【BOM】

> 原创内容，转载请注明出处！

一、BOM是什么

BOM（Browser Object Model，浏览器对象模型）是 JS 与 浏览器窗口交互的接口。

一些与浏览器改变尺寸、滚动条滚动相关的特效，都要借助 BOM 技术。

二、window对象

window 对象是当前 JS 脚本运行所处的窗口，而这个窗口中包含 DOM 结构，`window.document` 属性就是 `document` 对象。

在有标签功能的浏览器中，每个标签都拥有自己的 `window` 对象；也就是说，同一个窗口的标签页之间不会共享一个 window 对象。

三、全局变量是window的属性

全局变量会成为 window 对象的属性。

```
var a = 10;
console.log(window.a == a); // true
```

这就意味着，多个 js 文件之间是共享全局作用域的，即：js 文件没有作用域隔离功能。

四、内置函数普遍是window的方法

如 `setInterval()`、`alert()` 等内置函数，普遍是 window 的方法。

```
console.log(window.alert == alert); // true
console.log(window.setInterval == setInterval); // true
```

五、窗口尺寸相关属性

属性	意义
`innerHeight`	浏览器窗口的内容区域的高度，包含水平滚动条（如果有的话）
`innerWidth`	浏览器窗口的内容区域的宽度，包含垂直滚动条（如果有的话）
`outerHeight`	浏览器窗口的外部高度
`outerWidth`	浏览器窗口的外部宽度

获得不包含滚动条的窗口宽度，要用：

`document.documentElement.clientWidth`

> 浏览器的外宽指的是浏览器窗口边框的宽度。
>
> 当浏览器窗口全屏时：浏览器的外宽 == 浏览器内宽（包含滚动条）
>
> 当浏览器窗口不全屏时：浏览器的外宽 > 浏览器内宽（包含滚动条）

六、resize事件

在窗口大小改变之后，就会触发 `resize` 事件，可以使用 `window.onresize` 或者 `window.addEventListener('resize')` 来绑定事件处理函数。

七、已卷动高度

`window.scrollY` 属性表示在垂直方向已滚动的像素值。

`document.documentElement.scrollTop` 属性也表示窗口卷动高度。

```
// 可以利用此种方式获得窗口卷动的高度
var scrollTop = window.scrollY || document.documentElement.scrollTop;
```

- document.documentElement.scrollTop 是可以手动给定值的，以达到跳动到任何指定滚动高度处
- window.scrollY 是只读的，不可以手动给值

八、scroll事件

在窗口被卷动之后，就会触发 scroll 事件，可以使用：

`window.onscroll` 或者 `window.addEventListener('scroll')` 来绑定事件处理函数。

九、Navigator对象

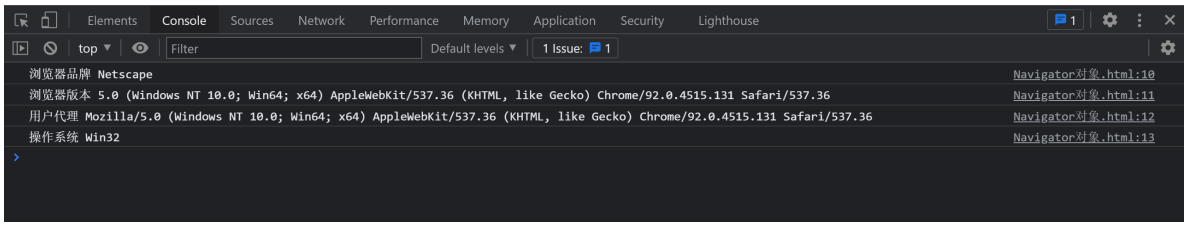
`window.navigator` 属性可以检索 navigator 对象，它内部含有用户此次活动的浏览器的相关属性和标识。

属性	意义
`appName`	浏览器官方名称
`appVersion`	浏览器版本
`userAgent`	浏览器用户代理（含有内核信息和封装壳信息）
`platform`	用户操作系统

【案例】

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
```

```
<body>
    <script>
        console.log('浏览器品牌', navigator.appName);
        console.log('浏览器版本', navigator.appVersion);
        console.log('用户代理', navigator.userAgent);
        console.log('操作系统', navigator.platform);
    </script>
</body>
</html>
```



十、识别用户浏览器品牌

识别用户浏览器品牌通常使用 `navigator.userAgent` 属性。

```
var sUsrAg = navigator.userAgent;

if (sUsrAg.indexOf("Firefox") > -1) {
} else if (sUsrAg.indexOf("Opera") > -1) {
} else if (sUsrAg.indexOf("Edge") > -1) {
} else if (sUsrAg.indexOf("Chrome") > -1) {
} else if (sUsrAg.indexOf("Safari") > -1) {
} else {
}
```

十一、History对象

`window.history` 对象提供了操作浏览器会话历史的接口。

常用操作就是模拟浏览器回退按钮。

```
history.back(); // 等同于点击浏览器的回退按钮
history.go(-1); // 等同于 history.back();
```

【案例】

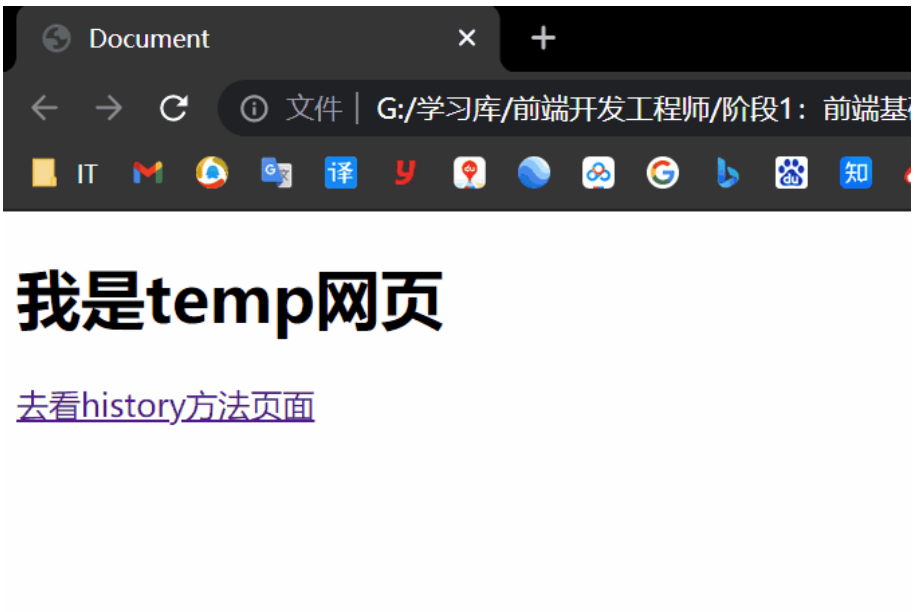
- temp.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>
    <h1>我是temp网页</h1>
    <a href="history方法.html">去看history方法页面</a>
</body>
```

```
</html>
```

- history.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>
    <h1>我是history方法网页</h1>
    <button id="btn">回退</button>
    <!--  链接可以使用内嵌 javascript 脚本的方式 -->
    <a href="javascript:history.back();">链接：回退</a>

    <script>
        var btn = document.getElementById('btn');

        btn.onclick = function () {
            // history.back();
            history.go(-1);
        };
    </script>
</body>

</html>
```



十二、Location对象

`window.location` 标识当前所在网址，可以通过给这个属性赋值命令浏览器进行页面跳转。

```
window.locaiton = 'http://www.imooc.com';
window.location.href = 'http://www.imooc.com';
```

十三、重新加载当前页面

可以调用 location 的 `reload` 方法以重新加载当前页面，参数 true 表示强制从服务器强制加载。

```
window.location.reload(true);
```

【案例】

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>
    <button id="btn1">点我去看慕课</button>
    <button id="btn2">刷新</button>
    <script>
        var btn1 = document.getElementById('btn1');
        var btn2 = document.getElementById('btn2');

        btn1.onclick = function () {
            window.location = 'http://www.imooc.com';
        };

        btn2.onclick = function () {
            window.location.reload(true);
        };
    </script>
</body>

</html>
```

十四、GET请求查询参数

`window.location.search` 属性即为当前浏览器的 GET 请求查询参数。

比如网址： https://www.imooc.com/?a=1&b=2

```
console.log(window.location.search);    // "?a=1&b=2"
```

> 关于 GET 及 POST 的详细内容在 Ajax 中介绍。

十五、BOM特效开发

15.1 返回顶部按钮制作

返回顶部的原理：改变 `document.documentElement.scrollTop` 属性，通过定时器逐步改变此值，则将用动画形式返回顶部。

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        body {
            height: 5000px;
            background-image: linear-gradient(to bottom, rgb(255, 0, 149), rgb(7, 185,
255), rgb(0, 255, 76));
        }

        .backtotop {
            width: 60px;
            height: 60px;
            background-color: rgba(255, 255, 255, .6);
            position: fixed;
            bottom: 100px;
            right: 100px;
            /* 小手状 */
            cursor: pointer;
        }
    </style>
</head>

<body>
    <div class="backtotop" id="backtotopBtn">返回顶部</div>

    <script>
        var backtotopBtn = document.getElementById('backtotopBtn');

        var timer;
        backtotopBtn.onclick = function () {
            // 设表先关
            clearInterval(timer);

            // 设置定时器
            timer = setInterval(function () {
                // 不断让scrollTop减少
                document.documentElement.scrollTop -= 200;
                // 定时器肯定要停
                if (document.documentElement.scrollTop <= 0) {
                    clearInterval(timer);
                }
            }, 20);
        };
    </script>
</body>

</html>
```

15.2 楼层导航小效果

DOM 元素都有 `offsetTop` 属性，表示此元素到定位祖先元素的垂直距离。

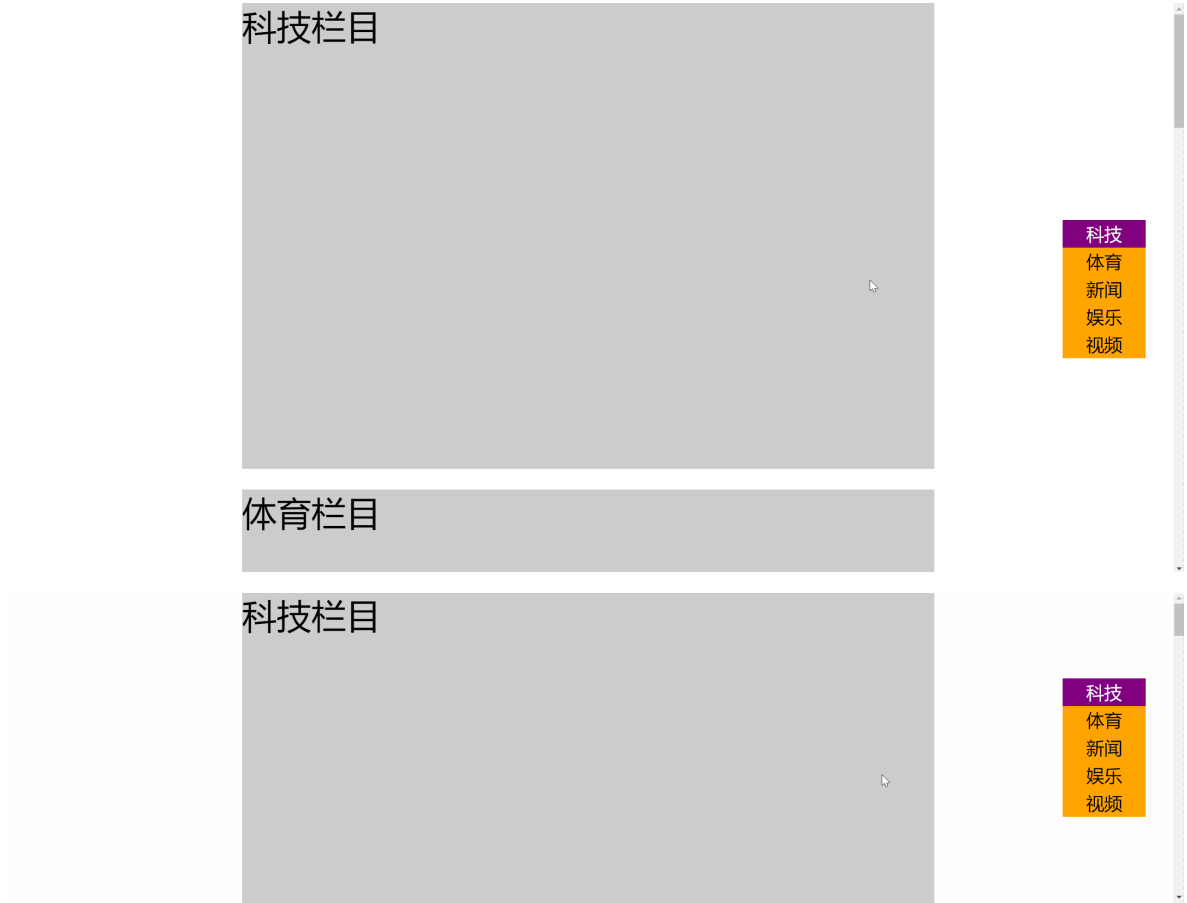
定位祖先元素：在祖先中，离自己最近的且拥有定位属性的元素。

> 即：offsetTop 属性可以得到该元素与离自己最近且拥有定位的祖先元素顶部的距离值。
>
> 假如，没有祖先有定位，那么直接得到该元素距离页面顶部的距离值。

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        .content-part {
            width: 1000px;
            margin: 0px auto;
            margin-bottom: 30px;
            background-color: #ccc;
            font-size: 50px;
        }

        .floornav {
            position: fixed;
            right: 40px;
            top: 50%;
            margin-top: -100px;
            width: 120px;
            height: 200px;
            background-color: orange;
        }

        .floornav ul {
            list-style: none;
        }

        .floornav ul li {
            width: 120px;
            height: 40px;
            line-height: 40px;
            text-align: center;
            font-size: 26px;
            /* 小手指针 */
            cursor: pointer;
        }

        .floornav ul li.current {
            background: purple;
            color: white;
        }
```

```
    </style>
</head>

<body>
    <nav class="floornav">
        <ul id="list">
            <li data-n="科技" class="current">科技</li>
            <li data-n="体育">体育</li>
            <li data-n="新闻">新闻</li>
            <li data-n="娱乐">娱乐</li>
            <li data-n="视频">视频</li>
        </ul>
    </nav>

    <section class="content-part" style="height:674px;" data-n="科技">
        科技栏目
    </section>

    <section class="content-part" style="height:567px;" data-n="体育">
        体育栏目
    </section>

    <section class="content-part" style="height:739px;" data-n="新闻">
        新闻栏目
    </section>

    <section class="content-part" style="height:574px;" data-n="娱乐">
        娱乐栏目
    </section>

    <section class="content-part" style="height:1294px;" data-n="视频">
        视频栏目
    </section>

    <script>
        // 使用事件委托给li添加监听
        var list = document.getElementById('list');
        var contentParts = document.querySelectorAll('.content-part');
        var lis = document.querySelectorAll('#list li');

        list.onclick = function (e) {
            if (e.target.tagName.toLowerCase() == 'li') {
                // getAttribute表示得到标签身上的某个属性值
                var n = e.target.getAttribute('data-n');

                // 可以用属性选择器(就是方括号选择器)来寻找带有相同data-n的content-part
                var contentPart = document.querySelector('.content-part[data-n=' + n +
']');

                // 让页面的卷动自动成为这个盒子的offsetTop值
                document.documentElement.scrollTop = contentPart.offsetTop;
            }
        }


        // 在页面加载好之后，将所有的content-part盒子的offsetTop值推入数组
        var offsetTopArr = [];
```

```
        // 遍历所有的contentPart，将它们的净位置推入数组
        for (var i = 0; i < contentParts.length; i++) {
            offsetTopArr.push(contentParts[i].offsetTop);
        }
        // 为了最后一项可以方便比较，我们可以推入一个无穷大
        offsetTopArr.push(Infinity);

        console.log(offsetTopArr);

        // 当前所在楼层
        var nowfloor = -1;

        // 窗口的卷动
        window.onscroll = function () {
            // 得到当前的窗口卷动值
            var scrollTop = document.documentElement.scrollTop;

            // 遍历offsetTopArr数组，看看当前的scrollTop值在哪两个楼层之间
            for (var i = 0; i < offsetTopArr.length; i++) {
                if (scrollTop >= offsetTopArr[i] && scrollTop < offsetTopArr[i + 1]) {
                    break;
                }
            }
            // 退出循环的时候，i是几，就表示当前楼层是几
            // 如果当前所在楼层，不是i，表示环楼了
            if (nowfloor != i) {
                console.log(i);
                // 让全局变量改变为这个楼层号
                nowfloor = i;

                // 设置下标为i的项有cur
                for (var j = 0; j < lis.length; j++) {
                    if (j == i) {
                        lis[j].className = 'current';
                    } else {
                        lis[j].className = '';
                    }
                }
            }
        };
    </script>
</body>

</html>
```


科技栏目
体育栏目
科技
体育
新闻
娱乐
视频

科技栏目
科技
体育
新闻
娱乐
视频
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
top
Filter
Default levels
No Issues
(6) [0, 704, 1301, 2070, 2674, Infinity]
楼层导航小效果.html:114