Array destructuring

Here's a **detailed explanation of Array Destructuring** in JavaScript:

🔍 What is Array Destructuring?

Array destructuring allows you to unpack values from arrays and assign them to variables in a concise, readable way.

🔶 Basic Syntax:

```
const array = [value1, value2];
const [a, b] = array;
```

🧪 1. Basic Example

```
const colors = ['red', 'green', 'blue'];
const [first, second, third] = colors;

console.log(first);  // 'red'
console.log(second); // 'green'
console.log(third);  // 'blue'
```

✨ 2. Skipping Elements

You can skip elements by leaving empty commas.

```
const numbers = [1, 2, 3, 4];
const [first, , third] = numbers;

console.log(first); // 1
console.log(third); // 3
```

🏷️ 3. Default Values

Provide default values if the array doesn't have enough elements.

```
const arr = [10];
const [a = 1, b = 2] = arr;

console.log(a); // 10
console.log(b); // 2
```

🔁 4. Swapping Variables

Neat trick to swap values without a temp variable:

```
let a = 5;
let b = 10;

[a, b] = [b, a];

console.log(a); // 10
console.log(b); // 5
```

🔄 5. Nested Destructuring

Handle arrays inside arrays:

```
const nested = [1, [2, 3]];
const [one, [two, three]] = nested;

console.log(two); // 2
console.log(three); // 3
```

〓 6. Destructuring in Function Parameters

You can destructure directly in function parameters.

```
function printCoordinates([x, y]) {
  console.log(`X: ${x}, Y: ${y}`);
}

printCoordinates([10, 20]); // X: 10, Y: 20
```

📦 7. Using Rest Operator

Use the rest operator to collect remaining elements:

```
const [first, ...rest] = [1, 2, 3, 4];

console.log(first); // 1
console.log(rest);  // [2, 3, 4]
```

✅ Summary

Use Case	Example
Basic assignment	`[a, b] = [1, 2]`
Skipping values	`[a, , b] = [1, 2, 3]`
Default values	`[a = 1] = []`
Swapping variables	`[a, b] = [b, a]`
Nested arrays	`[a, [b]] = [1, [2]]`
Function parameters	`function([x, y]) {}`
Rest with destructuring	`[a, ...rest] = [1, 2, 3, 4]`

Let me know if you want a visual cheat sheet or code exercises to practice!