The Pragmatic Programmers

JavaScript Brain Teasers

Exercise Your Mind

Faraz K. Kelhini
edited by Margaret Eldridge

This extract shows the online version of this title, and may contain features (such as hyperlinks and colors) that are not available in the print version.

For more information, or to purchase a paperback or ebook copy, please visit https://www.pragprog.com.

Puzzle 4

Mortal Koncatenation

mortal_koncatenation/mortal_koncatenation.js

```
const femaleMKCharacters = [
  "Sonya Blade",
  "Sindel",
  "Cassie Cage",
  "Sheeva"
];

const maleMKCharacters = [
  "Scorpion",
  "Sub-Zero",
  "Raiden",
  "Johnny Cage"
];

const MKCharacters = femaleMKCharacters + maleMKCharacters;

console.log(MKCharacters);
```

Guess the Output

Try to guess what the output is before moving to the next page.

You might have anticipated that the output would be a merged array, but this code will actually output:

```
Sonya Blade,Sindel,Cassie Cage,SheevaScorpion,Sub-Zero,Raiden,Johnny Cage
```

Discussion

This JavaScript code defines two arrays: femaleMKCharacters and maleMKCharacters. The femaleMKCharacters array contains the names of female characters from the Mortal Kombat video game series, whereas the maleMKCharacters array contains the names of male characters from the same game.

The puzzle then attempts to combine two arrays into one big array using the + operator. But here's the problem: when you use the + operator with arrays in JavaScript, it doesn't merge them as you might expect. Instead, it treats them as strings and does string concatenation. So, the end result is that MKCharacters becomes a single string that smashes the two arrays together as if they were strings.

Fortunately, there are multiple ways to combine two arrays. Here are a few common approaches:

1. concat(): A quick way to combine two arrays is by using concat(). This method creates a new array that includes the elements from both arrays:

   mortal_koncatenation/mortal_koncatenation_ex1.js

   ```
   const array1 = [1, 2, 3];
   const array2 = [4, 5, 6];
   const combinedArray = array1.concat(array2);

   console.log(combinedArray);    // → [1, 2, 3, 4, 5, 6]
   ```

   You can also use concat() when you have a variable number of arrays to concatenate or if you need to concatenate arrays dynamically based on certain conditions.

2. Spread Operator: The spread operator (...) is a newer feature introduced in ECMAScript 2015 (ES6) that provides a concise way to combine arrays. It unpacks the elements of each array and creates a new array with all the elements:

   mortal_koncatenation/mortal_koncatenation_ex2.js

   ```
   const array1 = [1, 2, 3];
   const array2 = [4, 5, 6];
   const combinedArray = [...array1, ...array2];

   console.log(combinedArray);    // → [1, 2, 3, 4, 5, 6]
   ```

Compared to concat(), the spread operator is more readable, especially when you only need to combine a couple of arrays.

3. push() or unshift(): You can also use array manipulation methods like push() or unshift() to merge arrays:

mortal_koncatenation/mortal_koncatenation_ex3-1.js

```
const array1 = [1, 2, 3];
const array2 = [4, 5, 6];

// Combining arrays using push()
array1.push(...array2);
console.log(array1);    // → [1, 2, 3, 4, 5, 6]
```

mortal_koncatenation/mortal_koncatenation_ex3-2.js

```
const array1 = [1, 2, 3];
const array2 = [4, 5, 6];

// Combining arrays using unshift()
array1.unshift(...array2);
console.log(array1);    // → [4, 5, 6, 1, 2, 3]
```

The push() method appends the elements of the second array to the end of the first array, while unshift() adds them to the beginning.

4. splice(): If you want to insert the elements of an array into another array at a specific index, you can use the splice() method:

mortal_koncatenation/mortal_koncatenation_ex4.js

```
const array1 = [1, 2, 3];
const array2 = [4, 5, 6];

// Assuming you want to add array2 to array1 at index 2
const insertIndex = 2;

// Using splice() to insert the elements of array2 into array1
array1.splice(insertIndex, 0, ...array2);

console.log(array1);    // → [1, 2, 4, 5, 6, 3]
```

These are some of the common ways to combine two arrays in JavaScript. Keep in mind that concat() and the spread operator do not alter the existing arrays; instead, they return a new array. On the contrary, push(), unshift(), and splice() modify the contents of the array they are applied to. Choose the method that best suits your specific requirements and coding style.

Further Reading

A complete list of available array methods

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array#instance_methods