web API 第二天

一.操作节点对象的基本属性

1.1 操作节点对象的表单属性 【重点】

- 表单属性
  - 设置和获取节点对象的表单属性的方式

```
//设置节点对象的表单属性
节点对象.属性 = 值;
//获取节点对象的表单属性
节点对象.属性;
```

  - 常见的表单属性
    - 节点对象.value;【所有表单标签都可以用】
    - 节点对象.checked;【针对多选框和单选框用的多】
    - 节点对象.readOnly;【针对文本框或文本域】
    - 节点对象.disabled【针对按钮用的多】

1.2 操作节点对象的自定义属性 【了解】

- 什么是自定义属性

> 针对html标签的属性可以分为两类：
>
> - 标签自带属性（语言设计者提供的属性）
>   id、title、src、href、name、type等

- 自定义标签属性
  用户根据需求，自己给标签添加的自己定义的标签属性

```
如：
<img src='wc.jpg' bigImg='bigWc.jpg' />
bigImg='bigWc.jpg' 就是用户自定义的标签属性
```

- 操作方式
  - 获取

```
节点对象.getAttribute('属性名');    //会返回标签的属性的值
```

  - 设置

```
节点对象.setAttribute('属性名','值');    //会修改或添加标签属性
```

  - 删除

```
节点对象.removeAttribute('属性名');    //会删除标签的属性
```

二. 操作节对象的style属性

2.1 为什么要获取节点对象的style属性 【了解】

目的：操作样式。
怎么样操作：
　　获取到指定节点对象的style属性。style属性是一个对象，通过style属性可以获取或设置对应标签的样式表。

2.1 获取节点对象的style属性 【重点】

- 方式一：
  - 语法：

    ```
    节点对象.style;  //获取节点对象的style属性对象
    ```

  - 获取指定样式属性对应的值：

    ```
    节点对象.style.样式属性名;
    节点对象.style['样式属性名'];
    //注意：对于像font- 、text-、background-等这些带有“-”样式属性名，“-”要      去掉，并且后面的单词首字母要大写。如：backgroundColor
    //缺点：仅仅只能够获取行内样式属性，内部和外联样式属性无法获取。
    ```

  - 代码：

```html
<!DOCTYPE html>
<html>
<head lang="en">
  <meta charset="UTF-8">
  <title></title>
  <style>
    div{
      height: 100px;
      background-color: red;
    }
  </style>
</head>
<body>
  <div style="width:100px;" id="dv"></div>
  <script>
    var dvNode = document.getElementById('dv');
    console.log(dvNode.style.width);  //100px 可以获取行内样式
    console.log(dvNode.style.height); //空 无法获取内部或外联样式
    console.log(dvNode.style.backgroundColor); //空 无法获取内部或外联样式
  </script>
</body>
</html>
```

- 方式二：
  - 语法

```
window.getComputedStyle(节点对象,null);   //获取节点对象的style属性对象
//window可以省略
//缺点：有兼容性问题（在IE8及IE8以下无法使用getComputedStyle）
```

- 获取指定样式属性对应的值：

```
getComputedStyle(节点对象,null).样式属性名;
getComputedStyle(节点对象,null)['样式属性名'];
//缺点：有兼容性问题（在IE8及IE8以下无法使用getComputedStyle）
```

- 代码

```
<!DOCTYPE html>
<html>
<head lang="en">
  <meta charset="UTF-8">
  <title></title>
  <style>
    div{
      height: 100px;
      background-color: red;
    }
  </style>
</head>
<body>
<div style="width:100px;" id="dv"></div>
<script>
  var dvNode = document.getElementById('dv');

console.log(getComputedStyle(dvNode,null).width); //100px  可以获取行内样式
  console.log(getComputedStyle(dvNode,null)['height']);  //100px  可以获取内部样式

console.log(getComputedStyle(dvNode,null).backgroundColor);  //rgb(255, 0, 0) 可以获取内部样式

</script>
</body>
</html>
```

- 方式三：
  - 语法

```
节点对象.currentStyle;//获取节点对象的style属性对象
//缺点：仅仅支持IE,其他浏览器不兼容
```

- 获取指定样式属性对应的值：

```
节点对象.currentStyle["样式属性名"];
节点对象.currentStyle.样式属性名;
//缺点：仅仅支持IE,其他浏览器不兼容
```

- 代码

```
<!DOCTYPE html>
<html>
<head lang="en">
  <meta charset="UTF-8">
  <title></title>
  <style>
    div{
      height: 100px;
      background-color: red;
    }
  </style>
</head>
<body>
<div style="width:100px;" id="dv"></div>
<script>
  var dvNode = document.getElementById('dv');
  console.log(dvNode.currentStyle.width);
//100px  可以获取行内样式
  console.log(dvNode.currentStyle['height']);
//100px  可以获取内部样式

console.log(dvNode.currentStyle.backgroundColor)
;  //rgb(255, 0, 0) 可以获取内部样式

</script>
</body>
</html>
```

- 兼容处理
  - 处理方式:

```
/*
  获取指定节点对象的指的样式的属性值：
  参数：
    nodeObj 节点对象
    propertyName  样式属性名

  返回值：样式属性对应的值
*/
function getStyleValue(nodeObj,propertyName){
  if(window.getComputedStyle){ //如果支持 谷歌 其他浏览器
    return getComputedStyle(nodeObj,null)[propertyName];
  }else{// IE浏览器
    return nodeObj.currentStyle[propertyName];
  }
}
```

- 代码：

```html
<!DOCTYPE html>
<html>
<head lang="en">
  <meta charset="UTF-8">
  <title></title>
  <style>
    div{
      height: 100px;
      background-color: red;
    }
  </style>
</head>
<body>
<div style="width:100px;" id="dv"></div>
<script>
  /*
    获取指定节点对象的指的样式的属性值：
    参数：
      nodeObj 节点对象
      propertyName  样式属性名

    返回值：样式属性对应的值
  */
  function getStyleValue(nodeObj,propertyName){
    if(window.getComputedStyle){ //如果支持 谷歌其他浏览器
      return getComputedStyle(nodeObj,null)[propertyName];
    }else{// IE浏览器
      return nodeObj.currentStyle[propertyName];
    }
  }
  var dvNode = document.getElementById('dv');
  console.log(getStyleValue(dvNode,'width'));
```

```
  console.log(getStyleValue(dvNode,'height'));

console.log(getStyleValue(dvNode,'backgroundColor'));

</script>
</body>
</html>
```

2.2 设置节点对象的style属性 【重点】

- 语法：

```
节点对象.style.样式属性名 = 值;
节点对象.style['样式属性名'] = 值;
```

- 代码：

```html
<!DOCTYPE html>
<html>
<head lang="en">
  <meta charset="UTF-8">
  <title></title>
  <style>
    div{
      height: 100px;
      background-color: red;
    }
  </style>
</head>
<body>
<div style="width:100px;" id="dv"></div>
<script>

  var dvNode = document.getElementById('dv');
  //点击div更改大小和背景色
  dvNode.onclick = function(){
    this.style.width = '500px';
    this.style.height = '500px';
    this.style['backgroundColor']='#000';
  }

</script>
</body>
</html>
```

三. 节点对象的层级关系 【重点】

3.1 节点对象的关系 【了解】

- 祖孙关系

- 父子关系
- 兄弟关系

3.2 根据关系查找节点对象 【重要】

- 节点对象.parentNode;　　【重点掌握】

  > 获取父节点对象

- 节点对象.childNodes;　　【重点掌握】

  > 获取所有的子节点对象，包含空白文本节点和标签节点对象

- 节点对象.children;　　【重点掌握】

  > 获取所有的子节点对象，仅仅包含标签节点对象

- 节点对象.nextSibling;

  > 获取下一个同级的节点对象，包含空白文本节点对象

- 节点对象.previousSibling;

  > 获取上一个同级的节点对象，包含空白文本节点对象

- 节点对象.nextElementSibling;

  > 获取下一个同级的标签节点对象，有兼容问题IE9以下不支持

- 节点对象.previousElementSibling;

  > 获取上一个同级的标签节点对象，有兼容问题IE9以下不支持

- 节点对象.firstChild;

  > 获取第一个子节点对象，包含空白文本节点对象

- 节点对象.lastChild;

> 获取最后一个子节点对象，包含空白文本节点对象

- 节点对象.firstElementChild;

> 获取第一个子标签节点对象，有兼容问题IE9以下不支持

- 节点对象.lastElementChild;

> 获取第最后一个子标签节点对象，有兼容问题IE9以下不支持