一、

1.api文件夹：存放各页面组件利用jsonp跨域获取接口的jsonp函数包装

2.base文件夹：存放各个基础组件

二、

1.recommend页面组件：

（1）引入recommend.js，用于获取轮播图数据和下面的列表数据

（1）getRecommend：获取轮播数据，将返回值的slider复制给data的recommends

（2）getDiscList：获取推荐列表，将返回值的list复制给data的discList

（2）引入slider.vue组件，实现轮播图和下面的原点映射

slider组件：

* slider组件是实现轮播的组件
* 利用slot插槽，有父组件传入dom结构
* 利用：class绑定，设置轮播点dots的active状态
* 它的props接收两个属性loop：是否循环播放 autoplay：是否自动播放
* 由于轮播的 容器会根据浏览器大小设定，所以需要动态设置，在nethods里定义一个设置宽的函数，主要功能是：

（1）获取slider容器的可见宽，获取所有滚动li，遍历li的宽度为可见宽

（2）将每个li的宽度累加再\*2所得值复制给slider容器的ul的宽

* 在methods定义初始化slider的函数：

（1）importbetter-scroll，并定义一个BScroll的实例对象this.slider,其中snap参数是slider专用参数

（2）监听滚动结束scrollEnd事件：利用BScroll getCurrentPage事件获取当前滚动页，在循环模式下，将当前滚动页-1，将所得值设置为当前滚动页得索引，在自动播放模式下，执行播放函数

* 在methods定义自动播放的函数：

（1）获取初始化slider函数中的当前页索引值，并+1

（2）在循环循环模式下，索引要不断+1（用if条件判断是否loop）

（3）设置定时器，利用BScroll的goToPage函数，让li根据索引值轮播

* 在methods定义初始化dots圆点的函数：

（1）定义一个数组对象的实例this.dots，长度为li的个数

（3）在li上添加点击事件，进入详情页：

* 将当前li项作为参数传给点击函数，利用vue的this.$router.push（），将路由指向详情页路径/recommend/item.id
* 引入vuex的 mapMutations，在store中有SET\_DISC的mutation函数，用于渲染详情页的头部标题和背景图

2.singer歌手页面：

（1）引入基础组建list-view：

* 在计算属性中返回旁边的字母索引和每一块的字母标题
* 在props设置需要外部传入的data
* 由于list-view是基础组件，所以没有逻辑处理的代码，所以点击歌手li时，有一个seletcItem点击事件，只是向外派发一个select事件
* 在methods中定义onShortcutTouchStart函数，用于获取开始触点的位置，及所在索引，将索引传递给\_scrollTo函数
* 在methods中定义onShortcutTouchMovet函数，用于获取触点移到的位置和所在的字母块，及所在索引，将索引传递给\_scrollTo函数
* 在methods中定义 \_calculateHetght函数，用于设置歌手li的宽度
* 在methods中定义\_scrollTo函数，用于是列表滚动指定的索引位置
* 在watch里面监听歌手li的高度，设置计时器，实时更新高度，以防浏览器改变印象布局
* 在watch里监听scrollY，当滚动距离在一个li中间高度的时候，将歌手索引设置为前一个歌手的索引，并设置diff（表示当前歌手li出去滚动的高度外，还是多少高度）
* 在watch里面监听diff的变化，并利用tansform设置字母表题在凉快快要重合时的滑动效果

（2）在methods中定义list-view抛出的点击事件select的函数selectSinger：利用vue的this.$router.push，将路由指向歌手详情的链接，并利用引入vuex的mapMutations的SET\_SINGER方法渲染详情页面

（3）利用api文件夹中singer.js封装的jsonp获取歌手列表，将数据返回给data，并传给list-view组件

（4）在methods中定义\_normalizeSinger函数用于处理获取到的打他数据：数据处理分为热门数据和按字母分类的数据，将热门歌手放进hot数组，一般歌手放进ret数组，在利用sort（）将ret进行排序

3.rank排行页面

（1）引入api的rank.js封装的jsonp获取数据，并返回给data函数

（2）引入Scroll基础组件，将返回的数据传入，使排行列表滑动

（3）在li上绑定selectItem点击事件，跳转到详情页，此时同样要利用this.$router.push和maoMutations，实现跳转和渲染

4.search搜索页面：

（1）页面的搜索栏是引入searchBox组件：

searchBox组件：

* 定义一个搜索框，给定placeholder默认值
* v-model双向绑定搜索框的内容
* 在清除符号上绑定清空搜索框内容的点击事件
* 在methods中定义blur方法，是搜索框失去焦点
* 引入common/js/util中的debounce，监听搜索框的内容变化，并实时派发内容改变的函数

（2）引入api的search.js封装的jsonp，获取热门推荐的数据

（3）引入suggest组件，显示搜索结果列表：

suggest组件：

* 引入Scroll基础组件，并传入data，pullup，beforeScroll等属性值
* 引入common/js/song.js的createSong函数，并在methods中定义一个\_normalizeSongs函数将传入的歌曲数据实例化为一个song类
* 在methods中定义一个\_genResult函数，运用vue的扩展符和\_normalizeSongs方法将歌曲数据规范为歌手：xxx；专辑：xxx的类似形式
* 在methods中定义一个search函数，引入api/search.js封装好的jsonp获取数据，并调用\_genResult函数对数据进行规范，得到想要的形式
* 由于suggest是基础组件，所以在搜索列表的li上绑定click事件selectItem，向外派发select时间，如果搜索出歌手，就路由到歌手详情页，并引入vuex的mapMutations中的SET\_SINGER渲染页面，如果搜索出歌曲，则引入vuex的mapActions的insertSong，渲染歌曲详情页面
* 在搜索结果列表中得i标签绑定class的判断函数getIconCls，判断歌手还是歌曲，来选择渲染的图标
* 在搜索结果的name上绑定getDisplayName函数，并用v-html转义，如果是歌手，就返回歌手名，如果是歌曲就返回歌曲名+歌手名
* 在methods里定义\_checkMore函数，用于检查是否还有更多数据，在song.curnum + song.curpage \* perpage（perpage是每页长度）大于song.totalnum（totalnum是总的数据数）时，将定义的hasMore设置为false
* 在scroll组件上绑定searchMore函数，用于加载下一个搜索结果，在函数中使page++，并调用search方法，获取下一页数据，并调用\_checkMore方法检查是否还有更多数据
* 在watch中实时监听搜索列表诗句query的变化，即调用search方法获取数据

（4）在推荐的li上绑定一个addQuery事件，该函数调用searchBox派发的setQuery事件，将点击的li内容更新到搜索框中

（5）搜索历史块，显示条件是searchHistory.length

（6）同样在搜索历史的li上添加addQuery事件，将内容更新到搜索框

（7）给搜索历史的清空垃圾桶的标志添加一个显示确认清空弹框的点击事件

（8）引入search-list基础组件来显示搜索历史列表

search-list组件：

* 将ul标签改为<transition-group name="list" tag="ul">，以保证在删除列表项时有平稳的滑动，此时li要绑定kry值，即：key=”“
* 在li上绑定selectItem函数，以此向外派发select事件
* 在删除符号的icon上绑定deleteOne事件，以此向外派发delete事件
* 设置props的searches属性，有父组件传入

（9）定义search-list派发的select事件安慰addQuery，delete事件为deleteSearchHistory

（10）引入base/confirm/confirm组件，设置确认清空历史记录的弹框

comfirm组件：

* 设置showFalg的boolean值（初始值为false）来控制弹框是否显示
* 当调用show函数时，将showFalg设为true，调用hide函数时，设为false
* 在清空按钮上绑定confirm事件按，向外派发确认事件，在取消按钮上绑定cancel事件，向外派发取消事件

（11）向comfirm组件传入confirm事件的方法clearSearchHistory

（12）在watch中监听query的变化，设置定时器实时更新