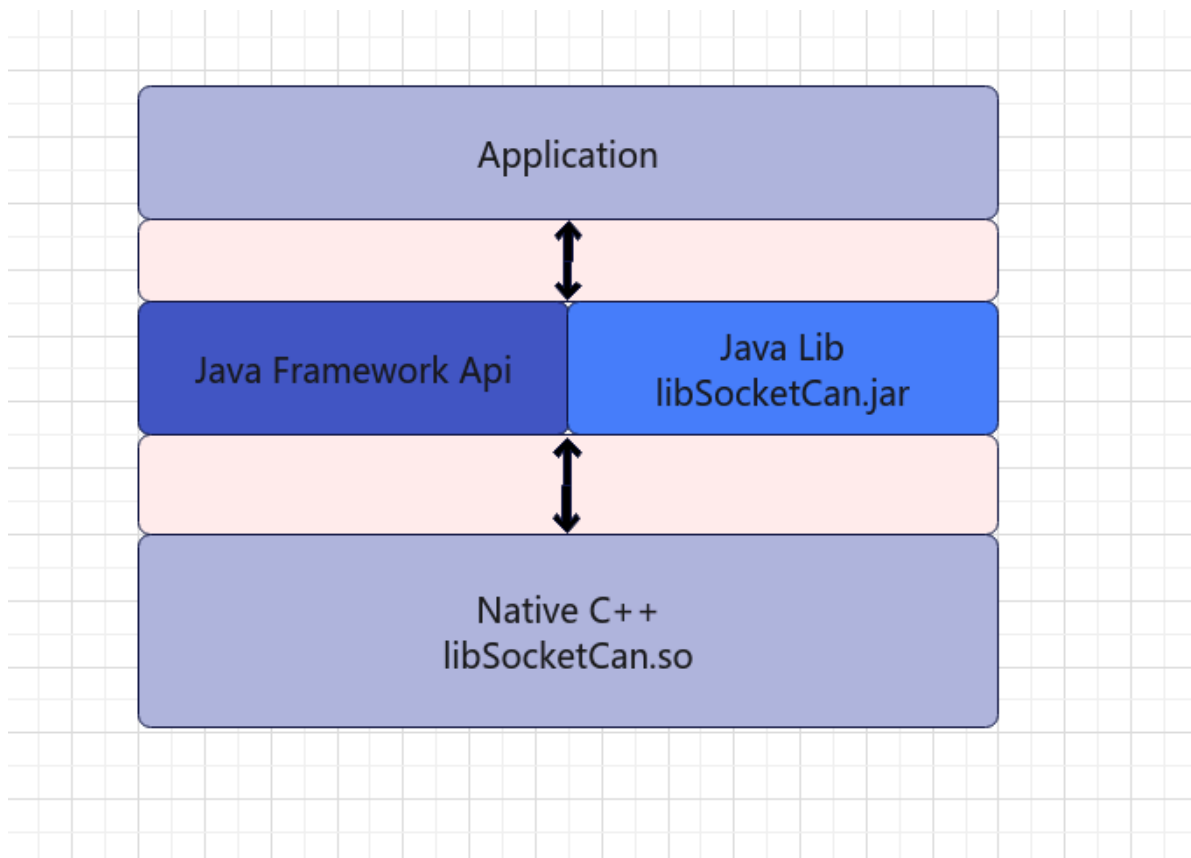


# 介绍

## libSocketCan SDK 概述

libSocketCan SDK 是一组控制 imx8m tpc\_1xx 安卓设备 CAN 总线的 API 开发库，包括(配置、打开、关闭、发送和接收 CAN 数据帧、设置滤波)，SocketCan 仅支持 can0 接口的控制

libSocketCan SDK 为应用程序提供 API 模块，用于控制 CAN 进行各种操作，该 SDK 的软件栈如下图所示



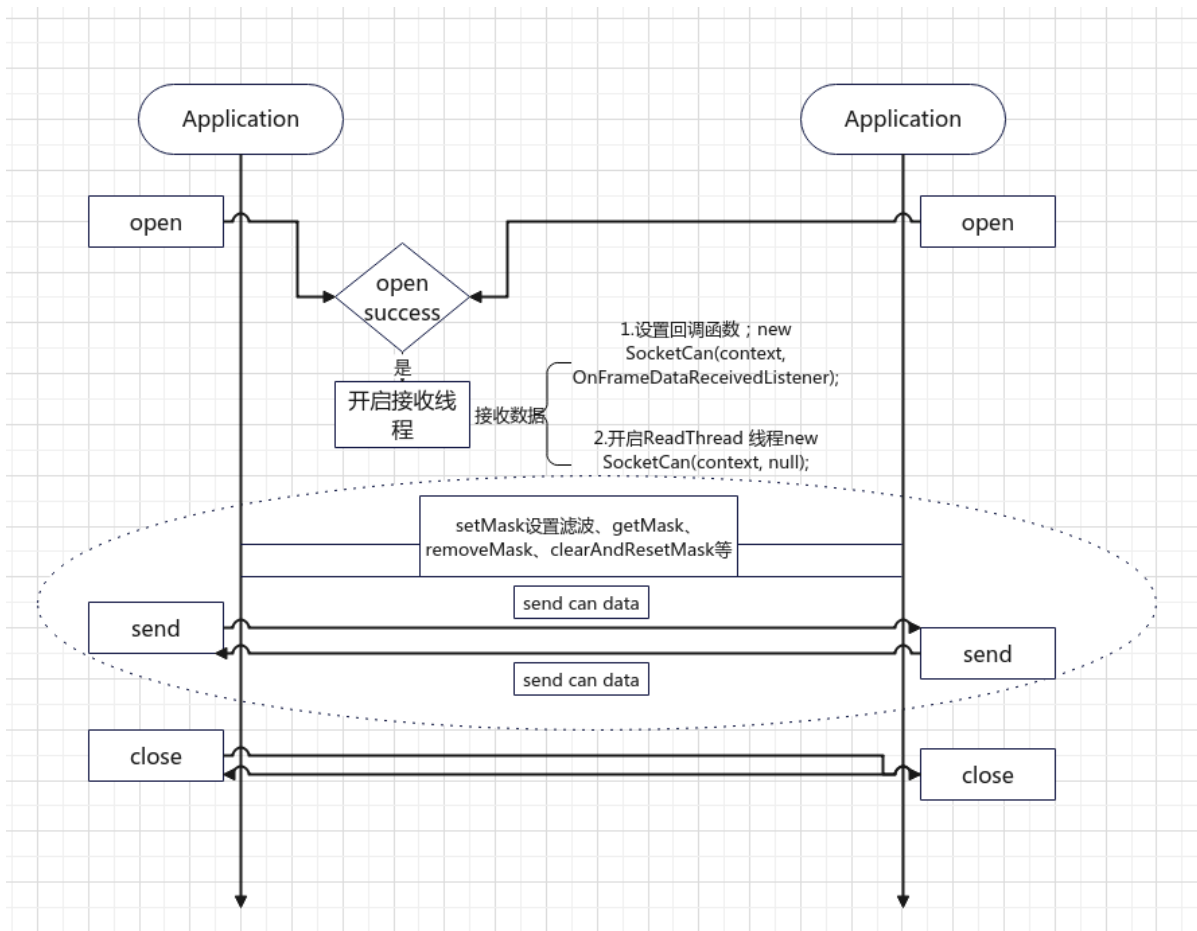
libSocketCan API 主要包括两个部分

- libSocketCan.jar Java 方法库
- libSocketCan.so 本地方法库

## 用法

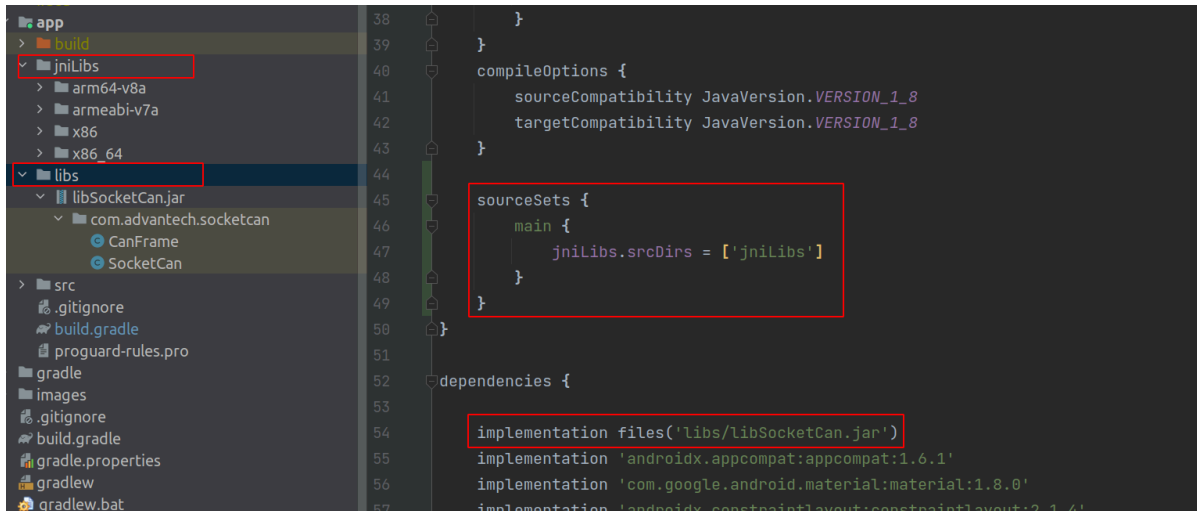
### SDK API 调用流程

我们导入成功 libSocketCan SDK 之后，就可以开始开发我们的 app，具体的使用流程如下图所示



## 在 Android Studio 中导入 SDK

要使用 application 访问 libSocketCan 的功能，您必须将 libSocketCan.jar 和 libSocketCan.so 导入到您的 AndroidStudio 工程中。对应的目录及配置如下图所示。



# API 接口说明

## 接口

接口名	方法成员
<b>OnFrameDataReceivedListener</b>	<i>can data 数据回调</i> <b>void onDataReceived(CanFrame canFrame)</b>

## 类

类名	变量成员	方法成员
<b>CanFrame</b>	<i>can id</i> <b>private int canId</b> <i>can data 是一个 byte 数组</i> <b>private byte[] data</b> <i>can data length 数据长度</i> <b>private int canDlc</b> <i>extended frame 扩展帧</i> <b>private boolean isExtended</b> <i>remote frame 远程帧</i> <b>private boolean isRemote</b>	<i>构造方法</i> <b>public CanFrame(int canId, byte[] data, int canDlc, boolean isExtended, boolean isRemote)</b>
<b>ErrorCodeEnum</b>	<i>数据发送失败</i> <b>ERR_SEND_FAIL</b> <i>参数解析错误</i> <b>ERR_INVALID_ARGUMENT</b> <i>can data 数据长度匹配错误</i> <b>ERR_MISSING_CAN_DLC</b> <i>没有找到目标类</i> <b>ERR_NOT_FIND_CLASS</b> <i>数据解析错误</i> <b>ERR_PARSING_DATA</b> <i>打开或关闭 socket fd 错误</i> <b>ERR_FD_STATUS</b> <i>bind socket failed</i> <b>ERR_BIND_SOCKET</b> <i>端口没有打开</i> <b>ERR_SOCKET_NOT_OPEN</b> <i>*can id out of range</i> <b>ERR_CAN_ID_OUT_OF_RANGE</b> <i>错误码</i> <b>private final int errorCode</b> <i>对应错误码的描述*</i> <b>private final String value</b>	<i>根据错误码获取描述</i> <b>public static String getValue(int errorCode)</b>
<b>Mask</b>	<b>private int filterId1</b> <b>private int mask1</b> <b>private int filterId2</b> <b>private int mask2</b>	<i>判断第一组滤波是否有效</i> <b>public boolean isGroup1Valid()</b> <i>判断第二组滤波是否有效</i> <b>public boolean isGroup2Valid()</b>

类名	变量成员	方法成员
SocketCan	<pre>private Context mContext private boolean isOpened private int mSocketFd private int mPort private String mSpeed private Mask maskFilter private OnFrameDataReceivedListener mDataReceivedListener</pre>	<p>构造函数，用于构造 SokcetCan 实例</p> <pre>public SocketCan(Context context, OnFrameDataReceivedListener listener)</pre> <p>打开 can 接口</p> <pre>public synchronized int open(int port, String speed)</pre> <p>关闭 can0 接口</p> <pre>public synchronized int close()</pre> <p>发送数据帧</p> <pre>public int send(CanFrame frame)</pre> <p>接收数据</p> <pre>public CanFrame recv()</pre> <p>设置滤波</p> <pre>public int setMaskFilter(Mask mask)</pre> <p>移除滤波</p> <pre>public int removeMaskFilter(int maskId)</pre> <p>获取滤波</p> <pre>public Mask getAllMaskFilter()</pre> <p>重置滤波</p> <pre>public int clearAndResetMaskFilter()</pre>

## 函数方法

### SocketCan

- Syntax

```
public SocketCan(Context context, OnFrameDataReceivedListener listener)
```

- Description

构造 SocketCan 实例

- Parameters
  - context：上下文
  - listener：用于监听 can 数据的回调接口，如果传入的 listener 不为 null，则须实现回调方法接收数据，否则需主动起一个线程调用 recv 方法接收数据
- Returns

none

- Remarks

如果传入的第二个参数listener为null，这时候需要代码主动监听数据，需要在can 接口打开成功后，手动开启 ReadingThread 线程

## open

- Syntax

```
public synchronized int open(int port, String speed)
```

- Description

打开 can 总线

- Parameters
  - port : can 接口序号, 比如 0 或 1
  - speed : 用于配置 CAN 接口的速率, 一般情况下可将速率设置为 100 000, 250 000

- Returns

返回值为负值代表打开失败, 反之成功

- Remarks

目前只支持 can0 接口

## close

- Syntax

```
public synchronized int close()
```

- Description

关闭 can0

- Parameters

none

- Returns

返回值为 0 表示关闭成功, 否则失败

- Remarks

none

## send

- Syntax

```
public int send(CanFrame frame)
```

- Description

发送 can 数据

- Parameters

[CanFrame](#) 结构体, 发送数据时需要构造 CanFrame 结构, 传入 canid、data、len 等

- Returns

返回值为负数代表失败，发送成功会返回一个发送数据 data 的长度

- Remarks

仅支持单帧发送

## recv

- Syntax

```
public CanFrame recv()
```

- Description

接收 can 数据

- Parameters

none

- Returns

返回值一个 [CanFrame](#)

- Remarks

调用该函数需要其一个读线程，一直监听数据

## setMaskFilter

- Syntax

```
public int setMaskFilter(Mask mask)
```

- Description

设置滤波 Mask

- Parameters

[mask](#)

- Returns

设置成功返回0

## removeMaskFilter

- Syntax

```
public int removeMaskFilter(int filterId)
```

- Description

移除已经设置过的Mask，

- Parameters

参数传入 filterId

- Returns

移除成功则返回 0

## getAllMaskFilter

- Syntax

```
public Mask getAllMaskFilter()
```

- Description

获取设置的所有滤波

- Parameters

none

- Returns

[Mask](#)

## clearAndResetMaskFilter

- Syntax

```
public int clearAndResetMaskFilter()
```

- Description

重置滤波

- Parameters

none

- Returns

成功则返回 0

## 代码参考

详细使用请参考 [SocketCanSample code](#)