

Element



-  [document](#)
-  [document](#)
 -  [document](#)
 -  [document](#)
 - [Element.attributes](#)
 - [Element.className](#) [Element.classList](#)
 - [Element.dataset](#)
 - [Element.innerHTML](#)
 - [Element.outerHTML](#)
 - [Element.clientHeight](#) [Element.clientWidth](#)
 - [Element.clientLeft](#) [Element.clientTop](#)
 - [Element.scrollHeight](#) [Element.scrollWidth](#)
 - [Element.scrollLeft](#) [Element.scrollTop](#)
 - [Element.offsetParent](#)
 - [Element.offsetHeight](#) [Element.offsetWidth](#)
 - [Element.offsetLeft](#) [Element.offsetTop](#)
 - [Element.style](#)
 - [Element.children](#) [Element.childElementCount](#)
 - [Element.firstElementChild](#) [Element.lastElementChild](#)
 - [Element.nextElementSibling](#) [Element.previousElementSibling](#)
-  [document](#)
 -  [document](#)
 - [Element.querySelector\(\)](#)
 - [Element.querySelectorAll\(\)](#)
 - [Element.getElementsByClassName\(\)](#)
 - [Element.getElementsByTagName\(\)](#)
 - [Element.closest\(\)](#)
 - [Element.matches\(\)](#)

- [`Element.scrollToView\(\)`](#)
- [`Element.getBoundingClientRect\(\)`](#)
- [`Element.getClientRects\(\)`](#)
- [`Element.insertAdjacentElement\(\)`](#)
- [`Element.insertAdjacentHTML\(\)`](#) [`Element.insertAdjacentText\(\)`](#)
- [`Element.remove\(\)`](#)
- [`Element.focus\(\)`](#) [`Element.blur\(\)`](#)
- [`Element.click\(\)`](#)



nodeType 1

Element  Node  Node  Element 



1

Element.id

Element.id

id

```
// HTML <p id="foo">
var p = document.querySelector('p');
p.id // "foo"
```

id

<p id="foo">

<p id="F00">

id

2 Element.tagName

Element.tagName

nodeName

```
// HTML
// <span id="myspan">Hello</span>
var span = document.getElementById('myspan');
span.id // "myspan"
span.tagName // "SPAN"
```

3 Element.dir

Element.dir

"ltr"

"rtl"

4 Element.accessKey

Element.accessKey

```
// HTML
// <button accesskey="h" id="btn"></button>
var btn = document.getElementById('btn');
btn.accessKey // "h"
```

btn

h

Alt + h

5 Element.draggable

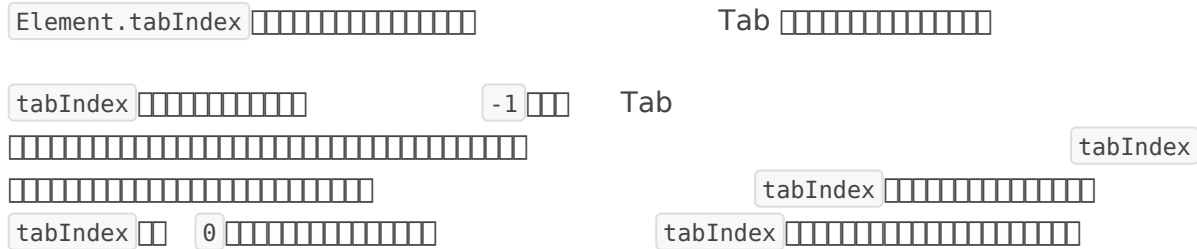
Element.draggable

6 Element.lang

Element.lang

```
// HTML 文档
// <html lang="en">
document.documentElement.lang // "en"
```

7 Element.tabIndex



8 Element.title

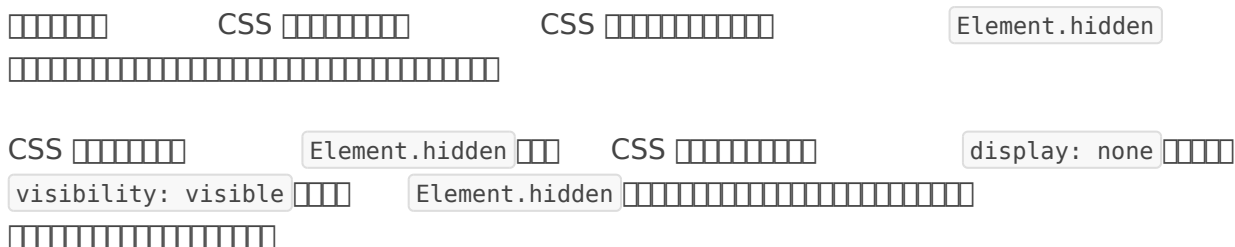


1 Element.hidden

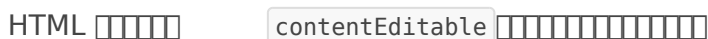


```
var btn = document.getElementById('btn');
var mydiv = document.getElementById('mydiv');

btn.addEventListener('click', function () {
  mydiv.hidden = !mydiv.hidden;
}, false);
```



2 Element.contentEditable Element.isContentEditable



```
<div contenteditable>123</div>
```

contenteditable

- "true" 
- "false" 
- "inherit" 

contenteditable	
-----------------	--

Element.attributes

[illegible]

```
var p = document.querySelector('p');
var attrs = p.attributes;

for (var i = attrs.length - 1; i >= 0; i--) {
    console.log(attrs[i].name + '->' + attrs[i].value);
}
```

p

Element.className vs Element.classList

class

`class`

```
// HTML <div class="one two three" id="myDiv"></div>
var div = document.getElementById('myDiv');

div.className
// "one two three"

div.classList
// {
```

```
// 0: "one"
// 1: "two"
// 2: "three"
// length: 3
// }
```

className classList
 length class

classList

- add() class
- remove() class
- contains() class
- toggle() class
- item() class
- toString() class

```
var div = document.getElementById('myDiv');

div.classList.add('myCssClass');
div.classList.add('foo', 'bar');
div.classList.remove('myCssClass');
div.classList.toggle('myCssClass'); // 是否 myCssClass 存在
div.classList.contains('myCssClass'); // 是否 true 或 false
div.classList.item(0); // 获取 Class
div.classList.toString();
```

className classList class

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

// 添加class
foo.className += 'bold';
foo.classList.add('bold');

// 移除class
foo.classList.remove('bold');
foo.className = foo.className.replace(/^bold$/, '');
```

toggle true false

```
el.classList.toggle('abc', boolValue);
```

```
// 空白
```

```
if (boolValue) {  
  el.classList.add('abc');  
} else {  
  el.classList.remove('abc');  
}
```

Element.dataset

属性名

data-属性名

```
<div data-timestamp="1522907809292"></div>
```

属性名

<div>属性名

data-timestamp属性名

Element.dataset属性名

data-属性名

```
// <article  
//   id="foo"  
//   data-columns="3"  
//   data-index-number="12314"  
//   data-parent="cars">  
//   ...  
// </article>  
  
var article = document.getElementById('foo');  
article.dataset.columns // "3"  
article.dataset.indexNumber // "12314"  
article.dataset.parent // "cars"
```

属性名 dataset属性名

HTML属性名

data-属性名

-属性名

.属性名

:属性名

属性名

_属性名

JavaScript属性名

dataset属性名

- 属性名 data-属性名
- 属性名
- 属性名

属性名 data-abc-def属性名 dataset.abcDef属性名 data-abc-1属性名 dataset["abc-1"]属性名

dataset data- Element.getAttribute() Element.setAttribute()

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

mydiv.dataset.foo = 'bar';
mydiv.getAttribute('data-foo') // "bar"
```

Element.innerHTML

Element.innerHTML HTML

innerHTML

```
el.innerHTML = '';
```

el el

& < > innerHTML

& < > element.textContent

```
// HTML <p id="para"> 5 > 3 </p>
document.getElementById('para').innerHTML
// 5 &gt; 3
```

HTML DOM

<script> script

```
var name = "<script>alert('haha')</script>";
el.innerHTML = name;
```

innerHTML

```
var name = "<img src=x onerror=alert(1)>";
el.innerHTML = name;
```

alert textContent

innerHTML

Element.outerHTML

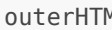
Element.outerHTML 


HTML

```
// HTML   
// <div id="d"><p>Hello</p></div>  
var d = document.getElementById('d');  
d.outerHTML  
// '<div id="d"><p>Hello</p></div>'
```

outerHTML 

```
// HTML   
// <div id="container"><div id="d">Hello</div></div>  
var container = document.getElementById('container');  
var d = document.getElementById('d');  
container.firstChild.nodeName // "DIV"  
d.nodeName // "DIV"  
  
d.outerHTML = '<p>Hello</p>';  
container.firstChild.nodeName // "P"  
d.nodeName // "DIV"
```

```
var div = document.createElement('div');  
div.outerHTML = '<p>test</p>';  
// DOMException: This element has no parent node.
```

Element.clientHeight

Element.clientWidth

document.body.scrollHeight

scrollHeight

```
// HTML  
// <div id="myDiv" style="height: 200px; overflow: hidden;">...</div>  
document.getElementById('myDiv').scrollHeight // 200
```

myDiv

CSS

200

scrollHeight

Element.scrollLeft Element.scrollTop

Element.scrollLeft

Element.scrollTop

0

document.documentElement

document.documentElement.scrollLeft

document.documentElement.scrollTop

Element.offsetParent

Element.offsetParent

CSS position

static

```
<div style="position: absolute;">  
  <p>  
    <span>Hello</span>  
  </p>  
</div>
```

span

offsetParent

div

Element.offsetTop Element.offsetLeft

offsetParent

display

none

position

fixed

offsetParent

null

```
<div style="position: absolute;">
  <p>
    <span style="display: none;">Hello</span>
  </p>
</div>
```

□□□□□

span □□□

offsetParent □□□

null □

□□□□□□□□□□

position □□□□

static □□

Element.offsetParent □□□□

<body> □□□

Element.offsetHeight□

Element.offsetWidth

Element.offsetHeight □□□□□□□□□□

CSS □□□□□□□□□□□□□□□□

padding □ border□□□□□□□□□□□□□□□□

Element.offsetWidth □□□□□□

CSS □□□□□□□□□□

Element.offsetHeight □□□

□□□□□□□□□□

Element.clientHeight □

Element.clientWidth

□□□□□□□□□□

CSS □□□□□□□

display: none; □□□□

0 □

Element.offsetLeft□ Element.offsetTop

Element.offsetLeft □□□□□□□□

Element.offsetParent □□□□□□

Element.offsetTop

□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□

```
function getElementPosition(e) {
  var x = 0;
  var y = 0;
  while (e !== null) {
    x += e.offsetLeft;
    y += e.offsetTop;
    e = e.offsetParent;
  }
  return {x: x, y: y};
}
```

Element.style

Element.style HTMLCollection CSS

Element.children

Element.childElementCount

Element.children HTMLCollection
Node.childNodes

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

Element para

Node.childNodes

Element.childElementCount Element.children.length

Element.firstElementChild

Element.lastElementChild

Element.firstElementChild Element.lastElementChild

Element null

Element.nextElementSibling

Element.previousElementSibling

Element.nextElementSibling null

```
// HTML 文档
// <div id="div-01">Here is div-01</div>
// <div id="div-02">Here is div-02</div>
var el = document.getElementById('div-01');
el.nextElementSibling
// <div id="div-02">Here is div-02</div>
```

Element.previousElementSibling

null



- `getAttribute()`
- `getAttributeNames()`
- `setAttribute()`
- `hasAttribute()`
- `hasAttributes()`
- `removeAttribute()`



Element.querySelector()

Element.querySelector CSS



null

```
var content = document.getElementById('content');
var el = content.querySelector('p');
```



content

p

Element.querySelector

CSS

```
document.body.querySelector("style[type='text/css'], style:not([type])");
```



```
element.querySelector('div, p')
```

The diagram illustrates the relationship between CSS and HTML. A box labeled `querySelector` is connected to a long horizontal bar representing the DOM. The bar is divided into two sections: the left section is labeled **CSS** and the right section is labeled **HTML**.

```
var outer = document.getElementById('outer');
outer.querySelector('div p')
// <p>Hello</p>
```

```
var el = document.querySelector('#test');  
var matches = el.querySelectorAll('div.highlighted > p');
```

Element.getElementsByClassName()

Element.getElementsByClassName() HTMLCollection
class document.getElementsByClassName()

```
element.getElementsByClassName('red test');
```

HTMLCollection document

```
// HTML  
// <div id="example">  
//   <p class="foo"></p>  
//   <p class="foo"></p>  
// </div>  
var element = document.getElementById('example');  
var matches = element.getElementsByClassName('foo');  
  
for (var i = 0; i < matches.length; i++) {  
  matches[i].classList.remove('foo');  
  matches.item(i).classList.add('bar');  
}  
// HTML  
// <div id="example">  
//   <p></p>  
//   <p class="foo bar"></p>  
// </div>
```

matches class foo matches

Element.getElementsByTagName()

Element.getElementsByTagName() HTMLCollection
document.getElementsByTagName()

```
var table = document.getElementById('forecast-table');  
var cells = table.getElementsByTagName('td');
```

Element.closest()

Element.closest XXXX CSS
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXX CSS XXXXXXX null

```
// HTML XXXX
// <article>
//   <div id="div-01">Here is div-01
//     <div id="div-02">Here is div-02
//       <div id="div-03">Here is div-03</div>
//     </div>
//   </div>
// </article>

var div03 = document.getElementById('div-03');

// div-03 XXXXXXX
div03.closest("#div-02") // div-02
div03.closest("div div") // div-03
div03.closest("article > div") //div-01
div03.closest(":not(div)") // article
```

XXXXXXXX closest XXXXXXXXXXXXXXXXXXXX closest XXX div-03

Element.matches()

Element.matches XXXXXXXXXXXXXXXXXXXXXXX CSS XXX

```
if (el.matches('.someClass')) {
  console.log('Match!');
}
```

XXXXXX

XXXXX Element XXXXXXXXXXXXXXXX EventTarget XXXXXXXX

- ```
element.addEventListener('click', listener, false);
element.removeEventListener('click', listener, false);

var event = new Event('click');
element.dispatchEvent(event);
```

The diagram illustrates the relationship between `Element.scrollIntoView` and `window.location.hash`. On the left, a box labeled `Element.scrollIntoView` is connected to a long horizontal bar divided into 16 segments. Below this bar, there are four small squares. On the right, a box labeled `window.location.hash` is connected to the same long horizontal bar. This visualizes how `scrollIntoView` interacts with the hash value in the URL.

The diagram consists of three horizontal bars of increasing length. The first bar is short and labeled 'true'. The second bar is medium-length and labeled 'false'. The third bar is long and labeled 'true'.

Element.getBoundingClientRect   
CSS 

```
getBoundingClientRect
```

- x
  - y
  - height
  - width
  - left
  - right
  - top
  - bottom
- x

x + width

y

y + height

viewport  
left window.scrollX top window.scrollY  
getBoundingClientRect border width height  
padding border  
Object.keys

```
var rect = document.body.getBoundingClientRect();
Object.keys(rect) // []
```

rect Object.keys

## Element.getClientRects()

Element.getClientRects Rect  
bottom height left right top width  
<div> <p>  
<a> <em>  
Element.getBoundingClientRect()

```
Hello World Hello World Hello World
```

<span> getClientRects  
getClientRects

```
var el = document.getElementById('inline');
el.getClientRects().length // 3
el.getClientRects()[0].left // 8
el.getClientRects()[0].right // 113.908203125
el.getClientRects()[0].bottom // 31.200000762939453
el.getClientRects()[0].height // 23.200000762939453
el.getClientRects()[0].width // 105.908203125
```

```

 Hello World
 Hello World
 Hello World

```

## Element.insertAdjacentElement()

Diagram illustrating an array structure. The array has 25 slots. The first slot is highlighted in light blue and contains the text "Element.insertAdjacentElement". The last slot is highlighted in light blue and contains the text "null". All other slots are empty.

```
element.insertAdjacentElement(position, element);
```

Diagram illustrating the insertion of a new element into a linked list:

- A box labeled `Element.insertAdjacentElement` is shown.
- A horizontal sequence of 20 small squares represents the existing linked list.
- A new element, represented by a small square, is being inserted at the end of the sequence.

- beforebegin [ ]
- afterbegin [ ]
- beforeend [ ]
- afterend [ ]

```
var p1 = document.createElement('p')
var p2 = document.createElement('p')
p1.insertAdjacentElement('afterend', p2) // null
```

[illegible]

## Element.insertAdjacentHTML()

## Element.insertAdjacentText()

Element.insertAdjacentHTML

HTML

DOM

```
element.insertAdjacentHTML(position, text);
```

HTML

- beforebegin
- afterbegin
- beforeend
- afterend

```
// HTML <div id="one">one</div>
var d1 = document.getElementById('one');
d1.insertAdjacentHTML('afterend', '<div id="two">two</div>');
// HTML <div id="one">one</div><div id="two">two</div>
```

DOM

innerHTML

HTML

Element.insertAdjacentText

Element.insertAdjacentHTML

```
// HTML <div id="one">one</div>
var d1 = document.getElementById('one');
d1.insertAdjacentText('afterend', 'two');
// HTML <div id="one">one</div>two
```

## Element.remove()

Element.remove

ChildNode

```
var el = document.getElementById('mydiv');
el.remove();
```

el

DOM

# Element.focus() Element.blur()

Element.focus()

```
document.getElementById('my-span').focus();
```

preventScroll

```
function getFocus() {
 document.getElementById('btn').focus({preventScroll:false});
}
```

btn

document.activeElement

Element.blur()

# Element.click()

Element.click()  click



- Craig Buckler [How to Translate from DOM to SVG Coordinates and Back Again](#)