## 一、Better-scroll滑动组件

说明：这是一个移动端滑动的组件！

在Vue中使用：

案例1：实现左右滑动的轮播

Betters-croll这个库有时候会和fastClick这个库有冲突

|  |
| --- |
| <script>  import {addClass} from "@/common/js/dom";  import BScroll from 'better-scroll'   export default {  props: {  loop: {  type: *Boolean*,  default: true  },  autoPlay: {  type: *Boolean*,  default: true  },  interval: {  type: *Number*,  default: 2000  }  },  data() {  return {  dots: [],  currentPageIndex: 0  }  },  mounted() {  this.$nextTick(()=>{  this.\_setSliderWidth();  this.\_initDots();  this.\_initSlider();  if(this.autoPlay){  this.\_play()  }  })  // setTimeout(() => {  // this.\_setSliderWidth();  // this.\_initDots();  // this.\_initSlider();  // if(this.autoPlay){  // this.\_play()  // };  // }, 200)  //当移动端视口大小变化的时候，轮播图会出错，这里监听视口！  *window*.addEventListener('resize',()=>{  if(!this.slider){//没有初始化，计算框框的宽度  return;//直接返回  }  this.\_setSliderWidth(true);//重新计算框的大小！  this.slider.refresh()//重新初始化  })  },  methods: {  \_setSliderWidth(*isResize*) {  this.children = this.$refs.sliderGroup.children  let width = 0  let sliderWidth = this.$refs.slider.clientWidth  for (let i = 0; i < this.children.length; i++) {  let child = this.children[i]  addClass(child, 'slider-item')  child.style.width = sliderWidth + 'px'  width += sliderWidth  }  if (this.loop && !*isResize*) {  //第一次克隆数+2,后面视口改变的时候不加了  width += 2 \* sliderWidth  }  this.$refs.sliderGroup.style.width = width + 'px'  },  \_initSlider() {  this.slider = new BScroll(this.$refs.slider, {  scrollX: true,  scrollY: false,  momentum: false,  snap: {  loop: true,  threshold: 0.3,  speed: 400  },  // bounce: false,  // stopPropagation: true,  // click: this.click  })  this.slider.on('scrollEnd', () => {  //当前的索引  let pageIndex = this.slider.getCurrentPage().pageX;  // if (this.loop) {//如果循环索引-1，克隆多了2个，版本升级后不需要了  // pageIndex -= 1  // }  this.currentPageIndex = pageIndex;  if (this.autoPlay){  //清除定时器，防止自动轮播的过程中，用户触碰影响  clearTimeout(this.timeId);  this.\_play()  }  })  },  \_initDots() {  this.dots = new *Array*(this.children.length)  },  \_play(){  this.timeId = setTimeout(()=>{  this.slider.next();//自动轮播！  },this.interval)  }  },  } </script> |

1. 与fastclick冲突问题

Fastclick：是一款移动端的点击无延迟的插件

安装：cnpm i fastclick --save

用法：在main.js文件中引入,

|  |
| --- |
| import fastclick from 'fastclick'  fastclick.attach(*document*.body) |

当滚动的图片是可以点击的时候，better-scroll 默认要传参数 click:true;

但是这时候就会和fastClick 冲突了。

解决方案：

在需要点击的Img中加入一个class属性 [class=” needclick ”]

## 关于跨域的介绍

首先搞懂一点，服务器访问服务器是不存在跨域问题的，所以做代理就是我们用本地的服务器去访问要访问的服务器接口！

1. jsonp跨域：

安装：npm install jsonp --save

Jsonp是动态创建script标签解决跨域，利用浏览器对script没有限制的同源策略限制！

不需要用到后端代理。

1. Axios

安装：npm i axios --save

axios同样是一个ajax请求，它在处理跨域问题的时候，比如在vue项目中，我创建了一个node服务器，那么在node下，请求要访问的url，就是利用服务器访问服务器，记得在vue.config.js文件中配置一下。

|  |
| --- |
| module.exports = {  devServer: {  proxy: {  '/goods': {  target: 'http://localhost:3000',  //ws: true, //是否自动打开页面  changeOrigin: true  },  '/users': {  target: 'http://localhost:3000',  //ws: true, //是否自动打开页面  changeOrigin: true  }  }  } } |

1. 在原生没有框架的文件中，发送ajax跨域请求，我们可以自己创建一个Node服务器！

！要先安装request模块

|  |
| --- |
| 1.html页面的ajax请求先访问自己的服务器，然后在自己的服务接口中，request(url,fn)方法到后端的服务器  ------  html  $.get('http://localhost:3000/abc',{},()=>{})  ------  2.node  app.get('http://localhost:3000/abc',(req,res)=>{  request方法帮忙访问后端服务器  requset('后端url'，(err,response,body)=>{  if(!err){  res.send(body)//body就是拿到的数据  }    })  }) |

## 懒加载vue-lazyload

npm 搜 vue-lazyload

安装：npm i --save vue-lazyload

用法：<https://www.npmjs.com/package/vue-lazyload>

|  |
| --- |
| 在main.js文件中引入：  import VueLazyload from 'vue-lazyload'      // or with options  Vue.use(VueLazyload, {  loading: require('./assets/static/loading-svg/loading-bars.svg') ,  })  Vue.use(VueLazyload) //这里注意要放在下面！  // loading 这里是提供的一种懒加载的方式，有哪些方式在文件夹 `常用的插件`里面有  // loading 这里是加载中，未加载到的图片显示的初始图片  <ul>  <li v-for="img in list">  <img v-lazy="img.src" >  </li>  </ul>  v-lazy 是库的一个属性！后面是加载的图片url |