# 问题

1:请你说一下前端三层?

HTML【标签】 结构层

CSS【样式】 样式层

JS【JavaScript】 行为层

2：请你描述一下JS组成？

ECMAScript:欧洲计算机协会大概每年6月中旬定制语法规范

DOM[document object model]：文档对象模型

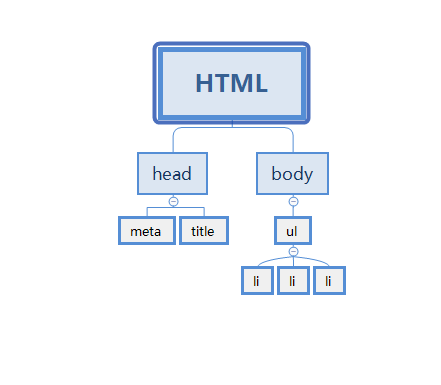
BOM【browser object model】：浏览器对象模型

# 节点树

节点【element】即为标签

概述：静态页面骨架是由标签组成，标签之间关系很像一颗大树，简称为节点树。

如下图：静态页面标签之间关系图，很像一颗大树，因为前端人称之为节点树。

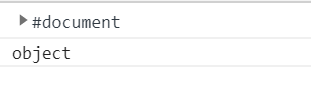


## 2.1认识DOM

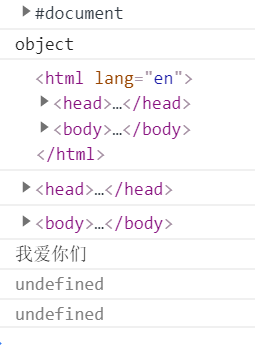
document:文档、页面

概述：DOM【全称：document object model】起始就是系统内置引用类型对象document，你可以认为他是整个节点树‘根元素’，

JS当中DOM对象可以通过一些手段操作结构层中标签。



### 2.1.1DOM四个小属性



* documentElement属性：可以获取页面中HTML标签
* head属性:可以获取到head标签
* body属性：可以获取到body标签
* title属性：可以操作title标签文本

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>我爱你们</title>

</head>

<body>

    <div></div>

</body>

</html>

<script>

    //DOM：起始就是内置document（页面）对象，你可以认为是当前页面的‘根元素’

    //DOM：是引用类型数据

    console.log(document);

    console.log(typeof document);

    //四个小属性

    console.log(document.documentElement);

    console.log(document.head);

    console.log(document.body);

    console.log(document.title);

    //下面是错误的写法

    console.log(document.div);

    console.log(document.meta);

</script>

### 2.1.2DOM方法

上面我们已经学习了四个属性:documentElement(获取HTML标签),head(获取head标签),body(获取body标签),title(获取标题文字内容)。

如果想获取其他页面中标签：需要通过一些方法获取，DOM方法很多，今天怎么暂时学习一个。

getElementById：DOM这个方法可以通过标签的ID选择器匹配任意节点树上标签。

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8" id="box">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <div>

        <ul id="box1">

            <li>吃饭</li>

            <li id="box2">睡觉</li>

            <li>打豆豆</li>

        </ul>

    </div>

</body>

</html>

<script>

    //getElementById:是DOM方法，可以通过标签ID属性值获取任意节点

    var meta = document.getElementById("box");

    console.log(meta);

    //获取ul标签

    var ul = document.getElementById("box1");

    console.log(ul);

    //获取li标签

    var li = document.getElementById("box2");

    console.log(li);

</script>

* getElementById：是DOM方法，可以通过节点ID属性值获取节点树上任意标签
* Script标签一般放置程序最下方（因为加载先后顺序问题）

## 2.2操作标签文本

概述：我们可以通过DOM方法获取节点树上任意节点，初次之外，获取节点经常操作自己文本内容；

* 如果是表单元素：需要通过value属性操作文本【经常结合form标签一起使用input标签】
* 如果是非表单元素：需要通过innerHTML属性操作文本【非input标签】

下面红色地方是称之为标签文本：

<div>我是祖国的未来，妈妈说的呀</div>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <div>

        我是表单元素:<input type="text" value="最近天气好冷呀" id="cur">

    </div>

    <a href="http://www.baidu.com" id="box">跳转到百度</a>

    <p id="memeda">欢迎大家来贾老师课程~~快速了解web</p>

</body>

</html>

<script>

    //如果操作表单元素文本：需要使用标签的value属性

    //操作文本：即为可以在JS当中获取已有文本、但让也可以重新设置文本

    //第一步：获取表单元素

    var input = document.getElementById("cur");

    //获取表单元素文本

    console.log(input.value);

    //从新设置表单元素文本

    input.value = "我是通过JS动态修改表单元素文本";

    //非表单元素：就不是input即可

    //非表单元素操作文本：需要通过innerHTML属性

    var a = document.getElementById("box");

    //获取非表单元素文本

    console.log(a.innerHTML);

    //重新设置非表单元素文本

    a.innerHTML = "贾老师你好帅";

    //获取P标签

    var p = document.getElementById("memeda");

    //获取文本

    console.log(p.innerHTML);

    //动态修改

    p.innerHTML = "最近有的同学老不来上课";

</script>

## 2.3操作标签属性

下面红色部分称之为属性：

<img src=”./1.jpg”>

<div id=”box” class=”cur”>这里叫做文本</div>

<!-- 下面这行代码：是文档声明方式，不是标签（H5） -->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <img src="./img/1.jpg" alt="" id="box">

    <div class="cur" id="ha">我是一个粉刷匠</div>

</body>

</html>

<script>

    //获取img标签

    var img = document.getElementById("box");

    //获取标签属性值

    console.log(img.src);

    //重新修改图片src属性值

    img.src = "./img/2.jpg";

    //获取div标签

    var div = document.getElementById("ha");

    //获取div的class、id属性值

    console.log(div.id);

    console.log(div.className);

    //从新设置class属性值

    div.className = "rewrite";

    div.id = "gagaga";

    //修改文本

    div.innerHTML = "粉霜本领强";

</script>

* 大多情况下：操作标签属性标签.属性名字进行操作
* 但是有特例：class属性，如果操作标签class属性，需要通过className属性值操作

## 2.4操作标签样式

概述：我们通过DOM可以操作标签的行内样式。

区分web样式：

Link:称之为外部样式

Style：称之为内部样式

//下面写法称之为行内样式：

<h1 style=“color:red”>我是文本</h1>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <div style="width:100px;height: 100px;color:blueviolet;border: 1px solid black;" id="cur">12306</div>

</body>

</html>

<script>

    //获取div标签

    var div = document.getElementById("cur");

    //获取已有样式

    console.log(div.style.width);

    console.log(div.style.height);

    console.log(div.style.color);

    console.log(div.style.border);

    //设置元素样式

    div.style.width = "250px";

    div.style.color = "red";

    //如果样式中中间带有横杠变驼峰写法

    div.style.fontSize = "30px";

    div.style.textAlign = "center";

    div.style.background = "cyan";

</script>

* DOM能操作的只是标签的行内样式（外部样式、内部样式暂时操作不了）
* 样式中属性带有横杠变为驼峰写法
* 该有单位需要有单位

## 2.5事件\*\*\*\*\*\*\*

概述：事件英文【event】，事件是由用户触发，当用户触发事件的时候，标签可以做出相应的响应。

比如：用户触发单击事件、鼠标移上、鼠标移下.......

语法格式：

element.onxxxx = function(){

}

* 红色这个函数：称之为事件处理函数（当用户触发事件的时候才会执行一次）

### 2.5.1单击事件-onclick

概述：你可以给任意标签绑定单击事件（当用户触发单击事件的时候），事件处理函数才会执行一次。

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

    <style>

        \* {

            margin: 0;

            padding: 0;

        }

        #box {

            width: 600px;

            height: 100px;

            margin: 100px auto;

            background: red;

            color: white;

            text-align: center;

            line-height: 100px;

        }

        #box1 {

            /\* 定位元素 \*/

            position: relative;

            width: 100px;

            height: 100px;

            background: cyan;

            left: 0px;

        }

        #box2 {

            width: 300px;

            height: 100px;

            background: pink;

        }

    </style>

</head>

<body>

    <div id="box">文字</div>

    <div id="box1"></div>

    <div id="box2"></div>

</body>

</html>

<script>

    //给第一个div标签绑定单击事件

    var div = document.getElementById("box");

    var div1 = document.getElementById("box1");

    var div2 = document.getElementById("box2");

    //全局变量：存储一下起始字号大小

    var f = 16;

    var w = 600;

    //绑定单击事件

    div.onclick = function () {

        //字号加上1

        f++;

        w += 5;

        div.style.fontSize = f + "px";

        div.style.width = w + "px";

    }

    //第二个div

    var l = 0;

    div1.onclick = function () {

        l += 10;

        div1.style.left = l + "px";

    }

    //第三个div

    div2.onclick = function () {

        //随机三个数字

        var R = parseInt(Math.random() \* 255);

        var G = parseInt(Math.random() \* 255);

        var B = parseInt(Math.random() \* 255);

        //随机颜色

        div2.style.background = `rgb(${R},${G},${B})`;

    }

</script>

## 注意：JS属于脚本语言，脚本语言需要嵌套在静态页面中，给静态页面添加一些动态效果。

## 2.6动态创建标签

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <ul id="box">

    </ul>

</body>

</html>

<script>

    //先获取UL标签

    var ul = document.getElementById("box");

    //DOM提供给我们一个createElement方法，可以在JS当中动态创建标签

    //循环语句创建10000li

    for (var i = 0; i < 10000; i++) {

        var li = document.createElement("li");

        li.innerHTML = i;

        //UL追加子元素

        ul.appendChild(li);

    }

</script>