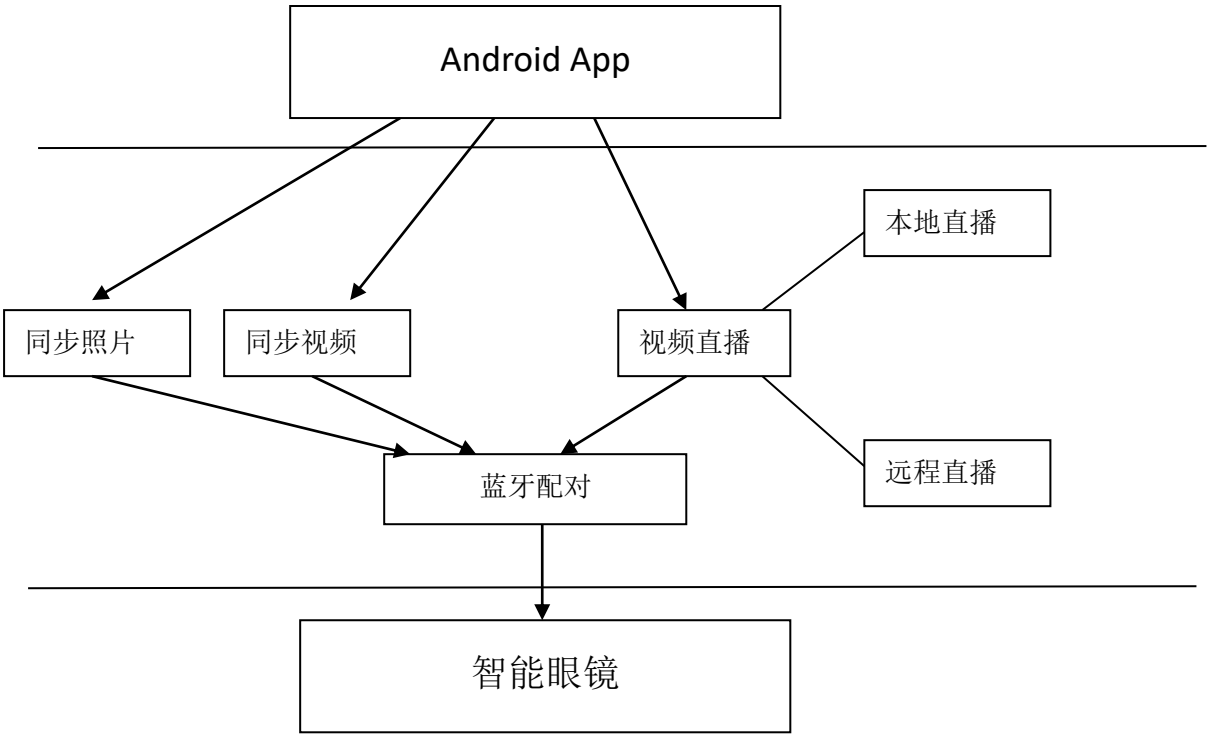
玖趣智能眼镜开发文档

目录

1. 智能眼镜概述

玖趣智能眼镜运行的是 Android 操作系统，通过蓝牙通讯，实现 Android app 和终端设备之间的数据传输和通讯。玖趣智能眼镜实现的功能流程如下：



2. 蓝牙配对

2.1 绑定界面

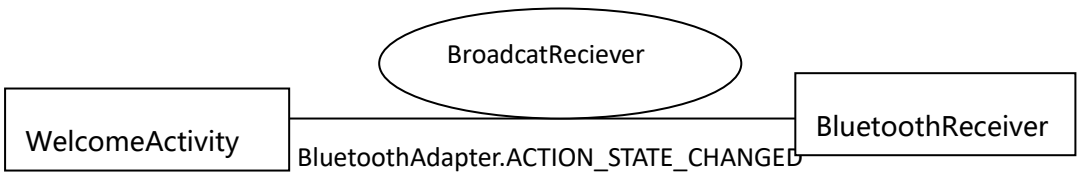
2.2 功能说明

使用 Android App 时，需要事先手动打开蓝牙。蓝牙配对方式有两种：搜索绑定和二维码扫描。智能眼镜每次智能绑定一台设备，所以，当新的 Andorid App 需要配对的时候，要确保没有其他手机配对。

2.3 涉及的主要类

JiuQuMobile/src/cn/ingenic/glasssync/WelcomeActivity.java
JiuQuMobile/src/ com/sctek/smartglasses/ui/MainActivity.java

2.4 结构框架



绑定成功以后就会进入 MainActivity.java 文件，该文件是智能眼镜所有功能的入口。如下：

3．同步照片

3.1 功能界面

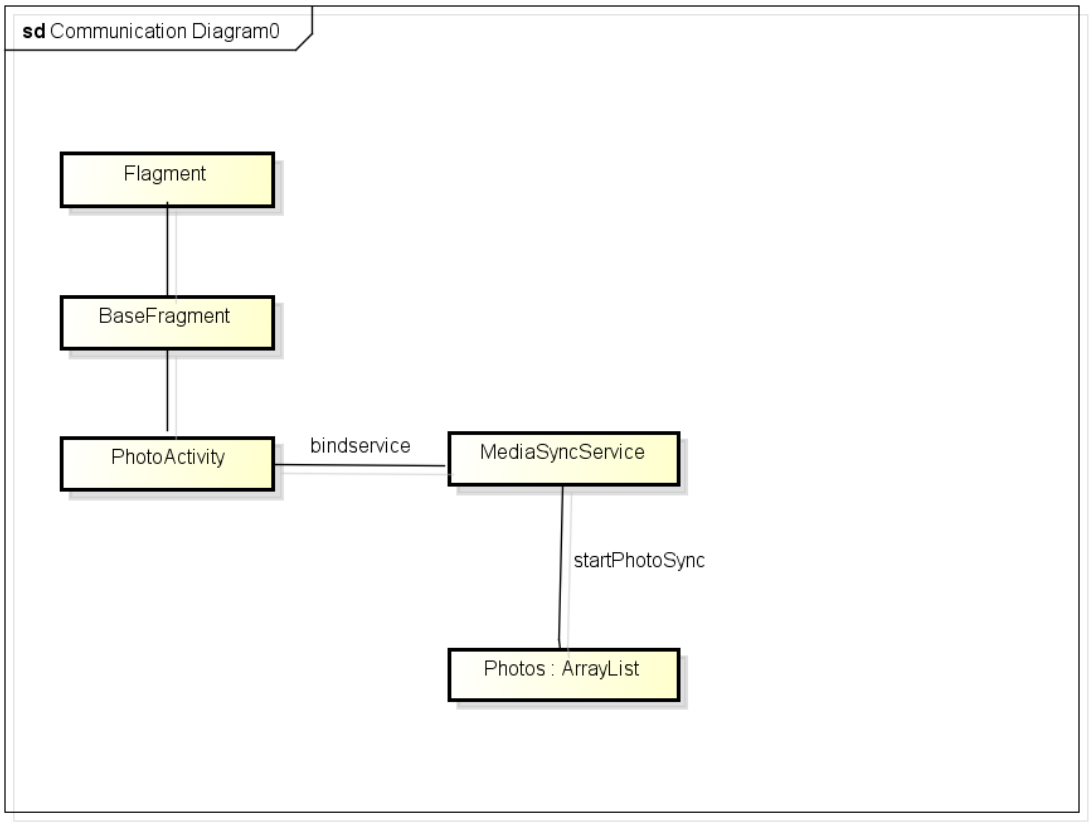
3.2 功能说明

该功能在确保手机和设备配对成功以后，Android App 可以通过蓝牙通讯，浏览设备端拍照片。并且可以将设备端的照片同步下载到手机，还可以分享和删除照片。

3.3 涉及的主要类

JiuQuMobile/src/ com/sctek/smartglasses/ui/PhotoActivity.java
JiuQuMobile/src/cn/ingenic/glasssync/MediaSyncService.java

3.4 代码结构

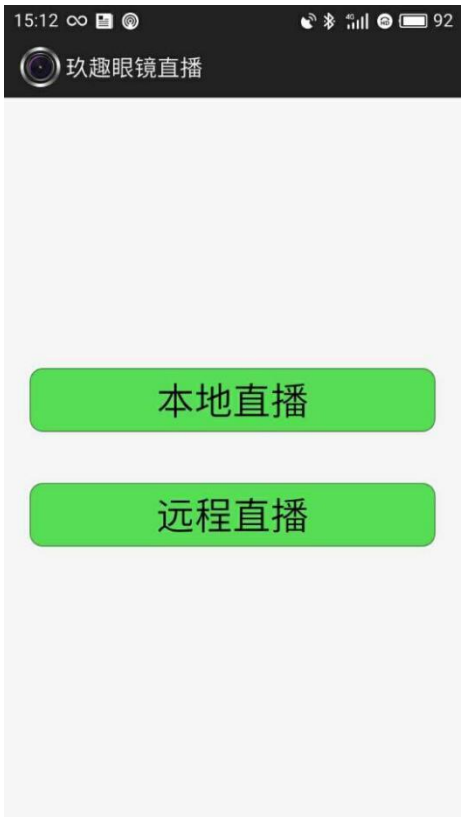


4. 同步视频

该功能和同步照片类似，通过蓝牙通讯，实现 Android App 和设备端之间的视频传输。

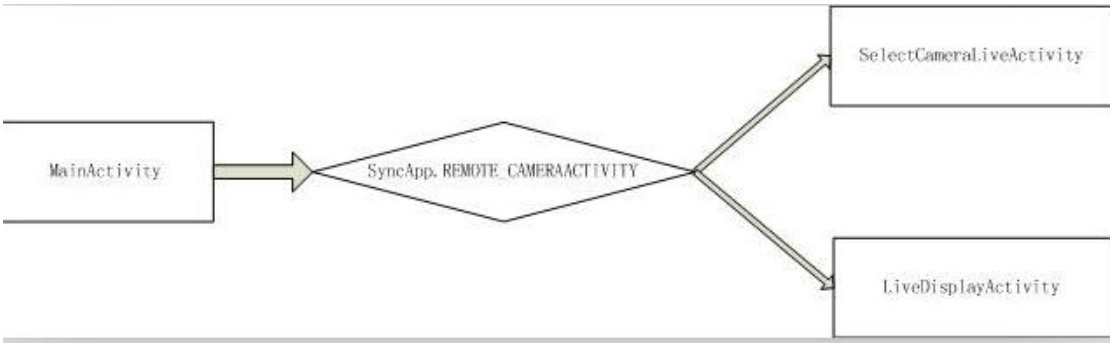
5. 视频直播

5.1 功能概述



视频直播分为本地直播和远程直播，本地直播不需要手机联网，只需要手机和设备在蓝牙配对的情况下，将设备端拍的视频传输到 Android App 端。

视频直播流程图：



远程直播涉及的文件比较多，流程也比较复杂。

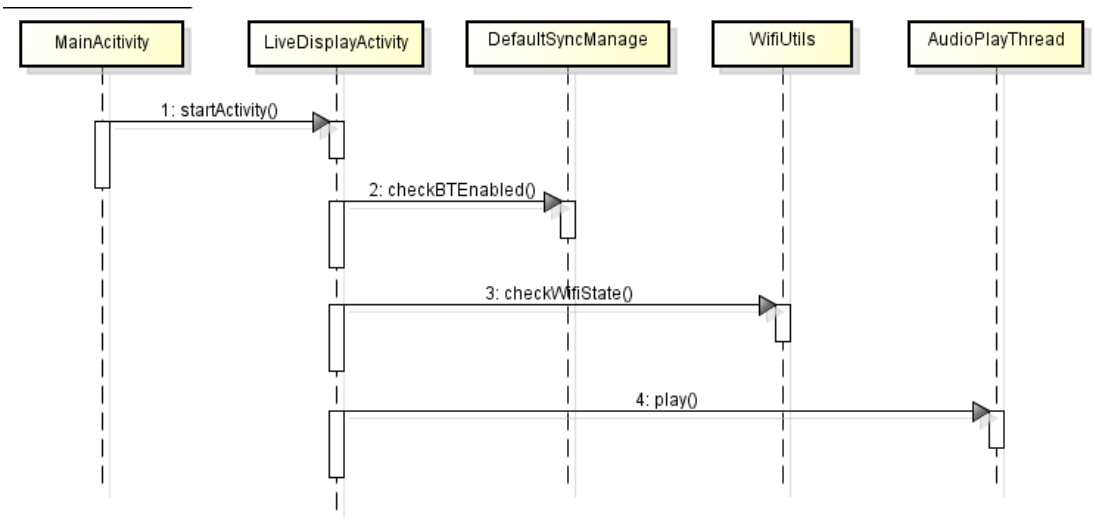
5.2 涉及文件

JiuQuMobile/src/ com/sctek/smartglasses/ui/MainActivity.java
JiuQuMobile/src/ com/sctek/smartglasses/ui/LiveDisplayActivity.java
JiuQuMobile/src/ com/sctek/smartglasses/ui/SelectCameraLiveActivity.java
JiuQuMobile/src/ cn/ingenic/glasssync/DefaultSyncManager.java
JiuQuMobile/src/ cn/ingenic/glasssync/camera/PhotoModule.java
JiuQuSync/src/ cn/ingenic/glasssync/utils/WifiUtils.java
JiuQuSync/src/ cn/ingenic/glasssync/camera/PhotoModule.java
GlassRmtpLive/src/ com/ingenic/glass/livepush/LiveReceiver.java
GlassRmtpLive/src/ com/ingenic/glass/livepush/MainActivity.java
GlassRmtpLive/src/ com/ingenic/glass/livepush/CameraButtonIntentReceiver.java

5.3 实现流程

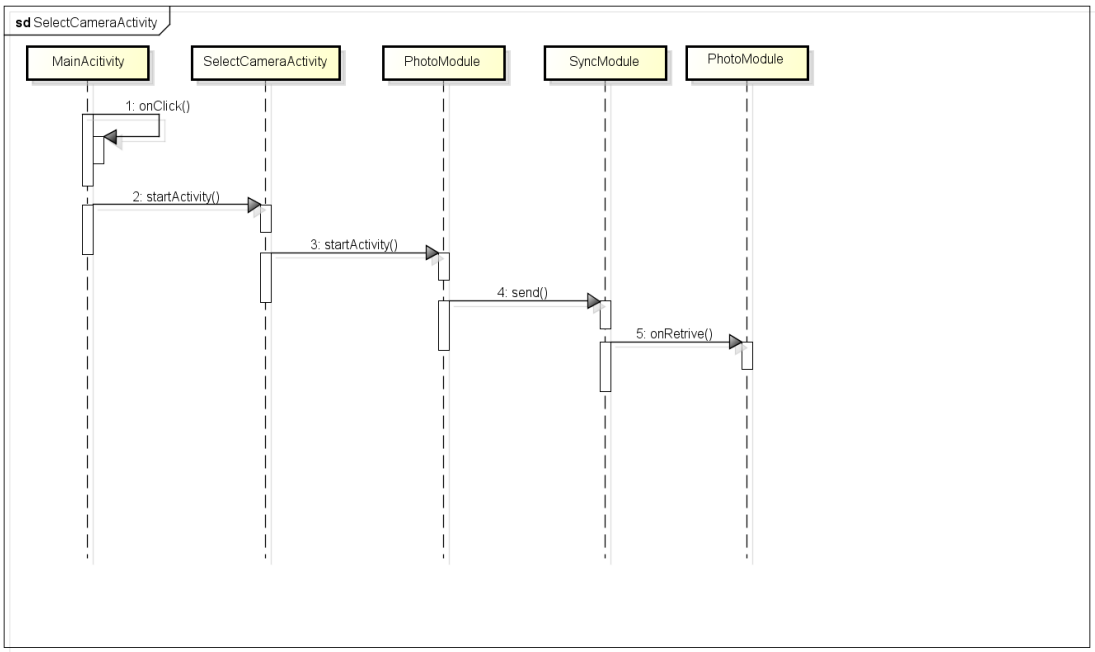
这里会根据 `SyncApp.REMOTE_CAMERA_LIVE` 来判断接下来会进入远程直播 SelectCameraActivity，还是本地直播 `LiveDisplayActivity`。

本地直播的实现过程如图所示。



远程直播手机在联网的情况下，将设备端拍到的视频推送到服务器上，Android App 通过服务器取得直播的房间号，进入房间就可以观看直播内容。远程直播是一种一对多方式，一台设备发起直播，多个用户可以在不同的地方通过 Android App 观看。

远程直播的实现过程如下：



Step 1　MainActivity.onClick

```
public void onClick(View v) {
                    // TODO Auto-generated method stub
                    switch (v.getId()) {
                        .....
                        case R.id.tv_other_menu:
                    if (mHanLangCmdChannel.isConnected()) {// bluetooth is connected or not

                        PhotoModule m = PhotoModule.getInstance(SelectCameraLiveActivity.this);
                        m.rmtv_live();
                        Intent rmtp_intent = new Intent(this, RTMPLiveMainActivity.class);
                        rmtp_intent.putExtra("unbind_state", true);
                        startActivity(rmtp_intent);
                        ......
                    }
        }
```

Step 2 首先，调用 SelectCameraLiveActivity 类，进入直播界面，

```
private OnClickListener mClickedListener = new OnClickListener() {
            public void onClick(View v) {
                    switch (v.getId()) {
                        .....
                        case R.id.live_tv:
                            if (SyncApp.REMOTE_CAMERA_LIVE) {
                              startActivity(new Intent(MainActivity.this,
                                                SelectCameraLiveActivity.class));
                            } else {
                              Intent intent = new Intent(MainActivity.this,
                                                LiveDisplayActivity.class);
                              startActivity(intent);
                            }
```

Step 3 PhotoModule.rmtv_live

```
public void rmtv_live() {
        SyncData data = new SyncData();
        data.putInt(CAMERA_TYPE, RMTP_LIVE);
        if (DEBUG)
            Log.i(TAG, "rmtv_live");
        try {
            send(data);
        } catch (SyncException e) {
            Log.e(TAG, "" + e);
        }
    }
```

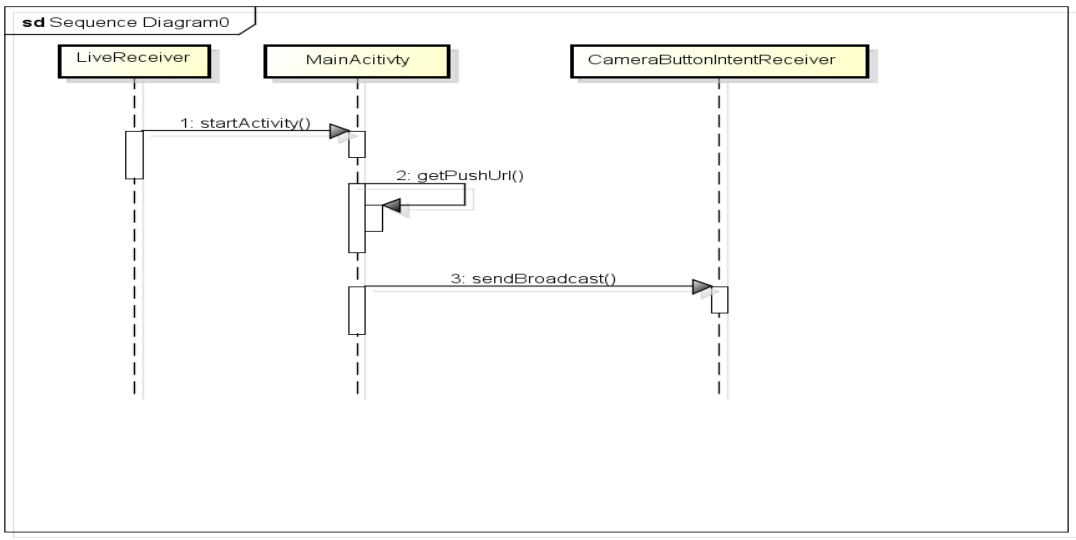
之后，进入 JiuQuSync 的 PhotoModule 中的 onRetrive()函数中。

Step 4 PhotoModule.onRetrive()

```
protected void onRetrive(SyncData data) {
    int message = data.getInt(CAMERA_TYPE);
   ......
    case RMTP_LIVE :
        Intent rmtp = new Intent();
        rmtp.setAction("cn.ingenic.glass.rtmplive");
        mContext.sendBroadcast(rmtp);
        break;
   ......
    }
```

在这里，通过广播的方式，发起远程直播。

接收端的流程如下：



Step 5 LiveReceiver.onReceive()

```
public void onReceive(Context context, Intent intent) {
        ......
        if (ACTION_RTMP_LIVE.equals(action)) {
            if (null != MainActivity.getMInstance()) {
                MainActivity main = MainActivity.getMInstance();
                main.finish();
            }else{
                Intent cameraLiveIntent = new Intent(context,
                        MainActivity.class);
                cameraLiveIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                context.startActivity(cameraLiveIntent);
            }
        }
}
```

LiveReceiver 类在 GlassRMTPLive App 中，这里调用 GlassRMTPLive App 中的 MainActivity。

Step 6 MainActivity.onCreate()

```
protected void onCreate(Bundle savedInstanceState) {
        ......
        IntentFilter filter = new IntentFilter();
        filter.addAction(ACTION_CAMERA_ACTION_STATUS);
        filter.addAction(ACTION_CAMERA_LIVE);
        filter.addAction(ACTION_ACTIVITY_FINISH);
        filter.addAction(ACTION_LIVE_TALK);
        mContext.registerReceiver(mReceiver, filter);

        .....
        mRtmpLiveReceive = new RtmpLiveReceive(this);
        mRtmpLiveReceive.setTalkListener(this);
        if(isNetAvailable()){
            getPushUrl();
            getTalkUrl();
        }else {
            finish();
        }
```

之后调用 getPushUrl()方法。

Step 7 MainActivity.getPushUrl()

```
private void getPushUrl() {
        ......
        mPushUrl = url;
        Intent rmtpLive = new Intent();
        rmtpLive.setAction(ACTION_CAMERA_LIVE);
              rmtpLive.putExtra("liveUrl", url);
              mContext.sendBroadcast(rmtpLive);
        getUrlSuccess =    true;
        ......
              }).start();
}
```

这里启动一个线程，并且以广播的方式，打开 camera 摄像头。

Step 7 CameraButtonIntentReceiver.onReceive()

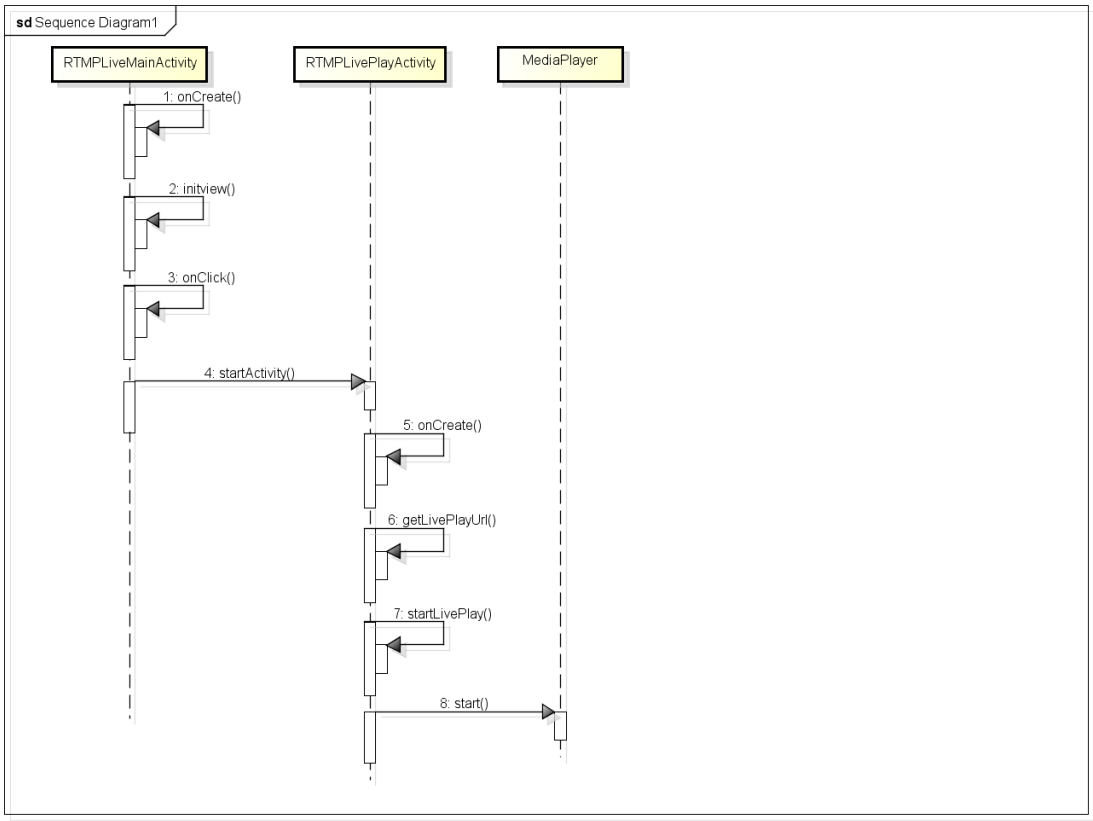
```
public void onReceive(Context context, Intent intent) {
        ......

        } else if (this.CAMERA_LIVE_KEY.equals(intent.getAction())) {
              ......
              Intent cameraLiveIntent = new Intent(context, CameraLive.class);
              cameraLiveIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
              context.startActivity(cameraLiveIntent);
        .......
}
```

之后就启动 CameraLive 类，再把 URL 地址传入 CameraLive 中进行播放。

远程直播的 URL 地址获取的流程就在 RTMPLiveMainActivity。下面看看他的实现过程。

他的实现流程如下：



Step 1　RTMPLiveMainActivity.onCreate

```
protectedvoid onCreate(BundlesavedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.rtmplive_main);
        mContext = this;
        mLiveRoomAdapter = new LiveRoomAdapter(this);
        mClientUid = Utils.getClientUid(mContext);
        Log.i(TAG, "clientuid="+mClientUid);
        initView();
        getLiveList();
        showProgress(getString(R.string.rtmp_livelist_getting));
    }
```

首先加载布局文件，接下来初始化视图界面。

Step 2　RTMPLiveMainActivity.initView

```
privatevoidinitView() {
        mLiveListView = (ListView)findViewById(R.id.live_listview);
        mLiveListView.setAdapter(mLiveRoomAdapter);
        mLiveListView.setOnItemClickListener(this);
        mProgressDialog = new ProgressDialog(this);
}
```

这里将房间以适配器的方式存放在 ListView 中，并且给每个房间增加了点击事件。当点击某一个房间的时候，会自动打开该房间的直播室。

Step 3 RTMPLiveMainActivity.onItemClick

```
publicvoid onItemClick(AdapterView<?>arg0, Viewarg1, intarg2,
            longarg3) {
String name = mLiveRoomList.get(arg2);
if(DEBUG)Log.d(TAG,"onItemClick name="+name);
Intent intent = new
Intent(RTMPLiveMainActivity.this,RTMPLivePlayActivity.class);
intent.putExtra("room_name", name);
startActivity(intent);
}
```

这里会通过 intent 跳转到 RTMPLivePlayActivity 中。

Step 4 RTMPLivePlayActivity.onCreate

```
protectedvoid onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.rtmplive_play);
        mGlassUid = getIntent().getStringExtra("room_name");
        mRTMPLivePush = new RTMPLivePush(this);
        mClientUid = Utils.getClientUid(this);
        initView();
        initIjkpalyer();

      getLivePlayUrl();
    }
```

这里首先会加载布局文件，接下来初始化视图，初始化播放器，最后再加载房间的 URL 地址。这里重点看一下获取 URL 地址的函数。

Step 5 RTMPLivePlayActivity.getLivePlayUrl

```
privatevoid getLivePlayUrl() {
    if(DEBUG)Log.i(TAG, "getLivePlayUrl");
new Thread(new Runnable() {
    @Override
    publicvoid run() {
        try {
            StringBuilder getLiveUrl = new StringBuilder(
                    URL_GET_LIVE_PLAY + "?");
        getLiveUrl.append(PARAMETER_GLASS_UID+"="+mGlassUid);
        getLiveUrl.append("&"+PARAMETER_CLIENT_UID + "=" + mClientUid);
        getLiveUrl.append("&" + PARAMETER_CMP_UID + "=" +SyncApp.COMPANY_UID);
        HttpGet httpGet = new HttpGet(getLiveUrl.toString());
        HttpClient httpClient = new DefaultHttpClient();
        HttpResponse httpResponse = httpClient.execute(httpGet);
        Log.i(TAG, "statusCode="
                + httpResponse.getStatusLine().getStatusCode());
        if (httpResponse.getStatusLine().getStatusCode() == 200) {
            HttpEntity entity = httpResponse.getEntity();
            String response = EntityUtils.toString(entity, "utf-8");
            JSONObject resp = new JSONObject(response);
            int errorCode = resp.getInt(PARAMETER_ERROR_CODE);
            String errorMessage = resp
                    .getString(PARAMETER_ERROR_MESSAGE);
            Log.i(TAG, "errorCode=" + errorCode + ",errorMessage="
                    + errorMessage);
            if (errorCode == 0) {
                String url = resp.getString(PARAMETER_PLAY_URL);
                if (url != null&& url.length() != 0) {
                    if(DEBUG)Log.i(TAG,"url="+url);
                        mPlayUrl = url;
                        mHandler.sendEmptyMessage(MSG_GET_PALYURL_SUCCESS);
                    }
                } else {
                    mHandler.sendEmptyMessage(MSG_GET_PALYURL_CLOSED);
                }
            }else {
                mHandler.sendEmptyMessage(MSG_GET_PALYURL_FAIL);
            }
        } catch (Exception e) {
            mHandler.sendEmptyMessage(MSG_GET_PALYURL_FAIL);
            e.printStackTrace();
        }
    }
    }).start();
}
```

mHandler.sendEmptyMessage(MSG_GET_PALYURL_SUCCESS)发用一条 Message 消息。接下来就是 Handler 接受此消息进行处理。

```
private Handler mHandler = new Handler(){
        publicvoid handleMessage(Message msg) {
            switch (msg.what) {
            case MSG_GET_PALYURL_SUCCESS:
                startLivePlay();
                break;
}
```

Step 6 RTMPLivePlayActivity.startLivePlay

```
protectedvoidstartLivePlay() {
        mIjkVideoView.setVideoPath(mPlayUrl);
        mIjkVideoView.start();
}
```

这里继续调用 IjkVideoView.start 方法。

Step 7. IjkVideoView.start

```
publicvoid start() {
    if(DEBUG)Log.i(TAG, "start");
if (isInPlaybackState()) {
          mMediaPlayer.start();
          mCurrentState = STATE_PLAYING;
  }
        mTargetState = STATE_PLAYING;
}
```

这里就会调用 MediaPlayer.start 方式进行播放了。
发起远程直播的方式有两种：一键直播和 App 远程直播。一键直播是在 Android 系统源码中实现的，App 直播其实就是通过发送广播的方式，其中直播室。

6. 一键拍照

6.1 功能概述

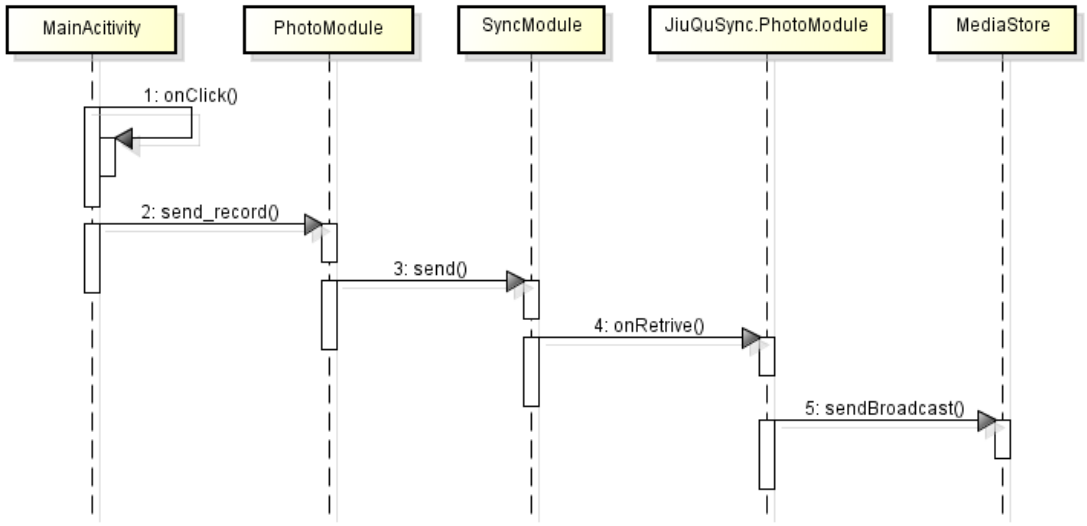
该功能在手机和设备配对的情况下，Android App 发起一键拍照，这种方式也属于蓝牙通讯方式。

6.2 涉及文件

JiuQuMobile/src/ com/sctek/smartglasses/ui/MainActivity.java
JiuQuMobile/src/ cn/ingenic/glasssync/camera/PhotoModule.java
JiuQuSync/src/ cn/ingenic/glasssync/camera/PhotoModule.java

6.3 实现流程

一键拍照的时序图如下：



Step 1 MainActivity.onClick

```
private OnClickListener mClickedListener = new OnClickListener(){
public void onClick(View v) {
......
case R.id.take_video_bt:
        if(mHanLangCmdChannel.isConnected()) {
        PhotoModule.getInstance(getApplicationContext()).send_record();
        takeVideoBt.setEnabled(false);
                handler.postDelayed(takeVideoRunnable, 2000);
......
}
```

在这里首先判断蓝牙是否连接成功，如果成功，再调用 PhotoModule.send_record()，之后将拍照按钮置为不可编辑。

Step 2　PhotoModule.send_record()

```
public void send_record(){
    SyncData data = new SyncData();
    data.putInt(CAMERA_TYPE, RECORD);
    if(DEBUG)Log.i(TAG,"send_start_record");
    try {
            send(data);
    } catch (SyncException e) {
            Log.e(TAG, "" + e);
```

这里 PhotoModule 继承了 SyncModule。在这个函数中，调用 SyncModule 的 send 方法，并且传入 SyncData 数据。

Step 3 PhotoModule.onRetrive

在 JiuquSync App 中，同样也有个继承 SyncModule 的 PhotoModule 类，负责接收传递过来的数据。

```
protected void onRetrive(SyncData data) {
    int message = data.getInt(CAMERA_TYPE);
    if (DEBUG) Log.d(TAG, "onRetrive data"+message);
    switch (message) {
    case TAKE_PHOTO :
        Intent intentCapture = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        mContext.sendBroadcast(intentCapture);
        break;
    case RECORD :
        Intent stopRecord = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
        mContext.sendBroadcast(stopRecord);
        break;
        .....
    }
    }
```

在这里，根据传递的参数：拍照还是录像，发送不同的广播，最终通过 MediaStore 来实现。

另外一种拍照方式是设备端实现的按键拍照。

7. 一键录像

该功能和一键拍照功能类似，只是在 JiuQuSync APP 下发送的广播不一样，拍照发送的是 MediaStore.ACTION_IMAGE_CAPTURE，
录像发送的是 MediaStore.ACTION_VIDEO_CAPTURE。在这里就不在赘述了。通过蓝牙通讯，实现 Android App 控制录像功能。