链滴

地址列表渲染实现，地址选配功能

作者：ChenforCode

原文链接：https://ld246.com/article/1572273863156

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

1.进行初始化，拿到mock中的地址数据

```
created() {
        this.init();
    },
    methods: {
        init() {
            this.axios.get("/mock/address.json").then((response) => {
                let res = response.data;
                this.addressList = res.data;

            });
        }
    }
```

2.利用v-for, v-bind指令对地址列表进行渲染

```
<li class="check" v-for="item in addressList" v-bind:key="item.addressId">
                        <dl>
                            <dt>{{item.Username}}</dt>
                            <dd class="address">{{item.streetName}}</dd>
                            <dd class="tel">{{item.tel}}</dd>
                        </dl>
```

3.在没有打开查看更多之前，只显示三个，如果打开，就显示多个

这个地方又用到了计算属性，只不过这个属性是一个地址的list，动态的改变然后去渲染，

```
computed: {
        addressFilter() {
            return this.addressList.splice(0, this.limit);
        }
    },
```

limit目前先写死就是3，那么目前就只能显示三个，然后只需要在点击查看更多的时候改变limit的数就可以了

limit改变的click事件：

```
expand() {
            window.console.log(this.limit);
            window.console.log(this.addressList.length);
            window.console.log(this.addressList);
            //如果是3，代表要展开
            if (this.limit == 3) {
                this.limit = this.addressList.length;
            } else {
                //否则就是折叠起来
                this.limit = 3;
            }
        }
```

然后这里就不再详细解释了，可以直接看注释。但是这里要记录一个点。这个地方出现了一个bug，是limit一直是3，所以addressList.length也一直是3，为什么呢，就想到了是不是因为在computed边被切割了。这个时候就想到了splice，百度一下，发现了他和slice的区别：

@1.slice(start,end)：方法可从已有数组中返回选定的元素，返回一个新数组，包含从start到end（包含该元素）的数组元素

@2.splice()：该方法向或者从数组中添加或者删除项目，返回被删除的项目。（该方法会改变原数组）

所以computed里边不应该用splice，而是用slice。bug解决！

4.改变箭头，有一个open类，代表现在是打开的，也就是数量是大于三的，因此只要在limit大于3的候绑定上一个open类就可以类

```
<a class="addr-more-btn up-down-btn" href="javascript:;" @click="expand" v-bind:class="{'
pen': limit>3}">
```

5.点亮默认的选配地址

首先在初始化的函数里找到默认的地址的索引

```
init() {
            this.axios.get("/mock/address.json").then((response) => {
                let res = response.data;
                this.addressList = res.data;
                //找到谁是默认的地址
                this.addressList.forEach((item, index) => {
                    if (item.checked) {
                        this.checkedIndex = index;
                    }
                });
            });
        },
```

6.然后再动态绑定check类

```
<li v-bind:class="{'check': checkedIndex==index}" v-for="(item, index) in addressFilter" v-bind
key="item.addressId">
```

7.绑定click事件，改变checkedIndex，从而改变渲染效果

```
<li v-bind:class="{'check': checkedIndex==index}" v-for="(item, index) in addressFilter" v-bind
key="item.addressId" @click="checkedIndex=index">
```

8.用v-if指令动态渲染是否显示默认地址和设为默认

```
<div class="addr-opration addr-set-default" v-if="!item.isDefault">
                                <a href="javascript:;" class="addr-set-default-btn"><i>设为默认</i>
/a>
                            </div>
                            <div class="addr-opration addr-default" v-if="item.isDefault">默认地
</div>
```

9.设为默认

```
setDefault(addressId) {
            this.addressList.map((item) => {
                if (addressId == item.addressId) {
                    item.isDefault = true;
                } else {
                    item.isDefault = false;
```

```
            }
        })
    }
```

需要吧这个地址的id传过来

```
<div class="addr-opration addr-default" v-if="item.isDefault">默认地址</div>
```

10.删除地址

```
//删除地址
    delAddress(addressId) {
        this.addressList.map((item, index) => {
            if (addressId == item.addressId) {
                this.addressList.splice(index, 1);
            }
        })
    }
```

同样需要把这个id给传进去

```
<a href="javascript:;" class="addr-del-btn" @click="delAddress(item.addressId)">
```

11.

在下一步那里加上一个弹框：

```
<modal :mdShow="modalConfirm" @close="closeModal">
        <template v-slot:message>
            <p>结束咯！！！</p>
        </template>
        <template v-slot:btnGroup>
            <a class="btn btn--m btn--red" href="javascript:;" @click="modalConfirm=false">
闭</a>
        </template>
    </modal>
```

12.在说一下从父组件传递过来的属性，在子组件中不能修改，如果想修改必须通过父组件，否者这属性会失去意义。

今天算是把这个课学完了，明天做一个总结把，然后就开始新的课程！