

NodeList []

HTMLCollection []

[]

- [NodeList \[\]](#)
 - [\[\]](#)
 - [NodeList.prototype.length](#)
 - [NodeList.prototype.forEach\(\)](#)
 - [NodeList.prototype.item\(\)](#)
 - [NodeList.prototype.keys\(\)](#) [NodeList.prototype.values\(\)](#) [NodeList.prototype.entries\(\)](#)
- [HTMLCollection \[\]](#)
 - [\[\]](#)
 - [HTMLCollection.prototype.length](#)
 - [HTMLCollection.prototype.item\(\)](#)
 - [HTMLCollection.prototype.namedItem\(\)](#)



NodeList []

[]

NodeList []

NodeList []

- `Node.childNodes`
- `document.querySelectorAll()`

```
document.body.childNodes instanceof NodeList // true
```

NodeList  length  forEach  pop  push 


```
var children = document.body.childNodes;
```

```
Array.isArray(children) // false
```

```
children.length // 34
```

```
children.forEach(console.log)
```

 NodeList  children  length  forEach 

 NodeList 

```
var children = document.body.childNodes;
var nodeArr = Array.prototype.slice.call(children);
```

 forEach  NodeList  for 

```
var children = document.body.childNodes;
```

```
for (var i = 0; i < children.length; i++) {
  var item = children[i];
}
```

 NodeList  DOM
 NodeList  Node.childNodes
 NodeList 

```
var children = document.body.childNodes;
children.length // 18
document.body.appendChild(document.createElement('p'));
children.length // 19
```

 NodeList  children  length  1

NodeList.prototype.length

length  NodeList 

```
document.querySelectorAll('xxx').length
// 0
```

document.querySelectorAll NodeList HTML length
0

NodeList.prototype.forEach()

forEach NodeList
forEach

```
var children = document.body.childNodes;
children.forEach(function f(item, i, list) {
  // ...
}, this);
```

f NodeList forEach
this

NodeList.prototype.item()

item

```
document.body.childNodes.item(0)
```

item(0)
item null item
item

```
document.body.childNodes[0]
```

NodeList.prototype.keys()

NodeList.prototype.values()

NodeList.prototype.entries()

Diagram illustrating ES6 methods for iterating over objects:

- `keys()` returns an array of keys.
- `values()` returns an array of values.
- `entries()` returns an array of entries (key-value pairs).
- `for...of` loop is used to iterate over the values of an object.

```
var children = document.body.childNodes;
```

```
for (var key of children.keys()) {
  console.log(key);
}
```

```
// 0
// 1
// 2
// ...
```

```
for (var value of children.values()) {
    console.log(value);
}
```

```
// #text
// <script>
// ...
```

```
for (var entry of children.entries()) {
  console.log(entry);
}
```

```
// Array [ 0, #text ]
// Array [ 1, <script> ]
// ...
```

HTMLCollection

--	--

HTMLCollection [16 elements] element

HTMLCollection [2] forEach [5 elements] for [4 elements]

NodeList [4 elements]

element

NodeList

HTMLCollection [] Document [] document.links [] document.forms [] document.images []

```
document.links instanceof HTMLCollection // true
```

HTMLCollection []

[] id [] name [] HTMLCollection [] id [] name [] null []

```
// HTML [ ]
// 

var pic = document.getElementById('pic');
document.images.pic === pic // true
```

[] document.images [] HTMLCollection [] [] id [] HTMLCollection []

HTMLCollection.prototype.length

length [] HTMLCollection []

```
document.links.length // 18
```

HTMLCollection.prototype.item()

item []

```
var c = document.images;
var img0 = c.item(0);
```

[] item(0) [] 0 []

[] 0 [] item [] null []

HTMLCollection.prototype.namedItem()

namedItem []

id []

name

[]

null []

```
// HTML [ ]
// 

var pic = document.getElementById('pic');
document.images.namedItem('pic') === pic // true
```

Collection.namedItem('value') []

Collection['value'] []

[] #2

[] [] 2018-03-01 14:13:39 UTC

[] [] 2021-10-29 08:05:50 UTC