一、BOM

1. BOM 介绍

BOM全称为“Browser Object Model”，浏览器对象模型。提供一系列操作浏览器的属性和方法。核心对象为window对象，不需要手动创建，跟随网页运行自动产生，直接使用，在使用时可以省略书写。

2. window对象常用方法

1）网页弹框

```
alert()     //警告框
prompt()    //带输入框的弹框
confirm()   //确认框
```

2）窗口的打开和关闭

```
window.open("URL")  //新建窗口访问指定的URL
window.close()      //关闭当前窗口
```

3）定时器方法

1. 间歇调用(周期性定时器)
   作用：每隔一段时间就执行一次代码
   开启定时器:

```
var timerID = setInterval(function,interval);
/*
参数 :
 function : 需要执行的代码,可以传入函数名;或匿名函数
 interval : 时间间隔,默认以毫秒为单位 1s = 1000ms
返回值 : 返回定时器的ID,用于关闭定时器
*/
```

关闭定时器 :

```
//关闭指定id对应的定时器
clearInterval(timerID);
```

2. 超时调用(一次性定时器)
   作用：等待多久之后执行一次代码

```
//开启超时调用:
var timerId = setTimeout(function,timeout);
//关闭超时调用:
clearTimeout(timerId);
```

window 对象常用属性

window的大部分属性又是对象类型

1）history

作用：保存当前窗口所访问过的URL
属性 :

```
length 表示当前窗口访问过的URL数量
```

方法 :

```
back() 对应浏览器窗口的后退按钮,访问前一个记录
forward() 对应前进按钮,访问记录中的下一个URL
go(n) 参数为number值,翻阅几条历史记录， 正值表示前进,负值表示后退
```

2）location

作用：保存当前窗口的地址栏信息(URL)
属性：

```
href 设置或读取当前窗口的地址栏信息
```

方法：

```
reload(param) 重载页面(刷新)
参数为布尔值，默认为false，表示从缓存中加载，设置为true,强制从服务器根目录加载
```

3）document

提供操作文档HTML 文档的方法，,参见DOM

二、DOM节点操作

DOM全称为“Document Object Model”，文档对象模型，提供操作HTML文档的方法。（注：每个html文件在浏览器中都视为一篇文档,操作文档实际就是操作页面元素。）

1）节点对象

JS 会对html文档中的元素，属性，文本内容甚至注释进行封装，称为节点对象，提供相关的属性和方法。

2）常用节点分类

- 元素节点 （操作标签）
- 属性节点（操作标签属性）
- 文本节点（操作标签的文本内容）

3）获取元素节点

1. 根据标签名获取元素节点列表

```
var elems = document.getElementsByTagName("");
/*
参数 : 标签名
返回值 : 节点列表,需要从节点列表中获取具体的元素节点对象
*/
```

2. 根据class属性值获取元素节点列表

```
var elems = document.getElementsByClassName("");
/*
参数 : 类名(class属性值)
返回值 : 节点列表
*/
```

3. 根据id属性值取元素节点

```
var elem = document.getElementById("");
/*
参数 : id属性值
返回值 : 元素节点
*/
```

4. 根据name属性值取元素列表

```
var elems = document.getElementsByName("");
/*
参数 : name属性值
返回 : 节点列表
*/
```

4）操作元素内容

元素节点对象提供了以下属性来操作元素内容

```
innerHTML : 读取或设置元素文本内容,可识别标签语法
innerText : 设置元素文本内容,不能识别标签语法
value : 读取或设置表单控件的值
```

5）操作元素属性

1. 通过元素节点对象的方法操作标签属性

```
elem.getAttribute("attrname");//根据指定的属性名返回对应属性值
elem.setAttribute("attrname","value");//为元素添加属性,参数为属性名和属性值
elem.removeAttribute("attrname");//移除指定属性
```

2. 标签属性都是元素节点对象的属性,可以使用点语法访问，例如：

```
h1.id = "d1";         //set 方法
console.log(h1.id);  //get 方法
h1.id = null;        //remove 方法
```

注意：

- 属性值以字符串表示

- class属性需要更名为className,避免与关键字冲突,例如：
  h1.className = "c1 c2 c3";

6）操作元素样式

1. 为元素添加id，class属性，对应选择器样式
2. 操作元素的行内样式,访问元素节点的style属性，获取样式对象；样式对象中包含CSS属性，使用点语法操作。

```
p.style.color = "white";
p.style.width = "300px";
p.style.fontSize = "20px";
```

注意：

- 属性值以字符串形式给出,单位不能省略
- 如果css属性名包含连接符,使用JS访问时,一律去掉连接符,改为驼峰. font-size -> fontSize

7）元素节点的层次属性

1. parentNode
   获取父节点
2. childNodes
   获取子节点数组,只获取直接子节点(包含文本节点和元素节点)
3. children
   获取子节点数组,只获取直接子元素,不包含间接元素和文本节点
4. previousSibling
   获取前一个兄弟节点(文本节点也可以是兄弟节点)
   previousElementSibling 获取前一个元素兄弟节点
5. nextSibling
   获取后一个兄弟节点
   nextElementSibling 获取下一个元素兄弟节点
6. attributes
   获取属性节点的数组

8）节点的创建，添加和删除

1. 创建元素节点

```
var elem = document.createElement("标签名");//返回创建好的元素节点
```

2. 节点的添加
   添加和移除操作都必须由父元素执行，方法如下：

- 在父元素的末尾添加子节点

```
parendNode.appendChild(node);
```

- 指定位置添加

```
parendNode.insertBefore(newNode,oldNode);//在oldNode之前添加子节点
```

3. 移除节点

```
parentNode.removeChild(node);//移除指定节点对象
```

三、DOM 事件处理

事件：指用户的行为或元素的状态。由指定元素监听相关的事件，并且绑定事件处理函数。
事件处理函数：元素监听事件，并在事件发生时自动执行的操作。

1） 事件函数分类

1. 鼠标事件

```
onclick      //单击
ondblclick   //双击
onmouseover  //鼠标移入
onmouseout   //鼠标移出
onmousemove  //鼠标移动
```

2. 键盘事件

```
onkeydown    //键盘按键被按下
onkeyup      //键盘按键被抬起
onkeypress   //字符按键被按下
```

3. 文档或元素加载完毕

```
onload       //元素或文档加载完毕
```

4. 表单控件状态监听

```
onfocus      //文本框获取焦点
onblur       //文本框失去焦点
oninput      //实时监听输入
onchange     //两次输入内容发生变化时触发,或元素状态改变时触发
onsubmit     //form元素监听,点击提交按钮后触发,通过返回值控制数据是否可以发送给服务器
```

2）事件绑定方式

1. 内联方式
   将事件名称作为标签属性绑定到元素上
   例：

```
<button onclick="alert()">点击</button>
```

2. 动态绑定
   获取元素节点，动态添加事件
   例：

```
btn.onclick = function (){

};
```

3）事件函数使用

1. onload
   常用于等待文档加载完毕再进行下一步操作
2. 鼠标事件
3. 表单事件
   onchange： 监听输入框前后内容是否发生变化;也可以监听按钮的选中状态
   onsubmit： 表单元素负责监听,允许返回布尔值,表示数据是否可以发送;返回true,允许发送;返回false,不允许提交
4. 键盘事件