Camera version support

The Android 5.0 (Lollipop) platform release adds a new app-level camera framework. This document outlines some logistical details that OEMs and SoC vendors need to know.

IN THIS DOCUMENT

Terms

The following terms are used in this document:

- *Camera API1*: The app-level camera framework on KitKat and earlier devices, exposed through the `android.hardware.Camera` class.
- *Camera API2*: The app-level camera framework on 5.0 and later devices, exposed through the `android.hardware.camera2` package.
- *Camera HAL*: The camera module layer that SoC vendors implement. The app-level public frameworks are built on top of the camera HAL.
- *Camera HAL3.2*: The version of the camera device HAL that is being released with Lollipop. KitKat launched with an earlier version (Camera HAL3.1).
- *Camera API1 CTS*: The set of camera Compatibility Test Suite (CTS) tests that run on top of Camera API1.
- *Camera API2 CTS*: An additional set of camera CTS tests that run on top of Camera API2.

Camera API2 overview

The new camera frameworks expose lower-level camera control to the app, including efficient zero-copy burst/streaming flows and per-frame controls of exposure, gain, white balance gains, color conversion, denoising, sharpening, and more. See this brief video overview from the Google I/O 2014 conference (https://www.youtube.com/watch?v=92fgcUNCHic&feature=youtu.be&t=29m50s) for additional details.

Camera API1 availability and deprecation in Android 5.0

The Camera API1 interfaces are still available for apps to use on Android 5.0 and later devices, and camera apps built on top of Camera API1 should work as before. Camera API1 is being marked as deprecated in Lollipop, indicating that it will be phased out over time and new platform development will focus on Camera API2. However, we expect this phase-out period to be lengthy, and Camera API1 apps will continue to be supported in Android for some time to come.

All earlier camera HAL versions, including Camera HAL1.0, will also continue to be supported.

Camera API2 capabilities and support levels

Android 5.0 and later devices feature Camera API2, however they may not fully support all of the new features of Camera API2. The `android.info.supportedHardwareLevel` property that apps can query through the Camera API2 interfaces report one of three support levels: `LEGACY`, `FULL`, and `LIMITED`.

Legacy devices expose a level of capabilities through the Camera API2 interfaces that are approximately the same as is exposed to apps through the Camera API1 interfaces; the legacy frameworks code conceptually translates Camera API2 calls into Camera API1 calls under the hood. Legacy devices do not support the new Camera API2 features including per-frame controls.

Full devices support all of the major capabilities of Camera API2. Full devices by necessity must have a Camera HAL version of 3.2 (shipping with Android 5.0) or later.

Limited devices are in between: They support some of the new Camera API2 capabilities, but not all of them, and must also comprise a Camera HAL version of 3.2 or later.

Individual capabilities are exposed via the `android.request.availableCapabilities` property in the Camera API2 interfaces. Full devices require both the `MANUAL_SENSOR` and `MANUAL_POST_PROCESSING` capabilities, among others. There is also a `RAW` capability that is optional even for full devices. Limited devices can advertise any subset of these capabilities, including none of them. However, the `BACKWARD_COMPATIBLE` capability must always be defined.

The supported hardware level of the device, as well as the specific Camera API2 capabilities it supports, are available as the following feature flags to allow Play Store filtering of Camera API2 camera apps; a device must define the feature flag if any of its attached camera devices supports the feature.

- `android.hardware.camera.hardware_level.full`
- `android.hardware.camera.capability.raw`
- `android.hardware.camera.capability.manual_sensor`
- `android.hardware.camera.capability.manual_post_processing`

CTS requirements

Android 5.0 and later devices must pass both Camera API1 CTS and Camera API2 CTS. And as always, devices are required to pass the CTS Verifier camera tests.

To add some context: For devices that don't feature a Camera HAL3.2 implementation and are not capable of supporting the full Camera API2 interfaces, the Camera API2 CTS tests must still be passed. However, in this case the device will be running in Camera API2 *legacy* mode (in which the Camera API2 calls are conceptually just mapped to Camera API1 calls); and any Camera API2 CTS tests that relate to features or capabilities beyond Camera API1 have logic that will skip them in the case of old (legacy) devices.

On a legacy device, the Camera API2 CTS tests that are not skipped are purely using the existing public Camera API1 interfaces and capabilities (with no new requirements), and any bugs that are exposed (which will in turn cause a Camera API2 CTS failure) are bugs that were already present in the device's existing Camera HAL and would also be a bug that could be easily hit by existing Camera API1 apps. The expectation is that there should be very few bugs of this nature. Nevertheless, any such bugs will need to be fixed.

Version history

3.2

Second revision of expanded-capability HAL:

- Deprecates get_metadata_vendor_tag_ops. Please use get_vendor_tag_ops in camera_common.h instead.
- register_stream_buffers deprecated. All gralloc buffers provided by framework to HAL in process_capture_request may be new at any time.
- Add partial result support. process_capture_result may be called multiple times with a subset of the available result before the full result is available.
- Add manual template to camera3_request_template. The applications may use this template to control the capture settings directly.
- Rework the bidirectional and input stream specifications.
- Change the input buffer return path. The buffer is returned in process_capture_result instead of process_capture_request.

3.1

Minor revision of expanded-capability HAL:

- configure_streams passes consumer usage flags to the HAL.
- flush call to drop all in-flight requests/buffers as fast as possible.

3.0

First revision of expanded-capability HAL:

- Major version change since the ABI is completely different. No change to the required hardware capabilities or operational model from 2.0.
- Reworked input request and stream queue interfaces: Framework calls into HAL with next request and stream buffers already dequeued. Sync framework support is included, necessary for efficient implementations.
- Moved triggers into requests, most notifications into results.
- Consolidated all callbacks into framework into one structure, and all setup methods into a single initialize() call.
- Made stream configuration into a single call to simplify stream management. Bidirectional streams replace STREAM_FROM_STREAM construct.
- Limited mode semantics for older/limited hardware devices.

2.0

Initial release of expanded-capability HAL (Android 4.2) [camera2.h]:

- Sufficient for implementing existing android.hardware.Camera API.
- Allows for ZSL queue in camera service layer
- Not tested for any new features such manual capture control, Bayer RAW capture, reprocessing of RAW data.

1.0

Initial Android camera HAL (Android 4.0) [camera.h]:

- Converted from C++ CameraHardwareInterface abstraction layer.
- Supports android.hardware.Camera API.