

Array.prototype.includes()

includes() 方法用来判断一个数组是否包含一个指定的值，如果是，酌情返回 `true` 或 `false`。

```
1  let a = [1, 2, 3];
2
3  a.includes(2);
4  // true
5
6  a.includes(4);
7  // false
```

语法

```
arr.includes(searchElement)
arr.includes(searchElement, fromIndex)
```

参数

searchElement

需要查找的元素值。

fromIndex 可选

从该索引处开始查找 `searchElement`。如果为负值，则按升序从 `array.length + fromIndex` 的索引开始搜索。默认为 0。

返回值

一个 `Boolean`。

示例

```
1  [1, 2, 3].includes(2);    // true
2  [1, 2, 3].includes(4);    // false
3  [1, 2, 3].includes(3, 3); // false
```

```
4 | [1, 2, 3].includes(3, -1); // true
5 | [1, 2, NaN].includes(NaN); // true
```

fromIndex 大于等于数组长度

如果 `fromIndex` 大于等于数组长度，则返回 `false`。该数组不会被搜索。

```
1 | var arr = ['a', 'b', 'c'];
2 |
3 | arr.includes('c', 3); //false
4 | arr.includes('c', 100); // false
```

计算出的索引小于 0

如果 `fromIndex` 为负值，计算出的索引将作为开始搜索 `searchElement` 的位置。如果计算出的索引小于 0，则整个数组都会被搜索。

```
1 | // 数组长度是3
2 | // fromIndex 是 -100
3 | // computed index 是 3 + (-100) = -97
4 |
5 | var arr = ['a', 'b', 'c'];
6 |
7 | arr.includes('a', -100); // true
8 | arr.includes('b', -100); // true
9 | arr.includes('c', -100); // true
```

includes() 作为一个通用方法

`includes()` 方法有意设计为通用方法。它不要求 `this` 值是数组对象，所以它可以被用于其他类型的对象（比如类数组对象）。下面的例子展示了在函数的 `arguments` 对象上调用的 `includes()` 方法。

```
1 | (function() {
2 |     console.log([].includes.call(arguments, 'a')); // true
3 |     console.log([].includes.call(arguments, 'd')); // false
4 | })( 'a', 'b', 'c' );
```

Polyfill

```
1 // https://tc39.github.io/ecma262/#sec-array.prototype.includes
2 if (!Array.prototype.includes) {
3   Object.defineProperty(Array.prototype, 'includes', {
4     value: function(searchElement, fromIndex) {
5
6       // 1. Let O be ? ToObject(this value).
7       if (this == null) {
8         throw new TypeError('"this" is null or not defined');
9       }
10
11      var o = Object(this);
12
13      // 2. Let len be ? ToLength(? Get(0, "length")).
14      var len = o.length >>> 0;
15
16      // 3. If len is 0, return false.
17      if (len === 0) {
18        return false;
19      }
20
21      // 4. Let n be ? ToInteger(fromIndex).
22      //    (If fromIndex is undefined, this step produces the value 0.)
23      var n = fromIndex | 0;
24
25      // 5. If n ≥ 0, then
26      //   a. Let k be n.
27      // 6. Else n < 0,
28      //   a. Let k be len + n.
29      //   b. If k < 0, let k be 0.
30      var k = Math.max(n >= 0 ? n : len - Math.abs(n), 0);
31
32      // 7. Repeat, while k < len
33      while (k < len) {
34        // a. Let elementK be the result of ? Get(0, ! ToString(k)).
35        // b. If SameValueZero(searchElement, elementK) is true, return true.
36        // c. Increase k by 1.
37        // NOTE: === provides the correct "SameValueZero" comparison needed here.
38        if (o[k] === searchElement) {
39          return true;
40        }
41        k++;
42      }
43    }
44  });
45 }
```

```
43 |
44 |         // 8. Return false
45 |         return false;
46 |     }
47 | });
48 | }
```

如果你需要支持那些不支持 `Object.defineProperty` 的废弃JavaScript 引擎，你最好不要 `polyfill` `Array.prototype` 方法，因为你不能使它们不可枚举。



规范

| Specification | Status | Comment |
|--------------------------------------------------------------------------------|-----------------------------------------|---------------------|
| ECMAScript 2016 (ECMA-262) Array.prototype.includes | <div><div>ST</div>Standard</div> | Initial definition. |
| ECMAScript Latest Draft (ECMA-262) Array.prototype.includes | <div><div>LS</div>Living Standard</div> | |

浏览器兼容性

| | Desktop | Mobile | | | | |
|---------------|---------|-----------------|-------------------|------|-------|--------|
| Feature | Chrome | Firefox (Gecko) | Internet Explorer | Edge | Opera | Safari |
| Basic support | 47 | 43 (43) | 未实现 | 14 | 34 | 9 |

相关链接

- `TypedArray.prototype.includes()` 
- `String.prototype.includes()` 
- `Array.prototype.indexOf()`

