ToiletPaper #136

Android CameraX: Let’s take a selfie

Author: Kevin Stieglitz / Software Engineer / Business Division Automotive World

✕ Problem

With Android, using the camera has always been a bit of a challenge. After Google marked the old android.hardware.camera API as deprecated with Android 5 (Lollipop) due to insufficient functionality, the Camera2 API was introduced in return. Now, manufacturers and developers have been able to integrate even complex functions into the camera. In return, however, the effort and complexity of the implementation has also increased considerably.

✔ Solution

To counter the increased complexity, the CameraX library was introduced at Google IO 2019, which is part of the Android Jetpack component set. CameraX is based on the Camera2 API and can be used from Android 5 and beyond. It implements the idea of use-cases to ensure a simple API. Currently, three use-cases are provided:

- Preview: displays a preview of the images
- Image analysis: provides access to a stream of images for subsequent processing
- Image capture: stores images in high resolution

CameraX does not only offer various use cases, but also takes over the configuration of device-specific settings. Google provides its own test lab for this purpose, in which many different devices are tested automatically.
In addition, an optional add-on called "extensions" is provided, which enables access to device-specific effects (such as HDR, night, or potrait mode).

➔ Example

The following snippet shows how little effort is required to integrate the preview use case into the system. The result can be seen in the screenshot above.

```
private fun bindPreviewUseCase() {
    // Get screen metrics used to setup camera for full screen resolution
    val metrics = DisplayMetrics().also { textureView.display.getRealMetrics(it) }
    val screenAspectRatio = Rational(metrics.widthPixels, metrics.heightPixels)

    // Set up the view finder use case to display camera preview
    val viewFinderConfig = PreviewConfig.Builder().apply {
        setLensFacing(CameraX.LensFacing.FRONT)
        setTargetAspectRatio(screenAspectRatio)
        setTargetRotation(textureView.display.rotation)
    }.build()

    // Use the auto-fit preview builder to automatically handle size/orientation changes
    preview = AutoFitPreviewBuilder.build(viewFinderConfig, textureView)

    // Apply declared configs to CameraX using the same lifecycle owner
    CameraX.bindToLifecycle(viewLifecycleOwner, preview)
}
```

✚ Further Aspects

- https://developer.android.com/training/camerax