



Chapter08

JavaScript 시작하기

HTML & JavaScript

자바스크립트 – 학습 사이트

- ❖ <http://www.digithome.pe.kr>
빨강콩의 자바스크립트 사이트로
css, javascript, dhtml 정보를 제공한다. 특히
도움말이 대화상자 형태로 제공된다.
- ❖ <http://www.jasko.co.kr>
J@SKO 사이트. **JavaScript Source Bank**. 방
대한 자바스크립트 소스를 제공한다. 유료 사이트다
.
- ❖ <http://javascripthouse.wo.to>
소스 베껴쓰기, 기초 예제를 제공한다.

자바스크립트 - 스크립트의 등장과 역할

- ❖ 최초의 스크립트 : **1987**년 애플사의**HyperCard**
- ❖ 발전계기 : **1990**년대초에 **MS**사에서 **VB**에서 사용할 수 있는 **VBA(VBApplication)** 개발
- ❖ **SunMicrosystems**사가 인터넷 프로그래밍 언어로 **Java**를 개발
- ❖ 넷스케이프사는 선사와 전략적 제휴를 통하여, **HTML** 기능을 수용하면서 프로그래밍 개념을 대폭 수용한 **JavaScript** 개발

자바스크립트 - 특징

- ❖ 서버가 아닌 클라이언트에서 인터프리터
- ❖ 다이나믹 바인딩이 된다.
- ❖ 객체 지향형 언어다.
 - 객체 : 윈도우, 프레임, URL, 폼, 버튼, 도큐먼트 등
- ❖ **HTML** 문서에 혼합하여 사용한다.
- ❖ 변수의 형(**type**)을 지정할 필요가 없다.
- ❖ 일반 사용자가 프로그래머 수준의 핸들링을 할 수 있다.

자바스크립트 – 사용 목적

- ❖ 인터랙티브(**interactive**)한 홈페이지
- ❖ 경제적인 가격의 컴퓨터로 서버 구축
- ❖ 플랫폼이 독립적이다.
- ❖ 역동적인 홈페이지 제작
- ❖ 웹 프로그램 사용 시 반드시 필요하다.

자바스크립트 – 자바와 자바스크립트

구분	자바	자바스크립트
해석위치	서버	클라이언트
언어형	컴파일러	인터프리터
존재	*.class 파일로 존재	HTML 문서 내에 기술
객체지향	객체 지향 언어	객체 기반 언어
보안성	있음	없음
사용	어려움	쉬움

자바스크립트 – 자바스크립트 기본

기본 사용법

삽입 및 실행법

사용자 정의 함수

작성시 주의사항

특수문자와 주석달기

자바스크립트 – 자바스크립트의 시작

❖ 자바 스크립트의 삽입 위치는?

❖ 기본구조

```
<SCRIPT LANGUAGE="JavaScript">
```

```
<!--
```

진짜 자바스크립트 코드

```
// ->
```

```
</SCRIPT>
```

❖ 자바스크립트의 실행시기는?

자바스크립트 – 자바스크립트 삽입과 실행(1)

- ❖ 내장형
- ❖ 행 입력형
- ❖ 함수형
- ❖ 링크형

자바스크립트 – 자바스크립트 삽입과 실행(2)

❖ 내장형

```
<script language=javascript>  
  ~~자바스크립트 소스~~  
</script>
```

❖ 행 입력형

```
<태그 이벤트핸들러="자바스크립트 소스">
```

자바스크립트 – 자바스크립트 삽입과 실행(3)

❖ 함수형

```
<script language=javascript>  
function 함수명( )  
{  
    ~~자바스크립트 소스~~  
}  
</script>  
<태그명 이벤트핸들러="함수명()">
```

❖ 링크형

```
<script language=javascript src="js 파일의 전체  
경로"></script>
```

자바스크립트 – 주의 사항

- ❖ 대소문자를 반드시 구분해야 한다
- ❖ 구문은 한 줄에 한 개씩 위치시킨다
- ❖ 객체, 속성, 메소드, 함수의 구분은 마침표(.) 연산자를 사용한다.
- ❖ 문자열 표시는 따옴표를 사용해야 한다
- ❖ 작은따옴표나 큰따옴표를 중첩해서 사용할 때는 반드시 나중에 시작한 따옴표를 먼저 닫아야 한다.
- ❖ 따옴표 자체를 문자열에 포함시켜야 할 경우에는 역슬래시(\)와 따옴표를 함께 사용한다.

자바스크립트 - 특수문자

- ❖ **\n** : 개 행 (한 줄 바꾸어 출력한다)
- ❖ **\t** : 탭 (일정한 수의 스페이스를 삽입한다)
- ❖ **** : 역슬래시 표시
- ❖ **\"** : 쌍따옴표 표시
- ❖ **\'** : 온따옴표 표시

자바스크립트 – 주식달기

❖ 한 행을 주식문 처리

//주식 처리할 행, 문장

❖ 두 행 이상에 걸치는 주식문 처리

/* 주식 처리할 영역 */

❖ **HTML** 문서의 주식

<!-- 주식 처리할 영역 -->

자바스크립트 – 객체

- 객체&속성&메소드 비교
- 객체의 계층 구조
- 객체 표현법
- Window 객체
- Document 객체

자바스크립트 – 객체, 속성, 메소드의 비교

구분	객체	속성	메소드
원어	object	property	method나 function
특징	프로그램의 대상이 되는 모든 것	객체의 속성, 성격, 특징	객체의 기능, 성능, 역할
예	창, 문서	색깔, 크기, 모양	저장, 닫기

자바스크립트 - 객체의 계층 구조

Window

parent
Frames
self, top
location
history
document

forms
links
anchors

Elements:

Text
fields
textarea
Checkbox
Password
Radio
Select
Button
Submit
reset

자바스크립트 – 객체 표현법

❖ 객체명.속성="값"

window.status="GO!"

❖ 객체명.메소드

window.close()

❖ 상위객체이름.하위객체이름.속성="값"

window.document.frm1.id.value="이도연"

자바스크립트 – **window** 객체

❖ 윈도우의 생성 및 제거 메소드

❖ **open()**

- `open(“문서명”, “창이름”, “옵션”)`
- 생성될 윈도우의 겉모양
`toolbar, location, status, menubar, scrollbars, resizable, copyhistory, width, height`

❖ **close()**

자바스크립트 – **document** 객체

- ❖ 웹 문서의 색상 설정과 관련된 속성
fgColor bgColor,alinkColor,linkColor ,vlinkColor ,
- ❖ 웹 문서와 관련된 정보를 다루는 속성
lastModified,location,referrer,title
- ❖ 웹 문서에 포함된 내용과 관련된 속성
anchors,cookie,forms,links
- ❖ **document** 객체의 메소드
open(),close(),clear(),write(),writeln()

자바스크립트 – 이벤트와 이벤트핸들러

이벤트 **&** 이벤트핸들러
이벤트핸들러의 종류

자바스크립트 – 이벤트와 이벤트핸들러

- ❖ 사용자가 마우스를 움직이거나 키를 누르는 등의 동작을 **event**라 한다.
- ❖ 이 이벤트 앞에 **on**을 붙이면 **event handler**가 된다.
- ❖ 사용자의 행위 자체는 이벤트
- ❖ 사용자의 행위를 전달하는 시점은 이벤트핸들러

자바스크립트 – 이벤트핸들러의 종류(1)

- ❖ **onLoad()** : 웹 브라우저적 홈페이지를 불러올 때
- ❖ **onUnload()** : 현재 홈페이지에서 빠져 나가려 할 때
- ❖ **onClick()** : 마우스를 클릭할 때

자바스크립트 – 이벤트핸들러의 종류

- ❖ **onFocus()** : 양식이나 홈페이지에 커서나 포커스가 위치했을 때
- ❖ **onBlur()** : 양식이나 홈페이지에서 커서나 포커스가 다른 곳으로 이동할 때
- ❖ **onMouseover()** : 마우스가 위로 왔을 때
- ❖ **onMouseout()** : 마우스가 영역을 벗어났을 때

변수, 연산자,
사용자 정의 함수



변수

배열과 객체

연산자

함수

배열, 연산자, 사용자 정의 함수 – 기본 실행문

❖ 변수 선언문 : **var i = 10**

❖ 대입문 : **i = 10**이나 **i = “masan”**

❖ 조건문

if(i < 10) document.write(“조건만족”)

❖ 순환문

```
for(var i = 0; i < 10; i++) {  
    document.write(i);  
}
```

배열, 연산자, 사용자 정의 함수 – 변수

❖ 변수의 데이터 형(**type**)

- Numbers(숫자형), String(문자열형)
- Boolean(논리형), Null(널)

❖ 변수의 명명시 주의사항

- 예약어, 함수명, 객체명, 속성명, 사용 중인 변수 등은 사용할 수 없다.
- 변수는 영자나 밑줄로만 시작한다.
- 대소문자를 구별하되, 의미있는 이름을 붙인다.

배열, 연산자, 사용자 정의 함수 – 배열 변수 선언법

❖ 배열(**array**)은 같은 형, 같은 길이의 데이터를 **2**개 이상 붙여서 동일한 변수로 처리하는 것

❖ 기본 형식

`var 배열 변수명 = new Array()`

`배열 변수명[0]=값`

`배열 변수명[1]=값`

`배열 변수명[2]=값`

❖ **var 배열 변수명 = new Array(배열개수)**

❖ **var jumsu = new Array(값1, 값2, 값3)**

배열, 연산자, 사용자 정의 함수 – 연산문

❖ 산술연산문 : **+, -, *, /, %**

- 증감연산 : ++, --

❖ 대입연산문 : **=, +=, -=, *=, /=, %=**

❖ 조건 연산자:

- 변수명=(조건식)? 명령1 : 명령2

❖ 논리연산문 : **&&, ||, !,**

- 관계연산자 : >, <, >=, <=

- 비교연산 : ==, !=

❖ 연결연산문 : **“happy” + “day”**

배열, 연산자, 사용자 정의 함수 – 연산기호의 우선순위

❖ 산술 > 논리 > 대입

1. $()$
2. $! ++ --$
3. $* / \%$
4. $+ -$
5. $< <= > >=$
6. $== !=$
7. $\&\&$
8. $||$
9. $= += -= *= /= \%=$

배열, 연산자, 사용자 정의 함수 – 사용자 정의 함수

❖ **FUNCTION**은 프로그램의 형식을 완전히 갖추지 않은 부속 프로그램으로, 복잡한 계산을 하거나 자주 사용되는 루틴을 정형화할 때 쓰인다.

❖ 함수의 정의

<!--

```
function makeWindow(){  
    window.open("allim.htm","new","width=200  
    height=200")}
```

//-->

❖ 함수의 호출

```
<body onload="makeWindow()">
```

배열, 연산자, 사용자 정의 함수 – 사용자 정의 함수의 종류

❖ 매개변수가 없는 함수

```
Function test(){...}
```

❖ 매개변수가 있는 함수

```
Function test(name){...}
```

❖ 리턴 값이 있는 경우

```
Function test(question){  
    Ans=confirm(question)  
    Return ans  
}
```

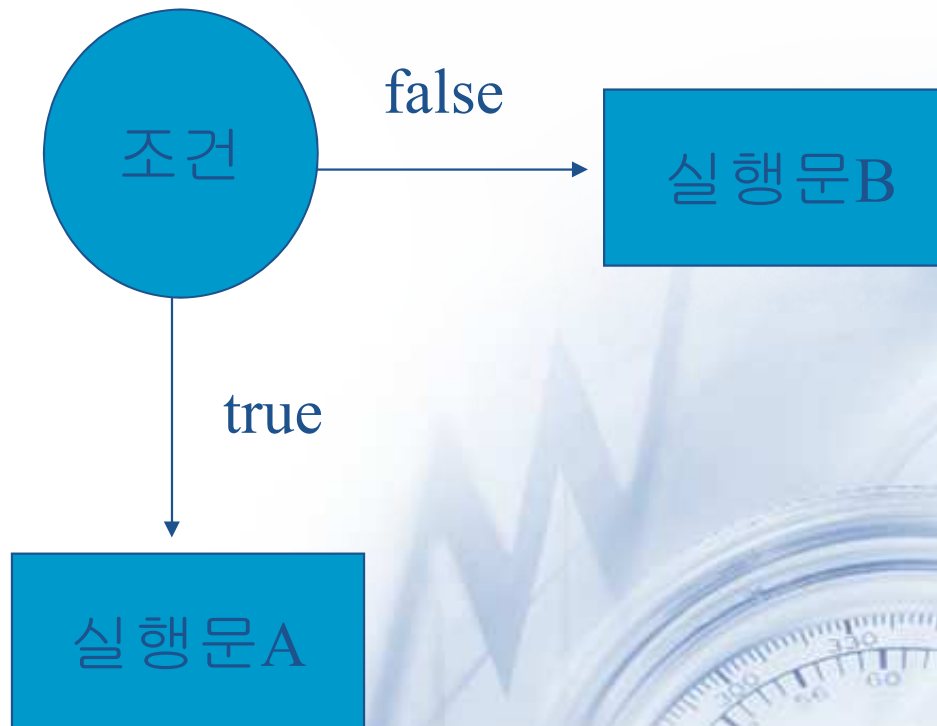

제어문과 내장 함수

제어문
내장 함수

제어문
내장 함수

제어문과 내장 함수 – **IF** 조건문(1)

if(조건)
 실행문**A**
else
 실행문**B**



제어문과 내장 함수 – **IF** 조건문(2)

❖ 형식1

```
if(조건)  
    명령문
```

❖ 형식2

```
if(조건){  
    명령문1  
    명령문2  
}  
else{  
    명령문1  
    명령문2  
}
```

❖ 중첩 IF문

```
if(조건)  
    명령문  
else if(조건)  
    명령문  
else if(조건)  
    명령문  
else(조건)  
    명령문
```

제어문과 내장 함수 – SWITCH 문

switch(표현식){

case value1:

명령문1;

Break;

case value2:

명령문;

Break;

.....

default

명령문n

}

제어문과 내장 함수 – **FOR** 문 - 반복문(1)

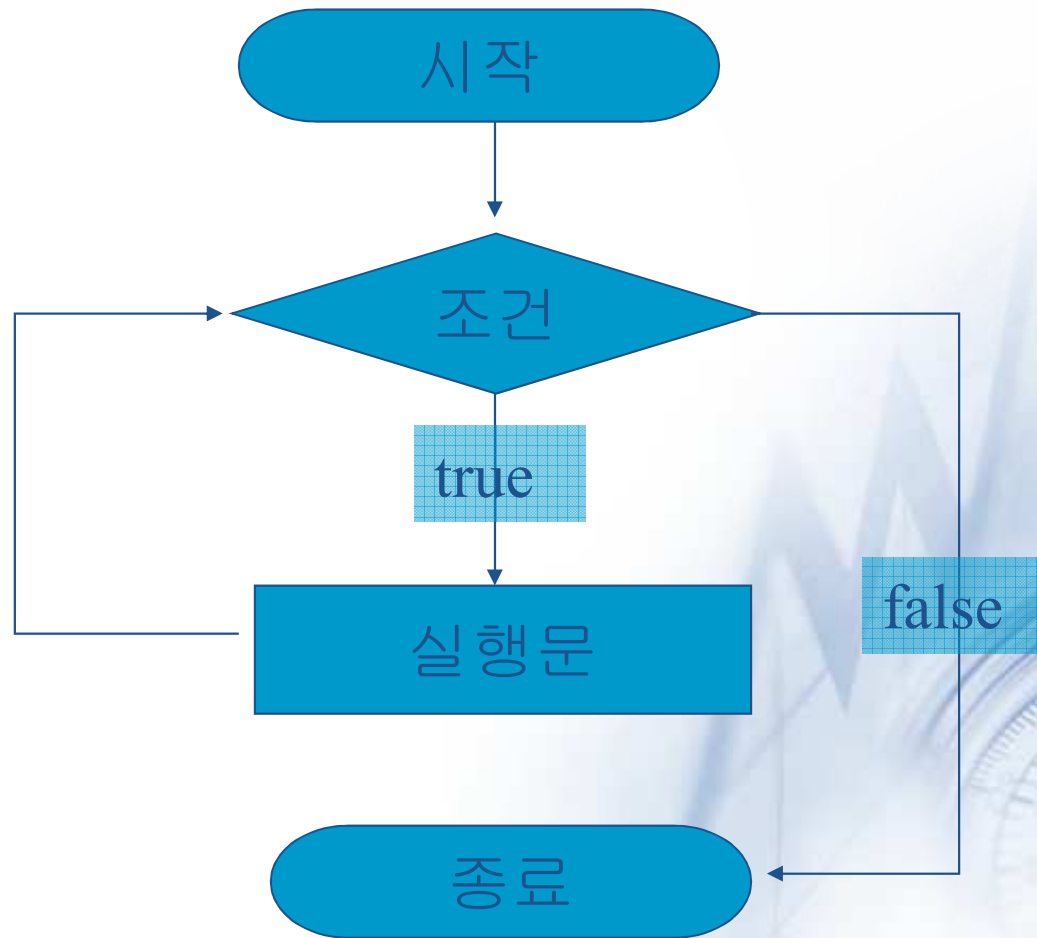
❖ 기본 형식

```
for(초기 값;조건부;증감식){  
    코드부  
}
```

❖ 예제

```
for(a=1;a<11;a++){  
    document.write(a+ "*" +a+ "=" +a*a+"<br>")  
}
```

제어문과 내장 함수 – **FOR** 문 - 반복문(2)



제어문과 내장 함수 – **WHILE** 문

❖ 기본 형식

```
while(조건){  
    명령문  
}
```

❖ 예제

```
a=1  
while(a<11){  
    document.write(a+ "*" +a+ "=" +a*a+ "<br>")  
    a++  
}
```

제어문과 내장 함수 – **DO WHILE** 문

❖ 기본 형식

```
do{  
    명령문  
} while(조건);
```

❖ 예제

```
a=1  
do{  
    document.write(a+ "*" +a+ "=" +a*a+"<br>")  
    a++  
} while(a<11);
```


제어문과 내장 함수 – **BREAK, CONTINUE** 문

❖ **Break** : 제어문 종료

❖ **Continue** : 제어문 반복

❖ 예제

```
a=10
while(true){
    a—
    if(a>10) continue
    if(a==0) break
    document.write(a+ "*" +a+ "=" +a*a+"<br>")
}
```

제어문과 내장 함수 – **RETURN** 문

❖ 함수에서 특정 값을 리턴 값으로 보내고 싶을 때
사용

❖ 예제

```
function square(a){  
    return a*a  
}
```

```
for(a=1;a<11;a++){  
    document.write(a+ "*" +a+ "=" +a*a+"<br>")  
}
```

제어문과 내장 함수 – **FOR IN**– 객체 조작문

- ❖ 객체가 가지는 속성 정보를 알려준다.
- ❖ 만약 객체의 모든 속성이 **5**개라면 **5**번 반복된다.
- ❖ 자바 스크립트는 완성된 언어가 아니므로 버전업되면서 새로 추가된 객체의 속성 정보를 알 수 있다.
- ❖ **for (variable in 객체)**
 {
 수행할 작업
 }

제어문과 내장 함수 – **WITH** – 객체 조작문

❖ 하나의 객체에 대해 여러 가지 속성들을 한꺼번에 조작할 때 사용한다.

❖ **with** (객체)

```
{  
    조작 내용  
}
```

❖ **with (document)**

```
{  
    bgcolor = "white";  
    fgcolor = "red";  
}
```

제어문과 내장 함수 – 내장 함수의 종류

표현식	설명
alert("메시지")	‘확인’ 버튼이 있는 메시지 창을 띄움
prompt("메시지","기본 문구")	입력상자가 있는 메시지 창을 띄움
confirm("메시지")	‘확인’, ‘취소’ 버튼이 있는 메시지 창을 띄움
eval()	문자열을 수식으로 바꿈
isNaN()	전달받은 값이 숫자인지 문자인지 판별하여 숫자가 아닌 경우 true 값을 반환
parseFloat()	문자열을 부동소수점으로 바꿈
parseInt()	문자열을 정수로 바꿈
escape()	ISO-Latin-1 문자 셋을 아스키 값으로 바꿈, 문자 값을 URL 표기형으로 변환
unescape()	아스키 값을 ISO-Latin-1 문자 셋으로 바꿈, URL 표기형을 문자로 변환
isFinite()	전달받은 값이 유리수인지 판단하여 유리수인 경우에만 true 값 반환
Number()	객체를 수치로 변환
String()	객체를 문자열로 변환

window, screen, document,
link, anchor 객체



window 객체

screen 객체와 document 객체

link 객체와 anchor 객체

Window, screen, document, link, anchor 객체 – screen 객체의 속성

속성	설명
defaultStatus	브라우저 상태 표시줄에 표시할 문자열의 초기 값 설정
status	상태 표시줄에 표시할 문자열 지정
window	창 자신을 가리킴
frames	창에 포함된 프레임을 배열 형태로 지정
opener	open() 메소드를 사용해서 새 창을 열었을 경우, 새 창을 열도록 한 문서를 가리킴
parent	주종 관계를 이루고 있는 프레임 문서에서 상위 프레임을 가리킴
self	현재 작업 중인 창, 즉 자기자신을 가리킴
top	프레임이 설정되기 전의 상태로 돌아감, 프레임을 모두 무시하고 창 하나만 남음
classes	문서 안에 정의된 모든 스타일시트의 정보를 갖음
tags	문서 안에 정의된 모든 태그의 정보를 나타냄
screenX	창의 x 좌표 반환
screenY	창의 y 좌표 반환
closed	창이 닫혀 있는지 확인한 후 true, false 반환
name	창의 이름 반환
length	창 안에 프레임 수 반환

Window, screen, document, link, anchor 객체 – window 객체의 메소드(1)

❖ 사용자의 응답을 요구하는 다이얼로그 상자를 생성하는 메소드

- alert("메시지") : 경고 창 띄움
- prompt("메시지", "기본 문구") : 입력 창 띄움
- confirm("메시지") : 확인, 취소 창 띄움

❖ 창 생성 및 제거 메소드

- open("문서명", "창이름", "속성") : 새 창을 엮
- close() : 창을 닫음

Window, screen, document, link, anchor 객체 – window 객체의 메소드(2)

❖ 브라우저 크기, 위치 설정

- `moveBy(x,y)` : 브라우저를 상대적인 좌표로 지정
한 픽셀만큼 이동
- `moveTo(x,y)` : 브라우저를 절대적인 좌표로 지정
한 위치로 이동
- `resizeBy(x,y)` : 브라우저의 크기를 상대적으로 지
정한 픽셀만큼 설정
- `resizeTo(x,y)` : 브라우저의 크기를 절대 값으로
지정한 픽셀만큼 설정
- `scroll(x,y)` : 창이나 프레임 안의 내용을 스크롤함
- `scrollBy(x,y)` : 스크롤을 상대좌표로 이동
- `scrollTo(x,y)` : 스크롤을 절대좌표로 이동

Window, screen, document, link, anchor 객체 – window 객체의 메소드(3)

❖ 동작 시간 간격을 조절하는 메소드

- `setTimeout("명령문", 시간간격)` : 일정한 시간 간격으로 명령문을 반복 실행(시간은 1/1000초 단위)
- `clearTimeout()` : `SetTimeout()`으로 동작되는 타이머 해제

❖ 기타

- `print()` : 현재 문서 출력

Window, screen, document, link, anchor 객체 –
open 메소드로 창 열기

window.open("문서명", "창이름", "옵션 설정");

Window, screen, document, link, anchor 객체 – window.open 메소드에 설정할 수 있는 창 옵션

옵션	값	설명
directories	yes/no	연결 등 디렉토리 메뉴 표시 여부
menubar	yes/no	메뉴 바 표시 여부
toolbar	yes/no	툴 바 표시 여부
location	yes/no	주소 입력줄 표시 여부
resizable	yes/no	크기 재설정 버튼 표시 여부
status	yes/no	상태 표시줄 표시 여부
scrollbars	yes/no	스크롤 바 표시 여부
copyhistory	yes/no	히스토리 정보를 저장할지 여부
channelmode		전체화면 모드
fullscreen		전체화면으로 표시
width	pixel	창 넓이
height	pixel	창 높이
left	pixel	창의 좌측 좌표 값
top	pixel	창의 위쪽 좌표 값

Window, screen, document, link, anchor 객체 – screen 객체의 속성

속성	설명
availHeight	작업 표시줄을 제외한 화면의 높이를 픽셀 값으로 표시
availWidth	작업 표시줄을 제외한 화면의 너비를 픽셀 값으로 표시
availTop	화면 표시 영역의 y 좌표 표시
availLeft	화면 표시 영역의 x 좌표 표시
height	화면의 높이를 픽셀 값으로 표시
width	화면의 너비를 픽셀 값으로 표시
colorDepth	컴퓨터에서 사용하고 있는 컬러 수를 표시
pixelDepth	화면의 컬러 해상도(bits/pixel)를 표시(넷스케이프 네비게이터용)

Window, screen, document, link, anchor 객체 – document 객체의 속성(1)

❖ 색상 지정

❖ **bgcolor** : 문서의 배경색 지정

❖ **fgcolor** : 문서의 글자색, 선택 지정

❖ **linkcolor** : 링크 문자색 지정

❖ **vlinkcolor** : 방문한 링크 문자색 지정

❖ **alinkcolor** : 선택 중인 링크 문자색 지정

Window, screen, document, link, anchor 객체 – document 객체의 속성(2)

- ❖ 웹 문서와 관련된 정보
- ❖ **lastModified** : 홈페이지가 마지막으로 갱신된 날짜 저장
- ❖ **location** : 웹 문서의 **URL** 주소 저장
- ❖ **URL** : 문서의 **URL** 주소 값 반환
- ❖ **domain** : 서버의 도메인 이름을 지정하거나 반환
- ❖ **title** : **<title>** 태그 사이의 문서 제목을 제공
- ❖ **cookie** : 쿠키 파일의 정보를 읽고 쓸 수 있음

Window, screen, document, link, anchor 객체 – document 객체의 속성(3)

- ❖ 웹 문서에 포함된 배열 객체
- ❖ **images** : 문서에 삽입된 그림을 배열로 제공
- ❖ **links** : 문서에 포함된 모든 링크의 이름을 배열로 제공
- ❖ **forms:<form>** 태그 입력순서대로 배열로 제공
- ❖ **anchors** : 문서에 포함된 하이퍼링크의 이름을 배열로 제공
- ❖ **Applets** : 문서에 포함된 배열들의 배열을 설정
- ❖ **Embeds** : 문서에 포함된 플러그인을 배열로 제공
- ❖ **layers** : 레이어의 배열 정보를 제공

Window, screen, document, link, anchor 객체 – document 객체의 메소드

- ❖ **clear()** : 문서의 모든 내용을 지움
- ❖ **open()** : 문서의 내용을 보여줌
- ❖ **close()** : **open()**으로 보여준 문서를 닫음
- ❖ **write()** : 태그를 포함하는 문자열을 출력
- ❖ **writeln()** : **<pre>** 태그와 함께 사용하면 행
마지막에서 자동 개행 자동 개행을 제외하면
write()와 동일한 기능을 함
- ❖ **getSelection()** : 마우스로 선택한 문자열을 반
환(넷스케이프 네비게이터에서 동작)

Window, screen, document, link, anchor 객체 – link 객체와 anchor 객체


❖ link 객체

- document.links[인덱스 번호].속성
- document.links.length
- document.링크이름.속성

❖ anchor 객체

- document.anchors[인덱스 번호].속성
- document.anchors.length
- document.anchors[책갈피명] 또는 document.all[책갈피명]

**navigator, history,
location, string 객체**



navigator 객체와 history 객체
location 객체와 string 객체

Navigator, history, location, string 객체 – **navigator** 객체의 속성

- ❖ **appName** : 브라우저의 코드명을 알려줌
- ❖ **appName** : 브라우저의 종류를 알려줌
- ❖ **appVersion** : 브라우저의 버전을 알려줌
- ❖ **userAgent**: 브라우저의 코드명, 버전, 운영체제와 같은 브라우저 정보를 알려줌
- ❖ **platform** : 시스템 코드를 알려줌

Navigator, history, location, string 객체 – history 객체의 메소드

- ❖ **back()** : 이전 페이지로 돌아감, ‘뒤로’ 이동 아이콘과 같은 역할
- ❖ **forward()** : 한 페이지 다음으로 이동, ‘앞으로’ 이동 아이콘과 같은 역할
- ❖ **go(n):n** 단계만큼 이동
 - go(정수), go(음수), go(문자열)
- ❖ **go(0)** : 현재 페이지, ‘새로고침’ 아이콘과 같은 역할
- ❖ **go(1)** : **history.forward()**와 같이 다음 페이지로 이동
- ❖ **go(-1)** : **history.back()**과 같이 이전 페이지로 이동

Navigator, history, location, string 객체 – location 객체



Navigator, history, location, string 객체 – location 객체의 속성

속성	설명
hash	# 다음에 오는 문자열, 즉 앵커 이름을 표시
host	hostname과 port번호 표시
hostname	호스트명을 표시
href	완전한 형태의 URL 주소 표시
pathname	문서의 경로 표시
port	포트번호 표시
protocol	프로토콜 종류 표시
search	검색엔진을 실행할 때 나타나는 ? 이후의 문자 표시

Navigator, history, location, string 객체 – location 객체의 메소드

- ❖ **reload()** : 문서를 다시 읽어옴(새로고침)
- ❖ **replace("URL 주소")** : 문서를 **URL** 주소로 대체하고 이전 페이지로 돌아갈 수 없게 설정(넷스케이프 네비게이터에서만 동작)

Navigator, history, location, string 객체 –

string 객체

❖ 기본 형식

- 변수 = "문자열"
- 변수.속성
- 변수.메소드

❖ "문자열".length

❖ 사용 예

- “환영합니다.”.bold()
- “환영합니다.”.bold().fontcolor(“red”)

Navigator, history, location, string 객체 –

문자열 객체에서 사용되는 글자 속성 관련 메소드

- ❖ **big() :<big>** 태그와 같이 글자크기를 크게 설정
- ❖ **small() :<small>** 태그와 같이 글자크기를 작게 설정
- ❖ **fontsize(숫자) :** 태그처럼 글자크기를 지정
- ❖ **fontcolor(“색상명”)** :글꼴색 지정
- ❖ **bold() :<bold>** 태그와 같이 글자를 진하게 설정
- ❖ **fixed() :<tt>** 태그와 같이 글자크기를 고정시킴
- ❖ **italic() :<i>** 태그와 같이 글자를 이탤릭체로 설정
- ❖ **strike() :<strike>** 태그와 같이 취소선을 설정
- ❖ **sup() :<sup>** 태그와 같이 글자를 위첨자로 설정
- ❖ **sub() :<sub>** 태그와 같이 글자를 아래첨자로 설정

Navigator, history, location, string 객체 – 문자열 객체의 정보를 가져오는 메소드(1)

❖ 문자열 위치와 관련된 메소드

❖ **indexOf(“문자”) indexOf(“문자”,n)**

- 문자열 객체 중에서 문자의 위치 값을 왼쪽부터 계산하여 숫자로 표시한다. 시작은 0번부터, 없으면 -1이 된다. 문자를 문자열의 n번째 문자부터 찾는다..

❖ **lastIndexOf(“문자”) lastIndexOf(“문자”,n)**

- 문자열 객체 중에서 문자의 위치를 오른쪽부터 계산하여 숫자로 표시한다. 시작은 0번부터, 없으면 -1이 된다. 문자를 문자열의 n번째 문자부터 찾는다.

❖ **charAt(n)**

- 문자열에서 n번째 위치한 문자를 찾아준다. 숫자는 0번부터 시작한다.

Navigator, history, location, string 객체 – 문자열 객체의 정보를 가져오는 메소드(2)

❖ 문자열에 포함된 문자 표시 방법

❖ **Substring(n,m)**

- 문자열의 n번째 문자부터 m번째 문자 까지 표시한다.
음수 값은 무시된다.

❖ **slice(n,m)**

- substring과 동일하고, 음수 값은 오른쪽부터 순번으로 계산된다.

❖ **subsrtn(n,m)**

- 문자열의 n번째 문자부터 m개의 문자를 표시한다.

Navigator, history, location, string 객체 – 문자열 객체의 정보를 가져오는 메소드(3)

❖ 문자열 분리 및 결합 메소드

❖ **split("구분문자")**

- 구분문자를 이용해서 문자열 객체를 분리시킨다.

❖ **concat("문자열")**

- 문자열을 문자열 객체에 결합시킨다.

❖ 대소문자 구분 설명

❖ **toUpperCase()**

- 모두 대문자로 표시한다.

❖ **toLowerCase()**

- 모두 소문자로 표시한다.

Navigator, history, location, string 객체 – 문자열 객체의 정보를 가져오는 메소드(4)

- ❖ **eval()**: 문자열을 수치로 바꾼다.
- ❖ **toString(n)**: 수치를 **n**진수로 바꾸어 표시한다.
- ❖ **match()** : 지정한 문자와 동일한 패턴을 찾는다. 없으면 널값 반환한다.
- ❖ **search()**: 문자열에서 지정한 문자 턴을 찾아 그 패턴의 오프셋 값(정수)을 반환한다.
- ❖ **replace()**: 지정한 문자를 찾아 지정한 다른 문자열로 바꾼다.
- ❖ **CharCodeAt(n)**: 문자열의 **n**번째 문자를 **ISO-Latin-1** 코드 값으로 표시한다.

Date, Array 객체

Date 객체
Array 객체

Date 객체
Array 객체

Date, Array 객체 – Date 객체를 생성하는 다양한 방법

- ❖ **now=new Date()**: 현재 날짜와 시각을 갖는 **now** 객체를 생성한다.
- ❖ **now=new Date(year, month, day)**: 연, 월, 일 정보를 갖는 **now** 객체를 생성한다.
- ❖ **now=new Date(year, month, day, hours, minutes, seconds)**: 연, 월, 일, 시, 분, 초의 정보를 갖는 **now** 객체를 생성한다.
- ❖ **now=new Date("month, day, year hours: minutes: seconds")**: 월, 일, 연 시:분:초의 정보를 갖는 **now** 객체를 생성한다.

Date, Array 객체 – 날짜와 시간을 표시하는 메소드

메소드명	설명
getFullYear()	1970년 이후의 년도를 구함
getMonth()	월을 구함(0~11, 1월은0, 12월은11)
getDate()	일을 구함(1~31)
getDay()	요일에 해당하는 숫자를 구함(0~6, 일요일은0, 토요일은6)
getHours()	시간을 구함(24시간 기준, 0~23)
getMinutes()	분을 구함 (0~59)
getSeconds()	초를 구함(0~59)
getTime()	1970년 1월 1일 00:00:00부터 지정한 시간까지를 1/1000초로 표시
getMilliseconds()	1/100초로 표시

Date, Array 객체 – 배열 객체 생성과 사용법

❖ 배열 객체 생성법

- 변수명=new Array(배열 개수)
- 변수명=new Array(값1, 값2, 값3....)

❖ 배열 객체 사용법

- 배열객체명.속성
- 배열객체명.메소드

Date, Array 객체 – Array 객체의 속성

속성	설명
length	배열의 개수를 나타냄
index	배열의 저장 공간을 가리키는 0부터 시작하는 정수
prototype	객체에 새로운 메소드나 속성을 추가할 때 사용
constructor	해당 객체를 만든 함수의 몸체(function body)를 반환

Date, Array 객체 – Array 객체의 메소드

메소드	설명
concat(a)	a 변수에 저장된 배열을 기존 배열에 추가
join("분리기호")	배열에 저장된 값을 분리 기호를 구분자로 사용해서 문자열로 표시
reverse()	배열에 저장된 값을 역순으로 바꿈
slice(n,m)	배열에서 n 위치부터 m 위치까지의 값으로 새로운 배열 생성
sort()	배열을 오름차순으로 정렬
shift()	배열의 첫 번째 요소를 제거하고 제거된 요소를 반환
unshift()	배열의 앞 부분에 하나 이상의 요소를 추가한 후 배열의 길이 값을 반환
pop()	배열의 마지막 요소를 제거하고 그 제거된 요소를 반환
push()	배열의 끝에 하나 이상의 요소를 추가하고 새로운 배열의 길이 값을 반환
splice()	지정한 위치에서부터 지정한 개수의 배열 요소를 제거하고 새로운 요소를 지정한 위치에 추가한 후 제거된 요소를 반환
toSource()	배열의 소스 코드에 해당하는 문자열을 반환
toString()	지정한 배열의 요소를 결합하여 하나의 문자열을 반환
value()	배열의 초기 값을 반환

frame, image, event, math,

layer 객체

frame 객체

image 객체

event 객체

math 객체

layer 객체

Frame, image, event, math, layer 객체 – frame 객체의 기본 사용법

- ❖ **window.frames**[인덱스 번호].속성
- ❖ **frames**[인덱스 번호].메소드
- ❖ **window.프레임명.속성**
- ❖ **프레임명.메소드**

Frame, image, event, math, layer 객체 – frame 객체의 속성

속성	설명
length	프레임 개수를 알려줌
name	해당 프레임의 이름을 알려줌
parent	해당 프레임을 포함하고 있는 상위 프레임을 가리킴
self	해당 프레임 자신을 의미
window	self와 같이 자신의 창을 의미

Frame, image, event, math, layer 객체 – image 객체의 기본 사용법

- ❖ **window.document.images[배열 번호].속성**
- ❖ **그림 이름.속성**

Frame, image, event, math, layer 객체 – image 객체의 기본 속성

- ❖ **name**:그림에 설정된 **name** 속성 값을 알려줌
- ❖ **Length**:문서에 삽입된 그림의 개수를 알려줌
- ❖ **src** :그림의 **src** 속성에 설정된 경로를 알려줌
- ❖ **lowsrc**:그림의 **lowsrc** 속성에 설정된 저해상도 그림의 경로를 알려줌
- ❖ **hspace**:그림의 **hspace** 속성에 설정된 좌우 여백 값을 알려줌
- ❖ **vspace** :그림의 **vspace** 속성에 설정된 상하 여백 값을 알려줌
- ❖ **width**:그림의 **width** 속성에 설정된 너비 값을 알려줌
- ❖ **height**:그림의 **height** 속성에 설정된 길이 값을 알려줌
- ❖ **border**:그림의 **border** 속성에 설정된 테두리 두께를 알려줌
- ❖ **complete**:그림 전송이 완료되면 **true**, 그렇지 않으면 **false** 값을 반환

Frame, image, event, math, layer 객체 – event 객체의 속성(1)

- ❖ **altKey** : **alt** 키를 누르면 **true**값 발생
- ❖ **altLeft** : 왼쪽 **alt** 키를 누른 경우 이벤트 발생
- ❖ **button** : 마우스 버튼을 누른 경우 이벤트 발생선택한 마우스 버튼의 종류에 따라 다른 값 전달 (왼쪽=1, 오른쪽=2, 왼쪽+오른쪽=3, 가운데=4, 왼쪽+가운데=5, 가운데+오른쪽=6, 모두 누른 경우=7)
- ❖ **clientX** : 윈도우 영역에서 마우스의 **x** 좌표 값
- ❖ **clientY** : 윈도우 영역에서 마우스의 **y** 좌표 값
- ❖ **ctrlKey** : **ctrl** 키를 누르면 **true**값 발생
- ❖ **ctrlLeft** : 왼쪽 **ctrl** 키를 누른 경우 이벤트 발생
- ❖ **fromElement** : **onmouseover**나 **onmouseout** 이벤트를 사용하는 경우의 마우스 객체를 가리킴
- ❖ **key** : **Code** 키를 눌렀을 경우의 키의 유니코드 값을 설정
- ❖ **keyCode** : 키보드의 키 값을 전달받음
- ❖ **offsetX** : 이벤트가 발생한 객체에서 마우스의 **x** 좌표 값
- ❖ **offsetY** : 이벤트가 발생한 객체에서 마우스의 **y** 좌표 값

Frame, image, event, math, layer 객체 – event 객체의 속성(2)

- ❖ **propertyName**: 이벤트를 발생한 객체에서 변경된 속성 이름을 설정
- ❖ **repeaton:keydown** 이벤트의 반복 횟수 설정
- ❖ **reason** :데이터의 전송 상태
- ❖ **returnValue** :이벤트에서 발생한 값을 **true** 또는**false**로 설정
- ❖ **screenX** :전체 화면에서 마우스의 **x** 좌표 값 설정
- ❖ **screenY** :전체 화면에서 마우스의 **y** 좌표 값 설정
- ❖ **shiftKey** :**shift** 키를 누르면 **true** 값을 발생
- ❖ **shiftLeft**:왼쪽 **shift** 키를 누른 경우 이벤트 발생
- ❖ **srcElement**:이벤트를 발생한 객체를 가리킴
- ❖ **srcFilter:onfilterchange** 이벤트를 발생시킨 **filter**객체를 설정
- ❖ **toElement**:마우스 포인터가 위치해 있는 객체를 설정
- ❖ **type** :이벤트객체의 이벤트 종류를 설정
- ❖ **X** :선택한 객체의 상대적인 **x**좌표 값을 설정
- ❖ **Y** :선택한 객체의 상대적인 **y**좌표 값을 설정

Frame, image, event, math, layer 객체 – math 객체의 속성

속성명	설명
E	오일러 상수(약2.718)
PI	원주율 값(약3.1415)
LN2	2의 자연로그 값(약0.693)
LN10	10의 자연로그 값(약2.302)
SQRT2	2의 제곱근 값(약1.414)
SQRT1_2	1/2 제곱근 값(약0.707)
LOG2E	밑이 2인 로그 값(약 1.442)
LOG10E	밑이 10인 로그 값(약 0.434)

Frame, image, event, math, layer 객체 – math 객체의 메소드(1)

- ❖ **sin(x)** : 사인 값
- ❖ **cos(x)** : 코사인 값
- ❖ **tan(x)** : 탄젠트 값
- ❖ **acos(x)** : 역코사인(아크코사인) 값
- ❖ **asin(x)** : 역사인(아크사인) 값
- ❖ **atan(x)** : 역탄젠트(아크탄젠트) 값
- ❖ **atan2(x,y)** : 역탄젠트
- ❖ **exp(x)** : **X**의 지수함수
- ❖ **log(x)** : **X**의 로그함수
- ❖ **pow(x,y)** : 지수함수 **f(x,y)=xy**
- ❖ **sqrt(x)** : 제곱근 값

Frame, image, event, math, layer 객체 – math 객체의 메소드(2)

- ❖ **round(x)** : x 반올림
- ❖ **abs(x)** : 절대 값
- ❖ **ceil(x)** : x보다 같거나 큰 수 중에서 가장 적은 정수
- ❖ **floor(x)** : x보다 같거나 작은 수 중에서 가장 큰 정수
- ❖ **random()** : 0~1 사이의 난수 발생
- ❖ **min(x,y)** : x, y 중 큰 수
- ❖ **max(x,y)** : x, y 중 작은 수

Frame, image, event, math, layer 객체 –

레이어 다루기

❖ 레이어의 스타일 속성 사용법

- `document.all[id명].style.스타일 속성`

❖ 레이어 보이거나 감추게 하기

- `document.all[id명].style.visibility=visible/hidden`

❖ 레이어 이동하기

- `document.all[id명].style.left=값`
- `document.all[id명].style.top=값`
- `document.all[id명].style.left=event.clientX`
- `document.all[id명].style.top=event.clientY`

폼 관련 객체



forms 객체

입력상자

체크상자, 라디오 버튼

목록상자

폼 관련 객체 – **form** 객체

- ❖ **document.폼이름.속성**
- ❖ **document.forms[배열번호].속성**
- ❖ **document.폼이름.elements[배열번호].속성**

폼 관련 객체 – **form** 객체가 제공하는 속성

속성	설명
action	<form> 태그의 action 속성에 기록된 정보를 알려줌
elements	입력상자, 라디오 버튼, 체크 버튼 등 폼 양식을 배열로 정의
encoding	<form> 태그의 encoding 속성에 기록된 정보를 알려줌
method	<form> 태그의 method 속성에 기록된 정보를 알려줌
target	<form> 태그의 target 속성에 기록된 정보를 알려줌
length	폼 양식의 개수를 알려줌
name	<form> 태그의 name 속성에 기록된 정보를 알려줌

폼 관련 객체 – **form** 객체의 메소드와 이벤트

❖ **form** 객체가 제공하는 메소드

- Blur() 커서를 사라지게 함
- Reset() 폼 양식에 입력된 값을 초기화
- Submit() 폼 양식에 입력된 값을 지정된 서버로 보냄

❖ **form** 객체에서 사용하는 이벤트

- Onreset 리셋 버튼을 누르면 이벤트가 발생한다.
- Onsubmit 제출 버튼을 누르면 이벤트가 발생한다.

폼 관련 객체 – **select** 객체의 속성

속성	설명
type	multiple 정보를 가져옴
length	목록의 개수를 알려줌
options	<select> 태그 안에 포함된 <option> 태그를 배열로 구성
selectedIndex	목록을 배열 번호로 표시하거나 배열 번호를 가져옴

폼 관련 객체 – **options** 속성 정보를 알려주는 속성

❖ 목록의 항목을 선택해 주는 **options** 속성

- 폼 이름.select이름.options[배열번호]

속성	설명
defaultSelected	선택한 목록이 <option> 태그에 selected 속성을 사용한 목록인지 확인
index	선택한 목록의 배열 번호를 가져옴
selected	선택한 목록이 선택되었는지 확인
text	선택한 목록에 입력된 내용을 가져옴
value	선택한 목록에 사용된 value 속성 값을 가져옴

폼 관련 객체 – 목록상자에 항목 추가/삭제하기

❖ 목록상자에 항목 추가하기

- 변수=new Option(내용, value 속성 값, 초기선택 상태, 선택)

❖ 목록상자에 항목 삭제하기

- form.newAdd.options[n] =null

자바스크립트

활용하기



스타일 제어

자바스크립트 활용하기

자바스크립트 무작정 베껴쓰기

자바스크립트 활용하기 – 스타일 제어하는 법

❖ 문서에 삽입된 스타일을 제어하는 법

- `document.객체명.style.스타일 속성`

❖ 레이어 보이거나 감추게 하기

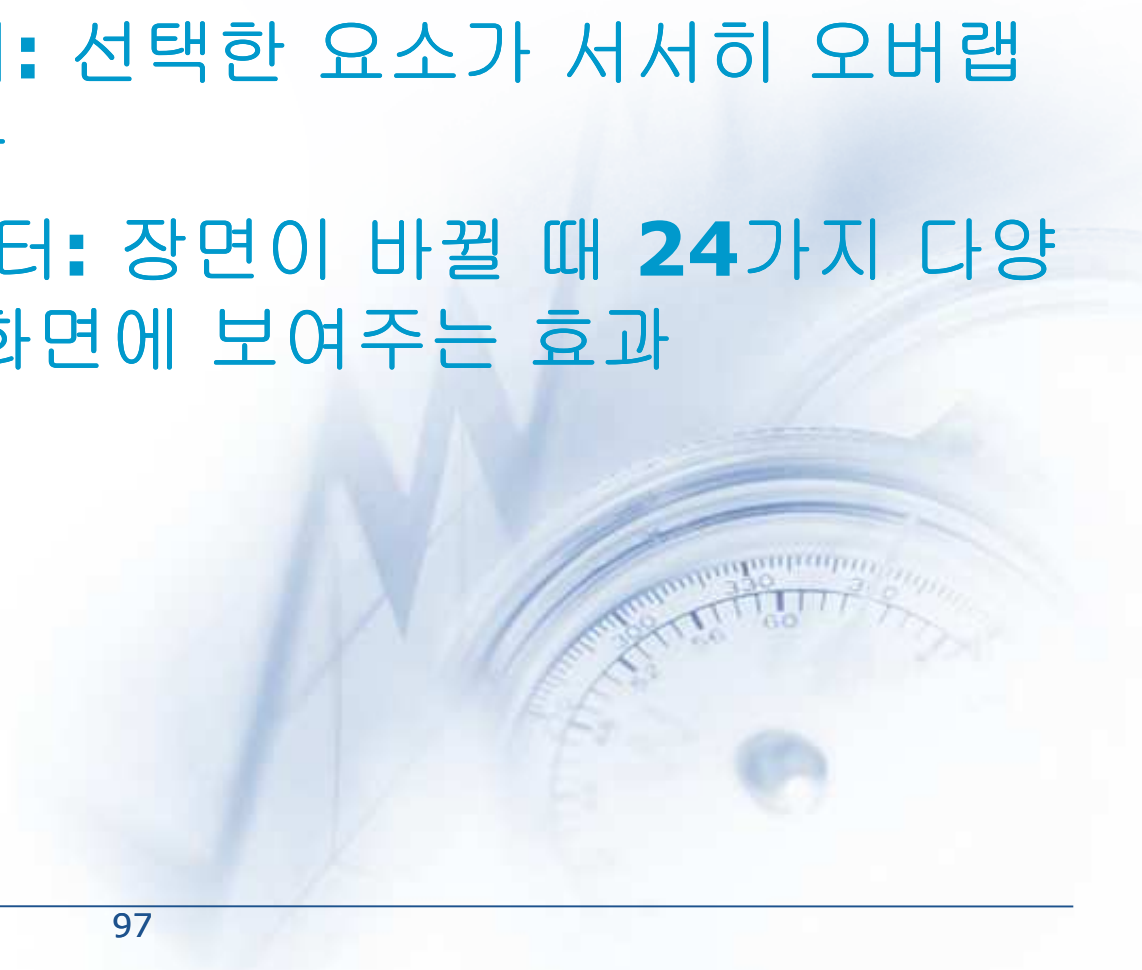
- `document.all[id명].style.visibility=visible/hidden`



자바스크립트 활용하기 -

blendTrans 필터와 **RevealTrans** 필터



- ❖ **blendTrans** 필터: 선택한 요소가 서서히 오버랩 되면서 바뀌는 효과
 - ❖ **RevealTrans** 필터: 장면이 바뀔 때 **24**가지 다양한 시각적 효과를 화면에 보여주는 효과
- 

자바스크립트 활용하기 – JS 외부 파일 연결법

- ❖ **js** 확장자로 저장된 자바스크립트 소스를 문서에 연결하는 법
- ❖ **<script language="javascript" src="js 확장자를 가지는 파일의 전체 경로"></script>**

자바스크립트 활용하기 -

자바스크립트 소스 베껴쓸 때의 주의사항

- ❖ **<head>** 태그 사이에 있는 소스는 복사해서 **<head>** 태그 사이에 붙여넣는다.
- ❖ **<body>** 태그 사이에 있는 소스는 복사해서 **<body>** 태그 사이에 붙여넣는다.
- ❖ **<body>** 태그에 함수 호출이 있을 경우 같이 호출해 주어야 한다.