1、需求分析，按照设计稿搞懂需求

2、项目资源准备，引入需要的图片素材，没有切好的图有自己在PSD文件切，一般移动端都有@2x和@3x的图片，这里涉及到设备像素比，参考文章为张鑫旭老师的文章：http://www.zhangxinxu.com/wordpress/2012/08/window-devicepixelratio/，比如iPhone6是2，iPhone6p是3；在使用webpack作为打包工具时，最好不使用精灵图，而是使用单张图片，因为webpack会把图片打包

3、图标字体的制作，如果有svg图片，参考http://blog.csdn.net/qq\_34543438/article/details/73825973

4、项目目录设计，在src目录下，

components存放各种组件，脚手架工具默认的是直接放.vue文件，但是建议，再加一个子目录

组件的一个原则是就近维护，比如说，在header文件夹下写一个header.vue的文件，同时还可以存放图 片文件

common 存放公共的模块资源，通常有js、fonts、stylus（css预处理器，与less，sass相似）

5、mock数据，即模拟后台数据，

打开build文件夹下的dev-server.js文件（webpack打包的入口文件），在var app = express()下面编写

如：

/\* 1读取data.json中的数据\*/

var appData = require('../data.json');

var seller = appData.seller;

var goods = appData.goods;

var ratings = appData.ratings;

/\* 2编写路由\*/

var apiRoutes = express.Router();

apiRoutes.get('/seller', function (req, res) {

res.json({

/\*返回的错误码\*/

errno: 0,

data: seller

});

});

apiRoutes.get('/goods', function (req, res) {

res.json({

errno: 0,

data: goods

});

});

apiRoutes.get('/ratings', function (req, res) {

res.json({

errno: 0,

data: ratings

});

});

/\*3挂载实例\*/

app.use('/api', apiRoutes);

6、组件拆分

添加reset.css文件，进行样式的初始化，参考：http://cssreset.com

在index.html引入reset.css文件，

在index.html加上视口<meta name="viewport"

content="width=device-width,initial-scale=1.0,maximum-scale=1.0,minimum-scale=1.0,user-scalable=no">

报semi错误时：(默认不能有分号，想要写分号的话)在.eslintrc.js中的rules对象中添加'semi': ['error','always']

报indent错误时：在.eslintrc.js中的rules对象中添加 'indent':0

/\* eslint-disable no-new \*/的作用：跳过规则校验，此时new Vue()没有赋值也不会报错

7、Vue-router的了解，在main.js中通过import 引入（这里自己用的是config.js）

1. import VueRouter from 'vue-router';//引入vue-router（可以在node\_modules中的vue-router包下的package.json中查看name对应的值即为from后的vue-router，import后的是一个变量）

2. Vue.use(VueRouter);//安装

3. <a v-link="{path:'/goods'}">商品</a>===================><router-link to="/goods">商品</router-link>

4.<router-view></router-view>路由外链，改变路由，这里的内容会发生改变

5. 在components下的goods组件中，创建goods.vue,分为template、script、style三大部分

template下是内容区域，script下写export default {}【用于导出】;style下是样式

在main.js中还需引入goods组件，import goods from 'components/goods/goods';

然后写在关联中的component后，即goods【其他的路由同上】

1.0路由的写法：

//1. 准备一个根组件

var app=Vue.extend(App);

//2. 准备路由

var router=new VueRouter();

//3. 关联

router.map({

'/goods':{

component:goods

}

});

//4. 启动路由（启动后才生效，把内容渲染到router-view标签，第二个参数是挂载点【挂载点在index.html中】）

router.start(app,'#box');

2.0路由的写法：

const routes = [{

path: '/',

redirect: '/goods'

}, {

path: '/goods',

component: goods

}, {

path: '/ratings',

component: ratings

}, {

path: '/seller',

component: seller

}];

const router = new VueRouter({

linkActiveClass: 'active',

routes

});

/\* eslint-disable no-new \*/

new Vue({

el: '#app',

router,

render: h => h(App)

});

8、在webpack.base.conf.js可以修改引入，如：import goods from 'components/goods/goods';的路径

alias：{。。。}

9、修改webpack的配置文件后，项目需要重新启动

10、路由（商品、评论、商家），a标签设置display:block，撑满父盒子

11、1像素边框的实现 http://blog.csdn.net/qq\_34543438/article/details/73839086

1. Vue-resource

**App.vue**

<template>

<div>

<v-header :seller="seller"></v-header> //3、v-bind的缩写，把seller对象传给header组件

<div class="tab border-1px">

<div class="tab-item">

<router-link to="/goods">商品</router-link>

</div>

<div class="tab-item">

<router-link to="/ratings">评论</router-link>

</div>

<div class="tab-item">

<router-link to="/seller">商家</router-link>

</div>

</div>

<keep-alive>

<router-view :seller="seller"></router-view>

</keep-alive>

</div>

</template>

<script type="text/ecmascript-6">

import {urlParse} from 'common/js/util';

import header from 'components/header/header.vue';

const ERR\_OK = 0;

export default {

data() {

return {

seller: { //2、这里的seller对象是ajax请求获得

id: (() => {

let queryParam = urlParse();

return queryParam.id;

})()

}

};

},

created() {

this.$http.get('/api/seller?id=' + this.seller.id).then((response) => {

response = response.body; //1、通过交互获得seller对象

if (response.errno === ERR\_OK) {

this.seller = Object.assign({}, this.seller, response.data);

}

});

},

components: {

'v-header': header

}

};

</script>

<style lang="stylus" rel="stylesheet/stylus">

@import "./common/stylus/mixin.styl"

............................

</style>

编写代码风格，数字提升为常量，例const ERR\_OK = 0;

**Header.vue**

<template>

<div class="header">

<div class="content-wrapper">

<div class="avatar">

<img width="64" height="64" :src="seller.avatar">

</div>

<div class="content">

<div class="title">

<span class="brand"></span>

<span class="name">{{seller.name}}</span>

</div>

<div class="description">

{{seller.description}}/{{seller.deliveryTime}}分钟送达

</div>

<div v-if="seller.supports" class="support"> //6、这里必须加上v-if 因为防止开始获取到App.vue中data中的seller空对象，created中的get请求是异步的过程，开始初始化的时候seller是一个空对象，把空对象传给header组件时，header.vue中的seller.supports就是undefined，[0]取第一个就会报错

<span class="icon" :class="classMap[seller.supports[0].type]"></span>

<span class="text">{{seller.supports[0].description}}</span>

</div>

</div>

<div v-if="seller.supports" class="support-count" @click="showDetail">

<span class="count">{{seller.supports.length}}个</span>

<i class="icon-keyboard\_arrow\_right"></i>

</div>

</div>

<div class="bulletin-wrapper" @click="showDetail">

//10、实现公告效果的重要css属性 white-space: nowrap

// overflow: hidden

// text-overflow: ellipsis

<span class="bulletin-title"></span><span class="bulletin-text">{{seller.bulletin}}</span>

<i class="icon-keyboard\_arrow\_right"></i>

</div>

<div class="background"> //11.背景使用的是z-index:-1底层,filter: blur(10px)模糊,.header的背景此时必须为background: rgba(7, 17, 27, 0.5)，透明度0.5是必须的

<img :src="seller.avatar" width="100%" height="100%">

</div>

<transition name="fade">

<div v-show="detailShow" class="detail"> //12、弹层，使用的是css Stricky footers布局（重点）[detail-wrapper设置最小高度为整屏的高度，detail-main

设置padding-bottom，detail-close设置负的margin-top值]

<div class="detail-wrapper clearfix">

<div class="detail-main">

<h1 class="name">{{seller.name}}</h1>

<div class="star-wrapper">

<star :size="48" :score="seller.score"></star>

</div>

<div class="title">

<div class="line"></div>

<div class="text">优惠信息</div>

<div class="line"></div>

</div>

<ul v-if="seller.supports" class="supports">

<li class="support-item" v-for="(item,index) in seller.supports">

<span class="icon" :class="classMap[seller.supports[index].type]"></span>

<span class="text">{{seller.supports[index].description}}</span>

</li>

</ul>

<div class="title">

<div class="line"></div> // 19、通过介绍遮罩层横线的书写，引出了Postcss

Postcss是根据caniuse（http://caniuse.com/）官网去写代码

<div class="text">商家公告</div>

<div class="line"></div>

</div>

<div class="bulletin">

<p class="content">{{seller.bulletin}}</p>

</div>

</div>

</div>

<div class="detail-close" @click="hideDetail">

<i class="icon-close"></i>

</div>

</div>

</transition>

</div>

</template>

<script type="text/ecmascript-6">

import star from 'components/star/star';

export default {

props: { //4、props属性去接收传过来的seller对象

seller: {

type: Object //5、类型是Object，接下来就是dom结构的书写了

}

},

data() { //13、vue中的data一定是function，return的对象中写上数据的初始值

return {

detailShow: false

};

},

methods: {

showDetail() {

this.detailShow = true;

},

hideDetail() {

this.detailShow = false;

}

},

created() {

this.classMap = ['decrease', 'discount', 'special', 'invoice', 'guarantee'];

// 9、这里面放的都是图片的名称，与样式中的bg-image()和结构中的 classMap[seller.supports[index].type]相联系

},

components: {

star

}

};

</script>

<style lang="stylus" rel="stylesheet/stylus">

@import "../../common/stylus/mixin"; //8、引入mixin，即可使用自己写的bg-image（）

............................

// 19、注意点：过度效果中vue1.0 与 vue 2.0 的区别

// backdrop-filter: blur(10px) 添加模糊的效果（ios支持）

</style>

在stylus文件夹下的mixin.styl中

**Mixin.styl**

border-1px($color)

position: relative

&:after

display: block

position: absolute

left: 0

bottom: 0

width: 100%

border-top: 1px solid $color

content: ' '

border-none()

&:after

display: none

bg-image($url) // 7、匹配@2x.@3x.的图片

background-image: url($url + ["@2x.png")](mailto:\"@2x.png\")) //普通一般都用@2x.png

@media (-webkit-min-device-pixel-ratio: 3),(min-device-pixel-ratio: 3)

background-image: url($url + ["@3x.png")](mailto:\"@3x.png\")) //如果最小的dpr为3，就使用@3x.png

**Star.vue**

<template>

<div class="star" :class="starType">

<span v-for="(itemClass,index) in itemClasses" :class="itemClass" class="star-item" key="index"></span>

</div>

</template>

<script type="text/ecmascript-6">

const LENGTH = 5;// 17、5代表5颗星，

const CLS\_ON = 'on';

const CLS\_HALF = 'half';

const CLS\_OFF = 'off';

export default {

props: { //15、接收size和score两个对象，类型是Number

size: {

type: Number

},

score: {

type: Number

}

},

computed: {

starType() { //16、通过此函数，确定五角星的类型 与结构中:class相匹配

return 'star-' + this.size;

},

itemClasses() { //18、这个函数是关键，写完后header.vue中引入并注册star组件，并在结构中找对位置写入<star :size="48" :score="seller.score"></star>

let result = [];

let score = Math.floor(this.score \* 2) / 2;

let hasDecimal = score % 1 !== 0;

let integer = Math.floor(score);

for (let i = 0; i < integer; i++) {

result.push(CLS\_ON);

}

if (hasDecimal) {

result.push(CLS\_HALF);

}

while (result.length < LENGTH) {

result.push(CLS\_OFF);

}

return result;

}

}

};

</script>

<style lang="stylus" rel="stylesheet/stylus">

@import "../../common/stylus/mixin.styl"

</style>

**Goods商品列表页开发**

1. **定义视口，给goods绝对定位，设定top与bottom的值，宽度为100%**
2. **使用flex布局，goods添加属性display:flex,左侧属性flex: 0 0 80px ；width: 80px ，这里的width：80px必须写，不写在安卓浏览器会出问题【搞懂fix后的三个值】**

flex 属性用于设置或检索弹性盒模型对象的子元素如何分配空间。

flex 属性是 flex-grow、flex-shrink 和 flex-basis 属性的简写属性。

**注意：**如果元素不是弹性盒模型对象的子元素，则 flex 属性不起作用。

1. **右测属性 flex:1**

**商品详情页实现**

**1、第一步还是，在food文件夹下新建一个food.vue 的组件，组件的最初结构为**

**<template></template>**

**<script type="text/ecmascript-6">**

**export default {**

**</script>**

**<style lang="stylus" rel="stylesheet/stylus">**

**@import "../../common/stylus/mixin.styl"**

**</style>**

1. **组建中的变量是私有的话变量前面加 \_ 如 \_show 对比与外部可调用的情况**
2. **padding-top:100% 与width:100%的配合使用，为的是得到一个高与宽相等的容器**

**Seller组件**

**如何使用better-scroll：**

**首先import引入，。。【这里涉及到点击重新刷新的设置】**

**最后总结**

**Npm run build 打包 其实就是执行的package.json下的script下的build**

**Webpack.prod.conf.js 中的ExtractTextPlugin是webpack提供的一个插件，可以吧编译过程中的CSS文件，提取出来，形成独立的文件，而不是打包到js文件中**

**自己写了个http server 名为：Prod.server.js 作为服务【或者说是异步api】**

**实际项目中，直接把dist文件夹扔给后台，或者上到CDN**

**过程就是npm run build打包，node prod.server.js开启服务【这里的端口号是在index.js文件中】**

此时在地址栏输入http://localhost:9000/#/seller可以进行预览，在

Network中可以看到打包后的css以及js文件

真是的环境下项目应该是不能进行调试的，可以在index.js下的build对象中吧productionSourceMap的值改为false，然后重新进行打包npm run build,[可以看到打包后的跟之前的哈希值不一样了]，最后**node prod.server.js开启服务,这时在浏览器中文件不可调试**

Vue官网

1、Vue.component() 就是注册组件

定义名为todo-item的新组件

// 定义名为 todo-item 的新组件

Vue.component('todo-item', {

template: '<li>这是个待办项</li>'

})

2、组件模板：<组件名称></组件名称>

<ol>

<!-- 创建一个 todo-item 组件的实例 -->

<todo-item></todo-item>

</ol>

3、props属性，用于子组件接收父组件的数据

|  |
| --- |
| Vue.component('todo-item', {  // todo-item 组件现在接受一个  // "prop"，类似于一个自定义属性  // 这个属性名为 todo。  props: ['todo'],  template: '<li>{{ todo.text }}</li>'  }) |

1. 当使用 vue-loader 或 vueify 的时候，\*.vue 文件内部的模板会在构建时预编译成 JavaScript。

5、插值语法中可以这样写

|  |
| --- |
| {{ number + 1 }}  {{ ok ? 'YES' : 'NO' }}  {{ message.split('').reverse().join('') }}  <p>Original message: "{{ message }}"</p>  <p>Computed reversed message: "{{ reversedMessage }}"</p> |

不可以这样

<!-- 这是语句，不是表达式 -->

{{ var a = 1 }}

<!-- 流控制也不会生效，请使用三元表达式 -->

{{ if (ok) { return message } }}

6、过滤器可以串联：

|  |
| --- |
| {{ message | filterA | filterB }} |

7、计算属性computed与methods的对比

相同点：我们可以将同一函数定义为一个 method 而不是一个计算属性。对于最终的结果，两种方式确实是相同的

不同点：****计算属性是基于它们的依赖进行缓存的****。计算属性只有在它的相关依赖发生改变时才会重新求值。这就意味着只要 message 还没有发生改变，多次访问 reversedMessage 计算属性会立即返回之前的计算结果，而不必再次执行函数。

相比而言，只要发生重新渲染，method 调用****总会****执行该函数。

如果你不希望有缓存，请用 method 替代。

8、Class属性的三种写法

1、 <div class="static"

v-bind:class="{ active: isActive, 'text-danger': hasError }">

</div>

data: {

isActive: true,

hasError: false

}

渲染为:

<div class="static active"></div>

2、你也可以直接绑定数据里的一个对象：

|  |
| --- |
| <div v-bind:class="classObject"></div> |

|  |
| --- |
| data: {  classObject: {  active: true,  'text-danger': false  }  } |

渲染的结果和上面一样。

3、我们也可以在这里绑定返回对象的**[计算属性](https://cn.vuejs.org/v2/guide/computed.html)**。这是一个常用且强大的模式：

|  |
| --- |
| <div v-bind:class="classObject"></div> |

|  |
| --- |
| data: {  isActive: true,  error: null  },  computed: {  classObject: function () {  return {  active: this.isActive && !this.error,  'text-danger': this.error && this.error.type === 'fatal',  }  }  } |

### [4、数组语法](https://cn.vuejs.org/v2/guide/class-and-style.html" \l "数组语法)

我们可以把一个数组传给 v-bind:class ，以应用一个 class 列表：

|  |
| --- |
| <div v-bind:class="[activeClass, errorClass]"> |

|  |
| --- |
| data: {  activeClass: 'active',  errorClass: 'text-danger'  } |

渲染为:

|  |
| --- |
| <div class="active text-danger"></div> |

如果你也想根据条件切换列表中的 class ，可以用三元表达式：

|  |
| --- |
| <div v-bind:class="[isActive ? activeClass : '', errorClass]"> |

此例始终添加 errorClass ，但是只有在 isActive 是 true 时添加 activeClass 。

不过，当有多个条件 class 时这样写有些繁琐。可以在数组语法中使用对象语法：

|  |
| --- |
| <div v-bind:class="[{ active: isActive }, errorClass]"> |

1. v-bind:style 的对象语法十分直观——看着非常像 CSS ，其实它是一个 JavaScript 对象。

CSS 属性名可以用驼峰式 (camelCase) 或 (配合引号的) 短横分隔命名 (kebab-case)： 【用法同上:class】

|  |
| --- |
| <div v-bind:style="{ color: activeColor, fontSize: fontSize + 'px' }"></div> |

1. 用key管理可复用的元素

元素加上key属性的时候，每次切换都会重新渲染

11、**[v-if 与 v-for 一起使用](https://cn.vuejs.org/v2/guide/conditional.html" \l "v-if-与-v-for-一起使用)**

当 v-if 与 v-for 一起使用时，v-for 具有比 v-if 更高的优先级。

1. vue2.0中的循环遍历

<li v-for="(item, index) in items">

你也可以用 of 替代 in 作为分隔符，因为它是最接近 JavaScript 迭代器的语法：

【使用of和使用in的意义相同】

|  |
| --- |
| <div v-for="item of items"></div> |

第二个的参数为键名：第三个参数为索引：

<div v-for="(value, key, index) in object">

**实例代码：**

<body>

<ul id="repeat-object" class="demo">

<li v-for="(value, key, index) in object">

{{ index }}. {{ key }} : {{ value }}

</li>

</ul>

</body>

<script>

new Vue({

el: '#repeat-object',

data: {

object: {

firstName: 'John',

lastName: 'Doe',

age: 30

}

}

})

</script>

**2.2.0+ 的版本里，当在组件中使用 v-for 时，key 现在是必须的。**

13、****Node Package Manager 简称 npm****