# Android Camera2 API专题

## Open/Close Camera代码实现

# Agenda

- 枚举Camera数量
- 判断Camera Facing
- 判断Logical MultiCamera
- OpenCamera流程
- CloseCamera流程

# 枚举Camera数量

- CameraControllerManager2

```
@Override
public int getNumberOfCameras() {
    CameraManager manager = (CameraManager) mContext.getSystemService(Context.CAMERA_SERVICE);
    try {
        String[] cameraIdArray = manager.getCameraIdList();
        if(MyDebug.LOG) {
            Log.d(TAG, msg: "getCameraIdList length:" + cameraIdArray.length);
        }
        return cameraIdArray.length;
    } catch(Throwable e) {
        if(MyDebug.LOG) {
            Log.e(TAG, msg: "exception trying to get camera ids");
        }
        e.printStackTrace();
    }
    return 0;
}
```

# 判断Camera Facing

- CameraControllerManager2

```
@Override
public CameraController.Facing getFacing(int cameraId) {
    try {
        String cameraIdS = mCameraManager.getCameraIdList()[cameraId];
        CameraCharacteristics characteristics = mCameraManager.getCameraCharacteristics(cameraIdS);
        switch(characteristics.get(CameraCharacteristics.LENS_FACING)) {
            case CameraMetadata.LENS_FACING_FRONT:
                return CameraController.Facing.FACING_FRONT;
            case CameraMetadata.LENS_FACING_BACK:
                return CameraController.Facing.FACING_BACK;
            case CameraMetadata.LENS_FACING_EXTERNAL:
                return CameraController.Facing.FACING_EXTERNAL;
        }
        Log.e(TAG, msg: "unknown camera_facing: " + characteristics.get(CameraCharacteristics.LENS_FACING));
    } catch(Throwable e) {
        if(MyDebug.LOG)
            Log.e(TAG, msg: "exception trying to get camera characteristics");
        e.printStackTrace();
    }
    return CameraController.Facing.FACING_UNKNOWN;
}
```

# 判断是否是Logical MultiCamera

- CameraControllerManager2

```
@Override
public boolean isLogicalMultiCamera(Context context, int cameraId) {
    try {
        String cameraIdS = mCameraManager.getCameraIdList()[cameraId];
        CameraCharacteristics characteristics = mCameraManager.getCameraCharacteristics(cameraIdS);
        Set<String> phySicalCameraIds = characteristics.getPhysicalCameraIds();
        int[] capabilities = characteristics.get(CameraCharacteristics.REQUEST_AVAILABLE_CAPABILITIES);
        List<Integer> capabilitiesList = new ArrayList<>();
        for(Integer capability : capabilities) {
            capabilitiesList.add(capability);
        }
        int logicalMultiCameraId = CameraCharacteristics.REQUEST_AVAILABLE_CAPABILITIES_LOGICAL_MULTI_CAMERA;
        Log.d(TAG, msg: "(LogicalCamera: " + cameraIdS +
                ",PhysicalCameraIds:" + phySicalCameraIds.toString() +
                ",has LOGICAL_MULTI_CAMERA capability:" +
                capabilitiesList.contains(logicalMultiCameraId) + ")");
        return capabilitiesList.contains(logicalMultiCameraId) && (capabilities.length > 0);
    } catch (Throwable e) {
        if(MyDebug.LOG)
            Log.e(TAG, msg: "exception trying to isLogicalMultiCamera.");
        e.printStackTrace();
    }
    return false;
```

# OpenCamera流程


**实战代码流程**

Preview#onSurfaceTextureAvailable
Preview#mySurfaceCreated
Preview#openCamera
Preview#mOpenCameraTask.execute();
Preview#openCameraCore
new CameraController2
manager.openCamera

# CloseCamera流程

- 调用CameraDevice#close
- CameraController2

```
@Override
public void release() {
    if( MyDebug.LOG )
        Log.i(TAG, msg: "release: " + this);
//      synchronized( background_camera_lock ) {
//          if( captureSession != null ) {
//              captureSession.close();
//              captureSession = null;
//              //pending_request_when_ready = null;
//          }
//      }
    previewBuilder = null;
    previewIsVideoMode = false;
    if( mCameraDevice != null ) {
        mCameraDevice.close();
        mCameraDevice = null;
    }
```

实战代码流程:

Preview#onSurfaceTextureDestroyed

Preview#mySurfaceDestroyed

Preview#closeCamera

new CloseCameraTask

CameraController2#release

mCameraDevice.close

# 答疑

- https://deepinout.com/android-camera-official-documentation/android-camera2-api/android-camera-architecture-intro.html

# Thanks