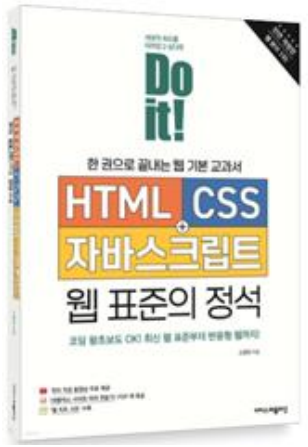
18. 자바스크립트와 객체



18-1 객체 알아보기

18-2 자바스크립트와 내장 객체

18-3 브라우저와 관련된 객체

객체 알아보기

객체(object)란

- 프로그램에서 인식할 수 있는 모든 대상
- 데이터를 저장하고 처리하는 기본 단위

자바스크립트 객체

자바스크립트 안에 미리 객체로 정의해 놓은 것

- 문서 객체 모델(DOM) : 문서 뿐만 아니라 웹 문서 안에 포함된 이미지·링크·텍스트 필드 등을 모두 별도의 객체로 관리
- 브라우저 관련 객체 : 웹 브라우저 정보를 객체로 관리
- 내장 객체 : 웹 프로그래밍에서 자주 사용하는 요소를 객체로 정의해 놓음.

사용자 정의 객체

필요할 때마다 사용자가 직접 만드는 객체

객체와 인스턴스

- 객체는 객체 자체가 아니라 인스턴스 형태로 만들어서 사용
- 인스턴스 : 객체를 틀처럼 사용해서 같은 모양으로 찍어낸 것.

기본형 new 객체명

예) 현재 날짜와 시간 정보 표시하기

```
<script>
  let now = new Date(); // 인스턴스 객체 만들고 변수에 할당하기
  document.write("현재 시각은 " + now) ; // 현재 날짜와 시간 표시하기
</script>
```

객체 알아보기

프로퍼티(property)와 메서드(method)

- 프로퍼티 : 객체의 특징이나 속성
- 메서드 : 객체에서 할 수 있는 동작
- 인스턴스는 객체의 프로퍼티와 메서드를 그대로 물려받음
- 프로퍼티와 메서드를 표시하려면
  인스턴스명 뒤에 마침표(.)를 붙이고 프로퍼티나 메서드 이름 작성

프로퍼티	메서드
제조사	시동 걸기
모델명	움직이기
색상	멈추기
배기량	주차하기

그림 16-2 자동차의 프로퍼티와 메서드

2) now 변수에 저장

```
<script>
let now = new Date();
document.write(`현재 시각은 ${now.toLocaleString()}`) ; // 로컬 형식으로 표시하기
</script>
```

1) Date 객체의 인스턴스를 만들어서

3) Date 객체의 메서드 사용

내장 객체 – Array 객체

배열 만들기

1) 초깃값이 없는 경우

```
let arr1 = new Array(); // 배열의 크기를 지정하지 않음
let arr2 = new Array(4); // 배열의 크기를 지정함
```

2) 초깃값이 있는 경우

```
let arr3 = ["one", "two", "three", "four"]; // 배열 선언
let arr4 = Array("one", "two", "three", "four"); // Array 객체를 사용한 배열 선언
```

배열 요소의 개수 – Array 객체의 length 프로퍼티

실습>> Arr1.html

```
<script>
  let numbers = ["one", "two", "three", "four"];     // 배열 선언 및 초기화
  for(let i = 0; i < numbers.length; i++) {          // 배열의 각 요소에 접근하기
    document.write(`<p>${numbers[i]}</p>`);
  }
</script>
```

배열의 각 요소

one

two

three

four

내장 객체 – Array 객체

Array 객체의 메서드

종류	설명
concat	기존 배열에 요소를 추가해 새로운 배열을 만듭니다.
every	배열의 모든 요소가 주어진 함수에 대해 참이면 true를 반환하고 그렇지 않으면 false를 반환합니다.
filter	배열 요소 중에서 주어진 필터링 함수에 대해 true인 요소만 골라 새로운 배열을 만듭니다.
forEach	배열의 모든 요소에 대해 주어진 함수를 실행합니다.
indexOf	주어진 값과 일치하는 값이 있는 배열 요소의 첫 인덱스를 찾습니다.
join	배열 요소를 문자열로 합칩니다. 이때 구분자를 지정할 수 있습니다.
push	배열의 맨 끝에 새로운 요소를 추가한 후 새로운 length를 반환합니다.
unshift	배열의 시작 부분에 새로운 요소를 추가합니다.
pop	배열의 마지막 요소를 꺼내 그 값을 결과로 반환합니다.
shift	배열에서 첫 번째 요소를 꺼내 그 값을 결과로 반환합니다.
splice	배열에 요소를 추가하거나 삭제합니다.
slice	배열에서 특정한 부분만 잘라 냅니다.
reverse	배열의 배치 순서를 역순으로 바꿉니다.
sort	배열 요소를 지정한 조건에 따라 정렬합니다.
toString	배열에서 지정한 부분을 문자열로 반환합니다. 이때 각 요소는 쉼표(,)로 구분합니다.

내장 객체 – Array 객체

배열끼리 합치는 concat( ) 메서드

- 서로 다른 배열 2개를 합쳐서 새로운 배열을 만듦
- 기존 배열에 영향을 주지 않음

(예) 배열 2개를 합쳐서 새로운 배열 만들기 **실습>> Arr2.html**

```
<script>
  let nums = [1, 2, 3];
  let chars = ['a', 'b', 'c', 'd'];

  let numsChars = nums.concat(chars);
  let charsNums = chars.concat(nums);
  document.write(`nums에 chars 합치면 : ${numsChars}, <br>
chars에 nums 합치면 ${charsNums}`);
  document.write(`<hr>`);
</script>
```

```
nums에 chars 합치면: 1,2,3,a,b,c,d
chars에 nums 합치면: a,b,c,d,1,2,3
```

배열 요소끼리 합치는 join( ) 메서드

- 배열 요소를 연결해서 하나의 문자열로 만듦
- 요소 사시에 원하는 구분자를 넣을 수 있음.
- 구분자를 지정하지 않으면 쉼표(,)로 구분

(예) 배열 안의 요소 합치기 **실습>> Arr3.html**

```
<script>
  let nums = [1, 2, 3];
  let chars = ['a', 'b', 'c', 'd'];
 ……
  let string1 = nums.join();
  document.write(`구분자 없이 : ${string1}<br>`);
  let string2 = chars.join('/');
  document.write(`/ 구분자 지정 : ${string2}`);
  document.write(`<hr>`);
</script>
```

```
구분자 없이: 1,2,3
'/' 구분자 지정: a/b/c/d
```

내장 객체 – Array 객체

새로운 요소를 추가하는 push( ), unshift( ) 메서드

- push( ) 메서드 : 배열 맨 끝에 요소 추가
- unshift( ) 메서드 : 배열 맨 앞에 요소 추가
- **배열의 길잇값 반환, 기존 배열이 바뀜**

(예) 배열에 새로운 요소 추가하기 **실습>> Arr4.html**

```
<script>
  let nums = [1, 2, 3];
  let chars = ['a', 'b', 'c', 'd'];
  ……

  let ret1 = nums.push(4, 5);    // 배열 끝에 추가
  document.write(`length : ${ret1} ¦ 배열 : ${nums}<br>`);
  let ret2 = nums.unshift(0);    // 배열 앞에 추가
  document.write(`length : ${ret2}, ¦ 배열 : ${nums}`);
  document.write(`<hr>`);
</script>
```

length: 5 | 배열: 1,2,3,4,5
length: 6 | 배열: 0,1,2,3,4,5

배열에서 요소를 꺼내는 pop( ), shift( ) 메서드

- pop( ) 메서드 : 배열 뒤쪽에서 요소를 꺼냄
- shift( ) 메서드 : 배열 앞쪽에서 요소를 꺼냄
- 꺼낸 요소를 반환, 기존 배열을 꺼낸 요소가 빠진 상태로 변경됨

(예) 배열에서 요소 꺼내기 **실습>> Arr5.html**

```
<script>
  let nums = [1, 2, 3];
  let chars = ['a', 'b', 'c', 'd'];
  ……
  let popped1 = chars.pop();    // 마지막 요소 꺼냄
  document.write(`꺼낸 요소 : ${popped1}, ¦ 배열 : ${chars}<br>`);
  let popped2 = chars.shift();    // 첫 번째 요소 꺼냄
  document.write(`꺼낸 요소 : ${popped2}, ¦ 배열 : ${chars}`);
  document.write(`<hr>`);
</script>
```

꺼낸 요소: d | 배열: a,b,c
꺼낸 요소: a | 배열: b,c

내장 객체 – Array 객체

중간에 요소를 추가하거나 삭제하는 splice( ) 메서드

- 배열 중간에 2개 이상의 요소를 추가하거나 삭제할 수 있음
- 새로운 배열이 결괏값으로 반환됨

1) 괄호 안에 인수가 1개일 경우

인수가 지정한 인덱스의 요소부터 배열의 맨 끝 요소까지 삭제

2) 괄호 안에 인수가 2개일 경우

- 첫 번째 인수는 인덱스값이고 두 번째 인수는 삭제할 요소의 개수
- 메서드를 실행한 후에는 삭제한 요소를 반환하고, 기존 배열은 나머지 요소만 남음

3) 괄호 안에 인수가 3개 이상일 경우

첫 번째 인수는 배열에서 삭제할 시작 위치, 두 번째 인수는 삭제할 개수, 세 번째 인수부터는 삭제한 위치에 새로 추가할 요소를 지정

실습>> Arr6.html

```
<script>
  let study = ['html', 'css', 'javascript', 'jquery',
'react', 'nodejs'];

  // 인수가 1개일 경우
  let js = study.splice(2);
  document.write(`반환된 배열 : ${js}<br>` );
  document.write(`변경된 배열 : ${study}`);
  document.write(`<hr>`);

  // 인수가 2개일 경우
  let jquery = js.splice(1,1);
  document.write(`반환된 배열 : ${jquery}<br>`);
  document.write(`변경된 배열 : ${js}`);
  document.write(`<hr>`);

  // 인수가 3개 이상일 경우
  let modernJs = js.splice(1, 0, 'typescript');
  document.write(`반환된 배열 : ${modernJs}<br>`);
  document.write(`변경된 배열 : ${js}`);
</script>
```

내장 객체 – Array 객체

기존 배열을 바꾸지 않으면서 추출하는 slice( ) 메서드

- 요소를 여러 개 꺼낼 수 있음
- 요소를 추출한 후에도 기존 배열이 바뀌지 않음

1) 괄호 안에 인수가 1개일 경우

인수가 지정한 인덱스의 요소부터 마지막 요소까지 꺼내서 반환

2) 괄호 안에 인수가 2개일 경우

첫 번째 인수는 시작 인덱스, 두 번째 인수는 끝 인덱스의 직전 인덱스

실습>> Arr7.html

```
<script>
  let colors = ["red", "green", "blue", "white", "black"];

  document.write(colors.slice(2)); //인덱스 2부터 끝까지
  document.write("<hr>");
  document.write(colors.slice(2, 4)); // 인덱스 2, 3
</script>
```

```
blue,white,black
blue,white
```

slice() 메서드는 기존 배열에 영향을 주지 않지만, splice() 메서드는 요소를 추가·삭제하면 기존 배열 자체가 수정됨
→ 기존 배열에서 꺼낸 요소로 새로운 배열을 만들어 사용하려면 slice() 메서드를 사용하고,
기존 배열의 일부 요소만 삭제하려면 splice() 메서드를 선택하는 것이 좋다

내장 객체 – Date 객체

Date 객체 인스턴스 만들기

현재 날짜로 설정할 경우

```
new Date();
```

특정 날짜로 설정할 경우 – 괄호 안에 날짜 또는 날짜와 시간 입력

```
new Date('2024-02-25')
```

```
new Date('2024-02-25T18:00:00)'
```

자바스크립트의 날짜와 시간 입력 방식

1) YYYY-MM-DD 형식

```
new Date("2024")
new Date("2024-02")
new Date("2024-02-25")
```

2) YYYY-MM-DDTHH 형식

```
new Date("2024-02-25T18:00:00")
new Date("2024-02-25T18:00:00Z")
```

3) MM/DD/YYYY 형식

```
new Date("02/25/2024")
```

4) 이름 형식

```
new Date("Mon Jan 20 2024 15:00:41 GMT+0900 (대한민국 표준시)")
```

실습>> days.html 사용

```
<script>
  // Do it! Date 객체 연습하기
  let now = new Date();       // 오늘 날짜를 객체로 지정
  let firstDay = new Date("2024-01-01");   // 시작 날짜를 객체로 지정

  let toNow = now.getTime();
  let toFirst = firstDay.getTime();
  let passedTime = toNow - toFirst;  // 첫날부터 오늘까지 지난 시간(밀리 초)

  passedTime = Math.round(passedTime/(1000*60*60*24));  // 밀리 초를 일 수로 계산하고 반올림

  document.querySelector('#result').innerText = passedTime;
</script>
```

내장 객체 – Date 객체

Date 객체의 메서드

날짜/시간 정보를 가져오는 메서드,
날짜/시간 정보를 설정하는 메서드,
날짜/시간 형식을 바꿔주는 메서드로 구분됨

구분		설명
날짜·시간 정보 가져오기	getFullYear()	연도를 4자리 숫자로 표시합니다.
	getMonth()	0~11 사이의 숫자로 월을 표시합니다. 0부터 1월이 시작되고 11은 12월입니다.
	getDate()	1~31 사이의 숫자로 일을 표시합니다.
	getDay()	0~6 사이의 숫자로 요일을 표시합니다. 0부터 일요일이 시작되고 6은 토요일입니다.
	getTime()	1970년 1월 1일 자정 이후의 시간을 밀리 초(1/1000초)로 표시합니다.
	getHours()	0~23 사이의 숫자로 시를 표시합니다.
	getMinutes()	0~59 사이의 숫자로 분을 표시합니다.
	getSeconds()	0~59 사이의 숫자로 초를 표시합니다.
	getMilliseconds()	0~999 사이의 숫자로 밀리초를 표시합니다.
날짜·시간 설정하기	setFullYear()	연도를 4자리 숫자로 설정합니다.
	setMonth()	0~11 사이의 숫자로 월을 설정합니다. 0부터 1월이 시작되고 11은 12월입니다.
	setDate()	1~31 사이의 숫자로 일을 설정합니다.
	setTime()	1970년 1월 1일 자정 이후의 시간을 밀리초로 설정합니다.
	setHours()	0~23 사이의 숫자로 시를 설정합니다.
	setMinutes()	0~59 사이의 숫자로 분을 설정합니다.
	setSeconds()	0~59 사이의 숫자로 초를 설정합니다.
	setMilliseconds()	0~999 사이의 숫자로 밀리초를 설정합니다.
날짜·시간 형식 바꾸기	toLocaleString()	현재 날짜와 시간을 현지 시간(local time)으로 표시합니다.
	toString()	Data 객체 타입을 문자열로 표시합니다.

내장 객체 – Math 객체

Math 객체의 특징

- 수학 계산과 관련된 메서드가 많이 포함되어 있지만 수학식에서만 사용하는 것은 아님.
- 무작위 수가 필요하거나 반올림이 필요한 프로그램 등에서도 Math 객체의 메서드 사용함
- Math 객체는 인스턴스를 만들지 않고 프로퍼티와 메서드 사용

Math 객체의 프로퍼티

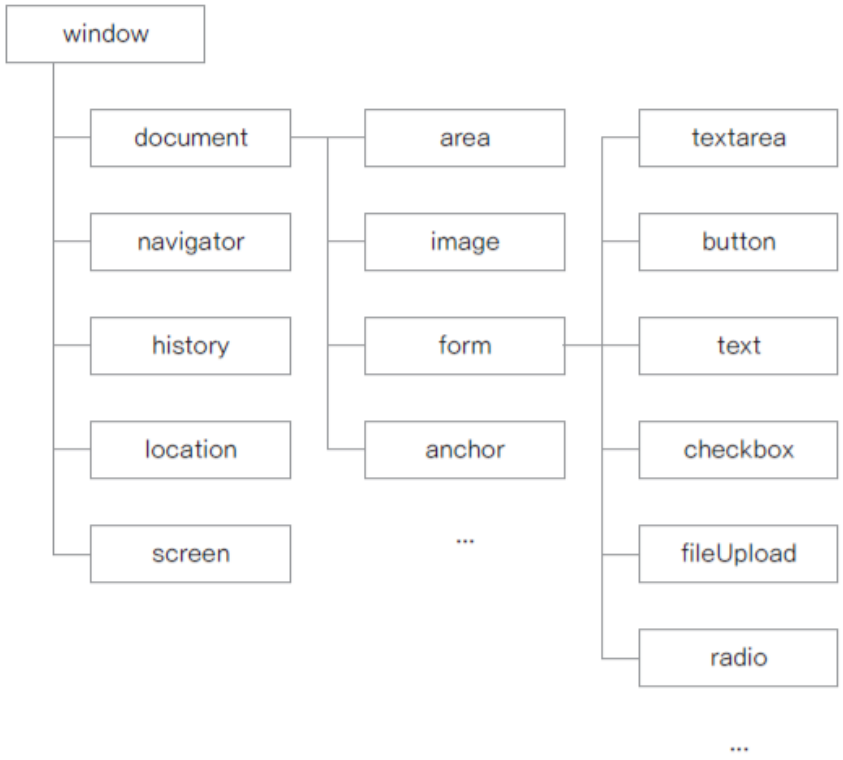
종류	설명
E	오일러 상수
PI	원주율(π) (약 3.1415926535897930의 값)
SQRT2	$\sqrt{2}$(약 1.41421356237309510의 값)
SQRT1_2	$1/\sqrt{2}$(약 0.70710678118654760의 값)
LN2	$\log_e 2$(약 0.69314718055994530의 값)
LN10	$\log_e 10$(약 2.30258509299404060의 값)
LOG2E	$\log_2 e$(약 1.44269504088896340의 값)
LOG10E	$\log_{10} e$(약 0.43429448190325180의 값)

Math 객체의 메서드

종류	설명
abs()	절댓값을 반환합니다.
acos()	아크 코사인(arc cosine)값을 반환합니다.
asin()	아크 사인(arc sine)값을 반환합니다.
atan()	아크 탄젠트(arc tangent)값을 반환합니다.
atan2()	아크 탄젠트(arc tangent)값을 반환합니다.
ceil()	매개변수의 소수점 이하 부분을 올립니다.
cos()	코사인(cosine)값을 반환합니다.
exp()	지수 함수를 나타냅니다.
floor()	매개변수의 소수점 이하 부분을 버립니다.
log()	매개변수에 대한 로그(log)값을 반환합니다.
max()	매개변수 중 최댓값을 반환합니다.
min()	매개변수 중 최솟값을 반환합니다.
pow()	매개변수의 지숫값을 반환합니다.
random()	0과 1 사이의 무작위 수를 반환합니다.
round()	매개변수의 소수점 이하 부분을 반올림합니다.
sin()	사인(sine)값을 반환합니다.
sqrt()	매개변수에 대한 제곱근을 반환합니다.
tan()	탄젠트(tangent)값을 반환합니다.

브라우저 관련 객체

브라우저 관련 객체의 계층 구조



종류	설명
window	브라우저 창이 열릴 때마다 하나씩 만들어집니다. 브라우저 창 안의 요소 중에서 최상위에 있습니다.
document	웹 문서마다 하나씩 있으며 <body> 태그를 만나면 만들어집니다. HTML 문서의 정보가 담겨 있습니다.
navigator	현재 사용하는 브라우저의 정보가 들어 있습니다.
history	현재 창에서 사용자의 방문 기록을 저장합니다.
location	현재 페이지의 URL 정보가 담겨 있습니다.
screen	현재 사용하는 화면 정보를 다룹니다.

브라우저 관련 객체 – window 객체

window 객체의 프로퍼티

주로 웹 브라우저 창의 정보를 가져오거나 값을 바꿀 때 사용

종류	설명
document	브라우저 창에 표시된 웹 문서에 접근할 수 있습니다.
frameElement	현재 창이 다른 요소 안에 포함되어 있을 경우 그 요소를 반환하고, 반대로 포함되어 있지 않으면 null을 반환합니다.
innerHeight	내용 영역의 높이를 나타냅니다.
innerWidth	내용 영역의 너비를 나타냅니다.
localStorage	웹 브라우저에서 데이터를 저장하는 로컬 스토리지를 반환합니다.
location	window 객체의 위치 또는 현재 URL을 나타냅니다.
name	브라우저 창의 이름을 가져오거나 수정합니다.
outerHeight	브라우저 창의 바깥 높이를 나타냅니다.
outerWidth	브라우저 창의 바깥 너비를 나타냅니다.
pageXOffset	스크롤했을 때 수평으로 이동하는 픽셀 수로 scrollX와 같습니다.
pageYOffset	스크롤했을 때 수직으로 이동하는 픽셀 수로 scrollY와 같습니다.
parent	현재 창이나 서브 프레임의 부모입니다.
screenX	브라우저 창의 왼쪽 테두리가 모니터 왼쪽 테두리에서 떨어져 있는 거리를 나타냅니다.
screenY	브라우저 창의 위쪽 테두리가 모니터 위쪽 테두리에서 떨어져 있는 거리를 나타냅니다.
scrollX	스크롤했을 때 수평으로 이동하는 픽셀 수를 나타냅니다.
scrollY	스크롤했을 때 수직으로 이동하는 픽셀 수를 나타냅니다.
sessionStorage	웹 브라우저에서 데이터를 저장하는 세션 스토리지를 반환합니다.

window 객체의 메서드

window 객체는 기본 객체이므로 'window.'를 생략하고 메서드 이름만 사용해도 됨

종류	설명
alert()	알림 창을 표시합니다.
blur()	현재 창에서 포커스를 제거합니다.
close()	현재 창을 닫습니다.
confirm()	[확인], [취소] 버튼이 있는 확인 창을 표시합니다.
focus()	현재 창에 포커스를 부여합니다.
moveBy()	현재 창을 지정한 크기만큼 이동합니다.
moveTo()	현재 창을 지정한 좌표로 이동합니다.
open()	새로운 창을 엽니다.
postMessage()	메시지를 다른 창으로 전달합니다.
print()	현재 문서를 인쇄합니다.
prompt()	프롬프트 창에 입력한 텍스트를 반환합니다.
resizeBy()	지정한 크기만큼 현재 창의 크기를 조절합니다.
resizeTo()	동적으로 브라우저 창의 크기를 조절합니다.
scroll()	문서에서 특정 위치로 스크롤합니다.
scrollBy()	지정한 크기만큼씩 스크롤합니다.
scrollTo()	지정한 위치까지 스크롤합니다.
sizeToContent()	내용에 맞게 창의 크기를 맞춥니다.
stop()	로딩을 중지합니다.

새 브라우저 창을 여는 open() 메서드

기본형 `window.open(경로, 창 이름, 창 옵션)`

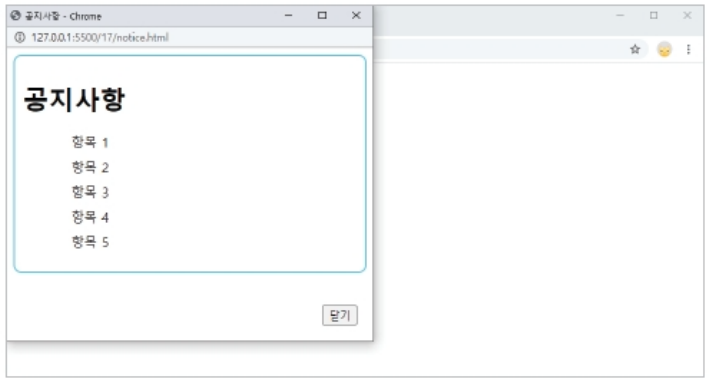
❶ 경로: 팝업 창에 표시할 문서나 사이트의 경로(주소)

❷ 창 이름: 팝업 창의 이름

❸ 창 옵션: left, top 속성을 사용해 위치를 정하거나 width, height 속성을 사용해 크기 지정

(예) 팝업 창 표시하기

```
<p>문서를 열면 팝업 창이 표시됩니다.</p>
<script>
  window.open("notice.html", "", "width=500, height=400");
</script>
```



창 이름 지정하기

창 이름을 지정하면 페이지를 새로고침해도 팝업 창은 한번 표시

팝업 창 위치 지정하기

- left 속성 : 화면 왼쪽 측면 기준으로 얼마나 떨어져 있는지 지정
- top 속성 : 화면 위쪽을 기준으로 해서 얼마나 떨어져 있는지 지정

(예) 팝업 창의 이름과 위치 지정하기

```
<p>왼쪽에서 100픽셀, 위에서 200픽셀 떨어진 위치에 <br> 팝업 창이 표시됩니다.</p>
<script>
  window.open("notice.html", "pop", "width=500, height=400, left=100, top=200");
</script>
```

새 브라우저 창을 여는 open() 메서드

팝업 차단 고려하기

중요한 내용을 팝업 창으로 보여 주어야 한다면 팝업 차단된 상태인지 체크하여 사용자에게 알려 주는 것이 좋다.

웹 브라우저에서 팝업을 차단하면 window.open()은 null 반환

→ window.open() 메서드를 실행한 후 반환값을 체크하면 팝업이 차단되었는지 알아낼 수 있다.

popup-check.html 확인

(예) 팝업이 차단된 브라우저의 알림 창 표시하기


```
<body onload="openPopup()">
<p>문서를 열면 팝업 창이 표시됩니다</p>
<script>
  let blocked = false;
    function openPopup() {
      let newWin = window.open('notice.html', 'pop', 'width=500, height=400');
      if (newWin == null) {
        alert("팝업이 차단되어 있습니다. 팝업 차단을 해제해 주세요.")
      }
    }
</script>
</body>
```

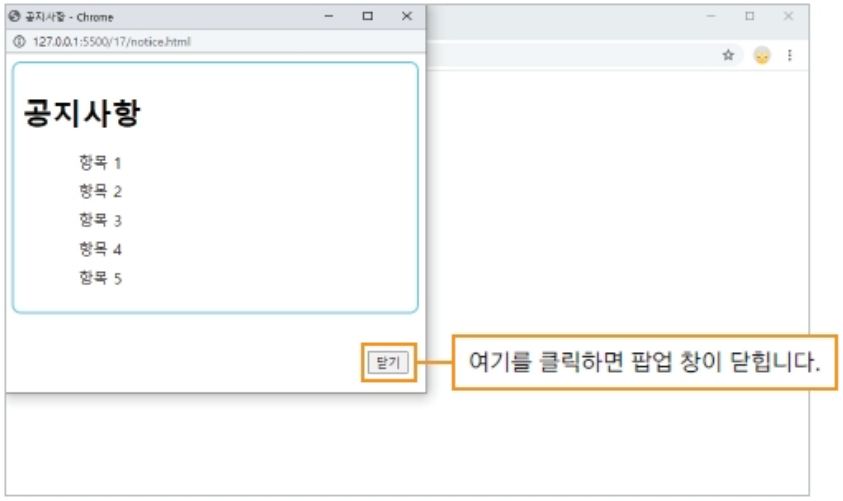
브라우저 창을 닫는 close() 메서드

기본형 `window.close()`

(예) 버튼을 사용해 팝업 창 닫기

```
<button onclick="javascript:window.close();">닫기</button>
```

notice.html 확인



브라우저 관련 객체 – navigator 객체, history 객체

navigator 객체

- 사용하는 브라우저가 많아지고, 웹 애플리케이션이 등장하면서 navigator 객체에 여러 프로퍼티가 등장하고 있음.
- 일부 브라우저에서만 지원하는 프로퍼티도 있음

주요 프로퍼티

종류	설명
battery	배터리 충전 상태를 알려 줍니다.
cookieEnabled	쿠키 정보를 무시하면 false, 허용하면 true를 반환합니다.
geolocation	모바일 기기를 이용한 위치 정보를 나타냅니다.
language	브라우저 UI의 언어 정보를 나타냅니다.
oscpu	현재 운영체제 정보를 나타냅니다.
userAgent	현재 브라우저 정보를 담고 있는 사용자 에이전트 문자열입니다.

history 객체

방문한 사이트 주소가 배열 형태로 저장됨

프로퍼티와 메서드

구분		설명
프로퍼티	length	현재 브라우저 창의 history 목록에 있는 항목의 개수, 즉 방문한 사이트 개수가 저장됩니다.
메서드	back()	history 목록에서 이전 페이지를 현재 화면으로 불러옵니다
	forward()	history 목록에서 다음 페이지를 현재 화면으로 불러옵니다
	go()	history 목록에서 현재 페이지를 기준으로 상대적인 위치에 있는 페이지를 현재 화면으로 불러옵니다. 예를 들어 history.go(1)은 다음 페이지를 가져오고, history.go(-1)은 이전 페이지를 불러옵니다.

브라우저 관련 객체 – location 객체, screen 객체

location 객체

- 현재 문서의 URL 주소 정보가 담겨 있음
- 이 정보를 편집해서 브라우저 창에 열 사이트/문서 지정

구분		설명
프로퍼티	hash	URL 중에서 #로 시작하는 해시 부분의 정보를 담고 있습니다.
	host	URL의 호스트 이름과 포트 번호를 담고 있습니다.
	hostname	URL의 호스트 이름이 저장됩니다.
	href	전체 URL입니다. 이 값을 변경하면 해당 주소로 이동할 수 있습니다.
	pathname	URL 경로가 저장됩니다.
	port	URL의 포트 번호를 담고 있습니다.
	protocol	URL의 프로토콜을 저장합니다.
	password	도메인 이름 앞에 username과 password를 함께 입력해서 접속하는 사이트의 URL일 경우에 password 정보를 저장합니다.
	search	URL 중에서 ?로 시작하는 검색 내용을 저장합니다.
	username	도메인 이름 앞에 username을 함께 입력해서 접속하는 사이트의 URL일 경우에 username 정보를 저장합니다.
메서드	assign()	현재 문서에 새 문서 주소를 할당해서 새 문서를 가져옵니다.
	reload()	현재 문서를 다시 불러옵니다.
	replace()	현재 문서의 URL을 지우고 다른 URL의 문서로 교체합니다.
	toString()	현재 문서의 URL을 문자열로 반환합니다.

location.html 확인

```
<div id="display">
  <script>
    document.write(`<p><b>location.href : </b>${location.href}</p>`);
    document.write(`<p><b>location.host : </b>${location.host}</p>`);
    document.write(`<p><b>location.protocol :
</b>${location.protocol}</p>`);
  </script>
</div>
<button onclick="location.replace('http://www.easyspub.com')">이지스
퍼블리싱 홈페이지로 이동하기</button>
```

브라우저 관련 객체 – location 객체, screen 객체

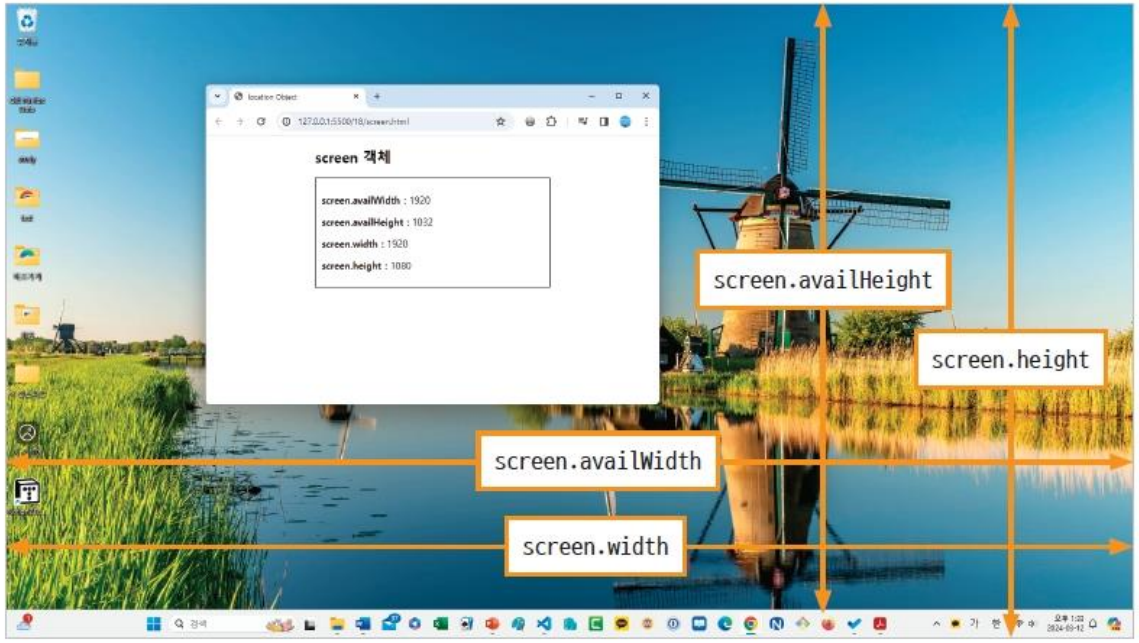
screen 객체

사용자의 화면 크기, 정보

구분		설명
프로퍼티	availHeight	UI 영역(예를 들어 윈도우의 작업 표시줄이나 Mac의 독)을 제외한 영역의 높이를 나타냅니다.
	availWidth	UI 영역을 제외한 내용 표시 영역의 너비를 나타냅니다.
	colorDepth	화면에서 픽셀을 렌더링할 때 사용하는 색상 수를 나타냅니다.
	height	UI 영역을 포함한 화면의 높이를 나타냅니다.
	orientation	화면의 현재 방향을 나타냅니다.
	pixelDepth	화면에서 픽셀을 렌더링할 때 사용하는 비트 수를 나타냅니다.
	width	UI 영역을 포함한 화면의 너비를 나타냅니다.
메서드	lockOrientation()	화면 방향을 잠급니다.
	unlockOrientation()	화면 방향 잠금을 해제합니다.

(예) 화면의 너비와 높이 알아내기

```
<script>
  document.write(`<p><b>screen.availWidth : </b>${screen.availWidth}</p>`);
  document.write(`<p><b>screen.availHeight : </b>${screen.availHeight}</p>`);
  document.write(`<p><b>screen.width : </b>${screen.width}</p>`);
  document.write(`<p><b>screen.height : </b>${screen.height}</p>`);
</script>
```



실습>> popup문서 가운데 나타나게 하기 / center.html 수정

```
<script>
  // Do it! Screen 객체 연습하기 - 화면 중앙에 팝업 창 표시하기

  // 기존 코드
  // window.open('notice.html', 'pop', 'width=500, height=400');

  // 수정 코드
  function openCenter(doc, win, width, height){
    let left = (screen.availWidth - width) / 2;  // 팝업 창의 왼쪽 좌표
    let top = (screen.availHeight - height) / 2;   // 팝업 창의 위쪽 좌표
    let opt = `left=${left}, top=${top}, width=${width}, height=${height}`;
    window.open(doc, win, opt);
  }
  openCenter('notice.html', 'pop', 500, 400)
</script>
```