# 第七章 javascript下

本节所讲内容：

7.1 JavaScript DOM

7.2 JavaScript事件

7.1 JavaScript DOM

DOM（Document Object Model）即文档对象模型，针对HTML和XML文档的API（应用程序接口）。DOM描绘了一个层次化的节点树，运行开发人员添加、移除和修改页面的某一部分。

DOM中的三个字母，D（文档）可以理解为整个Web加载的网页文档；O（对象）可以理解为类似window对象之类的东西，可以调用属性和方法，这里我们说的是document对象；M（模型）可以理解为网页文档的树型结构。

W3C提供了比较方便简单的定位节点的方法和属性，以便我们快速的对节点进行操作。

7.1.1查找元素

|  |  |
| --- | --- |
| 方法 | 描述 |
| document.getElementById(*id*) | 通过元素 id 来查找元素 |
| document.getElementsByTagName(*name*) | 通过标签名来查找元素 |
| document.getElementsByClassName(*name*) | 通过类名来查找元素 |

1. getElementById()方法

getElementById()方法，接受一个参数：获取元素的ID。如果找到相应的元素则返回该元素的HTMLDivElement对象，如果不存在，则返回null。

document.getElementById('box'); //获取id为box的元素节点

PS：上面的例子，默认情况返回null，这无关是否存在id="box"的标签，而是执行顺序问题。

解决方法：

1.把script调用标签移到html末尾即可；

2.使用onload事件来处理JS，等待html加载完毕再加载onload事件里的JS。

window.onload = function () { //预加载html后执行

document.getElementById('box');

};

2.getElementsByTagName()方法

getElementsByTagName()方法将返回一个对象数组HTMLCollection(NodeList)，这个数组保存着所有相同元素名的节点列表。

document.getElementsByTagName('\*'); //获取所有元素

document.getElementsByTagName('li'); //获取所有li元素，返回数组（列表）

3.getElementsByName()方法

getElementsByName()方法可以获取相同名称(name)的元素，返回一个对象数组HTMLCollection(NodeList)。

document.getElementsByName('add') //获取input元素

7.1.2 增添数据

|  |  |
| --- | --- |
| 方法 | 描述 |
| element.innerHTML = *new html content* | 改变元素的 inner HTML |
| element.attribute = *new value* | 改变 HTML 元素的属性值 |
| element.setAttribute(*attribute*, *value*) | 改变 HTML 元素的属性值 |
| element.style.property = *new style* | 改变 HTML 元素的样式 |
| element .removeAttribute() | 去除 HTML 元素的样式 |

attribute [əˈtrɪbjuːt , ˈætrɪbjuːt]：属性

property[ˈprɒpəti]:财产

<div id = 'box1' >定位测试1</div>  
<div id = 'box2' >定位测试2</div>  
<script>  
 var *element* = *document*.getElementById('box1');  
 *// element.innerHTML = '哈哈';  
 // element.innerText = '这只是个测试';  
 // element.class = 'change';  
 // element.setAttribute('id','1')  
 element*.style.color = 'red';  
</script>  
</body>  
</html>

如需改变 HTML 元素的样式，请使用这个语法：

document.getElementById(id).style.属性名=新样式

var box = document.getElementById('box'); //获取box

box.style; //CSSStyleDeclaration

box.style.color; //red

box.style.fontSize; //20px

注意:如果遇到css样式具有横杠例：font-size需要改写成fontSize，去除横杠并将首字母大写

1.通过style内联获取元素的大小

var box = document.getElementById('box'); //获取元素

box.style.width; //200px、空

box.style.height; //200px、空

7.1.3改变HTML元素

|  |  |
| --- | --- |
| 方法 | 描述 |
| document.createElement(*element*) | 创建 HTML 元素 |
| document.removeChild(*element*) | 删除 HTML 元素 |
| document.appendChild(*element*) | 添加 HTML 元素 |
| document.replaceChild(*element*) | 替换 HTML 元素 |
| document.write(*text*) | 写入 HTML 输出流 |

## 7.1.4 添加事件处理程序

|  |  |
| --- | --- |
| 方法 | 描述 |
| document.getElementById(id).onclick = function(){*code*} | onclick 事件添加事件处理程序 |

## 查找 HTML 对象

首个 HTML DOM Level 1 (1998)，定义了 11 个 HTML 对象、对象集合和属性。它们在 HTML5 中仍然有效。

后来，在 HTML DOM Level 3，加入了更多对象、集合和属性。

7.1.5 HTMl属性

|  |  |  |
| --- | --- | --- |
| 属性 | 描述 | DOM |
| document.anchors | 返回拥有 name 属性的所有 <a> 元素。 | 1 |
| document.applets | 返回所有 <applet> 元素（HTML5 不建议使用） | 1 |
| document.baseURI | 返回文档的绝对基准 URI | 3 |
| document.body | 返回 <body> 元素 | 1 |
| document.cookie | 返回文档的 cookie | 1 |
| document.doctype | 返回文档的 doctype | 3 |
| document.documentElement | 返回 <html> 元素 | 3 |
| document.documentMode | 返回浏览器使用的模式 | 3 |
| document.documentURI | 返回文档的 URI | 3 |
| document.domain | 返回文档服务器的域名 | 1 |
| document.domConfig | 废弃。返回 DOM 配置 | 3 |
| document.embeds | 返回所有 <embed> 元素 | 3 |
| document.forms | 返回所有 <form> 元素 | 1 |
| document.head | 返回 <head> 元素 | 3 |
| document.images | 返回所有 <img> 元素 | 1 |
| document.implementation | 返回 DOM 实现 | 3 |
| document.inputEncoding | 返回文档的编码（字符集） | 3 |
| document.lastModified | 返回文档更新的日期和时间 | 3 |
| document.links | 返回拥有 href 属性的所有 <area> 和 <a> 元素 | 1 |
| document.readyState | 返回文档的（加载）状态 | 3 |
| document.referrer | 返回引用的 URI（链接文档） | 1 |
| document.scripts | 返回所有 <script> 元素 | 3 |
| document.strictErrorChecking | 返回是否强制执行错误检查 | 3 |
| document.title | 返回 <title> 元素 | 1 |
| document.URL | 返回文档的完整 URL | 1 |

7.2 JavaScript事件

JavaScript事件是由访问Web页面的用户，引起的一系列操作，例如：用户点击。当用户执行某些操作的时候，再去执行一系列代码。

//在HTML中把事件处理函数作为属性执行JS代码

<input type="button" value="按钮" onclick="alert('Lee');" /> //注意单双引号

//在HTML中把事件处理函数作为属性执行JS函数

<input type="button" value="按钮" onclick="box();" /> //执行JS的函数

由于内联模型违反了HTML与JavaScript代码层次分离的原则。为了解决这个问题，我们可以在JavaScript中处理事件。这种处理方式就是脚本模型。

var input = document.getElementsByTagName('input')[0]; //得到input对象

input.onclick = function () { //匿名函数执行

alert('Lee');

};

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 事件处理函数 | 影响的元素 | 如何发生 | FF | IE |
| onabort | 图像 | 当图像加载被中断时 | 3 | 4 |
| onblur | 窗口、框架、所有表单对象 | 当焦点从对象上移开时 | 2 | 3 |
| onchange | 输入框，选择框和文本区域 | 当改变一个元素的值且失去焦点时 | 2 | 3 |
| onclick | 链接、按钮、表单对象、图像映射区域 | 当用户单击对象时 | 2 | 3 |
| ondblclick | 链接、按钮、表单对象 | 当用户双击对象时 | 4 | 4 |
| ondragdrop | 窗口 | 当用户将一个对象拖放到浏览器窗口时 | 3 | 4 |
| onError | 脚本 | 当脚本中发生语法错误时 | 2 | 3 |
| onfocus | 窗口、框架、所有表单对象 | 当单击鼠标或者将鼠标移动聚焦到窗口或框架时 | 4 | 3 |
| onkeydown | 文档、图像、链接、表单 | 当按键被按下时 | 4 | 3 |
| onkeypress | 文档、图像、链接、表单 | 当按键被按下然后松开时 | 4 | 3 |
| onkeyup | 文档、图像、链接、表单 | 当按键被松开时 | 2 | 3 |
| onload | 主题、框架集、图像 | 文档或图像加载后 | 4 | 4 |
| onunload | 主体、框架集 | 文档或框架集卸载后 | 6 | 3 |
| onmouseout | 链接 | 当图标移除链接时 | 4 | 4 |
| onmouseover | 链接 | 当鼠标移到链接时 | 2 | 3 |
| onmove | 窗口 | 当浏览器窗口移动时 | 4 | 4 |
| onreset | 表单复位按钮 | 单击表单的reset按钮 | 3 | 4 |
| onresize | 窗口 | 当选择一个表单对象时 | 4 | 4 |
| onselect | 表单元素 | 当选择一个表单对象时 | 2 | 3 |
| onsubmit | 表单 | 当发送表格到服务器时 | 2 | 3 |
| 事件处理函数 | 影响的元素 | 何时发生 | 2 | 3 |

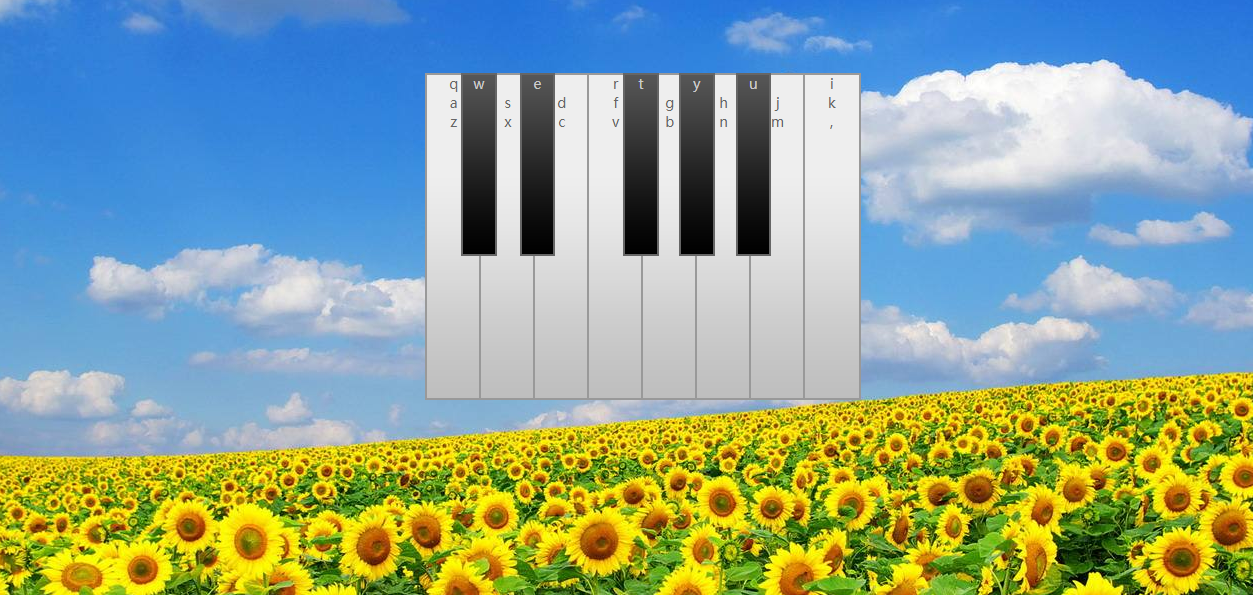
<style>  
 #box1{  
 width: 300px;  
 height: 230px;  
 border: 1px solid black;  
 overflow: hidden;  
 }  
</style>

……

<div id = 'box1' >定位测试1<img src="1.gif"></div>  
<div id = 'box2' >定位测试2</div>  
<input type="text" value="写入一些信息" id="box3" >  
<script>  
 */\*  
 var element = document.getElementById('box1');  
 element.onlclick = function () {  
 alert('王境泽');  
 };//鼠标点击事件  
 element.ondblclick = function () {  
 alert('王境泽');  
 };//鼠标双击事件  
 element.onmousedown = function () {  
 alert('王境泽');  
 };//当鼠标点击下去触发事件；  
 \*/* var *box3* = *document*.getElementById('box3');  
 */\*box3.onselect = function () {  
 alert('调皮');  
 };  
 box3.onchange = function () {  
 alert('改变');  
 };  
 box3.onfocus = function(){  
 alert('onfocus');  
 }  
 box3.onblur = function(){  
 alert('onblur');  
 };*

*box3.onkeydown = function () {  
 alert('嘿嘿')  
};  
box3.onkeypress = function () {  
 alert('嘿嘿')  
};  
box3.onkeyup = function () {  
 alert('嘿嘿')  
};\*/*

</script>

实战：利用上边的代码实习简单的钢琴界面  


总结：

7.1 JavaScript DOM

7.2 JavaScript事件