

Node

□□

- [□□](#)
 - [Node.prototype.nodeType](#)
 - [Node.prototype.nodeName](#)
 - [Node.prototype.nodeValue](#)
 - [Node.prototype.textContent](#)
 - [Node.prototype.baseURI](#)
 - [Node.prototype.ownerDocument](#)
 - [Node.prototype.nextSibling](#)
 - [Node.prototype.previousSibling](#)
 - [Node.prototype.parentNode](#)
 - [Node.prototype.parentElement](#)
 - [Node.prototype.firstChild](#) □ [Node.prototype.lastChild](#)
 - [Node.prototype.childNodes](#)
 - [Node.prototype.isConnected](#)
- [□□](#)
 - [Node.prototype.appendChild\(\)](#)
 - [Node.prototype.hasChildNodes\(\)](#)
 - [Node.prototype.cloneNode\(\)](#)
 - [Node.prototype.insertBefore\(\)](#)
 - [Node.prototype.removeChild\(\)](#)
 - [Node.prototype.replaceChild\(\)](#)
 - [Node.prototype.contains\(\)](#)
 - [Node.prototype.compareDocumentPosition\(\)](#)
 - [Node.prototype.isEqualNode\(\)](#) □ [Node.prototype.isSameNode\(\)](#)
 - [Node.prototype.normalize\(\)](#)
 - [Node.prototype.getRootNode\(\)](#)



Node.prototype.nodeType

nodeType

```
document.nodeType // 9
```

9

Node

```
document.nodeType === Node.DOCUMENT_NODE // true
```

nodeType Node.DOCUMENT_NODE

nodeType

- document 9 Node.DOCUMENT_NODE
- element 1 Node.ELEMENT_NODE
- attr 2 Node.ATTRIBUTE_NODE
- text 3 Node.TEXT_NODE
- DocumentFragment 11 Node.DOCUMENT_FRAGMENT_NODE
- DocumentType 10 Node.DOCUMENT_TYPE_NODE
- Comment 8 Node.COMMENT_NODE

nodeType

```
var node = document.documentElement.firstChild;
if (node.nodeType === Node.ELEMENT_NODE) {
  console.log('');
}
```

Node.prototype.nodeName

nodeName

```
// HTML 
// <div id="d1">hello world</div>
var div = document.getElementById('d1');
```

```
div.nodeName // "DIV"
```

```


nodeName
  DIV


```

```

  nodeName

```

- document #document
- element
- attr
- text #text
- DocumentFragment #document-fragment
- DocumentType
- Comment #comment

Node.prototype.nodeValue

```
nodeValue
```

```

  text
  comment
  attr
  nodeValue
  null
  nodeValue

```

```
// HTML
// <div id="d1">hello world</div>
var div = document.getElementById('d1');
div.nodeValue // null
div.firstChild.nodeValue // "hello world"
```

```

div
  nodeValue
  null
  div.firstChild

```

Node.prototype.textContent

```
textContent
```

```
// HTML
// <div id="divA">This is <span>some</span> text</div>

document.getElementById('divA').textContent
// This is some text
```

```

textContent
HTML

```

 HTML 

document.doctype.textContent null
document.documentElement.textContent

□□□□□ `baseURI` □□□□ `<base>` □□□□□

```
document [ ] ownerDocument [ ] null [ ]
```

Node.prototype.nextSibling

Node.nextSibling 

null 

```
// HTML   
// <div id="d1">hello</div><div id="d2">world</div>  
var d1 = document.getElementById('d1');  
var d2 = document.getElementById('d2');  
  
d1.nextSibling === d2 // true
```



d1.nextSibling 

d1 

d2 



<!-- comment -->



nextSibling 

```
var el = document.getElementById('div1').firstChild;  
  
while (el !== null) {  
  console.log(el.nodeName);  
  el = el.nextSibling;  
}
```



div1 

Node.prototype.previousSibling

previousSibling



null 

```
// HTML   
// <div id="d1">hello</div><div id="d2">world</div>  
var d1 = document.getElementById('d1');  
var d2 = document.getElementById('d2');  
  
d2.previousSibling === d1 // true
```



d2.previousSibling 

d2 

d1 


```
p1.firstChild.nodeName // "SPAN"
```

DOM Tree

p [] span []

firstChild []

```
// HTML DOM
// <p id="p1">
//   <span>First span</span>
// </p>
var p1 = document.getElementById('p1');
p1.firstChild.nodeName // "#text"
```

DOM Tree

p [] span [] firstChild []

lastChild [] null []

firstChild []

Node.prototype.childNodes

childNodes [] NodeList []

```
var children = document.querySelector('ul').childNodes;
```

DOM Tree

children [] ul []

childNodes []

```
var div = document.getElementById('div1');
var children = div.childNodes;

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

DOM Tree

document [] docType [] HTML []

```
var children = document.childNodes;
for (var i = 0; i < children.length; i++) {
  console.log(children[i].nodeType);
}

// 10
```

[illegible]

```
document.body.appendChild(div);
```

--	--	--	--

myDiv

```
document.body
```

11

DocumentFragment

DocumentFragment

[illegible]

DocumentFragment

DocumentFragment

Node.prototype.hasChildNodes()

hasChildNodes

```
var foo = document.getElementById('foo');
```

```
if (foo.hasChildNodes()) {
    foo.removeChild(foo.childNodes[0]);
}
```

--	--	--	--	--	--	--	--	--

foo

[illegible]

```
hasChildNodes
```

--	--	--	--	--	--

true

- `node.hasChildNodes()`
- `node.firstChild !== null`
- `node.childNodes && node.childNodes.length > 0`

hasChildNodes

firstChild

```
nextSibling
```

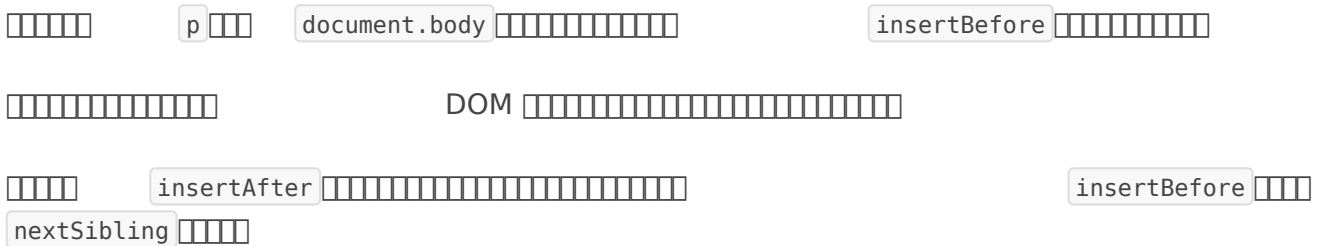
```
function DOMComb(parent, callback) {
    if (parent.hasChildNodes()) {
        for (var node = parent.firstChild; node; node = node.nextSibling) {
            DOMComb(node, callback);
        }
    }
    callback(parent);
}
```

// ☐

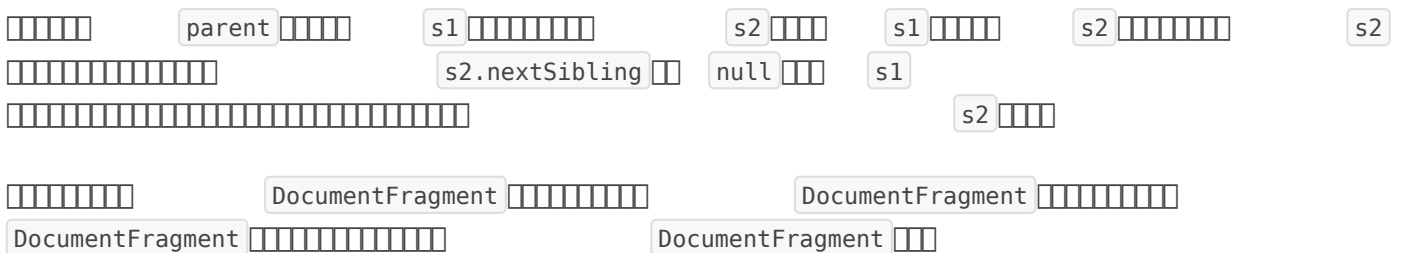
```
DOMComb(document.body, console.log)
```



```
var p = document.createElement('p');
document.body.insertBefore(p, null);
```



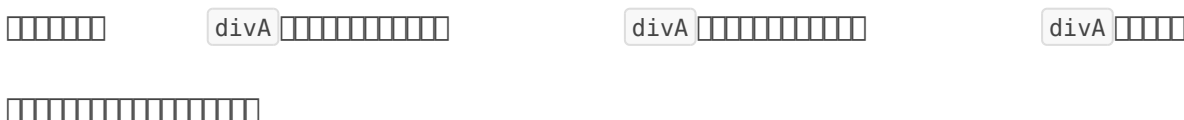
```
parent.insertBefore(s1, s2.nextSibling);
```



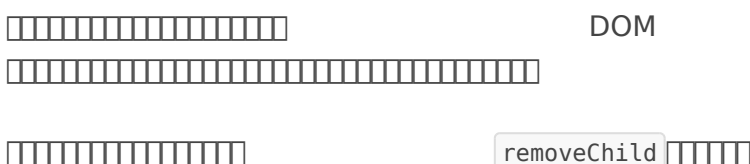
Node.prototype.removeChild()

```
removeChild
```

```
var divA = document.getElementById('A');
divA.parentNode.removeChild(divA);
```



```
var element = document.getElementById('top');
while (element.firstChild) {
  element.removeChild(element.firstChild);
}
```



Node.prototype.replaceChild()

replaceChild []

```
var replacedNode = parentNode.replaceChild(newChild, oldChild);
```

[] replaceChild [] newChild []
oldChild [] oldChild []

```
var divA = document.getElementById('divA');  
var newSpan = document.createElement('span');  
newSpan.textContent = 'Hello World!';  
divA.parentNode.replaceChild(newSpan, divA);
```

[] divA []

Node.prototype.contains()

contains []

- []
- []
- []

```
document.body.contains(node)
```

[] node []

[] contains [] true []

```
nodeA.contains(nodeA) // true
```

Node.prototype.compareDocumentPosition()

compareDocumentPosition [] contains

[]

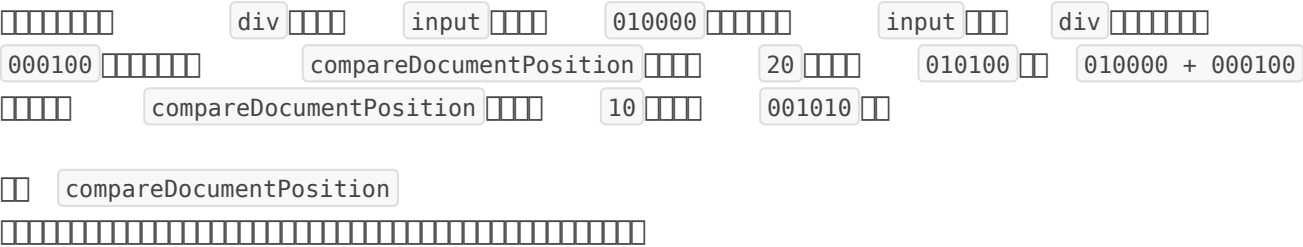
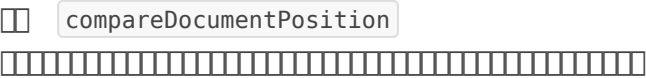
[]	[]	[]
000000	0	[]
000001	1	[] []

0000	0000	00
000010	2	00000000
000100	4	00000000
001000	8	00000000
010000	16	00000000
100000	32	000000

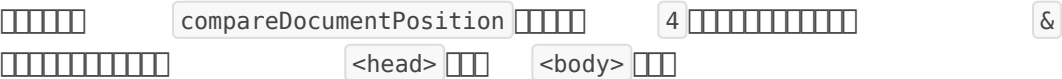
```
// HTML
// <div id="mydiv">
//   <form><input id="test" /></form>
// </div>

var div = document.getElementById('mydiv');
var input = document.getElementById('test');

div.compareDocumentPosition(input) // 20
input.compareDocumentPosition(div) // 10
```



```
var head = document.head;
var body = document.body;
if (head.compareDocumentPosition(body) & 4) {
  console.log('');
} else {
  console.log('<body> <head> ');
}
```



Node.prototype.isEqualNode()
Node.prototype.isSameNode()

isEqualNode



```
var p1 = document.createElement('p');
var p2 = document.createElement('p');

p1.isEqualNode(p2) // true
```

isSameNode



```
var p1 = document.createElement('p');
var p2 = document.createElement('p');

p1.isSameNode(p2) // false
p1.isSameNode(p1) // true
```

Node.prototype.normalize()

normalize



text



```
var wrapper = document.createElement('div');

wrapper.appendChild(document.createTextNode('Part 1 '));
wrapper.appendChild(document.createTextNode('Part 2 '));

wrapper.childNodes.length // 2
wrapper.normalize();
wrapper.childNodes.length // 1
```



normalize



wrapper



normalize



Text.splitText



Text



Node.prototype.getRootNode()

getRootNode()



document



ownerDocument



```
document.body.firstChild.getRootNode() === document
// true
document.body.firstChild.getRootNode() === document.body.firstChild.ownerDocument
// true
```

□□□□□

document □□□□□□□□

document.ownerDocument □□

```
document.getRootNode() // document
document.ownerDocument // null
```

□□ #2

□ □□ □□ 2018-02-27 13:26:46 UTC

□ □□ □□ 2019-03-21 11:29:48 UTC