DOM

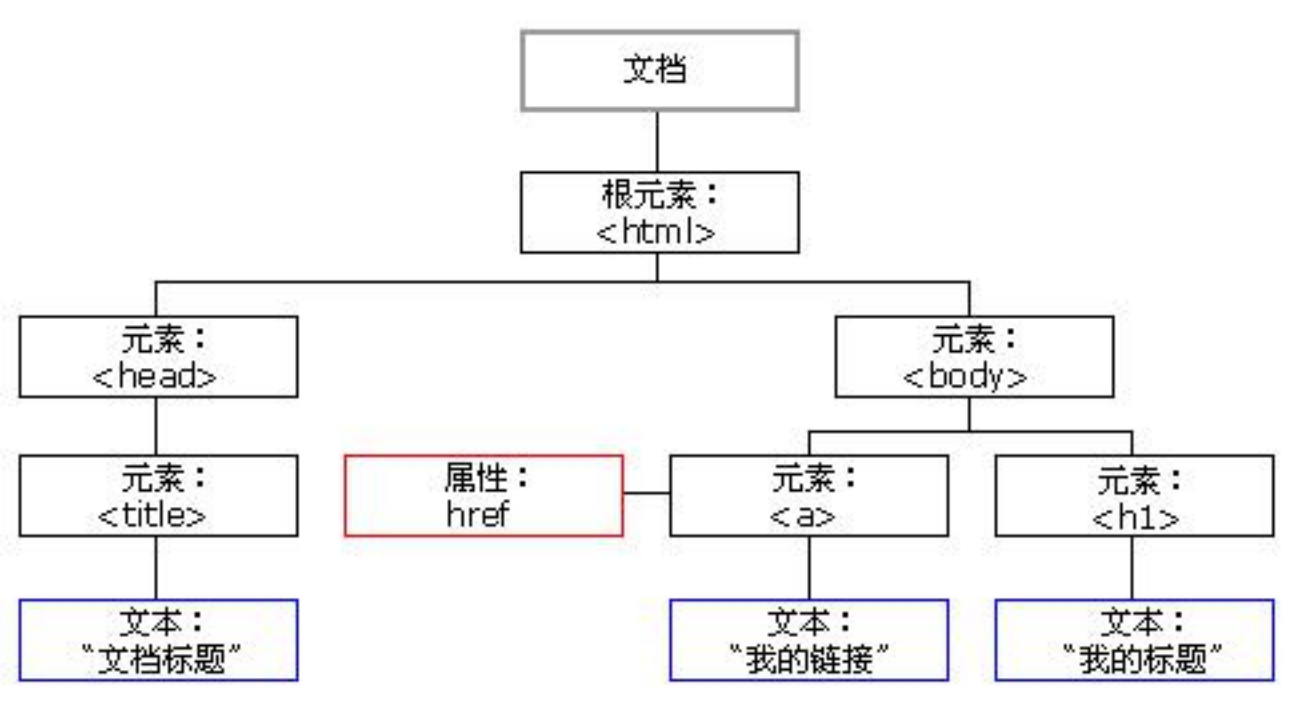
从入门到工作2020版

版权声明

- 本内容版权归属杭州饥人谷教育科技有限公司所有。
- 任何媒体、网站、个人未经本公司授权不得转载、链接、转帖，或以其他方式复制、发布、发表。
- 已获得饥人谷授权的媒体、网站、个人在使用时必须注明“资料来源：饥人谷”
- 如咨询课程，微信联系xiedaimala02 或xiedaimala03 ，或者点击链接直接咨询https://dwz.cn/pPsjntxY

DOM

- 文档对象模型 (Document Object Model, DOM) 是HTML的编程接口。用对象和HTML结构一一映射，通过修改对象达到修改HTML节点对目的。



document

- 包含文档对基本信息

- ✓ document.head
- ✓ document.body
- ✓ document.title
- ✓ document.location
  - document.location.href
  - document.location.hash
  - docuemnt.location.search
  - document.location.reload()
  - document.location.assign('https://jirengu.com')

Element

- DOM元素对象

✓ 常见属性如下

- node.nodeName：元素标签名，还有个类似的tagName
- node.nodeType：元素类型
- node.className：类名
- node.id：元素id
- node.children：子元素列表（HTMLCollection）
- node.childNodes：子元素列表（NodeList）
- node.firstChild：第一个子元素
- node.lastChild：最后一个子元素
- node.nextSibling：下一个兄弟元素
- node.previousSibling：上一个兄弟元素
- node.parentNode：父元素

获取元素

```
//ES3写法，不推荐
document.getElementById('id')                 //Element
document.getElementsByClassName('class')      //HTMLCollection
document.getElementsByTagName('p')            //HTMLCollection
document.getElementsByName('username')        //NodeList

//ES5写法，推荐
document.querySelector('.box')                //Element
document.querySelectorAll('.box')             //NodeList
```

注意

1. getElementsByClassName获取的是一个类数组对象，不是一个数组
2. 不要少写getElementsByClassName 中的s
3. querySelector的参数是一个任意合法的CSS3选择器
4. querySelectorAll的到的是一个类数组对象
5. NodeList和HTMLCollection的差异是前者有forEach方法

遍历类数组对象

- ✓ HTMLCollection没有forEach方法

```
//NodeList可以直接遍历
document.querySelector('.box').forEach(node => {
  console.log(node.className)
})

//HTMLCollection需要转换为数组才能用forEach
Array.form(document.getElementsByClassName('box')).forEach(node => {
  console.log(node.className)
})
[...document.getElementsByClassName('box')].forEach...
```

DOM事件

- 事件绑定
  - ✓ node.onclick = function(e){}
  - ✓ node.addEventListener('click', function(e){})

- 注意
  - ✓ 在函数里面可以使用事件对象e
  - ✓ 函数里面的this代表node本身

Tips：这里为事件的入门介绍，详细知识后面讲解

class操作

- 常见API
  - node.className　老方法，用起来不严谨，尽量不用
  - node.classList.add
  - node.classList.remove
  - node.classList.contains
  - node.classList.toggle

```
<div class="box"></div>
<button>切换</button>
<style>
  .box { width: 100px; height: 100px; background: red; }
  .box.active {background: yellow;}
</style>
<script>
const button = document.querySelector('button')
const box = document.querySelector('.box')
button.onclick = function() {
  box.classList.toggle('active')
}
</script>
```

属性操作

- 常见API
  - node.getAttribute('id')
  - node.setAttribute('data-id', '100')
  - node.removeAttribute('checked')
  - document.createAttribute('checked')
  - node.setAttributeNode(attrNode)

- 动手
  - 实现点击全选，和取消全选的切换

元素创建

- 常见API

✓ document.createElement('div')
  - 创建DOM节点，参数是标签名

✓ document.createTextNode("Hello")
  - 创建文本节点，参数是字符串

✓ document.createDocumentFragment()
  - 创建一个虚拟的DOM，用于提升性能，避免高频DOM操作

元素复制添加修改删除

- 常见API

✓ parentNode.appendChild(childNode)
  - 在父亲的末尾添加元素

✓ parentNode.insertBefore(newNode, referenceNode)
  - 把newNode插入到referenceNode元素之前

✓ parentNode.replaceChild(newChild, oldChild)
  - 把oldChild替换为newChild

✓ parentNode.removeChild(childNode)
  - 从parentNode里删除childNode

✓ node.cloneNode(true)
  - 克隆一个元素，参数true的时候会深复制，也就是会复制元素及其子元素，false的时候只复制元素本身

修改样式

- 修改style属性

```
document.querySelector('p').style.color = 'red'
document.querySelector('.box').style.backgroundColor = '#ccc'

function css(node, cssObj) {
  Object.assign(node.style, cssObj)
}

css(document.body, {
  color: 'red',
  backgroundColor: 'blue'
})
```

Tips: 一般通过添加删除class的方式修改样式

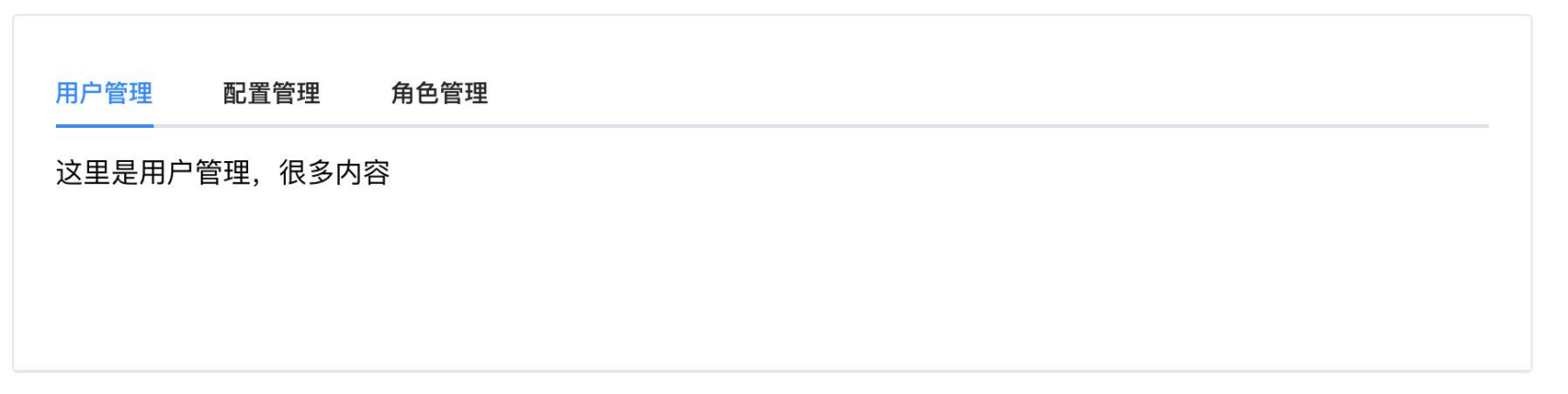
获取样式

- ✓ 使用getComputedStyle获取元素计算后的样式，不要通过 node.style.属性 获取

```
const node = document.querySelector('p')
const color = window.getComputedStyle(node).color
console.log(color)
```

小实战：点击按钮实现指定元素字体变大变小

实战1 实现Tab选项卡



实战2 手风琴效果

一致性 Consistency

反馈 Feedback

控制反馈：通过界面样式和交互动效让用户可以清晰的感知自己的操作；
页面反馈：操作后，通过页面元素的变化清晰地展现当前状态。

效率 Efficiency

可控 Controllability