esper

Esper Guide

Considerations for Building Android Apps for Company-Owned Devices

Get Started

Start here

Your app delivers your customer (or employee) experience, making it a crucial piece of your overall solution. Use this guide to understand the considerations of designing and building your perfect app for dedicated device use cases on Android hardware.

Click to jump to a specific slide or use the navigation bar to continue on.

Creating apps for the dedicated device world

Design considerations

Build considerations

Manage considerations

More resources

Creating apps for the dedicated device world

What makes designing an app in the dedicated device world different?

The app is the center point of your user experience. You're using this app as a core part of your business strategy, whether it's how you monetize with end customers or operationally-oriented.

This has implications across your business and brings up many factors to consider, such as:

Area	Question
Strategy	• How does your app fit into your business model? • How are you planning to monetize it?
Users	• What experience are you trying to deliver? • Who are your target users? Are they tech savvy? • Can users intuitively use your app?
Operations	• What tech stack is associated with your desired capabilities? • How will you deploy, manage, or update your app? • How will you track app health and performance?

Dedicated device problem scenario

Scenario
You are rolling your app out to the field and it's not working as intended. The Android app install model preserves your app cache when you roll forward, keeping all your login information and app data. But when you rollback, you're essentially uninstalling that current version and reverting to an old one, meaning your app cache is flushed.

Esper solution
Keep related user and app cache data in the cloud, and have a scheme for refreshing it back to your app when you do a rollback. Using this solution to successfully resolve this scenario requires upfront system design.

Tip! Some dedicated device scenarios aren't supported by the Android OS itself. With Esper Foundation you can seamlessly tackle them. Learn more here.

Design

Design

Click on a category to jump to the slide or use the navigation bar to continue on.

Hardware agnostic

- Platform support
- Operational aspects
- Telemetry
- UI
- Deployment mode: Kiosk
- Deployment mode: Multi-app

Hardware dependent

- Tech stack
- Distribution: Google Play Store
- Distribution: Google Play Store apps
- Distribution: Private apps
- Distribution: Web apps
- Managed config

Platform support

Define what platforms you need to support your app once it's deployed.

Tips

- **Focus on your use case.** Consider what type of experience you want to create for your end users and prototype it to ensure functionality. OSs are generally tailored to specific use cases or audiences. For example iOS is focused on prosumer use cases, while Linux is primarily designed for IoT.
- **Consider cross-platform, but be cautious.** Cross-platform can increase your efficiency, but you are dependent on the cross-platform framework (i.e. how they maintain it and what they offer) instead of being able to work with the app development capabilities offered for each OS.

Dedicated device problem scenario

Scenario
For your use case you're looking to payload on iOS, Windows, or Chrome in addition to Android).

Esper solution
Look into cross-platform technologies such as Xamarin, MAUI, and Fuchsia. Note these offer only the least common denominator feature set, so you can't use any special functionality of a particular OS.

Operational requirements

Determine how the app will operate when deployed to ensure capabilities are easily and appropriately accessible to users.

Tips

Consider these questions as you design your app.

- How do you plan to deploy, manage, or update your app?
- What common user scenarios are you going to run into?
- Do your deployment and distribution modes support a seamless user experience in these scenarios?
- What capabilities, both in your app and outside of your app, will users have access to?
- How are you planning to track app health and performance?

Dedicated device problem scenario

Scenario

Your app is running in kiosk mode, taking away the needed ability for users to interact with the Android settings. Without a scheme to make this functionality available to them, you're forcing technical engagement — possibly even a truck roll — to address the issue, which isn't economically viable.

Esper solution

Esper settings offer a way for users to interact with curated system settings, such as Wi-Fi access point, in a pinch. With the settings included, your users can access them with minimal involvement from your end.

You can add Esper settings to your config or later on when you provision your device.

Telemetry

Define the telemetry set you need to identify issues before they become problems and continuously improve the user experience.

Tips

- **First, determine what telemetry data you're looking for**. What do you want to see? Things like app crashes, how the user interacts with the device, etc.
- **Then, find the tech stack that will deliver what you want.** For example, Firebase is a free, economical option, if it serves your needs and you like how it delivers data, but it requires running on GMS. Crashlytics, on the other hand, is also available for AOSP. You can also pipe this telemetry and build it into your app itself with third-party services.
- **Your distribution method could impact what you're allowed to see.** For instance, if you're deploying a web app, you'll have less of an ability to capture telemetry as things like intent are cut off when using Chrome.

Dedicated device problem scenario

Scenario

You want to proactively resolve issues with quick and quiet updates before customers know something's wrong.

This requires harnessing the power of device data to take quick action. Your APK leverages telemetry to gain insight into product usage in the app experience, but the Android system itself is a data black hole.

Esper solution

Esper provides a base set of telemetry spanning system resources, and wireless and network connectivity you can monitor.

With Esper Foundation, you can go to exactly what you want to monitor. If there are system events that are useful to track and populate into your data lake, you can pipe in and stream them from Foundation. And if a telemetry set is missing, we can work with you to add it into our platform.

UI

Design the best UI for your app to deliver the perfect experience for your end users.

Tips

- **Don’t underestimate designing your UI.** Determining UI can be unexpectedly complex. Something as simple as deciding whether to include a navigation bar can surface many questions. If you include one, you’ll need to decide things like where it’s located, how your app will sit on it, or if you’re doing swipe gestures with it. If you don’t include it, what are your navigation elements?
- **AOSP often allows for UI modifications,** like the presence of a nav bar. Some OEMs provide SDKs that enable you to determine that behavior.

Dedicated device problem scenario

Scenario
You want to customize your app navigation bar.

Esper solution
With Esper Foundation we give you the ability to dynamically control your UI and create new UX experiences more easily. For example, with our Device SDK, your app can decide when to expose the navigation bar and what elements to make available to the user.

Deployment mode: Kiosk

Find the best way to deliver your use case to end-users while getting to market quickly.

Tips

- **Lockdown your device.** Kiosk mode is used when you need to lock down a device to a single app to prevent other uses. This isn't appropriate for use cases that need to have a home screen with different app choices.
- **Primarily used for consumer-facing scenarios,** whether a customer-facing kiosk or employee-facing with the cash register.
- **Android hardware agnostic.** Kiosk mode can be used with either GMS or AOSP hardware.

Dedicated device problem scenario

Scenario
Users are exploiting the device by accessing non-essential apps and/or downloading additional apps.

Esper solution
Through Esper and our launcher you can lock your device into kiosk mode, restricting usage to only approved apps and prevent access to the Google Play Store. You can lock your device to only one app on your homepage or have the app fill the screen.

Learn more about kiosk mode with Esper.

Deployment mode: Multi-app

Find the best way to deliver your use case to end-users while getting to market quickly.

Tips

- **Allows approved apps only.** Multi-app mode is when you lock your devices to only the apps approved by the admin. Users access all apps through a home screen.
- **Primarily used for employee-facing use cases** as you can enable utilities or additional abilities such as analyzing the Wi-Fi network at your deployment location.
- **Android hardware agnostic.** Kiosk mode can be used with either GMS or AOSP hardware.

Dedicated device problem scenario

Scenario
You want to provide users access to a specific set of apps selected by an admin.

Esper solution
Through Esper you can set which apps are available for user access. Only the shortcuts we expose and lock-in on the home screen with the launcher are made available to the end user.

You also get fine grain control of the APKs that ship in the image built for the device. Apps in ROM will be running in the background and you can choose to disable them.

Tech stack

Determine what tech stack you need as part of your implementation.

GMS

Offers a set of developer tech you can utilize, including apps, Play Store, Firebase, and developer services.

These are great tools to use, if you don't mind having less control over your app.

AOSP

You'll get more design control on AOSP, but it only has a subset of capabilities available on GMS.

Many apps built by ISVs have dependencies on the GMS dev tech stack so they won't run on AOSP. This includes enterprise apps, Google Play Store apps, and apps available from an ISV for private use. You'll have to work with the ISV to remove the dependencies and create a build for AOSP.

Dedicated device problem scenario

Scenario
You want to use an app available on the Google Play Store, but it's not compatible with your AOSP-based device.

Esper solution
Esper can work with ISVs to discuss your GMS app dependencies and the trade-offs of making an AOSP version with your end-customer experience in mind.

If you need to build a custom app, Esper provides easy-to-use tools, like our Android Studio and CI/CD pipelines, so you can create and roll out apps seamlessly.

Learn more about this in the Build section of this guide.

Distribution: Google Play Store

Learn how to best use the Google Play Store in the dedicate device space.

Tips

- **The consumer use case is the Play Store's main focus.** The vast majority of app development is centered around consumers wanting to payload via Google Play. Google is working to improve this process for app devs with things like taking ownership of the signing key for your app or doing app bundles (primarily for games).
- **The Play Store can only be used with GMS hardware.** Apps on the Google Play Store are created by GMS App Devs and ISVs. They are often only compatible with a GMS device. AOSP devices don't have access to the Google Play Store.

Dedicated device problem scenario

Scenario
You want to use app bundles for your dedicated use case.

App bundles allow you to do a small payload when the app installs and have it progressively download, enabling end users to have a great out of box experience instead of waiting for a full download.

Esper solution
Don't optimize for Google Play or app bundles if you're doing dedicated use cases.

Esper follows the previous Google Play Store conventions and is designed to work with release key signing. If you sign with your release key and refrain from app bundles, you'll be able to work with the Esper Cloud and other dedicated device app distribution methods.

Distribution: Play Store apps

Leverage the wide range of existing apps available on the Google Play Store for GMS devices.

GMS

Have the option to pool existing APKs on the Google Play Store. This is a great option if the apps can serve your business purpose.

Don’t typically see Google Play Store apps used in kiosk mode, but they can be a good option for multi-app mode.

AOSP

For AOSP, Esper takes care of the app distribution infrastructure as the Google Play Store doesn’t exist for AOSP.

Dedicated device problem scenario

Scenario
You want to give device users access to apps on the Google Play Store.

Esper solution
Esper offers access to the Managed Google Play Store, a special aspect of the Google Play Store meant for managed devices. With this you can control which apps from Google Play Store are available to your device fleet for download.

Distribution: Private apps

Private apps can give you more control over you app development and management. They are fully compatible with Managed Google Play.

GMS

Can use Managed Google Play to distribute your app in a private manner relative to your Esper endpoint or cloud instance.

AOSP

There's no native app cloud built into AOSP. You have the control to build and distribute your app through the Esper Cloud.

Dedicated device problem scenario

Scenario
You want to deploy a private enterprise app to your users' devices.

Esper solution
Managed Google Play Store gives you a private, secure app store for your organization. You can use it to deploy and update proprietary apps without user intervention. However, it isn't designed for dedicated devices. Dedicated fleets face edge conditions consumer devices don't and have a mix of device types like, POS, kiosks, or digital signage, that need nuanced support.

Esper offers you your own app deployment solution with tons of customization and powerful, code-driven extensibility. You can deploy directly to any of your devices whether in the lab or field.

Learn more about this in the Build section of this guide.

Distribution: Web apps

Web apps are a great way to deliver the experience you've already invested in and curated on your website through an app. They're quick to build, prototype, and deploy.

GMS

Can create web apps through Managed Google Play.

Web apps require Chrome to be shown on the home screen for the app to work. This isn't a problem in kiosk mode as the Chrome shortcut is hidden behind the kiosk mode app. However, in multi-app mode, Chrome will be available for use. You'll need to consider how to apply a managed config to control what users can do with it.

AOSP

Web apps built through Managed Google Play will only work on GMS devices. Through Esper, you have the ability to do rapid prototyping with web apps on AOSP that doesn't exist with GMS.

Dedicated device problem scenario

Scenario
You want to deliver an experience via a web app that is dependent on Chrome.

Esper solution
Esper can build web apps for our customers using Chromium that are independent of Chrome and can run on either AOSP or GMS.

Managed config

Managed config is a way to provide a data payload that will configure your app based on that payload. Example: Limiting app behavior such as what websites the Chrome app can go to.

GMS

With GMS, you're stuck with an atomic configuration. Any install for the app is going to utilize that managed config across your entire fleet. This isn't great for dedicated device scenarios as different customers often need different configurations.

AOSP

Managed config was conceived to be specific to GMS and Managed Google Play. Through Esper's genericized version you can access it for AOSP. For this, you will need to build your app to support manage config. This isn't hard to do, but it is an additional aspect to consider as you design your app.

Dedicated device problem scenario

Scenario
You want to build an app that works across GMS and AOSP.

Esper solution
Since you can't use the GMS dev tech stack in this scenario, Esper created the ability to support managed config across GMS and AOSP.

You get the tools to programmatically control your config and push out data payloads in a fine grained manner to apps running on your devices. With Esper you can group your devices and create a specific managed config for a certain set of devices, customers, etc.

Build

Build

Click on a category to jump to the slide or use the navigation bar to continue on.

Esper Plugin for Android Studio

Emulators

Device SDK

Android apps on x86 hardware

Esper Plugin for Android Studio

The Esper Plugin for Android Studio is an external third-party model that enables you to add capabilities to Android Studio.

Benefits

- Streamline the development flow from build to test lab to deployment with a code-driven approach
- Easily install apps on your device fleet

Capabilities

Upload APKs directly from Android Studio to an Esper Endpoint. Once uploaded you can use the Esper APIs, SDK, or CLI from Android Studio Terminal to provision devices with the APK for testing or deployment purposes.

Trigger pipelines for automated test lab and canary deployment

Emulators

Esper's Android Virtual Device is an emulator, which helps developers test their apps by defining characteristics of a device to simulate real device capabilities.

Benefits

- Reduce device shipping cost by testing hardware virtually
- Grow your ISV developer base by providing custom device emulators to kickstart app development instantaneously
- Test app-device compatibility right from the development phase using a custom emulator

Capabilities

Almost all Android Studio emulators are centered on GMS. To help, Esper built emulators for AOSP and Esper Foundation. With them, you can:

- Deploy applications to emulators from the cloud emulating application deployment on virtual devices.
- Track device performance and behavior easily using unique device IDs for every virtual device.
- Emulate peripherals such as printers, card readers, and barcode scanners.

For larger customers, Esper helps create custom emulators for their specific hardware to enable large application development teams and specialized ISVs that want to target their bespoke hardware.

Device SDK

Esper provides a Python client library to communicate with the Esper APIs to programmatically control and monitor your Android dedicated devices.

Benefits

- Easily perform privileged operations on managed devices
- Develop apps which need to perform seamless operations on or retrieve vital information from a device
- Turbocharge your application and enhance the types of experiences you can create

Capabilities

Access information most APKs can't, such as device serial number. Use the Device SDK on Esper-provisioned devices to fetch the device serial number, enabling 1:1 coordination between your Cloud and your APK on every device in your fleet. These capabilities are available on any AOSP or GMS device, and of course are built into Esper Foundation.

Fine-tune with Esper Foundation, such as fully customizing the behavior of the Android navigation bar, the permissions model, and system notifications. You aren't stuck with what your device maker has decided to do, or the decisions made by the vendor of the BSP they've chosen.

Android Apps on x86 hardware

Running your Android app on x86-based devices? Here's how to make sure your app is compatible.

ABI requirements

When you build an Android app, you create an application binary interface (ABI) which is specific to the underlying silicon.

To run an app on x86 it will also need an x86 ABI. Both ABIs are built by default if you're using the Android Studio.

Using third-party apps

Some APKs from third-parties won't work on x86 as they weren't built with the x86 ABI.

The simple solution is to have the app dev modify the build to include the x86 ABI. You can use an emulator, but these are slower and modifying the build is very easy to do. This won't be a big deal for dedicated use cases, as you're most likely not using Play Store apps and technically GMS isn't available on x86-based Android.

Building custom apps

If you're building your own app, make sure your developer understands you're running the app on x86 (AOSP) hardware.

You can also give them test hardware that includes an emulator running AOSP or Esper Foundation in the Android Studio with the precise target.

Manage

Manage

Click on a category to jump to the slide or use the navigation bar to continue on.

Bug reports

Secure ADB

Bug reports

Seamlessly service and interact with the devices in the field that are running your app.

Benefits

- **Reduce OpEx.** Dedicated devices are often in hard to reach places and end-users aren't always tech savvy. Reliable access to bug reports can save you from costly and unnecessary truck rolls.
- **Debug your fleet in real time.** Esper facilitates getting your Logcat for you through our infrastructure and with Esper Foundation on AOSP, it's seamless. You simply send a request for a bug report to Esper. It's then generated and made available for download through the Esper console.

Dedicated device problem scenario

Scenario
Obtaining your Logcat when in kiosk mode

Esper solution
You can capture your Logcat on a GMS or AOSP device, but sending it requires specific user permissions.

This can be tricky if you're using kiosk mode as it's common in this mode to disable the notification bar. This means you won't be able to respond to the notification. Through Esper you can remotely (or locally if you've set it up this way) take the app out of kiosk mode to respond to the notification and send the bug report.

Secure ADB

Resolve customer or field device issues remotely with Android Debug Bridge (ADB)

Tips

- **Devices may need setup before running ADB sessions.** Setup typically doesn't survive reboot so you can't configure it at the factory and have it survive in the field. You could prepare a subset of devices when they're being shipped so they're readily available for ADB sessions or have a field tech set the device for the ADB session.

  If you have to do a truck roll, you can set up Secure Remote ADB so a developer can debug the device remotely. With Esper Foundation, we offer seamless Secure Remote ADB which enables you fire up an ADB session whenever you want.

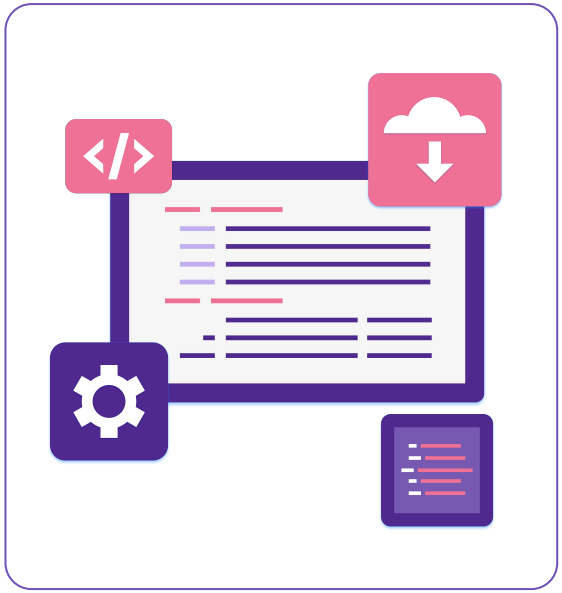
Dedicated device problem scenario

Scenario
Running remote ADB in the field securely

Esper solution
Esper's Secure Remote ADB is the same tool as Google's Android ADB, but instead of requiring a physical or local network connection, it can be used from anywhere in the world through an encrypted tunnel to the remote device.

Our ADB is designed to limit the security risks associated with exposing ADB. If you enable ADB for a USB port on your device, you can locally jack in and run a session. This works great for the local tech, but can leave your device exposed in the wild. Esper enables you to control your ADB session, turning it off or setting a session duration in case you forget to.

Need more help?



Learn more about Esper

Esper can help you automate your Android app development and device management for the cloud age. Our full-stack cloud developer tools help you build, deploy, and debug cloud-connected apps running on Android devices.

Learn more about how we can help you unlock a smarter approach to Android app development, visit our website or connect with an in-house expert.