Chapter 09

자바스크립트 기본 문법

Contents

학습목표

- 자바스크립트의 역할을 설명할 수 있다.
- 자바스크립트 코드를 HTML5 문서에 포함하는 방법을 설명할 수 있다.
- 자바스크립트 기본 문법을 알고 이를 활용하여 코드를 작성할 수 있다.
- 연산자와 제어문의 종류를 알고 이를 활용하여 코드를 작성할 수 있다

1. 자바스크립트의 역할

- **자바스크립트**
  - 웹 문서를 동적으로 제어하기 위해 고안된 프로그래밍 언어

- **웹 삼총사**
  - HTML: 모델 담당
  - CSS: 뷰 담당
  - 자바스크립트: 제어 담당

그림 9-1 웹 삼총사

1. 자바스크립트의 역할

- **자바스크립트의 역할**

① 요소의 추가 및 삭제

② CSS 및 HTML 요소의 스타일 변경

③ 사용자와의 상호작용

④ 폼의 유효성 검증

⑤ 마우스와 키보드 이벤트에 대한 스크립트 실행

⑥ 웹 브라우저 제어 및 쿠키 등의 설정과 조회

⑦ AJAX 기술을 이용한 웹 서버와의 통신

2. 자바스크립트 작성 방법

- **대소문자 구분하여 작성**

- **문장은 세미콜론(;)으로 구분**

바른 예	var age=25 document.write("당신의 나이는 " + age + "입니다.")
	var age=25; document.write("당신의 나이는 " + age + "입니다.");
	var age=25; document.write("당신의 나이는 " + age + "입니다.");
잘못된 예	var age=25 document.write("당신의 나이는 " + age + "입니다.")

- **큰따옴표(" ")와 작은따옴표(' ')를 구분하여 사용**

바른 예	document.write(" 자바스크립트 학습 ");
	document.write(' 자바스크립트 학습 ');
잘못된 예	document.write(" 자바스크립트 학습 ")

3. 자바스크립트 포함 방법

- **HTML 문서 내부에 코드를 작성하는 방법**

① <head> 태그 또는 <body> 태그 내에 코드 작성

```
<head>
    <meta charset="utf-8"/>
    <title>자바스크립트 예제</title>
    <script>
        // 자바스크립트 코드 작성
    </script>
</head>
<body>
    <script>
        // 자바스크립트 코드 작성
    </script>
</body>
```

② HTML 태그 안에 속성값으로 정의

```
<button type="button" onclick="alert('자바스크립트')">버튼 클릭</button>
```

3. 자바스크립트 포함 방법

예제 9-1 자바스크립트 코드의 실행 순서 살펴보기 ch09/01_js.html

```
<head>
    <meta charset="utf-8"/>
    <title>자바스크립트 예제</title>
    <script>
        var num=0;
        document.write("head 태그 내 실행 순서 : " + num + "<br/>");
    </script>
    <script>
        var num=1;
        document.write("head 태그 내 실행 순서 : " + num + "<br/>");
    </script>
</head>
<body>
    <script>
        var num=2;
        document.write("body 태그 내 실행 순서 : " + num + "<br/>");
    </script>
    <script>
        var num = 3;
        document.write("body 태그 내 실행 순서 : " + num + "<br/>");
    </script>
</body>
```

```
head 태그 내 실행 순서 : 0
head 태그 내 실행 순서 : 1
body 태그 내 실행 순서 : 2
body 태그 내 실행 순서 : 3
```

3. 자바스크립트 포함 방법

- **별도로 작성한 후 HTML 문서에서 참조하는 방법**
  - 외부 자바스크립트 파일을 만든 후 HTML 문서의 <script> 태그에 src 속성을 추가하여 참조

표 9-3 자바스크립트 파일 위치에 따른 src 속성값

위치	src 속성값
HTML 문서와 같은 디렉터리에 있는 경우	〈script src="myscript.js"〉〈/script〉
HTML 문서와 다른 디렉터리에 있는 경우	〈script src="./ejs/myscript.js"〉〈/script〉
HTML 문서와 다른 서버 디렉터리에 있는 경우	〈script src="http://www.hanbit.co.kr/jsfile/myscript.js"〉〈/script〉

<자바스크립트 파일을 외부에서 작성했을 때의 장점>

- 자바스크립트 파일을 HTML 문서와 분리하여 관리할 수 있음
- 자바스크립트 코드를 관리, 유지보수, 디버깅하기 쉬움
- 자바스크립트 코드의 보안성과 안전성을 높일 수 있음

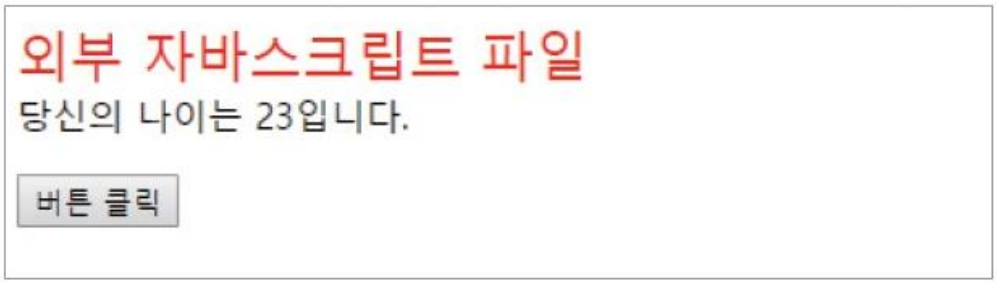
3. 자바스크립트 포함 방법

예제 9-2 외부 자바스크립트 문서 작성 후 참조하기 ch09/ejs.js

```
var age=23;
/* 문자에 스타일 속성 적용 */
document.write("<div style='color: red; font-size: 24px;'>외부 자바스크립트 파일</div>");
document.write("당신의 나이는 " + age + "입니다.");
```

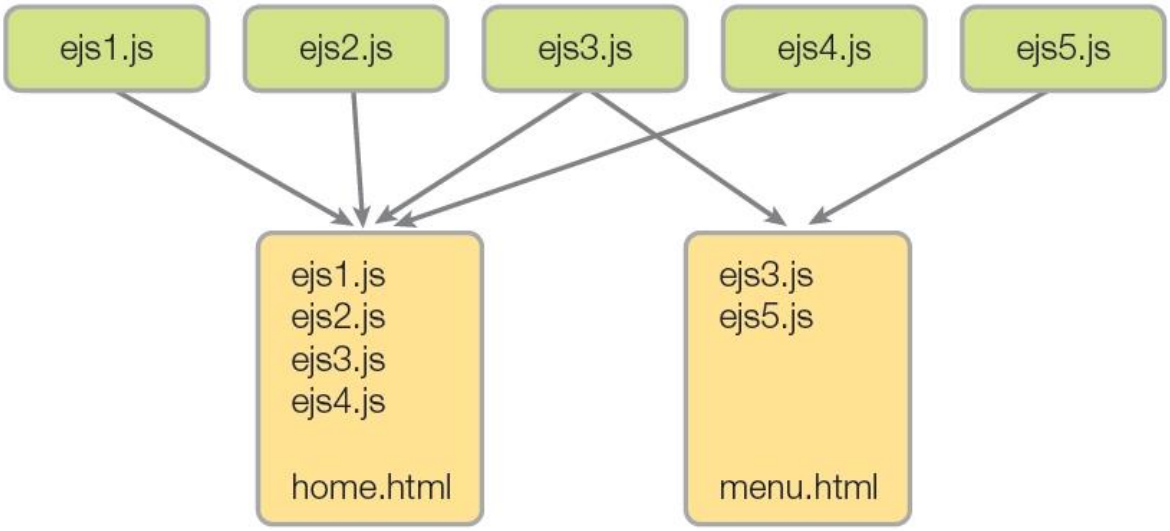
c09/02_js.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <script src="./ejs/ejs.js"> </script>
</head>
<body>
    <p>
    <!-- 버튼을 클릭하면 메시지 창 출력 -->
    <button type="button" onclick="alert('외부 자바스크립트 파일')">버튼 클릭</button>
    </p>
</body>
</html>
```

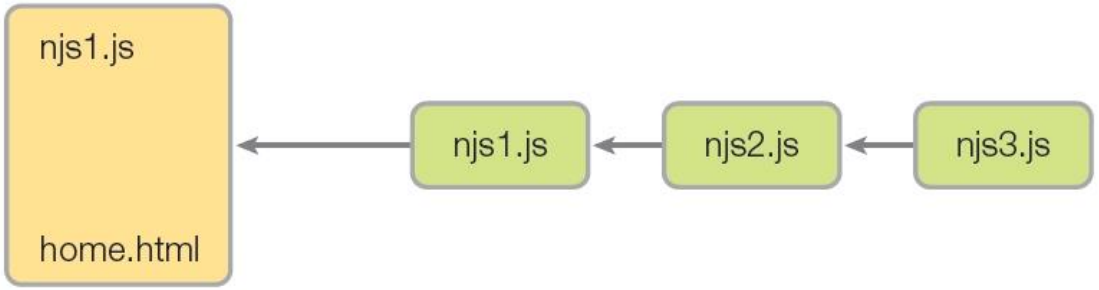


3. 자바스크립트 포함 방법

- **외부 자바스크립트 파일 참조 방법**



(a) 기능 단위로 분리된 여러 개의 자바스크립트 파일 참조



(b) 내포 관계를 가진 자바스크립트 파일 참조

3. 자바스크립트 포함 방법

예제 9-3 여러 개의 외부 자바스크립트 파일 참조하기 ch09/ejs1.js

```
document.write("ejs1.js");
document.write("<div style='color: red; font-size: 24px;'>외부 자바스크립트 파일</div>");
```

ch09/ejs2.js

```
document.write("ejs2.js");
document.write("<div style='color: blue; font-size: 20px;'>외부 자바스크립트 파일</div>");
```

ch09/ejs3.js

```
document.write("ejs3.js");
document.write("<div style='color: green; font-size: 16px;'>외부 자바스크립트 파일</div>");
```

ch09/03_js.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <script src="./ejs/ejs1.js"> </script>
</head>
<body>
    <script src="./ejs/ejs2.js"> </script>
    <script src="./ejs/ejs3.js"> </script>
</body>
</html>
```

ejs1.js
외부 자바스크립트 파일
ejs2.js
외부 자바스크립트 파일
ejs3.js
외부 자바스크립트 파일

3. 자바스크립트 포함 방법

예제 9-4 내포 관계인 자바스크립트 파일 참조하기 — ch09/njs1.js

```
document.write("njs1.js");
document.write("<div style='color: red; font-size: 24px;'>외부 자바스크립트 파일</div>");
document.write("<script src='./ejs/njs2.js'> </script>");
```

ch09/njs2.js

```
document.write("njs2.js는 njs1.js에 포함");
document.write("<div style='color: blue; font-size: 20px;'>외부 자바스크립트 파일</div>");
document.write("<script src='./ejs/njs3.js'> </script>");
```

ch09/njs3.js

```
document.write("njs3.js는 njs2.js에 포함");
document.write("<div style='color: green; font-size: 16px;'>외부 자바스크립트 파일</div>");
alert('Nested Script File');
```

ch09/04_js.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <script src="./ejs/njs1.js"> </script>
</body>
</html>
```

3. 자바스크립트 포함 방법

● 혼합 방법

예제 9-5 혼합 방법으로 자바스크립트 파일 포함하기 ch09/mjs.js

```
document.write("mjs1.js");
document.write("<div style='color: red; font-size: 24px;'>외부 자바스크립트 파일</div>");
```

ch09/05_js.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <script src="./ejs/mjs.js"> </script>
</head>
<body>
    <script>
        document.write("<div style='color: blue; font-size: 20px;'>내부 자바스크립트</div>");
    </script>
</body>
</html>
```

mjs1.js
외부 자바스크립트 파일
내부 자바스크립트

1. 데이터 타입

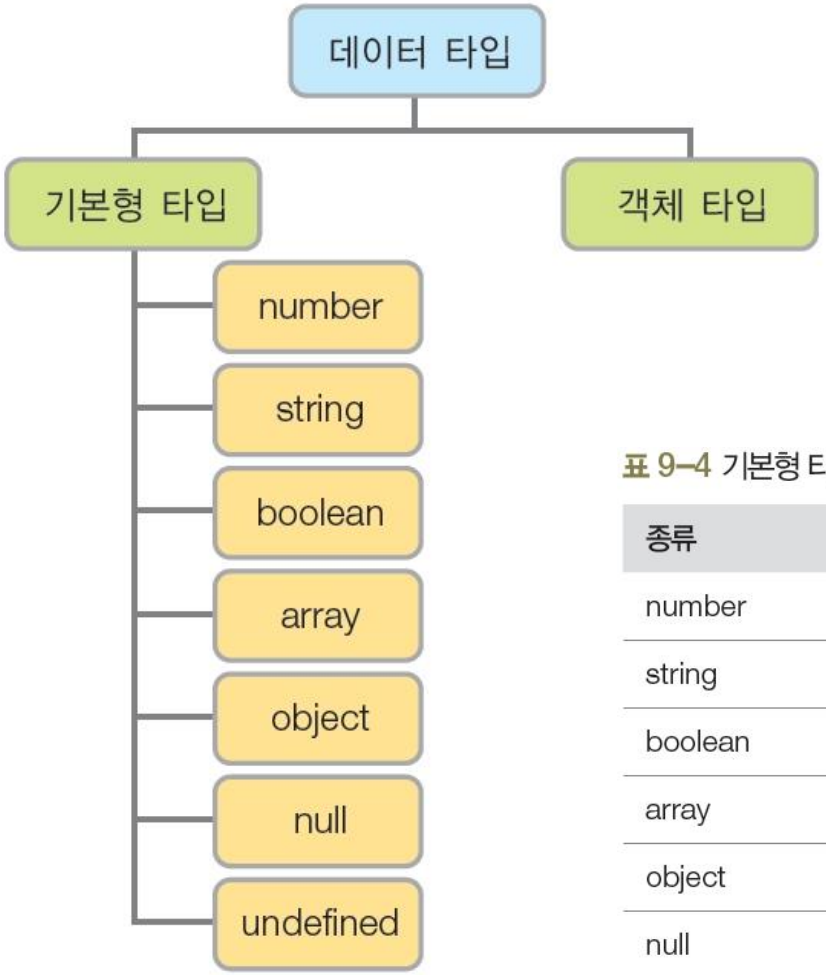


그림 9-4 자바스크립트의 데이터 타입

표 9-4 기본형 타입의 종류

종류	설명	사용 예
number	정수 혹은 실수	100, 10.5, 10e+3
string	문자 혹은 문자열	"홍길동", '홍길동'
boolean	참 혹은 거짓	true, false
array	데이터의 집합(배열, 객체로 취급)	["서울", "부산", "인천"]
object	데이터 속성과 값으로 이루어진 집합	{name: '홍길동', age: 25}
null	객체 값이 없음	null
undefined	데이터 값이 정해지지 않음	undefined

1. 데이터 타입

예제 9-6 typeof 연산자를 사용하여 데이터 타입 확인하기 ch09/06_js.html

```
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <script>
        var num;        // 변숫 값이 없음
        var obj=null;   // 객체 변숫 값이 없음
        document.write(typeof 100+"<br>");
        document.write(typeof 10.5+"<br>");
        document.write(typeof "홍길동"+"<br>");
        document.write(typeof true+"<br>");
        document.write(typeof [1,2,3]+"<br>");
        document.write(typeof {name:'홍길동', age:25}+"<br>");
        document.write(typeof num+"<br>");
        document.write(typeof obj+"<br>");
   </script>
</body>
```

number
number
string
boolean
object
object
undefined
object

2. 변수명 규칙

- **변수명 작성 규칙**
  - 문자, 밑줄(_), 달러 기호($)로 시작
  - 대소문자 구별('변수 A'와 '변수 a'는 서로 다른 변수)
  - 한글은 사용 가능하나 영문자 사용 권장
  - 자바스크립트에서 정한 예약어는 변수명으로 사용 불가능

표 9-5 자바스크립트 예약어

abstract	Arguments	boolean	break	byte
case	catch	char	class	const
continue	debugger	default	delete	do
double	else	enum	eval	export
extends	false	final	finally	float
for	function	goto	if	implements
import	in	instanceof	int	interface
let	long	native	new	null
package	private	protected	public	return
short	static	super	switch	synchronized
this	throw	throws	transient	true
try	typeof	var	void	volatile
while	with	yield		

3. 변수 사용 방법

● 변수 사용 예

- var x; // 변수 x 선언
- var y=10; // 변수 y 선언 및 초깃값 할당
- var x=y; // 변수 y의 값을 변수 x에 저장
- var a, b, c; // 변수 a, b, c 선언
- var a=10, b=11, c=12; // 변수 a, b, c 선언 및 각각 다른 초깃값 할당
- var a=b=c=10; // 변수 a, b, c 선언 및 같은 초깃값 할당
- var name="홍길동", age=25; // 변수 name, age 선언 및 각각 다른 초깃값 할당
- var total=a+b+c; // 변수 a, b, c 값을 더한 결과를 변수 total에 저장

3. 변수 사용 방법

- **변수 사용 시 문법적으로 오류가 발생한 사례**

```
- var 7num=100;          // 숫자로 시작하는 변수명 잘못 사용
- var &num=100;          // 특수 문자로 시작하는 변수명 잘못 사용
- var true=1;            // 예약어를 변수명으로 잘못 사용
- var 10=x;              // 좌변에 상수값 잘못 선언
- var a+b=20;            // 좌변에 연산식 잘못 선언
- var "홍길동"=name;      // 좌변에 문자열값 잘못 선언
- var get Number=100;    // 변수명 사이에 공백(space) 잘못 선언
- var a, b, c=100;       // 콤마로 구분한 변수명 잘못 선언
```

3. 변수 사용 방법

예제 9-7 변수의 데이터 타입 변경하기 ch09/07_js.html

```
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <script>
        var num=10;
        document.write("num 변수 : " + typeof num + " 타입<p/>");
        document.write("--- 데이터 값 변경 후 ---<p/>");
        var num="홍길동";
        document.write("num 변수 : " + typeof num + " 타입<p/>");
   </script>
</body>
```

num 변수 : number 타입

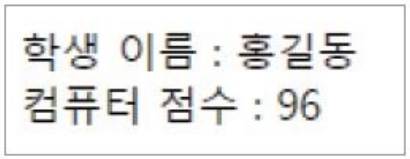
--- 데이터 값 변경 후 ---

num 변수 : string 타입

3. 변수 사용 방법

예제 9-8 변수명 재선언 시 데이터 값 변화 살펴보기 Ch09/08_js.html

```
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <script>
        stdName="홍길동";     // var 키워드 생략
        comGrade=96;          // var 키워드 생략
        var stdName;          // 변수명 재선언
        var comGrade;         // 변수명 재선언
        document.write("학생 이름 : " + stdName + "<br>");
        document.write("컴퓨터 점수 : " + comGrade + "<br>");
   </script>
</body>
```



4. 전역 변수와 지역 변수

- **변수의 메모리 수명**
  - 변수가 생성되어 역할을 다한 후 해제되기까지의 주기

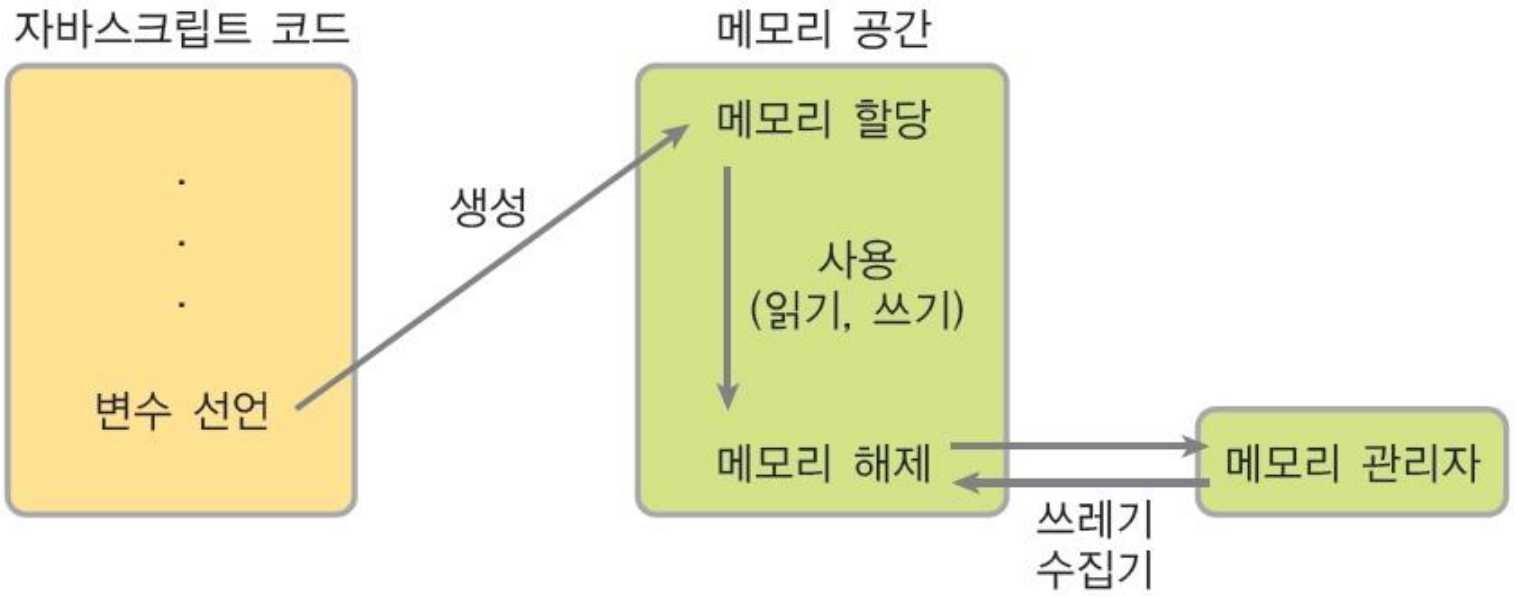


그림 9–5 변수의 메모리 수명

4. 전역 변수와 지역 변수

- **전역 변수**
  - 코드 내 어느 위치에서든 선언하여 전 영역에서 사용할 수 있는 변수
- **지역 변수**
  - 변수가 선언된 해당 블록에서 선언하여 범위 내에서만 유효하게 사용할 수 있는 변수

```
<script>
    var globValue1;     // 전역 변수 선언
        globValue2;     // 전역 변수 선언, var 생략

    function 함수() {
        var locValue;   // 지역 변수 선언
            globValue;  // 함수 내부에서 var 생략 시 자동 전역 변수로 선언
        locValue=10;    // 지역 변수 사용
    }

    globValue=10;       // 전역 변수 사용
</script>
```

그림 9-6 전역 변수와 지역 변수 선언

4. 전역 변수와 지역 변수

예제 9-9 전역 변수와 지역 변수 이해하기 1 ch09/09_js.html

```
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <script>
        function getGrade() {    // 함수 정의
        var kor=95;          // 지역 변수
        }
        var kor=100;            // 전역 변수
        getGrade();             // 함수 호출
        document.write("국어 점수 : " + kor + "<br>");
   </script>
</body>
```

국어 점수 : 100

예제 9-10 전역 변수와 지역 변수 이해하기 2 ch09/10_js.html

```
<script>
    function getGrade() {    // 함수 정의
        kor=95;              // 자동 전역 변수
    }
    var kor=100;             // 전역 변수
    getGrade();              // 함수 호출
    document.write("국어 점수 : " + kor + "<br>");
</script>
```

국어 점수 : 95

4. 전역 변수와 지역 변수

예제 9-11 전역 변수와 지역 변수 이해하기 3 ch09/11_js.html

```
<script>
    function getGrade() {  // 함수 정의
        var kor=95;        // 지역 변수
    }
    getGrade();            // 함수 호출
    document.write("지역 변수 값은 함수 외부에서 사용할 수 없습니다.<br>");
    document.write("국어 점수 : " + kor + "<br>");
</script>
```

지역 변수 값은 함수 외부에서 사용할 수 없습니다.

예제 9-12 전역 변수와 지역 변수 이해하기 4 ch09/12_js.html

```
<script>
    function getGrade() {   // 함수 정의
    var kor=95;          // 지역 변수
    return kor;
    }
    getKor=getGrade();      // 함수 호출
    document.write("국어 점수 : " + getKor + "<br>");
</script>
```

국어 점수 : 95

0. 개요

- **연산자**
  - 피연산자에게 연산 명령을 내리기 위해 사용하는 기호

- **연산자의 종류**

연산자	기호
문자열 연산자	+(문자열 연결)
산술 연산자	++(증가 연산), --(감소 연산), *(곱셈), /(나눗셈), %(나머지), +(덧셈), -(뺄셈)
비교 연산자	<(작다), <=(작거나 같다), >(크다), >=(크거나 같다), ==(값이 같다), !=(값이 다르다), ===(값과 타입 모두 같다), !==(값 또는 타입이 다르다)
논리 연산자	&(비트 AND), |(비트 OR), ^(비트 XOR), &&(논리 AND), ||(논리 OR)
조건 연산자	(판단) ? true : false;
대입 연산자	=, +=, -=, *=, /=, %=, <<=, >>=, >>>=, &=, |=, ^=

1. 문자열 연산자

- **문자열 연산자**
  - '+' 기호를 사용하여 문자열을 연결

```
var st="Hello"+"Javascript";     // 연산 결과 "Hello Javascript"가 출력
```

```
var st="100"+10;  // 문자열 연산 결과 "10010"이 출력
var st=100+10;    // 산술 연산 결과 110이 출력
```

2. 산술 연산자

- **산술 연산자**
  - 사칙 연산을 수행
  - 종류 : 더하기(+), 빼기(-), 곱하기(*), 나누기(/), 나머지(%), 증감(++), 감소(--)
    - 나머지(%) : 나눗셈 결과 나머지 값을 구함
    - 증가(++) : 변숫값을 증가시킴
    - 감소(--) : 변숫값을 감소시킴

예제 9-13 다양한 산술 연산자 활용하기 ch09/13_js.html

```
<script>
    var incData=1;
    var decData=5;
    var r1=r2=0;
    r1=15%6;    // 나머지 연산
    document.write("15%3 = " + r1 +"<br>");
    document.write("incData++ = "+ incData++ +"<br>");    // 후위 증가
    document.write("++incData = "+ ++incData +"<br>");    // 전위 증가
    document.write("decData-- = "+ decData-- +"<br>");    // 후위 감소
    document.write("--decData = "+ --decData +"<br>");    // 전위 감소
    r2=incData*decData;    // 곱셈 연산
    document.write("incData*decData = "+ r2 +"<br>");
</script>
```

```
15%3 = 3
incData++ = 1
++incData = 3
decData-- = 5
--decData = 3
incData*decData = 9
```

3. 비교 연산자

- **비교 연산자**
  - 두 피연산자의 값을 비교하여 참(true) 또는 거짓(false) 값을 반환

표 9-8 비교 연산자의 사용 예

비교 연산자	설명	사용 예	결과
==	값이 같은지 비교한다.	x=="5"	true
===	값과 타입이 같은지 비교한다.	x==="5"	false
!=	값이 다른지 비교한다.	x!="5"	false
!==	값 또는 타입이 다른지 비교한다.	x!=="5"	true

예제 9-14 비교 연산자 활용하기 ch09/14_js.html

```
<script>
    var x=5;
    var y="5";
    var result;
    result=(x>y);      // 비교 연산
    document.write(" x > y : " + result + "<br>");
    result=(x==y);     // 두 값이 같은지 비교
    document.write(" x == y : " + result + "<br>");
    result=(x===y);    // 두 값과 타입이 같은지 비교
    document.write(" x === y : " + result +" <br>");
    result=(x!=y);     // 두 값이 다른지 비교
    document.write(" x != y : " + result + "<br>");
    result=(x!==y);    // 두 값이 다르거나 또는 타입이 다른지 비교
    document.write(" x !== y : " + result + "<br>");
</script>
```

```
x > y : false
x == y : true
x === y : false
x != y : false
x !== y : true
```

4. 논리 연산자

● 일반 논리 연산자

논리곱(&&)	두 개의 피연산자 값이 모두 참일 때만 참이고, 하나라도 거짓이면 거짓
논리합(||)	두 개의 피연산자 값 중 하나라도 참이면 참이고, 모두 거짓이면 거짓
논리 부정(!)	피연산자 값이 참이면 거짓, 거짓이면 참

● 일반 논리 연산자의 진리표

A	B	A && B	A || B	!A
거짓	거짓	거짓	거짓	참
거짓	참	거짓	참	참
참	거짓	거짓	참	거짓
참	참	참	참	거짓

4. 논리 연산자

예제 9-15 일반 논리 연산자 활용하기 ch09/15_js.html

```
<script>
    var x=5;
    var y=7;
    var result;
    result=(x<10 && y>10);   // 논리곱
    document.write("(x<10 && y>10) : " + result + "<br>");
    result=(x<10 || y>10);   // 논리합
    document.write("(x<10 || y>10) : " + result + "<br>");
    result=!(x<10 && y>10);  // 논리 부정
    document.write("!(x<10 && y>10) : " + result + "<br>");
</script>
```

(x<10 && y>10) : false
(x<10 || y>10) : true
!(x<10 && y>10) : true

4. 논리 연산자

● 비트 논리 연산자

비트곱(&)	두 비트 모두 1일 때만 1이고, 하나라도 0이면 0
비트합(|)	두 비트 중 하나라도 1이면 1이고, 모두 0이면 0
비트 부정(~)	비트 값이 1이면 0, 0이면 1
배타적 비트합(^)	두 비트가 같을 때 0이고, 다를 때 1

● 비트 논리 연산자의 진리표

A	B	A & B	A | B	A ^ B	~A
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

4. 논리 연산자

예제 9-16 비트 논리 연산자 활용하기 ch09/16_js.html

```
<script>
    var x=5;    // 0101
    var y=7;    // 0111
    var result;
    result=(x & y);    // 비트곱
    document.write("x & y = " + result + "<br>");
    result=(x | y);    // 비트합
    document.write("x | y = " + result + "<br>");
    result=(x ^ y);    // 배타적 비트합
    document.write("x ^ y = " + result + "<br>");
    result=(~x);       // 비트 부정
    document.write("~x = " + result + "<br>");
</script>
```

```
x & y = 5
x | y = 7
x ^ y = 2
~x = -6
```

5. 조건 연산자

- **조건 연산자**
  - 조건식을 판별하여 참이냐 거짓이냐에 따라 다음 문장을 선택적으로 실행

```
max_value=(a>b) ? a : b;     // a, b 중 큰 값을 저장
```

예제 9-17 조건 연산자 활용하기 ch09/17_js.html

```
<script>
    var x=5;
    var y=7;
    var result;
    result=(x > y)? x : y;          // 조건 연산
    document.write("큰 값 : " + result + "<br>");
    result=(x > y)? x-y : y-x;      // 조건 연산
    document.write("큰 값-작은 값 : " + result + "<br>");
</script>
```

```
큰 값 : 7
큰 값-작은 값 : 2
```

5. 대입 연산자

● 대입 연산자

- '=' 기호를 사용하여 값이나 변수를 할당

예제 9-18 복합 대입 연산자 활용하기 ch09/18_js.html

```
<script>
    var x1=x2=x3=x4=x5=10;
    var st="Hello ";
    x1+=1;
    document.write("x1 : " + x1 + "<br>");
    x2-=2;
    document.write("x2 : " + x2 + "<br>");
    x3*=3;
    document.write("x3 : " + x3 + "<br>");
    x4/=4;
    document.write("x4 : " + x4 + "<br>");
    x5%=5;
    document.write("x5 : " + x5 + "<br>");
    st+="Javascript";
    document.write("st : " + st + "<br>");
</script>
```

```
x1 : 11
x2 : 8
x3 : 30
x4 : 2.5
x5 : 0
st : Hello Javascript
```

0. 개요

- **제어문**
  - 프로그램의 실행 과정을 제어하기 위해 사용하는 구문

- **자바스크립트 제어문**

유형	설명	구조
조건문	조건에 따라 다음 문장을 선택적으로 실행한다.	• If문 • if~else문 • 다중 if~else문 • switch~case문
반복문	동일한 명령을 여러 번 처리하거나 특정 연산을 반복적으로 처리한다.	• for문 • while문 • do~while문
보조 제어문	조건문을 만나면 건너뛰거나 반복 수행을 종료한다. 반복문 내에서 사용한다.	• continue문 • break문

1. if문

- **if문**
  - 조건식이 참(true)이면 블록 내의 문장을 처리하고, 거짓이면 블록을 빠져나감

```
if(조건식) {
    실행 문장;
}
```

```
if (조건식A) {
    실행 문장;
    if (조건식B) {
        실행 문장;
    }
}
```

2. if~else문

- **if~else문**
  - 조건식이 참(true)인 경우와 거짓(false)인 경우 처리할 문장이 각각 따로 있을 때 사용하는 제어문

```
if(age>=19) {
    result="성인입니다.";
}
else {
    result="미성년자입니다.";
}
```

2. if~else문

예제 9-19 성별과 성년을 구분하는 프로그램 만들기 ch09/19_js.h

```
<script>
    var gender="M";    // 남자(M), 여자(F)
    var age=21;
    if(gender=="M") {
        if(age>=19) {
            result="남자 성인입니다.";
        }
        else {
            result="남자 미성년자입니다.";
        }
    }
    else {
        if (age>=19) {
            result="여자 성인입니다.";
        }
        else {
            result="여자 미성년자입니다.";
        }
    }
    document.write("당신은 " + result + "<p/>");
</script>
```

당신은 남자 성인입니다.

2. if~else문

예제 9-20 로그인 프로그램 만들기 ch09/20_html.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <p>아이디, 비밀번호 입력</p>
    <script src="./ejs/script.js"> </script>
</body>
</html>
```

ch09/script.js

```
id=prompt('아이디 입력');
if(id=='admin') {
    password=prompt('비밀번호 입력');
    if(password==='123456') {
        location.href="20_login.html"
    }
    else {
        location.href="20_error.html"
    }
}
else {
    location.href="20_error.html"
}
```

2. if~else문

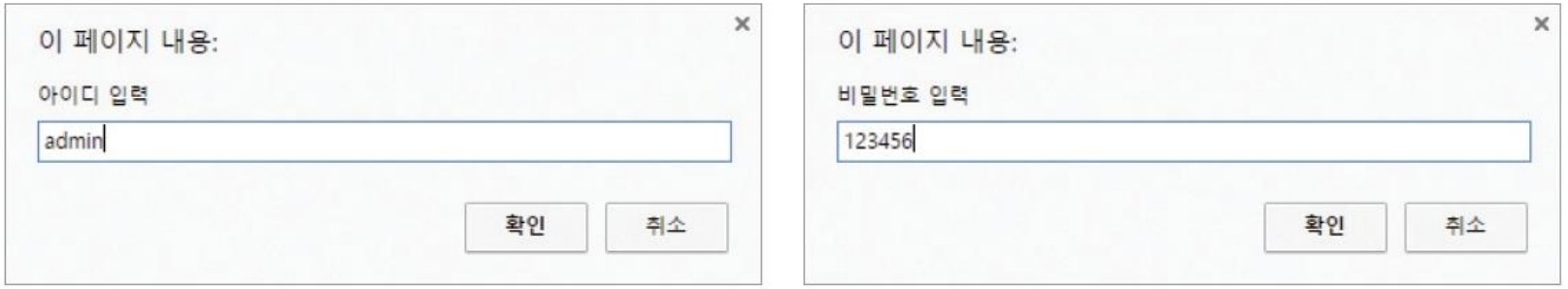
ch09/20_login.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <h2>회원 인증에 성공했습니다.</h2>
    <p>저자 홈페이지를 클릭하세요.</p>
    <a href="http://cafe.naver.com/go2web">차세대 웹 프로그래밍</a>
</body>
</html>
```

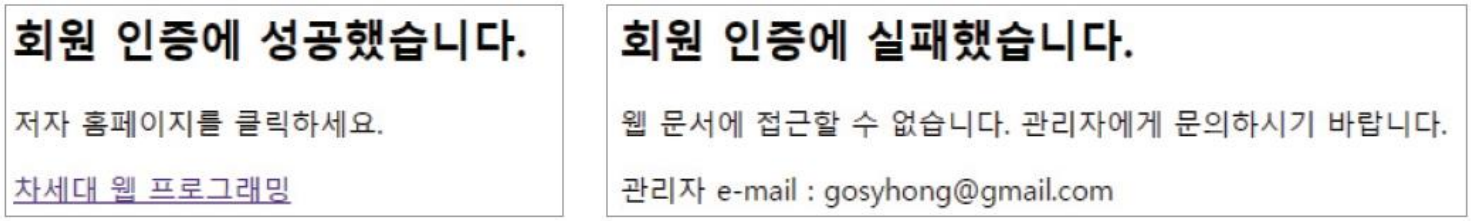
ch09/20_error.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
</head>
<body>
    <h2>회원 인증에 실패했습니다.</h2>
    <p>웹 문서에 접근할 수 없습니다. 관리자에게 문의하시기 바랍니다.</p>
    <p>관리자 e-mail : gosyhong@gmail.com</p>
</body>
</html>
```

2. if~else문



▲ 아이디/비밀번호 입력



▲ 로그인 성공/실패

3. 다중 if~else문

- **다중 if~else문**
  - 여러 조건을 체크해야 할 때 사용

```
if(조건식-1) {
    조건식-1의 결과가 참일 때 실행할 문장;
}
else if(조건식-2) {
    조건식-2의 결과가 참일 때 실행할 문장;
}
else {
    조건식-1, 조건식-2 모두 거짓일 때 실행할 문장;
}
```

3. 다중 if~else문

예제 9-21 학점 환산 프로그램 만들기 ch09/21_js.html

```
<script>
    var point=93;    // 과목 점수
    var grade="";
    if(point>100) {
        document.write("0~100점 사이 값을 입력해야 합니다." + "<p/>");
    }
    else if(point>=90) {
        grade="A";
        document.write("아주 잘했어요." + "<p/>");
    }
    else if(point>=80) {
        grade="B";
        document.write("잘했어요." + "<p/>");
    }
    else if(point>=70) {
        grade="C";
        document.write("조금만 노력하면 잘할 수 있어요." + "<p/>");
    }
    else if(point>=60) {
        grade="D";
        document.write("좀 더 노력하세요." + "<p/>");
    }
    else{
        grade="F";
        document.write("많이 노력하시기 바랍니다." + "<p/>");
    }
    document.write("학생의 학점은 <b>" + grade + "</b>입니다.<p/>");
</script>
```

아주 잘했어요.

학생의 학점은 **A**입니다.

4. swith~case문

- **switch~case문**
  - 조건문을 체크하여 다음에 처리할 문장의 위치를 파악한 후 해당 문장으로 가서 바로 처리

```
switch(상수값) {
    case n:
        실행 문장;
        break;
    case n:
        실행 문장;
        break;
    default:
        기본 실행 문장;
}
```

4. swith~case문

예제 9-22 요일을 알려주는 프로그램 만들기 ch09/22_js.html

```
<script>
    var day;
    var week=new Date().getDay();    // 0(일요일)~6(토요일)
    switch(week) {
        case 0:
            day="일요일";        break;
        case 1:
            day="월요일";        break;
        case 2:
            day="화요일";        break;
        case 3:
            day="수요일";        break;
        case 4:
            day="목요일";        break;
        case 5:
            day="금요일";        break;
        case 6:
            day="토요일";        break;
        default:
            day="없는 요일";
    }
    document.write("오늘은 <b>" + day + "</b>입니다. <p/>");
</script>
```

오늘은 **월요일**입니다.

4. swith~case문

예제 9-23 요일별 일정을 알려주는 프로그램 만들기 ch09/23_js.html

```
<script>
    var text;
    var week=new Date().getDay();    // 0(일요일)~6(토요일)
    switch(week) {
        case 1:    // 월요일
        case 2:    // 화요일
            text="HTML5";
            break;
        case 3:    // 수요일
        case 4:    // 목요일
            text="자바스크립트";
            break;
        case 5:    // 금요일
        case 6:    // 토요일
            text="영어";
            break;
        case 0:    //일요일
        default:
            text="수영";
    }
    document.write("오늘은 <b>" + text + "</b> 학습하는 날입니다. <p/>");
</script>
```

오늘은 **HTML5** 학습하는 날입니다.

5. for문

for문 형식

```
for(초기식; 조건식; 증감식) {
    실행 문장;
}
```

- 초기식 : 반복 변숫값을 초기화하며, for문이 처음 시작할 때 단 한 번만 실행됨
- 조건식 : 블록 내 문장을 얼마나 반복할지 결정하며, 조건식이 참인 동안 반복함
- 증감식 : 초기식에서 초기화한 변수의 값을 증가 또는 감소시킴

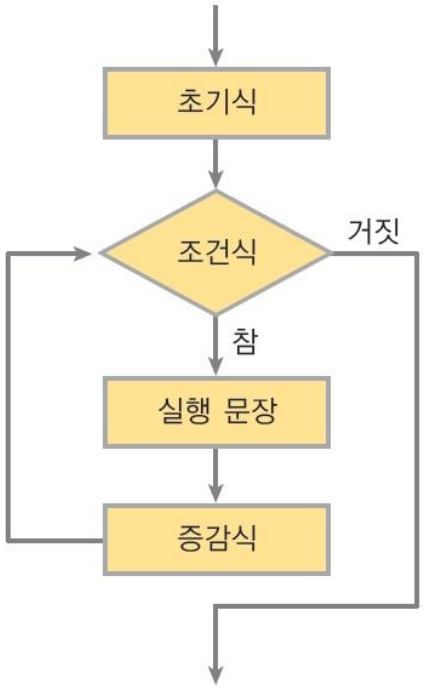


그림 9-7 for문의 순서도

5. for문

- **for문의 변칙적 사용**

① 초기식을 for문 이전에 먼저 선언을 했다면 for문에서는 생략 가능

```
<script>
    var num =1;     // 초깃값 선언
    for(  ; num<=5; num++) {
        document.write("for문 수행 : <b>" + num + "</b> <p/>");
    }
</script>
```

② 초기식은 여러 개 선언할 수 있음

```
<script>
    var num;
    var st="ABCDEF";     // 문자열 길이 6
    var ct;
    for(num=1, ct=st.length; num<ct; num++) {
        document.write("for문 수행 : <b>" + num + "</b> <p/>");
    }
</script>
```

5. for문

- **for문의 변칙적 사용**

  ③ for문의 블록 내에 증감식 문장을 포함한다면 for문 자체에서 증감식을 생략해도 됨

```
<script>
    var num =1;
    var st="ABCDEF";
    var ct=st.length;
    for(  ; num<ct;  ) {
        document.write("for문 수행 : <b>" + num + "</b> <p/>");
        num++;
    }
</script>
```

5. for문

- **for문의 변칙적 사용**

  ④ for(; ;)와 같이 초기식, 조건식, 증감식을 모두 작성하지 않으면 블록 내 문장을 무한 반복하게 됨

```
<script>
    var num=1;
    var st="ABCDEF";
    var ct=st.length;
    for(  ;  ;  ) {
        if(num<ct){
            document.write(" for문 수행 : <b>" + num + "</b> <p/>");
            num++;
            continue;
        }
        break;
    }
</script>
```

5. for문

예제 9-24 구구단 프로그램 만들기 ch09/24_js.html

```
<script>
    var x, y;
    for(x=2; x<=5; x++) {
        document.write("<b> ---[" + x + "단]--- </b>" + "<br>");
        for(y=1; y <= 9; y++) {
            document.write(x + "*" + y + "=" + (x * y) + "<br>");
        }
    }
</script>
```

---[2단]---
2*1=2
2*2=4
2*3=6
2*4=8
2*5=10
2*6=12
2*7=14

6. while문

- **while문의 형식**

```
while(조건식) {
    실행 문장;
}
```

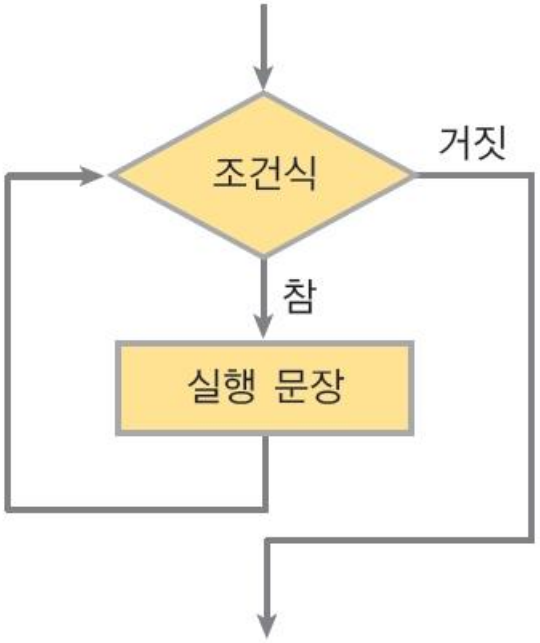


그림 9-8 while문의 순서도

6. while문

예제 9-25 1부터 100까지 합 구하기 ch09/25_js.html

```
<script>
    var x=1;
    var sum=0;
    while(x<=100) {
        sum+=x;
        x++;
    }
    document.write("1~100까지 합 : <b>" + sum + "</b>");
</script>
```

1~100까지 합 : **5050**

예제 9-26 1부터 10000까지 합 구하기 ch09/26_js.html

```
<script>
    var x=1;
    var sum=0;
    while(1) {    // 무한 반복
        sum+=x;
        x++;
        if(x==10001)
            break;
    }
    document.write("1~10000까지 합 : <b>" + sum + "</b>");
</script>
```

1~10000까지 합 : **50005000**

7. do~while문

● do~while문의 형식

```
do {
    실행 문장;
}
while(조건식);
```

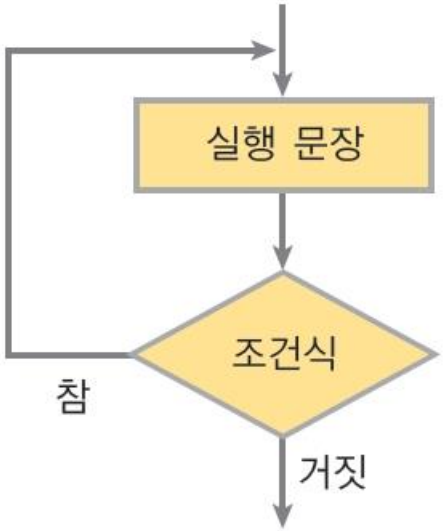


그림 9-9 do~while문의 순서도

예제 9-27 do~while문으로 1부터 100까지 합 구하기 ch09/27_js.html

```
<script>
    var x=1;
    var sum=0;
    do {
        sum+=x;
        x++;
    } while(x<=100);
    document.write("1~100까지 합 : <b>" + sum + "</b>");
</script>
```

1~100까지 합 : **5050**

8. break문

- **break문**
  - for문, while문, do~while문과 같은 반복문이나 switch~case문 내에서 해당 블록을 강제적으로 벗어나 다음 문장을 처리하도록 할 때 사용

예제 9-28 break문으로 1부터 100까지 수 중 3의 배수 합 구하기 ch09/28_js.html

```
<script>
    var x=0;
    var sum=0;
    while(1) {
        x+=3;   // 3의 배수
        if(x>100)
            break;
        sum+=x;
        document.write(x + " ");
    }
    document.write("<p/>");
    document.write("1~100까지 수 중 3의 배수 합 : <b>" + sum + "</b>");
</script>
```

3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69
72 75 78 81 84 87 90 93 96 99

1~100까지 수 중 3의 배수 합 : **1683**

9. continue문

- **continue문**
  - if문의 조건식이 참이면 continue문 이후의 문장을 처리하지 않고 제어를 반복문의 시작 위치로 옮김

예제 9-29 continue문으로 1부터 100까지 수 중 3의 배수 합 구하기 ch09/29_js.html

```
<script>
    var x=0;
    var sum=0;
    for(x=1; x<=100; x++) {
        if(x%3 != 0)
            continue;
        sum+=x;
        document.write(x + " ");
    }
    document.write("<p/>");
    document.write("1~100까지 수 중 3의 배수 합 : <b>" + sum + "</b>");
</script>
```

3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69
72 75 78 81 84 87 90 93 96 99

1~100까지 수 중 3의 배수 합 : **1683**

10. label문

- **label문**
  - 제어를 블록 바깥으로 옮김

예제 9-30 label문 활용하기 ch09/30_js.html

```
<script>
    var i, j;
    outloop:   // label name
    for(i=0; i<3; i++) {
        inloop:  // label name
        for(j=0; j<3; j++) {
            if(i===1 && j===0) {
                continue outloop;
            }
        document.write("i = " + i + ", j = " + j + "<br>");
        }
    }
</script>
```

```
i = 0, j = 0
i = 0, j = 1
i = 0, j = 2
i = 2, j = 0
i = 2, j = 1
i = 2, j = 2
```

Thank You