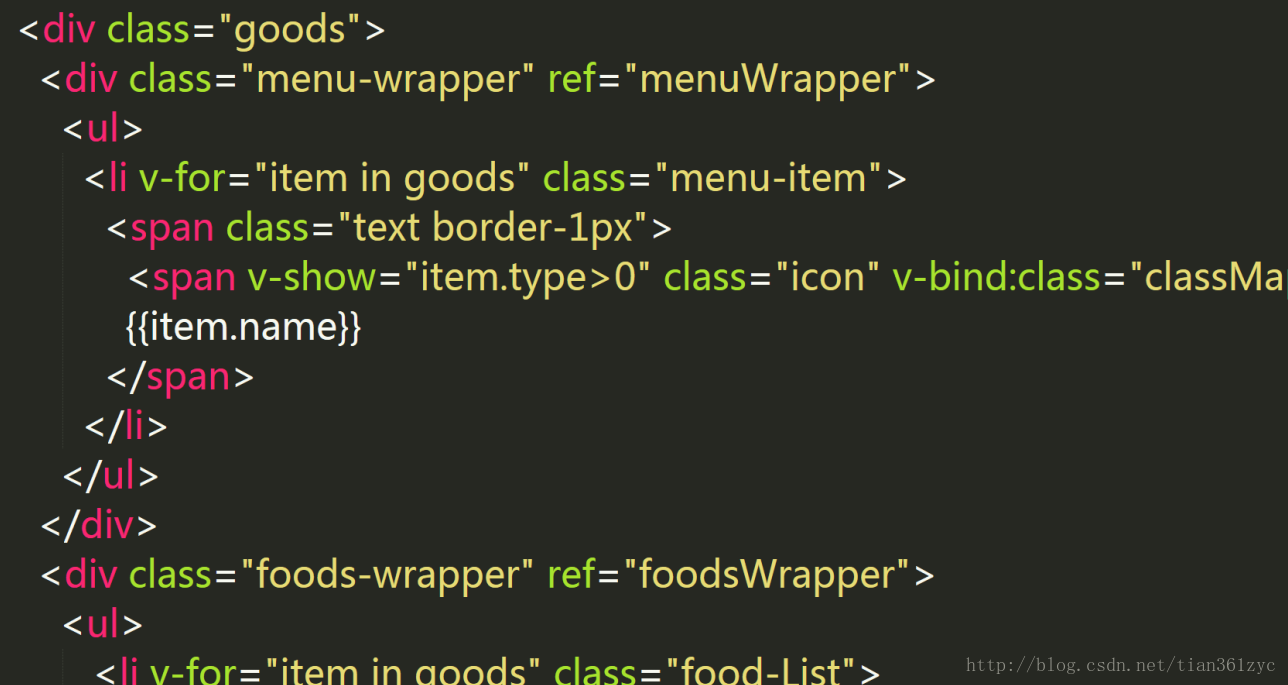
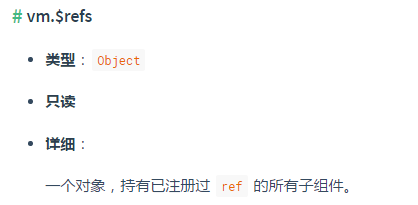
## **ref 被用来给[DOM](https://www.baidu.com/s?wd=DOM&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "https://blog.csdn.net/weixin_41646716/article/details/_blank)元素或子组件注册引用信息。引用信息会根据父组件的 $refs 对象进行注册。如果在普通的DOM元素上使用，引用信息就是元素; 如果用在子组件上，引用信息就是组件实例**

# **注意：只要想要在Vue中直接操作DOM元素，就必须用ref属性进行注册**





# 2.**[Vuex中mapState的用法](https://www.cnblogs.com/meiyh/p/6805987.html)**

import Vue from 'vue'

import Vuex from 'vuex'

import mutations from './mutations'

import actions from './action'

import getters from './getters'

Vue.use(Vuex)

const state = {

userInfo: { phone: 111 }, //用户信息

orderList: [{ orderno: '1111' }], //订单列表

orderDetail: null, //订单产品详情

login: false, //是否登录}

export default new Vuex.Store({

state,

getters,

actions,

mutations,

})

computed: {

...mapState([

'orderList',

'login'

]),

},

mounted(){

console.log(typeof orderList); ==>undefind

console.log(typeof this.orderList)==>object

}

mapState通过扩展运算符将store.state.orderList 映射this.orderList  这个this 很重要，这个映射直接映射到当前Vue的this对象上。

所以通过this都能将这些对象点出来，同理，mapActions, mapMutations都是一样的道理。

# 3.[Vue.nextTick 的原理和用途](https://segmentfault.com/a/1190000012861862)

* 用法：

在下次 DOM 更新循环结束之后执行延迟回调。在修改数据之后立即使用这个方法，获取更新后的 DOM。

Vue 实现响应式并****不是数据发生变化之后 DOM 立即变化****，而是按一定的策略进行 DOM 的更新。

1. 所有同步任务都在主线程上执行，形成一个执行栈（execution context stack）。  
   （2）主线程之外，还存在一个"任务队列"（task queue）。只要异步任务有了运行结果，就在"任务队列"之中放置一个事件。  
   （3）一旦"执行栈"中的所有同步任务执行完毕，系统就会读取"任务队列"，看看里面有哪些事件。那些对应的异步任务，于是结束等待状态，进入执行栈，开始执行。  
   （4）主线程不断重复上面的第三步。

# **vue中使用better-scroll实现滑动效果**

一、首先需要在项目中引入better-scroll

1. 在package.json 直接写入 "better-scroll":"^1.11.1"  版本以github上为准（目前最新）

2.cpnm install  在node\_modules  可以查看版本是否安装

3.直接在你的组件里面写入import BScroll from 'better-scroll';

二、better-scroll优点

1.体验像原生：滚动非常流畅，而且没有滚动条。

2.滚动位置固定：在vue中通过路由切换页面时组件会自动滚动到顶部，需要监听滚动行为才能让滚动位置固定，better-scroll解决了这个问题。