Android Camera架构分析 - xiaofengcanyue2013的专栏 - 博客频道

分类：

framework（38）

源代码版本：allwinner 4.0.4

frameworks代码：

frameworks/base/core/Java/Android/hardware/Camera.java

JNI层代码：

frameworks/base/core/jni/android_hardware_Camera.cpp

client端代码：

frameworks/base/libs/camera/

server端代码：

frameworks/base/services/camera/libcameraservice/

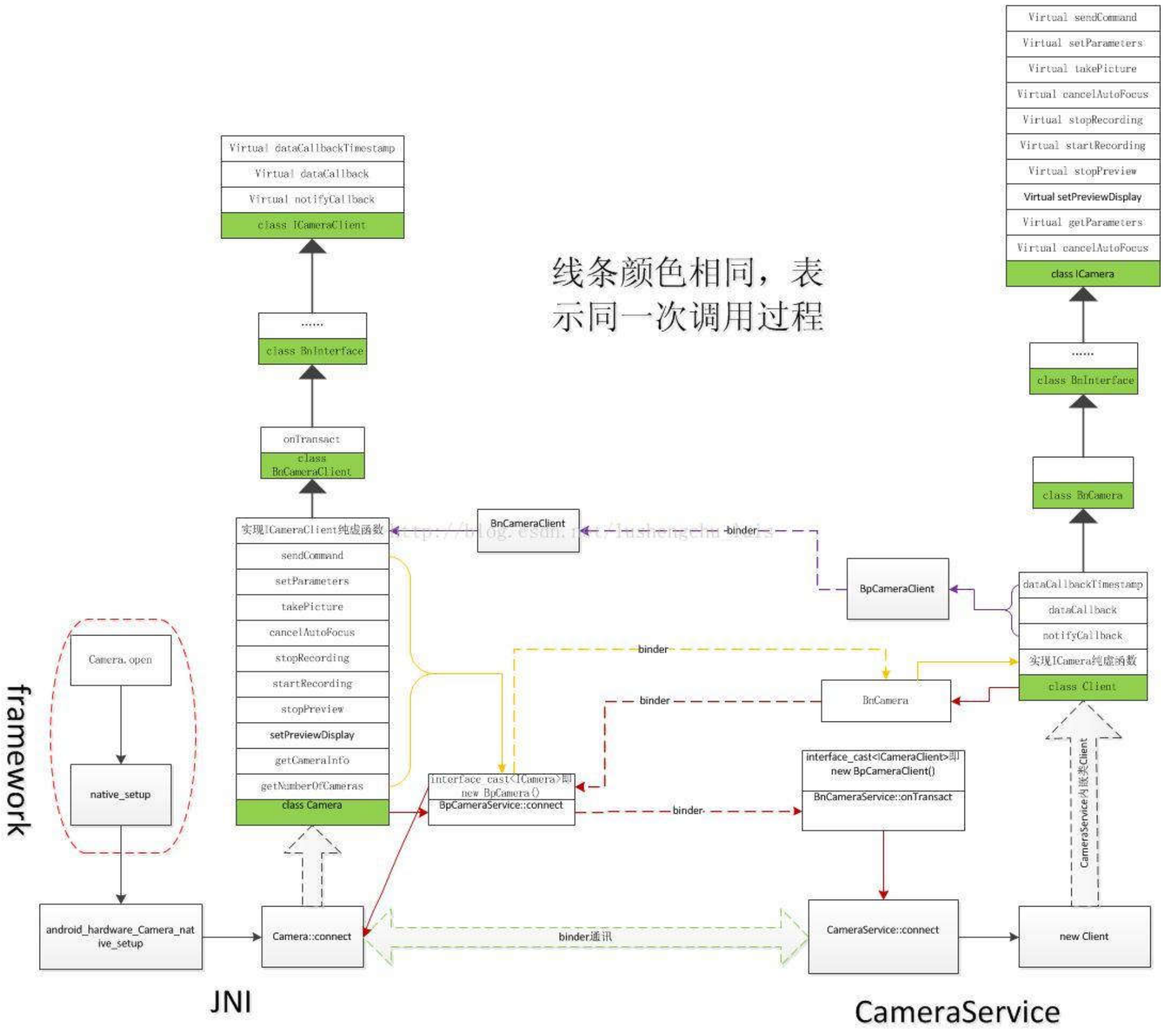
HAL层代码：

device/softwinner/common/hardware/camera/

camera配置文件：

system/etc/ camera.cfg

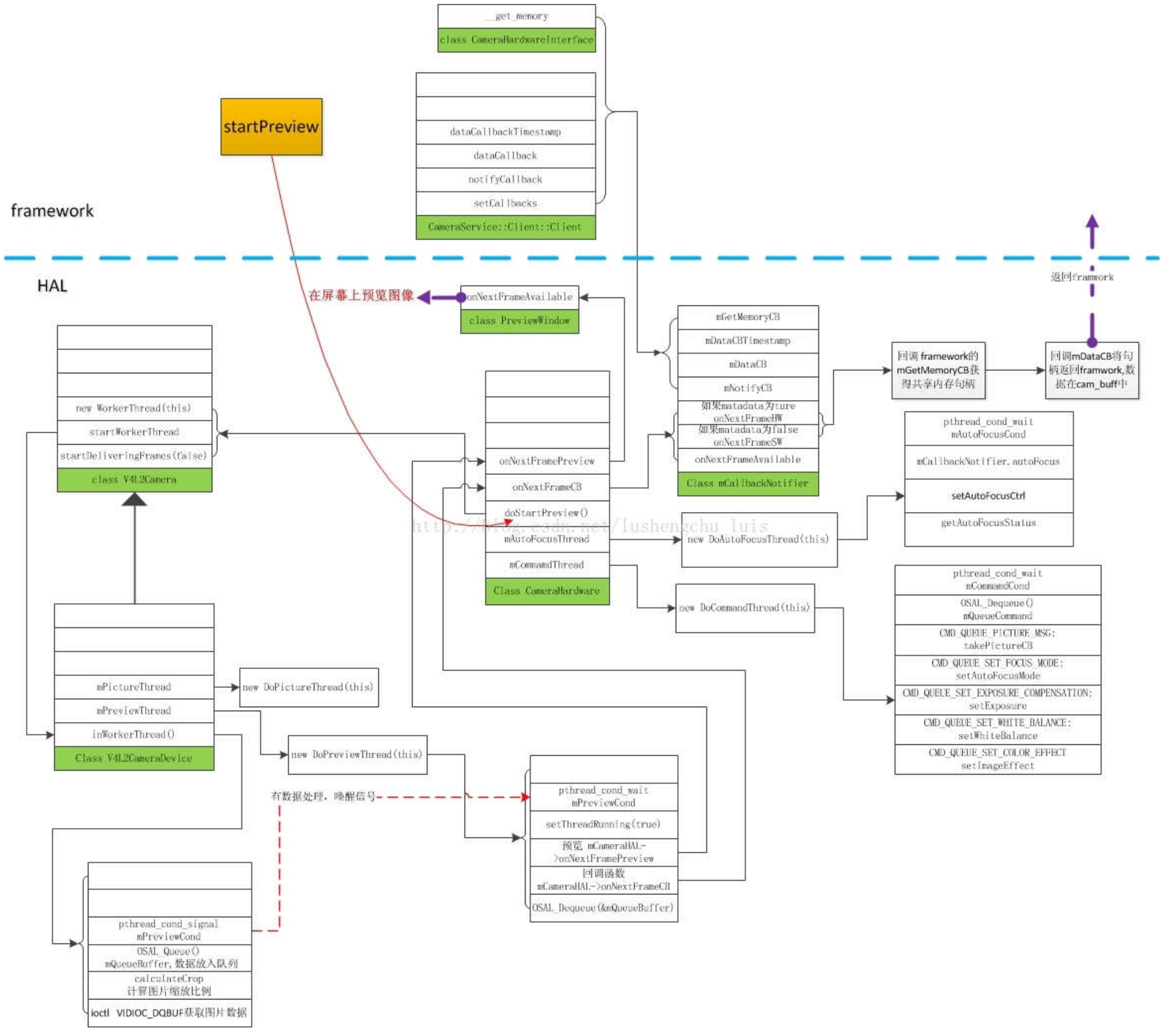
先从应用程序open camera分析，其调用流程如下图：



图中，JNI的Camera::connect函数通过跨进程调用，服务端CameraService会创建Client实体BnCamera, 并且经过binder转换，返回给JNI connet函数一个BpCamera代理，有了这个代理，就可以远程调用CameraService内嵌类client的方法,如startPreview，autoFocus等等；

并且，Camera::connect将Camera对象作为参数传递给服务端，Camera类继承自BnCameraClient，所以JNI 端Camera自身也创建了BnCameraClient实体，跨进程后，CameraService获得其代理BpCameraClient，有了这个代理，服务端就可以回调Camera端的notifyCallback，dataCallback，dataCallbackTimestamp等方法。在这种情景下，可以把CameraService视为客户端，而Camera则为服务端。

framework层的调用流程大概就如上所述，HAL层是厂商自己实现的，不同方案代码结构各有不同，这里分析的是allwinner的源代码。接着以startPreview为例子，看framework层是如何调用HAL层的。调用流程如下图所示：



对上述图中简要说明几点：

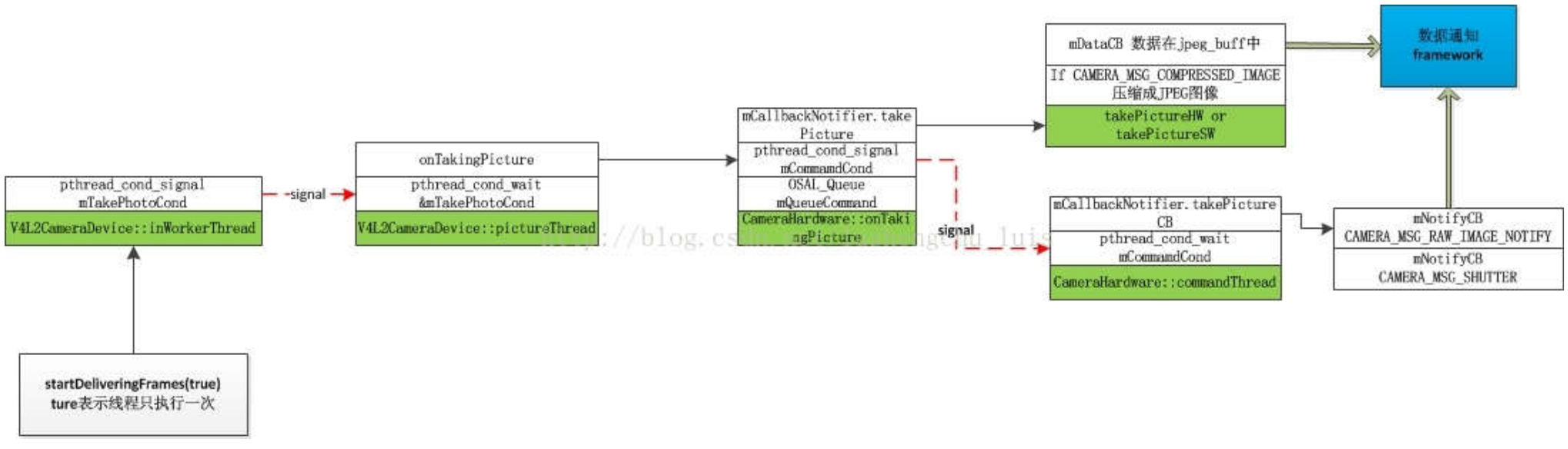
1. framework的几个callback函数在构造函数初始化时候就已经注册到了HAL中，以实现HAL层有数据时候回调framework，最终framework回调app中的注册函数

2. CameraHardware类在初始化时候开启了DoCommandThread和DoAutoFocusThread线程，用来监听上层的命令和动作，前者监听如设置摄像头的白光平衡，曝光度，快门模式等；后者用来监听自动对焦命令。

3. doStartPreview最终会调用V4L2Camera类的startWorkerThread来启动线程，该线程实现方法在V4L2CameraDevice类的inWorkerThread函数。该函数首先调用ioctl(mCamFd, VIDIOC_DQBUF, buf)获取图像数据，并将数据进行格式转换和大小缩放。最后用OSAL_Queue方法将数据入队列并发出通知信号。

4. V4L2CameraDevice类的previewThread线程收到信号，调用onNextFrameCB，回调framwork函数将数据返回上层；并调用onNextFramePreview将数据在窗口中进行预览，这就是打开摄像头看到的预览图像。

其他流程如takePicture，startRecording等和上述差不多，下面给出takePicture的流程图，其它不再重述。



最后看应用层调用takePicture拍照时候的数据回调过程。

应用层拍照的调用函数是　camera.takePicture(shutterCallback, rawCallback, jpegCallback);

shutterCallback类要实现Camera.ShutterCallback接口；　rawCallback, jpegCallback类要实现Camera.PictureCallback接口，他们的一般形式如下：

```
1. //返回照片的JPEG格式的数据
2. private JpegCallback jpegCallback = new PictureCallback(){
3.     public void onPictureTaken(byte[] data, Camera camera) {
4.         Parameters ps = camera.getParameters();
5.         if(ps.getPictureFormat() == PixelFormat.JPEG){
6.             //存储拍照获得的图片
7.             try {
8.                 String path = "/sdcard/"+System.currentTimeMillis()+".jpg";
9.                 File file = new File(path);
10.                if(!file.exists())
11.                    file.createNewFile();
12.                FileOutputStream fos = new FileOutputStream(file);
13.                fos.write(data);
14.                fos.close();
15.            } catch (Exception e) {
16.                e.printStackTrace();
17.            }
18.            //将图片交给Image程序处理
19.            Uri uri = Uri.fromFile(new File(path));
20.            Intent intent = new Intent();
21.            intent.setAction("android.intent.action.VIEW");
22.            intent.setDataAndType(uri, "image/jpeg");
23.            startActivity(intent);
24.        }
25.    }
26. };
```