2-4 网页特效

- D1-PC端网页特效(目标).pptx <a href="https://www.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2Fw2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx?from=https%3A%2F%2Fwww.yuque.com/office/yuque/0/2022/pptx/1379492/1641891515910-af010f83-f2e9-4355-9aa7-ad942d765201.pptx/1379492/16418915-af010f83-f2e9-4355-af010f83-f2e9-4355-af010f83-f2e9-4355-af010f83-f2e9-4356-af010f83-f2e9-4356-af010f83-f2e9-4356-af010f83-f2e9-4356-af010f83-f2e9-4356-af010f83-f2e9-4356-af010f83-
- D 02-移动端网页特效(目标).pptx <https://www.yuque.com/office/yuque/0/2022/pptx/1379492/1641891533911-bc49d3d1-91ad-4b5c-a044-fca39b2834b9.pptx?from=https%3A%2F%2Fwww.yuque.... (150 KB) ◎5

一、PC 端网页特效

1 元素偏移量 offset 系列

offset 翻译过来就是偏移量,我们使用 offset 系列相关属性可以动态的得到该元素的位置 (偏移)、大小等

- 获得元素距离带有定位父元素的位置
- · 获得元素自身的大小 (宽度高度)

如果父盒子没有定位则偏移量以body为参考, 父盒子有定位则偏移量以父盒子为主

offsetParent offsetTop <u>边框</u> 内边距 内边距 OffsetHeight offsetWidth

offset 与 style 区别

offset

- offset 可以得到任意样式表中的样式值
- offset 系列获得的数值是没有单位的
- offsetWidth 包含padding+border+width
- offsetWidth 等属性是只读属性,只能获取不能赋值
- 所以,我们想要获取元素大小位置,用offset更合适

style

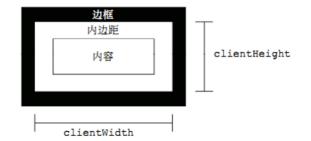
- style 只能得到行内样式表中的样式值
- style.width 获得的是带有单位的字符串
- style.width 获得不包含padding和border 的值
- style.width 是可读写属性,可以获取也可以赋值
- 所以,我们想要给元素更改值,则需要用style改变

2 元素可视区 client 系列

client 翻译过来就是客户端,我们使用 client 系列的相关属性来获取元素可视区的相关信息。通过 client 系列的相关属性可以动态的得到该元素的边框大小、元素大小等。

	client系列属性	作用		
	element.clientTop	返回元素上边框的大小		
	element.clientLeft	返回元素左边框的大小		
1	element.clientWidth	返回自身包括padding 、内容区的宽度,不含边框,返回数值不带单位		
1	element.clientHeight	返回自身包括padding 、内容区的高度,不含边框,返回数值不带单位		

client和offset最大的区别就是,client不包含边框(包含padding)



案例: 淘宝 flexible.js 源码分析

立即执行函数

1 (function() {})();

- 1 (function() {})(); 2 或者
- 3 (function() {}());

主要作用:创建一个独立的作用域,避免了命名冲突问题

里面的全部变量都是一个局部变量

C 笔记

JavaScript

```
// 1.立即执行函数: 不需要调用,立马能够自己执行的函数
3 = function fn() {
      console.log(1);
5
6
     fn();
     // 2. 写法 也可以传递参数进来
8
    // 1.(function() {})() 或者 2. (function(){}());
10 (function(a, b) {
11
      console.log(a + b);
      var num = 10;
12
    })(1,2); // 第二个小括号可以看做是调用函数
13
14
15 (function sum(a, b) {
16
      console.log(a + b);
17
      var num = 10; // 局部变量
    }(2, 3));
18
19 </script>
```

pageshow

下面三种情况都会刷新页面都会触发 load 事件

1. a 标签的超链接

2. F5或者刷新按钮 (强制刷新)

3. 前进后退按钮

但是火狐中,有个特点,有个"往返缓存",这个缓存中不仅保存着页面数据,还保存了DOM和JavaScript的状态;实际上是将整个页面都保存在了内存里,所以此时后退按钮不能刷新页面。

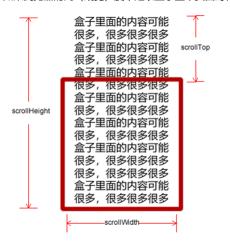
此时可以使用 pageshow 事件来触发,这个事件在页面显示时触发,无论页面是否来自缓存。在重新加载页面中,pageshow 会在 load 事件触发后触发;根据事件对象中的 persisted 来判断是否是缓存中的页面触发的pageshow 事件, 注意这个事件给window添加。

3 元素滚动 scroll 系列

scroll 翻译过来就是滚动的,我们使用 scroll 系列的相关属性可以动态的得到该元素的大小、滚动距离等。

scroll系列属性	作用	
element.scrollTop	返回被卷去的上侧距离,返回数值不带单位	
element.scrollLeft	返回被卷去的左侧距离,返回数值不带单位	
element.scrollWidth	返回自身实际的宽度,不含边框,返回数值不带单位	
element.scrollHeight	返回自身实际的高度,不含边框,返回数值不带单位	返回内容高度大小,如下图文字溢出, scroll显示的是里面文字的大小高度(即实际大

如果浏览器的高(或宽)度不足以显示整个页面时,会自动出现滚动条。当滚动条向下滚动时,页面上面被隐藏掉的高度,我们就称为页面被卷去的头部。滚动条在滚动时会触发 onscroll 事件。



案例: 仿淘宝固定右侧侧边栏

- 1. 原先侧边栏是绝对定位
- 2. 当页面滚动到一定位置,侧边栏改为固定定位
- 3. 页面继续滚动, 会让 返回顶部显示出来

案例分析

- 1. 需要用到页面滚动事件 scroll 因为是页面滚动,所以事件源是 document
- 2. 滚动到某个位置,就是判断页面被卷去的上部值。
- 3. 页面被卷去的头部:可以通过window.pageYOffset获得 如果是被卷去的左侧 window.pageXOffset
- 4.注意,元素被卷去的头部是 element.scrollTop , 如果是页面被卷去的头部 则是 window.pageYOffset
- 5. 其实这个值可以通过盒子的 offsetTop可以得到,如果大于等于这个值,就可以让盒子固定定位了

```
1 <style>
     .slider-bar {position: absolute;left: 50%;top: 300px;margin-left: 600px;
        width: 45px;height: 130px;background-color: pink;}
      .w {width: 1200px;margin: 10px auto;}
      .header {height: 150px;background-color: purple;}
      .banner {height: 250px;background-color: skyblue;}
      .main {height: 1000px;background-color: yellowgreen;}
      span {display: none;position: absolute;bottom: 0;}
10
11 = <div class="slider-bar">
12
     <span class="goBack">返回顶部</span>
13
    </div>
    <div class="header w">头部区域</div>
14
15 <div class="banner w">banner区域</div>
16 <div class="main w">主体部分</div>
17
18 - <script>
19
     //1. 获取元素
20
      var sliderbar = document.querySelector('.slider-bar');
      var banner = document.querySelector('.banner');
21
     // banner.offestTop 就是被卷去头部的大小 一定要写到滚动的外面
     var bannerTop = banner.offsetTop
23
24
      // 当我们侧边栏固定定位之后应该变化的数值
25
     var sliderbarTop = sliderbar.offsetTop - bannerTop;
26
     // 获取main 主体元素
27
      var main = document.querySelector('.main');
      var goBack = document.querySelector('.goBack');
28
29
      var mainTop = main.offsetTop;
30
      // 2. 页面滚动事件 scroll
31
32
      document.addEventListener('scroll', function() {
        // 3 . 当我们页面被卷去的头部大于等于了 172 此时 侧边栏就要改为固定定位
33
34 *
        if (window.pageYOffset >= bannerTop) {
35
          sliderbar.style.position = 'fixed';
36
          sliderbar.style.top = sliderbarTop + 'px';
37 =
          sliderbar.style.position = 'absolute';
38
          sliderbar.style.top = '300px';
39
40
        // 4. 当我们页面滚动到main盒子,就显示 goback模块
41
42 *
        if (window.pageYOffset >= mainTop) {
43
          goBack.style.display = 'block';
44 -
        } else {
45
         goBack.style.display = 'none';
46
47
      });
    </script>
```

页面被卷去的头部兼容性解决方案

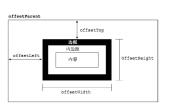
需要注意的是, 页面被卷去的头部, 有兼容性问题, 因此被卷去的头部通常有如下几种写法:

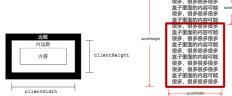
- ・声明了 DTD,使用 document.documentElement.scrollTop DTD:<!DOCTYPE html>代码页面头部加这一句话则是声明了DTD
- ・未声明 DTD, 使用 document.body.scrollTop
- · 新方法 window.pageYOffset 和 window.pageXOffset, IE9开始支持

```
1 function getScroll() {
       left: window.pageXOffset || document.documentElement.scrollLeft || document.body.scrollLeft|| 0,
       top: window.pageYOffset || document.documentElement.scrollTop || document.body.scrollTop || 0
     };
   使用的时候 getScroll().left
```

4 三大系列总结

三大系列大小对比	作用	
element.offsetWidth	返回自身包括padding 、 边框、内容区的宽度 , 返回数值不带单位	
element.clientWidth	返回自身包括padding 、内容区的宽度,不含边框,返回数值不带单位	
element.scrollWidth	返回自身实际的宽度,不含边框,返回数值不带单位	





offset是包含边框的,其他两个不包含边框 client返回的是自身的padding以及内容区宽度, 如果内容超出了盒子部分,则要使用scroll

他们主要用法:

- 1. offset系列经常用于获得元素位置 offsetLeft offsetTop
- 2. client 经常用于获取元素大小 clientWidth clientHeight
- 3. scroll经常用于获取滚动距离 scrollTop scrollLeft
- 4. 注意页面滚动的距离通过 window.pageXOffset 获得

5 mouseenter和mouseover的区别

当鼠标移动到元素上时就会触发 mouseenter 事件, 类似 mouseover , 它们两者之间的差别是

- · mouseover 鼠标经过自身盒子会触发,经过子盒子还会触发
- · mouseenter 只会经过自身盒子触发
- · 之所以这样,因为 mouseenter 不会冒泡,跟 mouseenter 搭配的鼠标离开 mouseleave 同样不会冒泡

6 动画函数封装

核心原理: 通过定时器 setInterval() 不断移动盒子位置

实现步骤

- 1. 获得盒子当前位置
- 2. 让盒子在当前位置加上1个移动距离
- 3. 利用定时器不断重复这个操作
- 4. 加一个结束定时器的条件

注意此元素需要添加定位,才能使用element.style.left

```
HTMI
    div {position: absolute; left: 0; width: 100px; height: 100px; background-color: pink; }
    </style>
                         要加定位才能动
5 <div></div>
7 = <script>
    // 动画原理
8
     // 1. 获得盒子当前位置
    // 2. 让盒子在当前位置加上1个移动距离
10
    // 3. 利用定时器不断重复这个操作
11
    // 4. 加一个结束定时器的条件
12
    // 5. 注意此元素需要添加定位,才能使用element.style.left
13
    var div = document.querySelector('div');
15 * var timer = setInterval(function() {
16 "
      if (div.offsetLeft >= 400) {
17
        // 停止动画 本质是停止定时器
18
        clearInterval(timer);
19
20
       div.style.left = div.offsetLeft + 1 + 'px';
21
    }, 30);
    </script>
```

动画函数简单封装

·注意函数需要传递2个参数, 动画对象和移动到的距离

动画函数给不同元素记录不同定时器

- ・如果多个元素都使用这个动画函数,每次都要 var 声明定时器。我们可以给不同的元素使用不同的定时器(自己专门用自己的定时器)。
- 核心原理: 利用 JS 是一门动态语言,可以很方便的给当前对象添加属性

```
HTMI
     div {position: absolute;left: 0;width: 100px;height: 100px;background-color: pink;}
    span {position: absolute;left: 0;top: 200px;display: block;
      width: 150px;height: 150px;background-color: purple;}
5
    </style>
    <button>点击夏雨荷才走</button>
8 <div></div>
    <span>夏雨荷</span>
10
11 * <script>
12
    // var obj = {};
13
     // obj.name = 'andy';
    // 简单动画函数封装obj目标对象 target 目标位置
     // 给不同的元素指定了不同的定时器
15
16
     function animate(obj, target) {
      // 当我们不断的点击按钮,这个元素的速度会越来越快,因为开启了太多的定时器
17
       // 解决方案就是 让我们元素只有一个定时器执行
18
       // 先清除以前的定时器, 只保留当前的一个定时器执
19
               val(obj timer);要先清除定时器,不然会出现越点越快的现象
20
       obj.timer = setInterval(function() { 用obj.timer命名定时器可以避免分不清哪个的定时器是哪个
21 _
22 =
         if (obj.offsetLeft >= target) {
          // 停止动画 本质是停止定时器
23
24
          clearInterval(obj.timer);
25
26
        obj.style.left = obj.offsetLeft + 1 + 'px';
27
28
       }, 30);
29
30
31
     var div = document.querySelector('div');
32
     var span = document.querySelector('span');
    var btn = document.querySelector('button');
33
    // 调用函数
34
                                                            匀速动画:盒子是当前位置+固定值 10
35
     animate(div, 300);
                                                            缓慢动画:盒子当前位置+变化的值(目标值-现在的位置)/10
    btn.addEventListener('click', function() {
36 -
37
      animate(span, 200);
38
    })
39 </script>
```

缓慢动画原理

1 让盒子每次移动的距离慢慢减小, 速度就会慢慢落下来

2.核心算法:(目标值-现在的位置)/10 作为每次移动的距离 也就是步长 3.停止条件是:让当前盒子位置等于目标位置就停止定时器 4.注意步长值要取整以避免小数精度丢失问题(Math.ceil向上取整,math.floor向下取整)

动画函数封装到单独JS文件里面: 因为以后经常使用这个动画函数,可以单独封装到一个JS文件里面,使用的时候引用这个JS文件即可。

1.单独新建一个JS文件 2.复制粘贴动画函数 3.引用<Script src="名字.js"><Script> 4.使用

节流阀:

防止轮播图按钮连续点击造成播放过快。

目的:

当上一个函数动画内容执行完毕,再去执行下一个函数动画,让事件无法连续触发。 核心实现思路:利用回调函数,添加一个变量来控制,锁住函数和解锁函数。

<https://service.weibo.com/share/share.php?url=https%3</p>

开始设置一个变量 var flag =true; %E7%A7%BB%E5%8A%A8%E7%AB%AF%E7%BD%91%E9%A1%B5%E7%89%B9%E6%95%88.pptx%E4%B8%80%E3%80%81PC%20%E7%AB%AF%E7%BD%91%E9 if(flag){flag=false;do something} 关闭水龙头

利用回调函数~动画执行完毕 , flag=true 打开水龙头

类名操作:ClassList返回元素的类名

eg:div.classlist[0]

添加类名:element.classList.add('类名')---是在后面添加类名,不会覆盖以前的类名,注意类名前面不要加.。 移除类名:element.classList.remove('类名') 切换类名:element.classList.toggle('类名')---判断有没有这个类,有的话就给你去掉,没有的话就给你加上。