day09_JavaScript高级

一.DOM、事件的简单学习

为了满足后续学习的案例需求，我们先简单了解一下DOM、事件的基本使用。

1.DOM的简单学习

DOM的功能：控制html文档的内容

(1).获取页面标签(元素)对象：

document.getElementById("id值")：通过元素的id获取元素对象

(2).操作获取到的标签(元素)对象：

①.修改标签的属性值：明确获取的标签是哪一个？查看API文档，找其中有哪些属性可以设置
②.修改标签体内容：使用innerHTML属性来修改标签体内容

#.注意：获取html标签对象时，<script>...</script>标签一定要写在html标签的下方。如果写在其上方，html标签还没有被加载，将获取不到其元素对象。

2.事件的简单学习

每个html标签都具有事件属性，事件属性的属性值就是js代码。例如：每个标签都具有单击事件属性onclick，当我们对某个标签的onclick属性进行设置后，单击该标签就会执行相应的js代码，就完成了事件的绑定操作。绑定事件有以下两种方法：

法一：直接设置html标签的事件属性，属性值就是js代码

法二：使用DOM获取元素对象，再通过元素对象来设置事件属性的属性值

#.注意：
①.在设置事件属性的js代码时，一般我们会将其封装为一个方法，利于程序的可读性、扩展性。
②.在使用法一的时候，html、js代码还是混合在一起的，并没有达到解耦的效果，所以推荐使用法二
③.在使用法二的时候，一般还会使用如下匿名方法的方式：

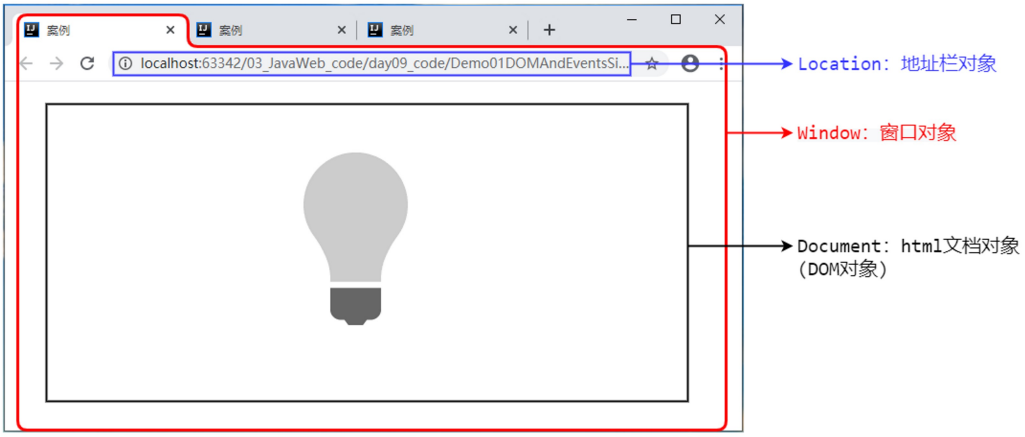
```
light2.onclick = function(){
    alert("on.gif被点击了");
};
```

3.案例：电灯开关，需要实现的效果见 "参考\案例1_电灯开关"

二.BOM

BOM(Browser Object Model)：浏览器对象模型，其中提供了一系列对象用于与浏览器窗口进行交互，这些对象统称为BOM对象。即：将浏览器的各个组成部分封装成对象，我们可以使用这些对象来与浏览器窗口进行交互。

首先我们要知道，浏览器的各个组成部分被封装成了什么对象？BOM浏览器对象模型如下图所示：



History：历史记录对象
Navigator：浏览器对象，与浏览器相关的操作
Screen：显示器屏幕对象，与显示器屏幕相关的操作

Window窗口对象是BOM对象中最重要的一个对象，一个浏览器窗口就对应一个Window对象。由于一个浏览器窗口中含有：地址栏、历史记录、该窗口所属的浏览器、该窗口所属的显示器屏幕、html文档等组成部分，所以该浏览器窗口所对应的Window对象也含有：Location地址栏对象、History历史记录对象、Navigator浏览器对象、Screen显示器屏幕对象、Document文档对象等。

什么意思呢？意思就是：上述对象均属于Window窗口对象，即每一个Window窗口对象都含有这些对象。

1.Window窗口对象

Window窗口对象不需要创建，可以直接使用window来调用其中的属性、方法。即：window.属性名；　window.方法名(); 并且window引用可以省略，直接写属性名、方法名即可。

(1).Window对象的方法：(window引用可以省略，直接写方法名即可)

①.与弹出框有关的方法：

alert()：弹出带有一段消息和一个确认按钮的警告框

confirm()：弹出带有一段消息以及确认按钮和取消按钮的对话框。参数为提示信息，返回值：点击确定则返回true，点击取消则返回false。

prompt()：弹出可提示用户输入的对话框。参数为提示信息，返回值为用户输入的值。

②.与打开、关闭有关的方法：

open()：打开一个新的浏览器窗口，返回值就是新打开的Window对象。并且参数传递URL，还可以打开指定URL的新窗口。

close()：关闭浏览器窗口。哪个Window对象调用close方法，就关闭哪个窗口。

③.与定时器有关的方法：

setTimeout(js代码，毫秒值)：在指定的毫秒数后调用函数或计算表达式。返回值为该定时器的唯一标识，用于取消定时器。

clearTimeout()：取消由setTimeout方法设置的定时器。

setInterval(js代码，毫秒值)：按照指定的周期来调用函数或计算表达式，即循环定时器。返回值为该定时器的唯一标识，用于取消定时器。

clearInterval()：取消由setInterval方法设置的定时器。

(2).Window对象的属性：(window引用可以省略，直接写属性名即可)

①.用来获取其他BOM对象：

Window对象具有location、history、Navigator、Screen属性，分别用来获取当前窗口的历史记录、地址栏、浏览器、显示器屏幕对象，所以上述4个对象不需要创建，直接使用Window对象即可获取。

②.用来获取DOM对象：

Window对象还具有document属性，用来获取当前窗口的Document文档对象，这是一个DOM对象。

#.注意：由Window对象具有的这些属性，也可以看出：其他BOM对象和Document文档对象都属于Window对象，即每一个Window窗口对象都含有这些对象！

(3).案例：轮播图，需要实现的效果见 "参考\案例2_轮播图"

2.Location地址栏对象

Location对象不需要创建，是使用Window窗口对象获取出来的，即：window.location，其中window引用可以省略。

(1).Location对象的方法和属性

reload()方法：重新加载当前文档，即：刷新页面

href属性：返回当前窗口地址栏中完整的URL

(2).案例：自动跳转首页，需要实现的效果见 "参考\案例3_自动跳转首页"

3.History历史记录对象

History历史记录对象：History对象不需要创建，是使用Window窗口对象获取出来的，即：window.history，其中window引用可以省略。

(1).History对象的方法：

back()：加载history列表中的前一个URL，即：后退

forward()：加载history列表中的下一个URL，即：前进

go(参数)：加载history列表中的某个具体页面。参数为正数：前进几个历史记录；参数为负数：后退几个历史记录

...

(2).History对象的属性：

length属性：返回当前窗口历史列表中的URL数量

...

#.注意：获取到的history历史记录对象，不是浏览器中的历史记录列表，而是当前窗口的历史记录列表！(当前窗口访问过的URL)

#.Navigator浏览器对象、Screen显示器屏幕对象的使用很少，在此不再赘述

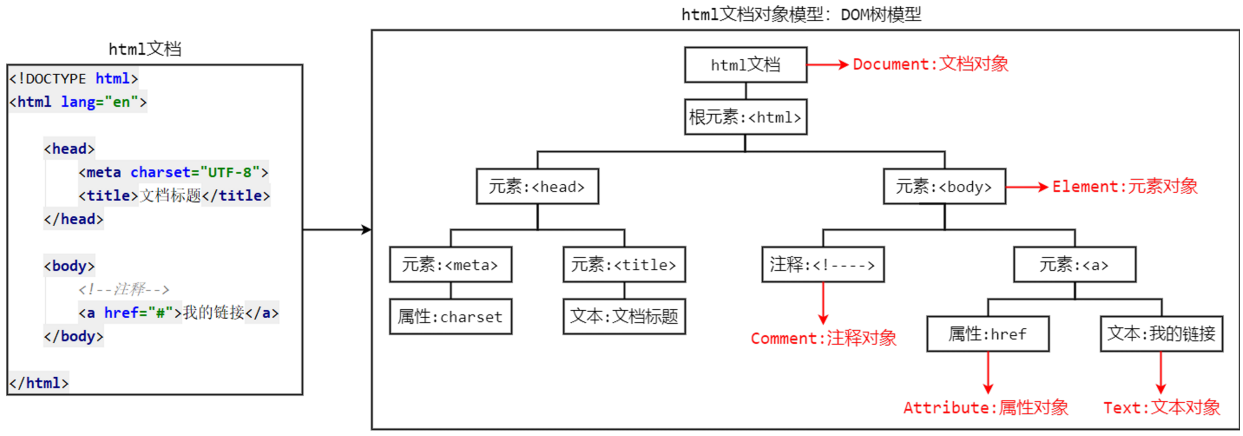
三.DOM

DOM(Document Object Model)：文档对象模型，将标记语言文档(html、xml文档等)的各个组成部分，封装为对象。可以使用这些对象，对标记语言文档中的标签元素进行CRUD的动态操作，这些对象统称为DOM对象。

由此可见，DOM文档对象模型是针对所有标记语言文档的，如：html文档、xml文档等，并不仅仅局限于某一种标记语言文档。所以DOM文档对象模型分为以下三种：

- 核心 DOM：一种针对所有标记语言文档的对象模型
- XML DOM：xml 文档对象模型，即：一种专门针对xml文档的对象模型
- HTML DOM：html 文档对象模型，即：一种专门针对html文档的对象模型

XML DOM、HTML DOM是在 "核心DOM" 的基础之上，分别针对xml、html文档做的进一步的扩展和封装。所以在这里我们只需要学习HTML DOM即可。首先我们要知道，html文档的各个组成部分被封装成了什么对象？html文档对象模型如下图所示：



当html文档加载进浏览器内存时，会被建模为上述DOM树模型。即：浏览器在显示html页面时，是根据其对应的DOM树结构来显示的。并且js代码对这些DOM对象的操作，会实时地更新DOM树结构

所以按照html文档对象模型的规则，html文档的各个组成部分会被封装成以下DOM对象：

- Document 文档对象：整个html文档会被映射为一个文档对象
- Element 元素对象：每个标签元素都会被映射为一个元素对象
- Attribute 属性对象、Text 文本对象、Comment 注释对象：每个属性、文本内容、注释都会被映射为一个属性对象、文本对象、注释对象
- Node 节点对象：上述 DOM 树中的所有节点都会被映射为一个节点对象，即：上述5个DOM对象都是节点对象

1.Document文档对象

Document文档对象：是DOM树中表示整个html文档的对象。Document文档对象不需要创建，可以使用window对象直接获取，即：window.document；其中window引用可以省略。

(1).获取Element元素对象的方法：可以使用Document对象来获取，html文档中各个标签所对应的Element元素对象

getElementById()：根据id属性值获取Element元素对象(id属性值唯一)
getElementsByTagName()：根据元素名称获取Element元素对象们，返回值是一个数组
getElementsByClassName()：根据class属性值获取Element元素对象们，返回值是一个数组
getElementsByName()：根据name属性值获取Element元素对象们，返回值是一个数组
...

(2).创建其他DOM对象的方法：可以使用Document对象，在当前文档中创建标签、属性、文本、注释所对应的DOM对象，返回值就是创建好的DOM对象

createElement()：创建Element元素对象，传入参数为html标签名称
createAttribute()：创建Attribute属性对象，传入参数为属性名称
createTextNode()：创建Text文本对象，传入参数为文本内容
createComment()：创建Comment注释对象，传入参数为注释内容
...

这些创建好的DOM对象并没有被添加到DOM树中，所以不会显示在html文档中。它们只是暂时存在于内存当中，想要把这些对象添加到DOM树中，需要使用后续的Node节点对象

2.Element元素对象

Element元素对象：html文档中的每个标签元素，都会被映射成DOM树中的一个元素对象。可以使用Document文档对象中的createElement()方法创建一个元素对象，也可以使用Document文档对象的方法直接获取现有的元素对象。

(1).Element元素对象的方法：控制标签的属性

setAttribute()：设置属性
removeAttribute()：删除属性
...

(2).Element元素对象的属性：

①.标签属性：可以直接使用对象名来获取、设置标签的属性，即：元素对象名.属性名 = "属性值";　(与setAttribute方法的作用一样)
②.innerHTML属性：返回标签的标签体内容

3.Node节点对象

Node节点对象：DOM树中的所有节点都会被映射为一个节点对象，即：DOM树中的每一个DOM对象都是一个Node节点对象，也就是说每一个DOM对象都可以被称为一个Node节点对象。它相当于是其他5个DOM对象的父对象，那么所有DOM对象都可以使用Node节点对象中的属性和方法。

Node节点对象是用来 CRUD DOM树的，所以js代码是通过Node节点对象来控制、修改DOM树，从而达到控制html文档的目的。

(1).Node节点对象的方法：

appendChild()：向节点的子节点列表的结尾添加新的子节点
removeChild()：删除(并返回)当前节点的指定子节点
replaceChild()：用新节点替换一个子节点
...

(2).Node节点对象的属性：

parentNode属性：返回节点的父节点
...

#.注意：
①.并不是所有的对象都拥有父节点或子节点。例如，文本节点不能拥有子节点，所以向类似的节点添加子节点就会导致DOM错误
②.在新的DOM规范中，标签的属性并不算是标签的子节点，所以属性的控制不能以父、子节点的方式来进行

4.案例：动态表格，需要实现的效果见 "参考\案例4 动态表格"

四.事件

事件：就是某些组件被执行了某些操作后，会触发某些代码的执行。主要分为以下几类事件：

1.点击事件

- onclick：单击事件
- ondblclick：双击事件

2.焦点事件

- onfocus：元素获得焦点时触发此事件
- onblur：元素失去焦点时触发此事件

3.加载事件

onload：页面、图片加载完成时触发此事件

#.页面加载事件 window.onload 的使用技巧：
之前为了能获取到html标签，我们都是把<script>标签写在html标签的下方，即：把<script>标签写在html标签的上方是获取不到标签的。现在使用 window.onload 就可以解决这个问题，我们可以把获取html标签的代码放在 window.onload的执行方法中，意味着只有等当前页面的所有html标签加载完成后才会执行获取标签的js代码，这样就能获取到下方的html标签了。

4.鼠标事件

- onmousedown：鼠标在某个元素上按下时触发此事件

- onmouseup：鼠标在某个元素上松开时触发此事件

- onmouseover：鼠标移到某元素之上时触发此事件
- onmouseout：鼠标从某元素上移开时触发此事件

- onmousemove：鼠标在某个元素上一旦被移动就会触发此事件

5.键盘事件

- onkeydown：某个键盘按键在某个元素上被按下时，会触发此事件
- onkeyup：某个键盘按键在某个元素上被松开时，会触发此事件
- onkeypress：某个键盘按键在某个元素上被按下又被松开后，会触发此事件

#.注意：可以使用 event.keyCode 取回被按下的字符

6.选择和改变事件

- onselect：文本被选中时会触发此事件
- onchange：域的内容被改变时会触发此事件

7.表单事件

- onsubmit：表单被提交时会触发此事件
- onreset：表单被重置时会触发此事件

#.onsubmit事件可以阻止表单提交：

当onsubmit事件被触发时，其执行的方法返回值为false，就会阻止表单的提交，即表单的数据不会被提交。其执行的方法返回值为true，或者没有返回值，表单数据才会被成功提交。(常用于表单提交时的数据校验，若数据校验不通过，则阻止表单的提交)

onreset事件也有同样的机制!

8.事件的案例：表格全选，需要实现的效果见 "参考\案例5_表格全选"

9.JavaScript综合案例：表单验证，需要实现的效果见 "参考\案例6_表单验证"