

Developer Guide for the ACE Platform

Disclaimer:

This document is a work in progress and subject to change. The information included may be updated, revised, or modified as new details become available. Please refer to the latest version for the most accurate and up-to-date content.

Confidentiality:

This document is shared under a Mutual NDA with specific 3rd parties.
Proprietary - Company Confidential Copyright ©2024 Teledyne FLIR LLC All Rights Reserved. Information and equipment described herein may require US Government authorization for export purposes. Diversion contrary to US law is prohibited.

Document Change Log:

Version	Where in document	Change description	Date of release
V09	2.9	Added network back-off guideline	2025-11-17
V08	2.1	Adjusted FLIR Atlas SDK section to more clear structure and text, additionally updated links	2025-10-03
V07	2.9.2	Updated SBOM instruction	2025-09-25
V06	2.10	Minor updates	2025-09-22
V05	2.6	Instructions on using Android WebView	2025-09-10
V04	2.10	Updated character limit	2025-04-25
v03	2.9.2	Document and share SBOM in the submission	2025-04-03
v02	2.9.1	How to shrink the APK size	2025-03-18
v01	2.9.1	Maximum APK size when submitting the app	2025-03-12

- [1. Introduction](#)
 - [1.1. Not supported Android functions](#)
 - [1.2. Datasheet](#)
- [2. Getting started](#)
 - [2.1. Flir Atlas SDK](#)
 - [2.2. Emulator](#)
 - [2.3. Enabling Developer Options](#)
 - [2.4. Notifications](#)
 - [2.5. Permission Requirement](#)
 - [2.6. Browser](#)
 - [2.7. Library](#)
 - [2.8. Android files](#)
 - [2.9. Network request best practices](#)
 - [2.10. Submitting your app to FLIR for review and release](#)
 - [2.11. Apps](#)
 - [2.12. Camera stream](#)
 - [2.13. Hardware Buttons](#)
 - [2.14. Design principles for UX/UI](#)
 - [2.15. App icons](#)

1. Introduction

This guide provides the foundational information required to develop and integrate applications with the ACE (Android Camera Ecosystem) platform. For additional details or technical support, please contact our developer support team.

1.1. Not supported Android functions

Function	Status
1.1.1.1. Android Settings app	Android Settings is replaced with FLIR Settings
1.1.1.2. Quick Settings	The Android quick settings and status bar are disabled. They are disabled to avoid users from accessing Android system settings.
1.1.1.3. Android Notification GUI	The Android notification GUI is disabled. This is to avoid users from having access to Android system settings. Read more about Notifications below.
1.1.1.4. Web browser	There is no standard web browser. We will compile and deploy our own Chromium to be embedded in some of our applications. Read more about Browser below.
1.1.1.5. Task switch	The Android task switcher is removed. The user will switch between apps using the Launcher app or by short cuts at specific locations in the apps.

1.1.1.6. System bar	The Android system bar is removed. It is removed to prevent giving the user access to parts of Android we do not want to expose. A user could potentially access Android features that would prevent out apps to work as expected.
1.1.1.7. Virtual system buttons	The Android virtual system buttons are removed. All inter-app navigation should use the physical buttons and the touch screen.

1.2. Datasheet

Feature	ixx
Thermal camera	
Thermal resolution	240 x 320, 480 x 640
Image frequency	30fps
Thermal focus method	Fixed
Thermal minimum focus distance	0.5m
Exchangeable optics	No
IR FoV	37° x 49°
NETD	50mK
Max scene temperature	450 °C, 550 °C
Measurement accuracy	±2°C
Visual camera	
Visual camera resolution	8MP
Visual camera FOV	51° x 65°
Visual camera autofocus	Yes
Sensors and peripherals	
LED	Yes
Laser Pointer	Yes
Laser distance meter	No
Microphone/speaker	No
GPS/GNSS	Yes
Compass	Yes
Accelerometer	Yes
Gyro	Yes
Connectivity	
Cellular connectivity	Yes (physical SIM slot)
Radio connectivity	WiFi 5 / BT 5.0
Electronics	
Viewfinder	No
Processor	Qualcomm QCS6490
RAM	LPDDR5 8GB
Flash	eMMC 128GB
Memory	Only internal
External connectors	USB-C
Screen size	5" wide/portrait
Screen resolution	720 x 1280
Power system	
Exchangeable batteries	Exchangeable
Charging time	2.5h 90%
Software	
Operating system	Android 14
Customized apps & OTA	Yes

Streaming capabilities	Screen sharing over USB/Display Port
------------------------	--------------------------------------

2. Getting started

2.1. Flir Atlas SDK

Download the FLIR Atlas Android SDK deliverables from the [Flir ACE - Developer Hub](#).

Items available for download:

- SDK deliverables
 - two AAR library files
 - SDK Javadoc documentation including "ACE User Guide" - a useful HTML guide for getting started with the SDK on ACE platform
- Android SDK samples including AceCamera SDK sample application - simple application showing and explaining a basic use case for live streaming from camera to a GLSurfaceView
- ACE emulator CSQ file (ace_emulator_04.csq) - which allows users to simulate some of the features of a real camera when put in the appropriate place in app-private space (more details in "ACE User Guide")

2.2. Emulator

Use a standard emulator, without [GMS](#) (Google Mobile Services).

2.3. Enabling Developer Options

To enable Developer Options on the ACE device:

1. Open the **Settings** application.
2. Navigate to **About Device** -> **Camera Information**.
3. Tap **Build Number** seven times.

2.4. Notifications

The ACE platform features a custom Notifications application that operates differently from the standard Android notifications system. To utilise this functionality a permission requirement is needed.

2.5. Permission Requirement

Add the following permission to your manifest and ensure it is granted:

```
<uses-permission android:name="com.flir.notifications.permission.POST_NOTIFICATIONS" />
```

Note: Without this permission, notifications can be sent but will not be displayed. This applies to both internal and third-party applications.

2.5.1. Required Notification Parameters

Due to the unique notification handling on the ACE platform, additional metadata must be provided using a `Bundle`. The following fields are required:

- **GRAVITY**: Determines the position of the notification. Options: **CENTER**, **TOP**, **FILLED**.
- **BUTTON_TEXT**: Text for the dismiss button.
- **SECONDARY_BUTTON_TEXT**: Text for the secondary button that redirects the user.
- **REDIRECT_PACKAGE_NAME**: The package name for the redirection target.
- **CLOSE_OPTION**: Indicates if the notification should include a dismiss option.

Example

```
Bundle().apply {
    putString("BUTTON_TEXT", "Close")
    putString("SECONDARY_BUTTON_TEXT", "Redirect")
    putString("REDIRECT_PACKAGE_NAME", "com.my.package")
    putInt("GRAVITY", Gravity.TOP)
    putBoolean("CLOSE_OPTION", true)
}
```

2.5.2. NotificationManager

Notifications are displayed using the standard Android `NotificationManager` API. As long as the required permission is granted and the metadata `Bundle` is included, notifications will appear as expected.

Example

```
notificationManager.notify(  
    0x1,  
    Notification.Builder(context, "my-channel")  
        .setContentTitle("Header")  
        .setContentText("Subtitle")  
        .setSmallIcon(android.R.drawable.ic_dialog_info)  
        .setVisibility(Notification.VISIBILITY_PUBLIC)  
        .addExtras(  
            Bundle().apply {  
                putInt("GRAVITY", Gravity.TOP)  
            }  
        )  
        .build()  
)
```

2.6. Browser

The ACE platform includes a custom Chromium-based browser as the default web rendering engine. While this browser does not appear in the launcher, it is used for rendering web applications and other web-based functionalities. Developers shall use the Android SDK `WebView` for similar purposes.

To maintain a secure environment on our camera device, external browser access is not permitted from within third-party apps. This includes any functionality that opens a browser outside the app context (e.g., Chrome).

What you must do:

- Use Android `WebView` to display web content within your app.
- `WebView` provides better control and reduces the risk of exposing users to unsafe external content.

Examples of disallowed behavior:

- Using `Intent.ACTION_VIEW` to open URLs in an external browser.
- Any link or button that launches a browser outside the app.

2.7. Library

The Library application on the ACE platform has the following restrictions:

- Only files located in the `**DCIM**` directory are displayed.
- Supported file formats are `**JPEG**` and `**MP4**`. All other formats are filtered out.

2.8. Android files

The Library app acts as a media browser for media items specifically captured using the camera itself. While the native Android File browser can be used to traverse the entire file system and display all files on the device. The Library app is limited to the root DCIM folder and only allows traversing sub-folders and showing media captured by the device.

2.9. Network request best practices

To prevent network congestion and "data storms," all third-party apps must:

- Use Android's [WorkManager](#), [JobScheduler](#), or similar frameworks for background network tasks. These frameworks ensure tasks are scheduled efficiently and reliably, even if the app is closed or the device restarts.
- Always implement a back-off strategy (such as exponential back-off) when retrying failed network requests, to avoid repeated immediate retries that can overload the device and cloud services.
- Avoid immediate or repeated retries without delay.

This ensures reliable operation for both the device and connected cloud services.

2.10. Submitting your app to FLIR for review and release

The "release candidate" for the application (APK file and release notes) is submitted to FLIR via a specific ACE Developer Hub page (a non-public page), and the link is shared with one developer per third-party company (one email address).

The application will undergo a 'smoke test,' meaning it will be uploaded to Apps+ and opened on the iXX camera to validate that it runs. The only functionality tested will be the camera stream. Allow a few weeks to get a reply from Teledyne FLIR.

Test results will be communicated via the Customer Support system. If multiple development releases are required, FLIR will provide support based on agreement.

Once the application is ready and approved for release by both the third-party developer and FLIR, a release date will be agreed upon, and FLIR will publish it on ACE Apps+

2.10.1. Recommended APK size

To ensure optimal performance and compatibility with the ACE platform, the submitted APK file must not exceed **250mb**.

Exceeding this size limit will result in rejection during the review process.

Ensure that your app follows below configuration to shrink your APK size.

Example

```
/**
 * Static configuration for packaging and ABI splitting.
 *
 * This configuration:
 * 1. Sets useLegacyPackaging to true - keeps the traditional APK structure which
 *    may result in more compatibility with older Android versions
 * 2. Enables ABI splitting with only arm64-v8a architecture - produces smaller APKs
 *    by including only 64-bit ARM binary code (most modern Android devices)
 */
android {
    // Configure packaging options with static true value for useLegacyPackaging
    packagingOptions {
        jniLibs {
            // Always use legacy packaging regardless of project properties
            useLegacyPackaging = true
        }
        dex {
            // Always use legacy packaging regardless of project properties
            useLegacyPackaging = true
        }
    }

    // Configure ABI splitting with static values
    splits {
        abi {
            // Always enable ABI splitting
            enable true
            reset()
            // Only include arm64-v8a architecture
            include 'arm64-v8a'
            // Do not create a universal APK containing all ABIs
            universalApk false
        }
    }
}
```

2.10.2. Document and share SBOM in the submission

You shall identify and document app components by drawing up a software bill of materials (SBOM).

An SBOM should provide a comprehensive inventory of all direct and transitive dependencies used in building the final app. This includes:

- List of all open-source and commercial/proprietary components.
- For each component: version and licensing information.
- Relationships between components.

SBOM format

- Preferred format for SBOM is CycloneDX, as it is widely supported and integrates well with Gradle via an available plugin.
- The SBOM must be provided in JSON format (sbom.json)

 *Note: Submissions missing the SBOM in CycloneDX JSON format may be considered incomplete*

An example is to add following in build.gradle:

Example

```
plugins {  
    id 'org.cyclonedx.bom' version '2.2.0'  
}
```

And then to generate sbom.json:
./gradlew cyclonedxBom

Download example SBOM file via the developer hub on [ACE Developer HUB | Teledyne FLIR](#).

2.11. Apps

The ACE platform features a dedicated Apps+ application for distributing system and third-party applications. To publish your application to the ACE Apps, submit the following details along with your APK:

Application Description

Requirement	Detail
Supported languages	-Obligatory: English (Locale: en)
	-Optional: The rest of the supported languages
	ar, cs, da, de, el, es, fi, fr, hu, it, ja, ko, nb, nl, pl, pt, ru, sv, tr, zh-CN, zh-TW.
Character Limit	Maximum 4000 characters for all languages.
Prohibited Content	-Avoid misleading claims, inappropriate language, and irrelevant keywords.

Company Name

Requirement	Detail
Character limit	3-50 Characters
Allowed Character	Alphabets, numbers, &, -, .

Application Name

Requirement	Detail
Character limit	3-50 Characters
Allowed Character	Alphabets, numbers, &, -, .

Release Notes

Requirement	Detail
Supported languages	-Obligatory: English (Locale: en)
	-Optional: The rest of the supported languages
	ar, cs, da, de, el, es, fi, fr, hu, it, ja, ko, nb, nl, pl, pt, ru, sv, tr, zh-CN, zh-TW.
Character Limit	20-500 Characters
Clarity	Explain changes clearly and concisely

Compatibility

Requirement	Detail
Model	List of camera models your app supports.

Example of how this file could look like.:

Example

```
# Application Name
IR Image Analyzer

# Company Name
VisionTech Solutions

# Application Description
## English
Analyze IR camera images with precision and ease. Detect heat patterns, measure temperature ranges, and
export results effortlessly.
## Spanish
Analiza imágenes de cámaras IR con precisión y facilidad. Detecta patrones de calor, mide rangos de
temperatura y exporta resultados fácilmente.

# Compatibility
## Supported devices
-Camera model name.

# Release Notes
## English
Version 1.2:
- Added support for exporting thermal data in CSV format.
- Enhanced image processing speed for high-resolution IR images.
- Fixed a bug that caused crashes on some Android devices.## Spanish
Versión 1.2:
- Se agregó soporte para exportar datos térmicos en formato CSV.
- Se mejoró la velocidad de procesamiento de imágenes de alta resolución IR.
- Se solucionó un error que provocaba fallos en algunos dispositivos Android.
```

2.12. Camera stream

Details for how to connect to a camera stream please see the section "[Flir Atlas SDK](#)".

2.13. Hardware Buttons

ACE devices use hardware buttons for navigation. It is essential that your application supports these buttons to ensure a seamless user experience.

Physical button	Kotlin event name	Typical action	lxx
Home	Key.F3	Go to Launcher app	x
Back	Key.Back	Step back in app or back to Launcher	x
Select	Key.DirectionCenter	Select focused item	x
Trigger	Key.F1	Save image/start recording	x
Up	Key.DirectionUp	Move focus (or focused item) up	x
Down	Key.DirectionDown	Move focus (or focused item) down	x
Left	Key.DirectionLeft	Move focus (or focused item) left	x
Right	Key.DirectionRight	Move focus (or focused item) right	x
Laser	Key.F4	Light the laser pointer	x
Power	N/A	Turn camera on/off	x



2.14. Design principles for UX/UI

Camera-first approach

Our fundamental principle is to recognize that ACE serves as a camera platform. With this in mind, every user journey should begin with the understanding that users will primarily capture infrared (IR) images. Prioritizing the camera experience ensures that our users' needs are met effectively.

Consistent UX Writing

Consistency in terminology matters. Features and actions within the camera should use uniform language across our suite of digital products. Whether it's "upload" or "transfer," users should seamlessly navigate our ecosystem and intuitively understand what to expect.

Seamless app transitions (breadcrumbs)

Navigating between apps should be straightforward and intuitive. Our goal is for users never to feel lost when switching between different apps on the platform. Utilizing the physical back button on the camera should ideally return users to their previous context. If that's not feasible, clear guidance back to the launcher screen ensures a smooth transition.

2.15. App icons

Providing icons

Launcher icons are provided to the Apps+ store, with the following requirements:

- Icon drawables in a PNG or vector format.
- Background and foreground layers without mask or background shadows.
- Consider that the platform supports light and dark mode.
- App name does not need to be included in the icon.
- Avoid text in the icon.

PNG images should have a size of minimum 512x512 px or dp.

For inspiration on how to design icons for Android devices, take a look at the [Android Material guidelines](#).

