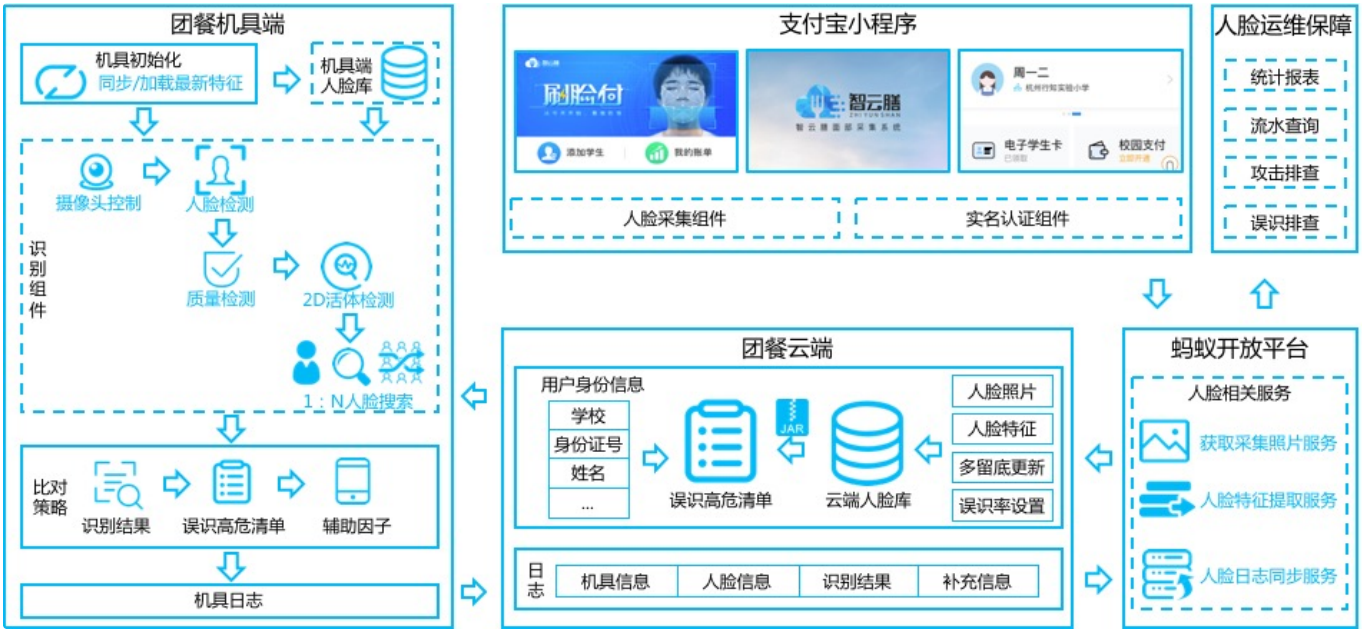
人脸sdk 1.1(原子化)集成文档

0、IoT SDK集成

人脸sdk依赖IoT SDK，为了保证人脸sdk正常初始化，请先集成IoT SDK。集成时，请参考：《支付宝IoTSDK（厂商）集成指南》，该指南说明支付宝IoTSDK的设计和运行原理，并指导厂商进行IoTSDK集成到设备系统运行，链接：https://doc.open.alipay.com/docs/doc.htm?docType=1&articleId=108940。获取IoT SDK和集成文档，请联系@劳宁。

注意：集成IoT SDK时，还会涉及到《支付宝IoTSDK（ ISV）开发手册》(链接：https://doc.open.alipay.com/docs/doc.htm?docType=1&articleId=108944)，此手册为支付宝IoT业务相关应用开发者（ISV）开发使用。**如果IoT SDK仅用于人脸场景，不需要关注此手册**，因为人脸sdk内部会按照此手册进行开发集成。

1、总体流程



1. 团餐机具端开机初始化时，在Application中调用ZFrameFacad.install(this)初始化人脸SDK设备环境
2. 启动刷脸UI前，异步调用接口ZFrameFacad.getSDKInfo获取人脸sdk的模型版本信息和支持的人脸库上限，根据模型版本信息加载人脸特征文件，从团餐服务端同步成功后，调用接口ZFrameFacad.storeFaceFeature加载人脸库到人脸sdk
3. 同时调用ZFrameFacad.init初始化人脸SDK，待init回调succss后继续下面步骤。

4. 调用ZFrameFacad.doVerify准备启动人脸检测识别
5. 人脸库加载完成后，开始学生检测识别时：

1）调用检测组件的ZFrameFacad.doVerify接口启动人脸检测识别，

2）在ZFrameFacadCallback的onInit接口中启动刷脸UI

3）检测过程中，检测状态回调ZFramePresenterCallback的onStateUpdated，业务方可以通过此回调更新UI，state参考接口说明

4）检测过程中，如果业务方设定timeout逻辑，一旦超时，可调用ZFramePresenter.retry继续检测和识别流程

5）活体攻击、检测失败回调onErrorEvent。业务方可调用ZFramePresenter.retry继续检测和识别流程

6）检测成功回调onComplete，识别成功后返回满足要求的人脸list信息

5. 识别完成后调用ZFramePresenter.retry，继续进行其他学生的检测和识别
6. 在团餐机具端退出时调用接口ZFrameFacad.releaseFaceFeature释放人脸库、调用ZFramePresenter.release()、ZFrameFacad.release()释放人脸sdk相关资源

2、刷脸界面定制

本方案中，摄像头控制，包括相机的打开、关闭以及帧数据回传都在CameraSuraceView中完成，业务方只需要在刷脸UI的layout中添加此控件。具体布局和实例化代码可以参考FaceActivity。引用如下：

2.1 实例化CameraSurfaceView

CameraSurfaceView是框架提供的surfaceview，用于帮助开发者渲染摄像头界面，提供最佳的显示效果。在刷脸UI的布局中引用，示例如下：

```
<com.alipay.zoloz.hardware.camera.widget.CameraSurfaceView
    android:id="@+id/surface_view"
    android:layout_width="480px"
    android:layout_height="480px"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="30dp"/>
```

2.2 初始化

```
/**
 * 建议在application中调用
 * ZFrameFacad
 */
public static void install(Context context)
/**
```

```
 * 建议在启动刷脸UI前的Activity中调用
 * ZFrameFacad
 */
public static void init(final Context context, String isvId,
                        final ZFrameInitCallback initCallback)
```

入参说明:

字段	类型	具体说明
context	Context	环境变量
isvId	String	分配的应用开发者（合作伙伴）PID
initCallback	ZFrameInitCallback	人脸sdk初始化结果回调接口，内部会进行设备授权检查。 确保收到callback成功的回调后再继续后续流程

初始化结果回调:

```
public interface ZFrameInitCallback {
    /**
     * 初始化结果回调
     * @param success 初始化成功
     */
    void onResult(boolean success);
}
```

2.3 释放

当流程结束后需要正确的释放资源，防止内存泄漏等问题。

```
/**
 * 释放UI和算法资源
 * ZFramePresenter
 */
public void release();
/**
 * 释放sdk资源
```

```
 * ZFrameFacad
 */
public static void release();
```

3、获取版本号

```
/**
* 获取人脸sdk版本信息
*/
public static FaceSDKInfo getSDKInfo() {
}
```

返回值说明：

人脸sdk版本信息

```
public class FaceSDKVersion {
    public String codeVersion;
    public String modelVersion;
    public int maxFaceCount;
}
```

字段	类型	具体说明
codeVersion	String	sdk工程版本号
modelVersion	String	sdk模型版本号。特征提取和加载时用
maxFaceCount	int	sdk支持的人脸库上限，超过上限加载失败

4、人脸库

4.1 人脸库加载

```
/**
* 调用此接口将特征库加载到人脸sdk，批量加载提高效率
* 需要在worker线程中调用，防止ANR
```

```
*/
public static FeatureStoreResult storeFaceFeature(List<FaceFeature>
faceFeatures) {
}
```

入参说明:

代表人脸特征和所有信息，

```
public class FaceFeature {
    public String faceID;
    public String version;
    public String feature;
    public int type;
    public int riskType;
}
```

字段	类型	是否必填	具体说明
faceID	String	是	用户特征的唯一标示
version	String	是	特征对应的模型版本号
feature	String	是	用户特征
type	int	否	特征类型，默认值：*FEATURE_TYPE_RGB*
riskType	int	否	risk类型，默认值：*RISK_TYPE_NORISK* *属于相似脸(高危人群)时，设置为 RISK_TYPE_RISK_ONE*

返回值说明:

```
public class FeatureStoreResult {
    public int retValue;
    public List<String> failedFaceID;
}
```

字段	类型	是否必填	具体说明
retValue	int	否	人脸库加载返回值(FeatureStoreResult):

			STORE_SUCCESS/STORE_FAILED_VERSION_ERROR /STORE_FAILED_OVERSIZE
failedFaceID	List<String>	否	返回版本号不匹配而无法加载的face信息， 人脸库加载成功时此参数为空

4.2 人脸库释放

```
/**
* 调用此接口释放人脸sdk加载的人脸库
*/
public static boolean releaseFaceFeature() {
}
```

返回值说明:

true： 释放成功

false：释放失败

5、检测识别

检测识别阶段，集成时重点关注参数配置ZFrameConfig(5.2.1)，此参数涉及到算法参数性能、工作流程等，请仔细阅读，根据实际应用场景来设置。

5.1 检测识别

```
/**
 * 人脸检测识别
 * 检测、识别过程中的状态和结果通过ZFramePresenterCallback回调业务方
 *
 */
public static void doVerify(ZFrameFacadCallback callback)
```

入参说明:

```
/**
 * 刷脸识别入口和出口的回调
 */
public interface ZFrameFacadCallback {
    /**
```

```
     * 初始化完成，获取初始化参数的singleTag
     * @param singleTag singleTag
     */
    void onInit(String singleTag);
    /**
     * 识别成功后待比对的人脸数据
     * 本地比对暂时不需要
     * @param result result
     */
    void onResult(String result);
}
```

doVerify示例：

```
ZframeFacad.doVerify(new ZFrameFacadCallback() {
    @Override
    public void onInit(String singleTag) {
        BioLog.i("singleTag是随机唯一标识，用于传入ZFramePrasenter");
        //启动刷脸UI
        Intent intent = new Intent(MainActivity.this, FaceActivity.class);
        intent.putExtra("appTag", singleTag);
        startActivity(intent);
    }
    @Override
    public void onResult(String result) {
        BioLog.i("result是识别的结果，当成功后会通过该回调拿到结果，去请求服务端比对接口
服务");
    }
});
```

注意：如果刷脸UI不是独立的activity，启动刷脸UI时直接把创建并显示即可。

5.2 实例化ZFramePresenter

在刷脸UI逻辑中实例化：

```
mZFramePresenter = new ZFramePresenter(mZFramePresenterCallback, "appTag",
                                       mCameraSurfaceView, zFrameConfig);
```

1. appTag有doVerify的callback通过onInit函数回传给业务方。
2. mZFramePresenterCallback定义参考5.3

5.2.1 ZFrameConfig

```
/**
 * 启动人脸时config信息
 */
public class ZFrameConfig {
    /**
     * 人脸sdk算法配置
     */
    public ToygerFaceAlgorithmConfig faceAlgorithmConfig;
    /**
     * 人脸sdk流程配置
     */
    public ToygerTaskFlowConfig taskFlowConfig;
    /**
     * 人脸比对配置
     */
    public ToygerFaceSearchConfig faceSearchConfig;
    /**
     * Camera设置
     */
    public DeviceSetting deviceSetting;
}
```

注意：所有参数均有默认值，建议有特殊需求时再设置新的参数。

5.2.2 ToygerFaceAlgorithmConfig

用于调整人脸检测和识别环节的相关参数，主要参数如下：

参数	含义	默认值	取值范围
capacity	启用的算法能力	DETECTION|LIVENESS	参考Capacity类，下面参数组合使用： 检测: DETECTION 活体: LIVENESS 抽特征: FEATURE 比对: VERIFY 本地比对场景设置为：DETECTION|LIVENESS|FEATURE|VERIFY
roiRect	人脸识别区域	{0,0,0,0}	默认全图检测人脸

stack_time	人脸保持不动的时间，用于算法采集该时间内质量最好的人脸图片。单位s	0.0f	0～2.0f 本地比对场景设置为0.0f
min_iod	帧图像中，人脸宽度与图像宽度的最小占比。可用于调整检测距离	0.18f	0～1.0f
max_iod	帧图像中，人脸宽度与图像宽度的最大占比。可用于调整检测距离	0.45f	0～1.0f
pose_integrity	人脸完整性	0.9f	0.9f～1.0f
quality_min_quality	检测质量分	20	20.0f～100.0f
liveness_combination	活体配置，类型：List<String>	”DragonflyLiveness”	为空时，不启用活体

5.2.3 ToygerTaskFlowConfig

人脸算法执行模式的设置，参数如下：

参数	含义	默认值	取值范围
livenessPowerMode	活体阶段PowerMode设置	POWER_MODE_LOW	POWER_MODE_LOW：串行 POWER_MODE_MIDDLE：并行
featurePowerMode	识别阶段PowerMode设置	POWER_MODE_LOW	POWER_MODE_LOW：串行 POWER_MODE_MIDDLE：并行
toygerPowerMode	检测与后续环节Power Mode设置	POWER_MODE_LOW	POWER_MODE_LOW：串行 POWER_MODE_MIDDLE：并行 监控场景：建议设置为并行 交易场景：建议设置为串行
careFaceIdChanged	是否换人敏感	true	true：换人时如果上一个还未完成识别，停止 监控场景：建议设置为false 交易场景：建议设置为true

careRecognizedIdChanged	是否track id敏感	false	true：识别成功后，相同track id的图像帧不执行活体和识别 监控场景：建议设置为true 交易场景：建议设置为false
needAutoRecognize	是否连续自动识别	false	true：连续检测识别图像帧， false：图像帧识别成功后，停止识别 监控场景：建议设置为true 交易场景：建议设置为false
detectQueueSize	支持的检测结果队列长度	1	目前仅支持长度为1的队列

5.2.4 ToygerFaceSearchConfig

人脸比对条件设置，参数如下：

参数	含义	默认值	取值范围
securityLevel	比对安全等级阈值	8	1～20
topN	返回topN个满足阈值的用户	1	-1:返回所有满足安全等级的人脸信息

5.2.5 DeviceSetting

相机设备参数设置，仅用于Android 2D摄像头。参数如下：

参数	含义	默认值	取值范围
cameraAuto	是否自动识别要启动的camera ID	true	
cameraID	人脸检测时启动camera的ID	1	设备有效的camera ID。 cameraAuto为true时，此参数无效
displayAuto	是否自动设置预览图像的角度	true	
displayAngle	预览图像角度	90	取值：0，90，180，270 displayAuto为true时，此参数无效

algorithmAuto	是否自动设置图像帧数据的角度	true	
algorithmAngle	图像帧数据角度	270	取值：0，90，180，270 algorithmAuto为true时，此参数无效
widthAuto	是否自动设置相机输出图像分辨率的宽度	true	
width	相机输出图像分辨率宽度	640	widthAuto为true时，此参数无效
zoom	相机焦距设置，可用于调整检测距离	0	取值：0～maxZoom 相机不支持变焦时，此参数无效

5.3 获取结果

```
public interface ZFramePresenterCallback {
    /**
     * 摄像头分辨率比例，用于调整摄像头图像大小
     * @param width width
     * @param height height
     */
    void onSurfaceviewInit(int width, int height);
    /**
     * 算法识别提示code，用于显示每帧图像的提示文案
     * @param state state
     * @param attr 人脸检测属性信息
     */
    void onStateUpdate(int state, ToygerFaceAttr attr);
    /**
     * 人脸检测时用于活体、识别的人脸图
     * @param bitmap 高清图
     * @param toygerFaceAttr 人脸检测属性信息
     */
    void onHighQualityFrame(Bitmap bitmap, ToygerFaceAttr toygerFaceAttr);
    /**
     * 返回的深度人脸图
     * @param bitmap 高清图
     */
    void onHighQualityDepthFrame(Bitmap bitmap);
    /**
```

```
     * 人脸检测或者识别成功
     */
    void onComplete(int result, FeatureVerifyResult verifyResult);
    /**
     * 错误类型，用于UI层弹框、中断逻辑
     * @param errorCode errorCode
     */
    void onErrorEvent(int errorCode);
}
```

5.3.1 状态state说明

```
Class:com.alipay.zoloz.toyger.algorithm.TGFaceState:
   TG_MESSAGE_NO_FACE: 没有检测到脸
   TG_MESSAGE_DISTANCE_TOO_FAR: 靠近一点
   TG_MESSAGE_DISTANCE_TOO_CLOSE: 离远一点
   TG_MESSAGE_FACE_NOT_IN_CENTER: 把脸移入圈内
   TG_MESSAGE_BAD_PITCH: 请正对手机
   TG_MESSAGE_BAD_YAW: 请正对手机
   TG_MESSAGE_IS_MOVING: 再清晰一点
   TG_MESSAGE_BAD_BRIGHTNESS: 脸部亮一点
   TG_MESSAGE_BAD_QUALITY: 请露出正脸
   TG_MESSAGE_BAD_EYE_OPENNESS: 请注视屏幕
   TG_MESSAGE_BLINK_OPENNESS: 眨眨眼
   TG_MESSAGE_STACK_TIME: 请保持不动
   TG_MESSAGE_DEPTH_DAMAGE: 请露出正脸
```

5.3.2 错误类型errorCode说明

```
Class: com.alipay.zoloz.zface.manager.AlertManager:
    ALERT_TYPE_SERVER_RETRY: 服务端错误
    ALERT_TYPE_NET_ERROR: 网络错误
    ALERT_TYPE_SYSTEM_ERROR: 系统错误
    ALERT_TYPE_CAMERA_NO_DEVICE: 无相机设备
    ALERT_TYPE_CAMERA_OPEN_FAIL: 相机打开识别
    ALERT_TYPE_CAMERA_STREAM_ERROR: 获取相机数据失败
    ALERT_TYPE_USER_BACK: 用户退出
    ALERT_TYPE_RECO_TIME_OUT: 超时
    ALERT_TYPE_RECO_OVER_TIME: 重试次数过多
    ALERT_TYPE_LIVENESS_ERROR: 活体失败
```

```
ALERT_TYPE_INIT_TOYGER_FAIL：算法初始化失败
ALERT_TYPE_INIT_TOYGER_SUCCESS：算法初始化成功
```

5.3.3 人脸比对结果说明

FeatureVerifyResult说明：

参数	含义	取值范围
retValue	比对失败详细原因	VERIFY_SUCCESS：比对成功找到人 VERIFY_SUCCESS_NOTFOUND：比对成功未找到人 VERIFY_FAILED：比对失败（异常） VERIFY_SUCCESS_RISK：比对成功，top1属于相似脸(高危人士)
listVerifyInfo	返回topN个满足阈值的用户	根据retValue返回值： VERIFY_SUCCESS：返回满足条件的人脸比对list VERIFY_SUCCESS_UNFOUND：返回top1的人脸比对list VERIFY_SUCCESS_RISK：返回满足条件的人脸比对list VERIFY_FAILED：list为空

FeatureVerifyInfo说明：

参数	含义	取值范围
faceID	人脸特征唯一标示	
securityLevel	比对等级	1～20
compScore	比对分数	0～100，此参数后续会Deprecated，不建议用于业务决策

5.3.4 onComplete result说明

```
/**
 * 0 - 成功
 */
int EVENT_CODE_SUCCESS = 0;
/**
```

```
     * -3 - 活体检测不通过
     */
    int EVENT_CODE_LIVENESS_FAILED = -3;
    /**
     * -15 - 抽feature失败
     */
    int EVENT_CODE_EXTRACT_FEATURE_FAILED = -15;
    /**
     * -17 - 比对失败
     */
    int EVENT_CODE_VERIFY_FAILED = -17;
```

5.4 重试与暂停

```
/**
 * 重新进行人脸检测
 *
 * 一旦检测、活体或者比对流程完成，即人脸sdk回调onComplete后，内部处于完成状态。如果需要重新发起检测，
 * 必须调用retry，以设置正确的人脸sdk状态
 * @param clearState true：重置sdk内部状态后继续检测  false：基于上一次状态继续检测
 */
public void retry(boolean clearState)
/**
 * 停止检测，并忽略相机输出的人脸帧数据
 */
public void stop()
```

6、Demo示例

demo： app-zframedemo-android.zip（16.52 MB）

demo源码完整示例了人脸sdk的集成过程，可参考demo中MYApplication、MainActivity和FaceActivity的相关实现。

7、集成常见问题

7.1 调用ZFrameFacad.init后回调长时间未返回或者返回false

措施：请检测机具是否按照本集成文档《0、IoT SDK集成》部分正确集成IoT SDK

7.2 调用ZFrameFacad.init时logcat中报class not found异常

措施：人脸sdk依赖的部分模块在业务方环境找不到，这些模块对于本地本地是非必须的，抛出的异常也不会影响正常流程。所以如果出现如下exception提示，请忽略：

```
com.alipay.android.phone.mobilesdk.apm.ApmReflectedEntry
com.alipay.mobile.tianyanadapter.monitor.MonitorReflectedEntry
```

7.2 调用doVerify接口启动人脸sdk后，无预览画面

措施：

1. 检查应用的camera permission是否开启
2. 人脸sdk分2d和3d摄像头版本，确认sdk版本和摄像头类型一致
3. 请确保布局中定义的CameraSurfaceView布局visible之前已经实例化ZFramePresenter(参考5.2）。

   对于CameraSurfaceView所在layout先于ZFramePresenter实例化的场景，建议设置成gone。

7.3 人脸特征库加载失败

措施：

1. 请检查本地特征库版本号与人脸sdk接口getSDKInfo的modelVersion是否一致
2. 请检查加载失败的返回值，明确失败的具体原因(参考4.1节的参数“retValue”说明)

7.4 双目摄像头情况下，打开摄像头不对

措施：请检查设置的camera ID是够正确，请参考5.2.5节参数“cameraAuto”与“cameraID”说明进行设置。

7.5 一直检测不到脸

措施：

1. 旋转机具朝向，看是否能在某一方位上检测出人脸
2. 如果某一方位检测人脸，说明设置的android算法角度有问题，参考5.2.5节参数“algorithmAuto”与“algorithmAngle”说明进行设置

7.6 一直未识别成功

措施：

1. 请确保人脸库中存在当前扫脸人的特征
2. 请检查capacity设置是否正确，参考第5.2.2节参数“capacity”说明进行设置。