훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ Array 객체

◆ Method 종류

종 류	설 명
join(“연결문자”)	배열 객체에 데이터를 연결 문자 기준으로 1개의 문자형 데이터로 반환
reverse()	배열 객체의 데이터의 순서를 거꾸로 바꾼 후 반환
sort()	배열 객체의 데이터를 오름차순으로 정렬
slice(index1, index2)	배열 객체에 데이터중 원하는 인덱스 구간만큼 잘라서 배열 객체로 반환
splice()	배열 객체에 지정 데이터를 삭제하고 그 구간에 새 데이터를 삽입
concat()	2개의 배열 객체를 하나로 결합
pop()	저장된 데이터중 마지막 인덱스에 저장된 데이터 삭제
push(new data)	마지막 인덱스에 새 데이터를 삽입
shift()	첫 번째 인덱스에 저장된 데이터를 삭제
unshift(new data)	가장 앞의 인덱스에 새 데이터 삽입

◆ Property 종류

종 류	설 명
length	배열에 지정된 총 데이터의 개수를 반환

❖ Array 객체

◆ splice()

* 배열의 중간 위치에 데이터를 다른 데이터로 변경하는 기능을 가진 함수

```
splice(시작위치, 개수, 데이터1, 데이터2, …);
```

* 시작 위치부터 지정한 개수 만큼의 데이터를 지정한 데이터로 변경한다.
* 개수와 뒤에있는 데이터의 개수는 아무 관계가 없다.
* 데이터는 없어도 된다. - 중간의 데이터를 삭제하는 기능의 효과가 된다.

❖ window 객체

◆ Method 종류

종 류	설 명
open()	새 창을 열 때 사용
alert()	경고 창을 띄울 때 사용 - 버튼 한개
prompt()	입력 창을 띄울 때 사용 - 버튼 두개 (확인/취소) / 변수에 담아서 사용
confirm()	확인/취소 창을 띄울 때 사용 - 버튼 두개(확인/취소) / 반환값 : true/false
moveTo()	창의 위치를 이동 시킬 때 사용
resizeTo()	창의 크기를 변경 시킬 때 사용
setInterval()	일정 간격으로 지속적으로 실행문을 실행 시킬 때 사용
setTimeout()	일정 간격으로 한번만 실행문을 실행시킬 때 사용

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ window 객체 method

* open() - 새 창을 띄울 때 사용

```
[window.]open("url 경로", "창 이름", "옵션");
```

· 팝업창 열기 옵션값의 종류

옵션값	값 지정법	설 명
toolbar	yes / no	툴바 아이콘의 표시 여부를 설정
location		주소 표시줄의 표시 여부 설정
status		상태 바의 표시 여부 설정
menubar		메뉴 표시줄의 표시 여부 설정
scrollbars		스크롤바의 표시 여부 설정
resizable		창의 크기를 조절 가능하게 할지 여부를 설정
width	픽셀값	창의 폭을 지정
height		창의 높이를 지정

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ window 객체 method

* alert() - 경고 창을 띄울 때 사용

```
[window.]alert("메세지");
```

* prompt() - 질의응답 창을 띄울 때 사용

```
[window.]prompt("질의내용", "기본답변");
```

* confirm() - 확인/취소 창을 띄울 때 사용

```
[window.]confirm("질의내용");
```

❖ 버튼클릭 반환값은 true / false

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ window 객체 method

* **moveTo() - 창의 위치를 이동시킬 때**

```
var 변수이름 = open(“url”, “창이름”, “옵션”);
변수이름.moveTo(x 위치값, y 위치값);
```

픽셀 단위 입력

* **resizeTo() - 창의 크기 변경**

```
var 변수이름 = open(“url”, “창이름”, “옵션”);
[window.]resizeTo(창 너비, 창 높이);
```

픽셀 단위 입력

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ window 객체 method

* **setInterval() - 일정 시간 간격으로 실행문 실행**

```
var 변수이름 = setInterval(“실행문(함수이름)”, 시간간격(1/1000초));
```

* **clearInterval() - 일정 시간 간격으로 실행문 실행을 취소**

```
clearInterval(변수이름);
```

* **setTimeout() - 일정 간격으로 실행문을 한 번만 실행**

```
var 변수이름 = setTimeout(“실행문(함수이름)”, 시간간격(1/1000초));
```

* **clearTimeout() - 일정 시간 간격으로 한 번 실행 실행문 실행을 취소**

```
clearTimeout(변수이름);
```

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ history 객체 method

◆ Method 종류

종 류	설 명
history.back()	이전 방문 페이지로 이동
history.forward()	다음 방문 페이지로 이동
history.go(이동숫자)	이동 숫자만큼 이동 - 이동 숫자가 -2이면 2단계 이전 페이지로 이동

◆ Property 종류

종 류	설 명
length	배열에 지정된 총 데이터의 개수를 반환

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ 선택자 종류

구분	종 류	설 명
직접 선택자	**document.getElementById(“아이디이름”)**	아이디로 요소 선택
	document.getElementsByTagName(“태그”)	태그를 이용해 선택 - 배열 객체로 반환
	document.formName.inutName	폼 요소에 name 속성을 부여하고 그것을 이용해 선택
인접 관계 선택자	**parentNode**	선택 요소의 부모(상위) 요소 선택
	childNodes	선택 요소의 모든 자식(하위) 요소 선택 - 배열 객체로 반환
	children	선택 요소의 자식(하위) 요소 태그만 선택 - 배열 객체로 반환
	firstChild	선택 요소의 첫번째 자식(하위) 요소만 선택
	previousSibling	선택 요소의 이전에 오는 형제 요소 선택
	nextSibling	선택 요소 다음에 오는 형제 요소 선택

❖ **CSS 선택자와 약간의 차이가 있으니 확인하세요!**

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ Event Handler

* 기본형

```
요소.이벤트 = function() {        실행문;      }
```

◈ callback 함수

* 콜백 함수는 파라메터를 통해 다음 실행지점을 지시하는 함수를 전달한다.
* 콜백 함수를 전달받은 함수는 실행 지점 마지막에 호출자 측으로 반환되는 대신 이 콜백 함수로 다시 한 번 진입한다. 즉 재귀함수와 비슷하게 동작한다.
* 호출자가 결과값을 반환받는다.
* 해당 함수가 실행하는 그 순간에 호출자로부터의 동기화가 끊긴다.

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ EventListener

* 자바 스크립트에서 이벤트를 설치하는 방법 - 주로 동적(변동이 있는)인 이벤트 설치에 사용한다.

```
요소.addEventListener("이벤트이름", 실행함수이름);

예 ]
    <script>
        var     temp = document.getElementById("???");
        temp.addEventListener("click", abc);
    </script>
```

```
요소.addEventListener("이벤트이름", function() {
                실행내용;
});
```

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ EventListener

* 하나의 요소에 여러 이벤트를 동시에 처리할 수 있다.
* 같은 이벤트에 여러 함수를 동시에 처리할 수 있다.- 이때는 등록된 함수가 순차적으로 실행된다.
* this를 사용할 수 있다.- 이때 this는 이벤트가 일어난 요소 자체를 의미하는 예약된 변수이다.
* 자바스크립트를 이용한 이벤트 등록에 있어서 함수를 내부적으로 만들어서 사용할 수 있다.
* 이벤트 처리 순서 변경 - HTML의 특성상 여러 요소가 겹쳐서 사용할 필요가 있다.

```
예 ]    <div>???<p>???</p>???</div>
```

* 만약 중첩된 요소에 이벤트가 동시에 설치되면 이벤트 처리 순서는
  안쪽 요소가 먼저 처리되고 바깥 요소가 나중에 실행된다.
* 필요하다면 이벤트 처리 순서를 변경할 수 있다.

```
요소.addEventListener("이벤트이름", 함수이름, true/false);
```

★ true : 바깥 요소 먼저 실행

★ false : 안쪽 요소 먼저 실행 (기본값)

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ removeEventListener

* 불필요한 이벤트는 제거할 수 있다.

```
요소.removeEventListener("이벤트이름", 사용함수);
```

◆ attachEvent()

* 이벤트 등록을 하는 함수로 addEventListener()에 해당

◆ detachEvent()

* 이벤트 제거를 히는 함수로 removeEventListener()에 해당

★ IE 8버전 이전에는 위의 함수를 사용해서 이벤트 등록을 해야만 했었다.

★ 하지만 IE9 이후 버전 부터는 add, remove를 이용해서 이벤트 등록을 하도록 권장하고 있다.

❖ DOM(Document Object Model) 객체

◆ 사용자 브라우저 환경에 따른 이벤트 처리

* 클라이언트가 어떤 버젼의 브라우저를 사용할지 모르므로....

```
var target = document.getElementById("???");
if(target.addEventListener) {
        // 현재 사용자가 웹 브라우저가 addEventListener을 제공하면…
        target.addEventListener("이벤트이름", 함수이름);
}
else if(target.attachEvent) {
        // 현재 사용자가 웹 브라우저가 attachEvent을 제공하면…
        target.attachEvent("이벤트이름", 함수이름);
}
```

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ DOM Node

＊ HTML 문서의 각각의 요소를 트리 구조식으로 만든것

◆ DOM Node에서 원하는 요소를 찾고 동적으로 DOM Node를 만들거나 삭제하는 방법

＊ 태그 안에 기록된 실제 데이터도 하나의 Node로 간주

＊ 실제 데이터를 알아내거나 수정하는 방법

종 류	설 명
innerHTML	강제로 데이터를 입력하거나 수정
firstChild.nodeValue	firstChild는 현재 노드에서 하위 노드를 지칭하는 예약어
childNode[?].nodeValue	childNode[?] 현재 노드에서 지정한 위치(?)의 자식 노드를 지칭하는 예약어

❖ **nodeValue - 실질적인 데이터를 의미하는 예약어**

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ DOM Node를 이용한 동적 HTML 만들기

* 처음에 서버가 클라이언트에게 응답하는 문서에는 존재하지 않는 내용을 필요한 순간에 새롭게 만들어서 사용하는 방법
* 주로 동적 폼을 만들때 사용

함 수	설 명
createElement	새로운 태그를 생성
createTextNode	새로운 데이터 node를 생성
appendChild	새로만든 태그를 필요한 위치에 추가
insertBefore	특정 요소 앞에 새로운 요소를 추가
removeChild	특정 요소를 삭제
replaceChild	특정 요소를 다른 요소로 변경 - 주로 innerHTML 로 처리

훈련교사 : 전 은 석
email : euns_jun@naver.com

❖ DOM(Document Object Model) 객체

◆ DOM Node를 이용한 동적 HTML 만들기

js_domnode01.html

```
<script type="text/javascript">
    var my_div = null;
    var newDiv = null;
    function addElement() {
        // 새로운 div 엘리먼트 생성
        // 내용을 작성
        newDiv = document.createElement("div");
        newDiv.innerHTML = "<h1>안녕! 반가워!</h1>";
        // 생성된 엘리먼트를 추가
        my_div = document.getElementById("org_div1");
        document.body.insertBefore(newDiv, my_div);
    }
</script>
<body onload="addElement()">
<div id='org_div1'> 위의 문장은 동적으로 만들어 진 것입니다.</div>
</body>
```