로 실행 (스크립트/인터프리터 기반 언어)	자바 프로그램을 컴파일 후 변환된 object code를 자바가상머신에서 실행하는 방식 (컴파일 기반 언어)
성격	객체기반(object-based)	객체지향(object-oriented)
작성 형태	HTML 파일 내에 포함되어 작성됨	별도의 자바 프로그램 파일로 작성
변수형 선언 및 타입 검사	변수의 선언이 따로 필요 없으며 타입 검사도 매우 느슨함	변수의 선언이 필요하며 변수 타입의 검사가 매우 엄격함

자바스크립트 실행 및 디버깅

- HTML 파일 내에 자바스크립트 코드가 있다면 웹 브라우저가 자체 인터프리터를 이용해 그 스크립트 코드를 해석하고 실행
- 자바스크립트를 실행하는 동안 오류가 발생하더라도 치명적 오류가 아니라면 기본적으로 웹 브라우저는 그 오류를 무시하고 진행
- 개발 단계에서는 자바스크립트 실행시 발생한 오류를 개발자가 확인하고 수정하는 것이 바람직

자바스크립트 오류 확인

- [Chrome 설정 및 관리] 버튼( )을 누른 후 [도구(L)] → [자바스크립트 콘솔(J)]을

8.2 자바스크립트 작성하기

8.2.1 자바스크립트 작성 방법
8.2.2 자바스크립트 기본 문법
8.2.3 화면 출력 및 키보드 입력
8.2.4 제어문 및 반복문

자바스크립트 작성 방법

- 자바스크립트 코드는 HTML 파일 없이 웹브라우저에서 독립적으로 실행 될 수 없음
  - 반드시 HTML 파일 내에 포함되어 있어야 한다
- HTML 파일 내에 포함 시키는 두가지 방식
  - 웹문서 내장 방식
  - 외부 파일 참조 방식

웹문서 내장 방식

```
1  <!-- HTML documents... -->
2  ....
3  <script type = "text/javascript">
4      <!--
5          // 자바스크립트 코드
6      // -->
7  </script>
8  ....
9  <!-- HTML documents... -->
```

3: 자바스크립트 포함의 시작을 알리는 **<script>** 태그
4: 특수 주석 시작
5: 실제 자바스크립트 코드들이 위치하는 곳
6: 특수 주석 끝
7: 자바스크립트 포함의 끝을 알리는 클로징 태그

외부 파일 참조 방식

- **<script>** 태그의 **src** 속성의 값으로 자바스크립트 파일의 경로를 지정

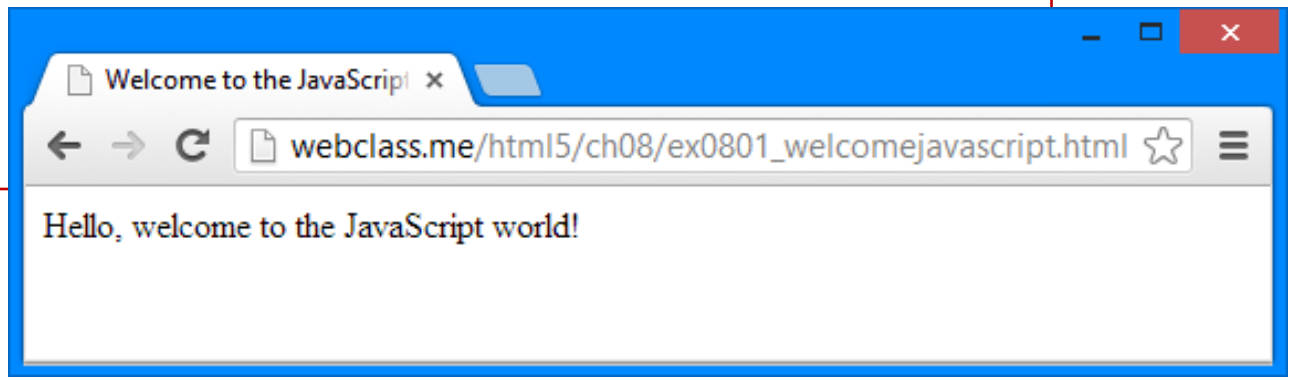
```
1 <!-- HTML documents... -->
2 ....
3 <script type="text/javascript" src="myscript.js">
4 </script>
5 ....
6 <!-- HTML documents... -->
```

- 자바스크립트 파일의 URL 경로 지정 가능

```
1 <!-- HTML documents... -->
2 ....
3 <script type="text/javascript"
4           src="http://webclass.me/html5/javascript_example.js">
5 </script>
6 ....
7 <!-- HTML documents... -->
```

자바스크립트 예제

```
1  <!DOCTYPE html>
2  <!-- hello.html
3     간단한 인사말을 화면에 표시하는 HTML/자바스크립트 기본 예제
4  -->
5  <html>
6  <head>
7  <title> Welcome to the JavaScript </title>
8  </head>
9  <body>
10  <script type="text/javascript">
11     <!--
12        document.write("Hello, welcome to the JavaScript world!");
13     // -->
14  </script>
15 </body>
16 </html>
```



- **"Hello, welcome to the JavaScript world!"**라는 문구를 출력

자바스크립트 기본 문법

- 기본 변수 타입
  - 대부분의 경우 자바스크립트 변수는 사용전에 미리 선언할 필요가 없음
  - 타입도 지정할 필요가 없다
- 내부적인 변수의 다섯가지 기본 형식

```
Number, String, Boolean, Undefined, Null
```

- 숫자의 표현 형태

```
125   1.25   0.125   .125   125.   12.e5   1.2e-5   12E5   12e5   .12e5
```

  - 정수든 실수든 관계없이 내부적으로 숫자는 모두 실수로 저장됨

자바스크립트 기본 변수 타입

기본 변수 타입	변수 값	비고
Number	정수, 실수 등 숫자 값을 가짐	
String	연속된 글자들로 이루어진 문자열 (공백도 가능함). 문자열의 시작과 끝은 작은 따옴표 (') 혹은 겹따옴표 (")로 지정	숫자 (Number)와 문자열 (String) 타입간에는 숫자 값에 대해 자동 형변환을 제공한다.
Boolean	true 혹은 false	조건식에서 사용
Undefined	undefined 만 가능	
Null	null 만 가능	

- 자바스크립트 변수형은 **typeof()** 연산자를 이용해서 확인 가능
  - `typeof(123)` → `"Number"`를 반환
  - `typeof("123")` → `"String"`을 반환

자바스크립트 변수 선언

- 변수를 사전에 선언 없이 사용하는 것이 가능
  - 전역 변수로 사용할때는 미리 선언되어 있어야 함
  - 변수 타입을 고려하지 않고 선언해서 사용하면 되므로 편리
- 별도의 변수 타입이 없으며 **var** 타입 한가지만 제공됨
  - 대신, 변수에 실제로 어떤 값이 저장될 때 그 값에 따라 내부적으로 변수 타입이 정해진다
- 변수의 선언 방식
  - 대소문자를 구분함

```
var 변수명;
  혹은
var 변수명 = 변수값;
```

자바스크립트 변수 선언 예제

```
var index, name = "모바일 웹";
var start = 0, end = 100.0;
var message, condition, sender, receiver;

var  a = "3";
var  b = 2;
c = b + 3 + a;      // c값은 "53"이 됨, 더하기 후 문자열 붙히기 연산
d = a + b;          // d값은 "3"+"2"="32", 문자열연산(Concatenation)
```

연산의 우선순위에서 b+3이 우선

연산의 우선순위에서 a가 우선하므로 b의 값 즉 숫자 2을 문자열 "2"으로 변환한 후 a의 값 "3"과 문자열 붙이기 연산 수행

- 모든 변수 타입에 대해 **var** 타입으로만 선언
  - 문자열, 정수, 실수 등

자바스크립트 기본 연산자

- 숫자 연산
  - 사칙연산자 (+, -, * , /) 제공
  - 증가, 감소 연산자 (++, --) 제공
  - 모든 숫자 연산은 내부적으로 실수 값으로 변환 후 처리
  - 자바 언어와 동일한 수식 기술 방식과 연산 우선순위를 가짐
- 문자열 붙이기 (Concatenation) 연산
  - '+' 연산자를 이용해서 두 문자열을 붙임

```
var first_name = "Steve";
var last_name = "Jobs";

var full_name1 = first_name + " " + last_name;
// full_name1: "Steve Jobs"

var full_name2 = last_name + ", " + first_name;
// full_name2: 'Jobs, Steve'
```

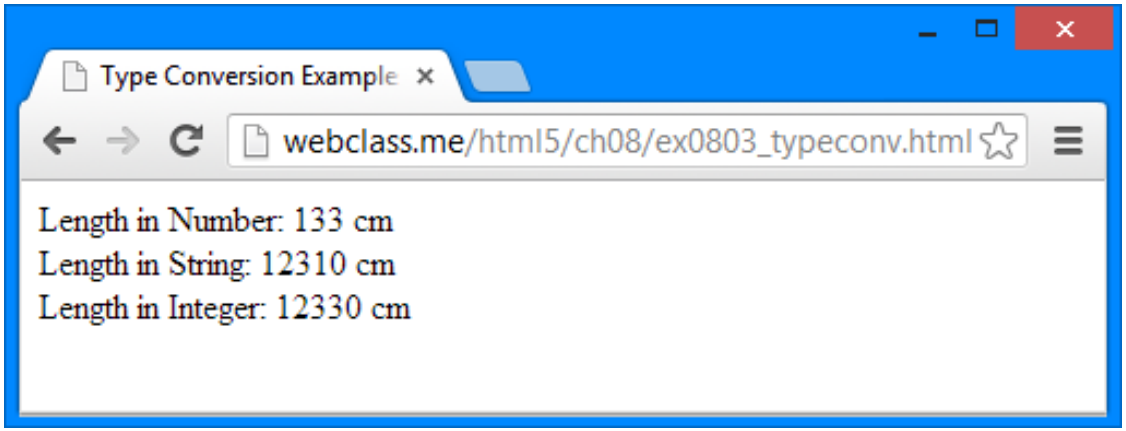
변수 형변환 (type conversion)

- 문자열 타입 → 숫자 타입
  - 문자열 변수를 **`parseInt()`** 혹은 **`parseFloat()`** 함수에 입력
- 숫자 타입 → 문자열타입
  - 숫자 형 변수에 **`toString()`** 메소드를 이용

NOTE: 메소드(Method): 객체에 미리 정의되어 포함되어 있는 함수

변수 형변환 예제

```
var length = 123, length_num, length_str;

length_num = length + 10;
length_str = length.toString() + 10;

document.write("Length in Number: " + length_num + " cm" + '<br />");
document.write("Length in String: " + length_str + " cm" + '<br />");

var num = parseInt(length_str) + 20;
document.write("Length in Integer: " + num + " cm");
```



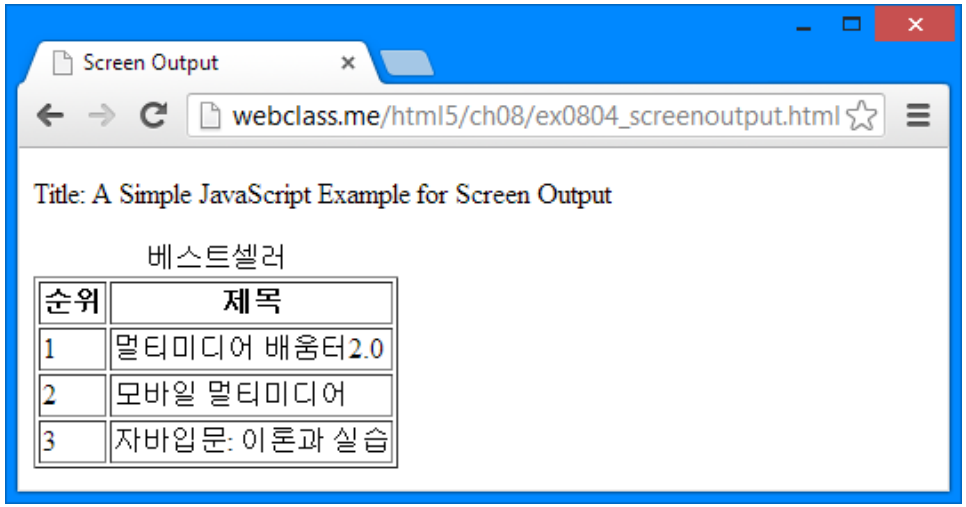
화면 출력

- **document.write()**라는 화면 출력 명령어
  - **document.write()**라는 명령어는 HTML 문서에 콘텐츠 추가
  - 콘텐츠가 삽입된 HTML 문서의 내용이 화면에 출력
    - HTML 태그를 추가할 경우에는 그 태그도 해석되어 화면에 출력

- HTML 문서는 **Document**라는 객체로 모델링 되어 있다
  - **document**라는 이름으로 접근
  - **Document** 객체의 **write()** 메소드

화면 출력 예제

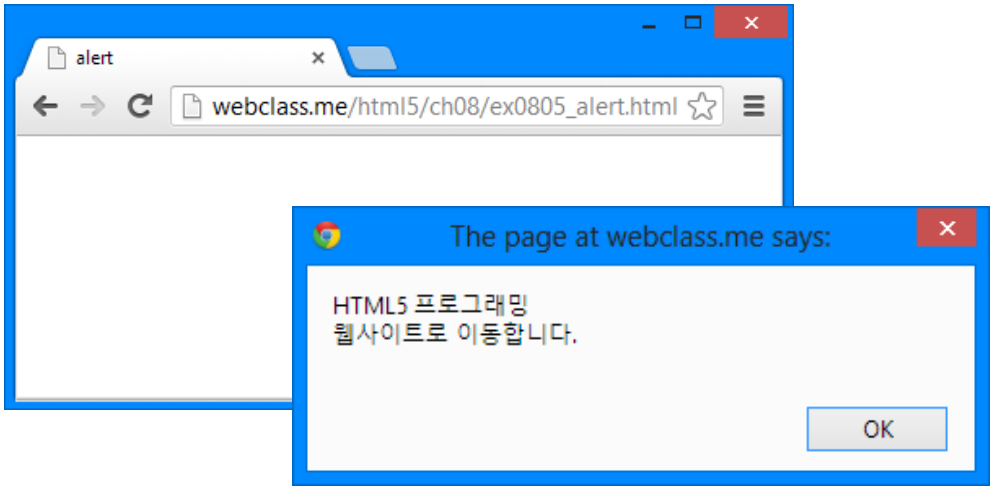
```
var title1 = "멀티미디어 배움터2.0";
var title2 = "모바일 멀티미디어";
var title3 = "자바입문: 이론과 실습";

document.write("<caption> 베스트셀러 </caption>");
document.write("<tr>");
document.write("<th> 순위  </th>");
document.write("<th> 제목 </th>");
document.write("</tr>");
document.write("<tr> <td> 1 </td> <td> " + title1 + " </td> </tr>");
document.write("<tr> <td> 2 </td> <td> " + title2 + " </td> </tr>");
document.write("<tr> <td> 3 </td> <td> " + title3 + " </td> </tr>");
```



대화상자로 메시지 출력

- 대화상자(dialog box)를 만들어 화면에 메시지를 출력하거나 키보드로부터 입력을 받을 수 있는 세가지 방법
- **alert()** 명령어
  - 사용자에게 경고사항이나 메시지를 전달
  - "확인" 버튼을 클릭하지 않으면 다음 자바스크립트 문장이 실행되지 않음

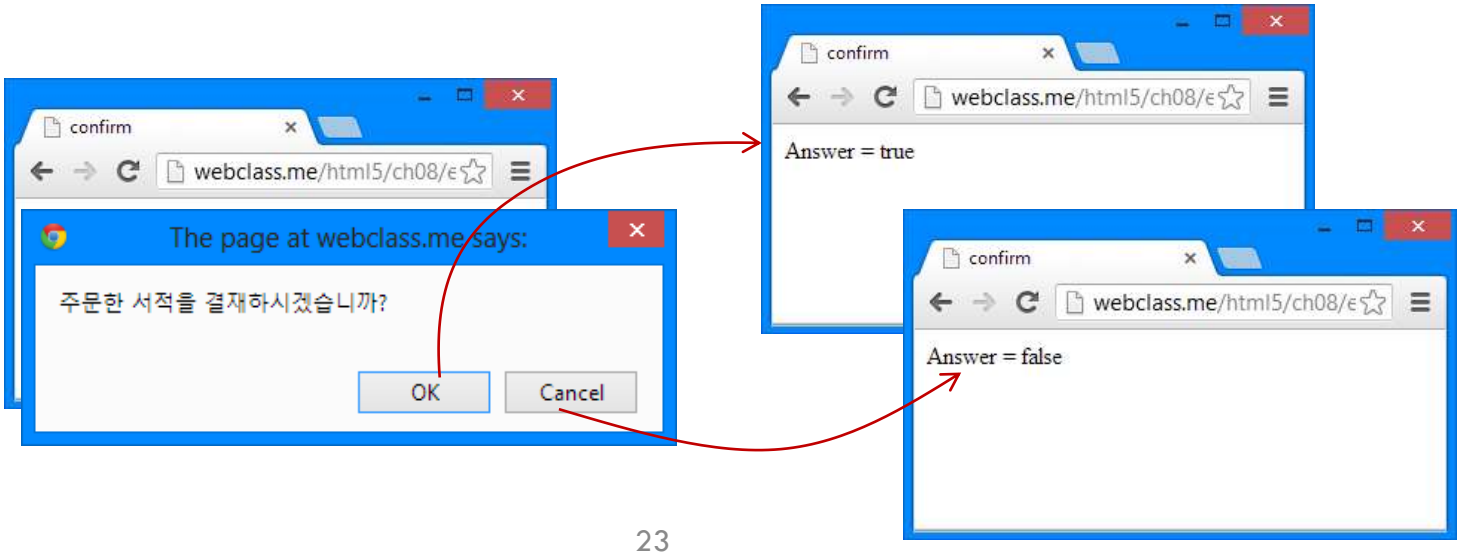
```
alert("HTML5 프로그래밍 \n웹사이트로 이동합니다.");
```



확인 입력 받기

- **confirm()** 명령어
  - 사용자에게 Yes/No 선택을 입력받기 위해 사용하는 방식
  - 대화상자 내에 메시지를 표시하고 "확인"과 "취소" 버튼 표시
  - 버튼을 클릭할 때까지 실행을 대기
  - 확인 버튼을 누르면 **true**, 취소 버튼을 누르면 **false**를 반환

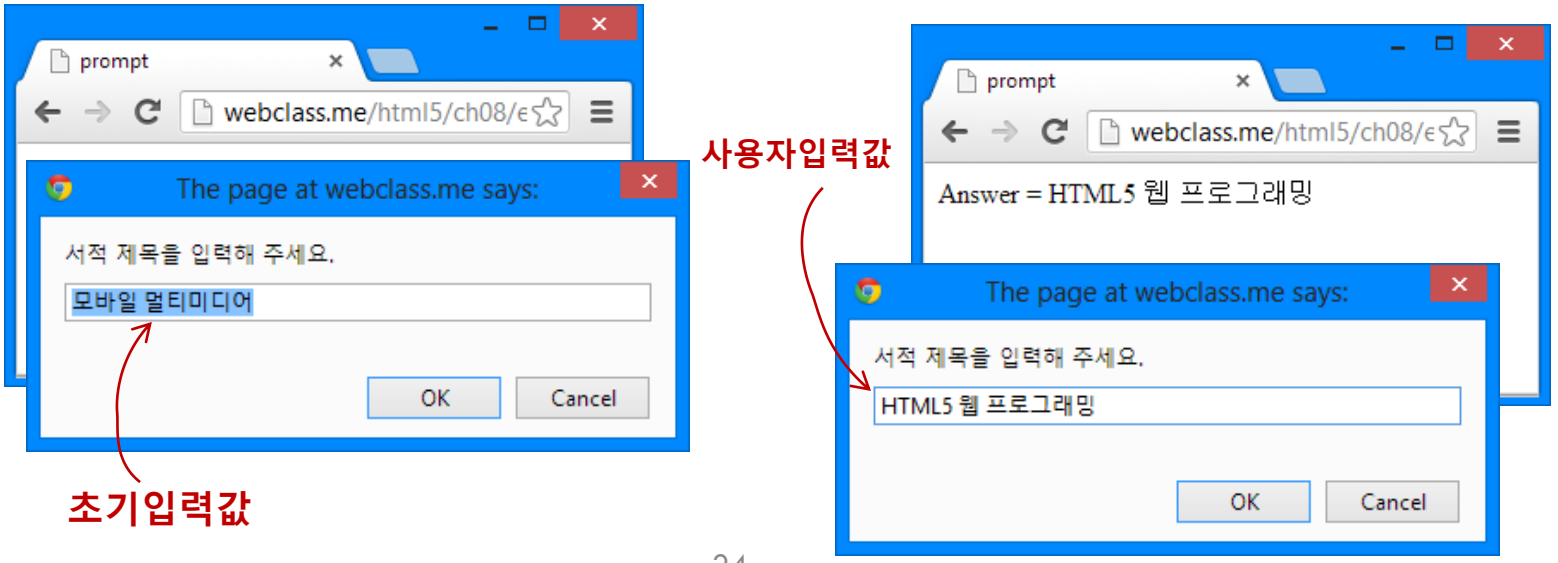
```
var answer = confirm("주문한 서적을 결재하시겠습니까?");
document.write("Answer = " + answer + "<br/>");
```



문자열 입력 받기

- **prompt()** 명령어
  - 사용자로부터 키보드를 통해 문자열을 입력 받는다
  - 대화상자 내에 메시지와 초기입력값이 입력 상자와 함께 표시됨
  - "확인"을 누르면 입력된 문자열, "취소"를 누르면 **null**을 반환

```
var answer = prompt("서적 제목을 입력해 주세요.", "모바일 멀티미디어");
document.write("Answer = " + answer + "<br/>");
```



자바스크립트 제어문

- 제어문으로 **if** 문과 **switch** 문을 제공

자바스크립트 제어문	문법 및 사용 형식	비고

if-then-else	if (조건식) {
	  // 조건식의 값이 true일 때 실행될 문장
	}
	else {
	  // 조건식의 값이 false일 때 실행될 문장
	}
		실행될 문장이 한 개인 경우에는 { }를 생략할 수 있다.

switch	switch (expression) {
	  case value_1:
	  // expression값이 value_1일 때 실행될 문장
	  break;
	  case value_2:
	  // expression값이 value_2일 때 실행될 문장
	  break;
	  case value_3:
	  // expression값이 value_3일 때 실행될 문장
	  break;
	  ...
	  default:
	  // case문에서 찾을 수 없을 때 실행될 문장
	}
		C/C++, 자바 언어와는 달리 (expression)에 정수형 이외의 타입도 사용할 수 있다. 예를 들면 문자열 형식의 값을 사용할 수도 있다.

자바스크립트 반복문

- **while, for, do-while** 문을 제공

자바스크립트 반복문	문법 및 사용 형식	비고

while	while (조건식) {
	// 조건식의 값이 true일 동안 반복해서 실행될 문장
	}
		실행될 문장의 개수가 하나인 경우에는 { }를 생략할 수 있다.

for	for (초기화 문장; 조건식; 증감문) {
	// 조건식의 값이 true일 동안 반복해서 실행될 문장
	}	

do-while	do {
	// 조건식의 값이 true일 동안 반복해서 실행될 문장
	} while (조건식)	

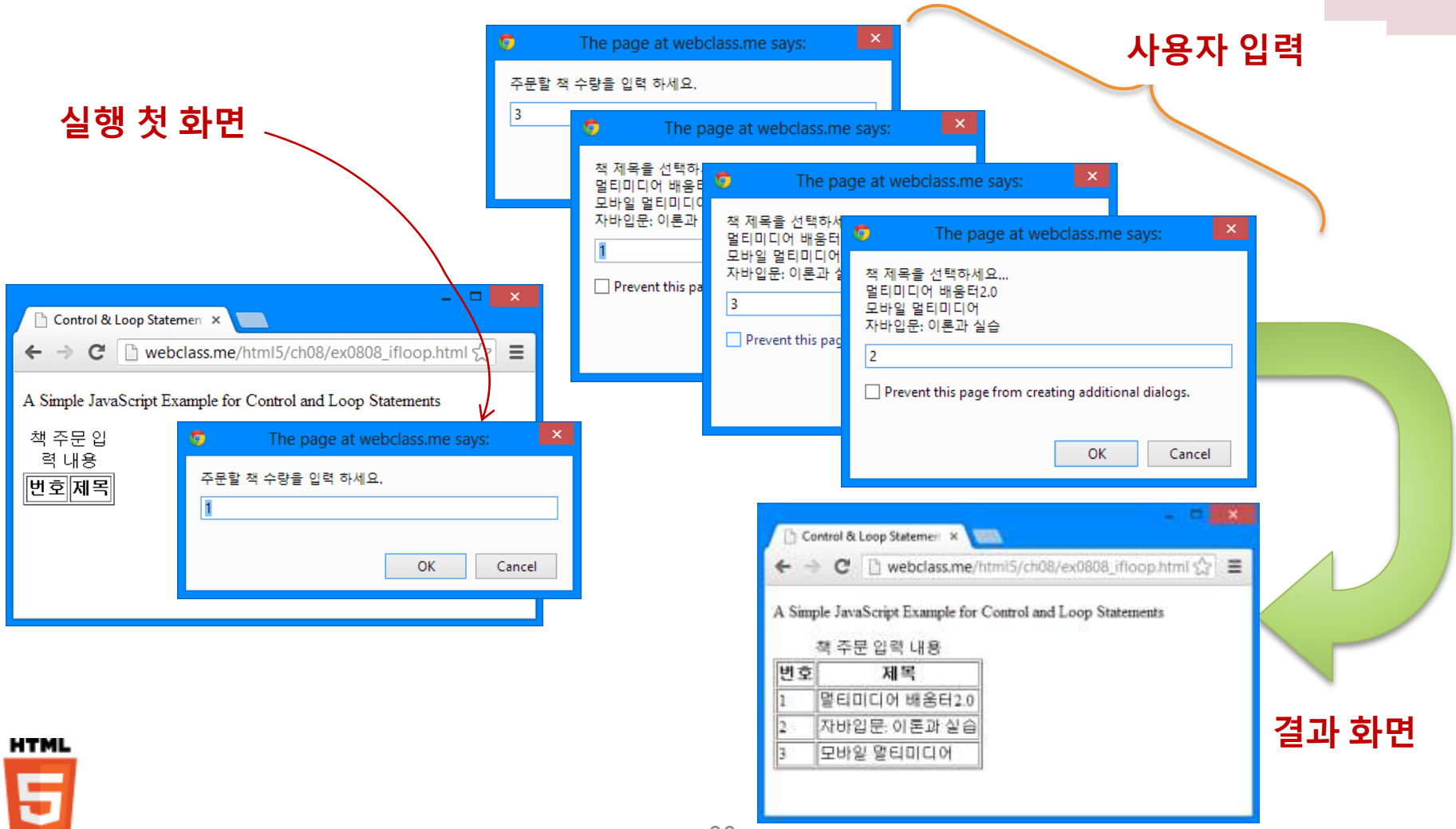
조건문과 반복문 예제

```
document.write("<caption> 책 주문 입력 내용 </caption>");
document.write("<tr>");
document.write("<th> 번호 </th> <th> 제목 </th>");
document.write("</tr>");

var book1 = "멀티미디어 배움터2.0";
var book2 = "모바일 멀티미디어";
var book3 = "자바입문: 이론과 실습";
var order_list = "";
var n = prompt("주문할 책 수량을 입력 하세요.","1");

for(i = 0; i < n; i++) {
  var book_list = book1 + "\n" + book2 + "\n" + book3;
  var choice = prompt("책 제목을 선택하세요...\n" + book_list, "1");

  if (choice == "1")
      title = book1;
  else if (choice == "2")
      title = book2;
  else if (choice == "3")
      title = book3;
  else {
      alert("리스트에 없는 책을 선택하셨습니다.");
      title = "";
  }

  document.write("<tr>");
  document.write("<td>" + (i+1) + "</td>");
  document.write("<td>" + title + "</td>");
  document.write("</tr>");
  order_list += "\n" + title + " ";
}
```

```
switch (choice) {
   case "1":
      title = book1; break;
   case "2":
      title = book2; break;
   case "3":
      title = book3; break;
   default:
      alert("리스트에 없는 책을 선택하셨습니다.");
      title = "";
}
```

switch 문 사용시

조건문과 반복문 예제 실행결과



8.3 자바스크립트 객체다루기

8.3.1 자바스크립트 내장 객체

8.3.2 배열 객체

8.3.3 사용자 정의 객체 생성 및 수정

8.3.4 함수 및 객체 생성자

자바스크립트 객체

- 자바스크립트 객체는 속성 (property)과 메소드 (method)를 가진다
- 객체의 속성 값으로 또 다른 객체를 가질 수 있다
  - 계층적 구조
- 내장 객체와 사용자가 정의한 객체

자바스크립트 내장 객체

- 자바스크립트에서 기본적으로 제공되는 객체
  - **Array, Date, Math, String**
  - 웹 브라우저가 제공하는 **window**와 **navigator** 등은 9장에서 설명
- **Date** 객체
  - 사용자의 컴퓨터에서 제공되는 날짜와 시간을 알아내거나 설정

메소드 이름	기능
getFullYear(), getMonth() getDate(), getDay() getHours(), getMinutes() getSeconds()	사용자 컴퓨터의 시계가 제공하는 현재 시간을 구하는 메소드들이다. 각각 연도, 월, 일, 요일, 시간, 분, 초 값을 리턴한다.
getTime()	1970년 1월 1일 이후 현재까지의 시간을 천분의 1초 단위로 리턴한다.
getTimezoneOffset()	표준시와 현지 시간 간의 표준시차를 분 단위로 리턴한다.
setFullYear(), setMonth() setDate(), setDay() setHours(), setMinutes() setSeconds(), setMillseconds()	사용자 컴퓨터의 시계을 설정하기 위한 메소드들이다. 각각 연도, 월, 일, 요일, 시간, 분, 초, 천분의 1초 값을 설정하는 메소드 들이다.

Math 객체

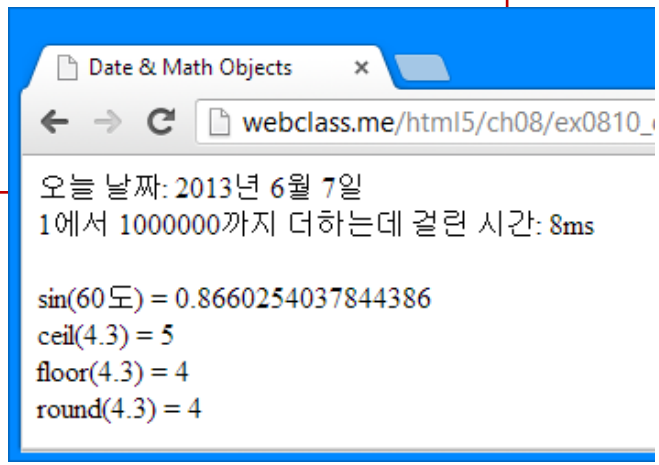
- 수학 계산을 위해 기본적으로 제공되는 객체
  - 별도의 선언이나 생성없이 바로 사용 가능
  - 상수값은 **Math** 객체의 속성으로 제공되며 주요 수학 함수는 **Math** 객체의 메소드로 제공됨

속성 이름	설명
E	Euler 상수 값 (약 2.718)
LN2	자연로그2, log2 (약 0.693)
LN10	자연로그10, log10 (약 2.302)
LOG2E	$\log_2 e$ (약 1.442)
LOG10E	(약 0.434)
PI	원주율 π (약 3.14)
SQRT1_2	$\sqrt{1/2}$ (약 0.707)
SQRT2	$\sqrt{2}$ (약 1.414)

메소드 이름	기능
cos(), sin(), tan()	삼각함수 코사인, 사인, 탄젠트 함수를 제공한다.
acos(), asin(), atan()	코사인, 사인, 탄젠트 함수의 역함수를 제공한다.
ceil(), floor(), round()	각각 올림, 내림, 반올림 값을 리턴한다.
max(), min(), abs()	입력 인자 값들 중 최대, 최소, 절대값을 리턴한다.
sqrt(x), pow(x,y)	각각 $\sqrt{x}$ 와 x^y 값을 리턴한다.
log(x), exp(x)	각각 $\log_e x$ 와 e^x 값을 리턴한다.

Date와 Math 객체 예제

```
<script  type = "text/javascript" >
    var today = new Date();
    var y = today.getFullYear();
    var m = today.getMonth() + 1;
    var d = today.getDate();

    document.write("오늘 날짜: " + y + "년 " + m + "월 " + d + "일<br/>");

    var start = new Date();
    var t1 = start.getTime();

    var sum = 0;
    for(i=0;i<1000000;i++) {
        sum = sum + i;
    }

    var end = new Date();
    var t2 = end.getTime();

    document.write("1에서 1000000까지 더하는데 걸린 시간: " + (t2 - t1) + "ms<br/>");
    document.write("<br/>");
    document.write("sin(60도) = " + Math.sin(Math.PI/3) + "<br/>");
    document.write("ceil(4.3) = " + Math.ceil(4.3) + "<br/>");
    document.write("floor(4.3) = " + Math.floor(4.3) + "<br/>");
    document.write("round(4.3) = " + Math.round(4.3) + "<br/>");
</script>
```



배열 객체

- 데이터 요소 여러 개를 묶어서 처리하고자 할 때 배열(array) 데이터 구조가 적합
- 자바스크립트 배열의 특징
  - 배열의 각 요소가 동일한 데이터 타입을 가지지 않아도 된다
    - 배열의 요소는 다양한 타입의 객체를 가질 수 있다
    - 예) 하나의 배열에 숫자 형이나 문자열 요소를 동시에 가질 수 있다
  - 배열의 크기가 언제라도 증가, 감소가 가능
    - 자바스크립트의 변수형의 자동 형변환과 객체의 동적 속성 추가 특징에 따른 장점

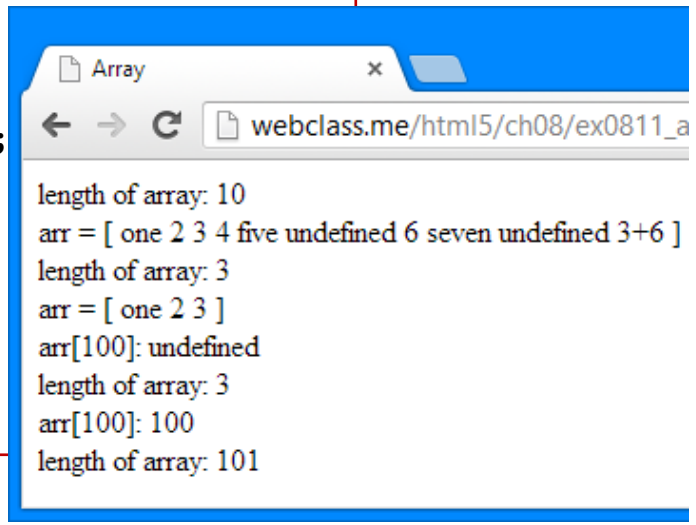
배열의 생성 및 접근

- 배열의 생성
  - **new** 연산자를 이용하거나 배열 리터럴을 이용해 생성
- 배열 요소의 접근
  - **배열이름[인덱스]**와 같이 각괄호 ([ ])를 이용해 접근

```
1  var book_arr = new Array("멀티미디어 배움터2.0", "생능출판사",
2                  "최윤철, 임순범", 25000, 442);
3
4  // 배열의 내용
5  // book_arr[0]: "멀티미디어 배움터2.0"
6  // book_arr[1]: "생능출판사 "
7  // book_arr[2]: "최윤철, 임순범"
8  // book_arr[3]: 25000
9  // book_arr[4]: 442
10
11
12 var book_arr2 = ["멀티미디어 배움터2.0", "생능출판사", "최윤철, 임순범",
13             25000, 442];
14
15 var arr100 = new Array(100);     // 요소 갯수가 100인 배열 생성
```

배열의 사용 예제

```
1  var arr = new Array("one", 2, "3", 4, "five");
2  // arr 내용 = ["one", 2, "3", 4, "five"]
3
4  arr[6] = 6;
5  arr[7] = "seven";
6  arr[9] = "3+6";
7  // arr 내용: ["one", 2, "3", 4, "five", undefined, 6, "seven", undefined, "3+6"]
8
9  document.write("length of array: " + arr.length + "<br/>");
10
11 document.write("arr = [");
12 for(i=0;i<arr.length;i++) document.write(" " + arr[i] + " ");
13 document.write("] <br/> ");
14
15 arr.length = 3;
16 document.write("length of array: " + arr.length + "<br/>");
17
18 document.write("arr = [");
19 for(i=0;i<arr.length;i++) document.write(" " + arr[i] + " ");
20 document.write("] <br/> ");
21
22 document.write("arr[" + 100 + "]: " + arr[100] + "<br/>");
23 document.write("length of array: " + arr.length + "<br/>");
24
25 arr[100] = 100;
26 document.write("arr[" + 100 + "]: " + arr[100] + "<br/>");
27 document.write("length of array: " + arr.length + "<br/>");
```



배열 객체의 메소드

메소드 이름	기능
reverse()	배열 내 요소들의 순서를 반대로 바꾸는 기능이다.
sort()	배열 내 요소들의 순서를 오름차순으로 정렬하는 기능이다. 숫자가 문자보다 앞선다.
join()	배열 내 요소를 모두 합쳐서 하나의 문자열로 만들어준다. 이때 요소 사이에 끼워 넣을 문자열을 지정할 수 있다.
concat()	배열의 뒤에 요소를 붙혀서 (concatenation) 배열의 내용을 추가하는 기능이다.
slice()	배열의 요소들 중 일부만을 배열로 만들어서 리턴하는 기능. 사용 형식은array.slice(첫 요소 index, 마지막 요소 index + 1) 과 같다..

사용자 정의 객체

- 사용자 정의 객체 생성
  - **Object** 생성자와 **new** 명령어를 이용해 생성

```
var book = new Object();
```

    - 아직 아무런 속성을 가지지 않는 빈(blank) 객체 생성
  - 객체 생성 후 속성 및 메소드를 언제라도 추가 가능
  - 점(dot, ".") 연산자를 붙혀서 속성과 메소드 접근

```
var book = new Object();
book.title = "멀티미디어 배움터2.0";
book.publisher = "생능출판사";
book.author = "최윤철, 임순범";
book.price = 25000;
book.pages = 442;
```

사용자 정의 객체 생성

- 초기화를 통한 객체 생성

```
var book = {title: " 멀티미디어 배움터2.0", publisher: " 생능출판사",
        author: " 최윤철, 임순범 ", price: 25000, pages: 442};
```

- 다양한 변수 형이 속성으로 사용 가능
  - 예) 문자열과 숫자형이 동시에 사용 가능
- 객체의 계층적 구조
  - 객체의 속성 값으로 또 다른 객체를 가짐

```
var book = new Object();
book.title = "멀티미디어 배움터2.0";
book.publisher = "생능출판사";
book.author = "최윤철, 임순범";
book.price = 25000;

book.info = new Object();
book.info.pages = 442;
book.info.date = "2010년 1월 30일";
book.info.ISBN10= "8970506470";
book.info.size = "188mm*254mm";
```

객체의 접근

- 객체의 속성과 메소드의 접근 방식
  - 점(dot, ".") 연산자를 이용
  - 배열 표시 방식("[ ]")
- 속성을 삭제하기 위해서는 **delete**라는 명령어 이용

```
// 객체의 속성 접근 방법
var property1 = book.title;
var property2 = book.info.price;
// 혹은
var property3 = book["title"];
var property2 = book.info["price"];

// 객체의 속성 삭제 방법
delete book.title;
delete book.info.price;
```

개선된 for 문

- 객체에 포함된 속성의 갯수나 이름을 모르더라도 객체내의 모든 속성을 접근할 수 있는 방법

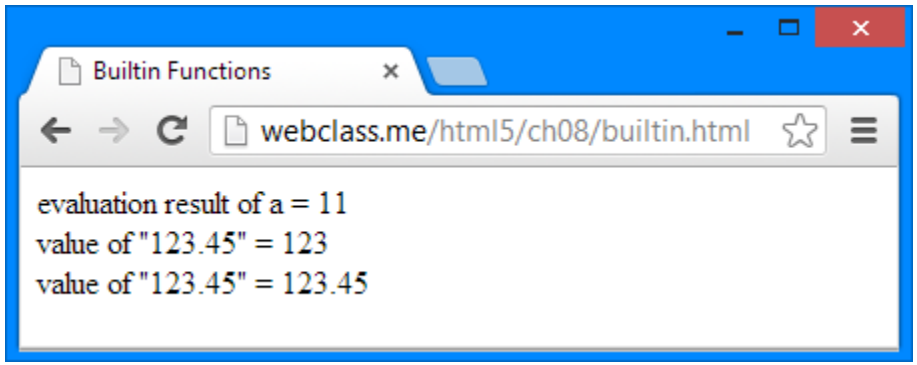
```
// 개선된 for 문을 이용한 객체의 속성 접근 방법
for (var p in book) {
    document.write(     "Property name: " + p +
            " Property value: " + book[p] + "<br/>");
}
```

- 개선된 for 문 안에서 객체 접근 방식은 점(".")에 의한 접근은 불가능
- 대신, 배열 방식("[ ]")을 이용해야 함
  - 속성의 이름을 모르기 때문에 속성 이름을 직접 지정해야 하는 점(".") 접근 방식은 사용할 수 없기 때문

자바스크립트 함수

- 자바스크립트 내장 함수
  - **eval()**
    - 문자열 입력을 계산하여 결과를 반환하는 함수
  - **parseInt(), parseFloat()**
    - 문자열 값을 각각 정수와 실수로 변환하는 함수

```
var a = eval("1+2*3+4");
document.write("evaluation result of a = " + a + "<br/>");
document.write("value of \"123.45\" = " + parseInt("123.45") + "<br/>");
document.write("value of \"123.45\" = " + parseFloat("123.45") + "<br/>");
```



자바스크립트 사용자 정의 함수

```
// 함수의 선언 규칙
function function_name (함수의 인수들) {
 // 함수의 명령문 들
}


// 함수의 사용 예
function print_value(name, v) {
     document.write("Name: " + name + ", ");
  document.write("Value: " + v + "<br/>");
 }
```

- 일반 프로그래밍 언어의 함수와의 차이점
  - 매개변수와 인수의 변수형이 동일한지 검사하지 않는다
  - 매개변수의 갯수와 함수의 인수의 갯수가 같은지 확인하지 않는다
    - 만약, 매개 변수의 갯수가 함수의 인수의 갯수보다 적다면 인수의 값은 **undefined**로 설정됨