6_JS(ES6)_PART_10

- 비구조화(Destructuring) 할당
- 스프레드(...) 연산자

비구조화(Destructuring) 할당 (1)

```
const restaurant = {
    name: 'Super Pizza',
    location: 'Rome, Italy',
    startMenus: ['Focaccia', 'Bruschetta', 'Garlic Bread']
    mainMenus: ['Pizza', 'Pasta', 'Risotto'],

    order: function(ftarterIndex, mainIndex){
        return [this.startMenus[starterIndex], this.mainMenus[mainIndex]]
    }
}
```

> 배열의 값을 인덱스로 참조하는 것이 일반적이지만, 다음과 같이 구조를 분해해서 할당 받을 수 있다. 이러한 방법을 `비구조화 할당`이라고 한다.
>
> ```
> const arr = [1, 2, 3]
> const [x, y, z] = arr;
> // 출력값 : 1 2 3
> ```
>
> ⭐ **협업에서 많이 사용하고 있는 문법이다.**
>
> ```
> const [first, ,third] = restaurant.startMenus;
> console.log(first);
> console.log(third);
> // 출력값 : Focaccia
> //         Garlic Bread
> ```
>
> - **메서드 실행하기**
>
> ```
> console.log(restaurant.order(1, 2));
> // 출력값 : Bruschetta, Risotto
> ```
>
> `this.startMenus[starterIndex]` 와 `this.mainMenus[mainIndex]` 가 잘 참조되어 출력이 될것이다.
>
> - ⭐ **비구조화 할당 사용해보기**
>
> ```
> const [startMenu, mainMenu] = restaurant.order(1, 2);
> console.log(startMenu, mainMenu);
> ```

- 중첩된 배열

```
const nested = [2, 4, [5, 6]];
const [i, , j] = nested;
console.log(i, j);

//출력된 값 : 2 [5, 6]
```

- 배열안에 배열 원소를 직접 넣어주고 싶다면? (비구조화 할당 응용)

```
const nested = [2, 4, [5, 6]];
const [i, , [j,k]] = nested;
console.log(i, j ,k);
//출력된 값 : 2, 5, 6
```

비구조화(Destructuring) 할당 (2)

```
const restaurant = {
    name: 'Super Pizza',
    location: 'Rome, Italy',
    startMenus: ['Focaccia', 'Bruschetta', 'Garlic Bread']
    mainMenus: ['Pizza', 'Pasta', 'Risotto'],

    order: function(ftarterIndex, mainIndex){
        return [this.startMenus[starterIndex], this.mainMenus[mainIndex]]
    },

    opening: {
        mon: {
            open: 11,
            close:22
        },
        tue: {
            open: 10,
            close: 23
        },
        sat: {
            open: 12,
            close: 20
        }
    }
}
```

- 비구조 할당으로 참조 가능

```
const { name, opening } = restaurant;
console.log(name);
console.log(opening);
```

```
{
    mon: {open: 11, close: 22},
    tue: {open: 10, close: 23},
    sat: {open: 12, close: 20}
}
```

`restaurant` 이라는 객체 안에 `name` 과 `opening` 이라는 프로퍼티가 있다.

restaurant의 name을 참조할것이고, restaurant의 opening을 참조할 것이다.

- **다른 이름으로 변수를 선언하고 싶다면? (비구조 할당의 자주 쓰이는 문법 중 하나)**

```
const { name: retaurantName, opening: hours } = restaurant;
console.log(retaurantName);
console.log(hours);
```

- **오브젝트도 중첩된 것들도 비구조 할당을 통해 선언 가능하다.**

```
const {
    name: restaurantName,
    opening: { mon, tue },
} = restaurant;

console.log(mon);
console.log(tue);

// 출력값
// { open: 11, close: 22 }
// { open: 10, close: 23 }
```

비구조화(Destructuring) 할당 (3)

```
const restaurant = {
    name: 'Super Pizza',
    location: 'Rome, Italy',
    startMenus: ['Focaccia', 'Bruschetta', 'Garlic Bread']
    mainMenus: ['Pizza', 'Pasta', 'Risotto'],

    order: function(ftarterIndex, mainIndex){
        return [this.startMenus[starterIndex], this.mainMenus[mainIndex]]
    },

    opening: {
        mon: {
            open: 11,
            close:22
        },
        tue: {
            open: 10,
            close: 23
        },
        sat: {
```

```
            open: 12,
            close: 20
        },
    },

    orderDelivery: function({startIndex, mainIndex, time, address }) {
        //this로 restaurant 메뉴를 받아오기
        console.log(`주문 접수 ${this.startMenus[startIndex]} 와
                        ${this.mainMenus[mainIndex]}. ${time} 시간에 맞춰서
                        ${address}로 배송 해주세요.`)
    }
}
```

- ⭐ 실행하기

```
restaurant.orderDelivery({
    satrtIndex: 1,
    mainIndex: 2,
    time: "14:00",
    address: "Seoul, Seoul Korea",
})
```

```
주문접수 Bruschetta와 Risotto. 14:00시간에 맞춰서 Seoul, South Korea로 배송해 주세요.
```

■ 스프레드(...) 연산자

: `스프레드(...) 연산자`는 이터러블 객체에서만 가능

🤔 이터러블 : 순회가능한 객체, For----of 반복문 사용이 가능

이터러블 (순회하다, 반복하다)	이터러블 아님
배열, 문자열, Map, Set	일반 객체 (Object)

```
// 배열은 이터러블 => 스프레드 연산자를 사용할 수 있다.
const arr = [1, 2, 3];
//...
const newArray = [5, 6, ...arr];
console.log(newArray);
// 출력값 : [5, 6, 1, 2, 3]
```

- 문자열 스프레드 연산자

```
const str = "Owen";
console.log([...str]);

//출력값 : ['O', 'w', 'e', 'n']
```

- 만약 배열은 지운다면?

```
const str = "Owen";
console.log(...str);

//출력값 :  O w e n
```

> 각각의 문자가 출력된다.

- 다른 예제

```
const [a, b, ...others] = [1, 2, 3, 4, 5];
console.log(others);

//출력값 :  [3, 4, 5]
```

```
const [Psizza, Risotto, ...otherFoods]
= [...restaurant.mainMenus, ...restaurant.startMenus]
```

■ 다른 예제

```
const { sat, ...weekdays } = retaurant.opening

console.log(weekdays);
```

```
const add = function (...numbers) {
  let sum = 0;
    for(let i = 0; i < numbers.length; i++) {
        sum += numbers[i];
    }
    return sum;
};

add(2, 3);
add(2, 3, 5, 6);
add(54, 564, 434, 234 55);
```

- JS 문법은 ES6 버전에 많이 바뀌었다.
- 비구조화 할당
- 스프레드 연산자