day17 运动下

运动的要素

- 当前值
- 目标值
- 迭代量

运动分类

- 匀速运动 （迭代量不变）
- 缓冲运动 （迭代量越来越小）
- 链式运动 （通过回调函数来执行下一个动画）

轮播图

控制对应的图片轮播，主要的方式有：

- 位置变化轮播
- 透明度轮播
- 图片切换轮播

位置变化轮播

- 要将对应的图片放在一行（横向）或者一列（纵向）
- 实际控制移动的是外面的大盒子（定位）的移动 （切换对应的left值 或者是top值）
- 大盒子的定位要基于展示的盒子

```
<!-- 用于展示的盒子 -->
<div class="show">
    <ul class="content">
        <li>
            <img
src="//img14.360buyimg.com/pop/s1180x940_jfs/t1/78233/4/23310/55539/63e0dc1dF618631fe/6fd733fb64b9743f.jpg.webp"
                alt="">
        </li>
        <li>
            <img
src="//imgcps.jd.com/ling4/10035789319416/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa1b/7cdfbcc3/cr/s/q.jpg"
                alt="">
        </li>
        <li>
            <img
src="//imgcps.jd.com/ling4/100015163119/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa17/0b842d03/cr/s/q.jpg"
                alt="">
        </li>
        <li>
```

```
            <img
src="//img20.360buyimg.com/pop/s1180x940_jfs/t1/192114/40/14091/51151/60f7cb1cE9e0614db/511192a85bddd648.jpg.webp"
                 alt="">
        </li>
        <li>
            <img
src="//imgcps.jd.com/ling4/100034738701/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02f9d0/049bd553/cr/s/q.jpg"
                 alt="">
        </li>
    </ul>
</div>
</body>
<script src="./animated.js"></script>
<script>
    //获取对应的ul这个元素
    var ul = document.querySelector('.content')
    //获取对应的显示的盒子的大小
    var show = document.querySelector('.show')
    var width = show.clientWidth
    var height = show.clientHeight
    //设置对应的定时器控制ul的移动
    //用一个下标来表示对应切换的图片是哪一张
    var index = 0
    setInterval(()=>{
        // ul.style.left = -1 * width * index +'px'
        bufferAnimation(ul,{
            left: -1 * width * index
        })
        index++
        //如果到达最后一张变成第一张
        if(index>4){
            index = 0
        }
    },2000)
</script>
</html>
```

透明度轮播

- 控制对应的li的透明度变化
- 第一个切换到第二个的时候 第一个需要隐藏 第二个需要显示

```
<!-- 用于展示的盒子 -->
<div class="show">
    <ul class="content">
        <li>
            <img
src="//img14.360buyimg.com/pop/s1180x940_jfs/t1/78233/4/23310/55539/63e0dc1dF618631fe/6fd733fb64b9743f.jpg.webp"
                 alt="">
        </li>
```

```
        <li>
            <img
src="//imgcps.jd.com/ling4/10035789319416/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa1b/7cdfbcc3/cr/s/q.jpg"
                alt="">
        </li>
        <li>
            <img
src="//imgcps.jd.com/ling4/100015163119/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa17/0b842d03/cr/s/q.jpg"
                alt="">
        </li>
        <li>
            <img
src="//img20.360buyimg.com/pop/s1180x940_jfs/t1/192114/40/14091/51151/60f7cb1cE9e0614db/511192a85bddd648.jpg.webp"
                alt="">
        </li>
        <li>
            <img
src="//imgcps.jd.com/ling4/100034738701/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02f9d0/049bd553/cr/s/q.jpg"
                alt="">
        </li>
    </ul>
</div>
</body>
<script src="./animated.js"></script>
<script>
    //获取对应的ul下面的所有li
    var lis = document.querySelector('.content').children
    //定时器控制对应的li变化
    //记录对应的下标
    var index = 0
    setInterval(()=>{
        //前一张的index
        prevIndex = index
        //当前的index
        index ++
        if(index>=lis.length){
            index = 0
        }
        //先隐藏前一张 再显示第二张
        bufferAnimation(lis[prevIndex],{
            opacity:0
        },()=>{

        })
        bufferAnimation(lis[index],{
            opacity:1
        })

    },2000)
</script>
```

```
</html>
```

切换图片轮播

控制对应的图片的切换

```
<!-- 用于展示的盒子 -->
<div class="show">
    <img

src="//img14.360buyimg.com/pop/s1180x940_jfs/t1/78233/4/23310/55539/63e0dc1dF618631fe/6fd733fb64b9743f.jpg.webp"
         alt="">
</div>
</body>
<script src="./animated.js"></script>
<script>
    var imgs = [

 '//img14.360buyimg.com/pop/s1180x940_jfs/t1/78233/4/23310/55539/63e0dc1dF618631fe/6fd733fb64b9743f.jpg.webp',

 '//imgcps.jd.com/ling4/10035789319416/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa1b/7cdfbcc3/cr/s/q.jpg',

 '//imgcps.jd.com/ling4/100015163119/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa17/0b842d03/cr/s/q.jpg',

 '//img20.360buyimg.com/pop/s1180x940_jfs/t1/192114/40/14091/51151/60f7cb1cE9e0614db/511192a85bddd648.jpg.webp',

 '//imgcps.jd.com/ling4/100034738701/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02f9d0/049bd553/cr/s/q.jpg'
    ]
    var img = document.querySelector('.show>img')
    //定时器控制对应的img的src路径变化
    var index = 0 //记录是哪张图片
    setInterval(()=>{
        if(index>=imgs.length){
            index = 0
        }
        img.src = imgs[index]
        index++
    },1000)
</script>
```

无缝轮播

- 保证第一张和最后一张之间没有缝隙
- 给最后一张后面添加一个第一张 （那么我们就有最后一张向第一张切换的过程）
- 当最后一张切换到第一张完成以后 需要将对应的定位变成第一张的位置

```
<script>
  //获取对应的ul这个元素
  var ul = document.querySelector('.content')
  //获取对应的显示的盒子的大小
```

```
  var show = document.querySelector('.show')
  var width = show.clientWidth
  //记录初始的个数
  var size = ul.children.length
  //在最后一张后面添加一个第一张
  ul.appendChild(ul.firstElementChild.cloneNode(true))
  //声明index记录对应的图片位置
  var index = 0
  setInterval(() => {
    index++
    //切换位置
    bufferAnimation(ul, {
      left: -1 * index * width
    }, () => {
      //到达最后一张变为0
      if (index >= size) {
        index = 0
        //对应的位置要变成第一张的位置
        ul.style.left = '0px'
      }
    })
  },2000)
</script>
```

带焦点的轮播图

```
<!-- 用于展示的盒子 -->
  <div class="show">
    <ul class="content">
      <li>
        <img

 src="//img14.360buyimg.com/pop/s1180x940_jfs/t1/78233/4/23310/55539/63e0dc1dF618631fe/6fd733fb64b9743f.jpg.webp"
          alt="">
      </li>
      <li>
        <img

 src="//imgcps.jd.com/ling4/10035789319416/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa1b/7cdfbcc3/cr/s/q.jpg"
          alt="">
      </li>
      <li>
        <img

 src="//imgcps.jd.com/ling4/100015163119/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p-5bd8253082acdd181d02fa17/0b842d03/cr/s/q.jpg"
          alt="">
      </li>
      <li>
        <img

 src="//img20.360buyimg.com/pop/s1180x940_jfs/t1/192114/40/14091/51151/60f7cb1cE9e0614db/511192a85bddd648.jpg.webp"
          alt="">
      </li>
```

```
      <li>
        <img

 src="//imgcps.jd.com/ling4/100034738701/5Lqs6YCJ5aW96LSn/5L2g5YC85b6X5oul5pyJ/p
-5bd8253082acdd181d02f9d0/049bd553/cr/s/q.jpg"
          alt="">
      </li>
    </ul>
    <!-- 焦点展示的ul -->
    <ul class="list">
      <li class="selected">1</li>
      <li>2</li>
      <li>3</li>
      <li>4</li>
      <li>5</li>
    </ul>
  </div>
</body>
<script src="./animated.js"></script>
<script>
  //获取对应的ul这个元素
  var ul = document.querySelector('.content')
  //获取对应的显示的盒子的大小
  var show = document.querySelector('.show')
  var width = show.clientWidth
  //记录初始的个数
  var size = ul.children.length
  //在最后一张后面添加一个第一张
  ul.appendChild(ul.firstElementChild.cloneNode(true))
  //声明index记录对应的图片位置
  var index = 0
  // var selectCir = 0 //max=4 0-4
  setInterval(() => {
    index++ //max最大值 为5 0-5
    // selectCir = index % size
    move()
  }, 2000)
  //获取所有的焦点包含ul元素
  var list = document.querySelector('.list')
  //封装一个函数 专门用于给对应的焦点添加选中效果
  function selected() {
    //排他思想
    //list下的所有li设置class为空
    Array.from(list.children).forEach((li) => {
      li.className = ''
    })
    //再将选中的li 设置为对应的selected
    // list.children[selectCir].className = 'selected'
    list.children[ index % size].className = 'selected'
  }
  //给焦点添加点击事件
  //利用事件委托
  list.onclick = function (e) {
    e = e || window.event
    if (e.target.tagName == 'LI') {
      //得到当前点击的li的下标
      var i = Array.from(list.children).findIndex((li) => {
        return li == e.target
```

```
      })
      //给全局的index赋值
      index = i
      //给选中的焦点赋值
      // selectCir = i
      //调用移动的方法
      move()
    }
  }

  function move() {
    //焦点切换
    selected()
    //切换位置
    bufferAnimation(ul, {
      left: -1 * index * width
    }, () => {
      //到达最后一张变为0
      if (index >= size) {
        index = 0
        //对应的位置要变成第一张的位置
        ul.style.left = '0px'
      }
    })
  }
</script>
```

进阶轮播封装

```
//声明对应的变量
var warp, size, ul, width, height, list, arrow
//声明index记录对应的图片位置
var index = 0
//声明定时器
var timer, timerout
//声明是否纵向
var isY
//封装对应的方法
//封装移动的方法
//isY表示对应的方向 true表示纵向 false表示横向
//isDesc 是否逆时针  right false  left true index值是增加还是减少
function move(isDesc) {
  let key = isY ? 'top' : 'left'
  let distance = isY ? height : width
  if (isDesc && index <= 0) {
    index = size
    //位置发生改变
    ul.style[key] = `${index * -1 * distance}px`
  }
  //判断对应的最大值
  if (!isDesc && index >= size) {
    index = 0
    //位置发生改变
    ul.style[key] = `${index * -1 * distance}px`
  }
  //是否逆时针 逆时针就减少 顺时针就增加
  isDesc ? index-- : index++
```

```
  //开始移动
  //焦点切换
  selected()
  //切换位置
  if (isY) {
    bufferAnimation(ul, {
      top: -1 * index * height
    })
  } else {
    bufferAnimation(ul, {
      left: -1 * index * width
    })
  }
}
//封装一个函数 专门用于给对应的焦点添加选中效果
function selected() {
  //排他思想
  //list下的所有li设置class为空
  Array.from(list.children).forEach((li) => {
    li.className = ''
  })
  //再将选中的li 设置为对应的selected
  list.children[index % size].className = 'selected'
}
//封装初始化的方法
function init(element, disction) {
  //赋值传进来的元素
  warp = element
  //赋值方向
  isY = disction
  //获取对应的元素
  ul = warp.querySelector('.content')
  width = warp.clientWidth
  height = warp.clientHeight
  //获取所有的焦点包含的ul元素
  list = warp.querySelector('.list')
  //获取箭头包含的div
  arrow = warp.querySelector('.arrow')
  //记录初始的个数
  size = ul.children.length
  //声明index记录对应的图片位置
  //根据图片的个数去生成对应的焦点
  Array.from(ul.children).forEach((v, i) => {
    let li = document.createElement('li')
    li.innerText = i + 1
    //给第一个添加对应的selected
    if (i == 0) {
      li.className = 'selected'
    }
    //再将声明的li添加给ul
    list.appendChild(li)
  })
  //在最后一张后面添加一个第一张
  ul.appendChild(ul.firstElementChild.cloneNode(true))
  //给对应的生成的li及对应的箭头添加事件
  //给焦点添加点击事件
  //利用事件委托
  list.onclick = function (e) {
```

```
    e = e || window.event
    if (e.target.tagName == 'LI') {
      //得到当前点击的li的下标
      var i = Array.from(list.children).findIndex((li) => {
        return li == e.target
      })
      //给全局的index赋值
      index = i - 1
      //调用移动的方法
      move()
      delay()
    }
  }
  //利用事件委托给箭头添加事件
  arrow.onclick = (e) => {
    e = e || window.event
    //判断是否点击的是a
    if (e.target.tagName = 'A') {
      e.preventDefault();
      //下标只有俩个 要不为0 要不为0
      move(!Array.from(arrow.children).findIndex((a) => {
        return a == e.target
      }))
      delay()
    }
  }
  //调用自动移动的方法
  autoMove()
}
//等待1s 再轮播
function delay(time = 1000) {
  clearTimeout(timerout)
  clearInterval(timer)
  //等待一秒
  timerout = setTimeout(() => {
    autoMove()
  }, time)
}
//封装一个自动轮播的方法
function autoMove() {
  timer = setInterval(() => {
    move()
  }, 2000);
}
```

第三方动画库 move.js

[github地址]

```
//move.js主要利用的css3的2d转换
move('.box') //传入选择器
    .to(500, 200) //位置到达500 ,200
    .rotate(180) //旋转180deg
    .scale(.5) //缩放
    .set('background-color', '#888') //设置颜色
    .set('border-color', 'black') //设置边框颜色
    .duration('2s') //时间
```

```
    .skew(50, -10)//偏移
    .then() //成功以后
    .set('opacity', 0) //透明度设置
    .duration('0.3s') //时间
    .scale(0.1) //缩放
    .pop() //删除第一个
    .end(); //结束
```

轮播图的第三方插件 swiper

[swiper中文网]

简单案例

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8" />
  <title>Swiper demo</title>
  <meta name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1, maximum-scale=1" />
  <!-- Link Swiper's CSS -->
  <link rel="stylesheet" href="./swiper-bundle.min.css" />

  <!-- Demo styles -->
  <style>
    html,
    body {
      position: relative;
      height: 100%;
    }

    body {
      background: #eee;
      font-family: Helvetica Neue, Helvetica, Arial, sans-serif;
      font-size: 14px;
      color: #000;
      margin: 0;
      padding: 0;
    }

    .swiper {
      width: 100%;
      height: 100%;
    }

    .swiper-slide {
      text-align: center;
      font-size: 18px;
      background: #fff;

      /* Center slide text vertically */
      display: -webkit-box;
      display: -ms-flexbox;
      display: -webkit-flex;
      display: flex;
```

```
      -webkit-box-pack: center;
      -ms-flex-pack: center;
      -webkit-justify-content: center;
      justify-content: center;
      -webkit-box-align: center;
      -ms-flex-align: center;
      -webkit-align-items: center;
      align-items: center;
    }

    .swiper-slide img {
      display: block;
      width: 100%;
      height: 100%;
      object-fit: cover;
    }
  </style>
</head>

<body>
  <!-- Swiper -->
  <div class="swiper">
    <div class="swiper-wrapper">
      <div class="swiper-slide">Slide 1</div>
      <div class="swiper-slide">Slide 2</div>
      <div class="swiper-slide">Slide 3</div>
    </div>
    <div class="swiper-pagination a"></div>
    <div class="swiper-button-prev"></div>
    <!--左箭头。如果放置在swiper外面，需要自定义样式。-->
    <div class="swiper-button-next"></div>
    <!--右箭头。如果放置在swiper外面，需要自定义样式。-->
  </div>

  <!-- Swiper JS -->
  <script src="./swiper-bundle.js"></script>

  <!-- Initialize Swiper -->
  <script>
    //构造函数的机制
    var swiper = new Swiper(".swiper", {
      //分页点
      pagination: {
        el: ".a", //生成的分页点显示的位置
      },
      loop: true,
      autoplay: true,
      navigation: {
        nextEl: '.swiper-button-next',
        prevEl: '.swiper-button-prev',
      },
    });
  </script>
</body>

</html>
```