JavaScript 核心与进阶

JavaScript基础
5天

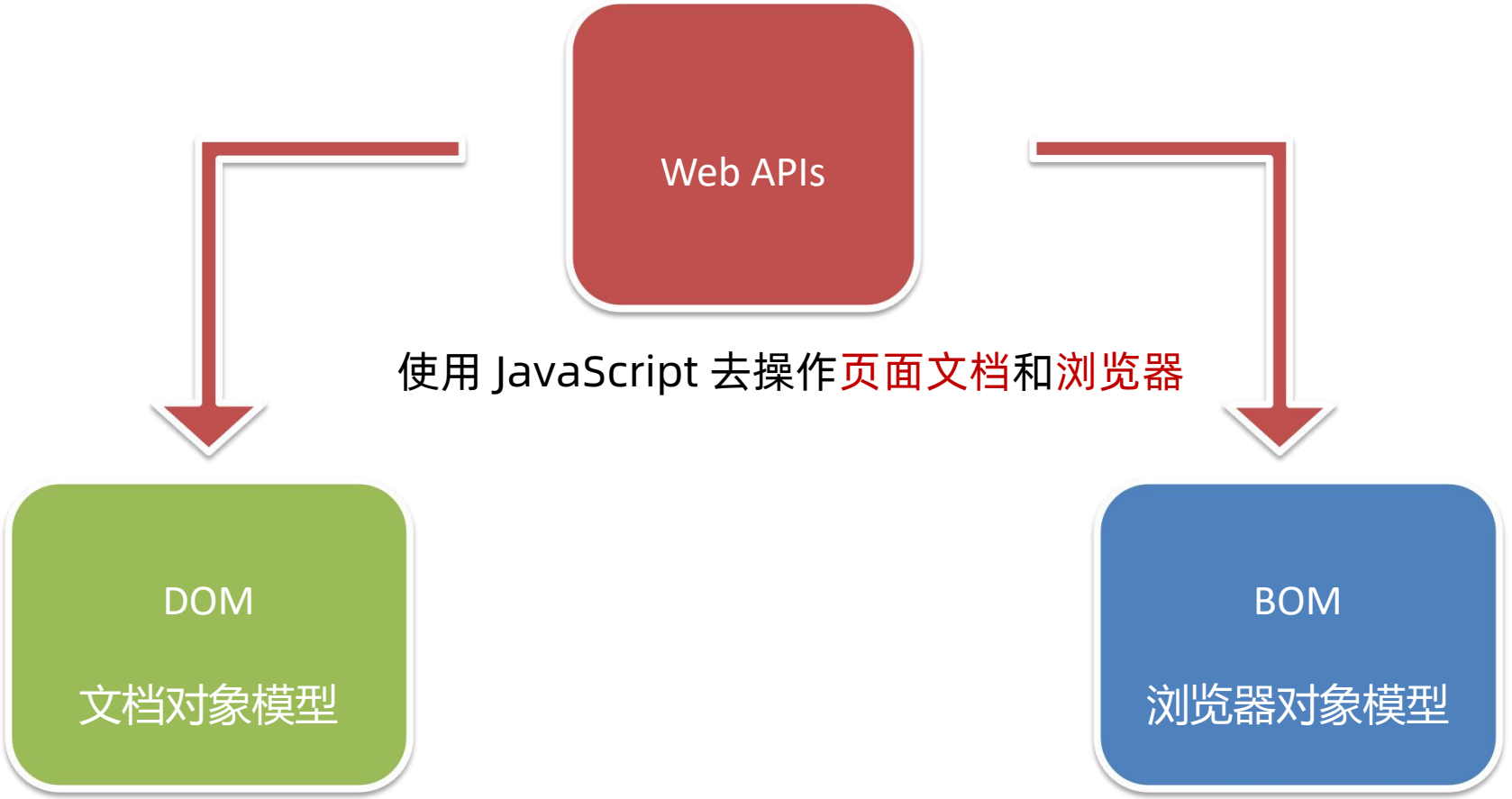
Web APIs
7天

JavaScript进阶
4天

基础：
基本写法，知道怎么控制计算机去执行代码....

应用：
操作页面元素做出各种好玩的效果....

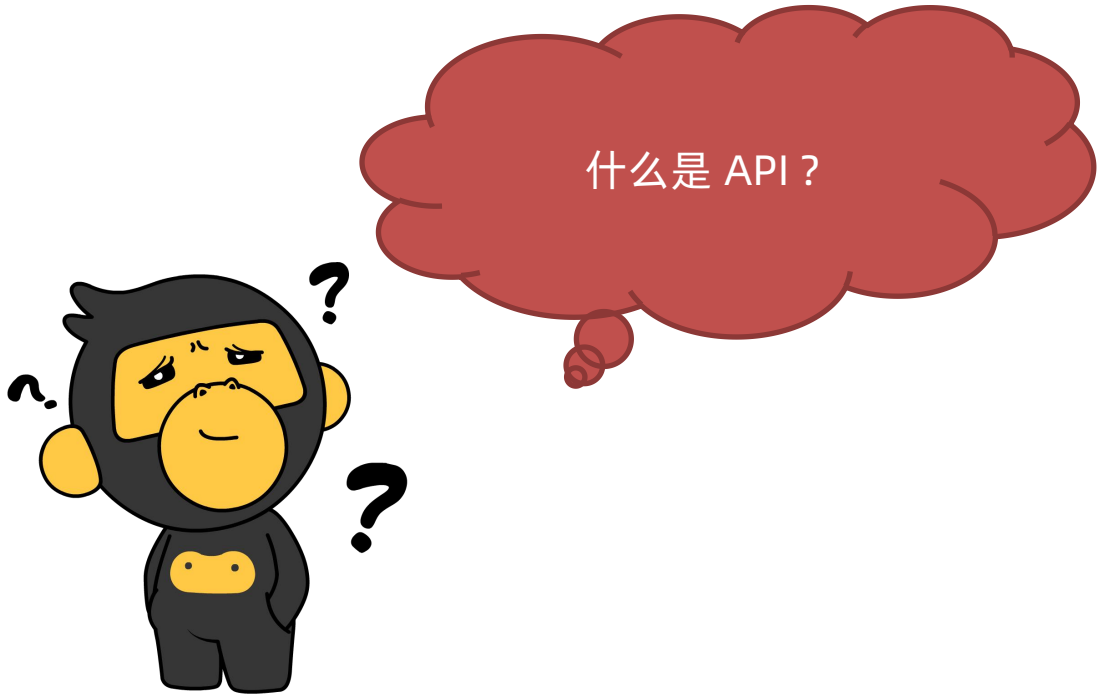
进阶：
高级语法、高频面试题、底层原理....



使用 JavaScript 去操作页面文档和浏览器

使用 JavaScript 去操作页面文档

使用 JavaScript 去操作浏览器


什么是 API？

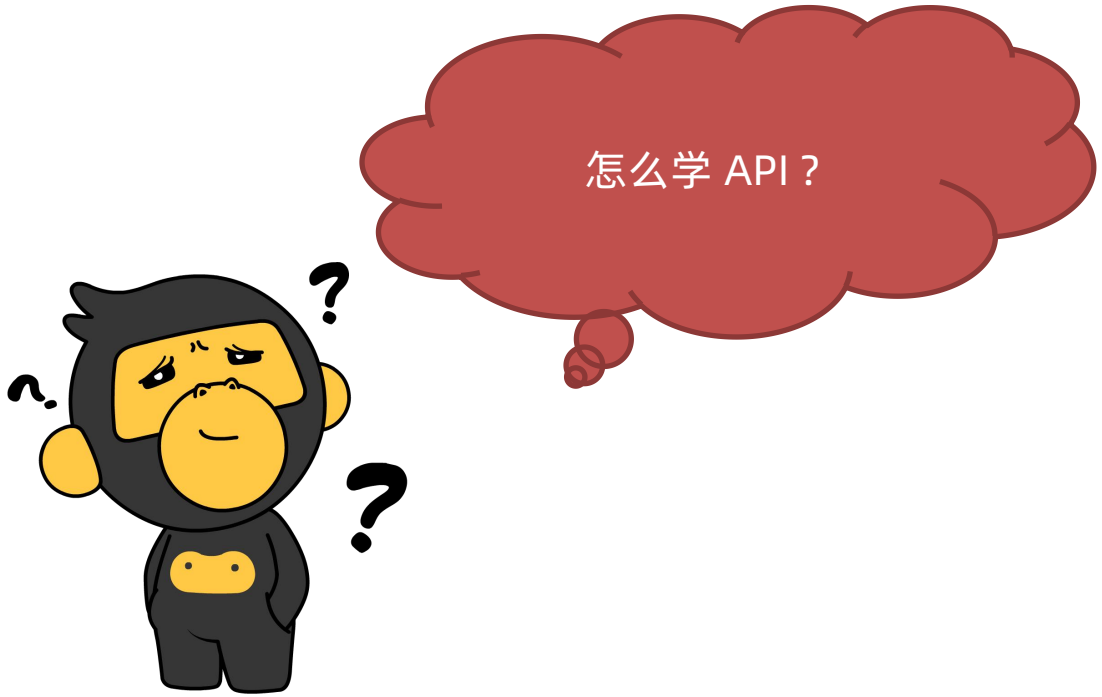
什么是API?

- API：应用程序接口（Application Programming Interface）
- 接口：无需关心内部如何实现，程序员只需要调用就可以很方便实现某些功能

什么是API?

- API：应用程序接口（Application Programming Interface）
- 作用：开发人员使用 JavaScript提供的接口来操作网页元素和浏览器

```
alert('我会弹出一个警示框')
```


怎么学 API？

怎么学习API?

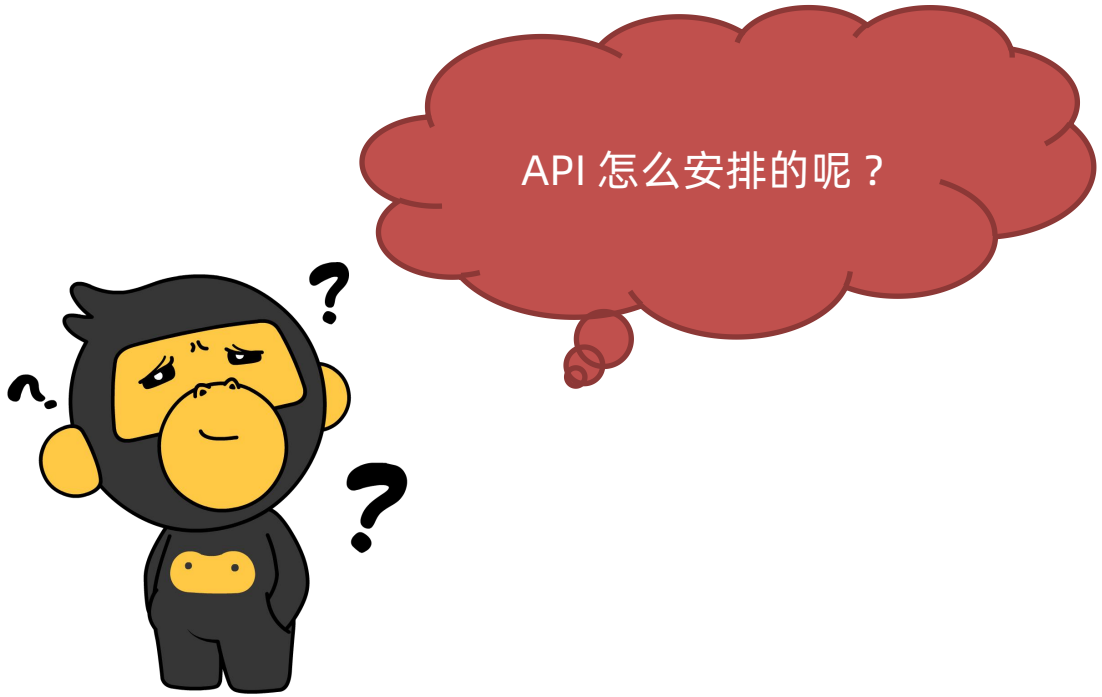
好习惯

理解API（是啥）

- 它能做什么？
- 它能解决什么需求？

查阅文档（咋用）

- API里面包含哪些属性和方法？
- 有哪些注意事项
- 怎么使用？
- MDN 强烈推荐
- https://developer.mozilla.org/zh-CN/


API 怎么安排的呢？

Web APIs 课程安排 7天

第一天

DOM-操作元素

获取元素、修改属性

第二天

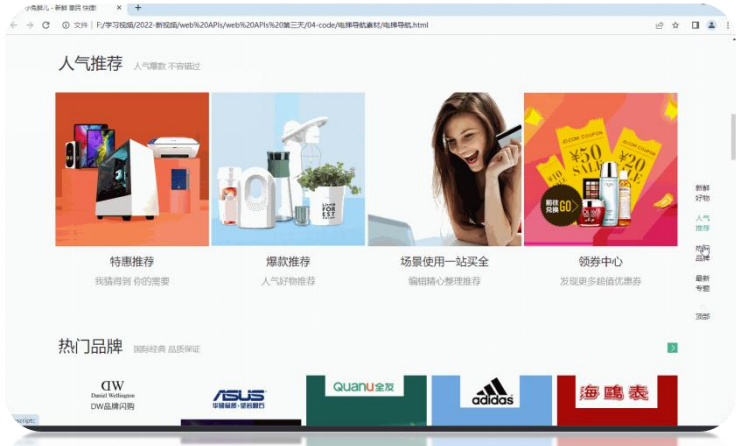
DOM-事件基础

注册事件、tab栏切换

第三天

DOM-事件进阶

事件对象、事件委托



Web APIs 课程安排 7天

第四天

DOM-节点操作

节点的增删改查

第五天

BOM-操作浏览器

BOM、插件、本地存储

第六天

正则表达式

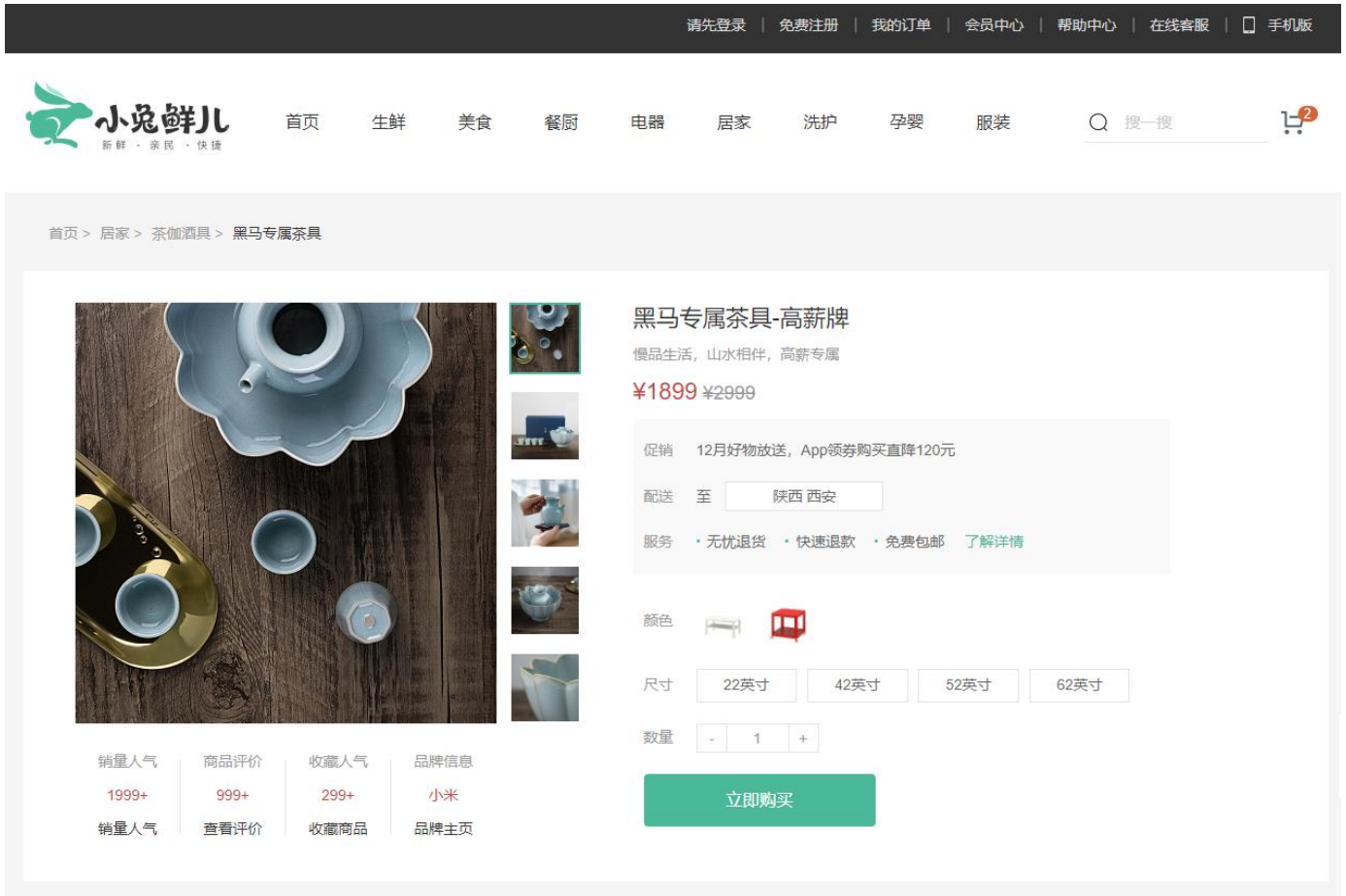
正则表达式、综合案例

Web APIs 课程安排 7天

第七天

实战

小兔鲜个人实战



web APIs 第一天

获取元素、操作元素、定时器

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

描述	属性/方法	效果
获取DOM对象	document.querySelector()	获取指定的第一个元素
	document.querySelectorAll()	获取指定的所有元素
操作元素内容	元素.innerText	操作元素内容，不解析标签
	元素.innerHTML	操作元素内容，解析标签
操作元素样式	元素.style.width	通过style操作样式
	元素.className	通过类名操作样式
	元素.classList.add()	增加类名
	元素.classList.remove()	删除类名
	元素.classList.toggle()	切换类名
间隔函数	setInterval(function() {}, 1000)	定时器，每隔指定时间重复执行

学习目标

Learning Objectives

1. 掌握DOM属性操作，完成元素内容设置
2. 元素属性设置，控制元素样式

目录

Contents

DOM 简介

- 什么是DOM

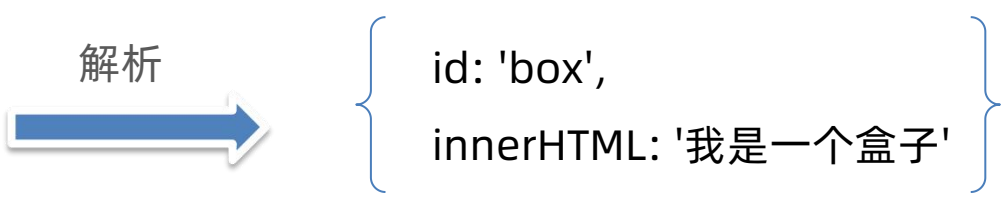
1. 什么是DOM

- DOM（Document Object Model——文档对象模型）
- 作用：DOM用来 操作网页文档，开发网页特效和实现用户交互
- DOM的核心思想就是把网页内容当做对象来处理，通过对象的属性和方法对网页内容操作

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>页面</title>
</head>
<body>

  <div id="box">我是一个盒子</div>

</body>
</html>
```



1. 什么是DOM

- DOM（Document Object Model——文档对象模型）
- 作用：DOM用来 操作网页文档，开发网页特效和实现用户交互
- DOM的核心思想就是把网页内容当做对象来处理，通过对象的属性和方法对网页内容操作

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>页面</title>
</head>
<body>

  <div id="box">我是box</div>

</body>
</html>
```

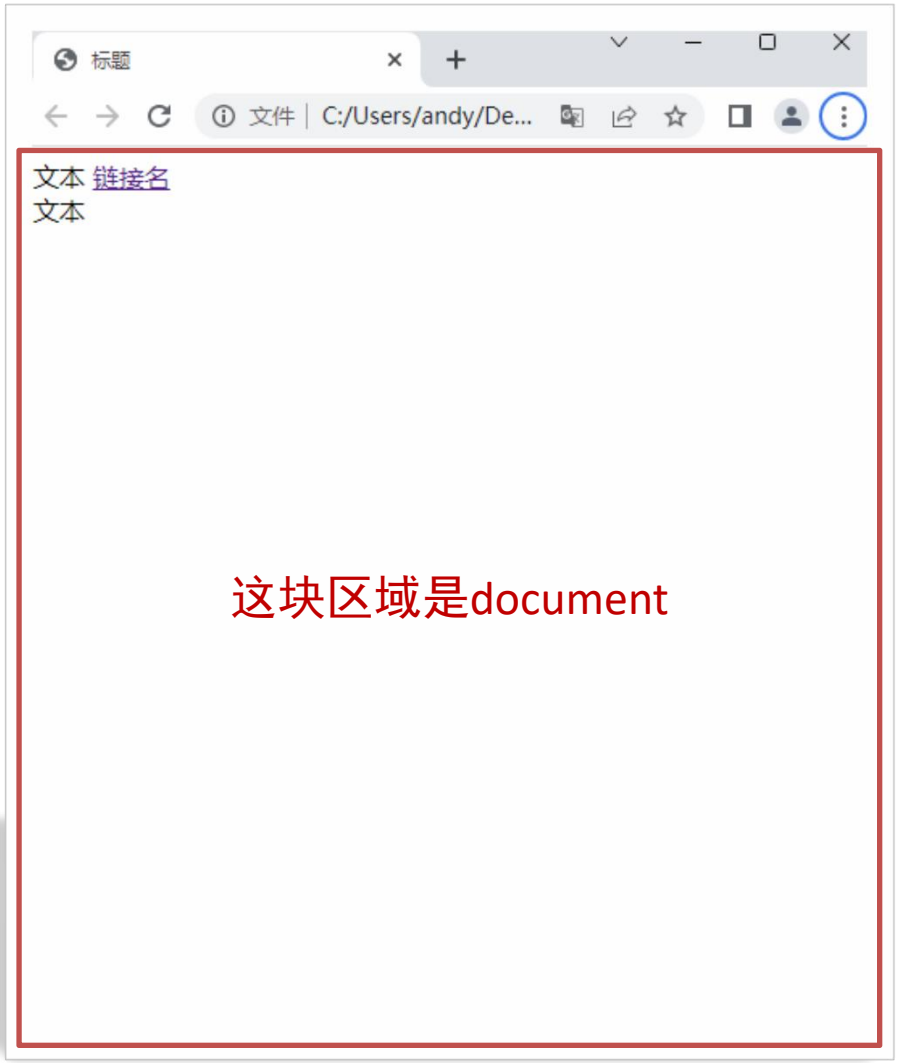
DOM对象

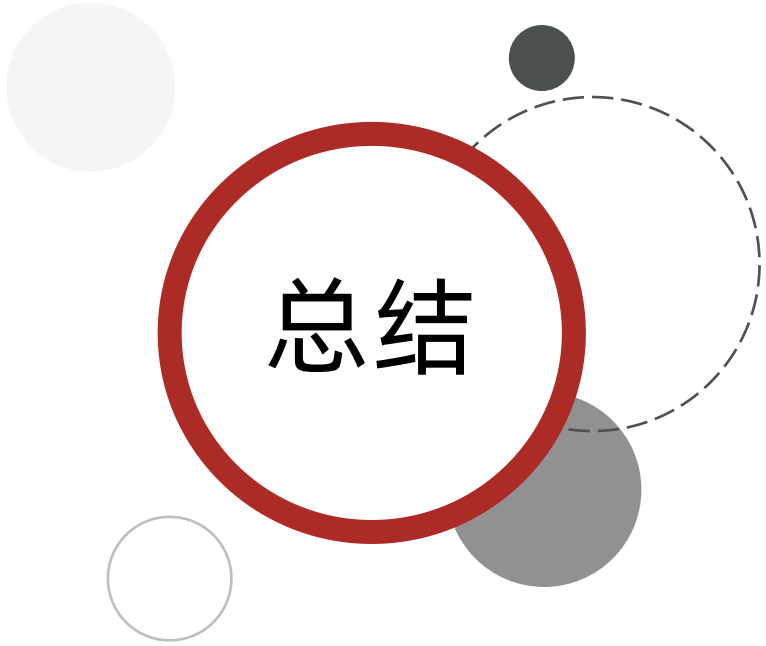
```
id: 'box',
innerHTML: '我是一个盒子'
```

对象.innerHTML = '我是box'

1. 什么是DOM

- document 对象
  - 是 DOM 里提供的一个**对象，**是DOM顶级对象
  - 作为网页内容的入口
  - 所以它提供的属性和方法都是**用来访问和操作网页内容的**
    - ✓ 例：document.write()





1. DOM是什么？作用是什么？
   - 文档对象模型
   - 操作网页文档，开发网页特效和实现用户交互
2. DOM核心思想是什么？
   - DOM核心就是把网页内容当做对象来处理
   - 比如DOM会把标签解析成 DOM对象
   - 修改这个对象的属性会自动修改到标签身上，从而可以修改标签的结构、样式或者内容
3. DOM的顶级对象是 document 对象

目录

Contents

02

获取DOM元素

- 根据CSS选择器来获取DOM元素 （重点）
- 其他获取DOM元素方法（了解）

2. 获取DOM元素

想要操作页面元素，那我们需要先利用DOM方式来获取（选择）这个元素

1. CSS选择器来获取DOM元素（重点）

document.querySelector()

document.querySelectorAll()

2. 其他获取DOM元素（了解）

document.getElementById()

document.getElementsByClassName()

document.getElementsByTagName()

document.getElementsByName()

2.1 根据CSS选择器来获取DOM元素（重点）

1. 选择匹配的第一个元素

语法：

```
document.querySelector('css选择器')
```

参数：

包含一个或多个有效的CSS选择器 字符串

返回值：

CSS选择器匹配的第一个元素对象

如果没有匹配到，则返回 null

多参看文档：https://developer.mozilla.org/zh-CN/docs/Web/API/Document/querySelector

练习

请获取最后一个li对象，并控制台输出

```
<ul class="nav">
    <li>我的首页</li>
    <li>产品介绍</li>
    <li>联系方式</li>
</ul>
```

2.1 根据CSS选择器来获取DOM元素（重点）

2. 选择匹配的多个元素对象

语法：

```
document.querySelectorAll('css选择器')
```

参数：

包含一个或多个有效的CSS选择器 **字符串**

返回值：

CSS选择器匹配的NodeList 伪数组

```
document.querySelectorAll('ul li')
```

2.1 伪数组

```
document.querySelectorAll('css选择器')
```

得到的是一个**伪数组**：

- 有长度有索引号的数组
- 但是没有 pop()　 push() 等数组方法

想要得到里面的每一个对象，则需要遍历（for）的方式获得

注意事项

哪怕只有一个元素，通过querySelectAll() 获取过来的也是一个**伪数组**，里面只有一个元素而已

总结

1. DOM中获取页面中的元素我们最常用哪两种方式？
   - querySelectorAll()
   - querySelector()
   - 参数是利用css选择器实现，注意是字符串类型
2. 他们两者的区别是什么？
   - querySelector() 只能选择第一个元素
   - querySelectorAll() 可以选择多个元素，得到的是伪数组，需要遍历得到每一个元素
3. 什么是伪数组？
   - 外形是数组，并且有索引号和长度
   - **但是没有一些数组常用的方法，比如 pop 、push等**

练习

请控制台依次输出 3个 li 的 DOM对象

```
<ul class="nav">
    <li>我的首页</li>
    <li>产品介绍</li>
    <li>联系方式</li>
</ul>
```

2.2 其他获取DOM元素方法（了解）

语法	实例	描述
getElementById	document.getElementById('box')	根据id获取元素，单个元素
getElementsByTagName	document.getElementsByTagName('li')	根据标签名获取元素，伪数组
getElementsByClassName	document.getElementsByClassName('one')	根据类名获取元素，伪数组
getElementsByName	document.getElementsByName('sex')	根据name属性值获取元素，伪数组

目录

Contents

操作元素内容

- 对象.innerText 属性
- 对象.innerHTML 属性

三、操作元素内容

- DOM对象可以操作页面标签，所以本质上就是操作DOM对象（增删改查）
- 如果想要操作标签元素的**内容**，则可以使用如下2种方式：
  1. 对象.innerText 属性
  2. 对象.innerHTML 属性

三、操作元素内容

1. 元素.innerText 属性

- "渲染"文本内容到标签里面
- 显示纯文本，不解析标签

举例说明

```
// 首先需要获取这个dom对象
const box = document.querySelector('.box')
//  改
box.innerText = '迪丽热巴'
```

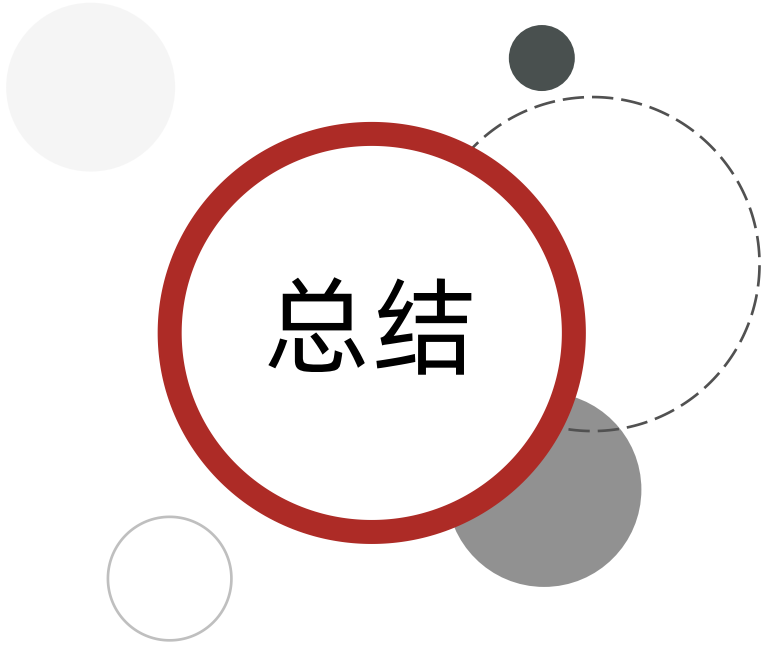
三、操作元素内容

2. 元素.innerHTML 属性

- "渲染"文本内容到标签里面
- 会解析标签

举例说明

```
// 2. 对象.innerHTML  会解析标签
box.innerHTML = '<strong>迪丽热巴</strong>'
```



1. 操作DOM元素内容我们学了哪两种方式？
   - 元素对象.innerText 属性
   - 元素对象.innerHTML 属性
2. 两者的区别是什么？
   - 元素.innerText 属性　只识别文本，不能解析标签
   - 元素.innerHTML 属性　能识别文本，能够解析标签
   - 如果还在纠结到底用谁，你可以选择innerHTML

案例

重构学成在线案例

需求：以前拼接的li字符串，需要在打印整个结构的时候放入的

分析：

①： 利用数据动态生成10个小li 的字符串不变

②： 获取ul容器，直接给 ul.innerHTML 赋值即可

案例

年会抽奖案例

需求：从数组随机抽取一等奖、二等奖和三等奖，显示到对应的标签里面

业务分析：

①：页面刷新随机抽取获奖姓名

②：不允许重复得奖

案例

年会抽奖案例

需求：从数组随机抽取一等奖、二等奖和三等奖，显示到对应的标签里面

分析：

①：**一等奖**:随机生成一个数字（0~数组长度），找到对应数组的名字

②：通过innerText 或者 innerHTML 将名字写入span元素内部

③：因为不允许重复抽奖，所以抽取一等奖的姓名要从数组中删除 splice

④： **二等奖、三等奖**依次类推同样操作

目录

Contents

04

操作元素属性

- 操作元素**常用**属性
- 操作元素**样式**属性
- 操作 **表单元素** 属性
- 自定义属性

4.1 操作元素常用属性

- 还可以通过DOM操作元素属性，比如通过 src 更换 图片地址
- 最常见的属性比如： href、title、src 等等
- 语法：

```
对象.属性 = 值
```

举例说明

```
// 1. 先获取这个元素
const img = document.querySelector('img')
// 2. 操作DOM元素常见属性
// 2.1 查
console.log(img.src)

// 2.2 改
img.src = './images/3.png'
```

练习

页面刷新，图片随机更换

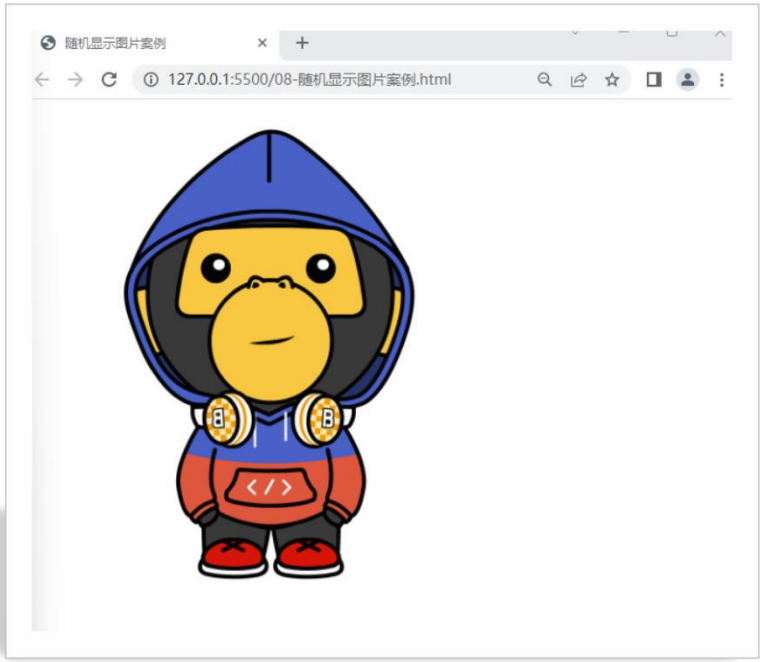
需求：当我们刷新页面，页面中的图片随机显示不同的图片

分析：

①：利用随机数抽取数组中的一个图片地址

②：修改图片元素的src地址（把图片地址赋值给src属性）

```
// 图片地址
const arr = [
  './images/1.png',
  './images/2.png',
  './images/3.png',
  './images/4.png'
]
```



04

操作元素属性

- 操作元素**常用**属性
- 操作元素**样式**属性
- 操作 **表单元素** 属性
- 自定义属性

4.2 操作元素**样式**属性

- 还可以通过 DOM对象修改标签元素的样式属性
  - 比如通过 轮播图小圆点自动更换颜色 样式
  - 点击按钮可以滚动图片，这是移动的的位置 translateX 等等

style **类名** **classList**

style属性操作CSS **类名操作CSS** **classList操作类**

4.2 操作元素**样式**属性

1.通过 style 属性操作元素样式

- 语法：

```
对象.style.样式属性 = 值
```

举例说明

```
const box = document.querySelector('.box')
// 修改元素样式
box.style.width = '200px'
box.style.marginTop = '15px'
box.style.backgroundColor = 'pink'
```

注意：

1. 修改样式通过style属性引出
2. 如果属性有-连接符，需要转换为小驼峰命名法
3. 赋值的时候，需要的时候不要忘记加css单位

总结

1. 操作元素样式属性通过 style 属性引出来？
2. 如果需要修改一个div盒子的样式，比如 padding-left, 如何写？
   - element.style.paddingLeft = '300px'
   - 小驼峰命名法
3. 因为我们是样式属性，一定别忘记，大部分数字后面都需要加单位

```
const box = document.querySelector('.box')
// 修改元素样式
box.style.width = '200px'
box.style.marginTop = '15px'
box.style.backgroundColor = 'pink'
```

练习

随机显示背景图片

需求：当我们刷新页面，页面中的背景图片随机显示不同的图片

分析：

①：利用随机数抽取数组中的一个图片地址

②：修改body元素的背景样式地址

```
// 背景图片地址数组
const arr = [
  './images/bg1.jpg',
  './images/bg2.jpg',
  './images/bg3.jpg',
  './images/bg4.jpg',
  './images/bg5.jpg'
]
```

4.2 操作元素**样式**属性

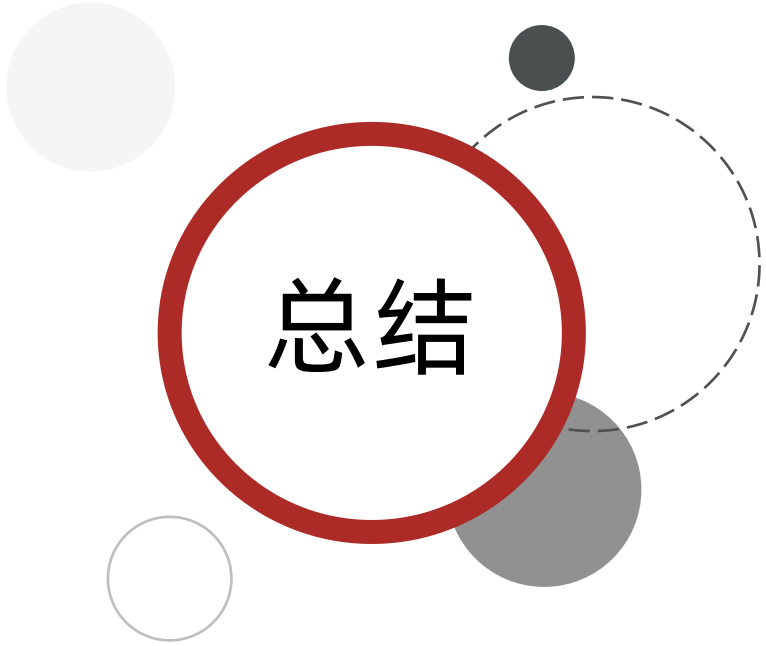
2. 通过类名(className) 操作元素样式（理解）

- 如果修改的样式比较多，直接通过style属性修改比较繁琐，我们可以通过借助于css类名的形式
- 核心：把多个样式放到css一个类中，然后把这个类添加到这个元素身上
- **语法：**

```
// active是一个类名
对象.className = 'active'
```

- **注意：**

1. 由于class是关键字，所以使用className去代替
2. className是使用新值换旧值，如果需要添加一个类，需要保留之前的类名



1. 使用 className 相比较 style 有什么好处？
   - 如果修改多个样式，会方便一些
2. 使用 className 有什么注意事项？
   - 直接使用 className 赋值会覆盖以前的类名

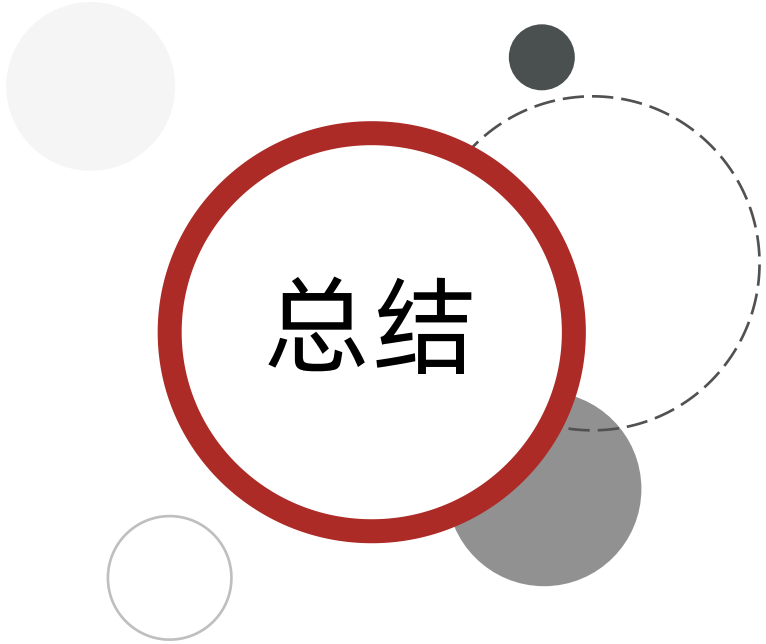
4.2 操作元素**样式**属性

3. 通过 classList 操作元素样式（推荐）

- 为了解决className 容易覆盖以前的类名，我们可以通过classList方式追加和删除类名
- 语法：

```
// 新增一个类名
对象.classList.add('类名')
// 移除一个类名
对象.classList.remove('类名')
// 切换一个类名
对象.classList.toggle('类名')
```

- **注意：** 这三个是方法要加小括号



1. 使用 style、 className 和classList 怎么选择？
   - 修改样式很少的时候，使用 style
   - 修改大量样式的可以选择类：className / classList
   - classList 是追加和删除不影响以前类名，更提倡

案例

轮播图随机版

目的：练习操作元素常见属性和样式属性

需求：当我们刷新页面，页面中的轮播图会显示不同图片以及样式

模块分析：

①：页面刷新随机显示图片、文字、背景颜色

②：对应小圆点高亮显示

案例

轮播图随机版

需求：当我们刷新页面，页面中的轮播图会显示不同图片以及样式

分析：

①： 准备一个对象数组，里面包含详细信息

②： 利用随机数，选出数组对应的对象，用于更换图片、文字和背景颜色

③： 利用上方随机数，让小圆点添加高亮的类（classList.add添加类）

```
// 1. 初始数据对象数组
const sliderData = [
  { url: './images/slider01.jpg', title: '对人类来说会不会太超前了？', color: 'rgb(100, 67, 68)' },
  { url: './images/slider02.jpg', title: '开启剑与雪的黑暗传说！', color: 'rgb(43, 35, 26)' },
  { url: './images/slider03.jpg', title: '真正的jo厨出现了！', color: 'rgb(36, 31, 33)' },
  { url: './images/slider04.jpg', title: '李玉刚：让世界通过B站看到东方大国文化', color: 'rgb(139, 98, 66)' },
  { url: './images/slider05.jpg', title: '快来分享你的寒假日常吧~', color: 'rgb(67, 90, 92)' },
  { url: './images/slider06.jpg', title: '哔哩哔哩小年YEAH', color: 'rgb(166, 131, 143)' },
  { url: './images/slider07.jpg', title: '一站式解决你的电脑配置问题！！！', color: 'rgb(53, 29, 25)' },
  { url: './images/slider08.jpg', title: '谁不想和小猫咪贴贴呢！', color: 'rgb(99, 72, 114)' },
]
```

04

操作元素属性

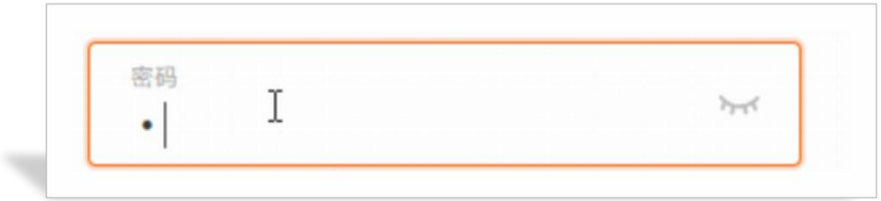
- 操作元素**常用**属性
- 操作元素**样式**属性
- 操作 **表单元素** 属性
- 自定义属性

4.3 操作表单元素属性

- 表单很多情况，也需要修改属性，比如点击眼睛，可以看到密码，本质是把表单类型转换为文本框
- 正常的有属性有取值的 跟其他的标签属性没有任何区别

➢ 获取：DOM对象.属性

➢ 设置：DOM对象.属性 = 新值

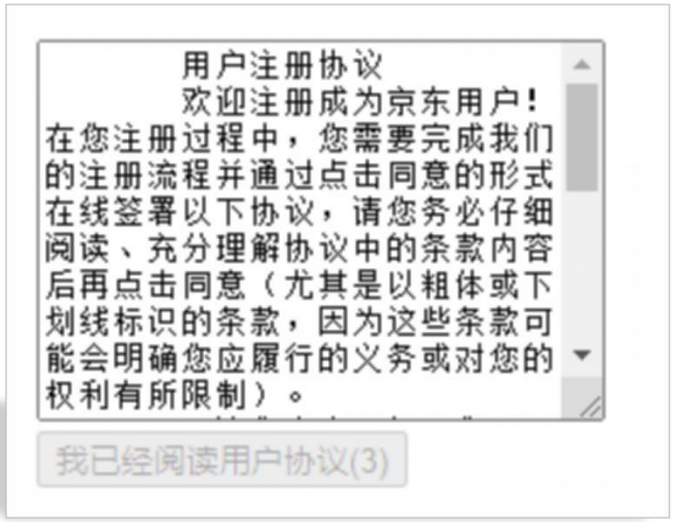



```
表单.value = '用户名'
表单.type = 'password'
```

4.3 操作表单元素属性

- 表单属性中添加就有效果，移除就没有效果，一律使用布尔值表示
- 比如实现禁用按钮，勾选按钮等
  - 如果为 true 代表添加了该属性
  - 如果是 false 代表移除了该属性
- 比如： disabled、checked、selected

全选	商品	商家	价格
☐	小米手机	小米	￥1999
☐	小米净水器	小米	￥4999
☐	小米电视	小米	￥5999



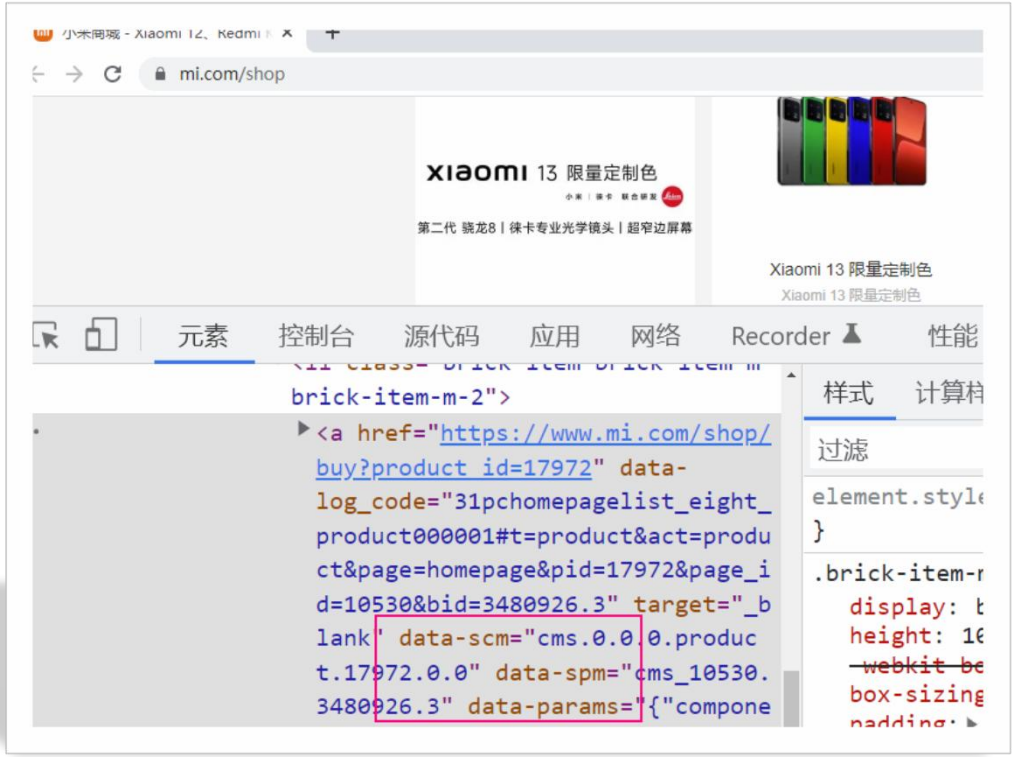
04

操作元素属性

- 操作元素**常用**属性
- 操作元素**样式**属性
- 操作 **表单元素** 属性
- 自定义属性

4.4 自定义属性

- **标准属性：**标签天生自带的属性 比如class、id、title等，可以直接使用点语法操作比如：对象.title
- **自定义属性：**
  - 在html5中推出来了专门的data-自定义属性



4.4 自定义属性

- **标准属性**：标签天生自带的属性 比如class、id、title等，可以直接使用点语法操作比如：对象.title
- **自定义属性**：
  - 在html5中推出来了专门的data-自定义属性
  - 使用场景：通过自定义属性可以存储数据，后期可以使用这个数据

就业榜

学号	姓名	年龄	性别	薪资	就业城市	操作
1	迪丽热巴	18	女	10000	北京	删除
2	古丽扎娜	19	女	11000	北京	删除
3	佟丽丫丫	20	女	12000	北京	删除
4	马尔扎哈	21	女	13000	北京	删除

```
▼<td>
    <a href="javascript:" data-id="0">删除</a>
  </td>
</tr>
<tr>
▼<td>
    <a href="javascript:" data-id="1">删除</a>
  </td>
</tr>
<tr>
▼<td>
    <a href="javascript:" data-id="2">删除</a>
  </td>
</tr>
<tr>
▼<td>
    <a href="javascript:" data-id="3">删除</a>
  </td>
```

4.4 自定义属性

- **标准属性**：标签天生自带的属性 比如class、id、title等，可以直接使用点语法操作比如：对象.title
- **自定义属性**：
  - 在html5中推出来了专门的data-自定义属性
  - 使用场景：通过自定义属性可以存储数据，后期可以使用这个数据
  - 在标签上一律以data-开头
  - 在DOM对象上一律以dataset对象方式获取

```
<body>
  <div class="box" data-id="10">盒子</div>
  <script>
    const box = document.querySelector('.box')
    console.log(box.dataset.id)
  </script>
</body>
```

目录

Contents

05

定时器-间隔函数

- 定时器函数介绍
- 定时器函数基本使用

五、定时器-间隔函数

- 网页中经常会需要一种功能：每隔一段时间需要自动执行一段代码，不需要我们手动去触发
- 例如：网页中的倒计时
- 要实现这种需求，需要定时器函数
- 定时器函数有两种：间隔函数 和 延迟函数

五、定时器-间隔函数

定时器函数可以开启和关闭定时器

1. 开启定时器

```
setInterval(函数, 间隔时间)
```

- 作用：每隔一段时间调用这个函数
- 注意：间隔时间单位是毫秒

举例说明

```
function repeat() {
    console.log('前端程序员，就是头发多咋滴~~')
}
// 每隔一秒调用repeat函数
setInterval(repeat, 1000)
```

注意：

1. 函数名字不需要加括号
2. 定时器返回的是一个id数字

五、定时器-间隔函数

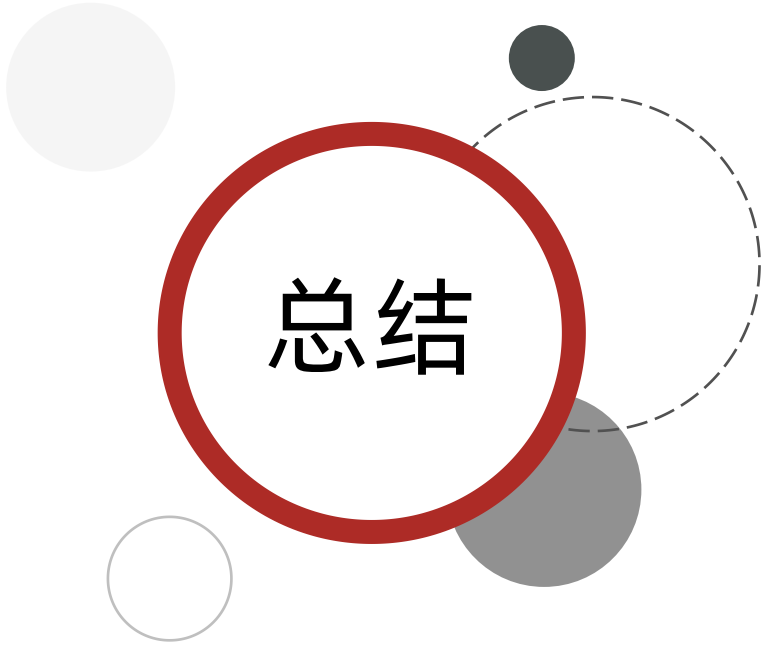
定时器函数可以开启和关闭定时器

2. 关闭定时器

```
let 变量名 = setInterval(函数, 间隔时间)
clearInterval(变量名)
```

一般不会刚创建就停止，而是满足一定条件再停止

```
let timer = setInterval(function() {
  console.log('hi~~')
}, 1000)
clearInterval(timer)
```



1. 定时器间隔函数有什么作用？
   - 可以每隔指定时间自动重复执行某些代码
2. 定时器函数如何开启？
   - setInterval(函数名, 时间)
3. 定时器函数如何关闭？
   - 需要定时器变量名来关闭
   - 返回的是一个唯一的数字

```
let 变量名 = setInterval(函数, 间隔时间)
clearInterval(变量名)
```

描述	属性/方法	效果
获取DOM对象	document.querySelector()	获取指定的第一个元素
	document.querySelectorAll()	获取指定的所有元素
操作元素内容	元素.innerText	操作元素内容，不解析标签
	元素.innerHTML	操作元素内容，解析标签
操作元素样式	元素.style.width	通过style操作样式
	元素.className	通过类名操作样式
	元素.classList.add()	增加类名
	元素.classList.remove()	删除类名
	元素.classList.toggle()	切换类名
间隔函数	setInterval(function() {}, 1000)	定时器，每隔指定时间重复执行

目录

Contents

案例

轮播图定时器版

效果需求：

①：每隔一秒钟自动切换图片、文本、颜色、小圆点

②：到了最后一张，自动切换到第1张

案例

轮播图定时器版

需求：每隔一秒钟切换一个图片

分析：和随机版本轮播图最大区别是自动按序播放

核心思想： 声明一个变量 i ，定时器1秒钟就 i++， 利用 i 来切换下一张图片

①：声明变量 i，开启定时器 i++，依次得到数组里面的下一个对象

②：获取相关元素

③：利用对象更换图片、文字、背景颜色、以及小圆点

④：处理图片自动复原从头播放（放到变量++后面，紧挨）

- 如果图片播放到最后一张（大于等于数组的长度）
- 则把变量 i 重置为0

案例 **轮播图定时器版**

小技巧

0	1	2	3	4	5	6	7	8	i
%	%	%	%	%	%	%	%	%	%
8	8	8	8	8	8	8	8	8	数组长度

0	1	2	3	4	5	6	7	0

传智教育旗下高端IT教育品牌