

目标

目标

学习目标

轮播图

功能9: 实现左侧按钮功能

节流阀

仿美丽网页js效果

javascript书写方式

五个常用的输出语句

javascript的注释

变量

运算符

分支结构

循环结构

数组

函数

对象

JSON数据格式

内置String对象

内置Math对象

内置Date对象

内置Array对象

DOM对象

事件

事件高级-注册事件（2种方式）

事件对象

事件对象的使用

事件对象的属性和方法

鼠标事件对象

键盘事件

键盘事件对象

BOM对象

什么是BOM

BOM的构成

window对象

定时器

清除定时器

this指向问题

location 对象的属性

navigator对象

history对象

元素偏移量offset系列

元素可视区 client 系列

元素滚动 scroll 系列

页面被卷去的头部兼容性解决方案

今日总结

今日作业

学习目标

- 完成轮播剩余功能
- 完成仿美丽说网页的js功能
- 完成弹幕效果案例
- js阶段复习

轮播图

功能9: 实现左侧按钮功能

```
1  1. 把右侧按钮功能代码复制一份
2  2. 修改相关代码
3  3. 注意边界值的处理 当我们现在是第一张图片,我们点击了左侧按钮,应该马上不做动画最后一张,
4  然后再动画移动到倒数第二张图片的位置
5  4. 核心代码如下:
6  arrow_left.onclick = function(){
7      if(num == 0 ){
8          num = bannerUl.children.length-1;
9          bannerUl.style.left = -num*bannerWidth+"px"
10     }
11
12     num--;
13     animate( bannerUl , -num*bannerWidth );
14
15     currentCircle--;
16     if( currentCircle < 0 ){
17         currentCircle = circle.children.length - 1;
18     }
19
20     for(var j=0;j<circle.children.length;j++){
21         circle.children[j].removeAttribute("class");
22     }
23     circle.children[currentCircle].className = "current";
24 }
```

```

// 功能9: 实现左侧按钮功能,左侧按钮功能跟右侧按钮功能非常相似,所以我们可以复制右侧按钮的代码进行修改,但是需要注意修改边界值(难点)
arrow_left.onclick = function(){
    if( num == 0){ // 到达边界值,快速不做动画把图片移动到最后一张图片上,然后再动画移动到 倒数第二张图片 这样才可以实现无缝滚动
        // num边界值处理 num变量是控制当前第n个图片轮播 num =0,就代表第1个图片
        num = bannerUl.children.length - 1; // 7 - 1 = 6
        bannerUl.style.left = -num*bannerWidth + "px";
    }

    num--;
    animate(bannerUl, -num*bannerWidth );

    // 点击左侧按钮,小圆圈跟随变化
    circleCurrent--;

    // 排他思想
    // circle.children就是代表circle层下的所有li标签
    for(var k=0;k<circle.children.length;k++){
        // 清除所有li的class属性值
        circle.children[k].removeAttribute("class");
    }

    // circleCurrent边界值的处理 circleCurrent控制当前第几个小圆点 circleCurrent等于0的时候,代表第一个小圆点
    if(circleCurrent < 0 ){ // 当小圆点小于0的时候,应该把circleCurrent设置为最后一个小圆点
        circleCurrent = circle.children.length - 1;
    }

    // 单独设置点击的那个li的class属性值
    circle.children[circleCurrent].className = "current";
}

```

功能10: 我们观察发现左侧按钮跟右侧按钮都有高亮某个小圆点的代码,所以我们可以对代码进行优化,把相同的代码放在一个函数中

封装一个circleChange函数,然后在左侧按钮跟右侧按钮的代码中调用circleChange函数即可

```

// 功能10: 我们观察发现左侧按钮跟右侧按钮都有高亮某个小圆点的代码,所以我们可以对代码进行优化,把相同的代码放在一个函数中
// 封装一个circleChange函数,然后在左侧按钮跟右侧按钮的代码中调用circleChange函数即可
function circleChange(){
    // 排他思想
    // circle.children就是代表circle层下的所有li标签
    for(var k=0;k<circle.children.length;k++){
        // 清除所有li的class属性值
        circle.children[k].removeAttribute("class");
    }
    // 单独设置点击的那个li的class属性值
    circle.children[circleCurrent].className = "current";
}

```

```

arrow_right.onclick = function(){
    // 功能5: 点击右侧按钮一次, 就让图片滚动一张
    // 核心原理: 就是点一次右侧按钮, 就让u1负一次1200px
    // 1. 点击右侧按钮一次, 就让图片滚动一张。
    // 2. 声明一个变量num, 点击一次, 自增1, 让这个变量乘以图片宽度, 就是 u1 的滚动距离。
    // 3. 图片无缝滚动原理
    // 4. 把u1 第一个li 克隆一份, 放到u1 的最后面, 需要注意修改u1的宽度以及生成小圆点的个数
    // 5. 当图片滚动到克隆的最后一张图片时, 让u1 快速的、不做动画的跳到最左侧: 也就是把 left 为0
    // 6. 同时num 赋值为0, 可以重新开始滚动图片了
    // console.log( num );

    if(num == bannerUl.children.length - 1){ // 代表移动到我们克隆的那张图片
        bannerUl.style.left = 0;
        num = 0;
    }
    num++;
    animate(bannerUl, -num*bannerWidth );

    // 功能7: 点击右侧按钮, 小圆圈跟随变化
    circleCurrent++;
    // console.log( circleCurrent );

    if( circleCurrent == circle.children.length){
        circleCurrent = 0;
    }

    // 调用改变小圆点功能的函数
    circleChange();
}

```

```

arrow_left.onclick = function(){
    if( num == 0 ){ // 到达边界值, 快速不做动画把图片移动到最后一张图片上, 然后再动画移动到倒数第二个图片 这样才可以实现无缝滚动
        // num边界值处理 num变量是控制当前第n个图片轮播 num = 0, 就代表第1个图片
        num = bannerUl.children.length - 1; // 7 - 1 = 6
        bannerUl.style.left = -num*bannerWidth + "px";
    }

    num--;
    animate(bannerUl, -num*bannerWidth );

    // 点击左侧按钮, 小圆圈跟随变化
    circleCurrent--;

    // circleCurrent边界值的处理 circleCurrent控制当前第几个小圆点 circleCurrent等于0的时候, 代表第一个小圆点
    if(circleCurrent < 0 ){ // 当小圆点小于0的时候, 应该把circleCurrent设置为最后一个小圆点
        circleCurrent = circle.children.length - 1;
    }

    // 调用改变小圆点功能的函数
    circleChange();
}

```

功能11: 自动播放功能

- 1 1. 自动播放功能
- 2 2. 添加一个定时器
- 3 3. 自动播放轮播图, 实际就类似点击了右侧按钮
- 4 4. 此时我们可以手动调用右侧按钮点击事件 语法: 对象.click()

- 5 5. 鼠标经过.banner就停止定时器
- 6 6. 鼠标离开.banner就开启定时器

```
// 功能11: 自动播放功能
// 1. 自动播放功能
// 2. 添加一个定时器
// 3. 自动播放轮播图,实际就类似点击了右侧按钮
// 4. 此时我们可以"手动调用右侧按钮点击"事件 语法:对象.click() 手动触发事件
// 5. 鼠标经过.banner就停止定时器
// 6. 鼠标离开.banner就开启定时器
var timer = window.setInterval(function(){
    // 手动调用右侧按钮点击事件
    arrow_right.click();
},1500);
```

```
// 功能1:鼠标经过轮播图模块,左右按钮显示,离开隐藏左右按钮
banner.onmouseover = function(){
    arrow_left.style.display = "block";
    arrow_right.style.display = "block";

    // 功能11: 鼠标移上.banner就停止定时器 timer是一个全局变量,在函数内是可以正常使用的
    clearInterval(timer);
}

banner.onmouseout = function(){
    arrow_left.style.display = "none";
    arrow_right.style.display = "none";
    // 功能11: 鼠标移出.banner继续开启定时器 timer是一个全局变量,在函数内是可以正常使用的
    timer = window.setInterval(function(){
        // 手动调用右侧按钮点击事件
        arrow_right.click();
    },1500);
}
```

功能12: 最后,通过.banner设置css的overflow:hidden样式,把超过宽度的图片隐藏起来

```
.banner{
    height: 400px;
    position: relative;
    /* 超出banner的内容隐藏 */
    overflow: hidden;
    margin-top: 50px;
}
```

整个轮播图效果,完整代码请查看老师源代码

如果出现了鼠标移上轮播图层停止不了定时器,可以尝试以下方式解决

```
// 功能1:鼠标经过轮播图模块,左右按钮显示,离开隐藏左右按钮
banner.onmouseover = function(){
    arrow_left.style.display = "block";
    arrow_right.style.display = "block";

    // 功能11: 鼠标移上.banner就停止定时器 timer是一个全局变量,在函数内是可以正常使用的
    clearInterval(timer);
    // 多加一句代码,设置timer的变量为空
    timer = null;
}

banner.onmouseout = function(){
    arrow_left.style.display = "none";
    arrow_right.style.display = "none";
    // 功能11: 鼠标移出.banner继续开启定时器 timer是一个全局变量,在函数内是可以正常使用的

    if( timer == null ){// 判断timer变量是否为null,为null才开启定时器
        timer = window.setInterval(function(){
            // 手动调用右侧按钮点击事件
            arrow_right.click();
        },1500);
    }
}
```

节流阀

目的: 防止轮播图按钮连续点击造成播放过快。

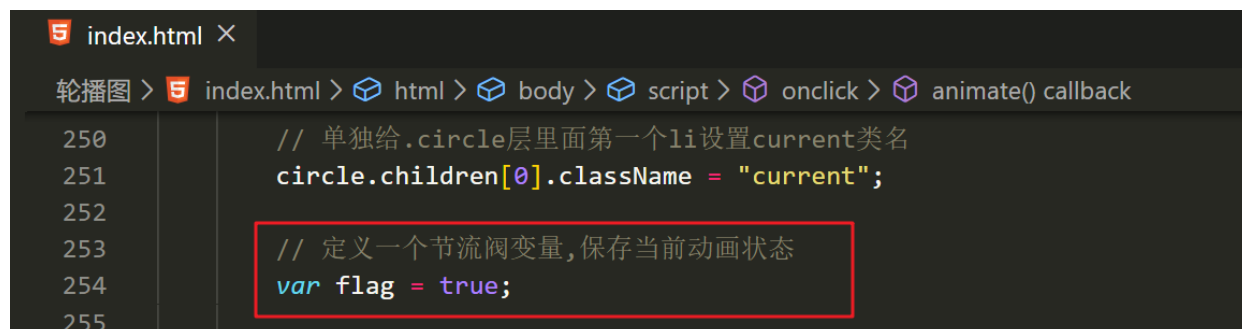
节流阀目的:当上一个函数动画内容执行完毕，再去执行下一个函数动画，让事件无法连续触发。

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。

开始设置一个变量var flag= true;

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



```
index.html ×
轮播图 > index.html > html > body > script > onclick > animate() callback
250 // 单独给.circle层里面第一个li设置current类名
251 circle.children[0].className = "current";
252
253 // 定义一个节流阀变量,保存当前动画状态
254 var flag = true;
255
```



```

arrow_right.onclick = function(){
    // 功能5: 点击右侧按钮一次, 就让图片滚动一张
    // 核心原理: 就是点一次右侧按钮, 就让u1负一次1200px
    // 1. 点击右侧按钮一次, 就让图片滚动一张。
    // 2. 声明一个变量num, 点击一次, 自增1, 让这个变量乘以图片宽度, 就是 u1 的滚动距离。
    // 3. 图片无缝滚动原理
    // 4. 把u1 第一个li 克隆一份, 放到u1 的最后面, 需要注意修改u1的宽度以及生成小圆点的个数
    // 5. 当图片滚动到克隆的最后一张图片时, 让u1 快速的、不做动画的跳到最左侧: 也就是把 left 为0
    // 6. 同时num 赋值为0, 可以重新开始滚动图片了
    // console.log( num );

    // 节流阀
    if( flag ){ // 当flag变量为true的时候, 才能点击右侧按钮实现动画
        flag = false; // 改变flag为false, 关闭节流阀

        if( num == bannerUl.children.length - 1 ){ // 代表移动到我们克隆的那张图片
            bannerUl.style.left = 0;
            num = 0;
        }
        num++;
        animate(bannerUl, -num*bannerWidth, function(){
            // 当动画执行完毕以后, 重新开启节流阀
            flag = true;
        });

        // 功能7: 点击右侧按钮, 小圆圈跟随变化
        circleCurrent++;
        // console.log( circleCurrent );

        if( circleCurrent == circle.children.length ){
            circleCurrent = 0;
        }

        // 调用改变小圆点功能的函数
        circleChange();
    }
}
}

```

```

// 功能9: 实现左侧按钮功能,左侧按钮功能跟右侧按钮功能非常相似,所以我们可以复制右侧按钮的代码进行修改,但是需要注意修改边界值(难点)
arrow_left.onclick = function(){
    // 节流阀
    if(!flag){
        flag = false; // 关闭节流阀

        if( num == 0){ // 到达边界值,快速不做动画把图片移动到最后一张图片上,然后再动画移动到倒数第二个图片 这样才可以实现无缝滚动
            // num边界值处理 num变量是控制当前第n个图片轮播 num = 0,就代表第1个图片
            num = bannerUl.children.length - 1; // 7 - 1 = 6
            bannerUl.style.left = -num*bannerWidth + "px";
        }

        num--;
        animate(bannerUl, -num*bannerWidth, function(){
            // 动画执行完毕以后,重新开启节流阀
            flag = true;
        });

        // 点击左侧按钮,小圆圈跟随变化
        circleCurrent--;

        // circleCurrent边界值的处理 circleCurrent控制当前第几个小圆点 circleCurrent等于0的时候,代表第一个小圆点
        if(circleCurrent < 0 ){ // 当小圆点小于的时候0,应该把circleCurrent设置为最后一个小圆点
            circleCurrent = circle.children.length - 1;
        }

        // 调用改变小圆点功能的函数
        circleChange();
    }
}

```

大家可以参考一下,更高级的js动画函数

<https://www.cnblogs.com/sunyan-blog/p/12073937.html>

仿美丽网页js效果

功能1: 宝贝、店铺点击可以相互切换



```
/* .logo .search .top span{
    width: 56px;
    height: 24px;
    background: #ff3366;
    float: left;
    text-align: center;
    line-height: 24px;
    color:white;
    cursor: pointer;
}
.logo .search .top span:last-child{
    background: #f2f2f2;
    color:#666666;
} */
```

修改index.css文件

```
.logo .search .top span{
    width: 56px;
    height: 24px;
    background: #f2f2f2;
    color:#666666;
    float: left;
    text-align: center;
    line-height: 24px;
    cursor: pointer;
}
/* 交集选择器,是span标签,并且span标签具有.current */
.logo .search .top span.current{
    background: #ff3366;
    color:white;
}
```

```
<div class="search fl">
  <div class="top">
    <span>宝贝</span>
    <span class="current">店铺</span>
  </div>

  <script>
    var searchSpans = document.querySelectorAll(".search .top span");
    for( var i=0 ; i<searchSpans.length ; i++ ){
      searchSpans[i].onclick = function(){
        // 排他思想
        for(var j=0;j<searchSpans.length;j++){
          searchSpans[j].removeAttribute("class");
        }
        this.className = "current";
      }
    }
  </script>
```

功能2: 选项卡



```
index.html index.css X
作业讲评-仿美丽说 > css > index.css > ...
410 .today_new_goods .title ul li:first-child{
411     margin-right: 38px;
412 }
413 /* .today_new_goods .title ul li:first-child span{
414     border-bottom: 2px solid #f36;
415     padding-bottom: 2px;
416 }
417 .today_new_goods .title ul li:first-child a{
418     color:#f36;
419 } */
420
421 .today_new_goods .title ul li.current span{
422     border-bottom: 2px solid #f36;
423     padding-bottom: 2px;
424 }
425 .today_new_goods .title ul li.current a{
426     color:#f36;
427 }
428
```

```
作业讲评-仿美丽说 > index.html > html > body > script > checker
186 <div class="today_new_goods w">
187     <!-- 标题部分 -->
188     <div class="title">
189         <a href="#" class="fr">查看更多</a>
190         <ul>
191             <li class="current"><a href="javascript:void(0)">今<span>日新</span>品
192             </a></li>
193             <li><a href="javascript:void(0)">一<span>周热</span>销</a></li>
194         </ul>
195     </div>
196
197     <!-- 内容部分 -->
198     <div class="content" style="display:block;">...
199 </div>
200
201     <!-- 再复制一遍.content层 -->
202     <div class="content" style="display:none;">...
203 </div>
204 </div>
```

```

<script>
    // 选项卡功能
    // 获取相关对象
    var lis = document.querySelectorAll(".today_new_goods .title ul li");
    var contents = document.querySelectorAll(".today_new_goods .content");

    for(var i=0;i<lis.length;i++){
        // 设置自定义属性
        lis[i].setAttribute("data-index",i);

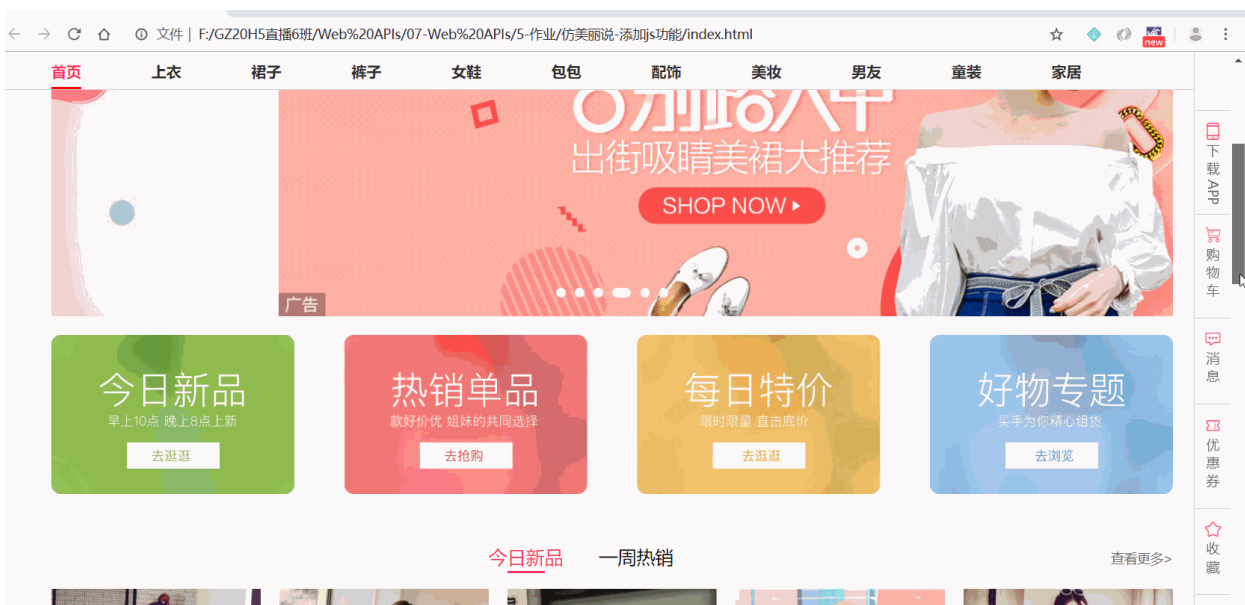
        lis[i].onclick = function(){
            // 排他思想
            for(var j=0;j<lis.length;j++){
                // 去掉所有的li上的类名
                lis[j].className = "";
                // 隐藏所有.content层
                contents[j].style.display = "none";
            }

            this.className = "current";

            // 获取li上面的自定义属性
            var index = this.getAttribute("data-index");
            // 显示li对应的.content层
            contents[index].style.display = "block";
        }
    }
</script>

```

功能3:导航栏吸顶效果



```
index.html index.css ×
作业讲评-仿美丽说 > css > index.css > .main_nav ul li
193 .main_nav{
194     height: 30px;
195     border-bottom: 1px solid #dedede;
196     font-size: 16px;
197 }
198 /* 专门定义一个类,提供给.main_nav固定定位以后使用 */
199 .main_nav_fixed{
200     position: fixed;
201     left: 0;
202     top: 0;
203     width: 100%;
204     z-index: 1;
205     background: white;
206 }
```

```
<script>
// 获取对象
var main_nav = document.querySelector(".main_nav");
// 获取.main_nav上外边距
var main_nav_top = main_nav.offsetTop;

// 功能3: .main_nav吸顶功能 滚动到页面一定位置的时候,.main变成固定定位
window.onscroll=function(){
    // 获取被页面卷去的距离 怎么获取
    var scrollTop = window.pageYOffset || document.body.scrollTop || document.documentElement.scrollTop;

    // 判断被页面卷去的距离大于 .main_nav上外边距 就设置.main_nav为固定定位
    if(scrollTop > main_nav_top){
        // 因为设置固定定位以后,会脱标,具有行内块特性,默认宽度由内容决定,会根据body来定位
        // main_nav.style.position = "fixed";
        // main_nav.style.left = "0px";
        // main_nav.style.top = "0px";
        // 固定定位以后,这个时候,设置百分比,是根据body
        // main_nav.style.width = "100%";
        // main_nav.style.zIndex = "1";
        // main_nav.style.backgroundColor = "white";

        // 加上main_nav_fixed类名
        main_nav.className = "main_nav main_nav_fixed";
    }else{
        main_nav.className = "main_nav";
    }
}
</script>
```

功能4: 动画回到顶部



```
<script>
// 获取对象
var main_nav = document.querySelector(".main_nav");
// 获取.main_nav上外边距
var main_nav_top = main_nav.offsetTop;

// 功能3: .main_nav吸顶功能 滚动到页面一定位置的时候,.main变成固定定位
window.addEventListener("scroll",function(){
// 获取被页面卷去的距离 怎么获取
var scrollTop = window.pageYOffset || document.body.scrollTop || document.documentElement.scrollTop;

// 判断被页面卷去的距离大于 .main_nav上外边距 就设置.main_nav为固定定位
if(scrollTop > main_nav_top){
// 因为设置固定定位以后,会脱标,具有行内块特性,默认宽度由内容决定,会根据body来定位
// main_nav.style.position = "fixed";
// main_nav.style.left = "0px";
// main_nav.style.top = "0px";
// 固定定位以后,这个时候,设置百分比,是根据body
// main_nav.style.width = "100%";
// main_nav.style.zIndex = "1";
// main_nav.style.backgroundColor = "white";

// 加上main_nav_fixed类名
main_nav.className = "main_nav main_nav_fixed";
}else{
main_nav.className = "main_nav";
}

if( scrollTop > 0 ){// 判断被页面卷去的距离是否大于0
backup.style.display = "block";// 显示向上按钮
}else{
backup.style.display = "none";// 隐藏向上按钮
}
})
}</script>
```

改为了事件监听绑定


```
index.html × JS animate_scroll.js index.css
作业讲评-仿美丽说 > index.html > html > body
207 <li><a href="#"><span class="iconfont">&#xe72c;</span>消息</a></li>
208 <li><a href="#"><span class="iconfont">&#xe639;</span>优惠券</a></li>
209 <li><a href="#"><span class="iconfont">&#xe610;</span>收藏</a></li>
210 <li><a href="#"><span class="iconfont">&#xe638;</span>足迹</a></li>
211 <!-- 因为a链接的href设置为#以后,点击a就会页面马上回到顶部,这是a的默认事件 -->
212 <li><a class="backup" href="#" style="display: none;"><span
class="iconfont">&#xe633;</span></a></li>
213
214 <!-- 解决方式一: href改为javascript:void(0) -->
215 <!-- <li><a class="backup" href="javascript:void(0)" style="display: none;
"><span class="iconfont">&#xe633;</span></a></li> -->
216 </ul>
217 </div>
218
```

```
<!-- 引入动画滚动函数 -->
<script src="../../js/animate_scroll.js"></script>
<script>
// 功能4: 回到顶部
// 获取对象
var backup = document.querySelector(".backup");
// 对象.on事件类型 这样的绑定方式叫做传统注册方式 如果一个对象一个事件,传统方式绑定了多次,以最后那个为准
// 但是如果使用的是事件监听的话,同一个对象同一个事件,可以依次按顺序触发
window.addEventListener("scroll",function(){
// 获取被页面卷去的距离 怎么获取
var scrollTop = window.pageYOffset || document.body.scrollTop || document.documentElement.scrollTop;
if( scrollTop > 0 ){// 判断被页面卷去的距离是否大于0
backup.style.display = "block";// 显示向上按钮
}else{
backup.style.display = "none";// 隐藏向上按钮
}
});

// 给回到顶部按钮绑定单击事件
backup.onclick = function(e){
// 解决方式二 阻止a连接href="#"的默认行为
// 阻止默认行为 的方式有哪些?

// 也可以使用事件对象属性和方法阻止默认行为
var e = e || window.event;
e.preventDefault();
// e.returnValue = false;

animate_scroll(window,0);

// return false;// return false虽然可以阻止默认行为,但是return false后面的代码就不会执行了
}
</script>
```

功能5: 轮播图(可以不做自动播放的功能,但是其他功能要完成)



实现步骤

1. 把上课做的轮播移植仿美丽说网页即可
2. 移植HTML结构
3. 移出CSS样式
4. 可以把相关代码放进一个js文件中,再引入这个js文件

javascript书写方式

嵌入式: 写在<script></script>标签中

外链式: 写在js文件中,通过<script src="js文件所在路径"></script>

行内式: <div onclick='alert("点击了我")'>div的内容</div>

五个常用的输出语句

document.write() 在页面上输出内容,如果输出的内容是标签,一样会解析标签

console.log() 在控制台输出内容

window.alert() 弹窗

window.prompt() 弹出一个输入框,供用户输入,得到的值是String类型

window.confirm() 确认框,点击确定,返回true,点击取消,返回false

window.可以省略,也就是alert(),prompt(),confirm()

javascript的注释

- 1 单行注释: // 注释内容
- 2
- 3 多行注释: /*注释内容*/

变量

什么是变量

如何使用变量

变量的命名规则和规范

变量的数据类型分为：两大类、七小种

- 基本数据类型 只能存储一个值
- 复合数据类型 至少存储一个值，可以存储多个值

如何获取变量的数据类型

运算符

算术运算符:常见的有+,-,*,/等

++ 自增运算符 前++跟后++ 共同点:最后的结果都会加1 不同的是,有其他代码的时候

前++,会自加一,再赋值 后++,会赋值,再自加一

-- 自减运算符 前-- 跟后 -- 共同点:最后的结果都会减1 不同的是,有其他代码的时候

前--,会自减一,再赋值 后--,会赋值,再自减一

% 取余(求模)

$10\%2 = 0$

$11\%2 = 1$

+号运算符

赋值运算符:

逻辑运算符

&& 逻辑与 两个条件都是true,才为true

|| 逻辑或 两个条件,只要有一个是true,最后结果就为true

! 逻辑非(取反) !true => false !false => true

分支结构

①单分支

```
1  if(条件表达式){  
2    成立执行的代码块;  
3  }
```

②双分支

```
1  if(条件表达式){  
2    成立执行的代码块;  
3  }else{  
4    不成立执行的代码块;  
5  }
```

③多分支

```
1  if(条件表达式1){  
2    条件1成立执行的代码块;  
3  }else if(条件表达式2){  
4    条件1不成立,但是条件2成立            执行的代码块;  
5  }else if(条件表达式3){  
6    条件1不成立,条件2也不成立,但是条件3成立            成立执行的代码块;  
7  }else{  
8    上述条件都不成立执行的代码块;  
9  }
```

④switch语句

```
1  switch(值){  
2    case 值1:  
3      当值全等于值1的时候,执行的代码块  
4      break;  
5    case 值2:
```

```
6         当值全等于值2的时候,执行的代码块
7         break;
8     case 值3:
9         当值全等于值3的时候,执行的代码块
10        break;
11    default:
12        找不到对应case,执行的代码块
13 }
```

循环结构

①for循环

```
1  for(初始化;判断条件;改变变量的值){
2      //循环体
3  }
```

②while循环

```
1  初始化;
2  while(判断条件){
3      //循环体
4      改变变量的值
5  }
```

③do...while循环

```
1  初始化;
2  do{
3      //循环体
4      改变变量的值
5  }while(判断条件)
```

④循环中断关键字

数组

定义数组

访问数组中某个元素

数组长度

遍历数组

二维数组的定义与二维数组的遍历

函数

定义函数

调用函数

return关键字

匿名函数

变量的作用域 分为： 和

对象

对象的分类

创建自定义对象

①使用{}

②new Object

③构造函数创建自定义对象

JSON数据格式

JSON格式的定义

定义JSON对象

访问 JSON对象的数据

遍历JSON对象的属性

JSON数组 就是把多个json对象放在一个数组中

内置String对象

如何创建String对象?

如何获取字符串的长度

字符串[下标]

字符串.substr(start[,length])

字符串.indexOf("要查找的字符或字符串")

字符串.split(分割符)

内置Math对象

Math.floor(数值)

Math.ceil(数值)

Math.random()

Math.floor(Math.random()*(最大值-最小值)+最小值)

```
Math.floor(Math.random()*(最大值-最小值+1)+最小值)
```

内置Date对象

如何创建Date对象？

如果我们想要 2008-08-08 08:08:08 格式怎么办？

我们需要^{手动}的得到这种格式

方法名	功能
getFullYear()	获取4位数的年份
getMonth()	获取月份 返回值 0~11 0表示1月 11表示12月
getDate()	返回一个月中的某一天 返回值：1~31
getHours()	小时 返回值0~23
getMinutes()	获取分钟 返回值：0~59
getSeconds()	获取秒数 返回值：0~59
getMilliseconds()	获取毫秒 返回值：0~999
getDay()	获取一周中的某一天 就是星期几 返回值： 0~6 0代表星期天 1代表星期一
getTime()	获取时间戳 返回从1970年1月1日 一直到现在的 毫秒数 ！
toLocaleString()	根据本地时间把 Date 对象转换为字符串，并返回结果。

注意 月份 和 星期 取值范围是从 0开始的。

`getDay` 方法所返回的值是一个处于 0 到 6 之间的整数，它代表了一周中的某一天，返回值与一周中日期对应关系如下：

值	星期
0	星期天
1	星期一
2	星期二
3	星期三
4	星期四
5	星期五
6	星期六

内置Array对象

如何创建Array对象,如何创建数组

检查是否为数组

数组添加和删除元素方法

方法名	说明	返回值
push (参数1..)	修改原数组，末尾添加一个或多个数组元素	并返回新的长度
pop()	删除 数组的最后一个元素，把数组长度减 1 无参数	返回它删除的元素的值
unshift(参数 1...)	向数组的开头添加一个或更多数组元素	并返回新的长度
shift()	把数组的第一个元素从其中删除，把数组长度减 1 无参数	返回它删除的元素的值

数组的所有元素合并到一个字符串中

方法名	说明	返回值
toString()	把数组转成字符串,逗号分隔每一项	返回一个字符串
join('分隔符')	用于把数组中的所有元素以 分隔符连接 ,合并为一个字符串; 如果这个参数分隔符没有写 则默认使用英文状态下的 逗号 进行连接	返回一个字符串

数组截取slice、数组删除splice

方法名	说明	返回值
slice(begin下标,end下标)	数组截取	返回截取出来的新数组 包括 begin，不包括 end ,这个 不会 影响原数组
splice (从第几个开始,要删除几个)	数组删除	返回被删除元素组成的数组 ,这个 会 影响原数组

DOM对象

获取对象的多种方法

描述	方法
根据ID获取元素	
根据标签名获取元素	
根据标签名获取元素	
根据类名获取元素	

根据CSS选择器获取元素	
根据CSS选择器获取元素	
获取body元素	
获取html元素	

DOM改变元素内容

DOM操作元素的属性

DOM操作元素的样式

自定义属性操作

- 获取属性值
- 设置属性值
- 移除属性

节点操作

- 父级节点
- 子节点
- 创建节点
- 添加节点
- 删除节点

事件

事件三要素

事件的绑定方式 行内绑定方式 和 动态绑定方式

常见的鼠标事件

鼠标事件	触发条件
onclick	鼠标点击左键触发
onmouseover	鼠标经过触发

onmouseout	鼠标离开触发
onfocus	获得鼠标焦点触发
onblur	失去鼠标焦点触发
onmousemove	鼠标移动触发
onmouseup	鼠标弹起触发
onmousedown	鼠标按下触发

事件高级-注册事件（2种方式）

给元素添加事件,称为注册事件或者绑定事件

注册事件有两种方式:传统方式和监听注册方式

传统注册方式

- 利用 on 开头的事件 onclick
- `<button onclick="alert('hi~')"></button>`
- `btn.onclick = function() {}`
- 特点: 注册事件的**唯一性**
- 同一个元素同一个事件只能设置一个处理函数,最后注册的处理函数将会覆盖前面注册的处理函数

监听注册方式

- w3c 标准 推荐方式
- `addEventListener()` 它是一个方法
- IE9 之前的 IE 不支持此方法,可使用 `attachEvent()` 代替
- 特点: 同一个元素同一个事件可以注册多个监听器
- 按注册顺序依次执行

删除事件（解绑事件）

1. 传统注册方式

```
1 eventTarget.事件名称 = null;  
2  
3 比如:eventTarget.onclick = null;
```

2. 方法监听注册方式

```
1 ① eventTarget.removeEventListener(type, listener[, useCapture]);  
2 ② eventTarget.detachEvent(eventNameWithOn, callback);
```

事件对象

事件发生后,跟事件相关的一系列信息数据的集合都放到这个对象里面,这个对象就是事件对象。

比如:

1. 谁绑定了这个事件。
2. 鼠标触发事件的话，会得到鼠标的相关信息，如鼠标位置。
3. 键盘触发事件的话，会得到键盘的相关信息，如按了哪个键。

事件对象的使用

事件触发发生时就会产生事件对象，并且系统会以实参的形式传给事件处理函数。

所以，在事件处理函数中声明1个形参用来接收事件对象。

```
1 eventTarget.onclick = function(event){
2     // 这个event 就是事件对象,我们还喜欢写成e或者evt
3 };
4
5 eventTarget.addEventListener("click",function(event){
6     // 这个event 就是事件对象,我们还喜欢写成e或者evt
7 });
8
9 eventTarget.attachEvent("onclick",function(event){
10    // 这个event 就是事件对象,我们还喜欢写成e或者evt
11 });
```

事件对象的属性和方法

事件对象属性方法	说明
e.target	返回触发事件的对象 标准
e.stopPropagation()	该方法阻止冒泡 标准
e.preventDefault()	该方法阻止默认事件(默认行为) 标准 比如不让链接跳转
return false	利用return false 也能阻止默认行为 没有兼容问题 但是return后面的代码不执行

鼠标事件对象

event 事件对象是事件相关的一系列信息的集合。

现阶段我们主要用鼠标事件对象MouseEvent 和 键盘事件对象KeyboardEvent

鼠标事件对象	说明
e.pageX	返回鼠标相对于 文档页面 的X坐标 IE9以后支持
e.pageY	返回鼠标相对于 文档页面 的Y坐标 IE9以后支持

键盘事件

键盘时间	触发条件
onkeyup	某个键盘按键被松开时触发
onkeydown	某个键盘按键被按下时触发
onkeypress	某个键盘按键被按下时触发 但是它不识别功能键 比如 ctrl shift 箭头等

键盘事件对象

需要传event事件对象参数才可以使用

键盘事件对象属性	说明
keyCode	返回 该 键的ASCII值

注意:

1. onkeydown和onkeyup不区分字母大小写,onkeypress区分字母大小写
2. 在我们实际开发中,我们更多的使用keydown和keyup, 它能识别所有的键(包括功能键)
3. keypress不识别功能键,但是keyCode属性能够区分大小写,返回不同的ASCII值

BOM对象

什么是BOM

BOM (Browser Object Model) 即浏览器对象模型, 它提供了独立于内容而与浏览器窗口进行交互的对象, 其核心对象是 **window**。

BOM 由一系列相关的对象构成, 并且每个对象都提供了很多方法与属性。

BOM 缺乏标准，JavaScript 语法的标准化组织是 ECMA，DOM 的标准化组织是 W3C，BOM 最初是 Netscape 浏览器标准的一部分。

DOM

- 文档对象模型
- DOM 就是把「文档」当做一个「对象」来看待
- DOM 的顶级对象是 **document**
- DOM 主要学习的是操作页面元素
- DOM 是 W3C 标准规范

BOM

- 浏览器对象模型
- 把「浏览器」当做一个「对象」来看待
- BOM 的顶级对象是 **window**
- BOM 学习的是浏览器窗口交互的一些对象
- BOM 是浏览器厂商在各自浏览器上定义的，兼容性较差

BOM的构成

BOM 比 DOM 更大，它包含 DOM。



window对象

页面（窗口）加载事件

```
1 window.onload = function(){}  
2 或者  
3 window.addEventListener("load",function(){});
```

定时器

清除定时器

this指向问题

this的指向在函数定义的时候是确定不了的，只有函数执行的时候才能确定this到底指向谁，一般情况下this的最终指向的是那个调用它的对象。

现阶段，我们先了解一下几个this指向

- 1. 全局作用域或者普通函数中this指向全局对象window（注意延时器跟定时器里面的this指向window）
- 2. 对象里面的方法,谁调用, this就指向谁
- 3. 构造函数中this指向构造函数的实例

location 对象的属性

location对象属性	返回值
location.href	获取或者设置 URL

navigator对象

navigator 对象包含有关浏览器的信息，它有很多属性，我们最常用的是 **userAgent**，该属性可以返回由客户机发送服务器的 user-agent 头部的值。

下面前端代码可以判断用户那个终端打开页面，实现跳转

```
1  if((navigator.userAgent.match(/(phone|pad|pod|iPhone|iPod|ios|iPad|Android|Mobile|BlackBerry|IEMobile|MQQBrowser|JUC|Fennec|wOSBrowser|BrowserNG|WebOS|Symbian|Windows Phone)/i))) {
2      window.location.href = "";      //手机
3  } else {
4      window.location.href = "";      //电脑
5  }
```

history对象

window对象给我们提供了一个 history对象，与浏览器历史记录进行交互。该对象包含用户（在浏览器窗口中）访问过的URL。

history对象方法	作用
history.back()	实现后退1个页面
history.forward()	实现前进1个页面
history.go(n)	参数n为数值,可以实现前进或者后退n个页面; n如果是正值代表前进,n如果是负值代表后退; 比如 history.go(1)代表前进一个页面; history.go(-1)代表后退一个页面;

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。

- 1. 获得元素距离带有定位父元素的位置
- 2. 获得元素自身的大小（宽度高度）
- 3. 注意:返回的数值都不带单位

offset系列属性	作用
element.offsetParent	返回作为该元素带有定位的父级元素 如果父辈元素都没有定位则返回body元素
element.offsetTop	返回元素相对带有定位父元素上方的偏移量 如果父辈都没有定位则返回相对body的上方偏移量
element.offsetLeft	返回元素相对带有定位父元素左方的偏移量 如果父辈都没有定位则返回相对body的左方偏移量
element.offsetWidth	返回自身包括padding、边框、内容区的总宽度，返回数值不带单位
element.offsetHeight	返回自身包括padding、边框、内容区的总高度，返回数值不带单位

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	返回上边框的大小
element.clientLeft	返回左边框的大小
element.clientWidth	返回自身包括padding、内容区宽度 ,不含边框，返回数值不带单位
element.clientHeight	返回自身包括padding、内容区高度 ,不含边框，返回数值不带单位

注意:clientTop可以获取上边框的大小,clientLeft可以获取左边框的大小,没有clientBottom跟clientRight

小结: client 宽度 和我们offsetWidth 最大的区别就是 不包含边框

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

注意: scrollTop跟clientHeight的区别在于,文字内容超出盒子高度的时候,scrollTop获取的是内容的高度,而clientHeight获取的还是盒子的高度

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：

1. 声明了 DTD，使用 `document.documentElement.scrollTop`

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <meta http-equiv="X-UA-Compatible" content="ie=edge">
8      <title>Document</title>
```

这个就是声明了DTD

2. 未声明 DTD，使用 `document.body.scrollTop`

3. 新方法 `window.pageYOffset`和 `window.pageXOffset`，**IE9 开始支持**

兼容写法如何写？

今日总结

xmind不做,需要完成js复习笔记

今日作业

除了完善js复习笔记,还要做一个代码题,弹幕效果,具体请查看作业文件夹

关于面试题

从明天开始,我们每天早上,下午各抽一个同学,回答面试,抽到的同学在QQ班级群发语言说答案

靠自觉不要打开word文档,对着念,需要同学们自己背出来(面试上有不懂的东西或者课堂上没讲的东西,可以自行百度) 以后找工作,面试是一个关