Atitit pagging翻页与按需加载图像 vue lazyload懒加载

目录

[1.1. 翻最好就是不翻页直接加载一千数据咯 1](#_Toc6169)

[1.2. 使用VueLazyload 1](#_Toc3410)

[1.3. 五.更加方便快捷的实现方式 2](#_Toc19390)

[1.4. 结束语：到这里，我们的小demo也就基本完成了，虽然还有其他方法，在这里我也就不一一阐述了，有兴趣的可以自行网上学习了解。需要本项目源代码的同学可以移步GitHub： 4](#_Toc19044)

## 翻最好就是不翻页直接加载一千数据咯

按需加载数据即可。。

<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>  
<script src="https://cdn.jsdelivr.net/npm/vue-resource@1.5.3"></script>  
<script src="https://unpkg.com/vue-lazyload/vue-lazyload.js"></script>

## 使用VueLazyload

Vue.use(VueLazyload)  
Vue.use(VueResource); //这个一定要加上，指的是调用vue-resource.js  
var example1 = new Vue({  
 el: '#' + targetDivName,  
 data: {items: [] },

<div class="videoBox-cover" v-lazy:background-image="vo.vod\_pic"  
 >

## **五.更加方便快捷的实现方式**

**1.了解一个API**

这种实现方式我们只需要了解一个API就行了：

getBoundingClientRect()//获取元素的大小及位置

*MDN上的解释：*

**2.实现方式**

通过上面的实验我们都知道，懒加载的一个重点就是要知道什么时候图片是进入了可视区内，那么就上面这张图而言，我们有什么方法判断图片进入了可视区呢。

其实很简单：

我们先获取图片到可视区顶部的距离,并获取到可视区的高度：

var bound = el.getBoundingClientRect();

var clientHeight = window.innerHeight;//这个和前面获取可视区高度的效果一样，只是兼容性问题

然后我们继续思考，当我们滚动条向下滚动的时候，bound.top值会变得越来越小，也就是图片到可视区顶部的距离也越来越小，所以当bound.top == clientHeight时，说明土片马上就要进入可视区了，只要我们在滚动，图片就会进入可视区，那么就需要请求资源了。也就是说，在bound.top<=clientHeight时，图片是在可视区域内的。

经过上面的思考，我们大致明白了如何实现，那么就来编写我们的代码了吧：

只需要把我们的js代码换成如下即可：

var imgs = document.querySelectorAll('img');

//用来判断bound.top<=clientHeight的函数，返回一个bool值

function isIn(el) {

var bound = el.getBoundingClientRect();

var clientHeight = window.innerHeight;

return bound.top <= clientHeight;

}

//检查图片是否在可视区内，如果不在，则加载

function check() {

Array.from(imgs).forEach(function(el){

if(isIn(el)){

loadImg(el);

}

})

}

function loadImg(el) {

if(!el.src){

var source = el.dataset.src;

el.src = source;

}

}

window.onload = window.onscroll = function () { //onscroll()在滚动条滚动的时候触发

check();

}

## **结束语：到这里，我们的小demo也就基本完成了，虽然还有其他方法，在这里我也就不一一阐述了，有兴趣的可以自行网上学习了解。需要本项目源代码的同学可以移步GitHub：**

**总结：**懒加载其实就两个重点，一个是元素到各个边距的距离，二个就是判断元素是否在可视区域内。

Ref