삼육대학교 SW융합교육원 김진호

자바스크립트 기초교육

<class>
</body>
<head>
<title>

06. distructuring assignment
(구조 분해 할당)

6-1. array distructring assignment basic syntax

6-2. object distructring assignment

6. distructuring assignment

구조 분해 할당을 사용하면 배열이나 객체를 변수로 '분해'하여 연결할 수 있다.

6-1. array distructring assignment basic syntax (배열 구조 분해 할당)

6-1-1. array distructring assignment basic syntax (배열 구조 분해 할당 기본 문법)

```
// 이름과 성을 요소로 가진 배열
let nameArr = ["Gildong", "Hong"];

// 구조 분해 할당을 이용해 firstName엔 nameArr[0]을 lastName엔 nameArr[1]을 할당한다.
let [firstName, lastName] = nameArr;
// let firstName = nameArr[0];
// let lastName = nameArr[1];

console.log(firstName);
console.log(lastName);

// 반환 값이 배열인 spilt 메서드를 활용한 예제
let [firstName2, lastName2] = "Saimdang Shin".split(' ');

console.log(firstName2);
console.log(lastName2);

// 쉼표를 사용하여 필요하지 않은 배열 요소를 버릴 수 있다.
let [firstName3, , lastName3] = ['firstName', 'middleName', 'lastName'];

console.log(firstName3);
console.log(lastName3);
```

6. distructuring assignment

6-1. array distructring assignment basic syntax (배열 구조 분해 할당)

6-1-2. various usage (다양한 사용법)

- 할당 연산자 우측엔 모든 이터러블이 올 수 있고 할당 연산자 좌측엔 뭐든지 올 수 있다.
  객체 프로퍼티도 가능하다.

```
let user = {};
[user.firstName, user.lastName] = "Gwansoon Yu".split(' ');

console.log(user.firstName);
```

- Object.entries()와 구조 분해를 조합하면 객체의 키와 값을 순회해 변수로 분해 할당할 수 있다.

```
for (let [key, value] of Object.entries(user)) {
    console.log(`${key}:${value}`);
}
```

6. distructuring assignment

6-1. array distructring assignment basic syntax (배열 구조 분해 할당)

6-1-2. various usage (다양한 사용법)

- 변수 교환 용도로도 사용할 수 있다.

```
let student = "유관순";
let teacher = "홍길동";

[student, teacher] = [teacher, student];

console.log(`학생 : ${student}, 교사 : ${teacher}`);
```

- rest parameter ...로 나머지 요소를 한 번에 가져올 수도 있다.

```
let [sign1, sign2, ...rest] = ["양자리", "황소자리", "쌍둥이자리", "게자리", "사자자리"];

console.log(sign1);
console.log(sign2);
console.log(rest);
```

6. distructuring assignment

6-1. array distructring assignment basic syntax (배열 구조 분해 할당)

6-1-2. various usage (다양한 사용법)

- 기본 값을 설정하고 사용할 수도 있다.

```
let [firstName4 = "아무개", lastName4 = "김"] = ["길동"];

console.log(firstName4);     // 배열에서 받아온 값
console.log(lastName4);      // 기본값
```

6. distructuring assignment

6-2. object distructring assignment (객체 구조 분해 할당)

6-2-1. object distructring assignment basic syntax (객체 구조 분해 할당 기본 문법)

```
// 상품명과 색상, 가격을 프로퍼티로 가지는 객체
let pants = {
    productName: "배기팬츠",
    color: "검정색",
    price: 30000
};

// 구조 분해 할당을 이용해 productName에는 pants.productName을 color엔 pants.color,
price에는 pants.price를 할당한다.
let { productName, color, price } = pants;

console.log(productName);
console.log(color);
console.log(price);

// 각 변수의 서술 순서는 무관하며 { 객체 프로퍼티 : 목표 변수 } 형식으로도 작성할 수 있다.
let { color: co, price: pr, productName: pn } = pants;
console.log(pn);
console.log(co);
console.log(pr);
```

6. distructuring assignment

6-2. object distructring assignment (객체 구조 분해 할당)

6-2-2. various usage (다양한 사용법)

- 객체에 존재하지 않는 프로퍼티는 기본 값을 설정해서 사용할 수 있다.
- 또한 콜론과 할당을 동시에 사용할 수 있다.

```
let shirts = {
    productName: "베이직셔츠"
};

let { productName: productName2 = "어떤 상품", color: color2 = "어떤 색상",
price: price2 = 0 } = shirts;

console.log(`productName2 : ${productName2}`);
console.log(`color2 :  ${color2}`);
console.log(`price2 : ${price2}`);
```

6. distructuring assignment

6-2. object distructring assignment (객체 구조 분해 할당)

6-2-2. various usage (다양한 사용법)

- 프로퍼티가 많은 복잡한 객체에서 원하는 정보만 뽑아오는 것도 가능하다.

```
let pants = {
    productName: "배기팬츠",
    color: "검정색",
    price: 30000
};

let { productName: productName3 } = pants;
// pants에서 productName만 변수로 뽑아내기
console.log(`productName3 : ${productName3}`);
```

6. distructuring assignment

6-2. object distructring assignment (객체 구조 분해 할당)

6-2-2. various usage (다양한 사용법)

- rest parameter ...로 나머지 요소를 한 번에 가져올 수도 있다.
- 단, 구버전 브라우저에서는 동작하지 않는다. 바벨을 사용해 동작 시킬 수 있다.

```
let { productName: productName4, ...rest } = pants;

console.log(`productName4 : ${productName4}`);
console.log(`rest.color : ${rest.color}`);
console.log(`rest.price : ${rest.price}`);

// let 없이 사용하는 예시
let productName5, color5, price5;

// {productName : productName5, color : color5, price : price5} = pants;
// 코드 블럭으로 인식해 오류가 발생한다.
({ productName: productName5, color: color5, price: price5 } = pants);
// 소괄호로 감싸주면 오류가 발생하지 않는다.

console.log(`productName5 : ${productName5}`);
console.log(`color5 :  ${color5}`);
console.log(`price5 : ${price5}`);
```

6. distructuring assignment

6-2. object distructring assignment (객체 구조 분해 할당)

6-2-3. nested distructring (중첩 구조 분해)

```
let product = {
    size: {
        width: 10,
        height: 30
    },
    items: ["doll", "robot"]
};

// 중첩 구조 분해 시 코드를 여러 줄에 걸쳐 작성해 의도하는 바를 명확히 드러낸다.
let {
    size: {
        width,
        height
    },
    items: [item1, item2],
    producer = "홍길동" // 기본 값 사용
} = product;

console.log(width);
console.log(height);
console.log(item1);
console.log(item2);
console.log(producer);
```

6. distructuring assignment

6-2. object distructring assignment (객체 구조 분해 할당)

6-2-4. function parameters (함수 매개변수)

■ 함수의 매개변수가 많거나 매개변수 기본값이 필요한 경우 등에 활용된다.

```
// function displayProduct(producer = "아무개", width = 0, height = 0, items = [] ) {}
// 위와 같은 함수는 넘겨주는 인수의 순서가 고정되어 있어 순서가 바뀌면 문제가 생기고 기본 값 사용
시에도 undefined를 이용해서 인수를 넘겨줘야만 한다.
// ex) displayProduct("신사임당", undefined, undefined, ["Coffee", "Donut"]);

// 구조 분해 할당을 이용하면 문제점이 해결 된다.
function displayProduct({ producer = "아무개", width = 10, height = 20, items = [] }) {
    console.log(`${producer} ${width} ${height}`);
    console.log(items);
}

// 함수에 전달할 객체
let exampleProduct = {
    items: ["Coffee", "Donut"],
    producer: "신사임당"
};

// 순서도 무관하고 기본 값 활용 시에도 별도의 처리가 불필요하다.
displayProduct(exampleProduct);
```