JS第二周第四天
1.sort与DOM映射
2.DOM
3.DOM(1)

JS第二周第四天

1.sort与DOM映射

```
<body>
<ul id="list">
    <li>3</li>
    <li>1</li>
    <li>6</li>
    <li>2</li>
</ul>
<button id="btn">排序</button>
<!--
DOM 映射: 同过DOM提供的方法获取的元素跟页面存在着一一对应的关系(DOM 映射)
注意:就算将元素放在一个数组中这个关系依然存在


-->
<script>


  let list = document.getElementById("list");
  let oLis = list.getElementsByTagName("li");
  //let ary=Array.from(oLis);
  let ary = [...oLis];

  list.innerHTML='';
  //ary  放着所有的li
  ary.sort(function (a, b) {
    //a,b 都是元素 根据元素中的内容排序
    return a.innerHTML - b.innerHTML;
  });

  console.log(ary);
  //此时数组ary已经排好序了 但是页面上的内容没有变化
  //    for (let i=0;i<ary.length;i++){
  //      oLis[i].innerHTML=ary[i].innerHTML;
  //     //debugger;
  //    }
```

```
    //    console.log(ary);

    //不能修改标签中的内容 只能重新创建标签放在页面上

    let str = ``;
    for (var i = 0; i < ary.length; i++) {
      str+= `<li>${ary[i].innerHTML}</li>`;
    }
    list.innerHTML=str;
</script>
</body>
</html>
<script>
    /*
let ary=[3,1,45,6,8,2];
ary.sort(function (a,b) {
//a,b是数组中的某一项
//根据 a-b的值是正值还是负值 如果是正值就去交换位置
//return后面不一定 a-b或者是b-a    其实他要的是一个确定的数字正或者负
return a-b;
});
ary=[1,3,2];
ary.sort(function () {
return 1;
});
console.log(ary);


ary=["a","c","b","f","d"];

//按照字母表去排序 如果a.localeCompare(b) 中a在b的前面 返回-1 反之返回1
console.log("a".localeCompare("b"));//-1
console.log("fg".localeCompare("fa"));//1
console.log("A".localeCompare("b"));//-1

//按照

//12345
//1 5   1-5 负数
//3 1   3-1 正
//d f   d-f 负数
//g-a  正

ary.sort(function (a,b) {
return a.localeCompare(b);
});
console.log(ary);
```

```
ary=[{name:"A",age:13},{name:"C",age:20},{name:"B",age:10}];
ary.sort(function (a,b) {
//a,b 都是对象 想要根据age属性排序
//return a.age-b.age;
return a.name.localeCompare(b.name)
})
console.log(ary);
*/
</script>
```

2.DOM

```
<body>
<ul class="ul" id="list">
    <!--注释-->
    <li>我是第一个孩子</li>
    <li>我是第二个孩子</li>
</ul>
<div id="box">
    <p></p>
    <p></p>
    <span></span>
    <i></i>
    <h2></h2>
</div>
</body>
</html>
<script>
    //1.获取元素
    // document.getElementById() 一个
    // 上下文.getElementsByClassName()  集合
    // 上下文.getElementsByTagName()   集合
    // document.getElementsByName()  集合

    // let ul=document.getElementsByClassName("ul");
    // console.log(ul.getElementsByTagName);//undefined
    // let list=ul.getElementsByTagName("li");

    //节点:出现在文档中的任何东西都是节点:元素节点(标签),文档节点(documen
t),文本节点(text),注释节点(comment)....

    let list=document.getElementById("list");
    //1.获取子节点 childNodes 类数组
    console.log(list.childNodes);
    /*
```

```
                    元素节点        文本节点          注释节点          文档节点
    nodeName        大写标签名      #text             #comment          #docu
ment
    nodeType        1               3                 8                 9
    nodeValue       null            文本内容          注释的内容        null
    */

    let list1=list.childNodes;
    //找出所有的元素子节点
    let list2=[...list1].filter(function (item) {
      return item.nodeType==1
    });
    console.log(list2);

    //2.获取元素子节点   元素集合类数组
    console.log(list.children);

    //写一个方法 获取指定标签名的元素子节点
    //既然是一个方法 那么别人去使用的时候需要知道 获取"谁"下面的  "标签名是
啥"的子元素

    function getChild(curEle,tagName) {
      //curEle :获取谁下面的?
      //tagName:标签名叫啥
      //准备一个数组存放返回值的
      let child=[];
      //先获取curEle下面的所有元素子节点,顺便将其变成数组
      let kids=[...curEle.children];
      //接下来从kids中找出 所有的标签名是tagName的  注意nodeName 是大写标签
名而函数执行的是时候传进来的tagName不一定是大写的
      child=kids.filter(function (item) {
        //留下的是 当前item的nodeName==tagName.toUpperCase()
        return item.nodeName==tagName.toUpperCase();
      });
      //最后将结果返回
      return child;
    }

    console.log(getChild(box, "p"));

    function getChild(curEle,tagName) {
      return [...curEle.children].filter(function (item) {
        return item.nodeName==tagName.toUpperCase();
      });
    }

    //firstChild:获取第一个子节点
    //lastChild:获取最后一个子节点
    console.log(list.firstChild);
```

```
    console.log(list.lastChild);
    //firstElementChild:获取一个元素子节点
    //lastElementChild:获取最后一个元素子节点
    console.log(list.firstElementChild);
    console.log(list.lastElementChild);

    //previousSibling 获取上一个哥哥节点
    //previousElementSibling 获取上一个哥哥元素节点
    console.log(box.previousSibling);
    console.log(box.previousElementSibling);

    //nextSibling:获取下一个弟弟节点
    //nextElementSibling:获取下一个弟弟元素节点
    console.log(box.nextSibling);
    console.log(box.nextElementSibling);

    //parentNode:获取父亲节点 是元素节点
    console.log(box.parentNode);

    //没有就是null
    console.log(list.previousElementSibling);

    //写一个方法获取一个元素的所有的哥哥元素
    //写一个方法获取一个元素的指定标签名的哥哥元素
    //写一个方法得到一个元素的索引

</script>
```

3.DOM(1)

```
<script>
  let B = document.getElementById("B");
  //1.动态创建一个标签
  //document.createElement("标签名");
  let box = document.createElement("div");
  //给元设置一些属性
  box.id = "box";
  box.className = "box";
  box.innerHTML = "我是一个box";

  //2.元素1.appendChild(元素2) 将元素2当做元素1的最后一个孩子
  B.appendChild(box);

  //特殊 创建一个img
  let img = new Image();
  img.src = "img/1.jpg";
```

```
    console.log(img);
    B.appendChild(img);


    //3.删除一个元素
    // 父级元素.removeChild(元素)
    B.removeChild(img);


    //4.克隆一个元素
    //元素.cloneNode()  只克隆当前元素本身
    //元素.cloneNode(true)  将元素的内容一并克隆
    let box1 = B.cloneNode(true);
    console.log(box1);

    //5.插入元素
    //父级元素.insertBefore(元素1,元素2)
    //将元素1插入到元素2的前面
    //如果插入的是页面上已经存在的元素只是位置的改变

    let p1 = B.getElementsByClassName("p1")[0];
    let p2 = B.getElementsByClassName("p2")[0];

    B.insertBefore(p2, p1);
    //自己写一个方法,往一个元素的后面插入一个元素
    //就相当给他的弟弟的前面插入  没有弟弟 实际上就是给父级元素的最后面加一个
    function insertAfter(curEle, ele) {
      //获取弟弟
      let next = curEle.nextElementSibling;
      //判断弟弟有没有
      if (next) {
        curEle.parentNode.insertBefore(ele, next)
      } else {
        //没有弟弟 说明大当期元素是最后一个 直接给当前元素的父级的最后面添加一个
元素 使用 appendChild即可
        curEle.parentNode.appendChild(ele);
      }
    }


    console.log(B.className);//"box1 box2 box3 box4"
    console.log(B.classList);
    //classList 获取元素的类名集合
    //add     增加类名  之前有了就默认不加 增加多个使用逗号隔开即可
    //remove  删除类名
    B.classList.add("box5");
    B.classList.add("box1", "box6");
    B.classList.remove("box1", "box2");
```

```
    //以后关于类名的设置 如果类名多余2个了使用classList 少的话直接使用className赋值即可

</script>
```