1. 快速开始
2. 下载并引用mescroll.min.css和mescroll.min.js
3. 在html页面中，添加以下布局

<div id=”mescroll” class=”mescroll”>

<div>

//内容

</div>

</div>

说明：

1. 父div（蓝色字体）中的id可以改名字，但是，class不能删除，也不能改名字，必须是mescroll。
2. 子div（绿色字体）不能删除，否则上拉加载的布局会错位。该div也可以换做ul或者是其他的容器标签。
3. 创建mescroll对象

var mescroll = new MeScroll("mescroll", {

down: {

callback: downCallback

},

up: {

callback: upCallback,

page: {

num: 0,

size: 10

},

htmlNodata: '<p class="upwarp-nodata">-- END --</p>',

noMoreSize: 5,

toTop: {

src: "../img/mescroll-totop.png",

offset: 1000

},

empty: {

warpId: "xxid",

icon: "../img/mescroll-empty.png",

tip: "暂无相关数据~"

},

lazyLoad: {

use: true

attr: 'imgurl'

}

}

});

说明：

1. 第一个参数"mescroll"（紫色字体）对应上面布局结构div的id（蓝色div中的id值）。
2. 参数down，是下拉刷新配置。
   1. downCallback是下拉刷新回调函数。
   2. 下拉刷新是重置列表数据,那么down完全可以不用配置。down.callback默认调用mescroll.resetUpScroll(),而resetUpScroll会将page.num=1,再触发up.callback。
3. 参数up，是上拉刷新。
   1. upCallback是上拉加载的回调函数。
   2. page参数，格式为{num:0,size:10}。num 为当前页 默认0,回调之前会加1; 即callback(page)会从1开始。每页数据条数,默认10。
4. htmlNodata，没有数据的时，显示的样式。
5. noMoreSize：作用是：如果列表已无数据,可设置列表的总数量要大于” noMoreSize”（例如5条数据）才显示无更多数据。
6. toTop：回到顶部按钮。
   1. src：图片路径,默认null,支持网络图
   2. offset：列表滚动什么位置（例如：1000px）才显示回到顶部按钮
7. empty：列表第一页无任何数据时,显示的空提示布局; 需配置warpId才显示。
   1. warpId：父布局的id（蓝色div中的id）
   2. icon：图标,默认null,支持网络图
   3. tip：提示语
8. lazyLoad：图片懒加载。
   1. use：是否开始懒加载，默认为false。
   2. attr：标签中网络图的属性名。例如：<img imgurl='网络图 src='占位图''/>
9. 处理回调函数。

//下拉刷新的回调

function downCallback() {

//ajax 与后台交互，获取数据

$.ajax({

url: 'xxxxxx',

success: function(data) {

//隐藏下拉刷新的状态;

mescroll.endSuccess(); //无参. 注意结束下拉刷新是无参的

//设置数据

//setXxxx(data);//自行实现 TODO

},

error: function(data) {

mescroll.endErr();//隐藏下拉刷新的状态

}

});

}

//上拉加载的回调

function upCallback(page) {

var pageNum = page.num; // 页码, 默认从1开始

var pageSize = page.size; // 页长, 默认每页10条

$.ajax({

url: 'xxxxxx?num=' + pageNum + "&size=" + pageSize,

success: function(data) {

var curPageData = data.xxx; // 接口返回的当前页数据列表

var totalPage = data.xxx; // 接口返回的总页数 (比如列表有26个数据,每页10条,共3页; 则totalPage值为3)

var totalSize = data.xxx; // 接口返回的总数据量(比如列表有26个数据,每页10条,共3页; 则totalSize值为26)

var hasNext = data.xxx; // 接口返回的是否有下一页 (true/false)

//设置列表数据

//setListData(curPageData);//自行实现 TODO

},

error: function(e) {

mescroll.endErr();//隐藏下拉刷新和上拉加载的状态

}

});

}

1. 参数说明
2. down 参数配置说明

|  |  |  |
| --- | --- | --- |
| down参数说明 | | |
| 参数名 | 默认值 | 说明 |
| use | true | 是否启用下拉刷新。如果配置false,则不会初始化下拉刷新的布局 |
| auto | true | 是否在初始化完毕之后自动执行一次下拉刷新的回调 callback |
| autoShowLoading | false | 如果设置[auto](http://www.mescroll.com/api.html?v=190426" \l "tagAuto)=true ( 在初始化完毕之后自动执行下拉刷新的回调 ) ,那么是否显示下拉刷新的进度需配置down的callback才生效 |
| isLock | false | 是否锁定下拉刷新。如果配置true,则会锁定不可下拉,可通过调用mescroll.lockDownScroll(false)解锁 |
| isBoth | false | 下拉刷新时,如果滑动到列表底部是否可以同时触发上拉加载 |
| offset | 80 | 在列表顶部,下拉大于80px,松手即可触发下拉刷新的回调 |
| inOffsetRate | 1 | 在列表顶部,下拉的距离小于offset时,改变下拉区域高度比例;值小于1且越接近0,高度变化越小,表现为越往下越难拉 |
| outOffsetRate | 0.2 | 在列表顶部,下拉的距离大于offset时,改变下拉区域高度比例;值越接近0,高度变化越小,表现为越往下越难拉 |
| bottomOffset | 20 | 当手指touchmove位置在距离body底部20px范围内的时候结束上拉刷新,避免Webview嵌套导致touchend事件不执行。这是1.2.1版本新增的配置,请检查最新版~ |
| minAngle | 45 | 触发下拉最少要偏移的角度(滑动的轨迹与水平线的锐角值),取值区间 [0,90];默认45度,即向下滑动的角度大于45度(方位角为45°~145°及225°~315°)则触发下拉;而小于45度,将不触发下拉,避免与左右滑动的轮播等组件冲突;  注意:没有必要配置超出[0,90]区间的值,否则角度限制无效; 因为假设配置60, 生效的方位角就已经是60°到120° 和 240°到300°的范围了 |
| hardwareClass | "mescroll-hardware" | 硬件加速样式,解决iOS下拉因隐藏进度条而闪屏的问题 |
| warpClass | "mescroll-ownwarp" | 下拉刷新的布局容器样式 |
| mustToTop | false | 是否滚动条必须在顶部,才可以下拉刷新.默认false。当您发现下拉刷新会闪白屏时,设置true即可修复 |
| warpId | null | 可配置下拉刷新的布局添加到指定id的div |
| resetClass | mescroll-downwarp-reset | 下拉刷新高度重置的动画 |
| textInOffset | 下拉刷新 | 下拉的距离在offset范围内的提示文本 |
| textOutOffset | 释放更新 | 下拉的距离大于offset范围的提示文本 |
| textLoading | 加载中 ... | 加载中的提示文本 |
| htmlContent | '<p class="downwarp-progress"></p>  <p class="downwarp-tip"></p>' | 下拉刷新的布局内容 |
| inited | function ( mescroll, downwarp ) { ... } | 下拉刷新初始化完毕的回调 (mescroll实例对象,下拉刷新容器dom对象) |
| inOffset | function ( mescroll ) { ... } | 下拉的距离进入[offset](http://www.mescroll.com/api.html?v=190426" \l "tagOffset)范围内那一刻的回调 (mescroll实例对象) |
| outOffset | function ( mescroll ) { ... } | 下拉的距离大于[offset](http://www.mescroll.com/api.html?v=190426" \l "tagOffset)那一刻的回调 (mescroll实例对象) |
| onMoving | function ( mescroll, rate, downHight ) { ... } | 下拉过程中的回调,滑动过程一直在执行; rate : 拉区域当前高度与指定距离的比值 ( [inOffset](http://www.mescroll.com/api.html?v=190426" \l "tagInOffset): rate<1; outOffset: rate>=1 ) downHight : 当前下拉区域的高度 |
| beforeLoading | function ( mescroll , downwarp ) {      return false;  } | 准备触发下拉刷新的回调; 如果return true,将不触发[showLoading](http://www.mescroll.com/api.html?v=190426" \l "tagShowLoad)和[callback](http://www.mescroll.com/api.html?v=190426" \l "tagCallback) |
| showLoading | function ( mescroll ) { ... } | 显示下拉刷新进度的回调 |
| afterLoading | function(mescroll) { return 0 } | 结束加载中,准备隐藏下拉的回调  返回结束下拉的延时执行时间,默认0ms 常用于结束下拉之前再显示另外一小段动画,才去隐藏下拉刷新的场景 |
| callback | function ( mescroll ) {      mescroll.resetUpScroll();  } | 下拉刷新的回调; 默认重置上拉加载列表为第一页 |

1. up 参数配置说明

|  |  |  |
| --- | --- | --- |
| up参数说明 | | |
| 参数名 | 默认值 | 说明 |
| use | true | 是否启用上拉加载。如果配置false,则不会初始化上拉加载的布局 |
| auto | 1.3.1以前版本默认false 1.3.1版本默认true | 是否在初始化完毕之后自动执行一次上拉加载的回调 |
| isLock | false | 是否锁定上拉加载 如果配置true,则会锁定不可上拉,可通过调用mescroll.lockUpScroll(false)解锁 |
| isBoth | false | 上拉加载时,如果滑动到列表顶部是否可以同时触发下拉刷新 |
| isBounce | true | 是否允许ios的bounce回弹;默认true,允许回弹 |
| offset | 100 | 列表滚动到距离底部小于100px,即可触发上拉加载的回调 |
| noMoreSize | 5 | 如果列表已无数据,可设置列表的总数量要大于5条才显示无更多数据;避免列表数据过少(比如只有一条数据),显示无更多数据会不好看 |
| toTop | {    warpId : null ,    src : null ,    html: null ,    offset : 1000 ,    warpClass : "mescroll-totop" ,    showClass : "mescroll-fade-in" ,    hideClass : "mescroll-fade-out" ,    duration : 300 ,    supportTap : false    btnClick : null  } | 回到顶部按钮的配置:  warpId: 父布局的id; 默认添加在body中 (1.3.5版本支持传入dom元素)  src : 图片路径,必须配置src才会显示回到顶部按钮,不配置不显示  html: 标签内容,默认null; 如果同时设置了src,则优先取src (2017/12/10新增)  offset : 列表滚动1000px显示回到顶部按钮  warpClass : 按钮样式 showClass : 显示样式 hideClass : 隐藏样式 duration : 回到顶部的动画时长, 默认300ms supportTap: 如果您的运行环境支持tap,则可配置true,可减少点击延时,快速响应事件;默认false,通过onclick添加点击事件; (注:微信和PC无法响应tap事件) btnClick: 点击按钮的回调; 提示:如果在回调里return true,将不执行回到顶部的操作 |
| loadFull | {    use : false ,    delay : 500  } | use : 列表数据过少,不足以滑动触发上拉加载,是否自动加载下一页,直到满屏或无数据; 默认false,因为可调大[page.size](http://www.mescroll.com/api.html?v=190426" \l "tagPage)使数据满屏. delay : 延时执行的毫秒数; 延时是为了保证列表数据或占位的图片都已初始化完成,且下拉刷新上拉加载中区域动画已执行完毕; |
| empty | {    warpId : null ,    icon : null ,    tip : "暂无相关数据~" ,    btntext : "" ,    btnClick : null    supportTap : false  } | 列表第一页无任何数据时,显示的空布局 (需配置warpId才生效)  warpId : 父布局的id (1.3.5版本支持传入dom元素) icon : 空布局的图标路径  tip : 提示文本  btntext : 按钮文本  btnClick : 点击按钮的回调  supportTap: 如果您的运行环境支持tap,则可配置true,可减少点击延时,快速响应事件;默认false,通过onclick添加点击事件; (注:微信和PC无法响应tap事件) |
| clearId | null | 加载第一页时需清空数据的列表id 如果此项有值,将不使用[clearEmptyId](http://www.mescroll.com/api.html?v=190426" \l "tagClearEmptyId)的值 在vue中使用,不能配置此项 |
| clearEmptyId | null | 相当于同时设置了[clearId](http://www.mescroll.com/api.html?v=190426" \l "tagClearId)和[empty.warpId](http://www.mescroll.com/api.html?v=190426" \l "tagEmpty) 简化写法; 在vue中使用,不能配置此项 |
| hardwareClass | "mescroll-hardware" | 硬件加速样式,使动画更流畅 |
| warpId | null | 可配置上拉加载的布局添加到指定id的div |
| warpClass | "mescroll-upwarp" | 上拉加载的布局容器样式 |
| htmlLoading | '<p class="upwarp-progress mescroll-rotate"></p><p class="upwarp-tip">加载中..</p>' | 上拉加载中的布局 |
| htmlNodata | '<p class="upwarp-nodata"> -- END -- </p>' | 无数据的布局 |
| inited | function ( mescroll , upwarp ) { } | 初始化完毕的回调 回调(mescroll实例, upwarp上拉加载的布局Dom对象) |
| showLoading | function ( mescroll , upwarp ) { ... } | 显示上拉加载中的回调 回调(mescroll实例, upwarp上拉加载的布局Dom对象) |
| showNoMore | function ( mescroll , upwarp ) { ... } | 显示无更多数据的回调 回调(mescroll实例, upwarp上拉加载的布局Dom对象) |
| onScroll | null | 列表滑动监听, 默认null 例 onScroll : function(mescroll, y, isUp){ ... }; y为列表当前滚动条的位置;isUp=true向上滑,isUp=false向下滑) |
| callback | function ( page, mescroll ) { } | 上拉加载的回调; 回调 ( [page](http://www.mescroll.com/api.html?v=190426#tagPage)对象, mescroll实例 ) |
| page | {   num : 0 ,    size : 10 ,    time : null  } | num : 当前页码,默认0,回调之前加1,即[callback](http://www.mescroll.com/api.html?v=190426" \l "tagCallback)(page)从1开始; [如何修改从0开始 ?](http://www.mescroll.com/qa.html#q7) size : 每页数据的数量;  time : 加载第一页数据服务器返回的时间 (可空); 防止用户翻页时,后台新增了数据从而导致下一页数据重复; |
| scrollbar | {   use : ... ,    barClass : "mescroll-bar"  } | use : 是否开启自定义滚动条; PC端默认true开启自定义滚动条; 移动端默认false不使用   barClass : 自定义滚动条的样式; |
| lazyLoad | { use: false, attr: 'imgurl', showClass: 'mescroll-lazy-in', delay: 500, offset: 200 } | use: 是否开启懒加载,默认false attr: 标签中网络图片地址的属性名,默认"imgurl" showClass: 显示样式:渐变显示,参见mescroll.css delay: 列表滚动的过程中检查一次图片是否在可视区域的时间间隔,默认500 (单位ms) offset: 超出可视区域多少px的图片仍可触发懒加载 默认200 |

1. 常用方法

|  |  |
| --- | --- |
| 方法名 | 说明 |
| mescroll.endByPage(dataSize, totalPage, systime); | 隐藏下拉刷新和上拉加载的状态, 在联网获取数据成功后调用 dataSize : 当前页获取的数据总数(注意是当前页) totalPage : 列表的总页数 [systime : 加载第一页数据的服务器时间 (可空);](http://www.mescroll.com/qa.html#q9) |
| mescroll.endBySize(dataSize, totalSize, systime); | 隐藏下拉刷新和上拉加载的状态, 在联网获取数据成功后调用 dataSize : 当前页获取的数据总数(注意是当前页) totalSize : 列表的总数据量 systime : 加载第一页数据的服务器时间 (可空); |
| mescroll.endSuccess(dataSize, hasNext, systime); | 隐藏下拉刷新和上拉加载的状态, 在联网获取数据成功后调用 dataSize : 当前页获取的数据量(注意是当前页) hasNext : 是否有下一页数据true/false systime : 加载第一页数据的服务器时间 (可空); |
| mescroll.endErr(); | 隐藏下拉刷新和上拉加载的状态, 在联网获取数据失败后调用; mescroll内部会自动恢复原来的页码,时间等变量; |
| mescroll.resetUpScroll( isShowLoading ); | 重置列表为第一页 (常用于列表筛选条件变化或切换菜单时重新刷新列表数据) 内部实现: 把page.num=1,再主动触发up.callback isShowLoading 是否显示进度布局;  1.默认null,不传参,则显示上拉加载的进度布局  2.传参true, 则显示下拉刷新的进度布局 3.传参false,则不显示上拉和下拉的进度 (常用于静默更新列表数据) |
| mescroll.triggerDownScroll(); | 主动触发下拉刷新 |
| mescroll.triggerUpScroll(); | 主动触发上拉加载 |
| mescroll.setPageNum(num); | 设置当前page.num的值 |
| mescroll.setPageSize(size); | 设置当前page.size的值 |
| mescroll.scrollTo( y, t ); | 滚动列表到指定位置 y=0,则回到列表顶部; 如需滚动到列表底部,可设置y很大的值,比如y=99999 t时长,单位ms,默认300; 如果不需要动画缓冲效果,则传0 |
| mescroll.optDown; | 获取下拉刷新的配置 |
| mescroll.optUp; | 获取上拉加载的配置 |
| mescroll.lockDownScroll( isLock ); | 锁定下拉刷新 ( isLock=ture,null 锁定 ; isLock=false 解锁 ) |
| mescroll.lockUpScroll( isLock ); | 锁定上拉加载 ( isLock=ture,null 锁定 ; isLock=false 解锁 ) |
| mescroll.os | mescroll.os.ios 为true, 则是ios设备; mescroll.os.android 为true, 则是android设备; mescroll.os.pc 为true, 则是PC端; |
| mescroll.setBounce(boolean) | mescroll.setBounce(true) 允许bounce; mescroll.setBounce(false) 禁止bounce |
| mescroll.lazyLoad(delay) | 主动触发懒加载: 自动加载可视区域的图片. delay:延时加载图片的时间,默认500ms.目的是确保dom元素已渲染完成. |
| mescroll.showDownScroll(); | 显示下拉刷新的进度布局mescroll.triggerDownScroll()和 mescroll.resetUpScroll() 内部有调用 |
| mescroll.endDownScroll(); | 隐藏下拉刷新的进度布局mescroll.endSuccess() 和 mescroll.endErr() 内部有调用 |
| mescroll.endUpScroll(isShowNoMore) | 结束上拉加载的状态 isShowNoMore=true, 显示无更多数据; isShowNoMore=false, 隐藏上拉加载; isShowNoMore=null, 保持当前状态 |
| mescroll.showUpScroll(); | 显示上拉加载的进度布局mescroll.triggerDownScroll() 和 mescroll.resetUpScroll() 内部有调用 |
| mescroll.showNoMore(); | 显示上拉无更多数据的布局 mescroll.endUpScroll() 内部有调用 |
| mescroll.hideUpScroll(displayAble); | 隐藏上拉加载的布局 mescroll.endUpScroll() 内部有调用参数 displayAble: 是否通过display:none隐藏默认false通过visibility:hidden的方式隐藏 |
| mescroll.clearDataList(); | 清空上拉加载的数据列表mescroll.resetUpScroll() 和 mescroll.endSuccess() 内部有调用 |
| mescroll.showEmpty(); | 显示无任何数据的空布局mescroll.endSuccess() 内部有调用 |
| mescroll.removeEmpty(); | 移除无任何数据的空布局mescroll.endSuccess() 内部有调用 |
| mescroll.showTopBtn(time); | 显示回到顶部的按钮 time: 显示的动画时长,默认0.5秒 |
| mescroll.hideTopBtn(time); | 隐藏回到顶部的按钮  time: 隐藏的动画时长,默认0.5秒 |
| mescroll.setTopBtnFadeDuration(time); | 设置回到顶部按钮的显示和隐藏的动画时长  time: 显示隐藏动画时长,默认0.5秒 |
| mescroll.getScrollTop(); | 获取滚动条的位置y; 也可以在up配置onScroll监听滚动条的位置 |
| mescroll.getBodyHeight(); | 获取body的高度 |
| mescroll.getClientHeight(); | 获取滚动容器的高度 |
| mescroll.getScrollHeight(); | 获取滚动内容的高度 |
| mescroll.getToBottom(); | 获取当前滚动条到底部的距离 |
| mescroll.getStep(star, end, callback, t, rate); | star : 开始值;  end : 结束值;  callback(step,timer) : 回调 function(step,timer),  t : 计步时长; 传0则直接回调end值; 不传则默认300ms ;  rate : 周期; 不传则默认30ms计步一次 ;  此方法相当于默认在300ms内,每30ms返回star到end之间的阶梯值step; 可用于模拟帧动画  比如mescroll的回到顶部缓冲动画,轮播导航案例的顶部菜单滚动都是通过getStep实现 (注: 您可根据实际情况在 callback 通过 window.clearInterval(timer) 提前结束计步器) |
| mescroll.version; | mescroll的版本号 |
| mescroll.destroy(); | 销毁mescroll |