## **flutter与原生交互原理**

Flutter与Native之间的通信是采用双通道事件传递机制，通过MethodChannel进行传递数据，通信原理如下图（官网图片）



这里贴上官网的[文档地址](https://flutter.dev/docs/development/platform-integration/platform-channels%22%20%5Ct%20%22https%3A//segmentfault.com/a/_blank)，方便大家查阅，官网的获取充电的信息的例子网上也有很多，我就不带大家一起敲了，这里主要讲解一下与国内Badidu地图实战的过程方便大家加深理解

## **需求背景**

* 需求：需要在Flutter里面展示地图，并且能够进行点击，导航，定位等交互。
* 需求分析（实现方法）：
1. 采用Flutter组件库里面的google地图进行开发（能够快速开发，而且有成熟的api，但是由于在国内需要翻墙才能用google地图，而且基本是英文，所以弃用）
2. 使用webview嵌入Flutter程序进行开发（能够灵活的实现各种样式，但是每次都需要加载地图比较缓慢，暂不采用）
3. 使用原生的baidu地图与Flutter进行通信（增加了开发的工作量，但是能够很好的克服上面的缺点）

## **Flutter端代码**

首先Flutter程序里面先展示原生的Android Activity作为组件进行展示，然后创建MethodChannel和EventChannel对象进行事件进行传递，EventChannel用来监听事件，MethodChannel用来发送事件，原理很简单，大家不要想得太复杂，动手实现一下就能够明白

**import** 'package:flutter/material.dart';**import** 'package:flutter/services.dart';**import** 'package:my\_project/widget/map\_mood\_card.dart';**import** 'mood\_detailed\_content.dart';**import** 'package:my\_project/common/utils/navigator\_utils.dart';**import** 'package:my\_project/widget/shuofen\_card.dart';**import** 'package:my\_project/widget/xica\_card.dart';

**class** **MapPage** **extends** **StatefulWidget** {

 **const** MapPage({Key key}) : **super**(key: key);

 **@override**

 \_MapPageState createState() => \_MapPageState();

}

**class** **\_MapPageState** **extends** **State**<**MapPage**> **with** **AutomaticKeepAliveClientMixin** {

 **static** **const** platform = **const** MethodChannel('samples.flutter.io/getLocation');

 *// 需要跟MainActivity中的一致（com.example.my\_project/event）*

 **static** **const** EventChannel eventChannel = **const** EventChannel('com.example.my\_project/event');

 bool isShowCard = **false**;

 string eventString = '';

 **@override**

 **void** initState() {

 print('-------------initState--------------');

 **super**.initState();

 eventChannel.receiveBroadcastStream().listen(\_onEvent, onError: \_onError);

 }

 **void** \_onEvent(Object event) {

 **this**.setState(() {

 eventString = event;

 });

 print('-------------Message from native------------------' + event.toString());

 }

 **void** \_onError(Object error) {

 setState(() {

 print(

 '-------------Error occured on communicate between flutter and native------------------');

 });

 }

 **@override**

 **void** didChangeAppLifecycleState(AppLifecycleState state) {

 print('-------------didChangeAppLifecycleState-------------$state-');

 }

 **@override**

 **void** dispose() {

 print('----------dispose---------------');

 **super**.dispose();

 }

 **void** updateMapMarker() **async** {

 **await** platform.invokeMethod('refrashMap', "我是参数");

 }

 *// This widget is the root of your application.*

 **@override**

 Widget build(BuildContext context) {

 print('-------------build--------------');

 updateMapMarker();

 **return** Scaffold(

 body: Stack(

 children: <Widget>[

 Center(

 child: AndroidView(viewType: 'MyMap'),

 ),

 ],

 ));

 }

 **@override**

 bool **get** wantKeepAlive => **true**;

}

## **Android端代码**

怎么配置baidu地图的sdk这里就不细说了，baidu地图的文档给开发者提供了相当详细的说明，在代码里面主要看中MethodChannel和EventChannel的功能，MethodChannel可以通过MethodCall监听Flutter程序事件发来的事件的数据，然后可以通过EventChannel进行发送数据回去，两者结合再一起则是双通道的通信机制。这里发送的数据格式有要求，格式如下（官网图片）



MainActivity.java

com.example.my\_project

**import android.annotation.SuppressLint;import android.content.SharedPreferences;import android.graphics.Point;import android.os.Bundle;import android.os.Handler;import android.os.Message;import android.util.DisplayMetrics;import android.util.Log;import android.view.Gravity;import android.view.WindowManager;import android.widget.Toast;**

**import com.baidu.location.BDAbstractLocationListener;import com.baidu.location.BDLocation;import com.baidu.location.LocationClient;import com.baidu.location.LocationClientOption;import com.baidu.mapapi.map.BaiduMap;import com.baidu.mapapi.map.BitmapDescriptor;import com.baidu.mapapi.map.BitmapDescriptorFactory;import com.baidu.mapapi.map.MapStatus;import com.baidu.mapapi.map.MapStatusUpdate;import com.baidu.mapapi.map.MapStatusUpdateFactory;import com.baidu.mapapi.map.MapView;import com.baidu.mapapi.map.MyLocationConfiguration;import com.baidu.mapapi.map.MyLocationData;import com.baidu.mapapi.map.UiSettings;import com.baidu.mapapi.model.LatLng;**

**import org.jetbrains.annotations.NotNull;**

**import java.util.ArrayList;**

**import io.flutter.app.FlutterActivity;import io.flutter.plugin.common.EventChannel;import io.flutter.plugin.common.MethodCall;import io.flutter.plugin.common.MethodChannel;import io.flutter.plugin.common.MethodChannel.MethodCallHandler;import io.flutter.plugin.common.MethodChannel.Result;import io.flutter.plugins.GeneratedPluginRegistrant;**

**public** **class** **MainActivity** **extends** **FlutterActivity** {

 **private** **final** String eventString = "event";

 **private** **static** **final** String TAG = "MainActivity";

 **private** **static** **final** String CHANNEL = "samples.flutter.io/getLocation";

 **private** **static** **final** String eventChannel = "com.example.my\_project/event";

 **private** MethodChannel channel;

 *// 后台服务器地址*

 String host = null;

 String locationText;

 Double longitude, latitude;

 **static** User user = **new** User("", "");

 **private** **static** MapView mapView;

 **private** **static** BaiduMap mBaiduMap;

 **private** LocationClient mLocationClient;

 **private** **static** String myId = null;

 @Override

 **protected** **void** onCreate(Bundle savedInstanceState) {

 **super**.onCreate(savedInstanceState);

 GeneratedPluginRegistrant.registerWith(**this**);

 mapView = **new** MapView(**this**);

 MapRegistrant.registerWith(**this**, mapView);

 *//定位初始化*

 mLocationClient = **new** LocationClient(**this**);

 *//注册LocationListener监听器*

 MyLocationListener myLocationListener = **new** MyLocationListener();

 mLocationClient.registerLocationListener(myLocationListener);

 *// 不显示百度地图Logo*

 mapView.removeViewAt(1);

 mBaiduMap = mapView.getMap();

 *// 改变地图状态，使地图显示在恰当的缩放大小*

 mMapStatus = **new** MapStatus.Builder().zoom(18.0f).build();

 MapStatusUpdate mMapStatusUpdate = MapStatusUpdateFactory.newMapStatus(mMapStatus);

 mBaiduMap.setMapStatus(mMapStatusUpdate);

 mBaiduMap.setMyLocationEnabled(true);

 mBaiduMap.setMyLocationConfiguration(**new** MyLocationConfiguration(

 MyLocationConfiguration.LocationMode.FOLLOWING, true, null));

 *//实例化UiSettings类对象*

 UiSettings mUiSettings = mBaiduMap.getUiSettings();

 *//禁用地图旋转功能，启用后对显示屏幕范围内的Marker有一定影响*

 mUiSettings.setRotateGesturesEnabled(false);

 *//禁用地图俯视功能*

 mUiSettings.setOverlookingGesturesEnabled(false);

 **new** MethodChannel(getFlutterView(), CHANNEL).setMethodCallHandler(

 **new** MethodCallHandler() {

 @Override

 **public** **void** onMethodCall(MethodCall call, Result result) {

 *// 在这个回调里处理从Flutter来的调用*

 **switch** (call.method) {

 **case** "getLocationDics":

 **if** (locationText == "") {

 result.success("地球的某一个角落");

 **break**;

 }

 result.success(locationText);

 **break**;

 **case** "getLocationLongitude":

 result.success(longitude);

 **break**;

 **case** "getLocationLatitue":

 result.success(latitude);

 **break**;

 **case** "refrashMap":

 **if** (user.getUserId() != "") {

 mapView.onResume();

 **if**(!isFirstMapRender) {

 UpdateMapState();

 }

 Log.d("tag", call.arguments.toString());

 result.success(null);

 }

 **break**;

 **case** "setUserId":

 mapView.onResume();

 user.setUserId(call.arguments.toString());

 result.success(null);

 **break**;

 **case** "setUserToken":

 user.setToken(call.arguments.toString());

 result.success(null);

 **break**;

 **case** "openGps":

 addEmojiMarkers();

 addUserTagBitMaps();

 addUserTagBitMaps\_Personal();

 requestLocation();

 result.success(null);

 **break**;

 }

 }

 }

 );

 *//设置地图渲染完成回调*

 mBaiduMap.setOnMapRenderCallbadk(renderCallback);

 *//设置地图状态监听*

 mBaiduMap.setOnMapStatusChangeListener(listener);

 }

 **new** EventChannel(getFlutterView(), eventChannel).setStreamHandler(

 **new** EventChannel.StreamHandler() {

 @Override

 **public** **void** onListen(Object args, **final** EventChannel.EventSink events) {

 Log.d(TAG, "adding listener");

 mBaiduMap.setOnMarkerClickListener(**new** BaiduMap.OnMarkerClickListener() {

 events.success(eventString);*// 发送事件(eventString);*

 }

 **return** true;

 }

 });

 }

 @Override

 **public** **void** onCancel(Object args) {

 Log.d(TAG, "cancelling listener");

 }

 }

 );

 BaiduMap.OnMapRenderCallback renderCallback = **new** BaiduMap.OnMapRenderCallback() {

 */\*\**

 *\* 地图渲染完成回调函数*

 *\*/*

 @Override

 **public** **void** onMapRenderFinished() {

 **if**(isFirstMapRender) {

 Log.d("OnMapRenderCallback","地图首次渲染完成回调函数");

 isFirstMapRender = false;

 UpdateMapState();

 } **else** **if**(isSecondMapRender) {

 Log.d("OnMapRenderCallback","地图二次渲染完成回调函数");

 isSecondMapRender = false;

 UpdateMapState();

 }

 }

 };

 BaiduMap.OnMapStatusChangeListener listener = **new** BaiduMap.OnMapStatusChangeListener() {

 */\*\**

 *\* 手势操作地图，设置地图状态等操作导致地图状态开始改变。*

 *\**

 *\* @param status 地图状态改变开始时的地图状态*

 *\*/*

 @Override

 **public** **void** onMapStatusChangeStart(MapStatus status) {

 }

 */\*\**

 *\* 手势操作地图，设置地图状态等操作导致地图状态开始改变。*

 *\**

 *\* @param status 地图状态改变开始时的地图状态*

 *\**

 *\* @param reason 地图状态改变的原因*

 *\*/*

 *//用户手势触发导致的地图状态改变,比如双击、拖拽、滑动底图*

 *//int REASON\_GESTURE = 1;*

 *//SDK导致的地图状态改变, 比如点击缩放控件、指南针图标*

 *//int REASON\_API\_ANIMATION = 2;*

 *//开发者调用,导致的地图状态改变*

 *//int REASON\_DEVELOPER\_ANIMATION = 3;*

 @Override

 **public** **void** onMapStatusChangeStart(MapStatus status, int reason) {

 }

 **private** **void** requestLocation() {

 *//通过LocationClientOption设置LocationClient相关参数*

 LocationClientOption option = **new** LocationClientOption();

 option.setOpenGps(true); *// 打开gps*

 option.setCoorType("bd09ll"); *//坐标类型*

 option.setIsNeedLocationDescribe(true);

 *//设置locationClientOption*

 mLocationClient.setLocOption(option);

 *//开启地图定位图层*

 mLocationClient.start();

 }

 **public** **class** **MyLocationListener** **extends** **BDAbstractLocationListener** {

 @Override

 **public** **void** onReceiveLocation(BDLocation location) {

 *//mapView 销毁后不在处理新接收的位置*

 **if** (location == null || mapView == null){

 **return**;

 }

 locationText = location.getLocationDescribe();

 longitude = location.getLongitude();

 latitude = location.getLatitude();

 MyLocationData locData = **new** MyLocationData.Builder()

 .accuracy(0)

 *// 此处设置开发者获取到的方向信息，顺时针0-360*

 .direction(location.getDirection()).latitude(location.getLatitude())

 .longitude(location.getLongitude()).build();

 mBaiduMap.setMyLocationData(locData);

 }

 }

}