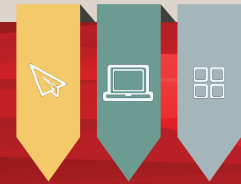


第 5 章 DOM 编程

主讲师：黄玲



回顾



-  document 对象提供了哪三种
- 访问 DOM 元素的方法？

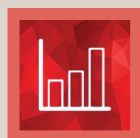




使用 getElement 系列方法实现 DOM 元素的查找和定位



使用 Core DOM 标准操作实现节点的增删改查



使用 HTML DOM 特有操作实现 HTML 元素内容修改

本章任务



制作动态添加表格的特效



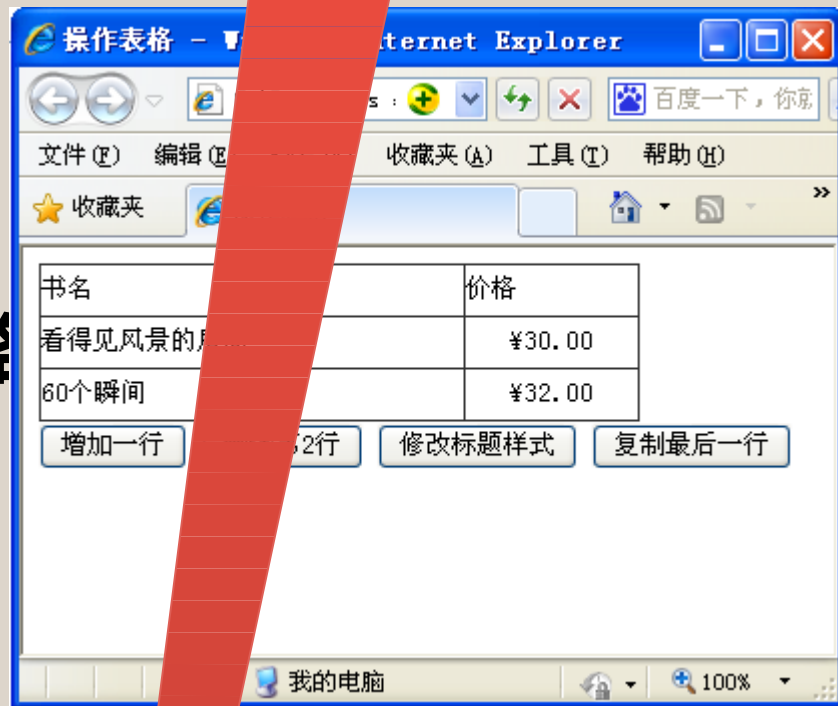
任务 1 : 画出页面的文档节点结构图



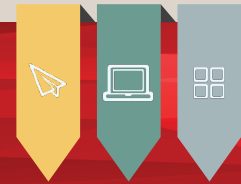
任务 2 : 使用 Core DOM 方式动态添加表格



任务 3 : 使用 HTML DOM 方式动态添加表格



任务 1: 画出页面的文档节点结构图



任务描述：

根据下面一段 html 代码，画出该 html 页面的 DOM 结构图。

```
<html>
```

```
<head>
```

```
<title>DOM 节点 </title>
```

```
</head>
```

```
<body>
```

```
  
```

```
  <h1> 喜欢的水果 </h1>
```

```
  <p>DOM 应用 </p>
```

```
</body>
```

```
</html>
```



任务 1: 画出页面的文档节点结构图



任务分析：

根据 DOM 模型，HTML 文档中的所有节点组成了一个文档树（或节点树）。HTML 文档中的每个元素、属性、文本等都代表着树中的一个节点。

树起始于文档节点，并由此继续伸出枝条，直到处于这棵树最低级别的所有文本节点为止。

1.1 回顾 DOM 模型的相关概念



什么是 DOM ?

文档对象模型 (Document Object Model)

通过 DOM 可以动态改变文档内容

动态改变文档内容的原理



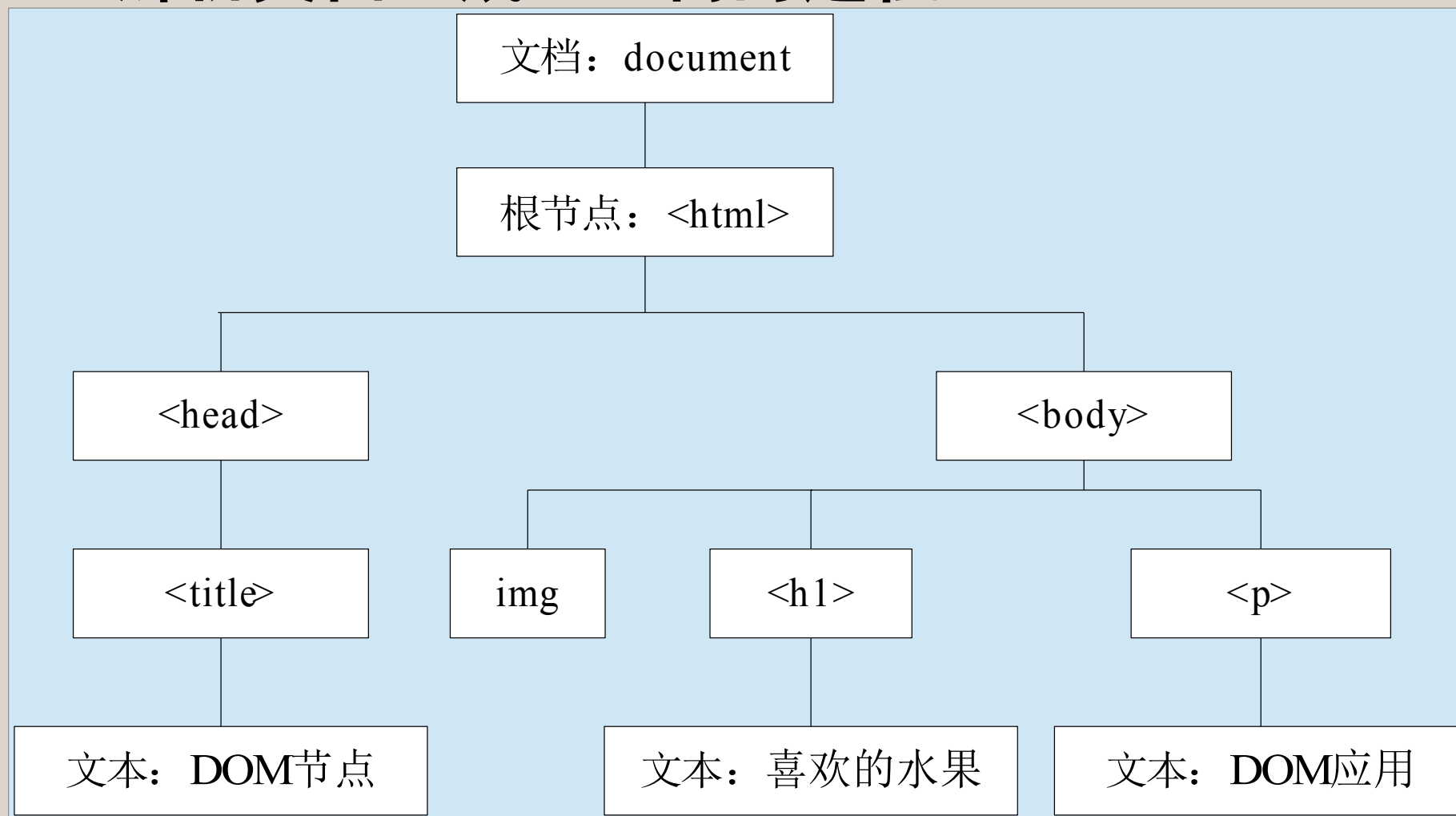
- 1、解析文档 (如 HTML) 并生成 DOM 树
- 2、通过 DOM 标准接口 + 编程语言改变文档内容



1.1 回顾 DOM 模型的相关概念



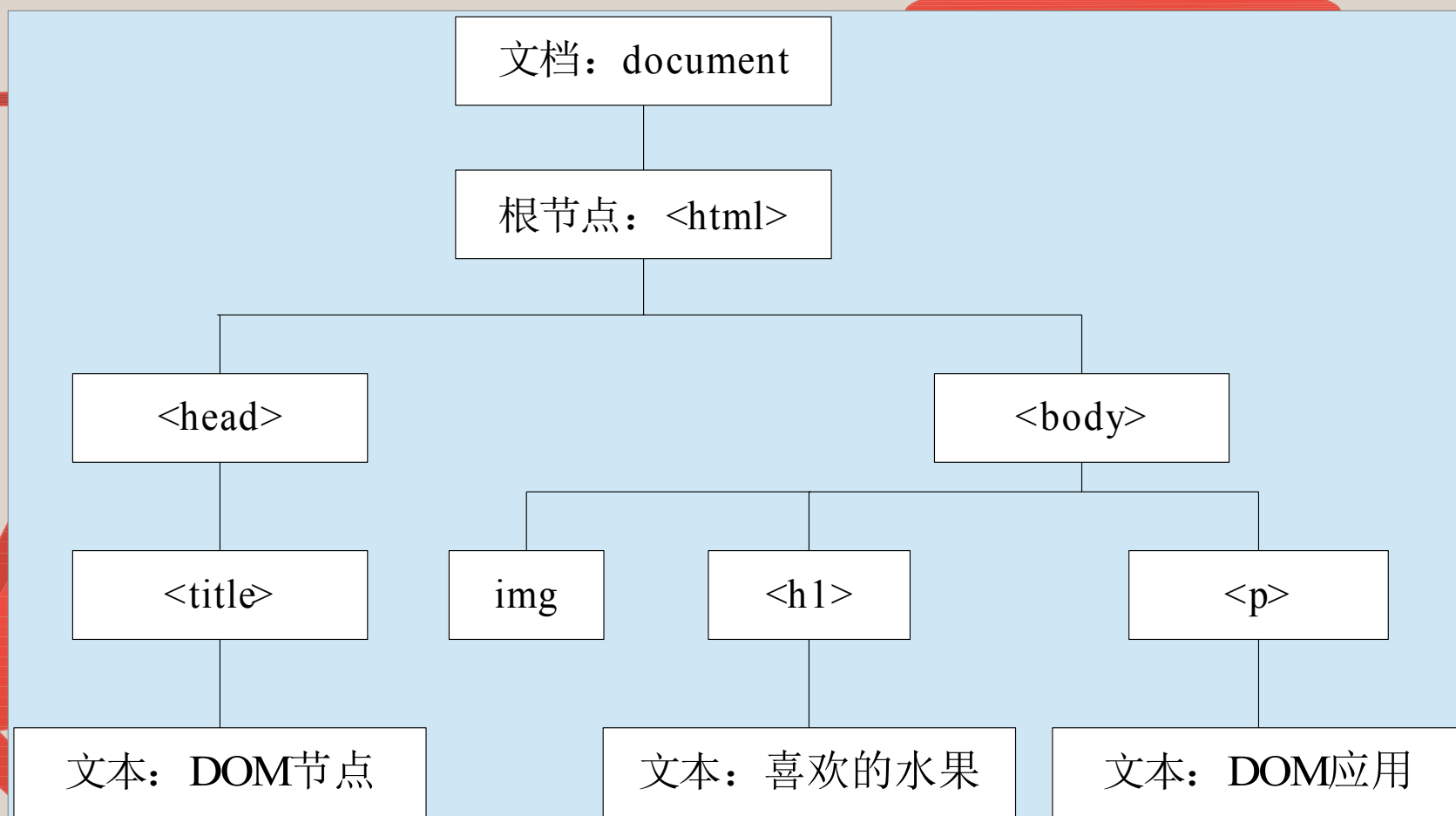
· 解析文档生成 DOM 树的过程



1.1 回顾 DOM 模型的相关概念



· DOM 树中的节点类型和节点关系



文本
节点

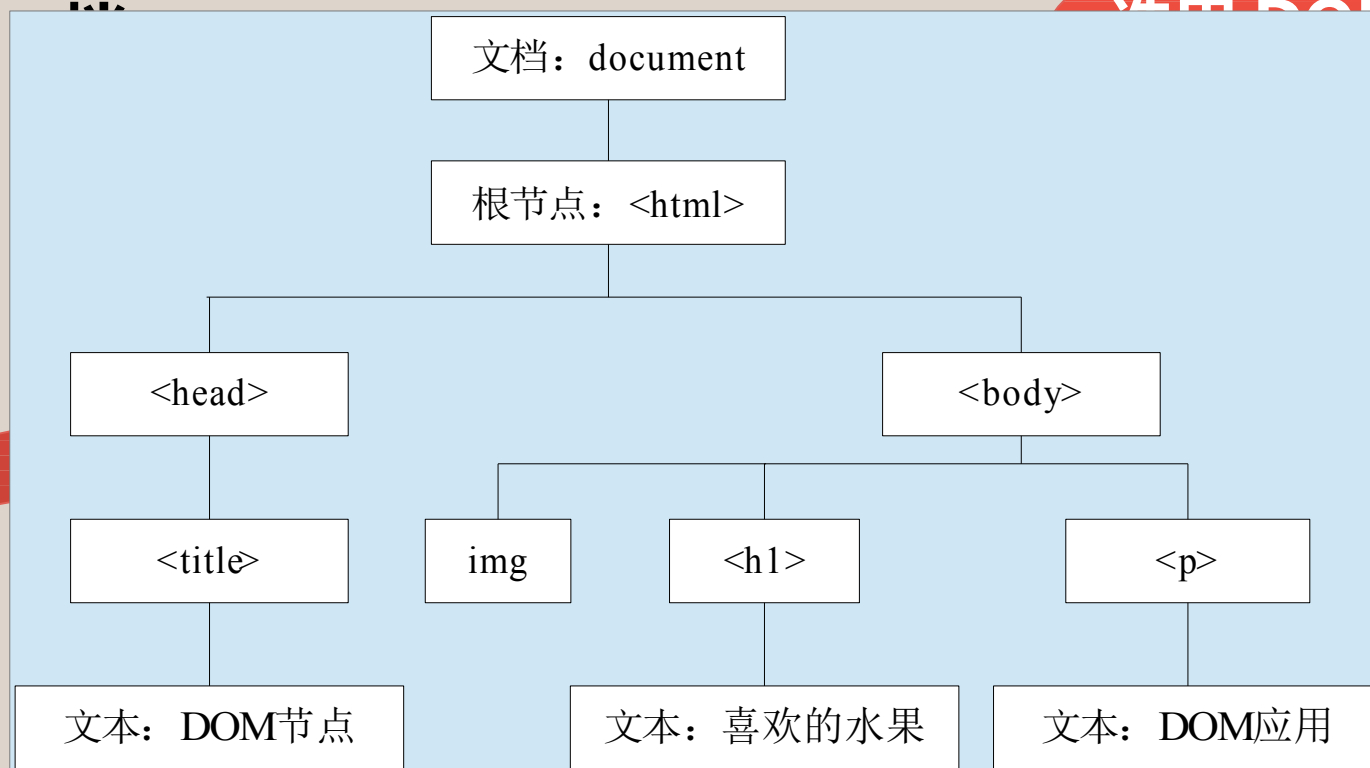


任务实现



W3C 规定的三类 DOM 标准接口

- Core DOM (核心 DOM), 适用于各种结构化文档
- XML DOM (Java OOP 学过), 专用于 XML 文档



选择 DOM 标准接口, 结合
编程语言访问 DOM

JavaScript、C#



任务 2：使用 Core DOM 方式动态添加表格



任务描述：

单击“增加一行”按钮，可以在书名为“60个瞬间”的所在行前插入一行，新插入行的价格一列居中显示。

单击“删除第2行”按钮可把书名为“看得见风景的房间”一行删除。

单击“修改标题样式”按钮，可以设置标题行居中显示，字体加粗，单元格的背景颜色变为“#cccccc”。

单击“复制最后一行”按钮，复制书名为“60个瞬间”所在行，并且追加到表格的末尾。



任务 2 ：使用 Core DOM 方式动态添加表格



任务分析：

完成该任务可以使用下列步骤：

- 1 . 在 `<head>` 标签对中嵌入 `<script></script>` 标签。
。
- 2 . 选择 JavaScript 语言编辑器编辑脚本代码。
- 3 . 嵌入脚本代码到 HTML 文档中。
- 4 . 使用浏览器浏览该 HTML 文档。



2.1 查看节点

- 访问指定节点的方法
 - `getElementById()` : 返回一个节点对象
 - `getElementsByName()` : 返回多个 (节点数组)
 - `getElementsByTagName()` : 返回多个 (节点数组)
- 查看 / 修改属性节点
 - `getAttribute(" 属性名 ")`
 - `setAttribute(" 属性名 ", " 属性值 ")`



```
function check() {  
    var favor=document.getElementsByTagName("input");  
    var like="你喜欢的水果";  
    for(var i=0;i<favor.length;i++){  
        if(favor[i].checked){  
            like+=favor[i].value+"\n";  
        }  
    }  
    alert(like);  
}  
  
function change(){  
    var imgs=document.getElementsByTagName("img");  
    imgs[0].setAttribute("src","images/grape.jpg");  
}
```



练一练



- 根据下图实现
- (1) 点击显示图片路径，弹出路径
- (2) 点击喜欢的水果，弹出选择后的水果。
- (3) 点击改变图片，图片变化





DOM 节点属性



DOM节点分为：文档节点、元素节点、属性节点和文本节点。

节点属性：

`nodeName`：节点名称。

`nodeValue`：节点的值。

`parentNode`：节点的父节点。每个元素、属性和文本都有一个父节点。

`childNodes`：节点的孩子节点列表。对于HTML，该列表仅对元素有意义，文本节点和属性节点都没有孩子。

`firstChild`：仅仅是`childNodes`列表中第一个节点的快捷方式。

`lastChild`：另一种快捷方式，表示`childNodes`列表中的最后一个节点。

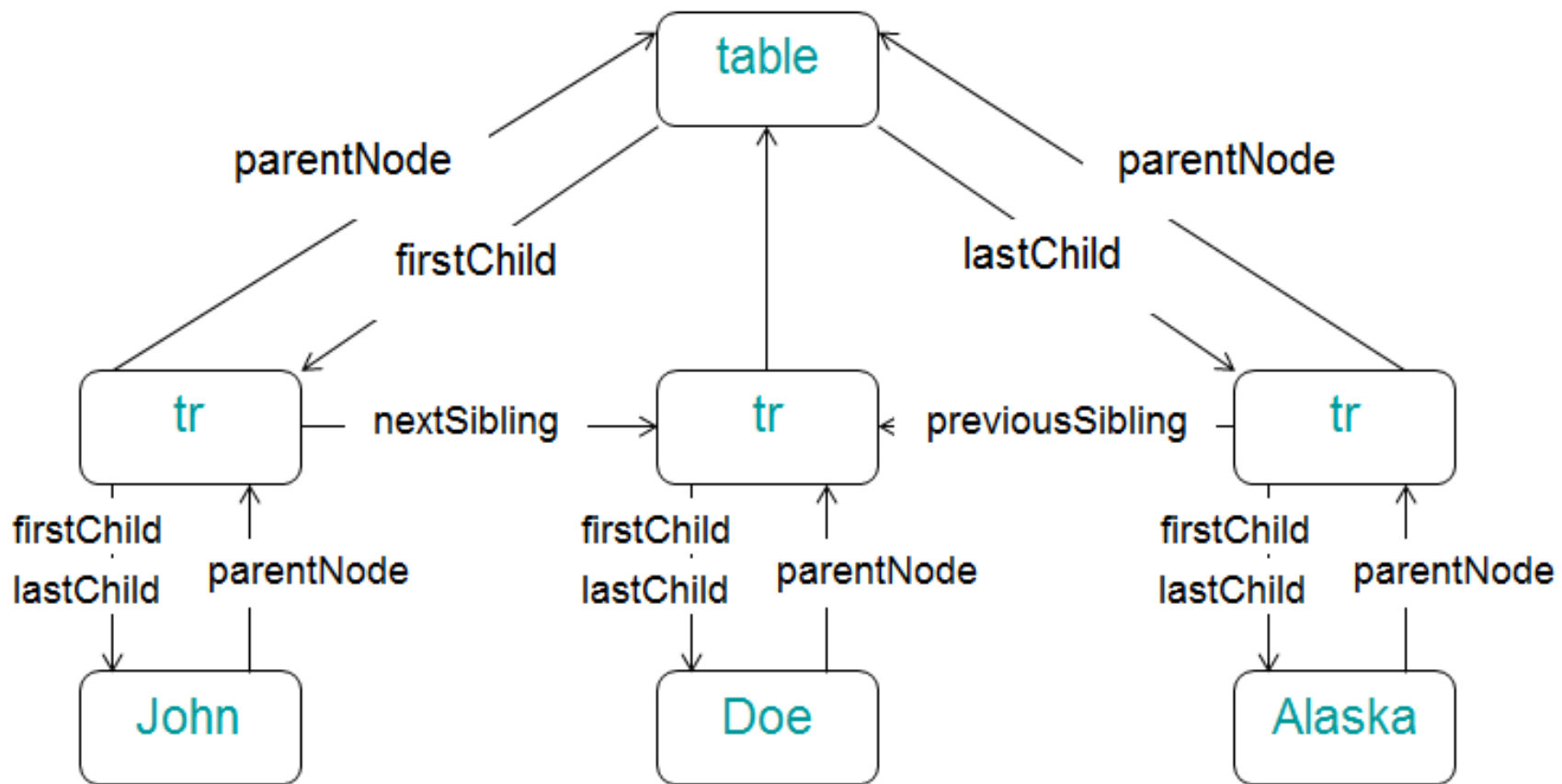
`previousSibling`：返回当前节点之前的节点。换句话说，它返回当前节点的父节点的 `childNodes` 列表中位于该节点前面的那个节点（如果感到迷惑，重新读前面一句）。

`nextSibling`：类似于`previousSibling`属性，返回父节点的`childNodes`列表中的下一个节点。

`attributes`：仅用于元素节点，返回元素的属性列表。



DOM 节点属性



2.1 查看节点



如果编程时希望访问某个元素的父元素，但又不知道父元素的 ID、name、tag，怎么办？

· 根据层次关系查找节点

- parentNode
- firstChild
- lastChild

该节点的
firstChild ?
lastChild ?

table >
<tr>

<td> 东方贺 <

<td> 语文 <

<td> 95 <

</tr>

</table>

该节点的
parentNode ?



节点列表 (nodeList)



- parentNode 、 firstChild 以及 lastChild
 - 这三个属性 parentNode 、 firstChild 以及 lastChild 可遵循文档的结构，在文档中进行“短距离的旅行”。
- 请看下面这个 HTML 片段：

```
<table id="tab1" width="200" border=1>
<tr>
<td>1</td> <td>2</td> <td>3</td> <td>4</td>
<tr>
</table>
```

以上 HTML 代码中，第一个 <td> 是 <tr> 元素的首个子元素 (firstChild)，

而最后一个 <td> 是 <tr> 元素的最后一个子元素 (lastChild)。

此外，<tr> 是各个 <td> 元素的父节点 (parentNode)。



2.1 查看节点

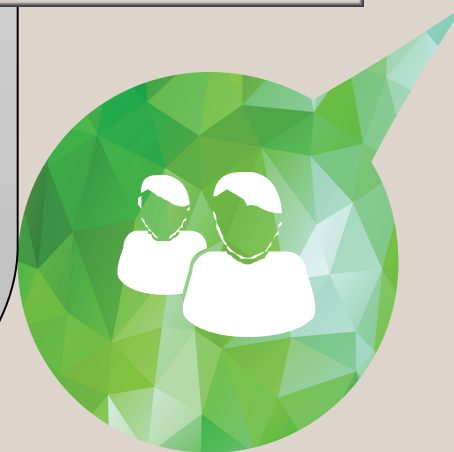
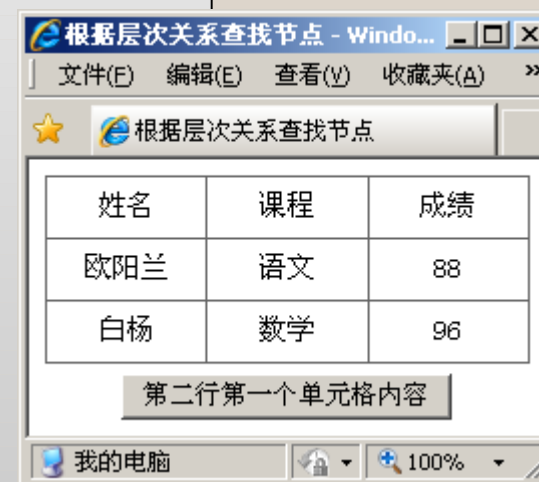


```
function checks(){  
    var objTable=document.getElementById("myTable");  
    var strHtml = objTable.lastChild.firstChild.firstChild.innerHTML;  
    alert(strHtml);  
}  
</script>
```

```
.....  
<table id="myTable">  
    <thead>  
        .....  
    </thead>  
    <tbody>  
        <tr>  
            <td> 欧阳兰 </td>  
            .....  
        </tr>  
    </tbody>  
</table>
```

该代码指哪个节点？

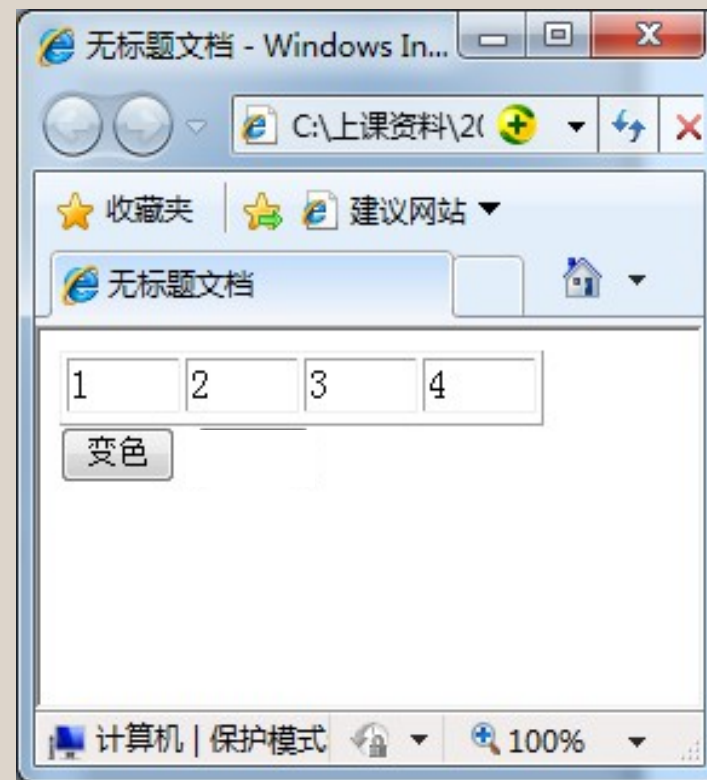
thead 和 tbody 标签用于表格数据的逻辑分组，W3C 不推荐，但写脚本访问表格数据时常用它们。



练习



- 实现修改表格特效
- 要求：
- （1）修改表格最后一个单元格的
- 文字颜色。
- （2）设置按钮 onclick 事件，
- 分别促发函数。

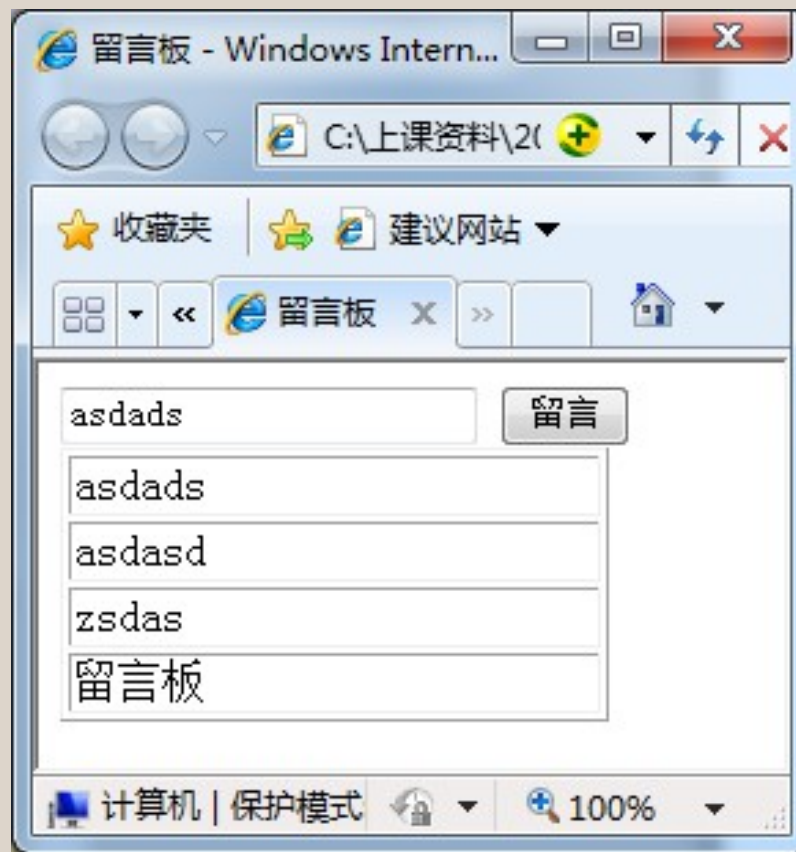


任务 2



· 留言版特效

- 1、找到文本框内容
- 2、定位留言版显示位置
- 3、添加留言内容



Document 对象常用方法



- Document 定义的方法大多数是生产型方法，
- 主要用于创建可以插入文档中的各种类型的节点。
- 常用的 Document 方法有：
- createAttribute(name) 用指定的名字创建新的 Attr 节点。
- createElement(name) 用指定的标记名创建新的 Element 节点。
- createTextNode(data) 用指定的文本创建新的 Text Node 节点。

```
//从最终的TextNode节点开始，慢慢形成<tbody>结构
//创建text节点
var text=document.createTextNode(a);
//创建td节点
var tda=document.createElement("td");
//创建tr节点
var tra=document.createElement("tr");
```



Node 对象



元素节点

属性节点

文本节点

```
<html>
  <head>
    <title> 购物网站 </title>
  </head>
  <body>
    <a href="href"> 我的链接 </a>
    <h1> 商品分类区 </h1>
  </body>
</html>
```



Node 对象常用方法



方法	描述
appendChild()	通过把一个节点增加到当前节点的 childNodes 组，给文档树增加节点。
cloneNode()	复制当前节点，或者复制当前节点以及它的所有子孙节点。（ ）中如果传递给它的参数是 true，它还将递归复制当前节点的所有子孙节点。否则，它只复制当前节点。
hasChildNodes()	如果当前节点拥有子节点，则将返回 true。
insertBefore()	给文档树插入一个节点，位置在当前节点的指定子节点之前。如果该节点已经存在，则删除之再插入到它的位置。
removeChild()	从文档树中删除并返回指定的子节点。
replaceChild()	从文档树中删除并返回指定的子节点，用另一个节点替换它。



Node 对象常用方法



- **appendChild(node)** 通过把一个节点增加到当前节点的 **childNodes[]** 组，给文档树增加节点。

参数	描述
<code>node</code>	必需。要追加的节点。

```
//从最终的TextNode节点开始，慢慢形成<tbody>结构
//创建text节点
var texta=document.createTextNode(a);
//创建td节点
var tda=document.createElement("td");
//创建tr节点
var tra=document.createElement("tr");
tda.appendChild(texta);
tra.appendChild(tda);
```





- **insertBefore(newchild,refchild)** 给文档树插入一个节点，位置在当前节点的指定子节点之前。
- 如果该节点已经存在，则删除之再插入到它的位置。

参数	描述
<code>newchild</code>	插入新的节点
<code>refchild</code>	在此节点前插入新节点

```
var liuyanban=document.getElementById("liuyan");  
liuyanban.insertBefore(tra,liuyanban.firstChild);
```



HTML DOM 任务 1



```
<head>
<script language="javascript">
function insertStr() {
  var f = document.form1;
  var value = f.str.value;
  if(value!="") {
    // 从最终的 TextNode 节点开始，慢慢形成 <tbody> 结构
    var text = document.createTextNode(value); // 新建一个 TextNode 节点
    var td = document.createElement("td"); // 新建一个 td 类型的 Element 节点
    var tr = document.createElement("tr"); // 新建一个 tr 类型的 Element 节点
    var tbody = document.createElement("tbody"); // 新建一个 tbody 类型的 Element 节点
    td.appendChild(text); // 将节点 text 加入 td 中
    tr.appendChild(td); // 将节点 td 加入 tr 中
    tbody.appendChild(tr); // 将节点 tr 加入 tbody 中
    var parNode = document.getElementById("table1"); // 定位到 table 上
    parNode.insertBefore(tbody,parNode.firstChild); // 将节点 tbody 插入到节点顶部
    //parNode.appendChild(tbody); // 将节点 tbody 加入节点尾部
  }
}
</script>
</head>
```



HTML DOM 任务 1



```
<body>
<form name="form1" method="post" action="">
  <input name="str" type="text" id="str" value="">
  <input name="insert" type="button" id="insert" value=" 留言 "
onClick="insertStr()">
</form>
<table width="400" border="1" cellspacing="0" cellpadding="0" id="table1">
<tbody>
  <tr>
    <td height="25"> 网友留言列表 : </td>
  </tr>
</tbody>
</table>
</body>
```



练习



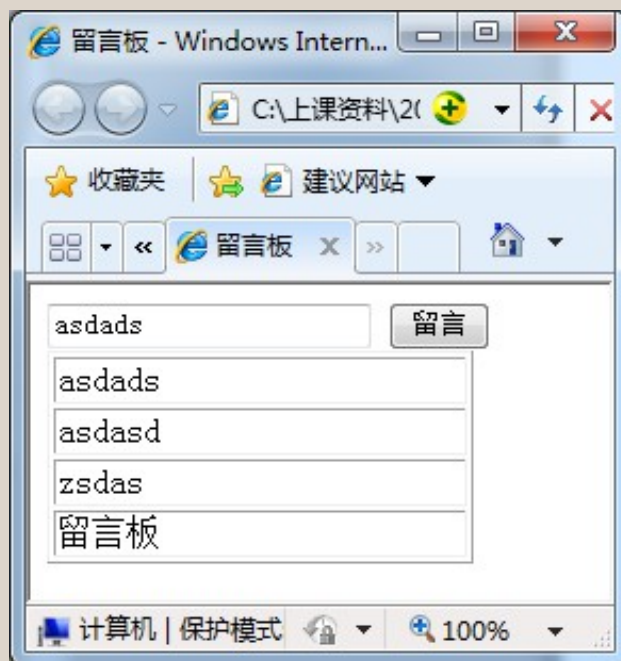
实现留言板特效

要求：（1）设置界面。

（2）获取文本框内容；

（3）创建各节点，插入节点。

（4）设置按钮实现点击后将留言加入到最前端。



创建和增加节点

创建和增加节点 的方法

createElement () : 创建节点

appendChild () : 末尾追加方式插入节点

insertBefore () : 在指定节点前插入新节点

cloneNode () : 克隆节点

```
function newNode(){  
    var oldNode=document.getElementById("sixty1");  
    var image=document.createElement("img");  
    image.setAttribute("src","images/newimg.jpg");  
    image.parentNode.insertBefore(image,oldNode);  
}
```

```
function copyNode(){  
    var image=document.getElementById("sixty1");  
    var copyImage=image.cloneNode(false);  
    image.parentNode.appendChild(copyImage);  
}
```

先创建个节点，
然后设置属性，
最后插入节点

ID 属性：sixty1

克隆图片节点，然后追加插入到文档



删除和替换节点



删除和替换节点的方法

removeChild () : 删除节点

replaceChild() : 替换节点

```
function delNode(){  
    var dNode=document.getElementById("sixty1");  
    dNode.parentNode.removeChild(dNode);  
}  
function repNode(){  
    var oldimage=document.getElementById("sixty1");  
    var newimage=document.createElement("img");  
    newimage.setAttribute("src","images/replace.jpg");  
    oldimage.parentNode.replaceChild(newimage,oldimage);  
}
```

先创建一个
节点，然后
替换原节点



ID 属性：
sixty1

练一练

实现图片特效

要求：（1）增加、复制图片。
（2）删除替换图片。



任务实现



操作表格 - Windows Internet Explorer

C:\Documents : +

文件(F) 编辑(E) 查看(V) 收藏夹(A) 帮助(H)

收藏夹 操作表格

书名	价格
幸福从手	¥18.50
一个瞬间	¥32.00
60个瞬间	¥32.00

增加一行 删除第2行 修改标题样式 删除最后一行

我的电脑 100%

修改的标题样式

增加的行



任务 3：使用 HTML DOM 方式动态添加表格



任务描述：

单击“增加订单”按钮插入一行订单，当多次单击“增加订单”按钮时，可增加多行。

单击“删除”按钮，可删除当前订单。

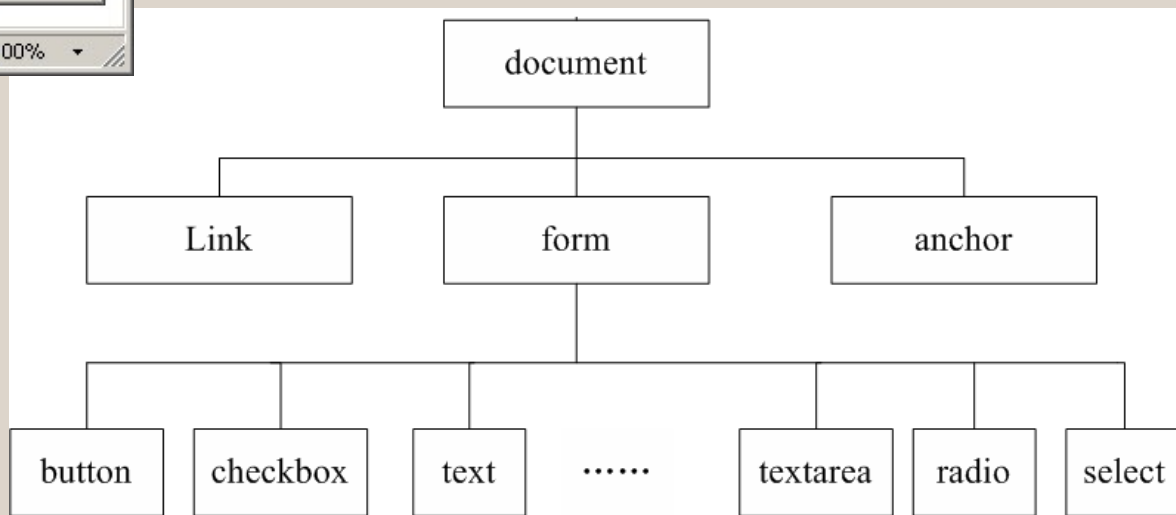


3.1 HTML DOM 的特有对象和操作

什么是 HTML DOM 对象



HTML 文档中的每个节点都是 DOM 对象，每类对象都有 1 套属性、方法。
最常用的是 `<table>` 表格和各类表单 `<form>` 元素对象



3.1 HTML DOM 的特有对象和操作



HTML DOM 对象 的属性访问

```
function change(){  
    var imgs=document.getElementById("s1");  
    imgs.src="images/grape.jpg";  
}  
  
function show(){  
    var hText=document.getElementById("s1").alt;  
    alert(" 图片的 alt 是 : "+hText)  
}  
</script>
```

不再使用
setAttribute () /
getAttribute () 方法
简化为对象名.属性值进
行读取或修改

书名	价格
看得见风景的房间	¥30.00
60个瞬间	¥32.00

增加一行 删除第2行 修改标题



3.2 HTML DOM 对象 -table 对象



· 思路分析

- 1、使用 Core DOM 标准操作获取表格对象
- 2、使用 HTML DOM 的 table 相关对象的属性、方法、事件操作表格



使用 HTML DOM
如何实现？



3.2.1 表格相关对象



table 表格对象

tableCell 单
元格对象

tableRow
表格行对象

书名	价格
看得见风景的房间	
泰晤士报	¥18.5
60个	¥32.00

增加一行 删除第2行 修改标题



3.2.1 表格相关对象



· table 表格对象

类别	名称	描述
属性	rows[]	返回包含表格中所有行的一个数组
方法	insertRow()	在表格中插入一个新行
	deleteRow()	从表格中删除一行

..... 代码片段.....

tableObj: <table>

tableObject.insertRow(index)

deleteRow(index)

.....

返回表格中所有行
,[] 中 0 表示返回
第一行

在表格指定位置插入
一行, 0 表示第一
行, -1 表示最后一
行

删除表格指定位置的
行, 0 表示第一行



3.2.1 表格相关对象



tableRow 表格行对象

类别	名称	描述
属性	cells[]	返回包含行中所有单元格的一个数组
	rowIndex	返回该行在表中的位置
方法	insertCell()	在一行中的指定位置插入一个空的 <td> 标签
	deleteCell()	删除行中指定的单元格

..... 代码片段.....

tableRowObject.insertCell(index);

tableRowObject.deleteCell(index);

.....

在表格一行指定位置插入
一个空的 <td>

删除表格一指定位置的
<td>



3.2.1 表格相关对象



· tableCell 单元格行对象

类别	名称	描述
属性	cellIndex	返回单元格在某行单元格集合中的位置
	innerHTML	设置或返回单元格的开始标签和结束标签之间的 HTML
	align	设置或返回单元格内部数据的水平排列方式
	className	设置或返回元素的 class 属性

..... 代码片段.....

```
cell2.innerHTML= " 改变单元格的 HTML 内容 ";  
cell2.align="center";
```

.....

用法示例



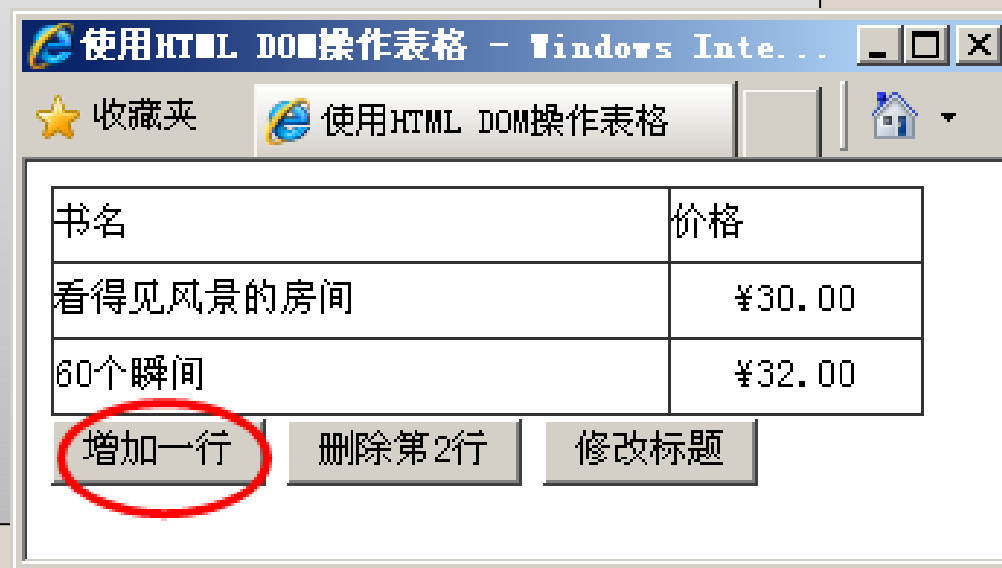
3.3 用 table 对象重新实现表格操作



实现“增加一行”的操作

增加一个行对象，然后插入单元格，最后设置单元格内容

```
function addRow(){  
    var newRow=document.getElementById("myTable").insertRow(2);  
    var col1=newRow.insertCell(0);  
    col1.innerHTML=" 幸福从天而降 ";  
    var col2=newRow.insertCell(1);  
    col2.innerHTML="¥18.5";  
    col2.align="center";  
}
```



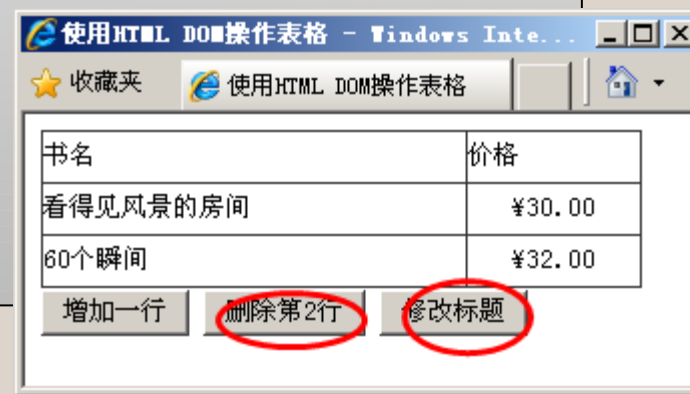
3.3 用 table 对象重新实现表格操作



实现“删除第2行”的操作
实现“修改标题”的操作

```
function delRow(){  
    document.getElementById("myTable").deleteRow(1);  
}  
  
function editRow(){  
    var uRow = document.getElementById("myTable").rows[0];  
    uRow.className="title";  
}
```

使用 class 样式属性直接修改标题样式



练一练



利用 HTML DOM 的方式实现下列实训

使用HTML DOM操作表格 - Windows Inte...

收藏夹 使用HTML DOM操作表格

书名	价格
看得见风景的房间	¥30.00
60个瞬间	¥32.00

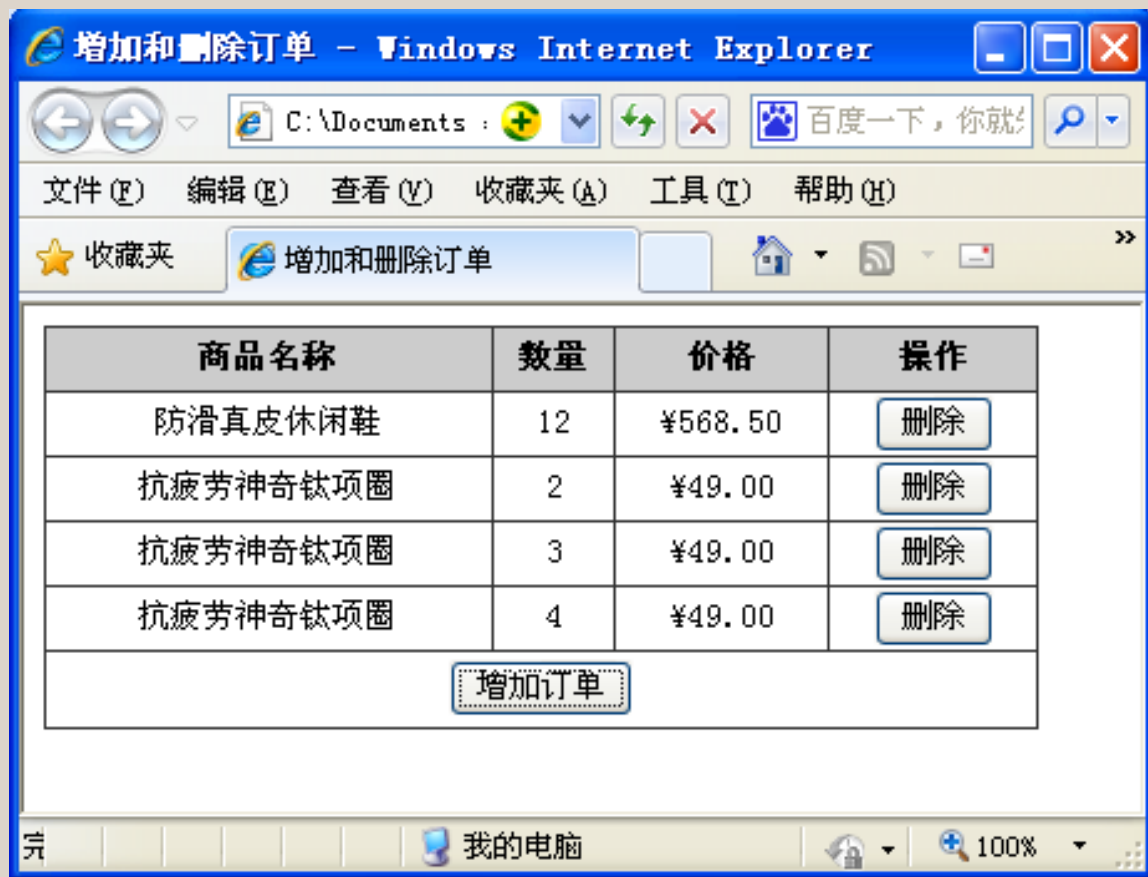
增加一行 删除第2行 修改标题



练一练



· 实现下列实训



任务实现



```
function addRow(){
    var addTable=document.getElementById("order");
    var row_index=addTable.rows.length-1;    // 新插入行在表格中的位置
    var newRow=addTable.insertRow(row_index); // 插入新行
    newRow.id="row"+row_index;                // 设置新插入行的 ID

    var col1=newRow.insertCell(0);
    col1.innerHTML=" 抗疲劳神奇钛项圈 ";

    var col2=newRow.insertCell(1);
    col2.innerHTML=row_index;

    var col3=newRow.insertCell(2);
    col3.innerHTML="¥49.00";

    var col4=newRow.insertCell(3);
    col4.innerHTML="<input name='del'+row_index+'\" type='button' value=' 删除'
onclick='\"delRow('row"+row_index+" ')\" />"; }
```



任务实现



```
function delRow(rowId){  
    var row=document.getElementById(rowId).rowIndex;  
    // 删除行所在表格中的位置  
    document.getElementById("order").deleteRow(row);  
}
```



总结



W3C 提供了 DOM 的哪三类标准？



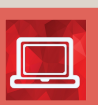
Core DOM 包括哪些常用的节点操作？

方法分别是什么？



Core DOM、HTML DOM 访问属性的方法

分别是什么？



HTML DOM 对象 -table 相关对象包括哪些？

分别包含哪些常用的属性、方法？



谢谢观看

