1. 开发项目流程
2. 产品立项：（产品经理、项目经理）
3. 客户的需求分析
4. 整理需求文档
5. 项目成立会
6. 产品原型
7. 设计产品原型图—Axure RP工具等
8. UI设计—ps工具、AI工具等
9. 项目开发

项目需求分析

项目工期评估

项目责任划分

前端

1. 静态页面制作 根据UI设计稿
2. 前端框架选型 vue react
3. 前端页面架构

后端

1. 数据库开发（数据库创建、配置、表的设计和创建）
2. API接口文档
3. API接口实现
4. 项目测试
   1. 开发测试
      1. 单元测试
      2. E2E测试
   2. 灰度测试
      1. 功能
      2. 压力
      3. 服务器
      4. 等
5. 项目上线
   1. 打包前端模块（代码合并）
   2. 远程服务器部署
   3. 监控异常

多人协作时的代码管理：

首先创建一条master主干，并init初始化；

然后创建一条develop的开发分支，开发人员提交代码到该分支，由项目经理进行代码合并；开发过程中某个人负责某个模块，也可以拉一条分支，等模块开发完毕再进行合并到develop上。

产生bug的时候，给bug拉一条单独的分支，bug解决后进行合入。

注：分支完成后记得删除。

初始化项目：

静态布局demo（非必要）

数据接口api(后端提供，未写好之前可以用假数据代替)

接口文档一般包含：请求方式get、post，请求头信息，请求返回结果

通过后端返回结果，前端把数据渲染到页面

搭建

接口文档：

正在热映：

即将上映：

[http://39.97.33.178/api/movieComingList?cityId=50](http://39.97.33.178/api/movieComingList?cityId=10)

搜索：

<http://39.97.33.178/api/searchList?cityId=10&kw=a>

城市：

<http://39.97.33.178/api/cityList>

电影详情：

<http://39.97.33.178/api/detailmovie?movieId=345808>

影院：

<http://39.97.33.178/api/cinemaList?cityId=10>

城市定位：

http://39.97.33.178/api/getLocation

1. <router-link>一般代替ul中的li作为导航栏的路由跳转；跳转之后vue会自动提供一个类：li**.router-link-active**{ color: ‘red’ } 通过该类可以设置跳转之后该li的样式

注：原来li的类的样式是li.active{ color: ‘red’ }

1. vue中组件化开发可以在src目录下创建views文件夹，里面存放各个模块的主要vue文件，其中每个组件(如：电影、影院、我的)中部分内容可以进行组件化（如：头部和尾部的导航栏切换），然后在该vue文件内进行引入。
2. 二级路由设置，比如当跳转到电影页面时，需要有城市切换、热映电影列表、搜索框；这就需要设置二级路由，在路由处设置children，然后在该模块的vue文件中的template中放置<router-view>；另，别忘了在<router-view>加上<keep-alive>
3. 在跳转的时候需要修改组件中的文字内容；如（movie模块中的header组建中的title）需要通过路由跳转到对应的模块时进行变化，这就需要用父子组件通信，使用props即可。
4. vue中使用了webpack，在对img的src属性进行绑定时，引入地址时需要包裹require()，

否则图片无法正常显示。

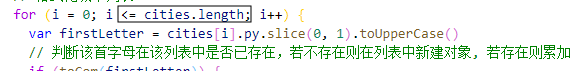




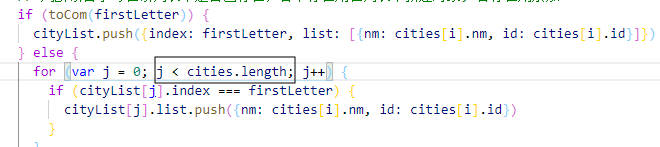
1. 当Error in mounted hook: "TypeError: Cannot read property 'xxx' of undefined"

这种报错的时候，查看xxx是否处在循环中，如果在循环中，检查循环或数组取值的index等是否超出范围。

例1：会报py of undefined,因为到最后一位时，没有py



例2：会报index of undefined，因为cities的长度比cityList长，所以后面并不能得到index



1. 设置position（absolute,fixed）导致flex布局不生效；

解决办法：将要设置display：flex的dom外面在套一个div，并且设置宽度，就可以了。

1. vw(viewport), vh, vmax, vmin 兼容性不错

（1）**%** 是相对于父元素的大小设定的比率，**vw**、**vh**是视窗大小决定的。

（2）**vw**、**vh**优势在于能够直接获取高度，而用 **%**在没有设置 **body**高度的情况下，是无法正确获得可视区域的高度的，所以这是挺不错的优势。

用途：

(1)做移动页面开发时，如果使用 **vw**、**wh**设置字体大小（比如 **5vw**），在竖屏和横屏状态下显示的字体大小是不一样的。

(2)由于 **vmin**和 **vmax**是当前较小的 **vw**和 **vh**和当前较大的 **vw**和 **vh**。这里就可以用到 **vmin**和 **vmax**。使得文字大小在横竖屏下保持一致。

1. vue组件中v-for不能用在跟元素上

Cannot use v-for on stateful component root element because it renders multiple elements

1. 文本超出文本框限制自动隐藏，并在末尾添加…；鼠标hover到省略部分显示全部

p:hover

{

  text-overflow:inherit;

  overflow:visible;

}

p{

  overflow: hidden;

  white-space: nowrap; // 不加会自动换行

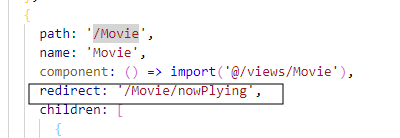
  text-overflow: ellipsis; // 隐藏超出部分用...结尾。clip则不已省略号结尾

......

}

css3属性：clip为默认值

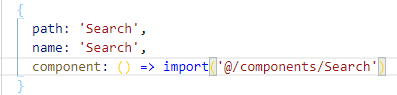
1. 设置默认路由redirect: ‘’



1. Conflict: Multiple assets emit to the same filename

当使用子组件时：并将组件名改为search，时报该错，改成大写后报错消失；

该报错和path和name无影响



网上解释：这是因为你有2个条目（app.js，index.html）输出到同一个文件appN.js. 尝试在输出的文件名中添加[name]。

1. 移动端的tap事件 zepto vue-touch **better-scroll**

点击触发，滑动不触发

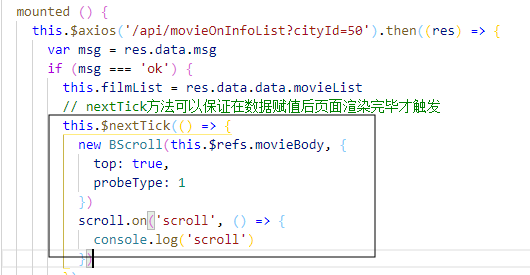
1. 滑动效果iscroll & **better-scroll**(iscroll的封装)

可以很方便的实现下拉刷新，锚点滚动等功能

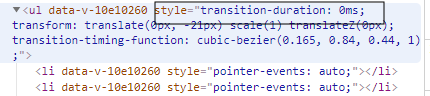
better-scroll原理：

一是内容大于最外层包裹的容器，出现溢出才会有滚动效果；二是内容加载完毕。

dom中：<div ref="movieBody">

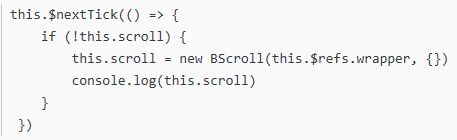


关于滑动效果不起作用问题：大概率子元素范围小于**等于**父元素

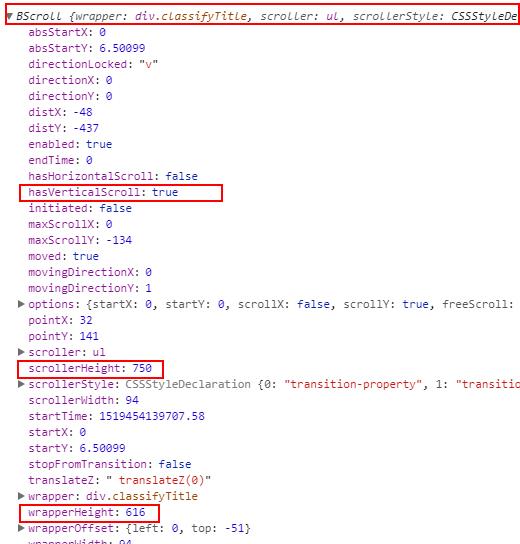


查看是否已经添加到dom中

测试：



查看



查看wrapperHeight是否小于scrollHeight，

hasVerticalScroll是否为true；

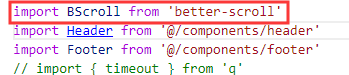
只有上面成立，才会出现滚动效果。

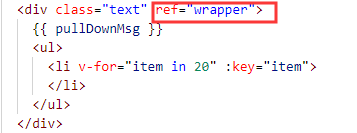
成功例子：让含有ref属性的模块使用绝对定位，获取固定的长度，

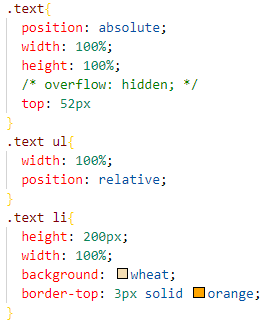
让其子元素比如(ul)进行100%继承长度，从而大于父元素的长度形成滚动。

因为要内容大于父元素时才会滚动，所以ul包裹的li需要固定高度将ul撑开，ul需要使用相对定位不设置高度，让li累积的高度成为他的高度；而ul的父元素设置了ref的div模块需要使用绝对定位，因为只有绝对定位之后才能让ul有超过的可能，可设置高度100%，通过f12查看该div高度是否为0或者和ul相等，如果是，还需要调整div的父元素高度。

使用better-scroll



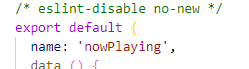




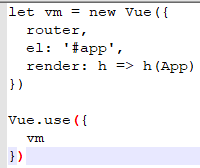


1. 在 new Vue 在ESLint报错Do not use 'new' for side effects

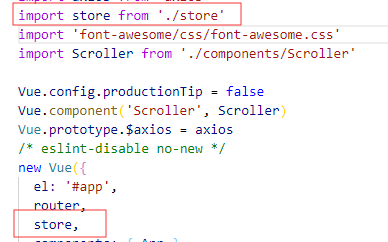
方法1：在new Vue上加/\* eslint-disable no-new \*/绕过检测



方法2：



1. vuex的运用



import vuex from 'vuex'

import Vue from 'vue'

Vue.use(vuex)

export default new vuex.Store({

  state: {

    citynm: '北京',

    cityid: 1

  },

  mutations: {

    getCitynm: (state, newCity) => {

      state.citynm = newCity.cityNm

      state.cityid = newCity.cityId

    }

  }

})