DAY10 DOM 属性操作

2023年5月5日 10:33

操作CSS属性(样式)

1.读(获取)css属性的值 - getComputedStyle(元素对象)

```
// var box = document.querySelector('.box');//获取元素
// console.log(window.getComputedStyle(box).width); //100px
// console.log(window.getComputedStyle(box)['height']); //200px
```

如果获取数字格式

```
// console.log(parseInt(window.getComputedStyle(box).width)); //100
```

注意如果属性名包含中杠(-),必须改写成小驼峰命名或者中括号获取

```
// console.log(window.getComputedStyle(box).fontSize);//12px
// console.log(window.getComputedStyle(box)['font-size']);//12px
//
console.log(window.getComputedStyle(box).backgroundColor);//rgb(255,
0, 0)
```

2.写(设置)css属性的值 - 三种方式

```
var box = document.querySelector('.box');//获取元素
```

2.1.逐个属性设置

```
// box.style.width = '300px'; //注意单位
// box.style.height = '300px';
// box.style.backgroundColor = 'orange';
```

2.2.整体设置

将css选择器里面的代码以css文本的形式添加

```
// box.style = 'width:200px;height:200px;background-color:blue;';
box.style.cssText = `
                width:200px;
                height:200px;
                background-color:red;
            `;
```

2.3.添加选择器

```
box.id = 'header';
```

操作元素的类名和classList

一.操作元素的类名

ES提供类的概念->生成对象->(属性和方法)

html提供类的概念->类选择器->给元素添加样式的

class提供给ES约定的类

className提供给html约定的类选择器

总结：操作标签里面的类名使用className,操作ES里面的类(构造函数)使用class

```
<div class="box" id="header"></div>
<div class="list list1 list2">list</div>
```

获取元素

```
// var box = document.querySelector('.box');
// console.log(box.id);//header
// console.log(box.class);//undefined,不存在这个属性
// console.log(box.className);//box
// box.className = 'box1';//赋值新的类名
// box.className += ' box1';//迭代新的类名
```

二.classList

每一个元素身上天生自带一个属性叫做 classList, 记录了该元素的所有类名

```
// var list = document.querySelector('.list');
// console.log(list.classList);//类名列表
// console.log(list.classList.length);//2
// console.log(list.classList[0]);//list
// console.log(list.classList[1]);//list1
```

object.classList属性会继续返回对象，继续使用提供的方法操作class

1.add():追加类名(获取原有的类名添加上当前的类名)

```
// var list = document.querySelector('.list');
// list.classList.add('box1');
// list.classList.add('box2');
// list.classList.add('box3');
// list.classList.add('box4');
```

2.remove():删除对应的类

```
// list.classList.remove('list1');
// list.classList.remove('list2');
```

3.toggle():如果存在某个类名将其删除，不存在添加这个类名

```
// list.classList.toggle('hehe');

// 案例：显示隐藏
// var btn = document.querySelector('button');//获取按钮
// var main = document.querySelector('.main');//获取盒子
// btn.onclick = function () {//添加事件
//   main.classList.toggle('show');
// }

<!-- 显示隐藏某个div-->
<button>显示隐藏</button>
<div class="main"></div>

  .main {
    width: 200px;
    height: 300px;
    background-color: orange;
    display: none;
    /*隐藏*/
  }
  .show {
    display: block;
    /*显示*/
  }
```

元素的属性操作

一.元素的属性介绍

解读：标签里面除了标签名，<>括号里面的其他内容都可以称之为属性

比如：a标签是元素，href,title,target就是属性

元素的属性划分：

默认属性：标签元素自带的属性

自定义属性：用户自定义的属性，不是标签自带的

二.元素的属性操作

1.原生对象的属性操作：利用JavaScript对象操作方式(点符号和中括号)

```
<a href="#" title="标题" target="_blank" username="zhangsan">链接的文本
</a>
```

1.1.读：获取属性值

```
// var link = document.querySelector('a');//获取元素对象
// console.log(link.title);//标题
// console.log(link['title']);//标题
// console.log(link.target);//_blank
```

注意：自定义的属性，而且是获取之前就存在元素对象中(一开始写死在页面结构中)

```
// console.log(link.username);//输出空白
// console.log(link.hehe);//输出undefined
```

1.2.写：设置属性名以及属性值

```
// var link = document.querySelector('a');//获取元素对象
// link.title = '我是修改后的标题内容';
// link.href = 'http://www.baidu.com';
```

注意：如果通过点或者中括号设置的自定义属性，通过点或者中括号获取 - 重点

```
// link.hehe = '250';//设置属性，结构中不可见
// console.log(link.hehe);//250  可以获取
```

总结：如果自定义属性以及值，在一开始写在了标签结构中，点或者中括号就无法获取，其他的情况都可以

2.利用系统提供的方法进行属性操作

```
// var link = document.querySelector('a');//获取元素对象
```

2.1.getAttribute() 获取特定元素节点属性的值,包括自定义属性和默认属性

```
// console.log(link.getAttribute('title'));//标题
// console.log(link.getAttribute('username'));//zhangsan
```

2.2.setAttribute() 设置特定元素节点属性的值, 包括自定义属性和默认属性

```
// link.setAttribute('age', '18岁');//设置属性，结构中可见
// link.setAttribute('title', '新的标题');
```

2.3.removeAttribute() 移除特定元素节点属性, 包括自定义属性和默认属性

```
// link.removeAttribute('title');
// link.removeAttribute('username');
```

3.H5自定义属性dataset操作

html5中我们可以使用data-前缀设置我们需要的自定义属性,来进行一些数据的存放,这里的data-前缀就被称为data属性,数量不受限制,

可以通过dataset属性直接进行操作.

```
// 对比上面的两种操作方式
var box = document.querySelector('.box');//获取元素
// 利用对象方式
console.log(box.dataName);//undefined  无法获取
// 利用方法
console.log(box.getAttribute('data-name'));//zhangsan
// 利用dataset属性操作
console.log(box.dataset);//data-开头的属性集合，放在一个对象中。{name: 'zhangsan', age: '18', sex: '男'}
console.log(box.dataset.name);//zhangsan
console.log(box.dataset.age);//18
console.log(box.dataset.sex);//男
box.dataset.sex = '女';//设置
```

读写元素内部的结构内容

一.读写元素内部的结构内容

var box = document.querySelector('.box');

1.innerHTML : 读写元素节点里的内容，从节点起始位置到终止位置全部内容,包括内部的元素。

console.log(box.innerHTML);

2.outerHTML : 读写元素节点里的内容，除了包含innerHTML全部内容外, 还包含元

素节点本身。

```
console.log(box.outerHTML);
```

3.innerText : 读写某个元素节点的文本内容。

```
console.log(box.innerText);
```

二.重点关注innerHTML

```
box.innerHTML = '<ul><li>66666</li></ul>'; //赋值
box.innerHTML += '<ul><li>66666</li></ul>'; //迭代，追加
```

注意：innerHTML是渲染的核心，区别=和+=