

课程回顾

1 Math+with()

2 Date

```
1  js提供方法:
2  toLocalString()
3  toLocalDateString()
4  toLocalTimeString()
5
6  js提供getter:
7  getFullYear()
8  getMonth()
9  getDate()
10 getDay()
11 getHours()
12 ...
```

3 String

```
1  substr()
2  subString()
3  细节:
4  没有equals()。比较两个字符串是否相等: ==
5  不区分大小写比较两个字符串是否向? ? toUpperCase() toLowerCase()
```

4 Array

```
1  类似java一样数组使用
2  循环方式:
3  普通for
4  for-in
5  foreach(fun:Function) =====
```

5 Function--函数

```
1  Function是一种数据类型
2  function 函数名(形参列表){
3      //函数体
4      //return 值;
5  }
6
7  调用:
8  var 变量=函数名(实参列表)
9  arguments模拟java的重载模式
```

课程目标

1 Function ==== 掌握

2 RegExp ==== 理解

3 DOM ==== 掌握 难点

4 常用事件 ==== 掌握

课程实施

1 Function函数

```
1  举例：
2  var arr=new Array(4);
3  //很少使用
4  var fun=new Function("形参列表", "函数体");
```

课堂案例：演示Function类型创建函数以及调用方式

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6      </head>
7
8      <script type="text/javascript">
9          //使用Function类型定义函数,缺点：阅读性差
10         var add=new Function("alert('hello');");
11         //调用Function对象
12         add();//调用无参无返回值
13         //使用Function类型定义带参有返回值的函数
14         var add2=new Function("a,b","return a+b;");
15         //调用Function对象
16         var sum=add2(12,12);
17         alert(sum);
18     </script>
19     <body>
20     </body>
21 </html>
```

1-1 匿名函数

特点：没有函数名。只能使用一次。

JavaScript中，匿名函数通常和DOM结合使用，完成标签事件的注册。

匿名函数的定义方式

```
1  function(){
2      //函数体
3      return 值;
4  }
```

2 事件

2-1 概念

事件是指：网页上面，用户发生的行为或动作

2-2 常见事件

- 1 load: 加载,
- 2 window.onload表示意思是整个html网页在浏览器完全加载完才会触发的一个事件。
- 3 click: 单击
- 4 focus: 获取焦点 举例: 文本框获取鼠标, 进入等待输入的状态
- 5 change: 用户选项发生改变触发。一般用于select使用

2-3 使用匿名函数给事件指定功能

给window添加onload监听器事件

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <title></title>
6   </head>
7
8   <script type="text/javascript">
9     //处理body的onload监听器, alert('')弹框
10    //window指html窗口
11    //
12    window.onload=function(){
13      //html网页在浏览器上已经完全加载完毕了
14      alert("aaa");
15    }
16  </script>
17  <body>
18  </body>
19 </html>
```

给按钮添加onclick监听器事件

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <title></title>
6   </head>
7
8   <script type="text/javascript">
9     //处理body的onload监听器, alert('')弹框
10    //window指html窗口
11    //
12    window.onload=function(){
13      //获取id=btn的标签对象
14      var aaa=document.getElementById("btn");
15      // aaa.onclick=function(){}等价于 onclick="add()"代码的效果
```

```

16         aaa.onclick=function(){
17             //按钮添加功能
18             alert("您单击了按钮");
19         }
20     }
21 </script>
22 <body>
23     <input id="btn" type="button" value="点击我看看"/>
24 </body>
25
26 </html>
27

```

3 DOM操作

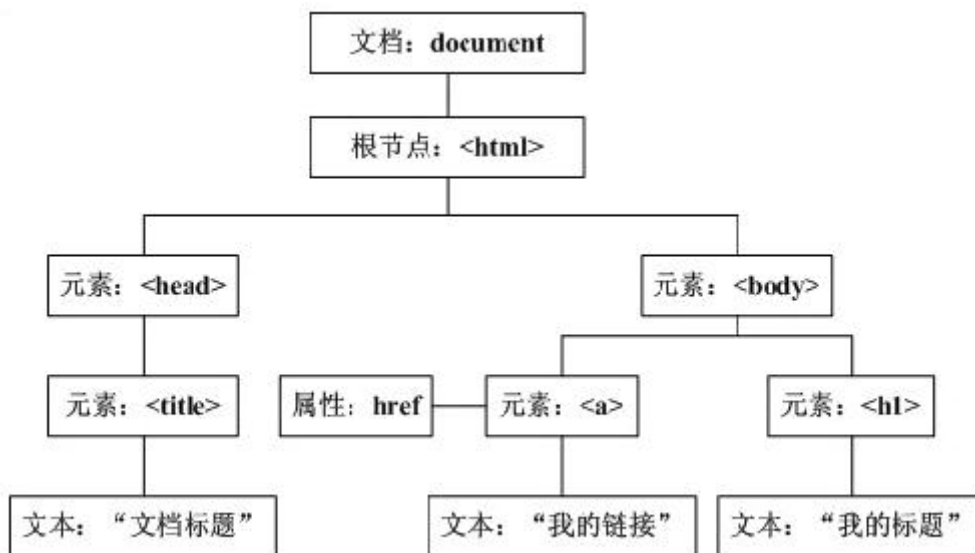
3-1 DOM概念和作用

1 | DOM: Document Object Model 文档对象模型

DOM作用

1 | 操作HTML文档上面所有的对象

3-2 DOM模型



1 | DOM模型将html网页以对象的模式进行管理的结构

3-4 DOM使用

获取对象的方式

```
1 根据标签的id属性获取唯一的一个html标签对象
2  getElementById(id:string):Element
3  根据标签的class属性获取一组html标签对象
4  getElementsByClassName(className:string):ElementList
5  根据标签名称获取一组html标签对象
6  getElementsByTagName(tagName:string):ElementList
7  根据name属性获取一组html标签对象
8  getElementsByName(name:String):ElementList
```

课堂案例：全选/全不选/反选

☐ 全选 ☒ 反选
☐ java ☐ html ☐ sql ☐ vue ☐ spring ☐ springboot ☐ springmvc

一段代码，希望代码根据用户操作的动作才执行。
分析三个术语：
事件源：发生用户行为的标签 全选复选框
事件：单击 click
功能：function 定义

注册事件监听器：将函数和事件绑定过程
on事件：监听器
标签.onclick=function()
}

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title></title>
6      </head>
7      <script type="text/javascript">
8          window.onload=function(){
9              //注册事件监听器
10             //1 获取事件源 全选按钮
11             var objCheckbox=document.getElementById("chkAll");
12             //2 注册事件监听器
13             objCheckbox.onclick=function(){
14                 //1.获取全选按钮的checked属性值
15                 // var strChecked=objCheckbox.checked;//objCheckbox是html上一个标
16                 //签，每个标签有哪些可以点击出来的属性：
17                 //2.获取所有的科目，将所有的科目的checked设置成第一步获取的属性值
18                 //2-1 获取所有的科目
19                 var arrCheckBox=document.getElementsByTagName("subject");
20                 //2-2 设置与全选checked属性值一样的
21                 for(var i=0;i<arrCheckBox.length;i++){
22                     arrCheckBox[i].checked=objCheckbox.checked;
23                 }
24             }
25             /**
26              * 反选功能：
27              * 1.获取事件源：用户操作的标签
28              * 2.确定事件：用户发生的行为（动作） 单击 click
29              * 3.注册事件监听器：
30              * 事件源.on事件=function(){
31              *
32              * }
```

```

33     //document:当前html文档对象
34     var objButton=document.getElementById("btnCheck");
35     //注册事件监听器
36     objButton.onclick=function(){
37         //事件处理功能: 函数体
38         for(var i=0;i<arrCheckBox.length;i++){
39             //设置反选
40             arrCheckBox[i].checked=!arrCheckBox[i].checked;
41         }
42     }
43 }
44 </script>
45 <body>
46     <input type="checkbox" id="chkAll"/>全选
47     <input type="button" value="反选" />
48     <br />
49     <!--
50         选中: checked=true 选中   checked=false 不选
51     -->
52     <input type="checkbox" name="subject" />java
53     <input type="checkbox" name="subject" />html
54     <input type="checkbox" name="subject" />sql
55     <input type="checkbox" name="subject" />vue
56     <input type="checkbox" name="subject" />spring
57     <input type="checkbox" name="subject" />springboot
58     <input type="checkbox" name="subject" />springmvc
59 </body>
60 </html>

```

使用DOM控制背景色

- 事件: focus
- 背景色控制: style属性对象

```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6      </head>
7      <script type="text/javascript">
8          window.onload=function(){
9              //1.文本注册focus事件, 设置文本框的背景色
10             var objText=document.getElementById("txt");
11             //注册事件监听器
12             objText.onfocus=function(){
13                 //监听器注册的匿名函数中, 通常事件源可以使用this指代
14                 //给文本框设置背景色style="background-color: gray;size:24px"
15                 this.style.backgroundColor="#ccc";
16             // alert(this);
17             }
18         }
19     </script>
20     <body>
21         <ul id="girlList">
22             <li>闭月</li>
23             <li>羞花</li>

```

```

24         <li>沉鱼</li>
25         <li>落雁</li>
26     </ul>
27     <input type="text" name="txt" id="txt" value="" />
28     <input type="button" class="btn" name="" id="btnReAdd" value="尾部追
加" />
29     <input type="button" class="btn" name="" id="btnAdd" value="前置插入"
/>
30     <input type="button" class="btn" name="" id="btnReplace" value="替换"
/>
31     <input type="button" class="btn" name="btnDel" id="btnDel" value="删
除" />
32     <input type="button" class="btn" name="" id="btnClone" value="克隆"
/>
33 </body>
34 </html>
35

```

新增对象

- appendChild()
- insertBefore()

```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6      </head>
7      <script type="text/javascript">
8          window.onload=function(){
9              //获取所有的案例
10             var arrButtons=document.getElementsByClassName("btn");
11             //尾部追加
12             arrButtons[0].onclick=function(){
13                 /**
14                  * 1.新增：尾部
15                  * 1-1 创建一个新的标签li createElement() createText()
16                  * 1-2 将新创建的li对象放置到ul的尾部
17                  */
18                 var objNewLi=document.createElement("li");
19
20                 //设置li内部的文本
21                 var objText=document.getElementById("txt");
22                 var strGir1=objText.value;
23
24                 var objNewText=document.createTextNode(strGir1);
25
26                 objNewLi.appendChild(objNewText);
27                 //放到html的ul尾部
28                 //父节点.添加子节点()
29                 //获取父节点
30                 var objParentUl=document.getElementById("girlList");
31                 //追加
32                 objParentUl.appendChild(objNewLi);
33             }
34         }

```

```

35     </script>
36     <body>
37         <ul id="girlList">
38             <li>闭月</li>
39             <li>羞花</li>
40             <li>沉鱼</li>
41             <li>落雁</li>
42         </ul>
43         <input type="text" name="txt" id="txt" value="" />
44         <input type="button" class="btn" name="" id="btnReAdd" value="尾部追
加" />
45         <input type="button" class="btn" name="" id="btnAdd" value="前置插入"
/>
46         <input type="button" class="btn" name="" id="btnReplace" value="替换"
/>
47         <input type="button" class="btn" name="btnDel" id="btnDel" value="删
除" />
48         <input type="button" class="btn" name="" id="btnClone" value="克隆"
/>
49     </body>
50 </html>
51

```

插入对象

- insertBefore()

```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6      </head>
7      <script type="text/javascript">
8          window.onload=function(){
9              //前置插入
10             arrButtons[1].onclick=function(){
11                 /**
12                  * 1.新增：在指定的节点前面插入
13                  * 1-1 创建一个新的标签li createElement() createText()
14                  * 1-2 ul.前置插入(new, old);
15                  */
16                 var objNewLi=document.createElement("li");
17
18                 //设置li内部的文本
19                 var objText=document.getElementById("txt");
20                 var strGirl=objText.value;
21
22                 var objNewText=document.createTextNode(strGirl);
23
24                 objNewLi.appendChild(objNewText);
25                 //放到html的ul尾部
26                 //父节点.添加子节点()
27                 //获取父节点
28                 var objParentUl=document.getElementById("girlList");
29                 //追加

```



```

30         //1-1 ul的子节点 childNodes获取指定元素的所有子节点，但是会包含空
    格，所以使用不方便
31         //推荐新方法：父节点.getElementsByTagName()
32         var
arrChildren=objParentUl.getElementsByTagName("li");//objParentUl.childNodes;
33         // alert(arrChildren.length);
34
35         objParentUl.insertBefore(objNewLi,arrChildren[1]);
36     }
37 }
38 </script>
39 <body>
40     <ul id="girlList">
41         <li>闭月</li>
42         <li>羞花</li>
43         <li>沉鱼</li>
44         <li>落雁</li>
45     </ul>
46     <input type="text" name="txt" id="txt" value="" />
47     <input type="button" class="btn" name="" id="btnReAdd" value="尾部追
加" />
48     <input type="button" class="btn" name="" id="btnAdd" value="前置插入"
/>
49     <input type="button" class="btn" name="" id="btnReplace" value="替换"
/>
50     <input type="button" class="btn" name="btnDel" id="btnDel" value="删
除" />
51     <input type="button" class="btn" name="" id="btnClone" value="克隆"
/>
52 </body>
53 </html>

```

替换对象

- replaceChild(new,old)

```

1 <!DOCTYPE html>
2 <html>
3     <head>
4         <meta charset="utf-8" />
5         <title></title>
6     </head>
7     <script type="text/javascript">
8         window.onload=function(){
9             //替换
10            arrButtons[2].onclick=function(){
11                /**
12                 * 替换：用新的节点替换旧的节点
13                 * 父节点.replaceChild();
14                 */
15                //获取父节点
16                var objParentUl=document.getElementById("girlList");
17                //追加
18                //1-1 ul的子节点 childNodes获取指定元素的所有子节点，但是会包含空
    格，所以使用不方便
19                //推荐新方法：父节点.getElementsByTagName()

```

```

20         var
arrChildren=objParentUl.getElementsByTagName("li");//objParentUl.childNodes;
21 //         alert(arrChildren.length);
22 //1.获取克隆节点 cloneNode(boolean): true:克隆当前节点及其后代
false:克隆当前节点 默认值false
23         var objDb1Li=arrChildren[1].cloneNode(true);//克隆第二个子节点
24
25         //3.克隆的节点，替换最后一个节点
26         objParentUl.replaceChild(objDb1Li,arrChildren[3]);
27     }
28 }
29 </script>
30 <body>
31     <ul id="girlList">
32         <li>闭月</li>
33         <li onclick="alert('aaaa');">羞花</li>
34         <li>沉鱼</li>
35         <li>落雁</li>
36     </ul>
37     <input type="text" name="txt" id="txt" value="" />
38     <input type="button" class="btn" name="" id="btnReAdd" value="尾部追
加" />
39     <input type="button" class="btn" name="" id="btnAdd" value="前置插入"
/>
40     <input type="button" class="btn" name="" id="btnReplace" value="替换"
/>
41     <input type="button" class="btn" name="btnDel" id="btnDel" value="删
除" />
42     <input type="button" class="btn" name="" id="btnClone" value="克隆"
/>
43 </body>
44 </html>

```

删除对象

- removeChild(要删除的子节点)

```

1 <!DOCTYPE html>
2 <html>
3     <head>
4         <meta charset="utf-8" />
5         <title></title>
6     </head>
7     <script type="text/javascript">
8         window.onload=function(){
9             //删除
10            arrButtons[3].onclick=function(){
11                /**
12                 * 删除：用新的节点替换旧的节点 重点
13                 * 父节点.removeChild();
14                 */
15                //获取父节点
16                var objParentUl=document.getElementById("girlList");
17
18                //删除所有的子节点：清空
19                //获取ul的所有的li，子节点：集合

```

```

20         var
arrChildren=objParentUl.getElementsByTagName("li");//objParentUl.childNodes;
21
22         /*for(var i=0;i<arrChildren.length;i++){
23             objParentUl.removeChild(arrChildren[i]);
24             i--;
25         }*/
26
27         while(arrChildren.length>0){
28             //删除第一个，第二个就变成了第一个
29             objParentUl.removeChild(arrChildren[0]);
30         }
31     }
32 }
33 </script>
34 <body>
35     <ul id="girlList">
36         <li>闭月</li>
37         <li onclick="alert('aaaa');">羞花</li>
38         <li>沉鱼</li>
39         <li>落雁</li>
40     </ul>
41     <input type="text" name="txt" id="txt" value="" />
42     <input type="button" class="btn" name="" id="btnReAdd" value="尾部追
加" />
43     <input type="button" class="btn" name="" id="btnAdd" value="前置插入"
/>
44     <input type="button" class="btn" name="" id="btnReplace" value="替换"
/>
45     <input type="button" class="btn" name="btnDel" id="btnDel" value="删
除" />
46     <input type="button" class="btn" name="" id="btnClone" value="克隆"
/>
47 </body>
48 </html>

```

克隆对象

- cloneNode(deep:boolean):默认false

```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8" />
5          <title></title>
6      </head>
7      <script type="text/javascript">
8          window.onload=function(){
9              //克隆
10             arrButtons[4].onclick=function(){
11                 //获取父节点
12                 var objParentUl=document.getElementById("girlList");
13                 //1. 获取克隆节点 cloneNode(boolean): true:克隆当前节点及其后代
false: 克隆当前节点 默认值false
14                 var objDbLi=arrChildren[1].cloneNode(true);//克隆第二个子节点
15                 //3. 在ul尾部追加克隆的节点
16                 objParentUl.appendChild(objDbLi);

```

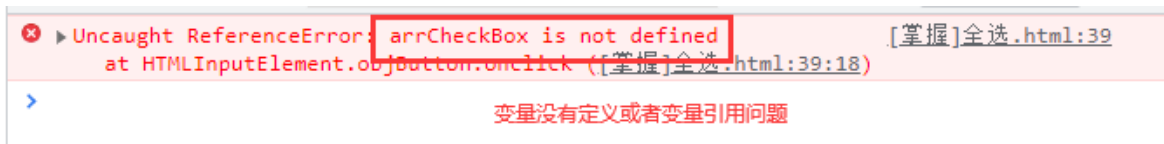
```

17     }
18     }
19 </script>
20 <body>
21     <ul id="girlList">
22         <li>闭月</li>
23         <li onclick="alert('aaaa');">羞花</li>
24         <li>沉鱼</li>
25         <li>落雁</li>
26     </ul>
27     <input type="text" name="txt" id="txt" value="" />
28     <input type="button" class="btn" name="" id="btnReAdd" value="尾部追
    加" />
29     <input type="button" class="btn" name="" id="btnAdd" value="前置插入"
    />
30     <input type="button" class="btn" name="" id="btnReplace" value="替换"
    />
31     <input type="button" class="btn" name="btnDel" id="btnDel" value="删
    除" />
32     <input type="button" class="btn" name="" id="btnClone" value="克隆"
    />
33 </body>
34 </html>

```

获取子对象的兼容性问题解决方案

- 1 父.childNodes: 会获取父节点内部的空白节点
- 2 父.getElementsByTagName("子节点的标签名"): 只获取父节点内部指定标签名的子节点



3-5 课堂案例：两级联动的数据加载

```

1 <!DOCTYPE html>
2 <html>
3     <head>
4         <meta charset="UTF-8">
5         <title></title>
6     </head>
7     <script type="text/javascript">
8         //定义数组保存所有的省
9         var province=["湖北省","河南省","湖南省","河北省"];
10        //使用省份名称为key，存入该省对应的市
11        province["湖北省"]=["武汉市","黄冈市","襄阳市","荆州市"];
12        province["河南省"]=["郑州市","信阳市","洛阳市","驻马店"];
13        province["湖南省"]=["长沙市","郴州市","岳阳市","衡阳市"];
14        province["河北省"]=["石家庄市","邯郸市","秦皇岛市"];
15        //一个格式，实现市和区的对应
16        province["武汉市"]=["洪山区","武昌区","汉口区"];
17        province["襄阳市"]=["襄州区","宜城市","谷城市"];
18        province["长沙市"]=["羊区","人区"];
19
20        //网页加载完毕，就加载所有的省份到select

```

```

21 window.onload=function(){
22     //获取省份的select
23     var objParentSelect=document.getElementById("selProvince");
24     //将数组中所有的省份追加到select中
25     /**
26      * 1.创建option createElement()
27      * 2.创建text createTextNode(数据从何而来???)
28      * 3.父.appendChild(子);
29      */
30     for(var i=0;i<province.length;i++){
31         var strProv=province[i];//获取省份名称
32         //获取省份，以option标签的形式，放在select列表显示
33         //1.创建option
34         var objNewOp=document.createElement("option");
35         //2.设置option内部显示的数据
36         var objText=document.createTextNode(strProv);
37         //3.将文本对象设置到option显示
38         objNewOp.appendChild(objText);
39
40         //4.将设置好的option，放到select显示
41
42         objParentSelect.appendChild(objNewOp);
43
44     }
45
46     /**
47      * 二级联动:
48      * 1.给省份下拉列表添加onchange
49      * 2.获取用户选择的省份对应的城市
50      * 3.创建option，显示城市，并将创建的option添加到市对应的select
51      */
52     objParentSelect.onchange=function(){
53         var objCitySel=document.getElementById("selCity");
54         //清空，保留第一项
55         var arrSon=objCitySel.getElementsByTagName("option");//获取
56         所有的城市
57         while(arrSon.length>1){
58             objCitySel.removeChild(arrSon[1]);
59         }
60
61         //获取用户选择的省份
62         alert("您选择的省份是："+this.value);//this指事件源 所有的表单项
63         //获取用户输入的或者选择的都是用value
64         var strProv=this.value;
65         //获取省份对应的城市
66         var arrCities=province[strProv];
67
68         for(var i=0;i<arrCities.length;i++){
69             var strCity=arrCities[i];//获取省份名称
70             //获取省份，以option标签的形式，放在select列表显示
71             //1.创建option
72             var objNewOp=document.createElement("option");
73             //2.设置option内部显示的数据
74             var objText=document.createTextNode(strCity);
75             //3.将文本对象设置到option显示
76             objNewOp.appendChild(objText);
77
78             //4.将设置好的option，放到select显示

```

```

77         objCitySel.appendChild(objNewOp);
78
79     }
80 }
81 }
82 </script>
83 <body>
84     省份:
85     <select id="selProvince">
86         <option>====请选择所在的省份====</option>
87     </select><br />
88     市: <select id="selCity">
89         <option>====请选择所在的市====</option>
90     </select><br />
91 </body>
92 </html>

```

课程总结

Function和function区别

```

1  Function: 一个类名
2  javascript创建函数的方式有两种:
3  第一种: 借助Function的构造方法完成
4  var 函数名=new Function(形参列表: string, 函数体:string);
5  调用:
6  var 返回值=函数名(实参列表);
7  缺点: 阅读性差, 很少使用
8
9  function: 一个函数定义关键字
10 第二种: 使用function关键词定义
11 function 函数名(形参列表){
12     函数体
13 }
14 调用:
15 var 返回值=函数名(实参列表);
16 优点: 阅读性好
17 匿名函数:
18 new Function();
19 function(){
20
21 }()

```

事件处理方式

```

1  1. 确认事件源
2  2. 确认事件
3  3. 注册事件监听器
4  事件源.on事件=function(){
5      //this获取当前事件源
6  }

```

DOM操作

```
1  获取类型方法：
2  getElementById()
3  getElementsByTagName()
4  getElementsByClassName()
5  getElementsByName()
6
7  追加、插入、替换、删除、克隆
```

预习安排

BOM: window history location

JQuery:

选择器: CSS

JQuery实现DOM操作

XML解析

mysql