APPFORUM 2019 | ELEVATING ENTERPRISE INTELLIGENCE

Gain an intelligent edge in Enterprise IT and software development

SAVE THE DATES

EUROPE APPFORUM 2019 | ELEVATING ENTERPRISE INTELLIGENCE

ZEBRA CAPTURE YOUR EDGE

4-6 JUNE, 2019 | WARSAW

ASIA-PACIFIC APPFORUM 2019 | ELEVATING ENTERPRISE INTELLIGENCE

ZEBRA CAPTURE YOUR EDGE

AUGUST 13-14, 2019 | SYDNEY

ASIA-PACIFIC APPFORUM 2019 | ELEVATING ENTERPRISE INTELLIGENCE

ZEBRA CAPTURE YOUR EDGE

AUGUST 20-21, 2019 | SINGAPORE

AMERICAS APPFORUM 2019 | ELEVATING ENTERPRISE INTELLIGENCE

ZEBRA CAPTURE YOUR EDGE

OCTOBER 1-2, 2019 | LAS VEGAS

DEVTALK - What’s New for Android ‘O’ and the impact on Zebra developers

Darryn Campbell, Software Architect, Zebra Technologies

17th October 2018

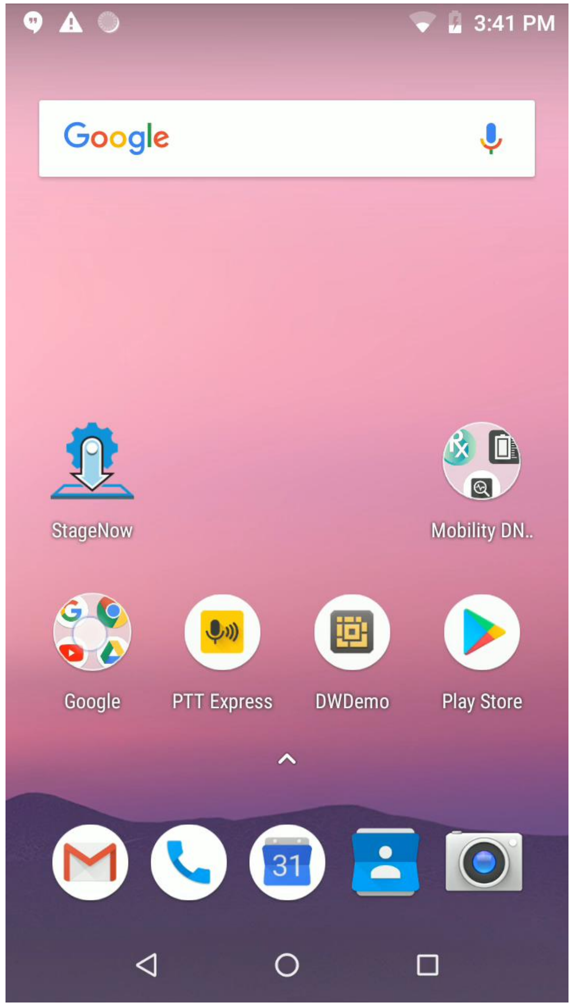
Agenda

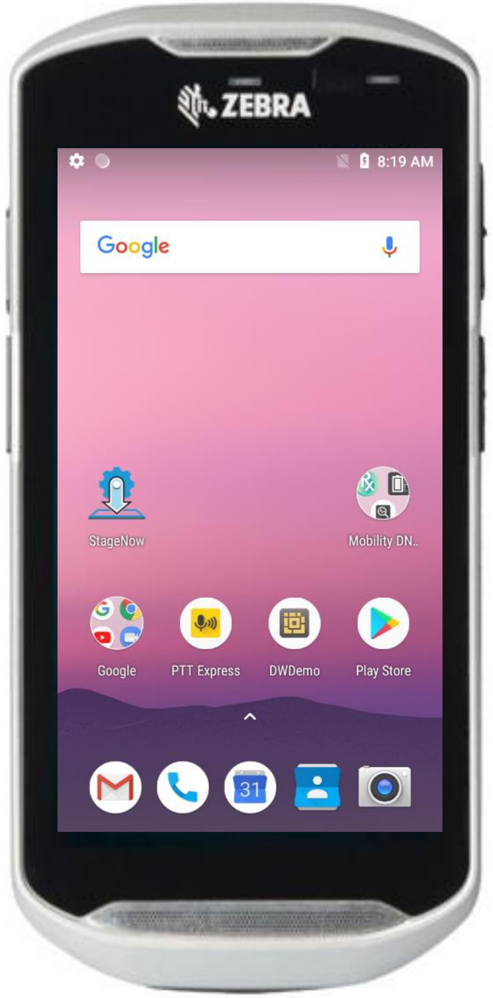
- **Brief overview of GMS Restricted**
  - A feature exclusive to Zebra devices introduced in Android Oreo for SD660 based devices
- **Changes in Android Oreo affecting enterprise Developers**

What is GMS Restricted?

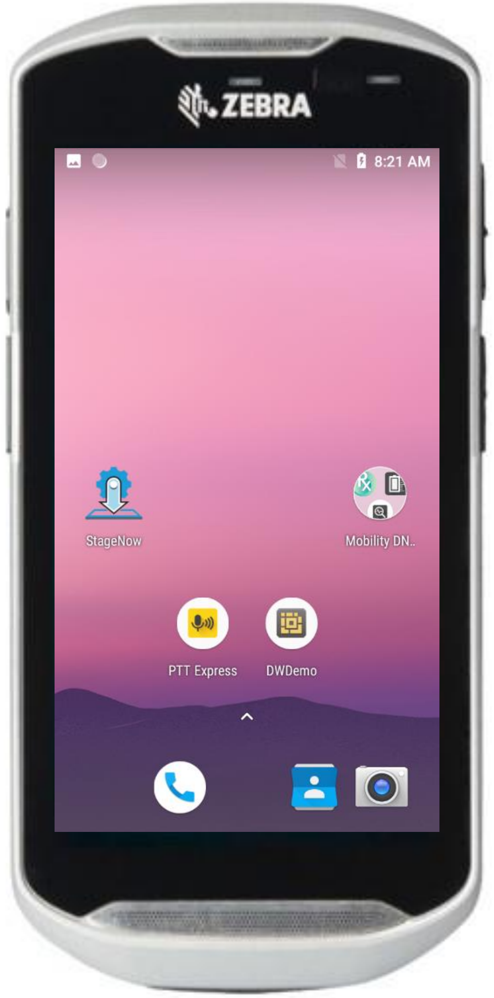
- Restricted? How?:
  - “Restricts” the capabilities of the device, no GMS apps available
  - “Restricts” the device from communicating with Google, potentially enhancing privacy
- All GMS Applications disabled (e.g. Play Store, Chrome, GMail, Maps etc)
  - Alternative / 3rd party applications used.
  - AOSP equivalent app for keyboard automatically switched to.
- All GMS services disabled
  - Automatic opt-out of Google analytics data collection & location services
  - Doze mode is disabled (to match AOSP behavior)
- NO data leaves the device from GMS apps & services or the platform

Demo: Putting a device into a Restricted State





Before



After

ZEBRA

Changes in Oreo affecting enterprise developers

Overview – Trends over time

Running in the background	Job Scheduler	Doze mode	Doze "on the go"	Background restrictions	Machine learning for intelligent restrictions
Notifications	Quick settings & notification shade	Long press to access options	Direct reply & bundled notifications	Notification channels & snooze	Enhanced messaging experience
One or Two other major changes affecting Enterprise	Material design	Runtime permissions	Multi-window	Changes to the Google Play Store policies	Non-SDK methods actively discouraged
Android Enterprise features	Android for Work, app restrictions	DO mode, lock task mode, managed configs	DPM API enhancements	DPM API enhancements	DPM API enhancements

Overview

Where to get more information

- Google publish documentation in each new release
  - Lollipop, Marshmallow, Nougat, Oreo, Pie
    - Includes samples, behaviour changes, API changes & other pertinent info
    - Blogs such as this one published yesterday on background execution
  - Android Enterprise changes for:
    - Nougat, Oreo, Pie. Mostly aimed at EMM partners but good to understand Android capabilities
- Zebra publish documentation in each new release
  - Marshmallow, Nougat, Oreo
    - Think of this as “Reading the Google documentation with an Enterprise mindset”
- Previous DEVTALK discussing changes to Marshmallow & Nougat

ZEBRA

Overview

Google's highlighted features

- Picture-in-picture
- Notification channels
- Autosizing TextView
- Adaptive icons
- Wide-gamut color
- Java 8 language APIs
- Multidisplay support
- Android Oreo (Go edition)
- Notification dots
- Autofill framework
- Downloadable fonts
- Shortcut pinning
- WebView features
- Media features
- Neural Networks API

Oreo Background Limitations

Overview

- Oreo is introducing restrictions on what an application can do in the background
- Three main types of restriction:
  - Receiving implicit broadcast intents declared in the manifest
  - Running services in the background
  - Update frequency from location APIs

Google conflates these two

- Continues the trend of restricting what an app can do in the background.
  - Trend continues into P which will limit access to user input & sensor data
- Developers are advised to work with the changes where possible. **Where not possible, make us aware**.

ZEBRA

Oreo Background Limitations

Receiving implicit broadcast intents declared in the manifest

The limitation:

- Applications cannot receive implicit broadcast intents which they have declared in their manifest

Oreo Background Limitations

Receiving implicit broadcast intents declared in the manifest

- What are implicit intents?
  - An implicit intent is an intent which lacks a package or component class name

Implicit intent	Explicit intent:

`Uri uri = Uri.parse("geo:0,0?q=London");`
`Intent intent = new Intent(Intent.ACTION_VIEW, uri);`
	`Uri uri = Uri.parse("geo:0,0?q=London");`
	`Intent intent = new Intent(Intent.ACTION_VIEW, uri);`
	`intent.setPackage("com.google.android.apps.maps");`

- What are broadcast intents?
  - An broadcast intent is received by a broadcast receiver and sent via the sendBroadcast() API

- Can I have an explicit broadcast intent? Isn't that a contradiction?
  - You **CAN** have an explicit broadcast intent, it will only be received by the destination component.

Oreo Background Limitations

Receiving implicit broadcast intents declared in the manifest

- What does “declared in the manifest” mean?
  - You can register your broadcast receiver dynamically at runtime or in the manifest at build time

Manifest

```
<receiver android:name=".WifiReceiver" >
   <intent-filter>
     <action
android:name="android.net.wifi.WIFI_STATE_CHANGED"
/>
   </intent-filter>
</receiver>
```

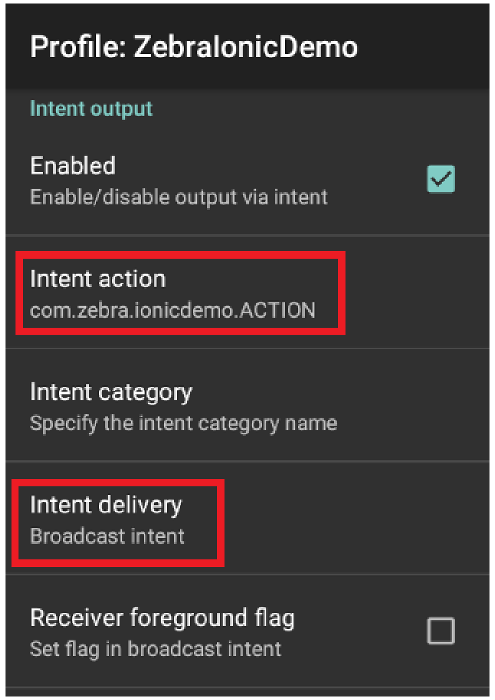
Dynamic registration:

```
BroadcastReceiver br = new MyBroadcastReceiver();
IntentFilter filter = new IntentFilter();
Filter.addAction(“android.net.wifi.WIFI_STATE_CHANG
ED”);
registerReceiver(br, filter);
//  Take care, if your application is in the
background it is subject to being killed by the
system
```

Oreo Background Limitations

Receiving implicit broadcast intents declared in the manifest

- Enterprise implications:


Profile: ZebraIonicDemo

Intent output

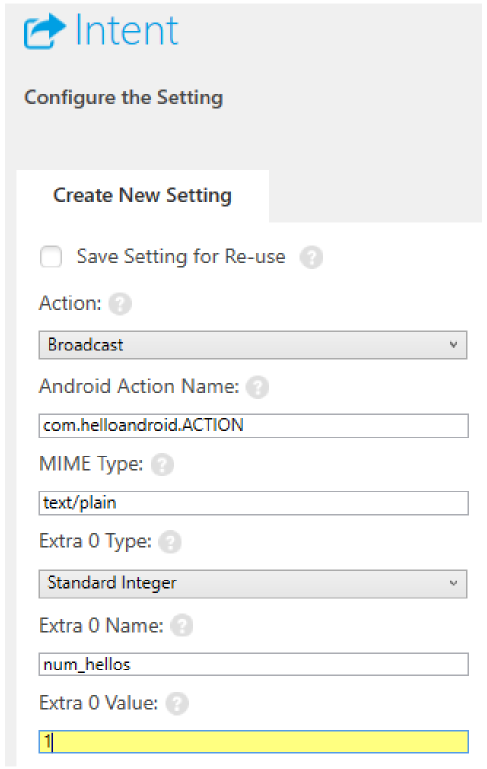
Enabled
Enable/disable output via intent ✓

Intent action
com.zebra.ionicdemo.ACTION

Intent category
Specify the intent category name

Intent delivery
Broadcast intent

Receiver foreground flag
Set flag in broadcast intent ☐


Intent

Configure the Setting

Create New Setting

☐ Save Setting for Re-use

Action: Broadcast

Android Action Name: com.helloandroid.ACTION

MIME Type: text/plain

Extra 0 Type: Standard Integer

Extra 0 Name: num_hellos

Extra 0 Value: 1

Datawedge can only send implicit intents

Intent CSP can only send implicit intents

Oreo Background Limitations

Receiving implicit broadcast intents declared in the manifest

- Mitigation:
    1. Use a dynamic broadcast receiver
    2. Switch to an explicit intent (if you have control over the sender)
    3. Continue to target your application at API level 25 or lower
    4. Per Google, “Use a scheduled job to check for the condition that would have triggered the implicit broadcast”

Oreo Background Limitations

Running services in the background

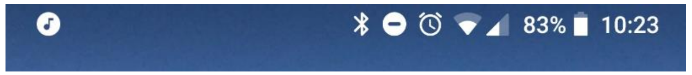
The limitation:

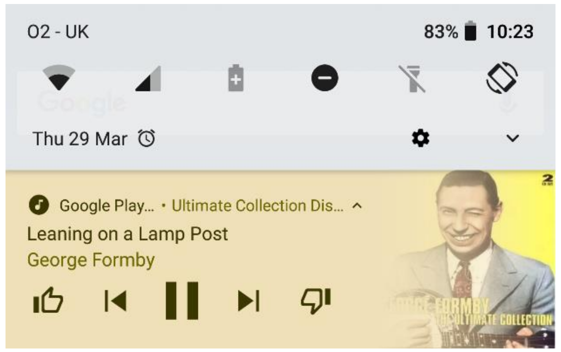
- Background applications built with API level 26 or higher and Android services associated with those background applications are subject to several limitations under Oreo to improve battery life and RAM usage. This includes IntentServices and PendingIntent services running in the background. Foreground applications are unaffected by these restrictions.
- “When the app goes into the background, it has a window of ***several minutes*** [emphasis added] in which it is still allowed to create and use services. At the end of that window, the app is considered to be idle. At this time, the system stops the app’s background services, just as if the app had called the services’ Service.stopSelf() methods.”

Oreo Background Limitations

Running services in the background

- What is a service?
  - From the docs: “A service is an Android application component that can perform long-running operations in the background, and it doesn’t provide a UI.” (think that definition might need updating!)
- Foreground service? Background service?
  - Most common use of a foreground service is the music player or GPS directions. A persistent notification is shown to the user, possibly with a rich UI.



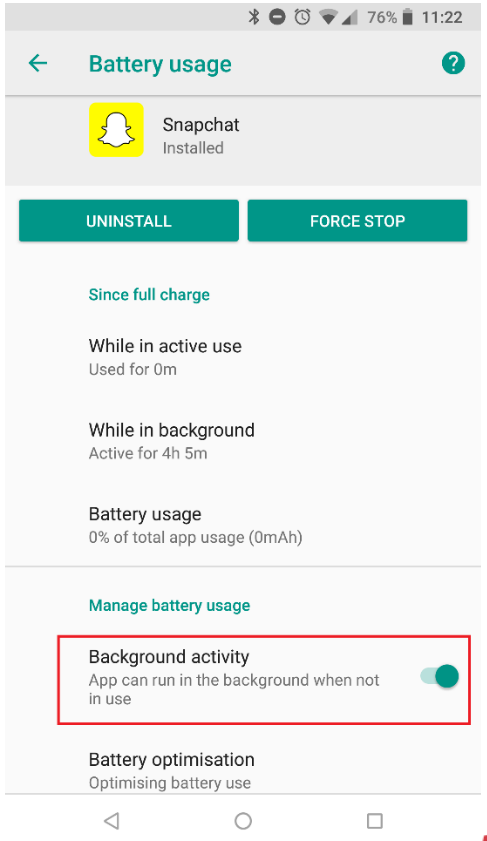


Music player notification before and after expansion

Oreo Background Limitations

Running services in the background

- "Built with 26 or higher"… so I can just target my application to API 25 and call it a day?
  - Not quite, apps targeting API 25 or lower and using a background service on a real device will present an option to the user for "Background activity" (*see screenshot, right*)
  - Setting is located under the battery options, under app info.
  - Default is 'enabled', i.e. not subject to Oreo background restrictions but the user can 'disable' it and the app WILL be subject to background restrictions.



Snapchat
Installed

Since full charge

While in active use
Used for 0m

While in background
Active for 4h 5m

Battery usage
0% of total app usage (0mAh)

Manage battery usage

Background activity
App can run in the background when not in use

Battery optimisation
Optimising battery use

Oreo Background Limitations

Running services in the background

More about the limitation:

- A system whitelist exists whereby applications are permitted to run and start services without limitation. Applications will be temporarily added to the whitelist "for several minutes" in order to handle common background tasks, such as:
  - Handling a high-priority Firebase Cloud Messaging (FCM) message.
  - Receiving a broadcast SMS or MMS message.
  - Executing a PendingIntent from a notification.
- This list is not exhaustive, Google qualifies the list as "when [an app] handles a task that is visible to the user," so there is room for expansion in future versions.

Oreo Background Limitations

Running services in the background

- Whitelist? Like Doze mode right? So, we have MX APIs to turn that off?
  - No, although the terminology is the same **it is not the same whitelist as used by Doze mode** and cannot be controlled by MX.

Oreo Background Limitations

Running services in the background

Enterprise implications:

- Many of the same enterprise apps affected by doze mode will also be affected by Oreo's limitation on background processes:
  - Running an on-premises push messaging system (not relying on FCM)
  - Downloading or uploading large files
  - Running a continual background service to check for network traffic
- Zebra value-added applications are also affected:
  - Datawedge's ability to call startService is curtailed since services cannot be started in the background
  - EHS' feature to launch a list of specified services is similarly curtailed

Oreo Background Limitations

Running services in the background

- Testing:

```
//  runs apps in background
adb shell am make-uid-idle <package>
//  force background limitations
adb shell cmd appops set <package> RUN_IN_BACKGROUND deny
//  returns app to normal behaviour (based on target SDK)
adb shell cmd appops set <package> RUN_IN_BACKGROUND allow
```

Oreo Background Limitations

Running services in the background

- Mitigation:
  1. Use the Android JobScheduler API to schedule a job to perform the task
  2. Make use of the temporary whitelist
     - Especially if the app is already using FCM or running a background task after receiving a PendingIntent (e.g. in response to a notification)
  3. Use a foreground service
  4. Per Google “Defer the background work until the application is naturally in the foreground”
  5. Continue to target your application with API level 25 or below
     - Not a long term solution. You will quickly run afoul of the new Play Store rules to force applications to move to “recent API levels”, from late 2018.
     - Can also be circumvented by the user if they have access to the application battery settings.

Oreo Background Limitations

Update frequency from location APIs

The limitation:

- Any background application or service making use of Android location APIs will only receive location updates “a few times each hour.”
- Unlike the previous two noted limitations, this limitation exists for any application running on an Oreo device regardless of target API level, and is not affected by the Background activity option.

Oreo Background Limitations

Update frequency from location APIs

There is not a single location API family on Android, these are the **affected** APIs:

- Fused Location Provider Client which replaced the Fused Location Provider API. Both are affected.
- Location Manager, any available Google location API to which you can ask ‘where am I’ is subject to this limitation.
- Wi-Fi Manager startScan will only perform a full scan ‘a few times each hour’, this prevents an application using Google’s Geolocation REST API to return the position based on nearby Aps
  - Note also that in ‘P’ this method is marked as deprecated.

Oreo Background Limitations

Update frequency from location APIs

There is not a single location API family on Android, these are the **unaffected** APIs:

- The batched version of the fused location provider.
- Geofencing. Used to determine if a device has entered or left an area.
- ActivityRecognitionClient which replaces the earlier ActivityRecognitionApi can be used to determine if the user is performing various activities such as walking or driving.
- Any indoor-based APIs, including 3rd party APIs that rely on BLE or other hardware dependant technologies such as the magnetometer.
- APIs for Zebra technologies such as RFID or Worry Free Wi-Fi.

Oreo Background Limitations

Update frequency from location APIs

Google's intention by implementing these restrictions is to lock down the APIs to a specific set of use cases:

- Turn by turn directions in the foreground app, bread-crumbing to determine historical location and geofencing.

Enterprise implications:

- Real time route planning for T&L use cases, updating existing routes based on the real-time outdoor position of the device.
- "Find my device" to locate a device in real time outdoors, e.g. to gather devices in preparation for the next shift or locate devices requiring battery replacement. Requires the device to maintain its location.
- Device tracking to provide historical location data. Although batched location is still available through the FLP this use case may have been previously met using the standard API, so it might require a code change in the application.

Oreo Background Limitations

Update frequency from location APIs

Mitigation:

1. **Foreground service**: Background location limits do not apply to applications with a foreground service or are themselves in the foreground. E.g. turn-by-turn directions.
2. **Passive listener**: If a different application in the foreground is requesting location updates, a background application can "piggy-back" on the request to receive updates at a faster rate, as if they too were in the foreground.
   - E.g. real-time route planning might piggy-back on the navigation app
3. **Bread-crumbing**: If your use cases revolve solely around logging the historical location of devices, consider using the batched location functionality of the Fused Location Provider.
4. **Geofencing**: If your use cases revolve solely around detecting whether a device is inside or outside of a specified area(s), a retail store, for example, consider using a Geofence.

Oreo Background Limitations

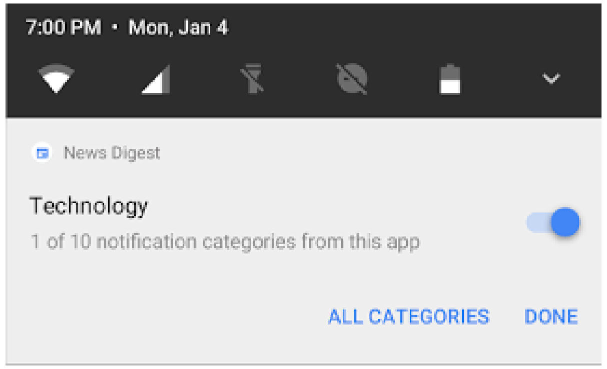
Summary

	Receiving Implicit broadcast intents	Running services in the background	Update frequency from Location APIs
Description	Implicit broadcast intents can no longer be registered for from the application manifest.	An application cannot run or have running services while in the background.	Many location APIs restrict the location update frequency to 3 or 4 times per hour.
Notable affected enterprise use cases	Relies on DataWedge intents or MX Intent capability	Non-FCM push based solutions, network monitoring, long running downloads or uploads	Functionality dependent on real-time location tracking (e.g. “find my device”)
Affects APIs targeting less than 26?	No, unless user manually re-enables it from the settings UI	No, unless user manually re-enables it from the settings UI	Yes
Recommended Mitigation	Rework app to use dynamic broadcast receivers in code. Target an API level less than 26 as an interim solution.	Use a Job scheduling API or a foreground service. Target an API level less than 26 as an interim solution.	Use a foreground service or a geofence / batched location (depending on use case)

ZEBRA

Oreo Notification Enhancements

Overview

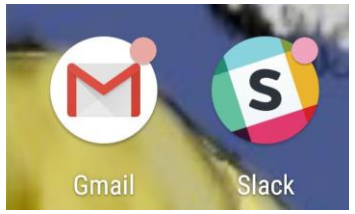

7:00 PM • Mon, Jan 4

News Digest

Technology

1 of 10 notification categories from this app

- Notification channels
  - When creating your notification, it is required to provide a channel id (int) associated with the notification. Enables the user to control notifications with greater granularity.
  - Control at enterprise level (Zebra APIs) remains that notifications are either on or off.
    - No current plans to add granular notification configuration
  - MX capabilities to lock down notification configuration:
    - AppManager can be used to prevent access to the AppInfo screen (from where notifications are controlled)
    - SettingsManager can be used to prevent the user accessing notification settings directly from the notification (slide → cog icon)
    - UIManager can disable access to the notification pulldown & quick settings.
- Notification dots (or badges)
  - Only available on supported launchers

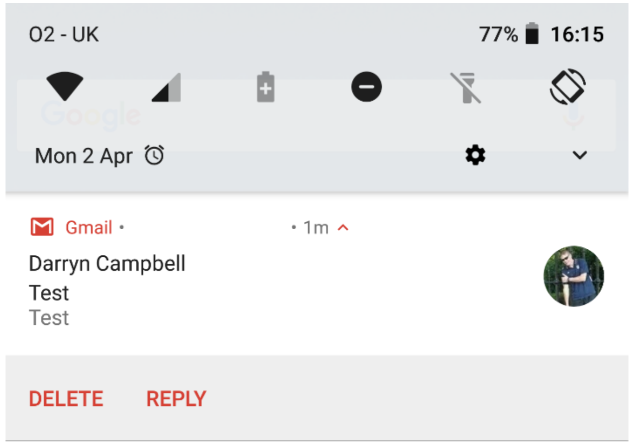


ZEBRA

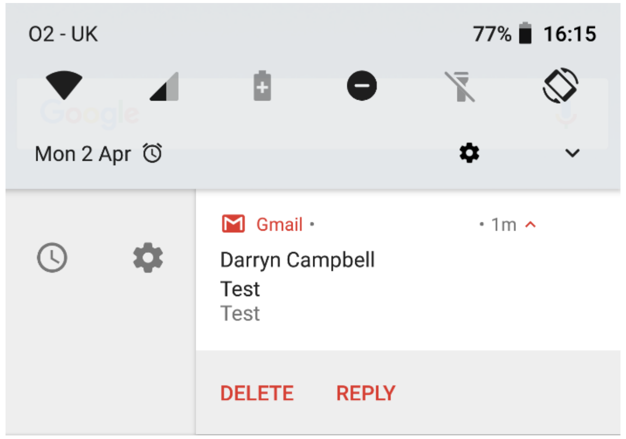
Oreo Notification Enhancements

Overview

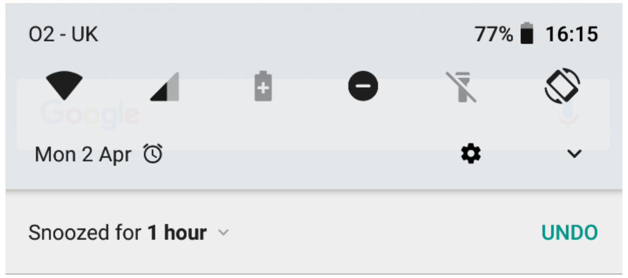
- Snoozing
  - By sliding the notification you can now snooze it
  - If you want **to prevent the user snoozing your notification then you can make it persistent**.


O2 - UK 77% 16:15
Mon 2 Apr
Gmail • • 1m
Darryn Campbell
Test
Test
DELETE REPLY

Slide


O2 - UK 77% 16:15
Mon 2 Apr
Gmail • • 1m
Darryn Campbell
Test
Test
DELETE REPLY

Snooze

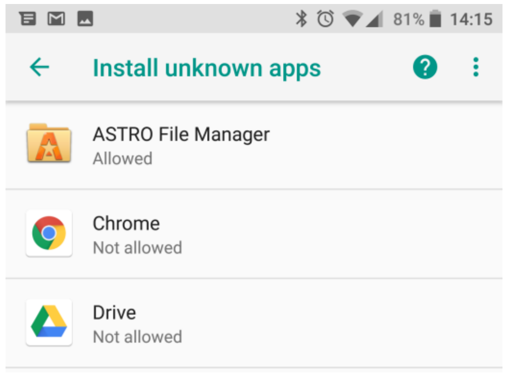

O2 - UK 77% 16:15
Mon 2 Apr
Snoozed for **1 hour** UNDO

ZEBRA

Installing apps from unknown sources

Overview

- The “Allow unknown sources” system setting has been removed and replaced with the “Install unknown apps” permission which can be granted to applications (see screenshots)
- **NOT** affected:
  - Installation via StageNow
  - Installation via EMDK Android
  - Installation via EMDK Xamarin

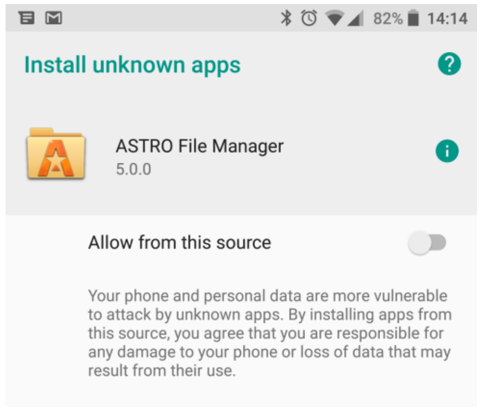

81% 14:15

Install unknown apps

ASTRO File Manager
Allowed

Chrome
Not allowed

Drive
Not allowed


82% 14:14

Install unknown apps

ASTRO File Manager
5.0.0

Allow from this source

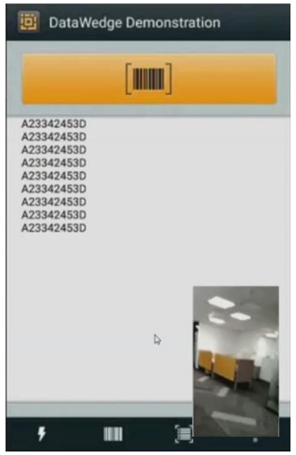
Your phone and personal data are more vulnerable to attack by unknown apps. By installing apps from this source, you agree that you are responsible for any damage to your phone or loss of data that may result from their use.

- Affected:
  - Deployments making use of MX SettingManager’s [Unknown sources parameter]

Picture-in-picture

Overview

- Applications now support picture-in-picture (PIP) mode
- Mostly consumer focussed
  - Designed for video playback or show a contact during a phone call
- No enterprise use cases identified
- Applications behind the PIP display are <u>not</u> in the background (as far as Android is concerned).
  - Zebra value-add applications have been sanity tested with PIP overlays
    - Example: Datawedge demo app running behind a VLC video



```
A23342453D
A23342453D
A23342453D
A23342453D
A23342453D
A23342453D
A23342453D
A23342453D
A23342453D
```

Webview APIs

Overview

- Several Webview APIs have been added in Oreo for applications that use an embedded Webview
- Most interestingly in terms of security is the Safe Browsing API which makes use of Google’s backend systems to prevent navigation to a site designated as unsafe.
  - Other APIs include Version, Termination handle & Renderer importance APIs
- Zebra’s Enterprise Browser will include an option for Safe Browsing in an upcoming release of that product

Google Safe Browsing

ZEBRA

Changes to the Google Play Store

Overview

To improve application security and performance Google are requiring applications in the Play Store to target a recent Android API level, this is to prohibit applications circumventing security features introduced in recent Android versions such as runtime permissions or trusting user-added certificate authorities. There is an associated Android blog post:

- **August 2018:** New applications added to the Play store are required to target API level 26 (Android 8.0) or higher
- **November 2018:** Updates to existing applications are required to target API level 26 or higher
- **2019 onwards:** Each year the targetSdkVersion requirement will advance. Within one year following each Android dessert release, new apps and app updates will need to target the corresponding API level or higher.

Changes to the Google Play Store

Overview

- This will affect Enterprise applications:
  - Managed Android devices will typically use the Managed Play store which is subject to the same new rules.
  - Updating applications will require increasing the target SDK level and considering any restrictions introduced in the newly supported level(s)
  - Targeting a lower SDK level to circumvent Android restrictions will no longer work for applications hosted in the Play Store. This has been a popular technique with consumer apps to avoid Marshmallow runtime permissions and Oreo background restrictions.
  - More robust workarounds are given in Zebra developer documentation.
- Existing applications that do not get updated will be allowed to stay in the Play store
- Application deployment that does not depend on the Play store will remain unaffected.

Android Enterprise features

Overview

- Google has started detailing “What’s new in Android” separately for Android Enterprise features.
  - Documentation is available for Nougat, Oreo, Pie
  - Most of the specific documentation is targeted at EMMs developing device owner applications
- Developers may need to liaise with those deploying applications and setting policies
  - Is the target device locked down?
    - Does my application depend on other applications (e.g. camera) that may be separately locked down?
  - Do I need to expose my configuration via managed configs to comply with EMM expectations?
  - Will the device be using a VPN so ports etc. need to be opened for my web services?

Coming in Pie…

Overview

- Background applications have further restrictions:
  - No access to microphone or camera
  - Sensor events (Accelerometers / gyroscopes) will not be received
  - New permission required for foreground services
  - Google are using *machine learning* to determine the priority given to background apps:
    - App Standby Buckets – less active apps will be able to run jobs less often
    - Background restrictions – notify the user when excessive wake locks or background services are used
    - Battery saver – only applies to battery saver mode but that mode is made more aggressive.
    - Doze mode - unaffected
- Non-SDK methods being actively discouraged
  - i.e. reflection in Java but also affects JNI calls
- Screen orientation has been reworked

Conclusions

Overview

- Biggest change in Oreo is background restrictions
- Trend of restricting what an application can do in the background is continuing and will continue to do into Android P (& likely beyond)
- Feedback so far is there will be limited impact on prepared developers
  - **Please raise feedback if background restrictions or any other Oreo feature has an impact on Enterprise development & deployment**
- Android Enterprise developer improvements typically concentrate around changes to the DevicePolicyManager API.
  - Though practically only called by Device Owner, it is useful information for all developers as it provides context for the full end-to-end solution.
- Developer post accompanying this presentation: Here

Questions?

ZEBRA DEVELOPER PORTAL

http://developer.zebra.com

Sign up for news

Join the ISV program