项目讲解vue

1. 目录组织
   1. Assets 放资源
      1. Css
         1. Base.css
         2. normalize.css
      2. Img
   2. Views 放页面
   3. Components 放公共组件
      1. Common 项目间的公共组件
      2. Content 项目内的公共组件
   4. Common 放项目中的公共内容(公共方法)
      1. Common.js
      2. Util
   5. Network 放axios用到的文件
   6. Store 放vuex用到的文件
      1. Index.js
      2. modules
   7. Router 放vue-router用到的文件
      1. index.js
2. 项目别名
   1. 新建vue.config.js文件
   2. 导出对象
      1. ConfigureWebpack对象
         1. Resolve对象
            1. Alias对象（别名）--在dom中使用前面加上~
            2. Extensions数组（省略的后缀名）
3. 引用组件
   1. 引用顺序与组件注册顺序应该一致
   2. 公共组件 子组件 引用方法 分别为一组 以空行分隔
4. 插槽
   1. 封装组件时，如果大体样式相同，但又不全相同时，使用插槽
      * 1. 如果样式完全一样，只有文字不同，则没必要使用
5. Position:sticky 一旦达到设置的位置(top left ...) 就固定 移动端一般支持 IE不定
6. created初始化内容
   1. 应当只展示逻辑，具体的代码应当放在methods中
7. methods中
   1. 监听事件的方法
   2. 网络请求的方法
   3. 空行分隔
8. 监听组件原生事件：event.native
9. 组件化开发思想+1：面对组件时，只需调用组件提供的方法、数据，而不应操控组件内部的具体逻辑；
   1. 组件化思想+1：在组件内部，只负责实现组件内的逻辑、提供数据、接口即可，不要掺杂其他组件的逻辑
   2. 组件化思想+1：当一个方法可能被多次、多个地方使用时，放在单独文件中，供各个组件使用
10. 数据比较复杂时，应该设计合适的数据结构，方便保存和使用
    1. 请求回来的数据复杂时，应该对数据进行整合（在公共文件中设置一个类，组件内部直接创建实例，把对应数据传入即可）
11. 监听图片加载完成：
    1. 原生：img.onload
    2. Vue:@load
12. Debounce 防抖动
    1. 刷新过于频繁时，可以添加延迟
    2. 针对频繁触发
    3. 多次事件，一次触发
13. 节流函数throttle
    1. 在一定时间内，限制一个动作只执行一次（针对频繁触发）
    2. 多次事件，平均分配触发
14. 事件总线：在vue的原型中添加属性$bus,赋值为vue实例
    1. 拥有vue实例中的$emit $on 属性
    2. 实现全局范围内事件监听、触发
    3. 适用于非父子组件、层级嵌套过多 的情况
15. 事件循环 setTimeout在事件循环中处于最后
16. 切换页面 保存位置状态
    1. Keepalive
    2. Actived unactived

联动：offsetTop、scrollToElement

17.vuex重构：

比较复杂的逻辑也需要放在actions中，mutations只负责单一简单的操作

State.list.find( () => { } ) 在list中寻找符合条件的项目 并返回

18.mixin混入 区分可否混入

新建对象（类似于vue的对象data methods components created mounted。。）

其中：相同的内容可以合并 （包括生命周期的钩子函数） 但是methods中具体的函数不能合并内容

Vue对象中mixins属性的值为数组，各个元素为新建的mixin对象

1. Getter中比较简单，在具体的组件中调用getters比较复杂，且个数较多时，用mapGetters
   1. Import {mapGetters} from ‘vuex’
   2. 使用方法
      1. ...mapGetters( [‘’,’’] ) 需要引用的部分
      2. ...mapGetters({ name: ‘ ’ }) 将引用的部分进行自命名
2. Swipe--vant 研究轮播图实现--》看视频？

Websocket

Let ws = new WebSocket(‘ws://url...’)

Ws.onopen = function(){}

Ws.onerror = function(){}

Ws.onmessage = function(msg){

//接收消息

JSON.parse(msg)

}

