复习：

数据类型：

引用类型：

原生

自定义

宿主：DOM和BOM

DOM操作文档内容：

1. 查找节点

getElementById() / ...ByName() / ...ByClassName() / ...ByTagName() / querySelector() / ...All() /

1. 新建节点——document

createElement() / createTextNode() / createAttribute() / createDocumentFragment()

1. 添加节点

appendChild() / insertBefore() / setAttributeNode()

1. 删除节点

removeChild() / removeAttribute()

1. 替换节点

replaceChild() / setAttributeNode()

BOM:

window

alert() / prompt() / confirm()

close() / open(url, name, features)

setInterval() / clearInterval() / setTimeout() / clearTimeout

document:

history:

navigator

location

event

screen

1.window.navigator对象的使用

常用的属性成员：

userAgent: 返回浏览器/操作系统的信息

2.window.location对象的使用

Location对象表示浏览器当前显示的页面的URL地址

href：读取或设置当前浏览器显示的页面的url，用于**页面跳转**

reload() 重新加载当前页面/刷新

|  |
| --- |
| 面试题：客户端可以实现页面跳转的方法：  (1)<a href=”url”></a>  (2)<form action=”url”></form>  (3)location.href=”url”; location.assign(‘url’);  (4)window.open(‘url’);  (5)<meta http-equiv=”Refresh” content=”3;url”/> |
| 服务器端可以实现页面跳转的方法——控制HTTP响应头 |

3.window.history对象的使用

History中保存着当前浏览器访问的所有页面的历史顺序

length：访问历史中共有多少个页面

back() 退回到访问历史中的上一个页面

forward() 前进到访问历史中的下一页页面

go(num) 直接跳转到访问历史中的下num个页面； go(1) <=> forward(); go(-1) <=>back();

4.window.screen对象的使用

Screen对象代表当前浏览器窗口所在的显示屏幕

width：当前屏幕的水平分辨率

height：当前屏幕的竖直分辨率

availWidth：当前屏幕的水平分辨率 – 任务条宽

availHeight：当前屏幕的竖直分辨率 – 任务条高

练习：

打开一个600\*500的新窗口，在屏幕中央显示，新窗口中的文本也显示在正中央

5.Event对象

每当浏览器中发生一个事件时，浏览器会自动创建一个event对象，并传递给指定的事件处理函数。浏览器中的可以分为三种类型：

1. MouseEvent：包含事件源、事件发生坐标(x / y)
2. KeyEvent：包含事件源、keyCode
3. StateEvent：state

**事件句柄**(event handler)：每个事件发生时，浏览器会查看该事件是否指定了对应的处理函数，该函数即为event hanlder，若指定了，则浏览器会自动调用该函数并把创建的event对象传递进来。

(1)<button onclick=”语句1;语句2;..”>

(2)<button onclick=”自定义函数()”>

(3)<button id=”bt3”> **把HTML(内容)与JS(行为)彻底分离**

**<script>**

**bt3.onclick = function(){**

**}**

**//或者**

**bt3.setAttribute(‘onclick’, ‘func()’);**

**</script>**

注意：若使用(1)和(2)进行了多次绑定，则只有第一次绑定生效；若使用(3)进行了多次绑定，只有最后一次绑定生效。

6.HTML元素中的事件的冒泡机制

<div id=”d1” onclick=””>

<div id=”d2” onclick=””>

<div id=”d3” onclick=””>

</div>

</div>

</div>

若#d1被单击，则单击事件只对#d1生效；

若#d2被单击，则单击事件会先后对#d2、#d1生效；

若#d3被单击，则单击事件会先后对#d3、#d2、#d1生效。

可以在事件处理函数中使用下述语句取消子元素触发的事件继续冒泡给父元素：

**event.cancelBubble = true;**

7.event对象的使用——浏览器兼容性相当不好！

**如何获取event对象**

IE: HTML/JavaScript中的事件句柄中都可以直接使用event对象

FF: HTML中的事件句柄可以直接使用event对象；JavaScript中的事件句柄不能直接使用event对象！

解决方案：

**<button onclick=”f1(event);”> js:function f1(e){ e... }**

**<button></button> js:btn.onclick = function(e){ e... }**

**获取event对象的属性**

cancelBubble： boolean类型，默认为false，指定是否取消冒泡，指定为true的话，当前事件不会再向父元素传递。

clientX：事件发生时相对于浏览器显示区域(body)的X坐标

clientY：事件发生时相对于浏览器显示区域(body)的Y坐标

screenX：事件发生时相对于浏览器显示器的X坐标

screenY：事件发生时相对于浏览器显示器的Y坐标

offsetX：事件发生时相对于事件源对象的X坐标

offsetY：事件发生时相对于事件源对象的Y坐标

keyCode：键盘事件发生时，按下的键的键码

srcElement：IE/Webkit中用于获取事件源

target: FF/Webkit中用于获取事件源

练习：

