

xCore Android SDK集成文档

本文档是由珞石机器人公司编写，用来引导客户使用SDK控制机器人；Android SDK参考c++版本SDK编写，采用java+jni技术，引用底层c++ SDK，实现机器人控制；因此，c++ SDK和Android SDK底层实现是共用一套代码；

一、修订

版本	修订人	修订时间	修订描述
v1.0.1	丁超	2022-12-30	基于xCore C++ SDK v1.7版本开发
v3.0.1	丁超	2023-09-12	基于xCore C++ SDK v0.3.x版本开发

版本更新

相比于Android SDK v1.0版本，v3.0.1版本有以下更新：

- 废弃了原有的异步函数，新增最新的同步请求函数；
- 废弃了原有的RobotManager类，修改为多个机器人模型单独处理；
- 增加了机器人笛卡尔位姿类（CartesianPosition）、坐标系类(Frame)、负载类（Load）、工具工件类（ToolSet）、轴坐标类（JointPosition）等；
- 增加了机器人类型：6轴协作机器人（XMateRobot）、七轴协作机器人（XMateProRobot）、标准工业机器人（StandardRobot）、PCB3轴机器人（PCB3Robot）、PCB4轴机器人（PCB4Robot）；
- 更新了新的一些函数，例如：setToolSet、projectsInfo、loadProject、ppToMain、runProject、setMaxCacheSize、adjustSpeedOnline....

二、SDK文件介绍

描述最新的Android SDK里面的文件、以及各个文件的使用说明；

- **.aar文件**：里面包含的是最新的SDK内容，包括机器人类型、坐标系类等使用方法；
- **api.xx.zip**：java doc文档，里面包含的是基于SDK开发完成后的api说明文档、以及各类之间的关系，例如：XMateRobot继承关系、XMateProRobot包含哪些函数方法.....
- **xCore Android SDK集成文档**：SDK集成步骤，SDK更新内容说明；
- **SDK_Example.zip**：SDK使用Demo，里面包含了SDK的一些使用用例，以及调用方法；

三、SDK集成步骤

1、新建Android 工程，

- 打开Android Studio编辑器，点击左上角菜单 File，选择 New -> New Project，然后根据步骤一步一步新建工程即可；
- 新建工程时，需要选择minSdk版本，大于或等于Api 28；

2、导入SDK aar文件：

- 打开Android Project，选择默认module名称为app；
- 新建libs文件夹，选择app文件夹，右键菜单，新建文件夹，输入名称libs；

- 然后复制.aar文件到libs目录下;
- 修改build.gradle文件, 在dependencies里面添加:

```
implementation(fileTree("libs")) // 添加外部依赖
```

四、SDK基本使用说明

xCore SDK只要包含有以下功能: 机器人基础信息获取和设置, RL工程加载和运行, 运动指令定义和执行以及其他配置项;

注: xCore SDK 操作机器人, 是需要通过网络传输数据的, 请尽量不要在主线程中进行相关操作, 避免网络引起的ANR问题;

1、第一步, 需要开发者确定实体机器人的类型, 或者实体机器人所属分类,然后根据对应的机器人类型, 新建机器人实例:

- 6轴协作机器人类型 (XMateRobot) ;
- 七轴协作机器人类型 (XMateProRobot) ;
- 标准工业机器人类型 (StandardRobot) ;
- PCB3轴机器人类型 (PCB3Robot) ;
- PCB4轴机器人类型 (PCB4Robot) ;

```
String ip = "192.168.0.160";
XMateRobot robot = new XMateRobot(ip);
XMateProRobot robot = new XMateProRobot(ip);
PCB4Robot robot = new PCB4Robot(ip);
PCB3Robot robot = new PCB3Robot(ip);
StandardRobot robot = new StandardRobot(ip);
```

2、第二步, 就可以使用机器人实例来执行相应的函数方法了:

```
RobotResult result = robot.connect(); // 连接机器人, result里面包含有返回信息;
result = robot.setPowerState(false); // 机器人下电
result = robot.setPowerState(true); // 机器人上电
result = robot.getPowerState(); // 获取机器人上电状态
result = robot.getRobotInfo(); // 获取机器人基本信息
result = robot.getOperateState(); // 获取机器人操作状态
result = robot.setOperateMode(RobotT.RobotMode.Operate_automatic); // 设置机器人操作状态
result = robot.getFlangePos(); // 获取机器人位姿
result = robot.posture(RobotT.CoordinateType.endInRef); // 获取机器人末端位姿
// ...
// 其他一些函数就不再一一列举;
```

注: 完整的robot方法, 请参考Java Doc Api, 完整的使用用例: 参考Example程序;

3、最后, 执行完代码后, 请记得断开连接, 避免过度连接数, 引起的网路问题;

```
robot.disconnect();
```