**MobikeSearch**

1. **小程序简介**

MobikeSearch是一款简单便捷的查询类小程序，主要是为用户提供地图，查询附近共享单车和查询最近共享单车的功能，利用微信小程序的map组件以及代理摩拜接口实现。

1. **小程序界面和功能介绍**

**界面一：”/pages/about/about”**



图 1 关于页面

小程序的入口页，主要是对MobikeSearch的一个简单介绍和制作目的。在样式是采用开源组件库ColorUI。主要特色之一，是对导航栏自定义，本页面导航栏使用component实现，利用ColorUI提供导航栏组件实现。基本小功能之一，点击Welcome按钮，触发toIndexPage事件，在事件函数中调用wx.redirectTo，跳转到首页。

**自定义导航栏核心代码：**

前提：引入ColorUI组件库

App.js 获得系统信息

|  |
| --- |
| onLaunch: function() {  wx.getSystemInfo({  success: e => {  this.globalData.StatusBar = e.statusBarHeight;  let custom = wx.getMenuButtonBoundingClientRect();  this.globalData.Custom = custom;  this.globalData.CustomBar = custom.bottom + custom.top - e.statusBarHeight;  }  })  }, |

App.json 配置取消系统导航栏,并全局引入组件

|  |
| --- |
| "window": {  "navigationStyle": "custom"  },  "usingComponents": {  "cu-custom":"/colorui/components/cu-custom"  } |

about.wxml 页面调用组件

|  |
| --- |
| <cu-custom bgColor="bg-gradual-green" isBack="{{false}}">  <view slot="content">关于</view>  </cu-custom> |

**跳转首页核心代码：**

为按钮提供bindtap事件toIndexPage

|  |
| --- |
| <button bindtap="toIndexPage" class="cu-btn bg-cyan shadow mybtn" data-class="shake" role="button" aria-disabled="false">Welcome</button> |

在about.js中，定义事件函数，并实现重定向

|  |
| --- |
| toIndexPage:function(res){  wx.redirectTo({  url: '/pages/index/index',  })  }, |

**界面二：”/pages/index/index”**



图 2 首页



图 3 查询附近共享单车



图 4 查询离你最近共享单车

小程序的首页，也是该小程序最核心的页面，主要是用于提供用户查询附近共享单车和查询最近共享单车的查询功能，同时提供地图显示，地图放大和地图缩小的基本功能。地图使用微信小程序原生map组件，通过小程序的map组件基本属性和数据绑定，实现了地图缩放，当前位置定位，标记点显示和路径导航显示功能。点击查询附近共享单车，绑定按钮bindtap事件，在index.js中定义事件函数，并在其中发起网络请求，用小程序提供的wx.request，向后台服务器发起请求，由后台服务器处理请求，并响应数据给小程序，小程序再展示到map中即可。点击查询最近共享单车，事件绑定和定义与查询附近共享单车同理，在事件函数中，通过遍历共享单车集合，找到最近的共享单车坐标，通过map标记在地图上并结合当前位置坐标，调用腾讯地图WebService API，实现路径导航显示。

**细节：**

缩放按钮，通过条件渲染wx:if，使其达到缩放极限时，不可使用。在未查询附近共享单车时，查询最近共享单车按钮，不可使用。

如下图所示

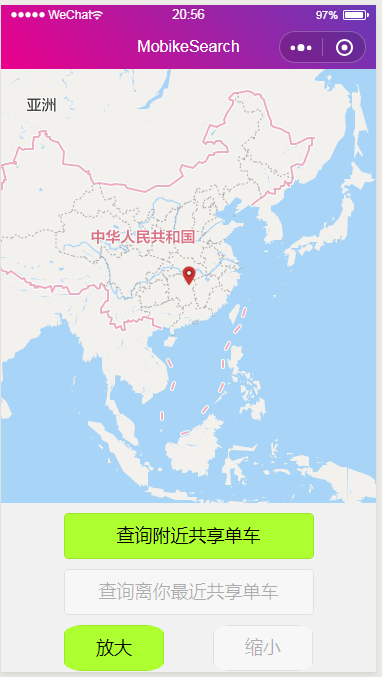
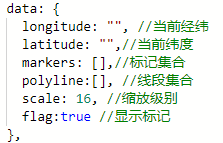
 

图 5 放大极限 图 6 缩小极限



图 7 禁用查询最近共享单车按钮

**Index.js data**



**当前位置显示核心代码：**

在index.js的onload生命周期函数中，获取当前位置，并标记

|  |
| --- |
| /\*\*  \* 生命周期函数--监听页面加载  \*/  onLoad: function (options) {  var that = this  //获得当前位置  wx.getLocation({  success: function (res) {  //console.log(res)  //初始化定位  var markers = [{  longitude: res.longitude,  latitude: res.latitude,  iconPath: "/images/location.jpg",  width: 20,  height: 20  }]  that.setData({  longitude: res.longitude,  latitude: res.latitude,  markers: markers  })  },  })  }, |

**缩放核心代码：**

为按钮提供bindtap事件zoomIn和zoomOut

|  |
| --- |
| <button class='btn zoomIn' bindtap='zoomIn' disabled='{{scale>=19}}'>放大</button>  <button class='btn zoomOut' bindtap='zoomOut' disabled='{{scale<=4}}' >缩小</button> |

在index.js中，改变scale的值

|  |
| --- |
| /\*\*  \* 放大  \*/  zoomIn: function (res) {  var scale = this.data.scale  scale = scale + 1  this.setData({  scale: scale  })  },  /\*\*  \* 缩小  \*/  zoomOut: function (res) {  var scale = this.data.scale  scale = scale - 1  this.setData({  scale: scale  })  }, |

**查询附近共享单车核心代码：**

为按钮提供bindtap事件searchBike

|  |
| --- |
| <button class='btn' bindtap='searchBike'>查询附近共享单车</button> |

在index.js中，向后台发起请求，并处理相应数据

|  |
| --- |
| /\*\*  \* 搜索附近共享单车  \*/  searchBike: function (res) {  var that = this  //console.log(res)  wx.request({  url: 'http://localhost:9090/getNearByBike',  data: {  longitude: that.data.longitude,  latitude: that.data.latitude,  cityCode: "021"  },  success: function (res) {  console.log(res)  //确认获取附近单车  if (res.data.code == 0) {  // that.data.markers = res.data.object.map(function(item){  // item.id = item.bikeIds  // item.latitude = item.distY  // item.longitude = item.distX  // item.iconPath = "/images/location.jpg"  // return item  // })  var markers =[]  //设置当前位置  markers[0] = {  id:"current",  latitude:that.data.latitude,  longitude:that.data.longitude,  iconPath: "/images/location.jpg",  width: 20,  height: 20  }  //设置共享单车位置  for (var i = 0; i < res.data.object.length; i++) {  var marker = {  id: "",  longitude: "",  latitude: "",  iconPath: "",  width: 20,  height: 20  }  marker.id = res.data.object[i].bikeIds  marker.latitude = res.data.object[i].distY  marker.longitude = res.data.object[i].distX  marker.distance = res.data.object[i].distance  marker.iconPath = "/images/danche.png"  markers[i+1] = marker  }  that.setData({  markers: markers,  flag:false  })  }  else {  console.log("调用失败")  }  }  })  }, |

**查询离你最近共享单车核心代码：**

为按钮提供bindtap事件searchNearBy

|  |
| --- |
| <button class='btn' disabled='{{flag}}' bindtap='searchNearBy'>查询离你最近共享单车</button> |

在index.js中，获取最近共享单车坐标，调用腾讯地图WebService API

|  |
| --- |
| /\*\*  \* 查询附近最近的共享单车  \*/  searchNearBy:function(res){  var markers = this.data.markers  if (markers.length>1){  //最小距离与索引  var minDistance = markers[1].distance  var index = 1;  for(var i = 0;i<markers.length;i++){  if(i==0){  continue  }  //进行判断  if(minDistance>markers[i].distance){  minDistance = markers[i].distance  index = i  }  }  //console.log(minDistance)  //设置气泡  var callout = {  content:"距离最近的单车",  fontSize:10,  borderRadius:20,  color:"#4169E1",  bgColor:"#D3D3D3",  display:"ALWAYS"  }  markers[index].callout = callout  wx.showLoading({  title: '加载中...',  })  var that = this  //路线规划  wx.request({  url: 'https://apis.map.qq.com/ws/direction/v1/walking/',  data:{  from:that.data.latitude+","+that.data.longitude,  to:markers[index].latitude+","+markers[index].longitude,  key:"4LABZ-AMZOF-MPQJK-NJMU4-LOY66-ASB27"  },  success:function(res){  //console.log(res)  //解压坐标，规划路线  var polyline = [{  points: [],  color: "#FF0000DD",  width: 2,  dottedLine: true  }  ]  //设置起始点  polyline[0].points.push({  longitude: that.data.longitude,  latitude: that.data.latitude  })  var coors = res.data.result.routes[0].polyline  for (var i = 2; i < coors.length; i++) {  coors[i] = coors[i - 2] + coors[i] / 1000000  }  //console.log(coors)  for(var i = 0;i<(coors.length+1)/2;i=i+2){  var point = {}  point.longitude = coors[i + 1]  point.latitude = coors[i]  polyline[0].points.push(point)  }  //设置终点  polyline[0].points.push({  longitude: markers[index].longitude,  latitude: markers[index].latitude  })  //console.log(polyline)  that.setData({  markers: markers,  polyline: polyline,  scale:19  })  },  fail:function(err){  console.log(err)  },  complete:function(e){  wx.hideLoading()  }  })  }  }, |

1. **小程序后台简介**

小程序后端使用SpringBoot框架，以Maven构建工程，由其集成内嵌Web服务器，自动装配，起步依赖等特性，实现一个基本的web工程，提供基本Web服务，同时利用HTTPClient发起内部网络请求。其中部分尝试地利用HttpClient+Jsoup，爬取西刺代理和米扑代理提供的代理ip和端口，但最终以失败告终，原因在不足就将说明。

在处理摩拜接口方面，根据百度，谷歌等信息资料，得知摩拜本身不提供官方API接口服务，而是通过抓包方式，将摩拜单车API抓取，对其入口参数和出口参数进行一定分析，通过本地服务器代理，作为小程序和摩拜接口的中转站，在接收到小程序的网络请求后，通过后端对摩拜接口发起网络请求，接收其摩拜单车数据，然后响应给小程序。

同时，摩拜接口对ip有一定限制，但可以通过代理池方式避免，通过获取大量代理ip和端口，经过校验其有效性，再通过代理ip和端口发起网络请求，伪装本地ip，来规避其对ip限制。代理池需要注意几点，第一是ip和端口的有效性检验，第二是定时获取最新ip和端口，并清洗过期的ip和端口。

**摩拜查询附近共享单车接口：**

https://mwx.mobike.com/mobike-api/rent/nearbyBikesInfo.do

入口参数：

Longitude，latitude，citycode，errMsg

出口参数：

bikeIds，biketype，boundary，distId，distNum，distX，distY，distance，operateType，type

**西刺代理：**

<https://www.xicidaili.com/>

**米扑代理：**

<https://proxy.mimvp.com/freesecret.php>

【后端工程已打包成jar包，在cmd中运行 java -jar xxx.jar即可启动】

1. **不足**
2. 小程序方面，内容和形式过于简单，不够丰富，有点显得太简陋了。
3. 小程序方面，功能过于单一，需要不断增加其他功能。
4. 后端方面，由于摩拜接口对ip访问进行限制，并不是每一次请求，都会成功响应，需要通过代理池方式，避免反爬虫。
5. 后端方面，代理池实现过程中，存在许多问题。首先通过了解，可以通过第三方提供代理ip和端口，避免接口对ip访问的限制，于是考虑通过爬取西刺代理和米扑代理中提供的ip，来创建自己的代理池。在西刺代理爬虫运行中，成功获取前5页的500个代理ip和端口，经过ip和端口有效性校验，最终大概还有50个代理ip和端口可使用，但由于爬取次数太多，西刺代理将本地ip封锁，不能再爬取到其可用ip和端口，十分可惜。米扑代理，爬取过程也类似，但由于其端口特地以图片形式放置，导致端口爬取难度加大，最终也失败。