flattening array in js

In JavaScript, flattening an array means converting a nested array (an array containing other arrays) into a single-level array. The primary method for this is the `flat()` method, introduced in ES2019.

Using `Array.prototype.flat()`:

The `flat()` method creates a new array with all sub-array elements concatenated into it recursively up to the specified depth.

JavaScript

```
// Example 1: Flattening to a default depth (1)
const nestedArray1 = [1, 2, [3, 4]];
const flattenedArray1 = nestedArray1.flat();
console.log(flattenedArray1); // Output: [1, 2, 3, 4]

// Example 2: Flattening to a specific depth
const nestedArray2 = [1, 2, [3, 4, [5, 6]]];
const flattenedArray2 = nestedArray2.flat(2); // Flatten two levels deep
console.log(flattenedArray2); // Output: [1, 2, 3, 4, 5, 6]

// Example 3: Flattening all levels using Infinity
const deeplyNestedArray = [1, [2, [3, [4, 5]]], 6];
const flattenedDeeply = deeplyNestedArray.flat(Infinity);
console.log(flattenedDeeply); // Output: [1, 2, 3, 4, 5, 6]

// Example 4: `flat()` also removes empty slots (holes) in an array
const arrayWithHoles = [1, , 3];
const flattenedWithHoles = arrayWithHoles.flat();
console.log(flattenedWithHoles); // Output: [1, 3]
```

Explanation:

When called without an argument, `flat()` defaults to a depth of 1, flattening only the first level of nested arrays.
You can pass a number as an argument to `flat()` to specify the desired depth of flattening.
To flatten an array to any depth (i.e., remove all levels of nesting), you can pass `Infinity` as the argument to `flat()`.

While other methods like using `reduce()` with `concat()` or recursive functions can also flatten arrays, `Array.prototype.flat()` is the most straightforward and recommended approach for

modern JavaScript development due to its simplicity and direct purpose

---

Revision #2
Created 29 October 2025 02:43:38 by AI API
Updated 5 December 2025 09:30:44 by AI Channel