

Application Custom Launcher

Note

A quick overview of this set of Application Notes

The Custom Launcher is the secondary point of entry when Jet Pro turns on – launched by the DashWarning Activity (which is started by ReconOS upon boot). This document will walk through the steps to building your own Custom Launcher.

Table of Contents

Preamble.....	3
Custom Launcher	3
Custom Launcher System Flow.....	4
Contents of the Carousel	5
Apps	5
Contents of apps.xml.....	5
Settings	5
Customizing displayed items.....	5
Customizing apps.xml	6
Customizing the app source code	7
BaseLauncherActivity.....	7
AllAppsActivity	7
MainActivity	7
ReconOSComponentsActivity	8
SettingsActivity	9

Preamble

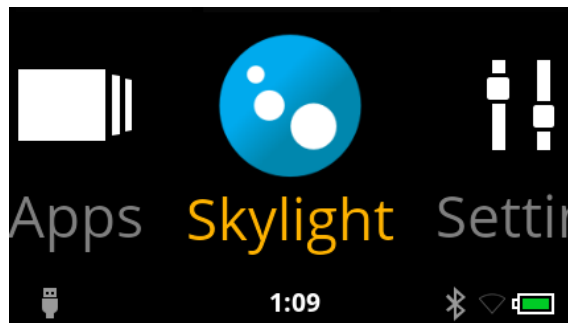
This Application Note assumes that you are familiar with the Application Note "AN101 – Setting up Android Debugging Bridge with Recon Jet Pro". It further assumes that the reader is familiar with using Android Studio to import a project, write code and lastly install and run an app on the Jet Pro device.

Requirements:

- Windows PC (Windows 8+) or Mac computer (OS X Yosemite+)
- Recon Jet Pro
- Micro-B USB cable
- Android Studio

Custom Launcher

The Custom Launcher has the following layout using the ReconOS UI Carousel component:



Custom Launcher System Flow

When Jet is powered on, the system starts key parts of ReconOS (via the DashWarning application). After start up has completed, the system looks for the installed app containing the CUSTOM_LAUNCHER action (within the apps manifest file). In order to create your own launcher from scratch, you will need to include the following line inside your manifest file:

```
...
<action android:name="com.reconinstruments.enterprise.CUSTOM_LAUNCHER"/>
...
```

More specifically, you will need to put the line inside the **<intent-filter>** tag, nested within the **<application>** tag of the AndroidManifest.xml file (see below):

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.reconinstruments.enterprisedemoapp">

    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/ReconTheme">
        <!-- This line uses the Recon UI theme for its graphical components, subject
        to change depending on developers' needs. -->
        <activity android:name=".MainActivity">
        <intent-filter>
        <action android:name="com.reconinstruments.enterprise.CUSTOM_LAUNCHER"/>
        ...
        ...
        ...
```

NOTE: Any application can call the CUSTOM_LAUNCHER action; doing so will make the application start as soon as the device finishes booting up.

Contents of the Carousel

The contents of the Carousel are populated when the Custom Launcher first boots. The Carousel may also be updated if one of the following 3 events occurs:

- The file at enterprise/apps.xml is modified
- A package (apk) is installed on Jet
- A package (apk) is uninstalled on Jet

Apps

The **Apps** item of the Custom Launcher will launch a new Activity displaying all apps which are bundled with ReconOS for Jet Pro e.g. Camera App, Compass App and Gallery App, as well as any apps which have been installed onto ReconOS.

Contents of apps.xml

The contents of apps.xml is loaded in the order in which the package names are entered in the apps.xml file. The contents of this file may be empty.

Settings

The **Settings** item of the Custom Launcher will launch a new Activity displaying a vertical list of options to modify settings on Jet. For example, Display settings and Battery Saving settings can be accessed from this Activity.

Customizing displayed items

A user can select which apps are shown on the carousel between **Apps** and **Settings** by modifying the contents of the file located at “enterprise/apps.xml”.

NOTE: If more than one app is specified to be shown in carousel, they will be ordered in the manner that they are entered in “enterprise/apps.xml”. The first app will be focused on by default. With zero apps installed, the **Apps** item will be selected by default when the Launcher opens.

Customizing apps.xml

The Custom Launcher uses the file “enterprise/apps.xml” to determine which apps display when the device is powered on. The xml file can be found in the enterprise folder at root folder level at “enterprise/apps.xml”. These apps will be displayed following the **Apps** icon. The file is based on a small schema with the following template.

```
<?xml version="1.0" encoding="utf-8"?>
<apps>
<app namespace="com.example.packagename" />
</apps>
```

The file must have a root element called <apps>, which contains one or more elements called <app>. The <app> element itself must contain only one attribute: namespace. The value of the namespace attribute must match the package namespace of the app that is to be displayed.

The file can be updated live, while the Launcher is being displayed.

NOTE: If the “enterprise/apps.xml” file is corrupted, missing, or empty no packages will be displayed – however Apps and Settings will still be displayed.

Customizing the app source code

There are 5 activities within the sample class:

- BaseLauncherActivity
- AllAppsActivity
- MainActivity
- ReconOSComponentsActivity
- SettingsActivity

BaseLauncherActivity

The BaseLauncherActivity contains a lot of the functionality that is shared between the MainActivity which is the primary entry point of the application, and the AllAppsActivity which presents a carousel of all apps installed on the Recon Jet Pro.

AllAppsActivity

The AllAppsActivity extends the BaseLauncherActivity. The onCreate() method is where the AllAppsActivity is initialized. This class inherits a significant amount of logic and is therefore quite thin. It should serve as a point of reference when comparing what applications are displayed between this AllAppsActivityclass and the MainActivityClass.

MainActivity

The MainActivity extends the BaseLauncherActivity. The onCreate() method is where the MainActivity is initialized. It will populate the Carousel contents, and add a FileObserver to the "apps.xml". The getCarouselContents() method is where the items are added to the Carousel.

The first item added to the Carousel is **Apps**. It uses the ImageCarouselItem class – a subclass of the ReconOS UI StandardCarouselItem component. Looking at the ImageCarouselItem class shows the extended behavior lies within the onClick() method – when the ImageCarouselItem is shown in the center of the Carousel, and the user presses

the Select button the Intent attached to that instance of the `ImageCarouselItem` is launched. The last item added to the Carousel is “Settings” which also uses the `ImageCarouselItem` class.

Beneath the code to add **Apps** to the Carousel is a call to `getDrawableCarouselItems()`. This method loads and parses the “apps.xml” file in order to generate an `ArrayList<String>` of package names.

Note the two **TODO** comments in this method. The first highlights the decisions to be made around handling Exceptions. Each implementation has its own decision to make around this. It is highly recommended to follow the guidelines at <https://software.intel.com/en-us/articles/design-guidelines-for-recon-jet> for best practice on how to handle interruptions to the user.

The second **TODO** comment provides an opportunity to quickly grasp how to add extra items to the Carousel. Uncomment the code in this area, click Run and note the new **Components** item added to the Carousel. Click on this item, and note the ReconOS UI List component in use here. Try navigating up and down this list, and press Select on each item in this list to discover what each item launches. The code for this activity is in `ReconOSComponentsActivity`.

The **Settings** item on the Carousel launches the `SettingsActivity`.

[ReconOSComponentsActivity](#)

This sample Activity highlights some of the Intents that can be launched within ReconOS. Note that the list of items on this list is not fully exhaustive of all intents available to launch within ReconOS. Example Intents to launch here include Overlay Menus such as the `RECON_POWER_MENU`. Further information on all Intents within ReconOS can be found within “AN005 – A list of available ReconOS Intents”.

SettingsActivity

Similar in behavior to `ReconOSComponentsActivity`, and highlights further uses of the ReconOS UI List component. Each item in the List will launch the ReconOS Settings panel specific to each item. For example clicking the Bluetooth `ListItem` will launch the ReconOS Bluetooth Settings Menu.

Display, Smartphone, Bluetooth, Camera, Device, Notifications, Battery Saving, Advanced are the default options for `SettingsActivity`.