심화학습프로그램

JavaScript

2020.01.01

JavaScript Section 3

3.1 기본형식 및 적용방법

적용 방법	위치	태그 설명 및 특징
문서 내부	<head>	<script></script>
문서 내부	<body>	<script></script>
문서 외부	<head>	<script src="abc.js"></script>

기본명령어

입력	prompt('입력제목','입력내용')
출력	document.write('출력내용')
	alert('출력내용')
	console/log('출력내용')
확인	confirm('확인내용')

3.2 변수와 상수

변수 선언	변수 설명
~~var~~	중복해서 변수 선언 가능
let	중복해서 변수 선언 불가능 var를 보완해서 나옴
const	고정된 값을 할당하는 변수 선언 상수

데이터 타입	데이터 타입설명
숫자 데이터	정수, 실수, 지수등을 표현 예) 10, 10.5, 1e+2
문자 데이터	', " " 등과 같이 표현하며 보통의 경우 작은 따옴표를 많이 사용
boolean 데이터	true, false
undefined 데이터	값이 지정되지 않은 변수 정의되지 않은 데이터
null 데이터	undefined과 유사하지만 변수를 빈 상태로 만들거나 값이 존재하지 않게 만듬 null=공백데이터; 없는 게 아니다.

이스케이프 시퀀시	설명
\n	행(줄) 바꿈
\t	탭 문자
\\	역슬러시 역슬러시를 문자로 표현
\'	작은 따옴표
\"	큰따옴표

3.3 연산자

산술연산자	설명
+	더하기 연산을 실시
-	빼기 연산을 실시
*	곱하기 연산을 실시
/	나눠서 몫을 사용
%	나눠서 나머지를 사용
++	1씩 증가
--	1씩 감소

대입 연산자	설명
=	우측값을 좌측으로 대입(저장)
+=	더하기 누적 예) num+=1 , num = num + 1
-=	빼기 누적 예) num-=1 , num = num - 1
*=	곱하기 누적 예) num*=1 , num = num * 1
/*	나누기 누적 예) num/=1 , num = num / 1
%=	나머지 누적 예) num%=1 , num = num % 1

비교 연산자		설명
〉	A가 B보다 크다	a > b 가 참이면 true, 거짓이면 false를 반환
〈	A가 B보다 작다	a < b 가 참이면 true, 거짓이면 false를 반환
〉=	A가 B보다 크거나 같다	a >= b 가 참이면 true, 거짓이면 false를 반환
〈=	A가 B보다 작거나 같다	a <= b 가 참이면 true, 거짓이면 false를 반환
==	A가 B와 같다	a == b 가 참이면 true, 거짓이면 false를 반환
!=	A가 B와 같지않다	a != b 가 참이면 true, 거짓이면 false를 반환
===	A가 B와 데이터 혹은 타입이 같다	a === b 가 참이면 true, 거짓이면 false를 반환
!==	A가 B와 데이터 혹은 타입이 같지않다	a !== b 가 참이면 true, 거짓이면 false를 반환

논리 연산자	설명
&& (AND연산자)	a && b 중 하나라도 false면 false
|| (OR연산자)	a || b 중 하나라도 true면 true
! (NOT연산자)	false를 true로 true를 false로 변경

3.3 제어문

조건문	설명 및 특징
if , else	조건문의 true, false에 따라 실행되는 문장이 다른 구조의 순서 제어문

JS 코드

```
var age = 20;
if (age > 19)
{ document.write(age+"은 성인입니다.");}
else
{ alert("성인이 아닙니다.")}
```

true

false

document.write는 html에 출력하라는 뜻

조건문	설명 및 특징
if , else if, else	조건문의 false면 다시 else if에서 true, false에 따라 문장이 실행되는 구조

JS 코드

```
var age = 40;
if (age > 30)
{ document.write(age+"은 30대 이상입니다.");}
else if (age > 20)
{ document.write(age+"은 20대 이상입니다.");}
else
{ alert("성인이 아닙니다.")}
```

조건문	설명 및 특징
switch	조건값과 일치하는 case를 실행하는 구조, 일치하지 않으면 default실행, 일치시 break 실시

JS 코드

```
var subject='javascript';
switch(subject) {
    case 'html' :
        subject += '은 1학점입니다';
        break;
    case 'css' :
        subject += '은 2학점입니다';
        break;
    case 'javascript' :
        subject += '은 3학점입니다';
        break;
    default :
        subject = '해당과목이 없습니다.';
}
console.log(subject); 이게 consol에 뿌리는거
```

반복문	설명 및 특징
while	조건값에 의해 반복횟수를 정할수 있음, True로 지정하면 무한반복 후 break로 종료가능

JS 코드

```
var n =1;
var sum = 0;
while (n < 8) {
    sum+=n;
    n++;
}
console.log(sum);
```

보통의 경우 조건문을 True로 하여 특정 조건을 if구간으로 설정하여 만족시 break구문을 실행하여 빠져 나오도록 설계한다. while을 위와 같이 사용할 경우 for구문을 대체할 수 있다.

반복문	설명 및 특징
do while	do의 내용을 반드시 한번은 실행 후 while의 조건을 확인 후 반복여부 결정

JS 코드

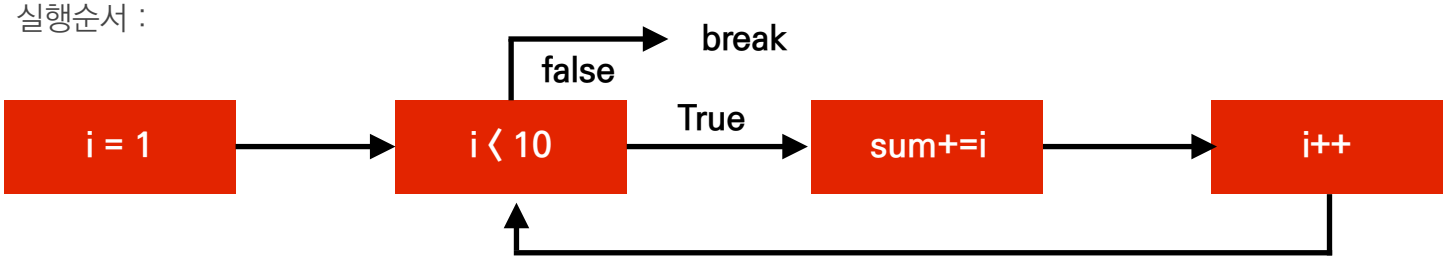
```
var n =1;
var sum = 0;
do {
    sum+=n;
    n++;
}
while (n < 8);
console.log(sum);
```

반복문	설명 및 특징
for	정해진 횟수만큼 반복한다. 횟수는 조건문의 True 로 판단한다

JS 코드

```
var sum = 0;
for(var i =1; i< 10; i++){
    sum+=i;
}
console.log(sum);
```

실행순서 :



중지하기	설명 및 특징
break	반복문 실행을 중지하고 빠져나온다.

JS 코드

```
var limitnum = 5;
var sum = 0;
for (var i =1; i<= 20; i++){
    sum =+ i;
    if (i === limitnum){
        break;
    }
}
console.log(sum);
```

건너뛰기	설명 및 특징
continue	실행을 중지하고 아래의 실행을 건너뛰고 반복문 처음으로 돌아간다.

JS 코드

```
var sum = 0 ;
var num = '3의 배수';
for (var i = 1; i < 10; i++){
    if (i % 3 === 0){
        num += i + ' ';
        continue;
    }
    sum += i;
}
console.log(num+'제외 총합 : '+ sum);
```

설계 순서	할일	힌트
1	구구단에서 왼쪽 수와 오른쪽 수가 가각 1씩 커지므로 2개의 for문을 사용할 때 바깥쪽 for문과 안쪽 for문을 어디에 사용할지 결정합니다.	
2	바깥쪽 for문을 작성합니다	카운터 변수 i
3	안쪽 for문을 작성합니다	카운터 변수 j

JS 코드

```
for(var i = 2; i <= 9; i++) {
    document.write("<div>");
    document.write("<h3>" + i + "단</h3>");
    for (var j = 1; j <= 9; j++) {
        document.write(i +" X " + j + " = " + i*j + "<br>");
    }
    document.write("</div>");
}
```

설계 순서	할일	힌트
1	팩토리얼 계산식에서 어떤 것을 변수로 놓을지 결정합니다	
2	계산식을 반복문으로 표현하기 위해 어떤 부분이 반복되는지 확인합니다	
3	사용자에게 숫자를 입력하게 합니다	prompt( )함수 사용
4	while( )함수를 사용하여 팩토리얼을 계산합니다	입력받은 수까지만 반복
5	팩토리얼 계산 값을 화면에 표시합니다	

JS 코드

```
var n = prompt("숫자를 입력하세요.");
var nFact = 1;
var i = 2;
while(i <= n) {
    nFact *= i;
    i++;
}
document.write(n + "!= " + nFact);
```

3.5 함수

분류	종류	설명
사용자 정의 함수	• 선언적 함수 • 익명함수	프로그램에서 필요한 기능을 사용자가 직접 정의해서 사용하는 함수
내장 함수	• 인코딩, 디코딩 함수 • 숫자 판별 함수 • 유한 무한 값 판별 함수 • 숫자변환 함수 • 문자 변환 함수 • 자바스크립트 코드 변경 함수	프로그램 개발에서 자주 사용되는 기능을 내부적으로 제공해주는 함수

선언적 함수	설명 및 특징
선언	function 함수이름 ( ) { 내용 }
호출	함수이름 ( );

JS 코드

```
function compute( ) {  리턴타입이 없음 왜냐? 데이터 타입이 없으니까
                       인자값이 없다.
    var x = 10;
    var y = 100;
    var result = x / y;
    console.log(result)
}
compute(); 함수를 실행
```

익명 함수	설명 및 특징	
선언	var 변수 = function ( ) { 내용; };	var는 일종의 참조변수라서 함수를 찾아갈 수 있다.
호출	변수이름 ( );	

JS 코드 변수이름으로 호출할 수도 있다. function 뒤에 이름이 없음

```
var compute = function( ) {
    var x = 10;
    var y = 100;
    var result = x / y;
    console.log(result)
};
compute();
```

반환	설명 및 특징
return	함수의 실행 결과를 돌려주는 명령어
매개변수	함수를 호출할 때 전달하는 변수 혹은 값

JS 코드

```
function process( ) {
    var x = 10;
    var y = 100;
    var result = x + y;
    return result;
}
console.log('결과값은 : '+process())
```

② 함수실행
③ 결과반환
① 함수호출

```
결과 값은 : 110
```

JS 코드

```
function process(x,y) {
    var result = x + y;
    return result;
}
console.log('결과값은 : '+process(10,100))
```

① 함수호출 10과 100을 x,y변수에 대입
② 함수실행
③ 결과반환

```
결과 값은 : 110
```

변수의 선언 위치	설명 및 특징
전역변수	프로그램 전체에 영향을 미치는 변수이며 어드서든지 자유롭게 사용가능
지역변수	특정 함수 내에서만 사용가능한 변수

JS 코드

```
var kor = 100
function process() {
    var kor=90; //var이 없으면 전역변수
    console.log(kor)
}
process();
console.log(kor)
```

kor은 전역변수

인코딩, 디코딩 함수	설명 및 특징
encodeURIComponent( )	영문,숫자와( )-_.~*!'를 제외한 한글과 같은 유니코드 문자를 인코딩 한다
decodeURIComponent( )	원상태로 디코딩한다.

숫자,유/무한 값 판별 함수 이다 아니다 boolean판단하는	설명 및 특징
isNaN( )	숫자인지 아닌지 판별하며 숫자이면 false아니면 true반환
isFinite( )	유한값이면 true 무한값이면 false를 반환

숫자, 문자 변환 함수	설명 및 특징
Number( )	숫자로 변환해 주는 함수
parseInt( )	숫자와 문자가 포함되어 있을 경우 정수 부분만 숫자로 변환해주는 함수
parseFloat( )	숫자와 문자가 포함되어 있을 경우 소수 부분까지 숫자로 변환해주는 함수
String( )	문자로 바꿔주는 함수

JS코드 변경 함수	설명 및 특징
eval( )	문자를 자바스크립트 코드로 변경해주는 함수

HTML 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>내장함수</title>
    <script>
        // 인코딩, 디코딩함수
        var encodeStr = '자바스크립트';
        console.log(encodeURIComponent(encodeStr));
        var decodeStr = encodeURIComponent(encodeStr);
        console.log(decodeURIComponent(decodeStr));
        // 숫자, 유한무한 값 판별 함수
        var num1 = '숫자';
        if (!isNaN(num1)) {
            console.log('숫자');
        }
```

HTML코드

```
        else {
            console.log('숫자아님');
        }
        var num2 = 1 / 0;
        if (isFinite(num2)) {
            console.log('유한값');
        } else {
            console.log('무한값');
        }
        // 숫자, 문자 변환 함수
        var num3 = '10';
        console.log(Number(num3));

        var num4 = '100px';
        console.log(parseInt(num4));
        var num5 = '33.3%';
        console.log(parseFloat(num5));
        var num6 = 10;
        console.log(typeof num6);
        console.log(typeof String(num6));
        // 자바스크립트 코드 변경 함수
        var str1 = 'var num7 = 10';
        var str2 = 'var num8 = 20';
        eval(str1); 문자가 아니라 코드다라는 걸 알림
        eval(str2);
        console.log(num7 + num8);
    </script>
</head>
<body>
</body>
</html>
```

설계 순서	할일	힌트
1	함수 선언	
2	함수 이름은 add로 지정하고, 매개변수 2개 사용하기	매개변수는 a,b로 지정
3	매개변수 두개를 더하고 sum변수에 저장 후 sum을 반환	retrun을 사용
4	함수 실행	
5	창을 통해 입력받은 값을 num1변수와 num2변수에 저장	prompt( ) 함수
6	num1과 num2값을 add함수 호출의 매개변수로 사용	
7	result에 결과값을 저장하고 출력	document.write( )

JS 코드

```
var num1 = parseInt(prompt("첫 번째 숫자는? "));
var num2 = parseInt(prompt("두 번째 숫자는? "));
var result = add(num1, num2);
document.write("<p>" + num1 + " + " + num2 + " = " + result +
               "</p>");
function add(a, b) {
    var sum = a + b;
    return sum;
}
```

HTML 코드

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>직접 만들어 보는 함수</title>
</head>
<body>
    <script src="function.js"></script>
</body>
</html>
```

3.6 객체

사용자정의 객체	설명 및 특징
객체란?	객체의 데이터는 이름:값의 쌍으로 이루어져 있으며 이것을 속성이라고한다
객체의 속성 : 프로퍼티	변수와 같으며 값을 저장
객체의 속성 : 메서드	함수와 같으며 기능을 정의
예)	var 변수 = {name:'홍길동',age:20, printout: function( ) {기능내용;}}

JS 코드

```
// 객체 리터럴
var circle = {
    color : 'yellow',
    diameter : 100,
    radius : function() {
    return this.diameter / 2;
    }
};
console.log(circle.color);
console.log(circle.diameter);
console.log(circle.radius());
// 객체 생성자 함수
function Triangle(b, h) {
    this.base = b;
    this.height = h;
    this.area = function() {
    return  b * h / 2;
    };
}
var triangle1 = new Triangle(15, 15);
console.log(triangle1.base);
console.log(triangle1.height);
console.log(triangle1.area());
var triangle2 = new Triangle(25, 10);
console.log(triangle2.base);
console.log(triangle2.height);
console.log(triangle2.area());
```

객체정보접근	설명 및 특징
for...in	순차적으로 객체의 속성에 접근할 수 있다.

JS 코드

```
var info = {
    subject: 'javascript',
    credit: 3,
    days: 20,
    tuition: 10000
};
for(var i in info) {
    console.log(i + ' : ' + info[i]);
}
```

JS 코드

```
var info = {
    subject: 'javascript',
    credit: 3,
    days: 20,
    tuition: 10000
};
for(var i=0; i<info.length; i++) {
    console.log(i + ' : ' + info[i]);
}
```

Number객체	설명 및 특징
객체생성	var num = new Number(10); 혹은 var num = 10;

Number 메서드	설명 및 특징
toFixed( )	toFixed(n)일때 n값만큼 소수점 자리수를 만들어 준다.
예)	num=328.575; console.log(num.toFixed(2)); //328.58

Number 메서드	설명 및 특징
toString( )	toString(n)일 때 n값의 진수로 만들어준다.
예)	num=12; console.log(num.toString(16)); //c (16진수)

String 객체	설명 및 특징
객체생성	var str = new String('자바'); 혹은 var str = '자바';

String 객체 프로퍼티	설명 및 특징
length	문자열의 개수를 취함

String 메서드	설명 및 특징
charAt( )	charAt(n)인 경우 n과 같은 index번호 문자를 반환 var str = 'javascript'; console.log(str.charAt(1)); //a출력
indexOf( )	특정 위치의 문자를 왼쪽 부터 검색하여 인덱스 번호를 반환 var str = 'javascript'; console.log(str.indexOf(a)); //1 출력, 만약 없으면 -1을 출력
lastIndexOf( )	특정 위치의 문자를 오른쪽 부터 검색하여 인덱스 번호를 반환 var str = 'javascript'; console.log(str.lastIndexOf(a)); //3 출력, 만약 없으면 -1을 출력
includes( )	해당 문자열에 특정 문자를 포함하고 있으면 True 아니면 False를 반환 var str = 'javascript'; console.log(str.includes(java)); //true 반환
substring( )	substring(4,9)인경우 인덱스4부터 8번까지 반환 번호가 하나인경우 해당번호부터 끝까지 반환 var str = 'javascript'; console.log(str.substring(4,9)); //scrip 반환
substr( )	substr(4,6)인 경우 4번째부터 6개의 문자를 반환 var str = 'javascript'; console.log(str.substr(4,6)); //script 반환
split( )	특정문자를 기준으로 분할 후 배열을 생성 var str = 'java_script'; var division = str.split('_'); //[0] java [1]script
replace( )	replace('x','y')인경우 x를 y로 변경 var str = 'javascript'; console.log(str.replace('java','Jquery')); //Jqueryscript 반환
concat( )	문자와 문자를 연결해준다 var str1='java'; var str2 = 'script'; console.log(str1.concat(str2)); // javascript반환

String 메서드	설명 및 특징
trim( )	문자열의 앞 뒤 공백을 제거해 준다
	var str=' java '; console.log('web'+str.trim( )); // webjavascript반환
toLowerCase( )	소문자로 변경
toUpperCase( )	대문자로 변경

JS 코드

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>String 객체</title>
    <script>
        // charAt()
        var str = 'Javascript'
        console.log(str.charAt(0));
        // indexOf()
        var str = 'Javascript'
        console.log(str.indexOf('a'));
        console.log(str.indexOf('k'));
        // lastIndexOf()
        var str = 'Javascript'
        console.log(str.lastIndexOf('a'));
        // includes()
        var str = 'Javascript';
        console.log(str.includes('script'));
        // substring()
        var str = 'http://itedunet.com';
        console.log(str.substring(0, 4));
        console.log(str.substring(7));
```

JS 코드

```
        // substr()
        var str = 'http://itedunet.com';
        console.log(str.substr(7, 4));
        // split()
        var str = 'Javascript_jQuery';
        var division = str.split('_');
        console.log(division[0] + ', ' + division[1]);
        // replace()
        var str = 'm_out.gif';
        console.log(str.replace('out', 'over'));
        // concat()
        var str1 = 'nav';
        var str2 = '_bg';
        console.log(str1.concat(str2));
        // trim()
        var str = ' removeblank ';
        console.log(str.trim());
        // toLowerCase(), toUpperCase()
        var str = 'LowerCase';
        console.log(str.toLowerCase());
        var str = 'UpperCase';
        console.log(str.toUpperCase());
    </script>
</head>
<body>
</body>
</html>
```

Array 객체	설명 및 특징
객체생성	var subject = new Array(10,20,'javascript');
객체생성	var subject = [10,20,'javascript'];

Array 객체 프로퍼티	설명 및 특징
length	문자열의 개수를 취함

Array 메서드	설명 및 특징
slice( )	slice(0,3)인 경우 인덱스 0번부터 2번까지 배열을 반환한다 번호가 하나인경우 해당번호 부터 끝까지 반환한다
	var alphabet = ['a','b','c','d','e']; console.log(alphabet.slice(0,3)); // ['a','b','c'] 반환
join( )	특정한 문자를 배열의 요소에 삽입하여 반환
	var alphabet = ['a','b','c','d','e']; console.log(alphabet.join('-')); // a-b-c-d-e 반환
concat( )	배열을 연결한다.

	var alphabet1 = ['a','b','c'];
	var alphabet2 = ['d','e'];
	console.log(alphabet1.concat(alphabet2)); // ['a','b','c','d','e'] 반환

toString( )	배열을 문자로 변환해준다.
	var alphabet = ['a','b','c','d','e']; console.log(alphabet.toString( )); // a,b,c,d,e 반환
shift( )	첫 번째 배열을 삭제한다

	var alphabet = ['a','b','c','d','e'];
	alphabet.shift( );
	console.log(alphabet); // ['b','c','d','e'] 반환

unshift( )	첫 번째 배열을 추가한다.

	var alphabet = ['b','c','d','e'];
	alphabet.unshift('a');
	console.log(alphabet); // ['a','b','c','d','e'] 반환

pop( )	마지막 배열을 삭제합니다

	var alphabet = ['a','b','c','d','e'];
	alphabet.pop( );
	console.log(alphabet); // ['a','b','c','d'] 반환

Array 메서드	설명 및 특징
splice( )	지정된 부분의 배열을 추가,삭제, 변경할 수 있습니다.

	var alphabet = ['a','b','c','d','e'];
	alphabet.splice(1,0,'f');
	console.log(alphabet); // ['a','f','b','c','d','e'] 반환, 1번에 0개 삭제(삭제가0개면 추가) f를 추가

	var alphabet = ['a','b','c','d','e'];
	alphabet.splice(1,1);
	console.log(alphabet); // ['a','c','d','e'] 반환, 1번에 1개를 삭제

	var alphabet = ['a','b','c','d','e'];
	alphabet.splice(1,1,'f');
	console.log(alphabet); // ['a','f','c','d','e'] 반환, 1번에 1개를 삭제후 f를 추가

	var alphabet = ['a','b','c','d','e'];
	alphabet.splice(0,3,'f','g','h');
	console.log(alphabet); // ['f','g','h','d','e'] 반환, 3번에 3개를 삭제후 f,g,h를 추가

reverse( )	배열의 순서를 바꾸어 줍니다
	var alphabet = ['a','b','c','d','e']; console.log(alphabet.reverse( )); // ['e','d','c','b','a'] 반환
sort( )	배열의 오름차순 정렬을 합니다. 배열의 정렬은 문자열로 비교하기 때문에 [2,41,11]을 정렬할 경우 41은 4로 11은 1로 처리되어 [11,2,41]순으로 정렬되는 문제가 발생합니다. 이것을 해결하는 방법은 다음과 같다.

JS 코드

```
var num = [2,41,11];
num.sort(function(a,b){
    return a-b; //[2,11,41]오름차순 정렬
});
num.sort(function(a,b){
    return b-a; //[41,11,2]내림차순 정렬
});
console.log(num.toString());
```

배열 요소 접근	설명 및 특징
for...of	순차적으로 배열의 값에 접근할 수 있다

JS 코드

```
// for...in의 i는 index값을 가진다.
var city = ['서울','부산','창원','대구'];
for (var i in city){
    console.log(i);
}
// for...of의 i는 '서울'과 같은 value값을 가진다.
var city = ['서울','부산','창원','대구'];
for (var i of city){
    console.log(i);
}
```

배열 요소 접근	설명 및 특징
forEach( )	배열의 요소에 순차적으로 접근하여 필요한 값을 생성할 수 있다.

JS 코드

```
var num = [1,2,3];
sum=0;
num.forEach(function(value,index,array){
    console.log(value); // 1 2 3 배열의 값
    console.log(index); // 0 1 2 배열의 인덱스값
    console.log(array); // [1,2,3] 배열 자신
});
num.forEach(function(value){
    return sum += value //0+1+2+3 = 6
});
console.log(sum);
```

배열 요소 접근	설명 및 특징
map( )	기존의 배열을 이용하여 새로운 배열을 생성할 수 있음

JS 코드

```
var base = [10,20,30];
base.map(function(value,index,array){
    console.log(value); // 10 20 30 배열의 값
    console.log(index); // 0 1 2 배열의 인덱스값
    console.log(array); // [10,20,30] 배열 자신
});
var area = base.map(function(value){
    return value * 8;
});
console.log(area.toString( )); //80,160,240
```

배열 요소 접근	설명 및 특징
filter( )	조건에 맞는 배열 요소들만 새로운 배열로 만들어 준다.

JS 코드

```
var data = ['javascript',20,30,'jQuery'];

base.map(function(value,index,array){
    console.log(value); // javascript 20 30 jQuery 배열의 값
    console.log(index); // 0 1 2 3 배열의 인덱스값
    console.log(array); // ['javascript',20,30,'jQuery'] 배열 자신
});
var num = data.filter(function(value){
    return typeof value === 'number';
});
console.log(num.toString( )); //20,30
```

Math 객체	설명 및 특징
Math 객체 생성	new연산자로 객체를 생성하지 않아오 되는 수학 연산을 위한 내장객체 생성방법은 Math.프로퍼티 Math.메서드

Math 객체 프로퍼티	설명 및 특징
PI	원주율 값(3.14) 이다.
E	자연 로그 밑인 상수 e 2.718입니다
LN2	2의 자연 로그 값 0.693입니다.
LN10	10의 자연 로그 값 2.302입니다
LOG2E	밑이 2인 e의 로그 1.442입니다
LOG10E	밑이 10인 e의 로그 0.434입니다.
SQRT1_2	1/2의 제곱근 0.707입니다.
SQRT2	2의 제곱근 1.414입니다

Math 객체 메서드	설명 및 특징
abs( )	Math.abs(n)일 경우 n의 절대값을 반환합니다.
	console.log(Math.abs(-2)); // 2
max( )	Math.max(n1,n2)일 경우 n1과 n2중 큰 값을 반환합니다.
	console.log(Math.max(10,20)); // 20
min( )	Math.min(n1,n2)일 경우 n1과 n2중 작은 값을 반환합니다.
	console.log(Math.min(10,20)); // 10
round( )	Math.round(n)일 경우 n의 소수점 이하가 5 이상이면 반올림 아니면 절삭한 값을 반환
	console.log(Math.round(10.5)); // 11
ceil( )	Math.ceil(n)일 경우 n의 소수점을 올림한 정수로 반환
	console.log(Math.ceil(10.4)); // 11
floor( )	Math.floor(n)일 경우 n의 소수점을 절삭한 정수 값을 반환
	console.log(Math.floor(10.4)); // 10

Math 객체 메서드	설명 및 특징
random( )	Math.random( )의 경우 0~1 사이의 난수 값을 반환
	console.log(Math.random( )*3); // 3을 곱하여 반환 console.log(Math.floor(Math.random( ) *3)); // 3을 곱한 난수에 소수점을 버리고 반환
sin( )	Math.sin(n)인 경우 n의 사인값을 반환
tan( )	Math.tan(n)인 경우 n의 탄젠트값을 반환
sqrt( )	Math.sqrt(n)인 경우 n의 제곱근값을 반환

JS 코드

```
var luckyNumber = [];
var num = 0;
for(var i = 1; i <= 100; i++) {
    luckyNumber.push(i); // 1(index 0) ~ 100(index 99)
}
num = Math.floor(Math.random() * luckyNumber.length); // 0 ~ 99
console.log(luckyNumber.toString());
console.log('오늘 행운의 당첨번호은 ' + luckyNumber[num] + '입니다.');
```

Date 객체	설명 및 특징
Date 객체 생성	날짜와 시간에 대한 정보 값을 얻거나 설정할 수 있다. var date = new Date( );

Date 객체 메서드	설명 및 특징
getFullYear( )	4자리 연도 값을 반환합니다
getMonth( )	월값을 반환합니다, 1월은 0이므로 +1을 해줍니다
getDate( )	일 값을 반환합니다, 1~31
getDay( )	요일값을 반환합니다, 0 : 일요일
getHours( )	시간값을 반환합니다, 0~23
getMinutes( )	분값을 반환합니다, 0~59
getSeconds( )	초값을 반환합니다, 0~59
getMilliseconds( )	밀리초(1/1000) 값을 반환합니다, 0~999
getTime( )	1970년 1월 1일 자정 이후부터 누적된 밀리초 값을 반환합니다.
gettimezoneOffset( )	UTC(국제 표준시)와의 시차 값을 반환합니다

Date 객체 메서드	설명 및 특징
setFullYear( )	4자리 년도 값을 설정합니다
setMonth( )	월 값을 설정합니다, 0~11
setDate( )	일 값을 설정합니다, 1~31
setHours( )	시간 값을 설정합니다, 0~23
setMinutes( )	분값을 설정합니다, 0~59
setSeconds( )	초값을 설정합니다, 0~59
setMilliseconds( )	1/1000초 값을 설정합니다, 0~999
setTime( )	1970년 1월 1일 자정 이후부터 경과한 밀리초 값을 설정합니다.
toString( )	날짜/시간 정보를 문자로 반환합니다.
toGMTString( )	그리니치 표준시 날짜/시간 정보를 문자열로 반환합니다
toUTCString( )	국제표준시 날짜/시간 정보를 문자열로 반환합니다.
toDateString( )	날짜를 문자열로 반환합니다
toTimeString( )	시간을 문자열로 반환합니다
toLocaleString( )	지역의 Date 정보를 문자열로 반환합니다
toLocaleDateString( )	지역의 날짜 정보를 문자열로 반환합니다.
toLocaleTimeString( )	지역의 시간 정보를 문자열로 반환합니다.
parse( )	날짜 문자열을 1970년 1월 1일 자정 이후 부터 경과한 밀리초를 반환합니다.
UTC( )	날짜를 1970년 1월 1일 자정 이후부터 경과한 밀리초 값을 반환합니다

순서	할일	힌트
1	여행에 필요한 준비물 저장할 빈 배열 만들기	
2	빈 배열에 내용을 추가할 addList( ) 함수 만들기	querySelect("텍스트 필드의 id이름").value
3	[추가] 버튼의 click이벤트와 addList( )함수 연결하기	addEventListener( ) 함수
4	itemList배열의 내용을 화면에 보여줄 showList( )함수 만들기	
5	HTML태그 문자열을 저장할 list 변수 만들기	태그와 태그
6	for문을 사용해 itemList배열의 요소를 차례로 가져온 후 ~ 태그로 묶어 list변수에 저장하기	for문은 0에서부터 itemList.length보다 작을때까지 반복동작
7	준비한 항목을 화면에서 삭제하는 removeList( )함수 만들기	
8	showList( )함수에서 각 항목을 표시하는 소스에 X 버튼을 함께 표시하기	태그
9	삭제 버튼의 click이벤트와 removeList( )함수 실행 연결	querySelectorAll( )함수를 사용해 삭제 버튼을 remove변수에 저장
10	이벤트가 발생한 삭제 버튼의 id값 알아내기	this.getAttribute(id)사용
11	splice( )함수를 사용해 해당 id값이 가리키는 itemList배열의 요소 삭제하기	splice(id,1)함수를 사용하면 id값이 가리키는 위치에서 요소 1개를 삭제함
12	showList( )함수를 사용해 변경된 배열을 화면에 표시	

HTML 코드

```
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width">
    <title>여행 준비물 점검 목록</title>
    <link rel="stylesheet" href="css/input.css">
    <link rel="stylesheet" href="css/list.css">
</head>
<body>
  <div id="wrapper">
      <h2>여행 준비물 점검 목록</h2>
      <form>
        <input type="text" id="item" autofocus="true">
        <button type="button" id="add" class="addBtn">추가</button>
```

HTML 코드

```
      </form>
      <div id="itemList"></div>
  </div>

<script src="js/diy-itemList-result.js"></script>
</body>
</html>
```

JS 코드

```
var itemList = [];

var addBtn = document.querySelector('#add');
addBtn.addEventListener("click", addList);

function addList() {
  var item = document.querySelector("#item").value;
  // 텍스트 필드 내용 가져옴
  if (item != null) {
    itemList.push(item);  // itemList 배열의 끝에 item 변수 값 추가
    document.querySelector("#item").value = "";
    // id="item"인 요소의 값을 지움
    document.querySelector("#item").focus();
    // 텍스트 필드에 focus( ) 메서드 적용
  }
  showList();
}
function showList() {
  var list = "<ul>";  // 목록을 시작하는 <ul> 태그 저장
  for (var i = 0; i < itemList.length; i++) {  // 배열 요소마다 반복
    list += "<li>" + itemList[i] + "<span class='close' id=" + i
    + ">X</span></li>";  // 요소와 삭제 버튼을 <li>~</li>로 묶음
  }
  list += "</ul>";   // 목록을 끝내는 </ul> 태그 저장
  document.querySelector('#itemList').innerHTML = list;
  // list 내용 표시
```

JS 코드

```
  var remove = document.querySelectorAll(".close");
  // 삭제 버튼을 변수로 저장. 배열 형태가 됨
  for (var i = 0; i < remove.length; i++) {
  // remove 배열의 요소 모두를 확인
    remove[i].addEventListener("click", removeList);
    // 요소를 클릭하면 removeList() 실행
  }
}

function removeList() {
  var id = this.getAttribute("id");
  // this(클릭한 삭제 버튼)의 id 값 가져와 id 변수에 저장
  itemList.splice(id, 1);  // itemList 에서 인덱스 값이 id인 요소 1개 삭제
  showList();  // 변경된 itemList 배열을 다시 화면에 표시
}
```

정규표현식 메서드	설명 및 특징
test( )	정규표현식과 일치하는 문자열이 있으면 true 없으면 false 반환

	var reg = /Javascript/;
	console.log(reg.test('Javascript')); // true
	console.log(reg.test('script')); // false

match( )	정규표현식과 일치하는 문자열을 배열로 만들어주며 일치하는 문자열이 없으면 null을 반환
	var reg = /Javascript/; str = 'Java script'; console.log(str.match(reg)); // null

정규식 주요 패턴	설명 및 특징
abc	abc문자열을 검색합니다 /abc/는 'abc'
[abc]	a,b,c중 하나의 문자를 검색합니다
[^abc]	a,b,c를 제외한 문자 하나를 검색합니다
[A-Z]	알파벳 대문자를 검색합니다
[0-9]	0-9까지의 숫자를 검색합니다
.	하나의 문자를 검색합니다. 만약 마침표 문자 그대로의 의미를 사용할 경우 \.으로 표현해야합니다
\w	알파벳,숫자,_를 검색합니다
\W	알파벳,숫자,_를 제외하고 검색합니다
\d	숫자를 검색합니다
\D	숫자를 제외하고 검색합니다
\s	하나의 공백을 검색합니다
\S	공백을 제외하고 검색합니다
^	행의 첫 문자가 일치해야 함을 의미
$	행의 끝 문자가 일치해야 함을 의미
*	*앞의 문자가 0번 이상 반복을 의미합니다
+	+앞의 문자가 1번이상 반복을 의미합니다
?	?앞의 문자가 0번 또는 1번 의미합니다
{ }	숫자의 갯수를 지정할 수 있음 / \d{3}/는 숫자 3개를 의미
a|b	a혹은 b를 의미
(abc)	그룹화를 의미

HTML 코드

```
<html>
<head>
<script type="text/javascript" src="./js/validation.js"></script>
<title>상품 등록</title>
</head>
<body>
    <div class="jumbotron">
        <div class="container">
            <h1 class="display-3">상품등록</h1>
        </div>
    </div>
    <div class="container">
        <form name="newProduct" action="#" class="form-horizontal"
         method="post" enctype="multipart/form-data">
            <div class="form-group row">
                <label class="col-sm-2">상품아이디</label>
                <div class="col-sm-3">
                    <input type="text" id="productId"
                           name="productId"  class="form-control" >
                </div>
            </div>
            <div class="form-group row">
                <label class="col-sm-2">상품이름</label>
                <div class="col-sm-3">
                    <input type="text" id="name" name="name"
                           class="form-control" >
                </div>
            </div>
            <div class="form-group row">
                <label class="col-sm-2">상품가격</label>
                <div class="col-sm-3">
                    <input type="text" id="unitPrice"
                           name="unitPrice" class="form-control" >
                </div>
            </div>
```

HTML 코드

```
<div class="form-group row">
    <label class="col-sm-2">상품설명</label>
    <div class="col-sm-5">
        <textarea name="description" cols="50" rows="2"
                  class="form-control"></textarea>
    </div>
</div>
<div class="form-group row">
    <label class="col-sm-2">제조사</label>
    <div class="col-sm-3">
        <input type="text" name="manufacturer"
               class="form-control">
    </div>
</div>
<div class="form-group row">
    <label class="col-sm-2">분류</label>
    <div class="col-sm-3">
        <input type="text" name="category"
               class="form-control" >
    </div>
</div>
<div class="form-group row">
    <label class="col-sm-2">재고수</label>
    <div class="col-sm-3">
       <input type="text" id="unitsInStock"
              name="unitsInStock"class="form-control" >
    </div>
</div>
```

HTML 코드

```
            <div class="form-group row">
                <label class="col-sm-2">제품상태</label>
                <div class="col-sm-5">
                    <input type="radio" name="condition"
                           value="New " >새상품
                    <input type="radio" name="condition"
                           value="Old" >중고상품
                    <input type="radio" name="condition"
                           value="Refurbished" >재생상품
                </div>
            </div>
            <div class="form-group row">
                <label class="col-sm-2">제품이미지</label>
                <div class="col-sm-5">
                    <input type="file" name="productImage"
                           class="form-control">
                </div>
            </div>
            <div class="form-group row">
                <div class="col-sm-offset-2 col-sm-10 ">
                    <input type="button" class="btn btn-primary"
                           value="확인" onclick="CheckAddProduct()">
                </div>
            </div>
        </form>
    </div>
</body>
</html>
```

JS 코드

```
function CheckAddProduct() {

    var productId = document.getElementById("productId");
    var name = document.getElementById("name");
    var unitPrice = document.getElementById("unitPrice");
    var unitsInStock = document.getElementById("unitsInStock");

    // 상품아이디 체크
    if (!check(/^P[0-9]{4,11}$/, productId,
            "[상품 코드]\nP와 숫자를 조합하여 5~12자까지 입력하세요\n첫 글자는 반드시 P로 시작하세요"))
        return false;
    // 상품명 체크
    if (name.value.length < 4 || name.value.length > 12) {
        alert("[상품명]\n최소 4자에서 최대 50자까지 입력하세요");
        name.select();
        name.focus();
        return false;
    }
    // 상품 가격 체크
    if (unitPrice.value.length == 0 || isNaN(unitPrice.value)) {
        alert("[가격]\n숫자만 입력하세요");
        unitPrice.select();
        unitPrice.focus();
        return false;
    }

    if (unitPrice.value < 0) {
        alert("[가격]\n음수를 입력할 수 없습니다");
        unitPrice.select();
        unitPrice.focus();
        return false;
    } else if (!check(/^\d+(?:[.]?[\d]?[\d])?$/, unitPrice,
            "[가격]\n소수점 둘째 자리까지만 입력하세요"))
        return false;
```

JS 코드

```
    // 재고 수 체크
    if (isNaN(unitsInStock.value)) {
        alert("[재고 수]\n숫자만 입력하세요");
        unitsInStock.select();
        unitsInStock.focus();
        return false;
    }

    function check(regExp, e, msg) {

        if (regExp.test(e.value)) {
            return true;
        }
        alert(msg);
        e.select();
        e.focus();
        return false;
    }

     document.newProduct.submit()
}
```

3.7 이벤트

마우스 이벤트	설명 및 특징
click	마우스를 클릭했을 때 이벤트가 발생
dbclick	마우스를 더블클릭했을 때 이벤트가 발생
mouseover	마우스를 오버했을 때 이벤트가 발생
mouseout	마우스를 아웃했을 때 이벤트가 발생
mousedown	마우스를 눌렀을 때 이벤트가 발생
mouseup	마우스를 떼었을 때 이벤트가 발생
mousemove	마우스를 움직였을 때 이벤트가 발생

키 이벤트	설명 및 특징
keydown	키를 눌렀을 때 이벤트가 발생
keyup	키를 떼었을 때 이벤트가 발생
keypress	키를 누른 상태에서 이벤트가 발생

폼 이벤트	설명 및 특징
focus	포커스가 이동되었을 때 이벤트가 발생
blur	포커스가 벗어났을 때 이벤트가 발생
change	값이 변경되었을 때 이벤트가 발생
submit	submit 버튼을 눌렀을 때 이벤트가 발생
reset	reset 버튼을 눌렀을 때 이벤트가 발생
select	input이나 textarea요소 안의 텍스트를 드래그하여 선택했을 때 이벤트가 발생

로드,기타 이벤트	설명 및 특징
load	로딩이 완료되었을 때 이벤트가 발생
abort	이미지의 로딩이 중단되었을 때 이벤트가 발생
resize	사이즈가 변경되었을 때 이벤트가 발생
scroll	스크롤바를 움직였을 때 이벤트가 발생

이벤트 연결	설명 및 특징
인라인 이벤트 연결	html요소에 직접 이벤트를 연결하는 방식

HTML 코드

```
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width">
    <script>
        function sum(n){
            alert(n);
        }
    </script>
</head>
<body>
    <button onclick='sum(10);'>클릭</button>
</body>
</html>
```

이벤트 연결	설명 및 특징
기본 이벤트 연결	html요소를 취득한 후 이벤트를 '객체의 메서드'형식(객체.메서드 = function( ){내용})으로 연결

HTML 코드

```
<body>
    <button id="bt">클릭</button>
    <script>
        var bt = document.getElementById('bt');
        bt.onclick = function( ) {
            console.log('ok');
        };
    </script>
</body>
```

만약 취득할 요소가 요소 취득 명령어 이후에 오면 반드시 load 이벤트를 적용해야합니다

HTML 코드

```
<!doctype html>
<html>
<head>
    <script>
        window.onload = function( ) {
            var bt = document.getElementById('bt');
            bt.onclick = function( ) {
            console.log('ok');
            };
        };
    </script>
</head>
<body>
    <button id="bt">클릭</button>
</body>
</html>
```

이벤트 연결	설명 및 특징
표준 이벤트 연결	객체.addEventListener('이벤트',함수)의 방식으로 이벤트를 연결

HTML 코드

```
<head> //기본이벤트모델은 동일한 이벤트 한번만 적용,표준이벤트 모델은 여러번 적용 가능
    <script>
        window.onload = function( ) {
            var bt = document.getElementById('bt');
            function view( ) {
            console.log('ok');
            }
            bt.addEventListener('click',view); // on을 붙이지 않음
        };
    </script>
</head>
<body>
    <button id="bt">클릭</button>
</body>
```

이벤트 객체	설명 및 특징
객체 생성	자바스크립트에서 기본적으로 제공되며 좌표값이라든지 이벤트 객체 정보를 쉽게 얻을수 있음

예)	bt.onclick = function(event) {
	event.프로퍼티;
	event.메서드;
	}

객체 프로퍼티	설명 및 특징
target	이벤트를 발생시킨 객체를 반환합니다
type	이벤트의 이름을 반환합니다
clientX	이벤트가 발생한 X좌표 값을 반환합니다 (브라우저 기준)
clientY	이벤트가 발생한 Y좌표 값을 반환합니다. (브라우저 기준)
screenX	이벤트가 발생한 X좌표 값을 반환합니다. (모니터 기준)
screenY	이벤트가 발생한 Y좌표 값을 반환합니다. (모니터 기준)
button	마우스 왼쪽(0), 가운데(1), 오른쪽(2) 버튼 값을 반환합니다.

HTML 코드

```
<!DOCTYPE HTML> //F12번 콘솔에 좌표 확인
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <style>
        div{
            height: 100px;
            background: #718c00;
            margin-top: 20px;
            color: #fff
        }
    </style>
    <script>
        window.onload = function() {
            var bt = document.getElementById('bt');
            var area = document.getElementById('area');
            bt.onclick = function(event) {
                console.log(event.target);
```

HTML 코드

```
                console.log(event.type);
                console.log(event.clientX);
                console.log(event.clientY);
                console.log(event.screenX);
                console.log(event.screenY);
            };
            area.onmousedown = function(event) {
                console.log(event.button);
            };
        };
    </script>
</head>
<body>
    <button id="bt">클릭</button>
    <div id="area">여기에 마우스 왼쪽, 가운데, 오른쪽 버튼 클릭</div>
</body>
</html>
```

객체 메서드	설명 및 특징
preventDefault( )	기본 이벤트의 실행을 막아 줍니다
stopPropagation( )	이번트 버블링을(자식 이벤트로인해 부모 이벤트까지 같이 실행) 방지해줍니다.

HTML 코드

```
<head>
    <script>
        window.onload = function() {
            var a = document.getElementById('linker');
            a.onclick = function() {
                alert('아이티에듀넷');
            } //event.preventDefault( ) 추가로 a태그 기본 이벤트 발생 중지
        }     //혹은 return false 추가 로도 가능
    </script>
</head>
<body>
  <div><a href="http://itedunet.com" id="linker">아이티에듀넷</a></div>
</body>
```

HTML 코드

```
<html lang="ko">
<head>
    <style>
        * { margin: 0; padding: 0; }
        #outer {
            background: #ff0000;
            padding: 20px;
        }
        #inner {
            background: #00ff00;
            padding: 20px;
        }
    </style>
    <script>
        window.onload = function() {
            var a = document.getElementById('outer');
            var b = document.getElementById('inner');
            outer.onclick = function() {
                alert('부모 이벤트');
            };
            inner.onclick = function(event) {
                alert('자식 이벤트');
                // event.stopPropagation();
            };
        };
    </script>
</head>
<body>
    <div id="outer">
        <p id="inner">클릭</p>
    </div>
</body>
</html>
```

순서	할일	힌트
1	HTML 문서에서 [상세 설명 보기] 버튼과 [상세 설명 닫기] 버튼의 id 값 확인	
2	HTML 문서에 js 파일 연결하기	
3	[상세 설명 보기] 버튼 소스에서 이벤트 처리기를 사용해 showDetail( )함수연결	onclick 사용
4	[상세설명닫기] 버튼 소스에서 이벤트 핸들러를 사용해 hideDetail( )함수 연결	onclick 사용

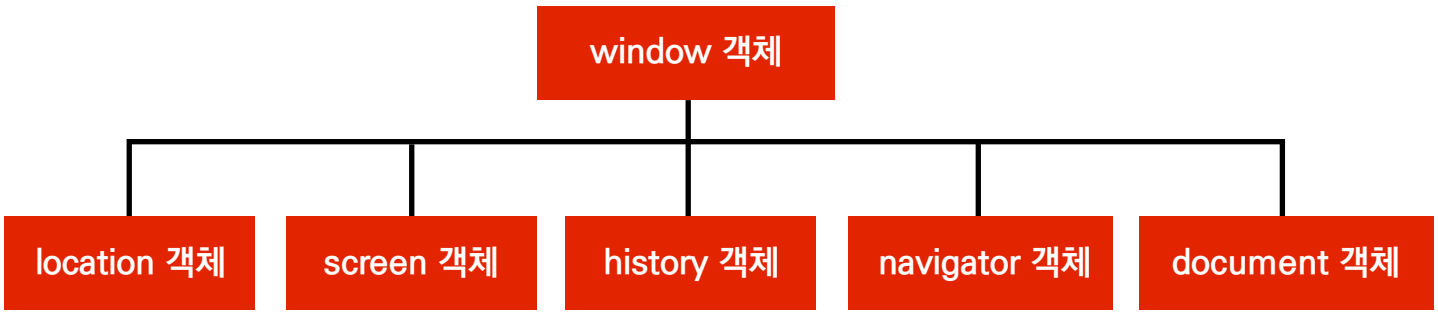
HTML 코드

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>자바스크립트 이벤트</title>
    <link rel="stylesheet" href="css/event.css">
</head>
<body>
    <div id="item">
        <img src="images/flower1.jpg" alt="">
        <button class="over" id="open">상세 설명 보기</button>
        <div id="desc" class="detail">
            <h4>민들레</h4>
            <p>어디서나 매우 흔하게 보이는 잡초로서 바닥에 딱 붙어서 꽃봉오리 하나가
               쏙 올라온다. 톱니 모양의 잎새와 눈에 확 띄는 노란 꽃이 인상적이다.
               특히 꽃이 지고나면 솜털모양의 깃을 가진 씨앗들이 나오는데 바람을
               타고 날아가서 널리 퍼진다.
            </p>
            <button id="close">상세 설명 닫기</button>
        </div>
    </div>
</body>
</html>
```

JS 코드

```
// '상세 설명 보기'를 클릭했을 때 상세 설명을 보여주는 함수
function showDetail() {
    document.querySelector('#desc').style.display = "block";
    // 상세 설명 부분을 화면에 표시
    document.querySelector('#open').style.display = "none";
    // '상세 설명 보기' 단추를 화면에서 감춤
}
// '상세 설명 닫기'를 클릭했을 때 상세 설명을 감추는 함수
function hideDetail() {
    document.querySelector('#desc').style.display = "none";
    // 상세 설명 부분을 화면에서 감춤
    document.querySelector('#open').style.display = "block";
    // '상세 설명 보기' 단추를 화면에 표시
}
```

3.8 BOM(Browser object Model)



윈도우 객체	설명 및 특징
open( )	새로운 윈도우를 만들어 주는 메서드 이다.
예)	window.open('문서 주소', '윈도우 이름', '옵션=값,옵션=값');

옵션	설명 및 특징
width = 픽셀값	윈도우의 가로 너비를 설정합니다.
height = 픽셀값	윈도우의 세로 너비를 설정합니다.
left = 픽셀값	윈도우의 left 위치를 설정합니다.
top = 픽셀값	윈도우의 top 위치를 설정합니다.
location = yes / no	윈도우의 주소창에 대한 show/hide를 설정합니다.
scrollbars = yes / no	윈도우의 스크롤바에 대한 show/hide를 설정합니다.
menubar = yes / no	윈도우의 메뉴바에 대한 show/hide를 설정합니다.
toolbar = yes / no	윈도우의 툴바에 대한 show/hide를 설정합니다.
status = yes / no	윈도우의 상태줄에 대한 show/hide를 설정합니다.

JS 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>open 메서드</title>
    <script>
        window.onload = function() {
            var bt = document.getElementById('bt');
            bt.onclick = function() {
                window.open('http://itedunet.com', 'itedunet',
                      'width=300, height=300, left=100, top=10');
            };
        };
    </script>
</head>
<body>
    <button id="bt">새창열기</button>
</body>
</html>
```

타이머 함수	설명 및 특징
setInterval( )	일정 시간마다 지정한 함수를 반복적으로 실행하는 함수
	setInterval(function( ){ 실행문 }, 밀리초(1/1000));
clearInterval( )	setInterval( )함수를 중지시키는 함수
setTimeout( )	설정한 시간이 흐른 뒤에 지정한 함수를 한 번만 실행하는 함수
	setInterval(function( ){ 실행문 }, 밀리초(1/1000));
clearTimeout( )	setTimeout( )함수를 중지시키는 함수

JS 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <script>
        var i = 0;
        setInterval(function() {
            i++;
            alert('2초 마다 실행' + i);
        }, 2000);
    </script>
</head>
<body>
</body>
</html>
```

JS 코드

```
<html lang="ko">
<head>
    <script>
        window.onload = function() {
            var bt = document.getElementById('bt');
            var i = 0;
            var increase = setInterval(function() {
                i++;
                alert(i);
            }, 2000);
            bt.onclick = function() {
                clearInterval(increase);
            };
        };
    </script>
</head>
<body>
    <button id="bt">멈춤</button>
</body>
</html>
```

location 객체	설명 및 특징
location	웹브라우저의 주소 URL관련 객체
프로퍼티	http://itedunet.com:8080/search?book=5#coding

location 객체 프로퍼티		설명 및 특징
hash	#coding	주소의 앵커명(#)을 반환
host	http://itedunet.com:8080	주소의 호스트명과 포트 번호를 반환
port	8080	주소의 포트 번호를 반환
pathname	/search	주소의 패스명을 반환
href	전체주소	주소의 값을 반환
protocol	http:	주소의 프로토콜명을 반환
search	?book=5	주소의 쿼리 문자열을 반환

location 객체 메서드	설명 및 특징
reload( )	현재 페이지를 다시 로드합니다.
	<button onclick="javascript:location.reload( );">reload</button>
replace( )	replace(url)일 때 url 값으로 이동합니다.
	<button onclick="javascript:location.replace('http://itedunet.com');">replace </button>

screen 객체	설명 및 특징
screen	모니터 화면정보 관련 객체

screen 프로퍼티	설명 및 특징
width	화면의 너비를 반환합니다
height	화면의 높이를 반환합니다.
availWidth	화면에서 작업표시줄을 제외한 너비를 반환합니다.
avaiHeight	화면에서 작업표시줄을 제외한 높이를 반환합니다.
colorDepth	화면에서 사용 가능한 색상수를 반환합니다.
pixelDepth	화면의 색상 해상도를 반환합니다.

history 객체	설명 및 특징
history	페이지가 이동한 정보를 관리하는 객체

history 메서드	설명 및 특징
back( )	이전페이지로 이동합니다
	<button onclick="javascript:history.back( );">back</button>
forward( )	이후 페이지로 이동합니다.
	<button onclick="javascript:history.forward( );">back</button>
go( )	go(n)일 때 n값에 따라 양수이면 이후 페이지 음수이면 이전 페이지로 이동합니다.
	<button onclick="javascript:history.forward(2);">back</button>

navigator 객체	설명 및 특징
navigator	브라우저 버전이나 브라우저명 등 브라우저 정보에 관한 객체.

navigator 프로퍼티	설명 및 특징
appCodeName	브라우저의 코드명을 반환해 줍니다
appName	브라우저명을 반환해 줍니다
appVersion	브라우저의 버전을 반환해 줍니다
platform	플랫폼명을 반환해 줍니다
userAgent	브라우저의 코드명과 저번을 반환해 줍니다

JS 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>navigator 객체</title>
    <script>
        var browser = navigator.userAgent;
        alert(browser);
        var browserName = '';
        if (browser.match(/Trident/)) {
            browserName = '인터넷 익스플러';
        }
```

JS 코드

```
        else if (browser.match(/Chrome/)) {
            browserName = '크롬';
        } else if (browser.match(/Firefox/)) {
            browserName = '파이어폭스';
        } else {
            browserName = '알 수 없는 브라우저';
        }
        alert(browserName);
    </script>
</head>
<body>
</body>
</html>
```

3.9 DOM(Document object Model)

요소선택 메서드	설명 및 특징
getElementById( )	HTML 요소중 id속성을 찾아서 선택한다
getElementsByClassName( )	HTML 요소중 class속성을 찾아서 선택한다
getElementsByTagName( )	HTML 요소중 태그명을 찾아서 선택한다
getElementsByName( )	HTML 요소중 name 속성을 찾아서 선택한다
querySelector( )	요소의 선택방법이 CSS선택방법과 같음
querySelecotrAll( )	위의 요소선택방법은 첫번째만 선택되는 반면 All( )은 모든 요소를 선택함

JS 코드

```
<head>
    <meta charset="UTF-8">
    <title>요소를 직접 선택하는 메서드</title>
    <script>
        window.onload = function() {
          var list1 = document.querySelector('#box1 > ul > li');
          var list2 = document.querySelectorAll('#box2 > ul > li');
          console.log(list1);
          console.log(list2);
          // list1.style.background = "#ff6600";
          // list2[0].style.background = "#ccc";
          // list2.item(1).style.background = "#ddd";
        };
    </script>
</head>
<body>
    <div id="box1">
        <ul> <li>내용1</li><li>내용2</li><li>내용3</li> </ul>
    </div>
    <div id="box2">
        <ul><li>내용4</li><li>내용5</li><li>내용6</li></ul>
    </div>
</body>
```

상태위치 요소선택	설명 및 특징
parentNode	선택된 요소의 부모 노드를 선택합니다
childNodes	선택된 요소의 자식 노드들(요소 노드, 텍스트 노드)을 선택합니다
children	선택된 요소의 자식 노드들을 선택합니다
nextSibling	선택된 요소의 다음 형제 노드를 선택합니다
previousSibling	선택된 요소의 이전 형제 노드를 선택합니다.
firstChild	선택된 요소의 자식 노드 중 첫 번째 노드를 선택합니다.
lastChild	선택된 요소의 자식 노드 중 마지막 노드를 선택합니다.
tagName	선택된 요소의 태그명을 반환합니다.
nodevalue	선택된 노드의 value값을 반환합니다.
nodeType	선택된 노드의 타입을 반환합니다. 1:요소, 2:속성, 3:텍스트
id	선택된 요소의 id값을 반환합니다.
className	선택된 요소의 class값을 반환합니다.

요소 생성	설명 및 특징
createElement( )	요소를 생성하는 메서드입니다.
createTextNode( )	텍스트를 생성하는 메서드입니다.
appendChild( )	요소를 부모와 자식의 관계로 만들어주는 메서드 입니다.

JS 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <style>
        div {
            position: fixed;
            left: 100px;
            top: 10px;
            width: 200px;
            height: 200px; background: #718c00;
        }
    </style>
```

JS 코드

```
    <script>
        function createEle() {
            var bt = document.getElementById('bt');
            function popup() {
                var div = document.createElement('div');
                var p = document.createElement('p');
                var txt = document.createTextNode('자바스크립트');
                p.appendChild(txt);
                div.appendChild(p);
                document.body.appendChild(div);
            }
            bt.onclick = popup;
        }
        addEventListener('load', createEle);
    </script>
</head>
<body>
    <button id="bt">요소생성</button>
</body>
</html>
```

요소 삭제	설명 및 특징
removeChild( )	요소를 제거해 주는 메서드 입니다. 부모요소.removeChild(자식 요소)

속성 추가 및 제거	설명 및 특징
getAttribute( )	요소의 속성 값을 취득할 수 있다
setAttribute( )	요소의 속성을 설정할 수 있다.
removeAttribute( )	요소의 속성값을 제거할 수 있다.

JS 코드

```
<html lang="ko">
<head>
    <style>
        div {
            position: fixed;
            left: 100px;
            top: 10px;
            width: 200px;
            height: 200px;
            background: #718c00;
        }
    </style>
    <script>
        function createEle() {
            var bt = document.getElementById('bt');
            function popup() {
                var div = document.createElement('div');
                var a = document.createElement('a');
                var txt = document.createTextNode('아이콕스');
                a.appendChild(txt);
                a.setAttribute('href', 'http://icoxpublish.com');
                a.setAttribute('target', '_blank');
                a.setAttribute('title', '새창');
                div.appendChild(a);
                document.body.appendChild(div);
            }
            bt.onclick = popup;
        }
        addEventListener('load', createEle);
    </script>
</head>
<body>
    <button id="bt">요소생성</button>
</body>
</html>
```

요소 생성	설명 및 특징
innerHTML	문자 방식으로 요소를 생성하는 방법

JS 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>innerHTML</title>
    <style>
        div {
            position: fixed;
            left: 0px;
            top: 0px;
            width: 200px;
            height: 200px;
            background: #718c00;
        }
        .m1{
            background: #fff;
        }
    </style>
    <script>
        function createEle() {
            var content = document.getElementById('content');
            content.innerHTML = '<p class="m1">자바스크립트</p>';
        }
        addEventListener('load', createEle);
    </script>
</head>
<body>
    <div id="content"></div>
</body>
</html>
```

순서	할일	힌트
1	[신청]버튼을 누르면 텍스트 필드의 이름을 가져와 명단에 추가하는 함수를 작성한다.	appendchild( )함수 사용
2	최근에 입력한 이름을 명단의 맨 위에 표시하도록 소스를 수정합니다.	insertBefore( )함수 사용
3	명단에 추가할 때 삭제 버튼도 함께 표시하도록 소스를 수정합니다.	
4	삭제 버튼을 누르면 해당 이름을 삭제합니다.	removeChild( )함수 사용

HTML 코드

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>DOM - Create & Add Node</title>
  <link rel="stylesheet" href="css/nameList.css">
</head>
<body>
  <div id="container">
    <h1>참가 신청</h1>
    <form action="">
      <input type="text" id="userName" placeholder="이름"
             required autocomplete="off">
      <button onclick="newRegister();return false;">신청</button>
    </form>
    <hr>
    <div id="nameList"></div>
  </div>
  <script src="js/register-result.js"></script>
</body>
</html>
```

JS 코드

```
function newRegister() {
  var newP = document.createElement("p");  // 새 p 요소 만들기
  var userName = document.querySelector("#userName");
  var newText = document.createTextNode(userName.value);
  // 새 텍스트 노드 만들기
  newP.appendChild(newText);
  // 텍스트 노드를 p 요소의 자식 요소로 연결하기
  var delBttn = document.createElement("span");// 새 button 요소 만들기
  var delText = document.createTextNode("X"); // 새 텍스트 노드 만들기
  delBttn.setAttribute("class", "del");  // 버튼에 class 속성 설정하기
  delBttn.appendChild(delText);
  // 텍스트 노드를 button 요소의 자식 요소로 연결하기
  newP.appendChild(delBttn);  //  del 버튼을 p 요소의 자식 요소로
  var nameList = document.querySelector("#nameList");
  nameList.insertBefore(newP, nameList.childNodes[0]);
  // p 요소를 #nameList 맨 앞에 추가하기
  // nameList.appendChild(newP);// p 요소를 #nameList의 자식 요소로 만들기
  userName.value = "";  // 텍스트 필드 지우기
  var removeBttns = document.querySelectorAll(".del");
  for (var i=0; i<removeBttns.length; i++) {
  // removeBttns에 있는 요소 전체를 반복
    removeBttns[i].addEventListener("click", function() {
     // i번째 버튼을 클릭했을 때 실행할 함수 선언
      if (this.parentNode.parentNode)
      // 현재 노드(this)의 부모 노드의 부모 노드가 있을 경우 실행
        this.parentNode.parentNode.removeChild(this.parentNode);
         //. '현재 노드(this)의 부모 노드의 부모 노드'를 찾아 '현재 노드(this)의
         부모 노드(p 노드)' 삭제
    });
  }
}
```

스타일 변경	설명 및 특징
스타일 형식	요소.style.속성 = '속성 값' , 주의할점은 margin-top을 선택할때는 '-' 기호 없이 선택 사용

JS 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>스타일</title>
    <script>
        window.onload = function() {
            var content = document.getElementById('content');
            content.style.width = '200px';
            content.style.height = '200px';
            content.style.border = '4px solid #718c00';
            content.style.textAlign = 'center';
            content.style.lineHeight = '200px';
        }
    </script>
</head>
<body>
    <div id="content">내용</div>
</body>
</html>
```

form 객체	설명 및 특징
form 객체 선택	document 객체의 하위 객체 중 하나로서 form객체의 name 속성으로 선택가능하다.

JS 코드

```
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>form 객체의 선택 방법</title>
    <script>
        window.onload = function() {
            var frm1 = document.frm1;
            var frm2 = document.frm2;
            console.log(frm1.search.placeholder);
            console.log(frm2.subject.placeholder);
            console.log(frm2.credit.placeholder);
            console.log(document.forms[0].elements[0].placeholder);
            console.log(document.forms[1][1].placeholder);
            console.log(document.forms['frm1'].elements['search'].
                        placeholder);
            console.log(document.forms['frm2']
                        ['subject'].placeholder);
        }
    </script>
</head>
<body>
    <form action="#" name="frm1">
        <input type="search" name="search" placeholder="검색어입력">
        <input type="submit" value="확인">
    </form>
    <form action="#" name="frm2">
        <input type="text" name="subject" placeholder="과목입력">
        <input type="password" name="credit" placeholder="학점입력">
        <input type="submit" value="확인">
    </form>
</body>
</html>
```

form 프로퍼티/메서드	설명 및 특징
value	input, textarea 요소의 value 값을 반환합니다.
checked	checkebox나 radio가 체크되어있으면 true, 아니면 false
disabled	요소가 활성화되면 false 비활성화면 true
defaultValue	초기 설정 값을 반환합니다.
length	요소의 개수를 반환합니다.
focus( )	요소에 포커스를 맞춥니다
blur( )	요소에서 포코스를 없애 줍니다
submit( )	form의 요소 값들을 전송합니다.
reset( )	form의 요소 값들을 리셋합니다.

순서	할일	힌트
1	로그인 시 아이디나 패스워드를 입력하지 않았을때 메시지로 경고	

JS 코드

```
<!DOCTYPE HTML>
<html lang="ko">
<head>
    <script>
        window.onload = function() {
            var login = document.login;
            login.onsubmit = function() {
                if(!login.id.value){
                    alert('아이디를 입력해주세요!');
                    login.id.focus();
                    return false;
                }
                if(!login.pw.value){
                    alert('비밀번호를 입력해주세요!');
                    login.pw.focus();
                    return false;
                }
            }
        }
```

JS 코드

```
    </script>
</head>
<body>
    <form action="#" method="post" name="login">
        <div>
            <label for="id">아이디</label>
            <input type="text" name="id" id="id">
        </div>
        <div>
            <label for="pw">비밀번호</label>
            <input type="password" name="pw" id="pw">
        </div>
        <div>
            <input type="submit" value="확인">
        </div>
    </form>
</body>
</html>
```

4.1 프로젝트 : 슬라이더 만들기

HTML코드

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/
libs/font-awesome/4.7.0/css/font-awesome.min.css" />
    <link rel="stylesheet" href="slider.css" />
</head>
<body>
    <div class="slides ">
        <div class="slide">
            <div class="slidecontainer">
                <div class="content">
                    <h1>slide 1</h1>
                    <p>첫번째 슬라이더</p>
                    <a href="javascript:;">Explore Now</a>
                </div>
            </div>
        </div>
        <div class="slide">
            <div class="slidecontainer">
                <div class="content">
                    <h1>slide 2</h1>
                    <p>두번째 슬라이더</p>
                    <a href="javascript:;">Explore Now</a>
                </div>
            </div>
        </div>
```

HTML 코드

```
        <div class="slide active">
            <div class="slidecontainer">
                <div class="content">
                    <h1>slide 3</h1>
                    <p>세번째 슬라이더</p>
                    <a href="javascript:;">Explore Now</a>
                </div>
            </div>
        </div>
        <div class="controls">
            <div class="prev">
                <i class="fa fa-arrow-left"></i>
            </div>
            <div class="next">
                <i class="fa fa-arrow-right"></i>
            </div>
        </div>
        <div class="indicatorContainer"></div>
    </div>
    <script src="./slide.js"></script>
</body>
</html>
```

CSS 코드

```css
body{
    width:100%;
    height: 100%;
    margin: 0;

    box-sizing: border-box;
}
.container_slide
{
    display: flex;
    justify-content: center;
    margin-bottom: 300px;
    height:1000px;

}
.slides
{
    min-height: 900px;
    position: relative;
    overflow: hidden;
    min-height: 900px;
    width: 100%;
   height: 100%;
    border-radius: 20px;
    justify-content: center;



}
.slide:nth-child(1)
{
    background-image: url("./img/slide1.jpeg");
}
.slide:nth-child(2){
    background-image: url("./img/slide2.jpeg");
}
```

CSS 코드

```css
.slide:nth-child(3)
{
    background-image: url("./img/slide3.jpeg");
}

.slide
{
    height: 100vh;
    width: 100%;
    background-repeat: no-repeat;
    background-size: cover;
    position: absolute;
    animation: sliding 2s ease;
    display: none;
}
.slide.active
{
    display: block;
}
@keyframes sliding
{
    from { opacity: 0; transform: scale(1.1); }
    to{ opacity: 1; transform: scale(1); }
}
.slidecontainer
{
    width: 100%;
    height: 100%;
    margin: auto;
    padding: 0 15px;
    position: relative;
}
```

CSS 코드

```css
.slide .content
{
    height: 100%;
    display: flex;
    align-items: flex-end;
    flex-direction: column;
    justify-content: flex-start;
    margin-top: 50px;
    color: #313131;
}
.slide .content h1 {
    font-size: 5rem;
    opacity: 0;
    animation: captionanimation 3s ease-in-out forwards;
    animation-delay: 0.6s;
}
.slide .content p {
   font-size: 18px;
   margin: 5px 0 30px;
   color: #222222;
   text-align: right;
   opacity: 0;
   animation: captionanimation 3s ease-in-out forwards;
   animation-delay: 0.8s;
}
.slide .content a
{   align-items: center;
    background-color: #313131;
    padding: 7px 15px;
    text-decoration: none;
    color: #fff;
    opacity: 0;
    animation: captionanimation 3s ease-in-out forwards;
}
```

CSS 코드

```css
@keyframes captionanimation {
    from { opacity: 1; transform: translateX(100px); }
    to { opacity: 1; transform: translateX(-100px); }
}
.slides .controls {
    width: 100%;
    height: 100%;
    /*margin: 50vh 0 ;*/
    align-items: center;
    position: absolute;
    display: flex;
    justify-content: space-between;
    padding: 0 0px;
}
.slides .controls .prev{
margin-left: 10px;
}
.slides .controls .next{
    margin-right: 10px;
}
.slides .next,
.slides .prev {
    cursor: pointer;
    background-color: #313131;
    width: 40px;
    height: 40px;
    text-align: center;
    font-size: 20px;
    line-height: 40px;
    border-radius: 50%;
    padding: 5px;
    color: #fff;
    opacity: 0.5;
    transition: 0.5s;
}
```

CSS 코드

```
.slides .prev:hover,.slides .next:hover{
   opacity: 1;
}
.slides .indicatorContainer {
    width: 100%;
    text-align: center;
    position: absolute;
    bottom: 30px;
}
.slides .indicatorContainer > div {
    border: 3px solid #313131;
    display: inline-block;
    width: 15px; height: 15px;
    margin-right: 5px; border-radius: 50%;
}
.slides .indicatorContainer > div:last-child {
    margin: 0;
}
.slides .indicatorContainer > div.active {
    background-color: #313131;;
}
/*CSS properties for mobile resposive*/
@media (max-width: 1140px){
    .slidecontainer{
        width: 100%;
        padding: 0 15px;
    }
    .slide{
        background-position-x:30%;
    }
}
@media(max-width:767px){
    .slides .controls{
        display: none;
    }
}
```

JS 코드

```
const slides = document.querySelectorAll(".slide");
const prev = document.querySelector(".prev");
const next = document.querySelector(".next");

let index = 0;

prev.addEventListener("click",function() {
    prevSlide();
});
next.addEventListener("click",function(){
    nextSlide();
});

function prevSlide() {
    if(index ==0)
    {
        index = slides.length - 1;
    }
    else
    {
        index--;
    }
    changeSlide();
}
function nextSlide() {
    if(index == slides.length -1){
        index = 0;
    }
    else{
        index++;
    }
    changeSlide();
}
```

JS 코드

```
function changeSlide(){
    slides.forEach(function(item){
        item.classList.remove("active");
    });

    slides[index].classList.add("active");
    //adding code to show active dot in the indicators
    let indicators=document.querySelectorAll(".indicatorContainer
                                                > div");
    indicators.forEach(function(indicator){
        indicator.classList.remove("active");
    });
    indicators[index].classList.add("active");
    resetAutoplay();
}

//creating dot indicators by using JavaScript start
const indicatorContainer =
document.querySelector(".indicatorContainer");
function buildIndicators(){
    for(let i =0; i< slides.length; i++){
        let element = document.createElement("div");
        element.dataset.i = i+1;
        element.setAttribute("onclick","gotoSlide(this)");

        //making first dot as active by default
        if(i==0){
            element.classList.add("active")
        }


        indicatorContainer.appendChild(element);
    }
}
```

JS 코드

```
buildIndicators();
//creating dot indicators by using JavaScript end

// creating gotoSlide Function
function gotoSlide(element){
    let k = element.dataset.i
    index = k-1;
    changeSlide();
}
//slide autoplay Functionality
let timer = setInterval(nextSlide, 4000);
//1000 = 1 second
function resetAutoplay(){
    clearInterval(timer); //stop the timer
    timer = setInterval(nextSlide,4000); //again start the timer
}
```