

<a> 元素

1. 属性

<a> 元素有一系列 URL 相关属性，可以用来操作链接地址。

```
// <a id="test" href="http://user:passwd@example.com:8081/index.html?bar=1#foo">test</a>
let a = document.getElementById('test');
a.hash      // 片段识别符 (以#开头) : "#foo"
a.host      // 主机和端口 (默认端口 `80` 和 `443` 会省略) : "example.com:8081"
a.hostname  // 主机名: "example.com"
a.href      // 完整的 URL
a.origin    // 协议、域名和端口: "http://example.com:8081"
a.password  // 主机名前的密码: "passwd"
a.pathname  // 路径 (以/开头) : "/index.html"
a.port      // 端口: "8081"
a.protocol  // 协议 (包含尾部的冒号): "http:"
a.search    // 查询字符串 (以 `?` 开头) : "?bar=1"
a.username  // 主机名前的用户名: "user"
```

除了 `origin` 属性是只读的，上面这些属性都是可读写的。

1.1. accessKey

`accessKey` 属性用来读写 <a> 元素的快捷键。

```
// <a id="test" href="http://example.com">test</a>
let a = document.getElementById('test');
a.accessKey = 'k';
```

上例设置 <a> 元素的快捷键为 `k`，以后只要按下这个快捷键，浏览器就会跳转到 `example.com`。

不同的浏览器在不同的操作系统下，唤起快捷键的功能键组合是不一样的。比如，`Chrome` 浏览器在 `Linux` 和 `windows` 系统下，需要按下 `Alt + k`，才会跳转到 `example.com`。

1.2. download

`download` 属性表示当前链接不是用来浏览，而是用来下载的。它的值是一个字符串，表示用户下载得到的文件名。

```
// <a id="test" href="foo.jpg">下载</a>
let a = document.getElementById('test');
a.download = 'bar.jpg';
```

上例中，`<a>` 元素是一个图片链接，默认情况下，点击后图片会在当前窗口加载。设置了 `download` 属性以后，再点击这个链接，就会下载对话框，询问用户保存位置，而且下载的文件名为 `bar.jpg`。

1.3. hreflang

`hreflang` 属性用来读写 `<a>` 元素的 HTML 属性 `hreflang`，表示链接指向的资源的语言，比如 `hreflang="en"`。

```
// <a id="test" href="https://example.com" hreflang="en">test</a>
let a = document.getElementById('test');
a.hreflang // "en"
```

1.4. referrerPolicy

`referrerPolicy` 属性用来读写 `<a>` 元素的 HTML 属性 `referrerPolicy`，指定当用户点击链接时，如何发送 HTTP 头信息的 `referer` 字段。

HTTP 头信息的 `referer` 字段，表示当前请求是从哪里来的。它的格式可以由 `<a>` 元素的 `referrerPolicy` 属性指定，共有三个值可以选择。

- `no-referrer`：不发送 `referer` 字段。
- `origin`：`referer` 字段的值是 `<a>` 元素的 `origin` 属性，即协议 + 主机名 + 端口。
- `unsafe-url`：`referer` 字段的值是 `origin` 属性再加上路径，但不包含 `#` 片段。这种格式提供的信息最详细，可能存在信息泄漏的风险。

```
// <a id="test" href="https://example.com" referrerpolicy="no-referrer">test</a>
let a = document.getElementById('test');
a.referrerPolicy // "no-referrer"
```

1.5. rel

`rel` 属性用来读写 `<a>` 元素的 HTML 属性 `rel`，表示链接与当前文档的关系。

```
// <a id="test" href="https://example.com" rel="license">license.html</a>
let a = document.getElementById('test');
a.rel // "license"
```

上例表示，该链接是一个许可。

1.6. tabIndex

`tabIndex` 属性的值是一个整数，用来读写当前 `<a>` 元素在文档里面的 `Tab` 键遍历顺序。

```
// <a id="test" href="https://example.com">test</a>
let a = document.getElementById('test');
```

```
a.tabIndex // 0
```

1.7. target

`target` 属性用来读写 `<a>` 元素的 HTML 属性 `target`。

```
// <a id="test" href="https://example.com" target="_blank">test</a>
let a = document.getElementById('test');
a.target // "_blank"
```

1.8. text

`text` 属性用来读写 `<a>` 元素的链接文本，等同于当前节点的 `textContent` 属性。

```
// <a id="test" href="https://example.com">test</a>
let a = document.getElementById('test');
a.text // "test"
a.textContent // "test"
```

1.9. type

`type` 属性用来读写 `<a>` 元素的 HTML 属性 `type`，表示链接目标的 MIME 类型。

```
// <a id="test" type="video/mp4" href="example.mp4">video</a>
let a = document.getElementById('test');
a.type // "video/mp4"
```

2. 方法

`<a>` 元素的方法都是继承的，主要有以下三个。

- `blur()`：从当前元素移除键盘焦点，详见 `HTMLElement` 接口的介绍。
- `focus()`：当前元素得到键盘焦点，详见 `HTMLElement` 接口的介绍。
- `toString()`：返回当前 `<a>` 元素的 HTML 属性 `href`。

```
document.links[14].href
document.links[14].toString()
// 都返回： "https://www.wangdoc.com/javascript/elements/a.html#tabindex-
%E5%B1%9E%E6%80%A7"
```