H2 Day03_JavaScript

元素类型之间的转换

1.块元素 block 自动换行,具有宽度和高度样式,当脱离文档流块标签之间没有间距

2.行内元素 inline 不会自动换行,没有高度和宽度的样式

3.行内块 inline-block 不会自动换行,但是有高度和宽度样式,在标准文档流中,标签之间有一定间距

img,input

display:block|inline|inlin-block|none(隐藏)

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title></title>
        <style type="text/css">
            img{
                /* border: 1px solid red; */
                height: 200px;
                width: 200px;
                display: block;
            }
            input{
                height: 200px;
                width: 300px;
            }
            div{
                border: 1px solid red;
                height: 300px;
                width: 300px;
                display: inline-block;
            }
```

```
        </style>
    </head>
    <body>
        <img src="img/下载.jpg" ><img src="img/下
载.jpg" >
        <input type="button"/><input type="button"/>
        <div></div>
        <div></div>
    </body>
</html>
```

H3 今日目标

```
1.了解JavaScript
2.JavaScript的组成
3.JavaScript引入方式
4.基本语法
    变量
    数据类型
    数组
    函数
    分支结构
5.事件
6.JavaScript的DOM操作
```

H3 1.JavaScript简介

```
JavaScript是一个脚本语言,能够实现页面与用户的动态的交互,
通过使用JavaScript的中事件能实现对页面的上的各种操作，比
如页面表单中的格式校验...
编写JavaScript脚本的时候不能自动报错，因此我们需要借助浏
览器的工具去实现对脚本语言的检查。
打开浏览器(fn+f12或者f12-->console(控制台))
```

H3 2.JavaScript的组成

```
1.ECMAScript JavaScript的核心语法，函数，事件..
2.DOM  Document Object Model  文档对象模型
3.BOM  Browse Object Model  浏览器对象模型
```

H3 3.JavaScript的引入

```
1.行内式
 直接将js代码写入到标签内部
<input type="button" value="点击"
onclick="alert('Hello JavaScript')"/>
2.内联式
在页面添加一个<script></script>标签(<head></head>内部
或者<body></body>下方)
然后在<script></script>中编写js代码
3.外联式
  1.创建.js文件,在该文件中添加js代码
  2.通过<script src=""></script>引入外部js文件
<script src="js/demo.js" type="text/javascript">
</script>
```

```
<html>
    <head>
        <meta charset="utf-8">
        <title></title>
        <script type="text/javascript">
            function demo(){
                alert("Hello World");
            }
        </script>
    </head>
    <body>
        <input type="button" value="点击" onclick="alert('Hello JavaScript')"/>
        <input type="button" value="内联式" onclick="demo()"/>
    </body>
</html>
```

内敛引入方式

H3 4.ECMASCrip

```
##1.变量声明
```

```
var 变量名;
var 变量名=值;
var 变量名,变量名;
输出方式:
alert()弹出框
console.log():将结果输出在浏览器的控制台
document.write();将结果输出到页面上
##2.数据类型
  1.基本数据类型
    number:数字(整数和小数)
    undefined:未定义,没有值的变量都是未定义。该类型是从null中派生出来的。
    null:空。    undefined与null的值相同，但是类型不相同。
    boolean:true,false
    string:字符串,使用单引号和双引号都可以。
  2.引用数据类型
    Object:对象
    Date
    Array
    function
##数据类型详解
#string:
属性:
length 字符串长度
获取字符串中某个字符:变量名[下标值]
substring(indexA,indexB):截取字符串 indexA: 起始的下标
 indexB 结束的下标(取不到)
split():按照某个字符分隔,返回是数组
#日期
var date = new Date(); 获取当前系统时间
var d = new Date();
```

```
console.log(new Date());//获取当前系统时间
console.log(d.getTime());//获取系统时间的毫秒数
console.log(d.toLocaleDateString());//获取本地日期(年月日),字符串
console.log(d.toLocaleTimeString());//获取本地时间(时分秒),字符串
```

##数组

Array

特点:长度可以随意改变，类型可以随意填写

创建数组:

```
var arr = new Array([len]);  //len 表示数组的初始长度，可以省略
var arr=[值1,值2,值3....];
arr[4]=12;
arr[5]="hello";
console.log(arr.length);
```

push();将值添加到数组的末尾

join();返回的是一个字符串，数组中的值按照某个字符串成字符串

##函数

函数等价于java中的方法

```
function 函数名(){
    return xxx;
}
```

#无参无返回值函数

```
function demo(){
    console.log("无参无返回值函数");
}
```

```
#无参有返回值函数
function demo(){
    console.log("无参有返回值函数");
    return "hello";
}
#有参无返回值函数
function demo(a,b){
    console.log("有参无返回值函数:"+(a+b));
}
#有参有返回值函数
function demo(a,b){
    return a+b;
}
#调用
函数名([实参]);
##分支结构
 运算符号:
 1.算术符号
   +,-,*,/,%,+=,-=,*=,/=,%=,++,--
 2.关系运算符号
   >,>=,<,<=,==(值大小),!=,===(比较的是值大小和类型)
 3.逻辑运算符号
  && ||  !
 4.三则运算符
 表达式1?表达式2:表达式3
分支结构的用法和java完全一样
var score =60;
if(score>=60){
    console.log("及格");
}else{
    console.log("不及格");
}
```

H3 5.事件

事件:在页面的操作(动作,行为)
常用的事件:
1.onload 页面加载事件
  当页面结构及内容加载完成后，才去触发这个事件。
2.onclick 鼠标点击事件
3.onblur 失去焦点事件
4.onfocus 获取焦点事件
5.onchange 内容改变事件,该事件通常和select标签配合使用
6.onmouseover 鼠标移入事件
7.onmouseout 鼠标移出事件
8.onkeyup   键盘按钮松开事件
9.onkeydown   键盘按钮按下事件
#使用方式:

```
<script>
    function 函数名(){

}
</script>
<标签名    事件名="函数名()"></标签名>
```

1.获取对应的标签 obj
2.给该标签添加事件

```
obj.事件名=function(){

}
```

#以下代码演示事件触发的时机

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
```

```
        <title></title>
        <style type="text/css">
            div{
                border: 1px solid red;
                width: 300px;
                height: 300px;
            }
        </style>
        <script type="text/javascript">
            function demo(){
                console.log("事件触发了....");
            }
            window.onload=function(){
                //获取按钮标签
                var btn=
document.getElementById("d1");
                btn.onclick=function(){
                    console.log("第二个按钮点击事件触
发了...");
                }
            }
            function missFoc(){//失去焦点事件
                console.log("失去焦点事件触发了...");
            }
            function getFoc(){//获取焦点事件
                console.log("获取焦点事件触发了...");
            }
            function demo2(){
                console.log("内容改变事件触发
了....");
            }
            function demo3(){
```

```
                console.log("内容改变事件触发了...");
            }
            function demo4(){
                console.log("鼠标的移入事件触发
了...");
            }
            function demo5(){
                console.log("鼠标的移出事件触发
了...");
            }
            function demo6(){
                console.log("键盘摁下事件触发了...");
            }
            function demo7(){
                console.log("键盘松开事件触发了...");
            }
        </script>
    </head>
    <body>
        <input type="button" onclick="demo()"
value="方式一"/>
        <input type="button"  value="方式二"
id="d1"/>
        <input type="text" onblur="missFoc()"
onfocus="getFoc()" onchange="demo2()"/>
        <select name="" onchange="demo3()">
            <option value="">上海</option>
            <option value="">南京</option>
            <option value="">北京</option>
        </select>
        <div onmouseover="demo4()"
onmouseout="demo5()" >
```

```
        </div>
        <input type="text" onkeydown="demo6()"
onkeyup="demo7()"/>
    </body>
</html>
```

```
        <meta charset="utf-8">
        <title></title>
        <script type="text/javascript">
            function demo(){
                console.log("事件触发了....");
            }
            window.onload=function(){
                //获取按钮标签
                var btn= document.getElementById("d1");
                btn.onclick=function(){
                    console.log("第二个按钮点击事件触发了...");
                }
            }
        </script>
    </head>
    <body>
        <input type="button" onclick="demo()" value="方式一"/>
        <input type="button"  value="方式二" id="d1"/>
    </body>
```

6.DOM操作

Document Object Model 文本对象模型

将文档看做一个对象,文档中的标签也看成对象，并且将标签中的属性我们就可以看成对象的属性，因此使用属性的时候，就可以如下使用：对象名.属性名。,通过该原理实现对页面标签,内容,样式的增删改查操作。

#标签的通用属性

innerHTML:标签内容(包含标签及样式)

innerText:标签内容中的纯文本(不包含标签及样式)

#以上两个属性用在封闭标签

value:单标签(select标签也是),标签域中的标签,只要是标签中自带value属性的就可以使用这个属性,

标签的值，或者内容

##查询,获取标签对象方式

1.document.getElementById(); //根据标签中的id属性值获取该标签对象，能找到唯一个标签对象

2.document.getElementsByClassName();根据标签中的class属性值获取标签对象,能找到多个对象,因此返回值为数组对象

3.document.getElementsByTagName();根据标签名查找标签对象,能找到多个标签对象，因此返回值是数组对象

4.document.getElementsByName();根据标签中name属性值来找标签对象,能找到多个对象,因此返回值为数组对象

##修改

1.修改标签内容

使用以上的三个属性innerHTML,innerText,value

2.修改属性的值

对象名.属性=值;

3.修改样式

对象名.style.样式名=值;

当样式名中带有-的时候，将该符号省略并将该符号后边的第一个字母大写,比如:font-size fontSize

H3 ## 7.作业

1.表单中进行用户名,手机号,邮箱的不为空和格式校验，并将提示信息显示在标签后边

2.省市的二级联动,两个下拉选列表，前边显示省市，后边对应市区，当前边省市改变后，后边的列表显示对应的市区