웹 개발자 부트캠프 과정

CodingOn

문자열 관련 내장 메소드

let str = “Harry Potter-!”

- **length** : 문자열의 길이를 반환(공백포함)

```
console.log(str.length); // 14
```

- **toUpperCase() & toLowerCase()** : 문자열 전체를 대문자, 혹은 소문자로 변경

```
console.log(str.toUpperCase()); //  HARRY POTTER-!
console.log(str.toLowerCase()); //  harry potter-!
```

- **indexOf(“)** : 매개변수로 문자열을 받아서 몇번째 인덱스인지 숫자 반환

```
console.log(str.indexOf("t")); // 8
```

let str = “Harry Potter-!”

- **slice(start, end)** : start 부터 end-1 까지 슬라이싱, 해당 부분 문자열 추출.
  매개변수로 음수값도 가능

```
console.log( str.slice(6, 12)); // "Potter"
```

- **replace(문자열1, 문자열2)** :문자열1을 문자열2로 변경

```
console.log(str.replace("t", "s")); // "Poster"
```

- **replaceAll(문자열1, 문자열2)** : 문자열1을 전부찾아서 문자열 2로 바꿔줌

```
console.log(str.replaceAll("t", "s")); // "Posser"
```

- **repeat(n)** : 문자열에 대해 n번 반복

```
console.log(str.repeat(3)); // "Harry Potter x 3"
```

let str = “Harry Potter-!”

- **trim()** : 문자열의 양끝 공백 없애기
- **split()** : 매개변수로 들어온 문자열을 기준으로 str을 쪼개서 배열로 저장

```
console.log(str.split(""));
```

```
▶ (14) ['H', 'a', 'r', 'r', 'y', ' ', 'P', 'o', 't', 't', 'e', 'r', '-', '!']
```

```
console.log(str.split(" "));
```

```
▶ (2) ['Harry', 'Potter-!']
```

배열 관련 내장 메소드

배열 관련 method

- arr.push(): 배열 끝에 추가
- arr.pop(): 배열 끝 요소 제거
- arr.shift(), arr.unshift(): 배열 앞에 제거/추가
- arr.include(요소): 배열에 해당 요소가 있는지 확인

배열 관련 method

- arr.length : 배열의 길이 반환
- arr.indexOf() : 문자열에서의 indexOf와 마찬가지로 매개변수에 해당하는 배열의 인덱스를 받아옴.
  단, 매개변수로 문자열만 넣을 수 있는 것은 아님!
- arr.reverse() : 배열 순서 뒤집어서 반환
- arr.join(): join 안의 문자열 기준으로 문자열로 병합

메소드 체이닝

메소드 체이닝 (method chaining)

- 여러 메소드를 연결해서 사용하는 개념!
- 단, 사용한 메소드가 반환(return) 값을 가지고 있는 경우에만 사용이 가능!

- ‘hello’.split(“”) → [‘H’, ‘e’, ‘l’, ‘l’, ‘o’] 라는 배열이 반환 됨

- 배열에는 reverse() 라는 메소드가 존재
- ‘hello’.split(“”).reverse() 는 [‘H’, ‘e’, ‘l’, ‘l’, ‘o’].reverse() 와 동일!
- [‘H’, ‘e’, ‘l’, ‘l’, ‘o’].reverse() → [‘o’,’l’,’l’,’e’,’H’] 와 동일

- ‘hello’.split(“”).reverse().join(“”) → [‘o’,’l’,’l’,’e’,’H’].join(“”) 과 동일

배열에서 반복

배열) 기본 for문 사용

```
let numbers = [1, 2, 3, 4, 5, 6];
let fruits = ["사과", "바나나", "수박", "포도", "파인애플"];

for (let i = 0; i < numbers.length; i++) {
  console.log(numbers[i]);
}

for (let i = 0; i < fruits.length; i++) {
  console.log(fruits[i]);
}
```

배열) for of 반복문

```
let numbers = [1, 2, 3, 4, 5, 6];
let fruits = ["사과", "바나나", "수박", "포도", "파인애플"];


for (let number of numbers) {
  console.log(number);
}

for (let fruit of fruits) {
  console.log(fruit);
}
```

배열) [].forEach

```
let numbers = [1, 2, 3, 4, 5, 6];
let fruits = ["사과", "바나나", "수박", "포도", "파인애플"];

numbers.forEach(function (number, index, array) {
  console.log(number, index, array);
});

fruits.forEach(function (fruit, i, arr) {
  console.log(fruit, i, arr);
});
```

배열의 합

```
let numbers = [1, 2, 3, 4, 5, 6];
var sum1 = 0;
var sum2 = 0;
var sum3 = 0;

for (let i = 0; i < numbers.length; i++) {
  console.log(numbers[i]);
  sum1 = sum1 + numbers[i];
}

for (let num of numbers) {
  sum2 = sum2 + num;
}

numbers.forEach((num) => {
  sum3 = sum3 + num;
});

console.log(sum1, sum2, sum3);
```

배열에서의 기타 메소드

- arr.filter()
  - 배열 내부에서 조건에 부합하는 요소만 찾아서 “배열로” 반환
- arr.find()
  - 배열 내부에서 조건에 부합하는 첫번째 요소를 찾아서 “값”으로 반환
- arr.map()
  - 익명함수에 쓰여진 연산결과를 새로운 배열로 반환
- 위의 메소드들은 **매개변수로 익명함수**가 들어간다는 공통점이 있음

참고) for ~ in...

- Object 에 사용가능한 반복문
- Key로 접근

```
for (let k in obj) {

}
```