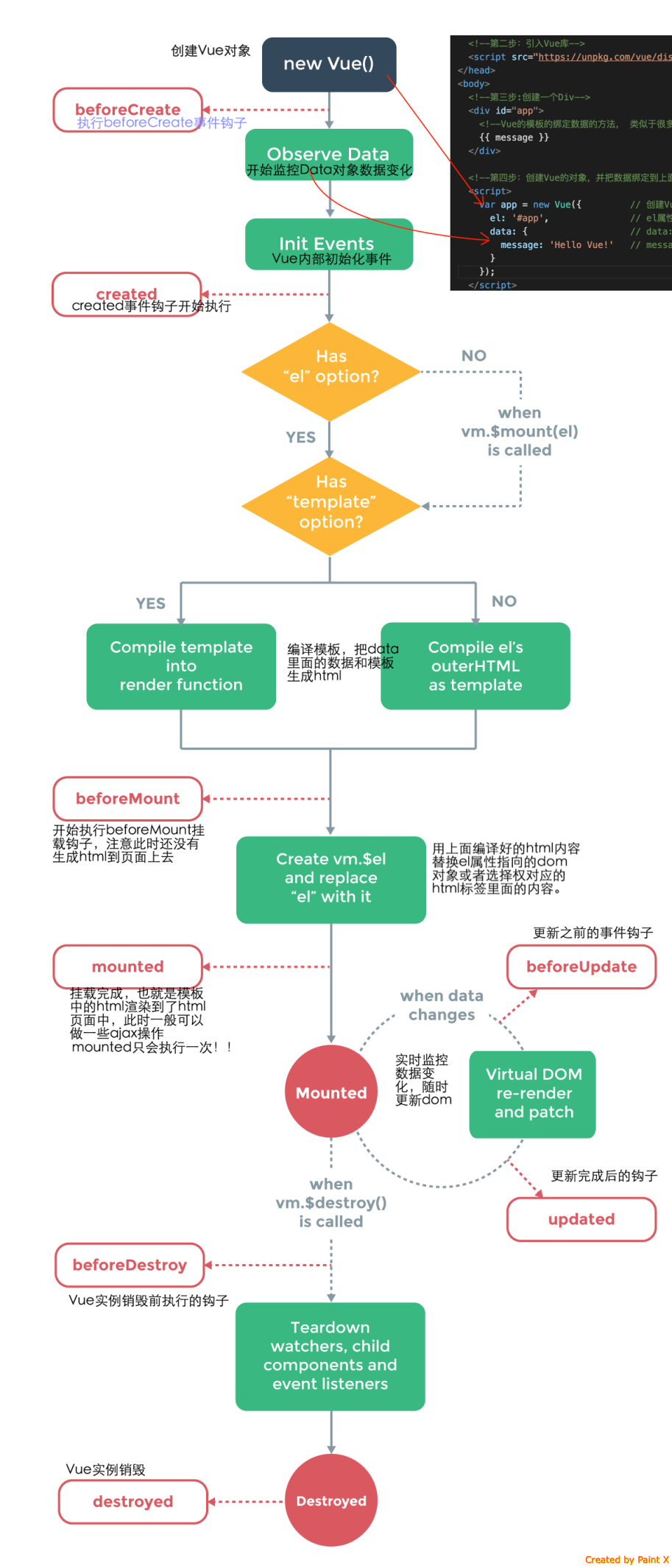
使用https://github.com/surmon-china/vue-awesome-swiper的一些困惑

轮播图需要lazyload，滚到下一张才显示真正的图片，所以需要对onSlideNextEnd回调方法进行处理。原出事想这样做的：但是发觉页面上swiper.slides.length=0

swiperOption: {  
 swiperSlides: 0,  
 autoplay: 50000,  
 notNextTick: **true**,  
 pagination: '.swiper-pagination',  
 onInit: **function** (swiper) {  
 console.log(swiper.slides.length)// 0  
 },  
 onSlideNextEnd: **function** (swiper) {  
 // console.log(swiper);  
 }  
},



参考地址：<http://www.cnblogs.com/fly_dragon/p/6220273.html>

1、data根据indexDate数据编译模板，生成html，但是此时indexDate数据为空数组，应该ajax并没有返回，所以生成出来html是这样的

<div data-v-645d8e3c="" class="container padding\_b\_120">  
 <div class="swiper-container swiper-container-horizontal" id="carrousel" data-v-645d8e3c="">  
 <div class="swiper-wrapper">  
 </div>  
 <div data-v-645d8e3c="" class="swiper-pagination swiper-pagination-bullets">  
 <span class="swiper-pagination-bullet swiper-pagination-bullet-active"></span>  
 </div>  
 </div>  
 <div data-v-645d8e3c="" class="item boutique">  
 <h1 data-v-645d8e3c="" class="title">精选频道</h1>  
 <div data-v-645d8e3c="" class="scroll">  
 <ul data-v-645d8e3c="">  
 </ul>  
 </div>  
 </div>  
</div>

所以为swiper.slides.length=0的原因

**return** {  
 swiperOption: {  
 swiperSlides: 0,  
 autoplay: 50000,  
 notNextTick: **true**,  
 pagination: '.swiper-pagination',  
 onInit: **function** (swiper) {console.log(swiper.slides.length)// 0  
 }  
 },  
 indexData: [],  
}

2、当挂载完毕，调用mounted（）中的异步请求数据回来了

mounted: **function** () {  
 **this**.$http.get('../../../static/api/home/homepageV3.json').then((response) => {  
 **this**.swiperSlides = response.data.banner\_list.length  
 console.log(**this**.swiperSlides)// 2  
 }, (response) => {  
 console.log(response)  
 })  
},

然后执行了updated来更新dom，这时候html生成的代码发生了改变

<div class="swiper-container swiper-container-horizontal" id="carrousel" data-v-645d8e3c="">  
 <div class="swiper-wrapper" style="transform: translate3d(0px, 0px, 0px);">  
 <div class="swiper-slide slide bg\_lazyload swiper-slide-active" lazy="loading" data-image="../static/api/Home/images/banner2.jpg" data-v-645d8e3c="" style="width: 1242px;">  
 <a data-v-645d8e3c="" href=""></a>  
 </div>  
 <div class="swiper-slide slide bg\_lazyload swiper-slide-next" lazy="loading" data-image="../static/api/Home/images/banner1.jpg" data-v-645d8e3c="" style="width: 1242px;">  
 <a data-v-645d8e3c="" href=""></a>  
 </div>  
 </div>  
 <div data-v-645d8e3c="" class="swiper-pagination swiper-pagination-bullets">  
 <span class="swiper-pagination-bullet swiper-pagination-bullet-active"></span><span class="swiper-pagination-bullet"></span>  
 </div>  
</div>

dom更新之后，那么如何使用之前的回调方法onSlideNextEnd进行处理呢？原生JS如下

onSlideNextEnd: **function** (swiper) {  
 **const** ele = swiper.slides[swiper.activeIndex];  
 **const** imgSrc = ele.getAttribute('data-image')  
 console.log(imgSrc)// ../static/api/Home/images/banner1.jpg  
 **if** (imgSrc !== **null**) {  
 // 直接使用new Image()eslint会报错  
 **const** img = **new** window.Image()  
 img.src = imgSrc;  
 img.onload = **function** () {  
 // 图片加载完毕  
 ele.removeAttribute('data-image')  
 ele.setAttribute('lazy', 'loaded')  
 ele.style.cssText = 'background-image:url(' + imgSrc + ')'  
 }  
 }  
}

由于有在onInit回调方法中也需要用到上面的代码，所以要把它们抽离出来，供onInit，onSlideNextEnd共同使用一个方法（因为从jquery转过来，这里公用方法貌似还有优化空间，也许应该是应用vue中数据驱动视图DOM思想来改一下）

methods: {  
 swiperSlideEnd: **function** () {  
 // 这个this.swiper为computed的swiper  
 **const** ele = **this**.swiper.slides[**this**.swiper.activeIndex];  
 **const** imgSrc = ele.getAttribute('data-image')  
 console.log(imgSrc)// ../static/api/Home/images/banner1.jpg  
 **if** (imgSrc !== **null**) {  
 // 直接使用new Image()eslint会报错  
 **const** img = **new** window.Image()  
 img.src = imgSrc;  
 img.onload = **function** () {  
 // 图片加载完毕  
 ele.removeAttribute('data-image')  
 ele.setAttribute('lazy', 'loaded')  
 ele.style.cssText = 'background-image:url(' + imgSrc + ')'  
 }  
 }  
 }  
}

onSlideNextEnd: **function** (swiper) {  
 // **this**.swiperSlideEnd()// this指向不是vue了，错误

self.swiperSlideEnd() //在 data()在定义了 **const** self = **this**  
}

不过这里用了ES6就可以使用箭头函数来把this指向vue，就不用定义const self = this那么old的方法啦。

onSlideNextEnd: () => {  
 **this**.swiperSlideEnd()// this 是vue  
}

init

mounted: **function** () {  
 **this**.$http.get('../../../static/api/home/homepageV3.json').then((response) => {  
 // indexData，然后html就可以遍历  
 console.log(response.data)  
 **this**.indexData = response.data  
 // 使用$nextTick来对更新后的mounted节点进行操作  
 **this**.$nextTick(**function** () {  
 **this**.swiperSlideEnd()  
 })  
 }, (response) => {  
 console.log(response)  
 })  
},

完整代码：

<template>  
 <div class="container padding\_b\_120">  
 <!--<swipe class="swipe" :auto="50000">  
 <swipe-item class="slide cover" v-for="banner in indexData.banner\_list"  
 v-bind:style="{ 'background-image': 'url(' + banner.image + ')' }">  
 <a href=""></a>  
 </swipe-item>  
 </swipe>-->  
 <swiper :options="swiperOption" id="carrousel" ref="mySwiperA">  
 <swiper-slide class="slide bg\_lazyload" lazy="loading" v-for="banner in indexData.banner\_list"  
 v-bind:data-image="banner.image"><a href=""></a></swiper-slide>  
 <div class="swiper-pagination" slot="pagination"></div>  
 </swiper>  
 <div class="item boutique">  
 <h1 class="title">精选频道</h1>  
 <div class="scroll">  
 <ul>  
 <li v-for="line in indexData.line\_list">  
 <a href="">  
 <div class="bg bg\_lazyload" v-lazy:background-image="line.image"></div>  
 </a>  
 </li>  
 </ul>  
 </div>  
 </div>  
 <div v-bind:class="{'item': list.data\_type==2,  
 'item surprise':list.data\_type==1&&list.display\_type==3,  
 'item surprise activity':list.data\_type==0&&list.display\_type==3,  
 'item surprise list':list.data\_type==1&&list.display\_type==2}"  
 v-for="list in indexData.data\_list">  
 <h1 class="title">{{list.title}}</h1>  
 <div class="scroll">  
 <ul>  
 <li v-for="item in list.data">  
 <a href="">  
 <div class="bg bg\_lazyload" v-lazy:background-image="item.image"></div>  
 <!--目的地-->  
 <div class="text flex flex\_v\_center" v-if="list.display\_type==3&&list.data\_type==2">  
 <div class="flex\_item">{{item.title}}</div>  
 </div>  
 <!--图文-->  
 <div class="bottom\_title"  
 v-if="list.display\_type==3&&list.data\_type==1||list.data\_type==1&&list.display\_type==2">  
 <h2>{{item.title}}</h2>  
 <div class="flex">  
 <div class="left flex\_item">已售{{item.sale\_amount}}份</div>  
 <div class="right flex\_item">{{item.price}}</div>  
 </div>  
 </div>  
 </a>  
 </li>  
 </ul>  
 </div>  
 </div>  
</template>  
<script>  
**import** { swiper, swiperSlide } **from** 'vue-awesome-swiper'  
**export default**{  
 data () {  
 **const** self = **this  
 return** {  
 swiperOption: {  
 swiperSlides: 0,  
 autoplay: 5000,  
 autoplayDisableOnInteraction: **false**,  
 notNextTick: **true**,  
 pagination: '.swiper-pagination',  
 onSlideNextEnd: () => {  
 **this**.swiperSlideEnd()  
 }  
 },  
 indexData: [],  
 swiperSlides: 0  
 }  
 },  
 computed: {  
 swiper () {  
 **return this**.$refs.mySwiperA.swiper  
 }  
 },  
 mounted: **function** () {  
 **this**.$http.get('../../../static/api/home/homepageV3.json').then((response) => {  
 // indexData，然后html就可以遍历  
 console.log(response.data)  
 **this**.indexData = response.data  
 // 使用$nextTick来对更新后的mounted节点进行操作  
 **this**.$nextTick(**function** () {  
 **this**.swiperSlideEnd()  
 })  
 }, (response) => {  
 console.log(response)  
 })  
 },  
 components: {  
 swiper,  
 swiperSlide  
 },  
 methods: {  
 swiperSlideEnd: **function** () {  
 // 这个this.swiper为computed的swiper  
 **const** ele = **this**.swiper.slides[**this**.swiper.activeIndex];  
 **const** imgSrc = ele.getAttribute('data-image')  
 console.log(imgSrc)// ../static/api/Home/images/banner1.jpg  
 **if** (imgSrc !== **null**) {  
 // 直接使用new Image()eslint会报错  
 **const** img = **new** window.Image()  
 img.src = imgSrc;  
 img.onload = **function** () {  
 // 图片加载完毕  
 ele.removeAttribute('data-image')  
 ele.setAttribute('lazy', 'loaded')  
 ele.style.cssText = 'background-image:url(' + imgSrc + ')'  
 }  
 }  
 }  
 }  
}  
</script>  
<style lang="scss" scoped>  
 // @import '../../assets/scss/home/home.scss'  
</style>"

要实现lazyload的话，总结一下执行顺序

1. data数据indexDate渲染到页面上，此时为空数据
2. 执行mounted中的ajax方法
3. Ajax数据返回赋值给indexDate，此时vue内部会进行dom更新，但是什么时候执行完，不知道
4. 所以使用$nextTick方法来获取dom更新后的节点
5. Swiper中init初始化调用了methods中swiperSlideEnd来实现
6. swiperSlideEnd又调用了computed中的swiper方法来获取swiper组件
7. 开始配置onSlideNextEnd回调方法使用了swiperSlideEnd来实现lazyload
8. end