KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

배열 메서드

[KB] IT's Your Life

KB 국민은행

배열 메서드

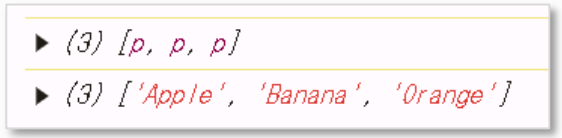
Array.from()

- 컬렉션을 매개변수로 받아 배열객체로 변환하는 static 함수
  - 유사 배열, NodeList, ...

```
const fruits = document.querySelectorAll('.fruits p');  // NodeList 객체
const fruitArray = Array.from(fruits);  // NodeList 객체를 Array로 변환
```

ex01-1.html

```
<!DOCTYPE html>
<html lang="ko">
  ...
  <body>
    <div class="fruits">
      <p>Apple</p>
      <p>Banana</p>
      <p>Orange</p>
    </div>

    <script>
      const fruits = document.querySelectorAll('.fruits p');
      // fruits는 3개의 P 태그를 포함한 노드 리스트(배열과 비슷한 구조)다.
      // 이제 fruits를 배열로 변환하자.
      const fruitArray = Array.from(fruits);
      console.log(fruitArray);
      // 이제 배열을 취급하므로 map()을 사용할 수 있다.
      const fruitNames = fruitArray.map((fruit) => fruit.textContent);
      console.log(fruitNames);
      // ["Apple", "Banana", "Orange"]
    </script>
  </body>
</html>
```

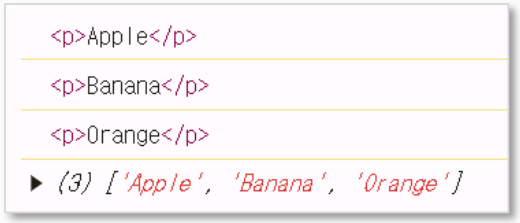


ex01-2.html

```
<!DOCTYPE html>
<html lang="ko">
  ...
  <body>
    <div class="fruits">
      <p>Apple</p>
      <p>Banana</p>
      <p>Orange</p>
    </div>

    <script>
      const fruits = Array.from(document.querySelectorAll('.fruits p'));
      const fruitNames = fruits.map((fruit) => fruit.textContent);
      console.log(fruitNames); // ["Apple", "Banana", "Orange"]
    </script>
  </body>
</html>
```

ex01-3.html

```
<!DOCTYPE html>
<html lang="ko">
  ...
  <body>
    <div class="fruits">
      <p>Apple</p>
      <p>Banana</p>
      <p>Orange</p>
    </div>

    <script>
      const fruits = document.querySelectorAll('.fruits p');
      const fruitArray = Array.from(fruits, (fruit) => {
        console.log(fruit);
        return fruit.textContent;
      });

      console.log(fruitArray); // ["Apple", "Banana", "Orange"]
    </script>
  </body>
</html>
```



배열 메서드

Array.of()

- 전달 받은 모든 인수로 배열을 생성하는 static 함수

  const digits = Array.of(1, 2, 3, 4, 5);

ex02.js

```
const digits = Array.of(1, 2, 3, 4, 5);

console.log(digits);
```

```
[ 1, 2, 3, 4, 5 ]
```

배열 메서드

메서드	설명
indexOf(item, start)	배열에서 요소를 찾아 위치를 리턴한다.
lastIndexOf(item, start)	역순으로 요소를 찾아 위치를 리턴한다.
push(a,b,c,....)	배열 끝에 요소를 추가한다.
pop()	마지막 요소를 제거하고 리턴한다.
shift()	배열 처음의 원소를 제거하고 리턴한다.
unshift(a,b,c,....)	배열 처음에 요소를 추가한다.
reverse()	배열을 거꾸로 뒤집는다.
sort(sortfunction)	배열을 정렬한다. 인수로 값을 비교하는 함수를 지정할 수 있으며 생략시 사전순으로 정렬된다.
slice(start, end)	start~end 범위의 요소를 따로 떼어내어 새로운 배열을 만든다.
splice(index,n,a, b, c, ...)	배열 일부를 수정한다. 일정 범위를 삭제하고 새로운 요소를 삽입한다.
concat(a,b,c,....)	여러 개의 배열을 합친다.
join(deli)	배열 요소를 하나의 문자열로 합친다. 구분자를 지정할 수 있으며 생략시 콤마로 구분한다.

01_join.js

```
const ar = [0, 1, 2, 3];

console.log("ar = " + ar.join());
console.log("ar = " + ar.join(", "));
console.log("ar = " + ar.join(" -> "));
console.log("ar = " + ar.toString());
console.log("ar = " + ar);
```

02_reverse.js

```
const ar = [0, 1, 2, 3];
console.log("ar = " + ar);

ar.reverse();
console.log("ar = " + ar);
```

03_indexof.js

```
const ar = ["태연", "유리", "윤아", "써니", "수영",
              "유리", "서현", "효연"];
console.log("ar = " + ar);

let sunny = ar.indexOf("써니");
console.log("써니는 " + sunny + "번째에 있다. ");

let yuri = ar.indexOf("유리");
console.log("유리는 " + yuri + "번째에 있다. ");
let yuri = ar.lastIndexOf("유리");
console.log("유리는 뒤에서 " + yuri + "번째에 있다. ");

let suji = ar.indexOf("수지");
if (suji == -1) {
    console.log("수지는 소녀시대가 아니다.");
}
```

04_pushpop.js

```
const ar = [0, 1, 2, 3];
console.log("ar = " + ar);

ar.push(100, 200);
console.log("ar = " + ar);

ar.push(300);
console.log("ar = " + ar);

ar.pop();
console.log("ar = " + ar);
```

05_shiftunshift.js

```
const ar = [0, 1, 2, 3];
console.log("ar = " + ar);

ar.unshift(100, 200);
console.log("ar = " + ar);

ar.unshift(300);
console.log("ar = " + ar);

ar.shift();
console.log("ar = " + ar);
```

배열 메서드

splice

splice(index,howmany,item1,.....,itemX)

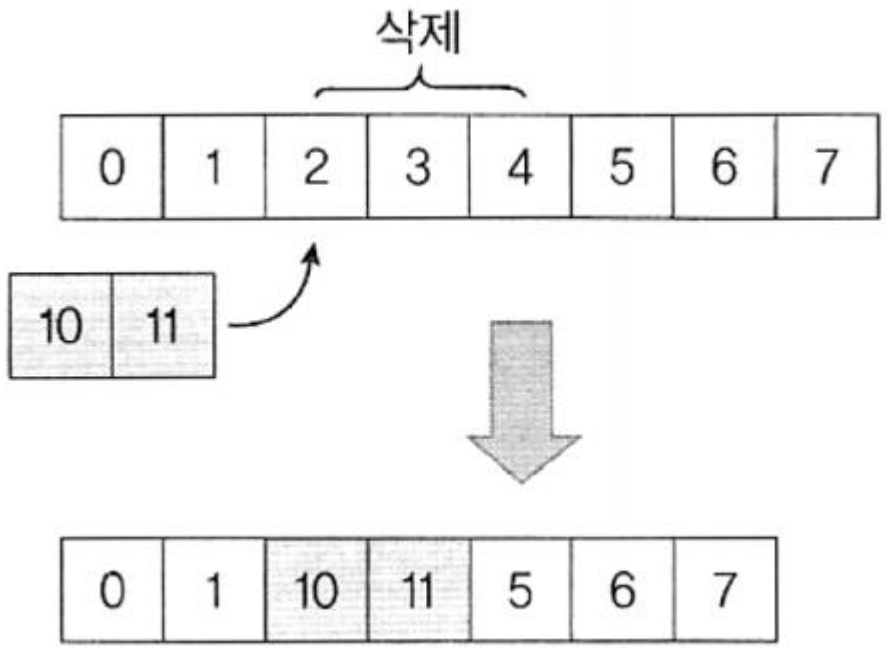
- splice(2, 1, 10, 11, 12)

splice(2, 0, a, b, c)는 2번째 위치에서 지우지는 말고 a, b, c를 삽입

배열 메서드

splice

splice(2, 3, a, b, c)



06_splice.js

```
// 삽입만 하기
const ar1 = [0, 1, 2, 3, 4, 5, 6, 7];
ar1.splice(2, 0, 10, 11, 12);
console.log("ar1 = " + ar1);

// 삭제만 하기
const ar2 = [0, 1, 2, 3, 4, 5, 6, 7];
ar2.splice(2, 3);
console.log("ar2 = " + ar2);

// 삭제 후 삽입 하기
const ar3 = [0, 1, 2, 3, 4, 5, 6, 7];
ar3.splice(2, 3, 10, 11);
console.log("ar3 = " + ar3);
```

07_slice.js

```
const ar = [0, 1, 2, 3, 4, 5, 6, 7];
const subar = ar.slice(2, 5);
console.log("ar = " + ar);
console.log("subar = " + subar);
```

08_concat.js

```
const ar1 = [0, 1, 2];
const ar2 = [3, 4, 5, 6, 7];
const ar3 = ["수지", "아이유", "김태희"];

const ar4= ar1.concat(ar2);
console.log("ar4 = " + ar4);

const ar5 = ar1.concat(ar2, ar3);
console.log("ar5 = " + ar5);
```

09_sortarray.js

```
const score = [82, 96, 54, 76, 92, 99, 69, 88];
console.log("before = " + score);

score.sort();
console.log("after = " + score);
```

10_sortcompare.js

```
const score = [82, 96, 54, 76, 92, 99, 69, 88];
console.log("before = " + score);

function compare(left, right) {
    return right - left;
}

score.sort(compare);
console.log("after = " + score);
```

11_descending.js

```
const score = [82, 96, 54, 76, 92, 99, 69, 88];
console.log("before = " + score);

score.sort();
score.reverse();
console.log("after = " + score);
```

12_numbersort.js

```
const score = [82, 96, 54, 76, 9, 100, 69, 88];
console.log("before = " + score);

score.sort();
console.log("after = " + score);
```

12_numbersort2.js

```
const score = [82, 96, 54, 76, 9, 100, 69, 88];
console.log("before = " + score);

score.sort(function (left, right) {
    return left - right;
});
console.log("after = " + score);
```

13_casesort.js

```
const country = ["korea", "USA", "Japan", "China"];
console.log("before = " + country);

country.sort();
console.log("after = " + country);
```

13_casesort2.js

```
const country = ["korea", "USA", "Japan", "China"];
console.log("before = " + country);

country.sort(function (left, right) {
    let left2 = left.toLowerCase();
    let right2 = right.toLowerCase();
    if (left2 < right2) return -1;
    if (left2 > right2) return 1;
    return 0;
});

console.log("after = " + country);
```

14_foreach.js

```
const score = [82, 96, 54, 76, 9, 100, 69, 88];
let sum = 0;

for (let i =0; i < score.length; i++) {
    sum += score[i];
}

console.log("sum  = " + sum);
```

14_foreach2.js

```
const score = [82, 96, 54, 76, 9, 100, 69, 88];
let sum = 0;

score.forEach(function(value) {
    sum += value;
});

console.log("sum  = " + sum);
```

15_map.js

```
const score = [82, 96, 54, 76, 9, 100, 69, 88];

const score2 = score.map(function(value) {
    return value * 2;
});

console.log("score2  = " + score2);
```

16_filter.js

```
const score = [82, 96, 54, 76, 9, 100, 69, 88];

const score2 = score.filter(function(value) {
    return value >= 80;
});

console.log("score2  = " + score2);
```

배열 메서드

reduce()

- 배열의 각 요소에 대해 주어진 리듀서(reducer) 함수를 실행하고, 하나의 결과값을 반환
- 형식
  - arr.reduce(callback[, initialValue])

- callback 함수이 인자
  - accumulator
    - 누산기. 콜백 함수의 반환값이 다음 호출의 인자로 전달
  - currentValue
    - 처리할 현재 요소
  - currentIndex(Optional)
    - 처리할 현재 요소의 인덱스
  - array(Optional)
    - reduce()를 호출한 배열

17_reduce.js

```
const arr = [1, 2, 3, 4, 5];
const result = arr.reduce((acc, cur, idx) => acc += cur, 0);
console.log(result);  // 15

const arr2 = [1, 2, 3, 4, 5];
const result2 = arr2.reduce((acc, cur, idx) => acc += cur, 10);
console.log(result2);  // 25
```