JavaScript

World Wide Web

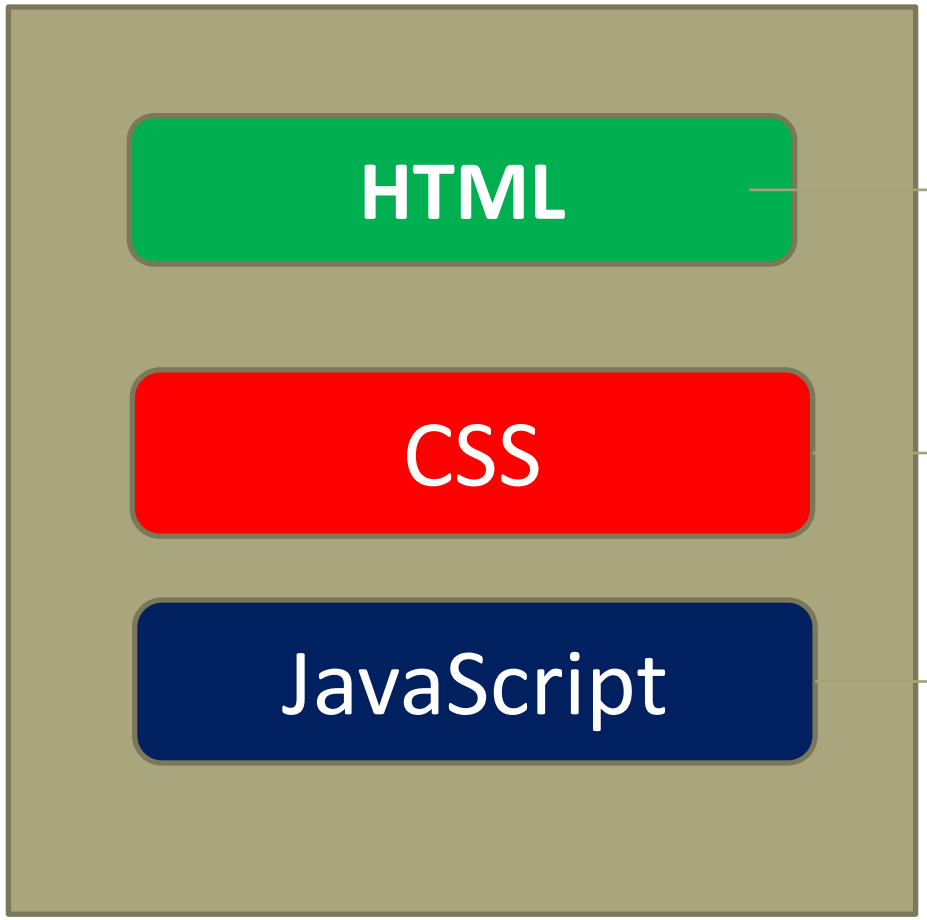
- “웹은 모든 사람들이 손쉽게 정보를 공유할 수 있는 공간이며 어떤 장애도 없이 이를 이용할 수 있어야 한다.”
- **-팀 버너스리 (www 창시자)**

1. 웹표준이란?

- 특정 브라우저에서만 사용하는 비 표준화된 기술을 배제하고, W3C라는 조직에서 정한 표준 기술만을 사용하여, 웹 문서의 구조와 표현 그리고 동작을 구.분해서 사용하는 것.
- 웹문서의 3요소

- 1. 구조 담당=> HTML
- 2. 표현 담당 => CSS
- 3. 동작 담당=> JAVASCRIPT

2. 웹의 3가지 계층



웹페이지의
내용 담당
[구조 계층]

웹페이지의
디자인 담당
[표현 계층]

웹페이지의
동작 담당
[동작 계층]
이벤트처리,
요소에 대한 동적
CRUD등

3. 자바스크립트란?

- JavaScript는 넷스케이프 사의 브렌던 아이크(Brendan Eich)가 처음에는 모카(Mocha)라는 이름으로, 나중에는 LiveScript라는 이름으로 개발하였으며, 최종적으로는 JavaScript라는 이름으로 발표됨.

- 객체기반의 스크립트 프로그래밍 언어로 웹 페이지 개발 시 동적인 처리를 구현하기 위해 주로 사용한다.

- 프로그래밍 언어로 저평가 받는 시기도 있었으나 Rich Content를 작성할 수 있는 Ajax(Asynchronous JavaScript+XML)의 등장으로 인해 자바스크립트의 가치는 재검토 되었다.
- HTML5에서 HTML5의 API로 JavaScript를 공식 채택

3. 자바스크립트란?

- 클라이언트 및 인터넷 응용 프로그램 개발을 위한 객체-지향 스크립팅 언어
- 스크립팅(Scripting) 언어
- 인터프리터(Interpreter)에 의해 실행됨
- 손쉬운 프로그램 개발과 제한적이지만 배우기 쉬운 명령어들과 문장 구조
- 잘 정의된 일련의 작업을 수행하는 프로그램의 개발이 용이함
- 단순하고 작은 프로그램을 위한 언어
- 단순하고 작은 프로그램 개발에 적합
- 분산된 CAD 문서 출력과 관리와 같은 환경에는 부적합
- 반복적인 작업 수행
- 반복적이고 발생 사건(Event)에 의해 수행되는 일에 적합

4. 웹Application의 언어 JavaScript

- **ECMAScript란?**
- 웹의 발전과 함께 브라우저 회사들의 과도한 경쟁으로 크로스 브라우징 구현이 어려웠으나, 국제적인 표준화 단체인 ECMA에서 착실하게 표준화를 진행하여 언어로서의 완성도를 높여 나감
- **ECMAScript는 ECMA 터네셔널의 ECMA-262 기술규격에 정의된 스크립트 프로그래밍 언어의 표준화된 스펙** 이다.
- 2016년 6월 ECMAScript 6판이 발표되었으며 현재 7판이 진행중.
- 현재 대부분의 브라우저들이 지원하는 JavaScript는 ECMAScript 6표준을 따르고 있다.

5. 자바스크립트의 장점

- 빠른 개발이 가능하다
- 컴파일 과정이 필요 없다
- 배우기가 쉽다
- 자바와 유사한 점도 많지만 자바의 복잡한 문장 구조와 규칙들을 채택하지 않음

- 웹 서버에 주는 부담이 적다 : 성능향상
- 자바스크립트 프로그램이 HTML 코드와 동일한 파일에 존재하기 때문에 스크립트 전송을 위한 별도의 네트워크 접속이 필요 없다

6. 자바스크립트의 활용범위

- **[1] 웹 클라이언트 개발**
- 웹이 발전하면서 서버에서 처리되던 기능들이 클라이언트로 이동되었으며, HTML5에서는 웹 클라이언트에서 처리하려는 기능들을 표준적인 방법으로 구현할 수 있게 지원하는 API들을 JavaScript로 제공한다.
- **[2] 웹 서버 개발**
- Node.js의 출현으로 JavaScript를 활용한 서버 개발도 가능하게 되었다. Express, socket.io등의 라이브러리는 보다 쉽게 JavaScript로 서버를 개발할 수 있는 환경을 제공해준다.
- **[3] 애플리케이션 개발**
- 웹이 하나의 플랫폼으로 진화하면서, 웹os를 표방한 여러 가지 프로젝트가 진행되고 있다. 구글의 크롬os를 비롯, hp의 웹os 등 다양한 웹기반 플랫폼에서 구동되는 애플리케이션 개발에 JavaScript는 없어서는 안될 핵심 언어가 됨

7. 자바스크립트의 핵심 개념

- [1] 객체 (object)
- JS의 거의 모든 것은 객체이다. 기본 데이터타입인 boolean, number, string, null, undefined를 제외하고 나머지는 모두 객체이다.
- 또한 앞의 세 가지 기본 자료형도 모두 객체로 다룰 수도 있다.
- [2] 함수 (function)
- JavaScript에서는 함수도 객체로 취급한다. 일반적인 객체보다 조금 더 많은 기능이 있는 객체라고 할 수 있다.
- [3] 프로토타입 (prototype)
- 모든 객체는 숨겨진 링크인 프로토타입을 가진다. 이 링크는 해당 객체를 생성한 생성자의 프로토타입 객체를 가리킨다.

8. JavaScript 정의 방법

- JavaScript코드는 html의 어느
- 부분에 삽입해도 가능하나
- 주로 필요에 따라 <head>태그
- 나 <body>태그에 삽입하게
- 된다.

- 자바스크립트 주석
- : 스크립트 영역 안에서 주석
- 처리
- [1] 단문 주석: //
- [2] 복문 주석: /* */

- Html영역에서는 html주석을
- <!-- --> 사용.

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>script 태그 위치</title>
        <!--스크립트 태그 위치1-->
        <script>
          alert("Hi JavaScript~");
        </script>
    </head>
    <body>
        <h1>Hello JavaScript</h1>
        <!--스크립트 태그 위치2-->
        <script>
          alert("즐거운 시간 되세요~ JavaScript~");
        </script>
    </body>
</html>
```

[1] HTML 문서 안에 자바스크립트 삽입하기

HTML 문서 안에 자바스크립트 삽입하기

<SCRIPT type="text/javascript"> ..... </SCRIPT>를 이용
자바스크립트를 HTML 파일에 삽입하기

***<SCRIPT* type="text/javascript">**
.... JavaScript 프로그램 ....
</SCRIPT>

실습 1 : 안녕하세요 JavaScript

- 실습1
  - document객체의 write라는 메서드를 사용해서 화면에 "안녕하세요 JavaScript"를 출력시켜보자.
  - 현재 document객체니 write메서드는 배우지 않았다. 하지만, 바로 작성을 해보면 알 수 있을 것이다.

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>실습 자바스크립트</title>
    </head>
    <body style="background-color: yellowgreen">
        <h1>Hello JavaScript</h1>
        <script type="text/javascript">
            /*자바스크립트 주석.
            여러 줄 주석 처리가 가능해요*/
          document.write("안녕하세요? 자바스크립트")
            //이건 단문 주석입니다.
        </script>
    </body>
</html>
```

[2] HTML 문서 안에 자바스크립트 삽입하기 - 외부js파일 참조하기

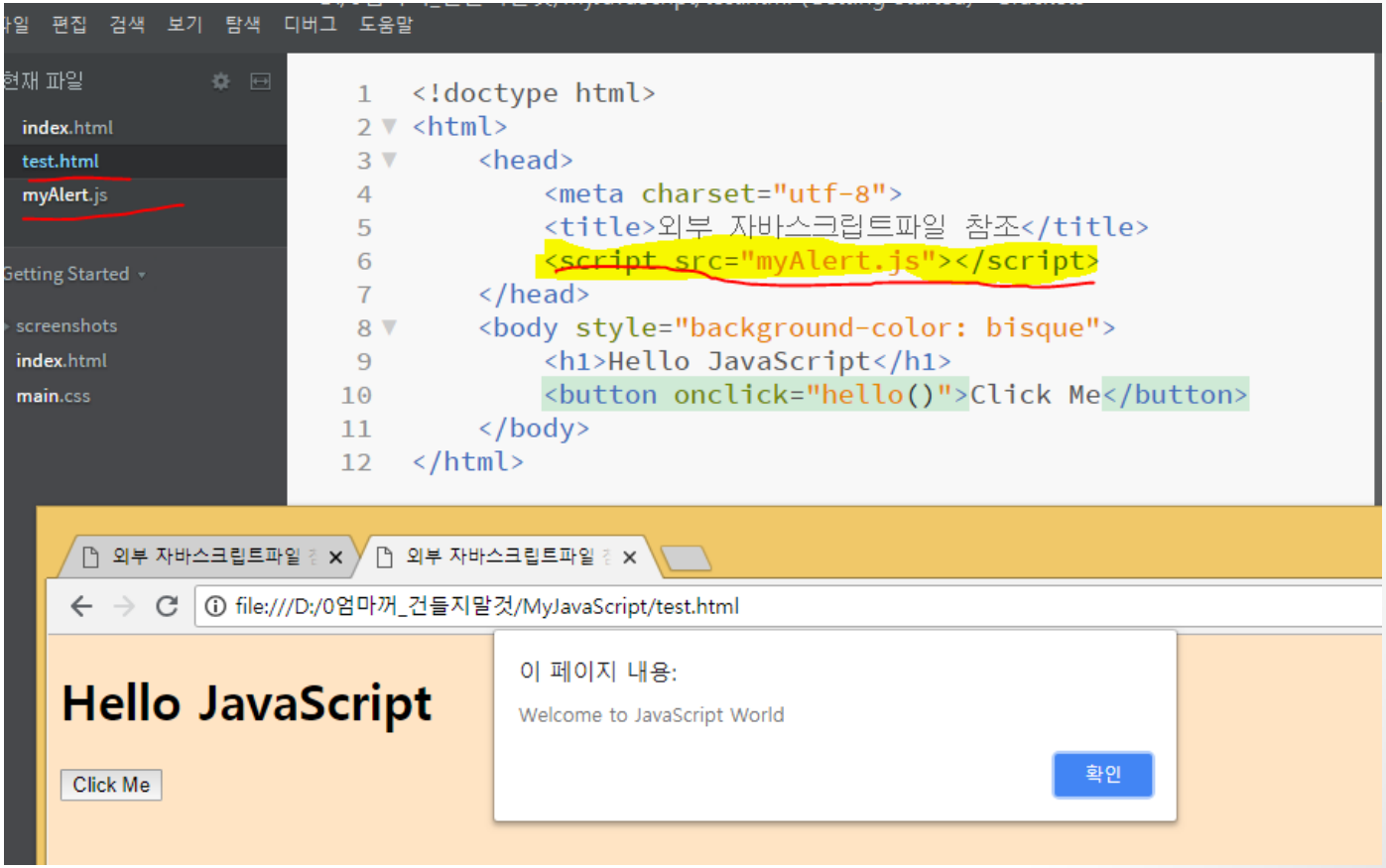
외부 파일에 존재하는 자바스크립트 프로그램 이용하기

<SCRIPT type="text/javascript"
SRC="http://www.my.com/myScript.js">
</SCRIPT>

SRC 속성이 기능을 발휘하기 위해서는 자바스크립트를 포함하고 있는 파일의 이름이 반드시 '.js'로 끝나야 한다

파일명: myAlert.js

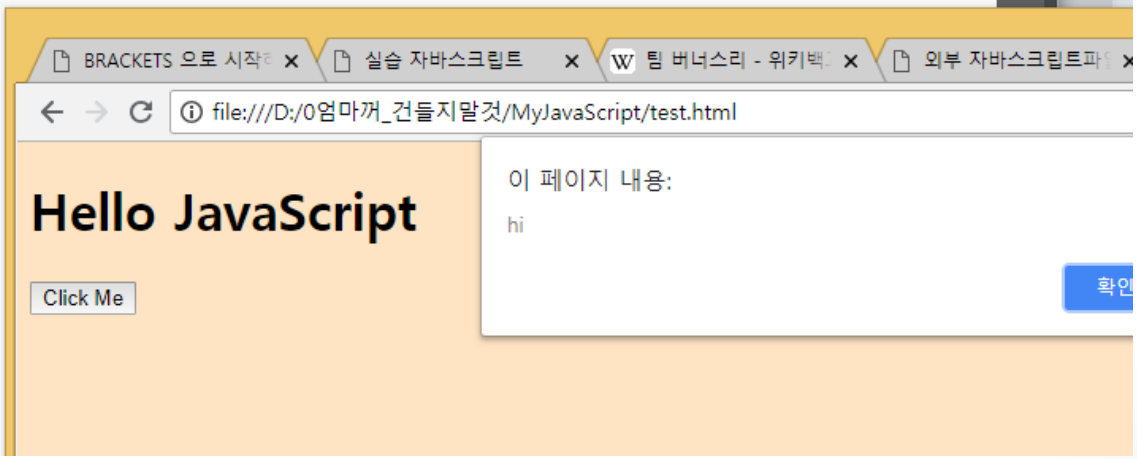
test.html에서 myAlert.js 파일 참조



[3] HTML 문서 안에 자바스크립트 삽입하기

- HTML 태그의 속성(이벤트처리기)에 속성값(자바스크립트 코드)으로 사용
- <태그명 이벤트핸들러="자바스크립트코드"> … </태그명>

```
1  <!doctype html>
2  <html>
3      <head>
4          <meta charset="utf-8">
5          <title>외부 자바스크립트파일 참조</title>
6      </head>
7      <body style="background-color: bisque">
8          <h1>Hello JavaScript</h1>
9          <button onclick="alert('hi')">Click Me</button>
10     </body>
11 </html>
```



복습

- HTML에 자바스크립트 삽입
  - <Script>…</Script> 사용 : <head> 또는 <body>태그내에 사용
  - 외부파일에 자바스크립트 소스코드 작성 : 확장자는 반드시 *.js
  - 태그내에 자바스크립트 코드 적용(이벤트 핸들러 사용) : on이벤트명

9. 데이터 타입

- Javascript는 데이터 타입이 느슨한 언어이긴 하지만 처리될 수 있는 데이터 타입이 다양하게 지원된다.
- 데이터 타입 종류

기본형 (primitive)	**number, boolean, string, undefined, null**
참조형 (reference)	object(배열 포함), function 등

9. 데이터 타입

자바스크립트 기초 – 자료형(데이터 유형)

- 자료형(데이터 유형)
  - 정수형
    - 10진수, 16진수, 8진수의 3가지 형태(양수나 음수)
  - 부동소수점(실수형)
    - 12.43, -4.555 와 같이 소수 부분이 있는 10진수.
    - 지수를 사용하여 자리수가 큰 수도 표현. 예를 들어 2×10^4 같은 숫자는 2E4 로 표현.
    - 가끔 부동소수점의 계산이 정확하지 못한 경우가 있으므로 가급적이면 정수형을 사용
  - Boolean
    - 크기가 1비트인 자료형으로 true와 false의 두 가지 값만을 가지며 주로 함수의 리턴 값이나 관계 연산자의 결과값으로 사용
  - null
    - 아무것도 없다는 것을 의미, 예외처리에 자주 사용
    - 프롬프트 대화상자에서 사용자가 “Cancel” 버튼을 선택할 경우
  - 문자열
    - 따옴표(“”, ‘ ‘)에 들어가는 모든 문자

10. 변수란?

- Variable, field, property…
- 변수는 물건을 보관했다가 필요할 때 다시 꺼내 사용하는 일종의 창고이다. 즉 데이터를 보관하는 장소.
- 데이터를 저장하는 장소. 즉 데이터를 읽고 쓸 수 있는 장소
- 프로그램 내에서 특정 자료형의 값을 가지고
- 있는 저장 장소를 의미.

- 변수선언 형식

var 변수명;

var 변수명 = 초기값;

11. 변수의 규칙

- 변수의 규칙
  - 변수는 반드시 알파벳 문자나 _ 로 시작.

    (예 : var aaa, ABc112, _finder, 단 자바스크립트는 대문자와 소문자를 구별한다)
  - 예약어는 변수로서 사용할 수 없음.(다음 슬라이드에 예약어 표시)
- 변수의 선언
  - 변수에 값 할당시 : str="JavaScript", count=1 등
  - var 명령어를 사용하여 선언
  - Var를 이용한 방법 : var count = 1
    - Var를 이용하면 같은 이름의 변수라도 현재 자신이 속해있는 영역 안에서만 효력이 발생됨.
  - 변수 선언시 데이터형을 같이 표기해주지 않는다.
  - 대/소문자 구별

12. 자바스크립트 keyword

- 자바스크립트 예약어의 종류

예약어 종류						
abstract	boolean	break	byte	case	catch	
char	class	const	continue	default	do	
double	else	extend	false	final	finally	
float	for	function	goto	if		
implements		import	in	instanceof		intinterface
long	native	new	null	package	private	
protected		public	return	short	static	super
switch	synchronized		this	throw	thorws	transient
true	try	var	void	while	with	

13. 자바스크립트 데이터 처리

- 데이터형 변환(casting)
  - 데이터 유형에 대한 제약이 비교적 약한 언어
  - 변수의 유형을 변수를 생성할 때 결정하지 않아도 되며, 변수의 내용에 따라 변수의 유형이 변경될 수 있다
  - 선언되어 있는 형태와 다른 데이터형을 할당해도 대부분 치명적인 문제가 생기지 않는다

    예) 3.5 + “10” = 3.510, “Count to” + 10 = Count to 10

14. 연산자 – 산술연산자,연결연산자

- 연산자(Operator)
  - 산술연산자
  - 연결연산자
  - 관계연산자
  - 논리연산자
  - 조건연산자
  - 비트연산자
  - 증감연산자
  - 대입연산자

- 산술연산자

더하기	a + b	빼기	a - b
곱하기	a * b	나누기	a / b
나머지	a % b		

- 연결연산자
  - '+' 기호를 사용해 문자열과 숫자를 연결

14. 연산자 – 관계 연산자

- 관계(비교)연산자
  - 두 피연산자를 조건에 따라 선택적으로 실행할 수 있도록
  - 참이면 **true**, 거짓이면 **false** 값을 반환

==	a와 b가 같다	a == b	!=	a와b가 같지않다	a!=b
>	a보다b가 작다	a > b	<	a보다b가 크다	a < b
>=	a보다b가 작거나 같다	a >= b	<=	a보다b가 크거나 같다	a <= b

14. 연산자 – 논리 연산자

- 논리연산자
  - **true**와 **false** 값을 가짐.
  - 연산자 논리표

좌변	우변	연산결과		
		||	&&	^
false(0)	false(0)	0	0	0
false(0)	true(1)	1	0	1
true(1)	false(0)	1	0	1
true(1)	true(1)	1	1	0

계산	boolean 연산
Or-ret = (3>6) || (4 < 8)	true = false || true
and_ret = (3>6) && (4 < 8)	false = false && true
Xor_ret = (3>6) ^ (4 < 8)	1 = false ^ true

14. 연산자 – 조건 연산자(삼항연산자), 비트연산자

- 조건연산자(3항연산자)
  - 조건에 따라 두 값 중에 하나를 연산결과로 전달
  - 형식 : (조건) ? 변수1 : 변수2
    - 조건인 참이면 앞 문장 수행, 거짓이면 뒷 문장 수행
- 비트식 연산자
  - 비트는 0과1값을 갖는 최소의 단위로서 좌변과 우변의 비트값에 따라 or, and, xor이 결합되어 다음과 같은 논리값을 가짐.

좌변	우변	연산결과		
		|	&	^
false(0)	false(0)	0	0	0
false(0)	true(1)	1	0	1
true(1)	false(0)	1	0	1
true(1)	true(1)	1	1	0

14. 연산자-증감연산자

- !(not) 연산자는 값이 0인 경우는 1로 1인 경우는 0으로 바꿔줌.
- ~연산자는 1의 보수로 만들어줌.
- 논리연산자는 ||, &&.... 비트식 연산자는 한 개씩임.
- >>, <<, >>> 연산자는 비트를 이동하는 연산자

이동연산자	설명
<<	왼쪽으로 비트 이동
>>	오른쪽으로 비트 이동, 양수와 음수에 상관없이 왼족 끝 한 비트는 항상 0으로 채워짐.
>>>	음수인 경우에는 왼쪽 끝 한 비트가 1로 채워지고 양수인 경우에는 왼쪽 끝 한 비트가 0으로 채워짐.

- 증감연산자
  - 편리하게 1씩 증가하거나 1씩 감소하는데 사용
  - ++, --로 표시 (++A와, A++의 차이점?)

14. 연산자-대입연산자

- 대입연산자(=, +=, -=, *=, /=, %=)
  - 우변의 값을 좌변의 변수에 할당하는 연산자
  - += : 앞에 변수에다 뒤의 변수값을 누적

```
a = 5; b = 10;
a += b;
//a = a + b;   //+=연산자와 같다.
document.write(a); // a의값은? 15
```

- 연산자 우선순위
  - 우선순위를 모르면 잘못된 식이나 결과를 만들어 버그의 원인이 됨.
  - 우선 순위를 잘 모를 경우에는 괄호를 사용
  - 자바스크립트의 연산자는 C언어의 연산자와 비슷함.

15. 연산자 우선순위

종류	연산자	우선순위
괄호/대괄호	., [], ()	1
부정/증감	!, ++, --	2
산술	*, /, %, +, -	3
비트식	<<, >>, >>>	4
관계	<, >, <=, >=	5
	==, !=	6
비트식	&	7
	^	8
	|	9
논리	&&	10
	||	11
	? :	12
대입/할당	=, +=, -=, *=, >>=, ^=, ….	13

16. 제어문

- [1] 조건문
  - <1> if, If ~ else, if ~ else if ~ else
  - <2> switch ~case
- [2] 반복문 (loop)
  - <1> for 루프
  - <2> while 루프
  - <3> do ~ while 루프
- [3] 보조 제어문
- <1> break
  - <2> continue

16. 제어문 – [1] 조건문

- 조건문
  - 조건을 판단한 후 조건에 따라 분기하여 수행
  - if 문
    - 조건을 만족하면 문장을 실행하고 다음 라인을 수행
    - if 문의 조건을 만족하지 않으면 수행을 한 번도 하지 않음.
    - 형식

```
if (조건식)
{
    문장;
}
```

조건문

- if ~ else 문
  - 형식

```
if (조건)
{
    문장 1 ;
}
else
{
    문장 2 ;
}
```

  - 조건을 만족하면 조건 다음의 문장1을 수행하고 조건을 만족하지 않으면 else 다음의 문장2를 수행

실습 : if_else문_성적확인프로그램

- 실습
  - 문제 : 입력상자에서 입력된 값이 60보다 작으면 “F입니다.”, 100보다 크면 “잘못 입력하였습니다.”를 출력하는 자바스크립트 작성

조건문

- 다중 if 문
  - 형식

```
if (조건1)
{
   문장 1 ;
}
else if (조건2)
{
   문장 2 ;
}
        (else if를 붙여 연속적으로 만들 수 있음)
else
{
   기본문장;
}
```

실습 : 다중if문_ 성적확인프로그램

- 실습
  - 문제 : 성적이 90에서100 사이면 ”A”와 “잘했습니다.”, 성적이 80에서 90사이이면 “B”와 “보통입니다.” 70보다 작으면 “F”와 “좀 더 노력하세요”를 출력하는 자바스크립트를 작성하시오.

조건문(선택문)

switch 문

- 특징
  - 다중 if문을 대신하여 switch문 사용
  - 변수에 따라 선택적으로 [상수]로 분기하므로 소스를 이해하기 편하고 작성도 용이
- 형식

```
switch (표현식)
{
case 값1 : 문장 1 ; break;
case 값2 : 문장 2 ; break;
case 값3 : 문장 3 ; break;
……
default : 문장 n;
}
```

- Switch문 활용하기
  - Case : 선택적으로 분기하여 수행하기 위해 사용
  - Break : switch()문을 중지

16. 제어문 – [2] 반복문

반복문 : 구간반복

- for 문
  - 특징
    - [시작값], [최종값], [증가값]을 설정하여 반복적인 수행을 할 수 있는 문장
    - 규칙적인 증가를 하는 경우에 많이 사용
  - 형식

```
for (초기값; 최종값; 증가식)
{
        문장 ;
}
```

    - 한 문장 이상인 경우에는 '{'와 '}' 기호를 이용하여 문장을 묶어줌.

- 실습 1
- 1부터 100까지의 수를 화면에 출력시켜보세요

- 실습 2
- 1부터 100까지의 합을 구하는 프로그램을 작성하세요.
- (+=연산자 사용)

- 실습 3
- 입력 대화상자를 통하여 입력된 수의 제곱을 구하는 프로그램을 작성하세요.

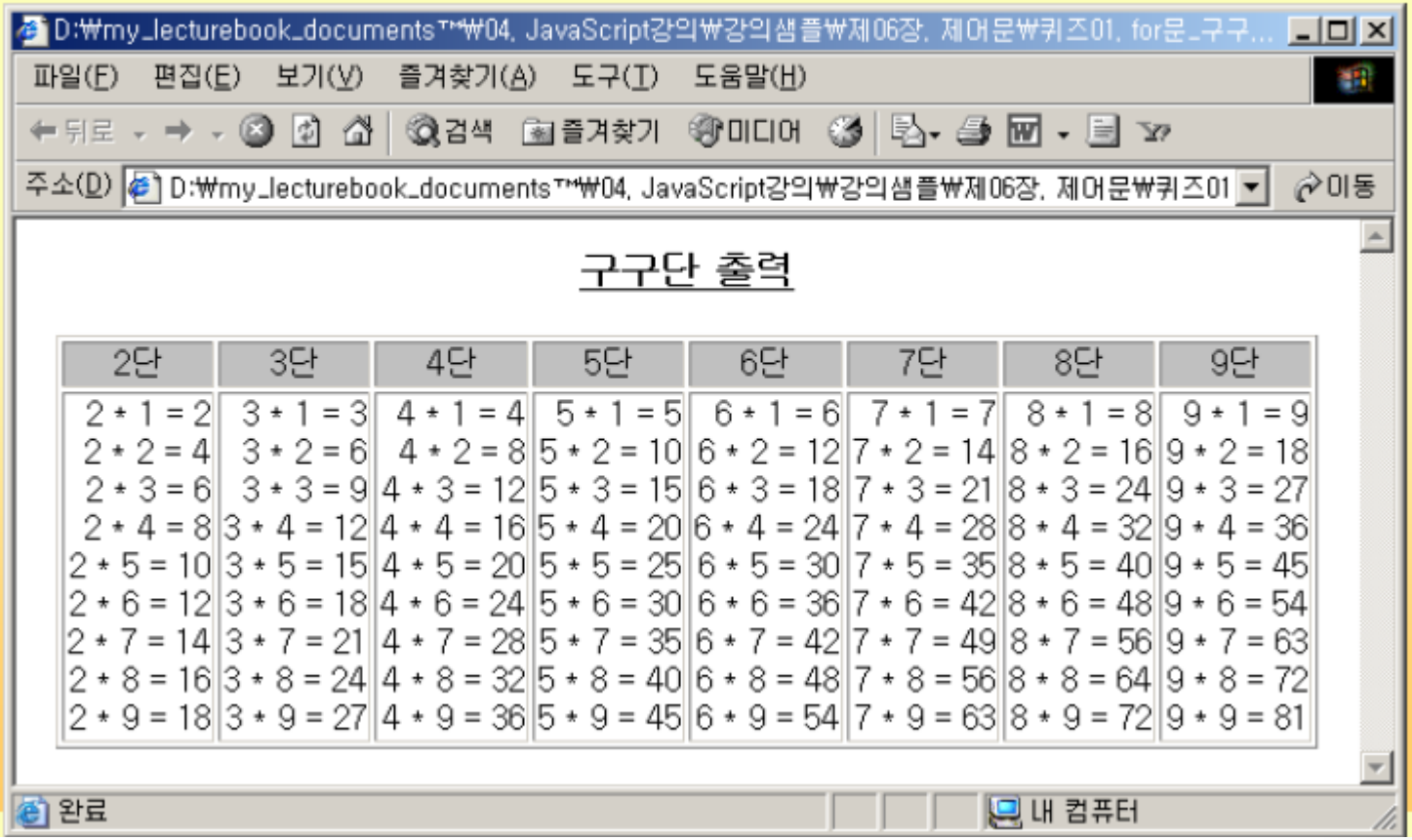
반복문

- 이중 for 문
  - 안쪽의 for문과 바깥쪽의 for 문이 쌍을 이루어 반복 수행
  - 형식

```
for (초기값; 최종값; 증가식)
{
        문장 ;
        for (초기값; 최종값; 증가식)
        {
                문장 ;
        }
}
```

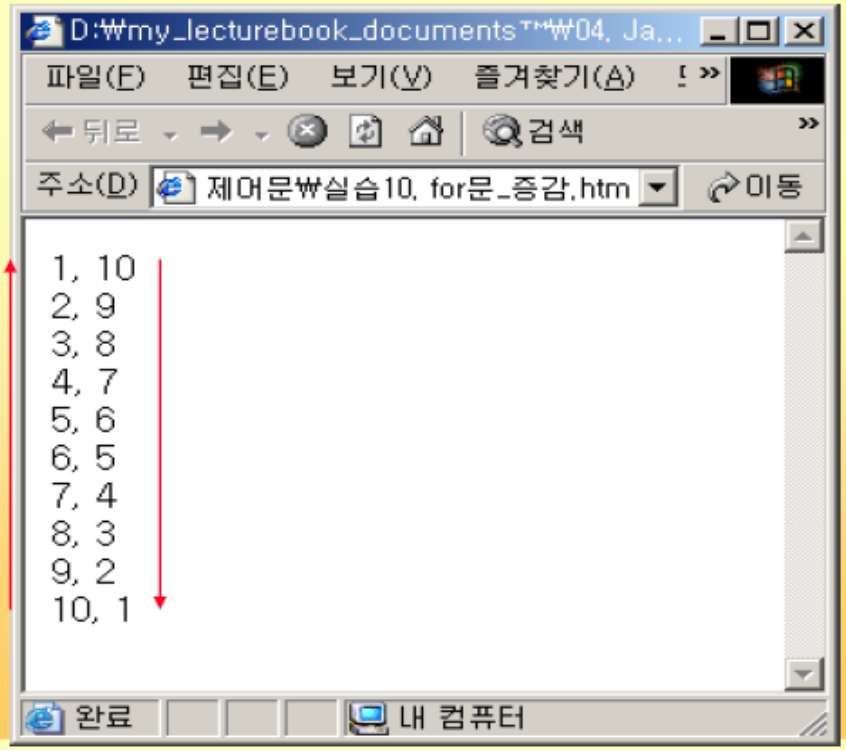
중첩 for루프 실습

- 다중 for문을 이용해서 구구단을 출력하는 프로그램을 작성해 보자.



2단	3단	4단	5단	6단	7단	8단	9단
2 * 1 = 2	3 * 1 = 3	4 * 1 = 4	5 * 1 = 5	6 * 1 = 6	7 * 1 = 7	8 * 1 = 8	9 * 1 = 9
2 * 2 = 4	3 * 2 = 6	4 * 2 = 8	5 * 2 = 10	6 * 2 = 12	7 * 2 = 14	8 * 2 = 16	9 * 2 = 18
2 * 3 = 6	3 * 3 = 9	4 * 3 = 12	5 * 3 = 15	6 * 3 = 18	7 * 3 = 21	8 * 3 = 24	9 * 3 = 27
2 * 4 = 8	3 * 4 = 12	4 * 4 = 16	5 * 4 = 20	6 * 4 = 24	7 * 4 = 28	8 * 4 = 32	9 * 4 = 36
2 * 5 = 10	3 * 5 = 15	4 * 5 = 20	5 * 5 = 25	6 * 5 = 30	7 * 5 = 35	8 * 5 = 40	9 * 5 = 45
2 * 6 = 12	3 * 6 = 18	4 * 6 = 24	5 * 6 = 30	6 * 6 = 36	7 * 6 = 42	8 * 6 = 48	9 * 6 = 54
2 * 7 = 14	3 * 7 = 21	4 * 7 = 28	5 * 7 = 35	6 * 7 = 42	7 * 7 = 49	8 * 7 = 56	9 * 7 = 63
2 * 8 = 16	3 * 8 = 24	4 * 8 = 32	5 * 8 = 40	6 * 8 = 48	7 * 8 = 56	8 * 8 = 64	9 * 8 = 72
2 * 9 = 18	3 * 9 = 27	4 * 9 = 36	5 * 9 = 45	6 * 9 = 54	7 * 9 = 63	8 * 9 = 72	9 * 9 = 81

- 아래와 같이 1부터 10까지 증가되고, 10부터 1까지 감소되는 프로그램을 작성해 보자.

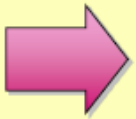


반복문 : 조건반복

- while 문
  - 특징
    - [조건만족]이 true인 동안은 문장을 반복하고 false일 경우에는 while 문 다음의 문장을 수행
    - 조건을 먼저 확인하기 때문에 한 번도 수행하지 않는 경우도 있음.
  - 형식

```
while (조건식)
{
        명령문;
}
```

```
초기값;
while (조건식)
{
        명령문;
        증감식;
}
```



```
for (초기값; 최종값; 증감식)
{
        명령문;
}
```

반복문 : 실행반복

do~while 문

- 특징
  - [반복문장]을 수행한 후에 [조건만족]을 확인
  - [조건 만족]이 맞던, 틀리던 [반복 문장]을 한 번 수행

- 형식

```
do
{
        문장 ;
} while(조건) ;
```

- 문제 : while문을 이용하여 1부터 100까지의 짝수의 합을 구하는 프로그램을 작성

- 문제 : do while문을 이용하여 1부터 100까지의 합을 구하는 프로그램을 작성

16. 제어문
– [3] 보조 제어문 continue

- continue 문
  - 특징
    - for문, while문, do while문의 반복중에 continue 문을 만나면 조건을 판단한 후 다음 구문을 수행
    - 다시 조건식으로 이동하는 명령문

- 실습

- 1에서 100까지의 정수 중 3의 배수를 제외한 수들의 합을 구하는 프로그램 (continue문 사용)

16. 제어문
– [3] 보조 제어문 break

- break문
  - 특징
    - for, while, do while 문의 반복중인 반복구문을 강제적으로 종료하는 경우에 사용
    - 반복문을 빠져나오는 명령문

반복문

- for ... in 반복문
  - 특정 객체의 모든 프로퍼티에 대해서 자동적으로 조치를 취하고자 할 경우

```
for (j in testObject)   {
        실행부분
}
```

  - for 루프는 testObject 객체에 있는 프로퍼티의 배열 인덱스 번호가 마지막 프로퍼티에 이르기까지 j를 0부터 하나씩 증가시키면서 명령어 블록을 수행