DARRYN CAMPBELL

Mobile computing and enterprise software development

Tutorial: The DataWedge Intent API (with Xamarin)

10th April 2018 0 *By* DARRYNCAMPBELL

This tutorial will show the basic principles of how to control DataWedge via the **DataWedge Intent API with a Xamarin application**. The DataWedge API can be quite powerful, I have written a fully featured application (in Java) in the past but frequently I find people looking for something simpler as a jumping off point.

This tutorial requires a Zebra device running DataWedge 6.3 or higher.

DataWedge configuration

There should be no special configuration required for DataWedge, this tutorial is designed to use the default profile. If something does not work then please see the troubleshooting section at the end.

The application

All the application will be doing is sending Android Intents to the DataWedge service. DataWedge is a service on the device that contains a Broadcast receiver listening for a specific Intent action and determining what to do based on the extras contained in that Intent.

The UI will have 2 features:

1. Toggle the Scanner beam (i.e. simulate pressing the trigger in software). This is the " ScanSoftTrigger" API
2. Enable or Disable the scanner. This is the " Scanner Input Plugin" API.

I have omitted the code to create the 3 buttons on our Android UI

In our code, define some constants for the Intent actions and extras, these strings are detailed in the documentation for each API

```
//  The Intent action is common for all DataWedge APIs
private string DATAWEDGE_ACTION = "com.symbol.datawedge.api.ACTION";

//  Toggle the scanner.  Define the intent extra and value
```

```
 private string DATAWEDGE_EXTRA_KEY_SCANNER_TRIGGER_CONTROL =
"com.symbol.datawedge.api.SOFT_SCAN_TRIGGER";
 private string DATAWEDGE_EXTRA_VALUE_TOGGLE_SCANNER = "TOGGLE_SCANNING";

 //  Enable / disable the scanner.  Define the intent extra key and values.
 private string DATAWEDGE_EXTRA_KEY_SCANNER_CONTROL =
"com.symbol.datawedge.api.SCANNER_INPUT_PLUGIN";
 private string DATAWEDGE_EXTRA_VALUE_ENABLE_SCANNER = "ENABLE_PLUGIN";
 private string DATAWEDGE_EXTRA_VALUE_DISABLE_SCANNER = "DISABLE_PLUGIN";
```

Then hook up the UI buttons in the onCreate method.

First the scanner beam toggle:

```
 Button toggleScanner = FindViewById<Button>(Resource.Id.btnToggleTrigger);
 toggleScanner.Click += delegate { onToggleTrigger(); };

 void onToggleTrigger()
 {
   Intent intent = new Intent();
   intent.SetAction(DATAWEDGE_ACTION);
   intent.PutExtra(DATAWEDGE_EXTRA_KEY_SCANNER_TRIGGER_CONTROL,
DATAWEDGE_EXTRA_VALUE_TOGGLE_SCANNER);
   SendBroadcast(intent);
 }
```

Then the two buttons for enabling or disabling the scanner:

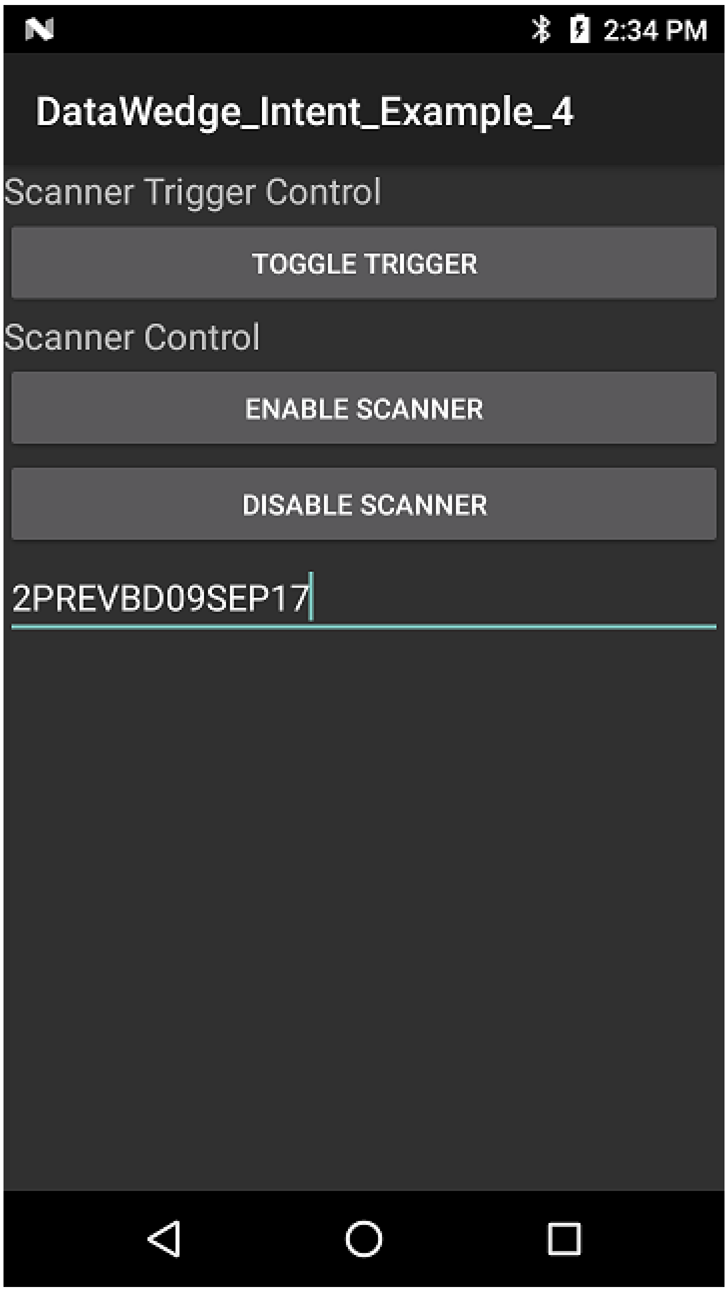
```
 Button enableScanner = FindViewById<Button>(Resource.Id.btnEnableScanner);
 enableScanner.Click += delegate { onEnableScanner(); };
 Button disableScanner = FindViewById<Button>(Resource.Id.btnDisableScanner);
 disableScanner.Click += delegate { onDisableScanner(); };

 void onEnableScanner()
 {
   Intent intent = new Intent();
   intent.SetAction(DATAWEDGE_ACTION);
   intent.PutExtra(DATAWEDGE_EXTRA_KEY_SCANNER_CONTROL,
DATAWEDGE_EXTRA_VALUE_ENABLE_SCANNER);
   SendBroadcast(intent);
 }

 void onDisableScanner()
 {
   Intent intent = new Intent();
   intent.SetAction(DATAWEDGE_ACTION);
   intent.PutExtra(DATAWEDGE_EXTRA_KEY_SCANNER_CONTROL,
DATAWEDGE_EXTRA_VALUE_DISABLE_SCANNER);
   SendBroadcast(intent);
 }
```

And that is really all you need to do to create an application which has basic control of the scanner using the DataWedge API. You can of course get a lot more complicated as the DW API has quite a few features

There is a sample application that accompanies this blog and shows all the code at https://github.com/darryncampbell/DataWedge-Intent-Example-4

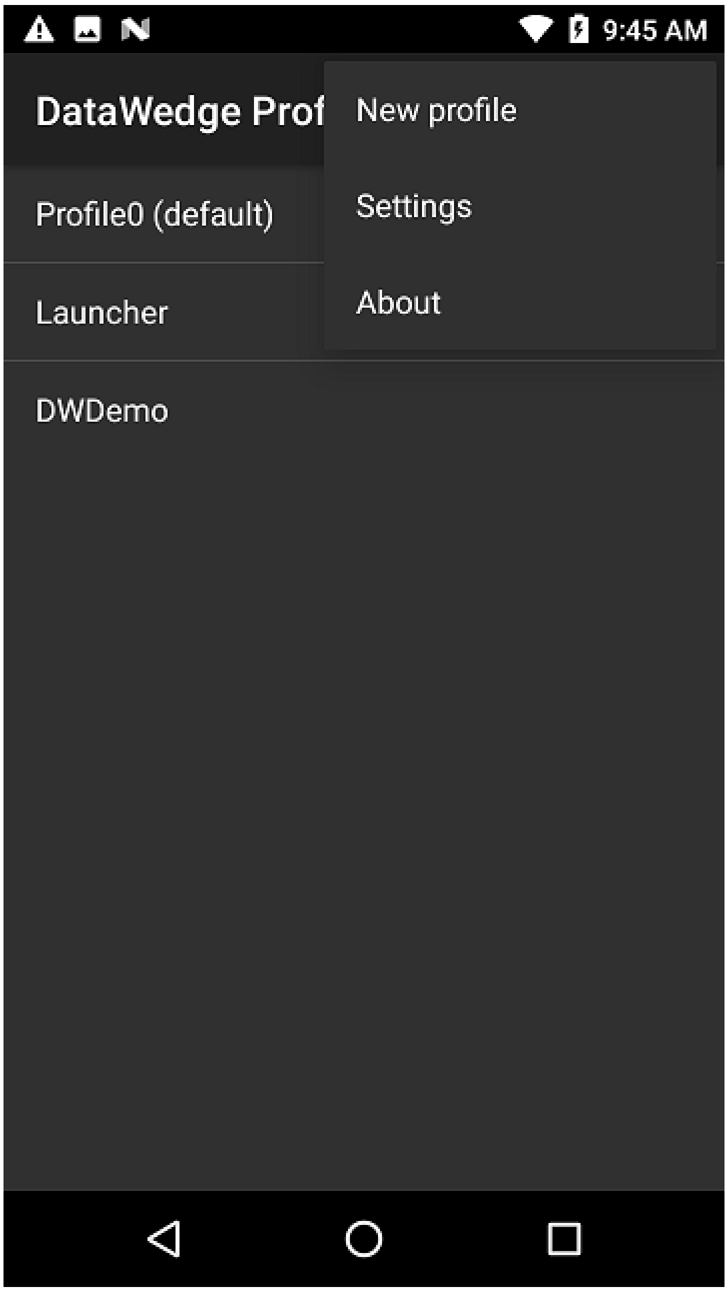


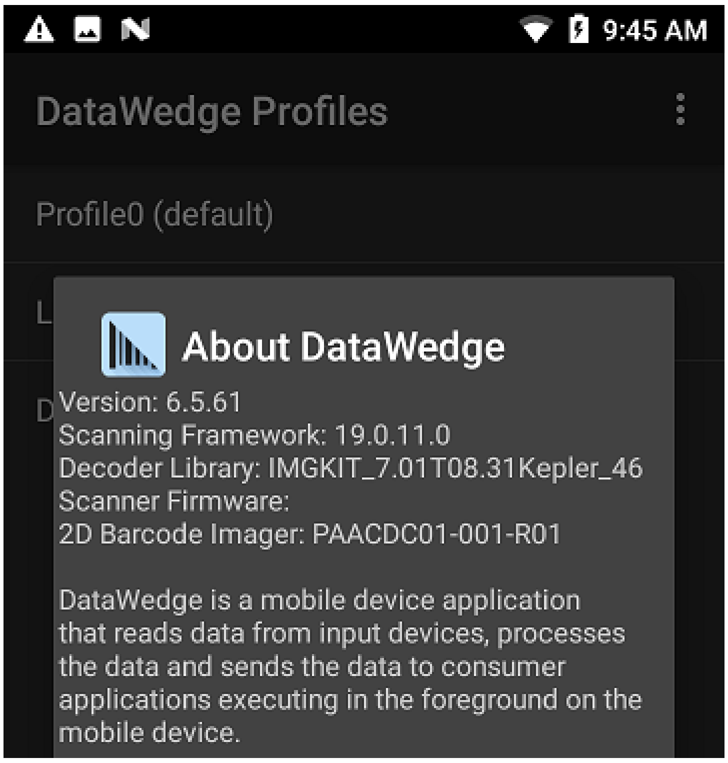
Troubleshooting:

If the application does not work for you then try some of these troubleshooting tips:

DataWedge Version

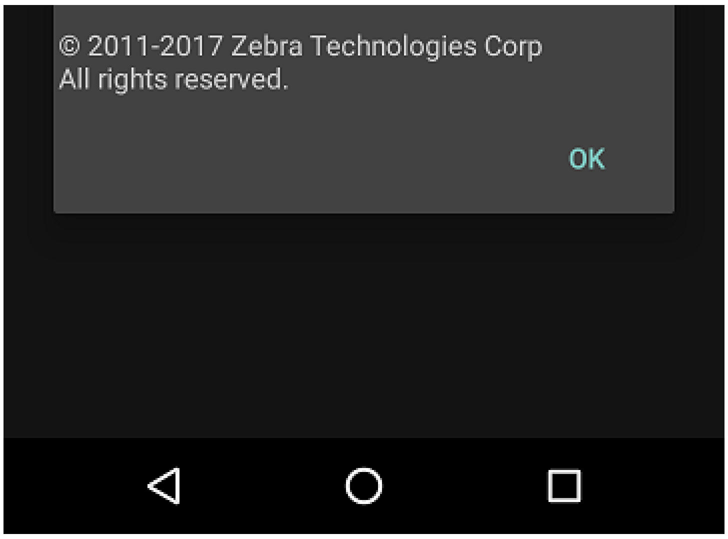
This application will only work on Zebra Android mobile devices running DataWedge 6.3 or higher. To check your datawedge version, launch the DataWedge application then go to the menu (3 dots), then 'about':





Version: 6.5.61
Scanning Framework: 19.0.11.0
Decoder Library: IMGKIT_7.01T08.31Kepler_46
Scanner Firmware:
2D Barcode Imager: PAACDC01-001-R01

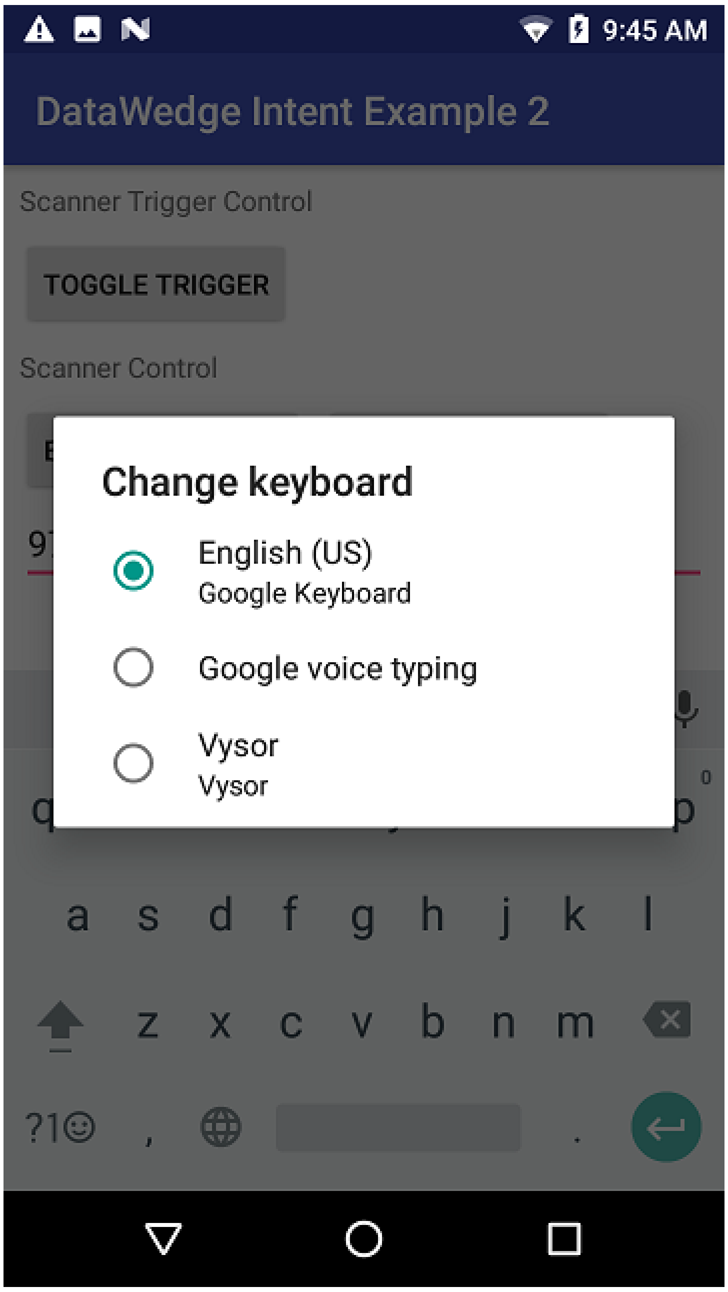
DataWedge is a mobile device application that reads data from input devices, processes the data and sends the data to consumer applications executing in the foreground on the mobile device.


© 2011-2017 Zebra Technologies Corp
All rights reserved.

It is possible to use a version of the DataWedge APIs that work on 6.0 or higher, for the sake of simplicity I have used the more recent 6.3+ APIs in this example but you can see the 6.0+ APIs in my more fully featured project (in Java).

Vysor or other keyboard applications

This application depends on DataWedge sending keyboard output to the application after a successful scan. Vysor has been known to interrupt this workflow, it is recommended to uninstall Vysor or ensure a non-Vysor keyboard is selected (you may need to select the Vysor keyboard and then re-select the Google keyboard):



DataWedge configuration

Unless you have changed any of the DataWedge configuration on the device, this application will use the default profile (Profile 0) with:

- Barcode input Enabled with common decoders
- Keystroke output Enabled

If you have changed any of these settings then you may find the application does not work e.g. disabling keystroke output or disabling the default profile.

There are multiple ways to resolve this including:

- You can restore DataWedge back to its default settings from the DataWedge application –> Menu (3 dots) –> Settings –> Restore
- You can create a new profile and associate it with this application

Share this: