项目开发：

新项目：

划分目录结构

引用了两个css文件

Vue.config.js和editorconfig

项目的模块划分:tabbar->路由映射关系

首页开发

Navbar的封装

网络数据的请求

轮播图

推荐信息

一、FeatureView

独立组件封装FeatureView

div>a>img

二、TabControl

独立组件的封装

Props->titles

div>根据titles v-for遍历div -> span{{title}}

css相关布局

选中哪一个tab，哪一个tab的文字颜色变色，下面border-bottom

currentIndex

三、首页商品数据的请求

3.1设计数据结构，用于保存数据

3.2 发送数据请求

在home.js中封装getHomeGoods(type,page);

在Home.vue中，又在methods中getHomeGoods(type);

调用getHomeGoods(pop)/ getHomeGoods(new)/ getHomeGoods(sell)

Page:动态获取对应的page

获取数据

This.goods[type].list.push(…res.data.list)

This.goods[type].page +=1

四、对数据进行展示

封装GoodsList.vue组件

Props:goods->list[30]

v-for goods->GoodsListItem[30]

GoodListItem(组件)->GoodsItem(数据)

封装GoodsListItem.vue组件

Props:goodsItem

GoodsItem取出数据，并且使用正确的div/span/img基本标签进行展示

五、对滚动进行重构：Better-Scroll

5.1 在index.html中使用Better-Scroll

Const bscroll = new BScroll(el,{})

注意：wrapper->content->很多内容

1. 监听滚动

probeType:0/1/2(手指滚动)/3(只要滚动)

bscroll.on(‘scroll’,(position)={})

1. 上拉加载

pullUpload:true

bscroll.on(‘pullingUp’,()=>{})

3.click:false

Button可以监听点击

Div不可以

5.2在Vue项目中使用Better-Scroll

在Profile.vue中简单的演示

对Better-Scroll进行封装:Scroll.vue

[Home.vue和Scroll.vue](http://Home.vue和Scroll.vue)之间进行通信

Home.vue将probeType设置为3

Scroll.vue需要通过$emit,试试将时间发阿松到Home.vue

六、回到顶部BackTop

6.1 对BackTop.vue组件封装

6.2 如何监听组件的点击

直接监听back-top的点击，但是不能直接监听，必须添加修饰.native。

回到顶部

Scroll对象，scroll.scrollTo(x,y,time)

This.$refs.scroll.scrollTo(0,0,500)

6.3 BackTop组件的显示和隐藏

isShowBackTop:false

监听滚动，拿到滚动的位置：

-position.y>1000->isShowBackTop:true

isShowBackTop =-position.y>1000

七、解决首页可滚动区域的问题

Better-Scroll在决定由多少区域可以滚动时，是根据scrollerHeight属性决定的

ScrollerHeight属性是根据放Better-Scroll的content中的子组件的高度，但是在该首页中，刚开始在计算scrollerHeight属性时，时没有将图片计算在内的；所以，计算出来的高度时错误的；后来图片加载进来之后有了新的高度，但scrollerHeight属性并没有进行更新，所以滚动出现了错误。

如何解决此问题：

监听每一张图片是否加载完成，只要有一张图片加载完成，执行一次refresh()

如何监听图片加载完成了？

原生的js监听图片：img.onload = function(){}

Vue中监听@load=’方法’

调用scroll的refresh()

如何将GoodsListItem.vue中的事件传入到Home.vue中

因为设计到非父子组件的通讯，所以选择事件总线：

Bus->总线；

Vue.prototype.$bus = new Vue();

This.bus.emit(‘事件名称’,参数);

This.bus.on(‘事件名称’,回调函数(参数))

问题一:refresh找不到的问题

1.在Scroll.vue中，调用this.scroll的方法之前，判断this.scroll对象是否有值；

2.在mounted生命周期函数中使用this.$refs.scroll而不是created中

需求：对refresh非常频繁的问题，进行防抖操作

防抖的过程：

如果直接执行refresh，则refresh函数会被执行30次；

将refresh函数传入到debounce函数，生成一个新的函数

之后再调用非常频繁的时候，使用新生成的函数

而新生成的函数，并不会非常频繁的调用，如果下次执行来的非常快，会将上一次取消掉：

function debounce(func, wait) {

let timeout = null

return function() {

let context = this

let args = arguments

if (timeout) clearTimeout(timeout)

timeout = setTimeout(() => {

func.apply(context, args) //这步有点没有懂

}, wait)

}

}

八、上拉加载更多

九、tabControl的吸顶效果

9.1 获取到tabControl的offsetTop

必须知道滚动到多少时，开始有吸顶效果，此时需要获取tabControl的offsetTop，但是，如果直接再mounted中获取tabControl的offsetTop，则值不正确。

如何获取正确的值？

监听HomeSwiper中的img的加载完成；加载完成后，发出时间，再Home.vue中，获取正确的值。

补充：为了不让HomeSwiper多次发出事件，可以使用isLoad的变量进行状态的记录。

9.2 监听滚动，动态的改变tabControl的样式

问题：动态的改变tabControl的样式时，出现两个问题：

问题一：下面的商品内容，会突然上移；

问题二：tabControl虽然设置fixed，但也随着Better-Scroll一起滚出去。

其他方案解决停留问题：

再最上面，多赋值一份PlaceHolderTabControl组件对象，利用它来实现停留效果。

当用户滚动到一定位置时，PlaceHolderTabControl显示出来 。

当用户滚动没有达到一定位置时，PlaceHolderTabControl隐藏起来。

十、让Home保持原来的状态

10.1 让Home不要随意销毁掉 keep-alive

10.2 让Home中的内容保持原来的位置

离开时，保存一个位置信息saveY；

进来时，将位置设置为原来保存的位置saveY信息即可：

注：最好回来时，进行一次refresh()

详情页实现思路：

一：点击商品进入详情页

GoodsListItem点击；

点击之后获取商品IID，跳转到详情页，并传入IID

二：解决——首页保持位置状态

Deactivated：记录离开时的位置；

Activated：通过scrollTo函数，设置离开时位置

三：详情页的导航栏实现

返回按钮；

标题列表的展示：center

四：请求详情的数据

接口：/detail?iid=?

五：流波图的实现：

Swiper/SwiperItem

六：商品基本信息的展示

数据俩字四面八方：

对数据进行汇总：一个对象当中；

一个对象传入到子组件中

七：店铺信息的展示

八：商品图片的展示

九：参数信息的展示

十：评论信息的展示

时间格式化；

服务器返回的时间戳->date->格式化 yyyy-MM-dd hh:mm:ss

十一：推荐数据的展示

请求推荐数据；

GoodsList展示数据

十二：mixin的使用

创建混入对象：const mixin = {}

组件对象中：mixins:[mixin]

十三：标题和内容的联动效果（点击标题，滚动到对应的主题；内容滚动，显示正确的标题））

十四：顶部工具栏的封装

十五：详情页回到顶部

十六：点击加入购物车

16.1 监听加入购物车按钮的点击，并且获取商品信息（监听；获取商品信息:iid/price/image/title/desc）

16.2 将商品添加到Vuex中（安装Vuex，配置Vuex，定义mutations，将商品添加到state.cartList，重构代码：将mutations中的diamagnetic抽取actions中(定义了两个mutations)，将mutations/actions单独抽取到文件中）

十七：购物车的展示

17.1 购物车的导航栏的展示

17.2 购物车商品的展示(CartList->Scroll(滚动问题),CartListItem->CheckButton)

17.3 商品的选中和不选中

修改模型对象，改变选中和补选中

17.4 底部工具栏的汇总(全选按钮,计算总价格,去计算)

十八：购物车全选按钮

显示状态：判断有一个不选中，则不选

十九：添加购物车弹窗

19.1

Actions可以返回一个Promise

mapActions的映射关系

19.2

Toast(吐司)封装(普通封装和插件封装)

二十：补充细节

20.1 在移动端：

fastClick减小移动端300ms的点击延迟(安装，导入，调用attach函数)

20.2 图片的懒加载

什么是图片懒加载？图片需要时再加载

使用vue-lazyload(安装，导入，Vue.use，修改 img:src->v-lazy)

20.3 px2vw插件使用

安装插件

在postcss.config.js中配置

面试题：

如何理解Vue生命周期

如何进行非父子组件通信

Vue的响应式原理