24.6.0 Android Release Notes

Minimum Requirements
New Features
Resolved Issues
Known Issues

Minimum Requirements

- Android 5.0 or later
- API Level 21 or later
- Workspace ONE UEM Console 2109 or later
- Android Studio with the Gradle Android Build System (Gradle) 8.2.2 or later

New Features

- Opt-Int and Out status APIs have been refactored to separate SDK and Telemetry Feature interaction.
  - The following have been added or refactored in the ***Crittercism*** singleton:

```
Refactored:
// Used to get/set SDK Opt-In / Opt-Out Status
void setOptInStatus(Crittercism.TelemetryType telemetryType, boolean optIn) -> void setSdkOptInStatus(boolean optIn)
boolean getOptInStatus(Crittercism.TelemetryType telemetryType) -> boolean getSdkOptInStatus()
Added:
// Used to get/set Telemetry Feature Opt-In Status
boolean getTelemetryOptInStatus(TelemetryFeature)
// Used to Opt-Out of all Telemetry Features
void telemetryOptOut();
```

- New API introduced to export data collected through our Telemetry Features!
  Users are now able to export data collected from the Telemetry Features they have Opted-Into. Use the following API to get Feature, Event, and Export formatted specific data in an Asynchronous fashion!

Telemetry Export API

```
/**
* Interface for handling Telemetry Feature exported data.
*/
interface TelemetryExportHandler {
    /**
    * Callback when data export has completed.
    * @param data String which will contain exported Telemetry Feature data,
    * null if error occurs.
    */
    fun onDataExport(data: String?)
}
/**
  * Asynchronously Export Data collected through enabled Telemetry Features.
  * @param feature Telemetry Feature to query data for.
  * @param eventType Type of Telemetry Event data to query for: Attributes, Events, or All.
  * @param exportType Format of the data exported, currently offering JSON support only.
  * @param dataHandler Interface called when data is ready to be exported.
  */
public static void exportTelemetryFeatureData( @NonNull TelemetryFeature feature,
                                                @NonNull TelemetryEventType eventType,
                                                @NonNull TelemetryExportType exportType,
                                                @NonNull TelemetryExportHandler dataHandler)

```

Telemetry Export Example

-
  ```
  // Implement the TelemetryExportHandler Interface to reach exported data asynchronously.
  val dataHandler = object:TelemetryExportHandler {
    // Callback method for export. Data parameter will either have formatted Telemetry
    // data or be null if an issue occurs.
    override fun onDataExport(data: String?) {
        Log.d(TAG, "Here is my exported data: ${data ?: "An issue has occurred."}")
      }
  }
  Crittercism.exportTelemetryFeatureData(
            TelemetryFeature.All, // Query data from all enabled features.
            TelemetryEventType.ALL, // Returns values for all data types.
            TelemetryExportType.JSON, // Format to export data in.
            dataHandler) // Exported data handler callback.


  ```

Resolved Issues

none

Known Issues

none