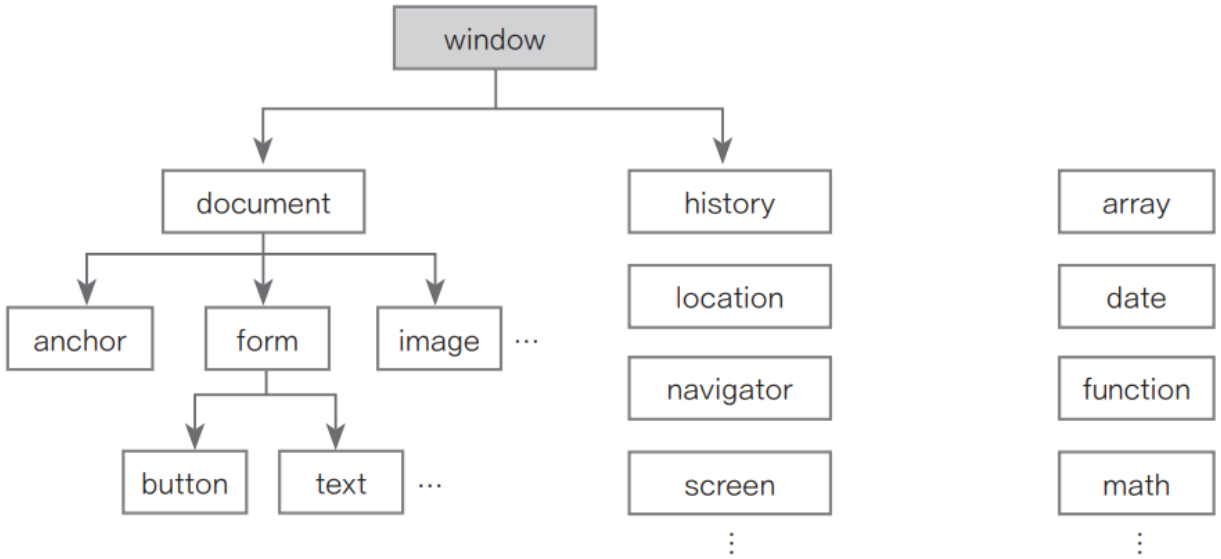
내장 객체

내장 객체란

- 사용자가 가져다 쓸 수 있도록 미리 만들어진 객체.
- 웹 브라우저에 웹 문서가 열리면 가장 먼저 window 객체가 만들어지고 하위에 웹 브라우저 요소에 해당하는 객체들이 만들어진다.
- window 객체를 비롯해 하위에 연결된 객체들은 모두 HTML 웹 API에 만들어진 객체들



문자열 객체 속성과 메서드

문자열의 길이 - length

문자열의 길이를 찾을 때에는 length 프로퍼티 사용

문자열.length

```
let str = "Good morning!";
let greeting = "안녕하세요?"
str.length   // 13
greeting.length    // 6
```

특정 위치의 문자에 접근하기 – charAt()

- charAt() 메서드 사용

문자열.charAt(*위치*)

```
str = "Good morning!";
str.charAt(3)   // "d"
```

문자열의 대소문자 바꾸기

영문자 문자열의 경우에는 문자열을 모두 대문자로, 또는 모두 소문자로 바꿀 수 있다.

```
문자열.toUpperCase()    // 문자열을 모두 대문자로 변환
문자열.toLowerCase()    // 문자열을 모두 소문자로 변환
```

```
str4 = "Good morning."
str4.toUpperCase()        // ‘GOOD MORNING.’
str4.toLowerCase()        // ‘good morning.’
```

문자열의 부분 문자열 추출하기 - substring()

- 시작 위치부터 끝 위치의 직전까지 추출해서 반환한다.
- 끝 위치를 지정하지 않으면 시작 위치부터 문자열 끝까지 추출해서 반환한다.

문자열.substring(시작 위치)

문자열.substring(시작 위치, 끝 위치)

```
// str4 = "Good morning."
str4.substring(5)      // ‘morning.’
```

구분자를 사용해 문자 쪼개기 – split()

문자열에서 구분자를 기준으로 문자열을 나눈다.

문자열.split(*구분자*)

```
str5 = "Hello everyone"
array1 = str5.split(" ")    // ["Hello", "everyone"]
array2 = str.split("")      // ["H","e","l","l","o"," ", "e","v","e","r","y","o","n","e"]
```

따옴표 사이에 공백이 없음

배열 객체

새로운 배열 만들기

1) 빈 배열을 만들고 값 할당하기

```
let season = []
season[0] = "spring"
season[1] = "summer"
season    // ["spring", "summer"]
```

인덱스	0	1	2
값	철수	영희	고은
길이	1	2	3

2) 리터럴 표기법으로 만들기

```
let pets = ["dog", "cat", "parrot"]
pets    // ["dog", "cat", "parrot"]
```

3) Array 객체의 인스턴스 만들기

```
let fruits = new Array("사과", "복숭아", "포도")
fruits    // ["사과", "복숭아", "포도"]
```

배열 값 수정하기 및 추가하기

배열은 인덱스를 사용해서 원하는 위치의 값을 변경할 수 있다.

이미 값이 있는 위치에 값을 할당하면 기존 값은 지워진다.

```
let pets = ["dog", "cat", "parrot"]
pets[1] = "hamster"
pets    //  ["dog", "hamster", "parrot"]
```

중간에 인덱스를 건너뛰고 값을 할당할 수 있다.

```
let fruits = new Array('사과', '복숭아', '포도')
fruits[4] = "배"
fruits       // ["사과", "복숭아", "포도", 비어 있음, "배"]
fruits[3]    // undefined
```

empty라고 표시될 수도 있음

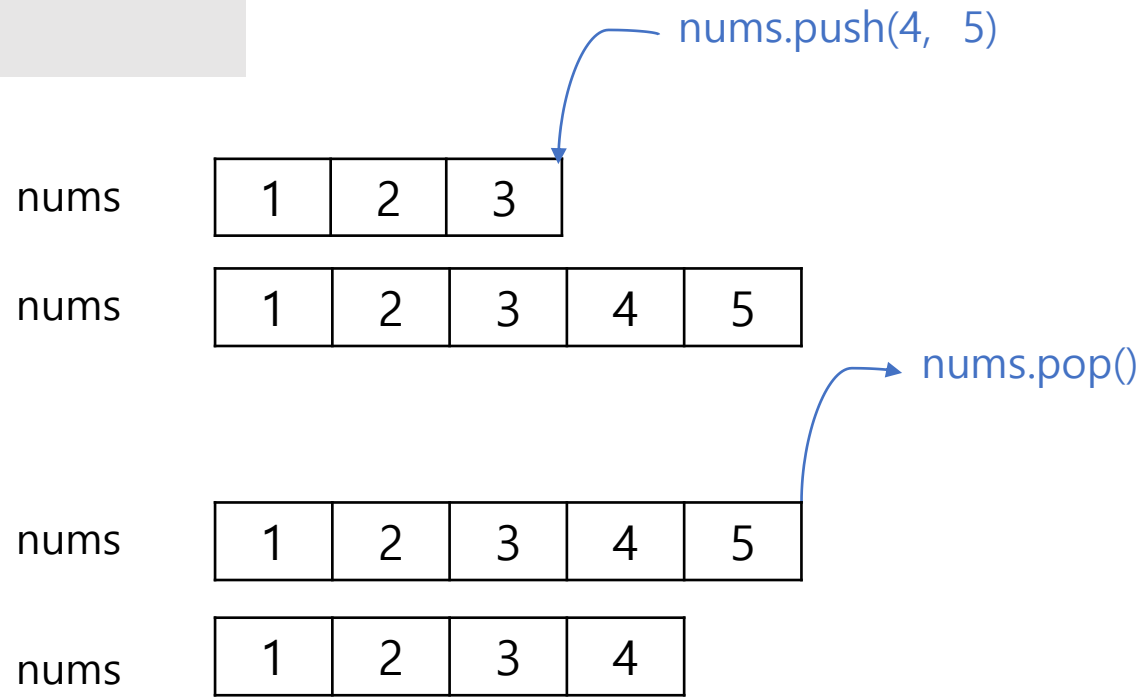
배열 끝에 값, 추가 삭제하기

- push() : 배열의 맨 앞 부분에 값 추가
- pop() : 배열의 맨 끝 값 제거.

```
배열.push(값)     // 맨 끝에 값 추가. 배열 개수 반환
배열.pop()        // 마지막 값 제거.
```

```
let nums = [1,  2,  3]
nums.push(4,  5)
nums
```

```
nums.pop()
```



배열 앞에 값, 추가 삭제하기

unshift() : 배열의 맨 앞에 값 추가

shift() : 배열 맨 앞에 있는 값 제거

```
배열.unshift(값)        // 맨 앞에 값 추가
배열.shift()            // 맨 앞의 요소 제거
```

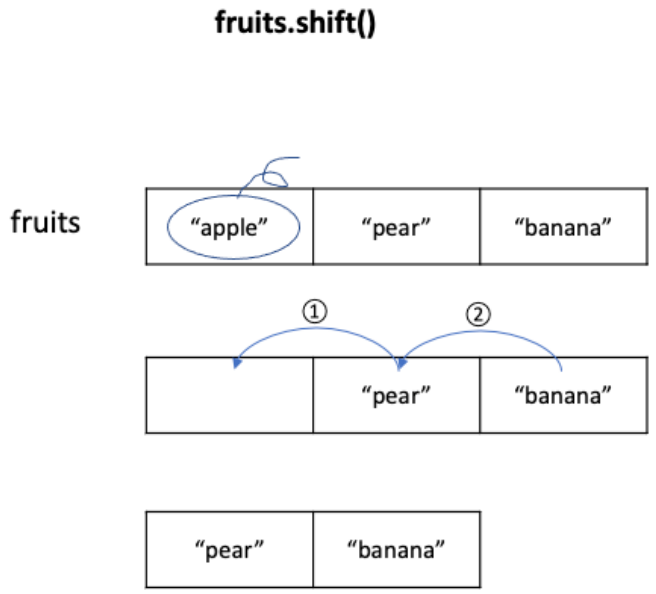
```
let fruits = ["apple", "pear", "banana"]
fruits.shift()     // "apple"
fruits             // ["pear", "banana"]
```

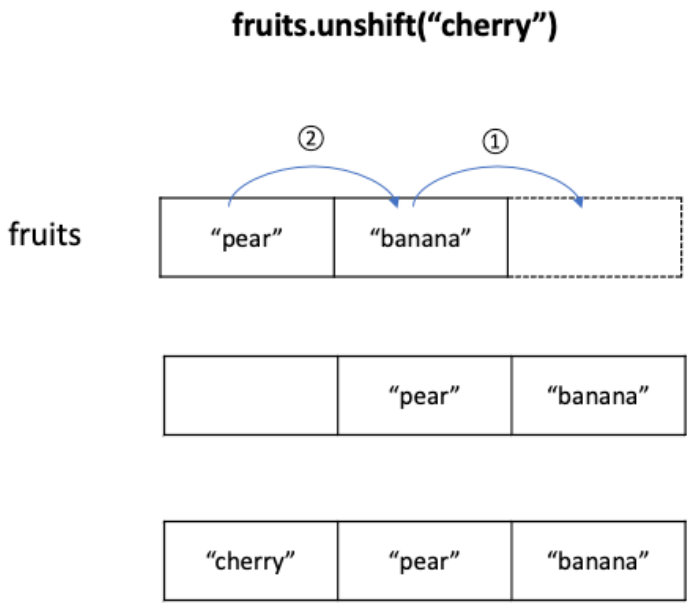
```
fruits.unshift("cherry")      // 3
fruits                        // ["cherry", "pear", "banana"]
```

배열 앞에 값, 추가 삭제하기

shift() 메서드와 unshift() 메서드는 배열에서 맨 앞의 요소를 변경하기 때문에 요소를 추가하거나 제거하는 작업 외에도 인덱스를 변경해야 한다.

→ 배열의 요소가 많거나 요소의 내용이 복잡할수록 shift(), unshift() 메서드의 실행 시간이 좀 더 길어진다.





Date 객체

Date 객체

자바스크립트 내장 객체 중에서 Date 객체는 날짜와 시간에 대한 정보를 조절할 수 있는 객체.
현재 날짜와 시간을 홈페이지에 출력하거나 달력을 표시할 수도 있고,
특정일까지 얼마나 남았는지 카운트다운하는 등 여러 가지로 응용할 수 있다.

객체와 인스턴스

자바스크립트에 정의된 내장 객체를 사용할 때에는 객체의 프로퍼티와 메서드를 가진 새로운 객체를 만든 후 여기에 식별자를 붙여 프로그래밍에서 사용한다.

→ Date 객체를 사용하려면 자바스크립트의 Date 객체를 똑같이 만들어서 그것을 사용한다.

→ 이렇게 내장 객체와 똑같은 모양으로 찍어낸 객체를 인스턴스(instance) 객체라고 한다.

Date 객체의 인스턴스 만들기

1) 현재 날짜를 기준으로 인스턴스 만들기

```
new Date()
```

```
let today = new Date()
today
```

today 변수에는 Date 객체의 인스턴스가 저장되었기 때문에
이제부터 today 변수는 Date 객체의 프로퍼티와 메서드를 사용할 수 있다

```
today.getDate()
```

Date 객체의 인스턴스 만들기

2) 특정 날짜를 기준으로 인스턴스 만들기

Date 다음의 괄호 안에 날짜 정보 입력

(예) '2025년 2월 25일'이라는 날짜 정보를 객체에 저장한 후 프로그램에 사용하려면 다음과 같이 입력한다.

```
new Date("2025-02-25")    // 2025년 2월 25일
```

시간 정보까지 함께 지정하려면 날짜 다음에 대문자 T를 추가한 후 그 뒤에 시간을 입력한다.

```
new Date("2025-02-25T18:00:00")
```

자바스크립트의 날짜/시간 입력 방식

1) YYYY-MM-DD: '연도-월-일' 형태로 지정. 연도(YYYY)나 월까지만(YYYY-MM) 사용할 수도 있다.

```
new Date("2025-02-25")
new Date("2025-02")
new Date("2025")
```

2) YYYY-MM-DDTHH:MM:SS : '연도-월-일-T-시:분:초'의 형태로 지정합니다.

```
new Date("2010-02-25T18:00:00")
```

3) MM/DD/YYYY: 슬래시를 사용해서 월/일/연도 순으로 지정합니다.

```
new Date("02/25/2010")
```

4) 전체 형식: 월과 요일 이름은 전체 이름이나 줄여쓴 이름 모두 사용할 수 있습니다.

```
new Date("Thu Aug 17 2017 15:00:41 GMT+0900 (대한민국 표준시)")
```

Date 객체의 메서드

메서드 이름 앞에 get이 붙어 있으면 날짜나 시간 정보를 가져오는 메서드

getFullYear()	현지 시간을 기준으로 연도값을 가져옴.
getMonth()	월값을 가져옴. 0~11 사이의 숫자가 반환.(0 - 1월, 11 - 12월)
getDate()	일값을 가져옴. 1~31 사이의 숫자로 반환.
getDay()	요일값을 가져옴. 0~6 사이의 숫자가 반환.(0 – 일요일, 6 – 토요일)
getTime()	1970년 1월 1일 00:00 이후의 시간을 밀리초로 표시.
getHours()	시값을 가져옴. 0~23 사이의 숫자로 반환.
getMinutes()	분값을 가져옴. 0~59 사이의 숫자로 반환.
getSeconds()	초값을 가져옴. 0~59 사이의 숫자로 반환.
getMilliseconds()	밀리초값을 가져옴. 0~999 사이의 숫자로 반환.

Date 객체의 메서드

메서드 이름 앞에 set이 붙어 있으면 날짜나 시간 정보를 설정하는 메서드

setFullYear()	현지 시간을 기준으로 연도를 설정.
setMonth()	현지 시간을 기준으로 월을 설정
setDate()	현지 시간을 기준으로 일을 설정
setTime()	1970년 1월 1일 00:00부터 지난 시간을 밀리초로 설정
setHours()	현지 시간을 기준으로 시를 설정.
setMinutes()	현지 시간을 기준으로 분을 설정.
setSeconds()	현지 시간을 기준으로 초를 설정.
setMilliseconds()	현지 시간을 기준으로 밀리초를 설정.

Date 객체의 메서드

메서드 이름 앞에 to가 붙어 있으면 국제 표준 형식으로 된 날짜 표시를 다른 형식으로 바꿔 주는 메서드

toGMTString()	'요일 일 월 연도 시:분:초 UTC' 형식으로 표시.
toLocaleString()	'월/일/년도 시:분:초' 형식으로 표시.
toString()	'요일 월 날짜 시:분:초 UTC+대한민국 표준시' 형식으로 표시.
toDateString()	Date에서 날짜 부분만 표시.
toTimeString()	Date에서 시간 부분만 표시.

Math 객체

Math 객체

삼각 함수나 로그 함수를 비롯한 수학 연산 함수를 가지고 있는 내장 객체

Math 객체는 따로 객체의 인스턴스를 사용하지 않고 속성이나 함수를 사용합니다

주요 함수

함수	설명
abs(x)	숫자의 절댓값을 반환합니다.
cbrt(x)	숫자의 세제곱근을 반환합니다.
ceil(x)	인수보다 크거나 같은 수 중에서 가장 작은 정수를 반환합니다(숫자의 소수점 이하를 올립니다).
floor(x)	인수보다 작거나 같은 수 중에서 가장 큰 정수를 반환합니다(숫자의 소수점 이하를 버립니다).
random( )	0과 1 사이의 무작위 수(난수)를 반환합니다.
round(x)	숫자에서 가장 가까운 정수를 반환합니다(숫자의 소수점 이하를 반올림합니다).

주요 속성

PI – 파이 값

Math 객체

콘솔 창에서 연습하기

1) 반지름이 10인 원의 원둘레와 넓이

```
radius = 10
2 * Math.PI * radius        // 원둘레(2πr)
Math.PI * radius * radius   // 원 넓이(πr²)
```

2) 소수점 처리

```
radius = 10
2 * Math.PI * radius                    // 62.83185307179586
Math.round(2 * Math.PI * radius)        // 63. 반올림
Math.ceil(2 * Math.PI * radius)         // 63. 올림
Math.floor(2 * Math.PI * radius)        // 62. 버림
```

- Math.ceil(값) : 소수점 이하 올림.
- Math.floor(값) : 소수점 이하 버림.
- Math.roud(값) : 소수점 이하 반올림.

Math 객체

소수점 이하 자릿수를 표시하려면?

toFixed() 함수 : 이 함수는 Number 객체의 메서드.

숫자.toFixed(자릿수)

콘솔 창에서 연습하기

```
(Math.PI).toFixed(3)    // 3.142 - 소수점 이하 셋째 자리까지
(Math.PI).toFixed(1)    // 3.1 -  소수점 이하 첫째 자리까지
```

Math 객체

random() : 0과 1 사이(1 포함 안됨)의 숫자를 무작위로 반환한다.

콘솔 창에 다음 명령을 반복해서 입력해 보자

```
Math.random()
```

```
> Math.random()
0.5425530194313202
> Math.random()
0.598716990886506
> Math.random()
0.5793898738696266
> Math.random()
0.9810186578896352
> Math.random()
0.27136637738771574
>
```

키보드에서 위로 화살표를 누르면 명령 반복 입력 가능

Math 객체

만일 1과 100 사이의 무작위 수를 구하려면?

1) random() 결과에 100을 곱한다 – 0.0000에서 99.9999 사이의 값이 반환됨.
2) 위 계산 값에 1을 더한다 – 1.0000에서 100.9999 사이의 값이 반환됨.
3) floor() 함수를 사용해 소수점 이하를 버린다

콘솔 창에 다음 명령을 반복해서 입력하기

```
Math.floor(Math.random() * 100 + 1)
```