DevCon 2025

Connect | Learn | Build

June 4 - 6, 2025

Amsterdam, Netherlands

ZEBRA

DevCon 2025

Connect | Learn | Build

Develop Applications for Zebra Mobile Devices by Use Case

Prashanth Kadur
Director, Software Engineering

ZEBRA

DevCon 2025

Connect | Learn | Build

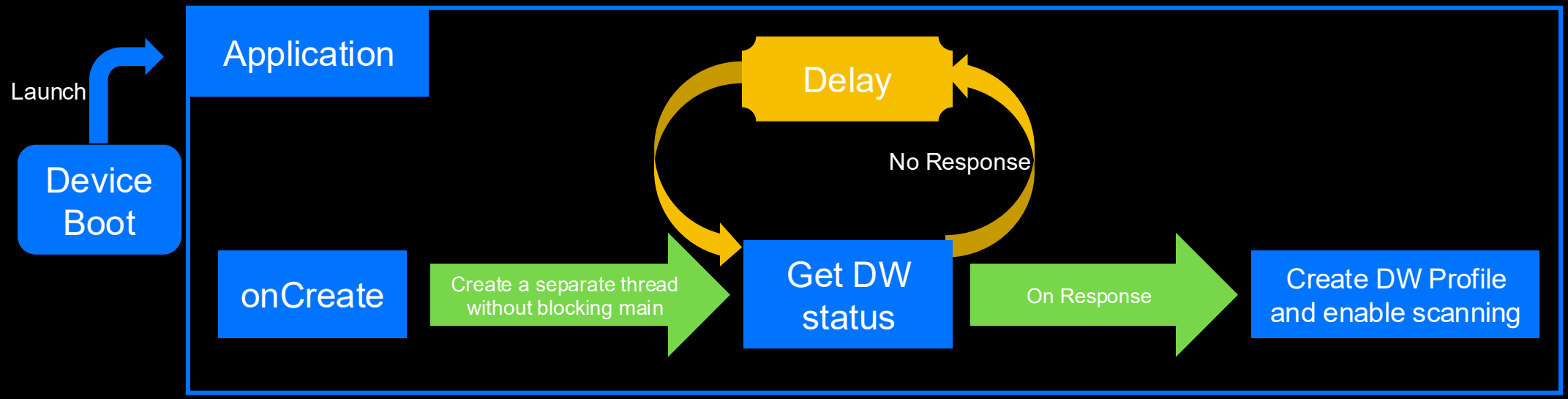
Capture data efficiently using DataWedge (DW)

ZEBRA

Capture data efficiently using DW

Programmatically Checking DataWedge Readiness Before Sending Requests

- I have an application that launches immediately after device reboot. Sometime my appl does not scan immediately.
- System takes time to prepare and invoke DW. App must check if DataWedge is fully initialized and ready before sending any requests to it. This ensures that the application operates reliably after the restart.
- **Future:** Notifications APIs will be introduced instead of polling to check the DataWedge readiness



Capture data efficiently using DW

Checking DW Status

DevCon 2025
Connect | Learn | Build

```
ScheduledExecutorService scheduler = Executors.newScheduledThreadPool( corePoolSize: 1);
Runnable statusQueryTask = () -> {
    if (Boolean.TRUE.equals(bIsDataWedgeReady)){
        Log.d(LOG_TAG, msg: "DataWedge is ready, create profile");
        createProfile();
        scheduler.shutdown();
    }else {
        Intent dwIntent = new Intent();
        dwIntent.setAction("com.symbol.datawedge.api.ACTION");
        dwIntent.putExtra( name: "com.symbol.datawedge.api.GET_DATAWEDGE_STATUS", value: "");
        this.sendBroadcast(dwIntent);
    }
};
// Schedule the task to run every 500 milliseconds
scheduler.scheduleWithFixedDelay(statusQueryTask, initialDelay: 0, delay: 500, TimeUnit.MILLISECONDS);
```

```
private BroadcastReceiver myBroadcastReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        Log.d(LOG_TAG, msg: "DataWedge Action:" + action);

        // Get DataWedge status info
        if (intent.hasExtra( name: "com.symbol.datawedge.api.RESULT_GET_DATAWEDGE_STATUS")) {
            String dwStatus = intent.getStringExtra( name: "com.symbol.datawedge.api.RESULT_GET_DATAWEDGE_STATUS");
            bIsDataWedgeReady = true;
            Log.i(LOG_TAG, msg: "DataWedge Status: " + dwStatus);
        }
    }
};
```

Capture data efficiently using DW

Programmatically create a profile

DevCon 2025
Connect | Learn | Build

- I need to programmatically create my profiles when I need and enable scanning

```
//create a profile with returning result
Intent intent = new Intent();
intent.setAction("com.symbol.datawedge.api.ACTION");
intent.putExtra(
        name: "com.symbol.datawedge.api.CREATE_PROFILE",
        value: "DWDataCapture1"
);
intent.putExtra( name: "SEND_RESULT",  value: "true");
intent.putExtra( name: "COMMAND_IDENTIFIER",  value: "1234");
this.sendBroadcast(intent);
```

Capture data efficiently using DW

Do not forget to register for notifications!

DevCon 2025
Connect | Learn | Build

- Why do I need to receive DW notifications in my app?

```
IntentFilter filter = new IntentFilter();
filter.addAction("com.symbol.datawedge.api.NOTIFICATION_ACTION");
filter.addAction("com.symbol.datawedge.api.RESULT_ACTION");
filter.addAction("com.zebra.datacapture1.ACTION");
filter.addCategory(Intent.CATEGORY_DEFAULT);
registerReceiver(myBroadcastReceiver, filter);
```

```
private final BroadcastReceiver myBroadcastReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        if(action.equals("com.zebra.datacapture1.ACTION")){
            String data = intent.getStringExtra( name: "com.symbol.datawedge.data_string");
        }
        else if (action.equals("com.symbol.datawedge.api.NOTIFICATION_ACTION")) {
            if (intent.hasExtra( name: "com.symbol.datawedge.api.NOTIFICATION")) {
                Bundle b = intent.getBundleExtra( name: "com.symbol.datawedge.api.NOTIFICATION");
                String NOTIFICATION_TYPE = b.getString( key: "NOTIFICATION_TYPE");
            }
        }

    }
};
```

`SCANNER_STATUS`:

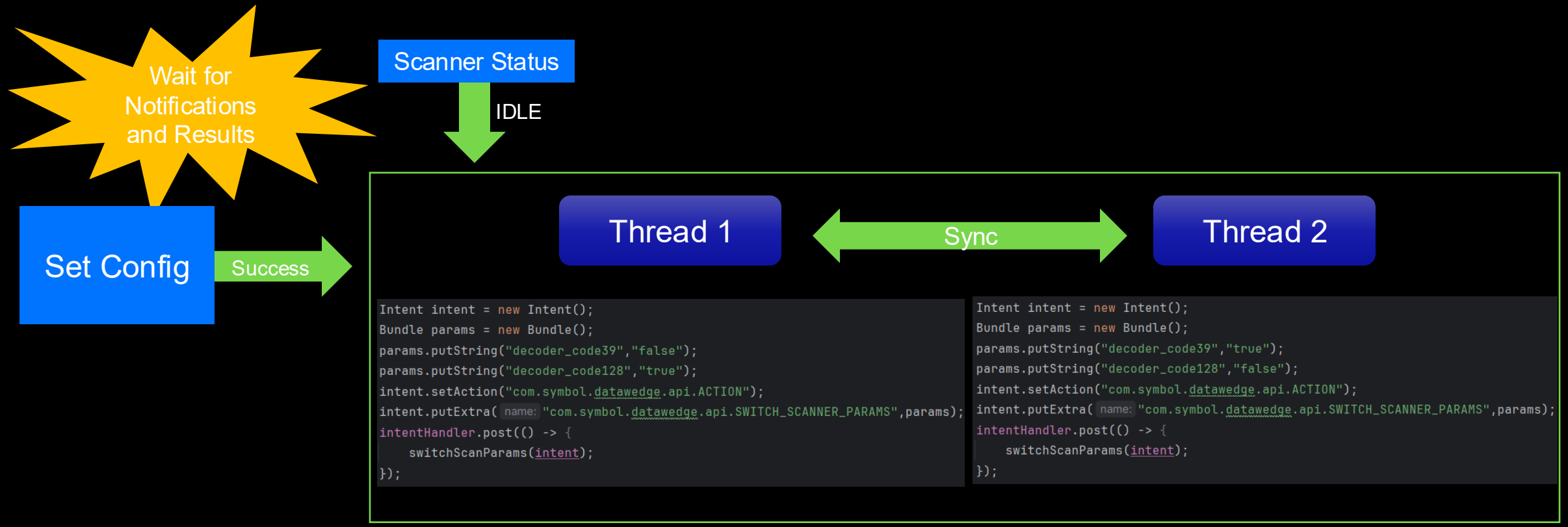
- **WAITING** - Scanner is enabled and ready to scan using a physical trigger or SOFT_SCAN_TRIGGER intent.
- **SCANNING** - Scanner has emitted the scan beam and scanning is in progress. This event does not prevent the application from disabling other controls as necessary.
- **CONNECTED** - A Bluetooth scanner has connected with the device and can now be enabled (or disabled) by the application. Scanner selection should be set to Auto in the currently active profile.
- **DISCONNECTED** - A Bluetooth scanner has disconnected from the device. Sending an intent to enable or disable the scanner in this state will enable/disable the current default scanner.
- **IDLE** - Scanner is in one of the following states: enabled but not yet in the waiting state, in the suspended state by an intent (e.g. SUSPEND_PLUGIN) or disabled due to the hardware trigger.
- **DISABLED** - Scanner is disabled. This is broadcasted by the scanner plug-in when the active profile becomes disabled manually or the scanner is disabled with an intent (e.g. DISABLE_PLUGIN).

Capture data efficiently using DW

I need to configure and scan in my multi-threaded application

DevCon 2025
Connect | Learn | Build

- The order of your calls matter. So, you must synchronize your calls.
- For example, when an app was launched, enabling different decoders commands were sent in separate threads; Sometime the correct decoder was enabled and sometime an incorrect decoder was enabled.



Capture data efficiently using DW

I need to configure and scan in my multi-threaded application

DevCon 2025
Connect | Learn | Build

```
public static BroadcastReceiver dwBroadcastReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        assert action != null;
        if (action.equals("com.symbol.datawedge.api.NOTIFICATION_ACTION")){
            if (intent.hasExtra( name: "com.symbol.datawedge.api.NOTIFICATION")){
                Bundle extras =
                        intent.getBundleExtra( name: "com.symbol.datawedge.api.NOTIFICATION");
                assert extras != null;
                String notificationType = extras.getString( key: "NOTIFICATION_TYPE");
                if (notificationType != null) {
                    switch (notificationType) {
                        case "SCANNER_STATUS":
                            // Change in scanner status occurred
                            isScannerIdle = extras.getString( key: "STATUS").equals("IDLE")
                                    && extras.getString( key: "PROFILE_NAME").equals(PROFILE_NAME);
                            //Handle other updates
                            break;
                        case "PROFILE_SWITCH":
                            // Received change in profile
                            break;
                        default:
                            // Handle default values
                            break;
                    }
                }

            }
        }
    }
};
```

```
private static final Object dwIntentCallLock=new Object();
1 usage
private static final String PROFILE_NAME="MyProfile";
2 usages
static boolean isScannerIdle=false;
2 usages
private final Handler intentHandler;

1 usage
public DWHandler(){
    HandlerThread handlerThread = new HandlerThread( name: "DWIntentThread");
    intentHandler = new Handler(handlerThread.getLooper());
}
```

```
public void switchScanParams(Intent intent){
    try {
        //Loop until scanner is IDLE
        while (!isScannerIdle){
            Thread.sleep( millis: 200L);
        }
        sendIntent(intent);
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }
}
1 usage
private void sendIntent(Intent intent){
    synchronized (dwIntentCallLock){
        App.getInstance().sendBroadcast(intent);
    }
}
```

Capture data efficiently using DW

I need to configure and scan in my multi-threaded application

DevCon 2025
Connect | Learn | Build

```
public static BroadcastReceiver dwBroadcastReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        assert action != null;
        if (action.equals("com.symbol.datawedge.api.NOTIFICATION_ACTION")){
            if (intent.hasExtra( name: "com.symbol.datawedge.api.NOTIFICATION")){
                Bundle extras =
                        intent.getBundleExtra( name: "com.symbol.datawedge.api.NOTIFICATION");
                assert extras != null;
                String notificationType = extras.getString( key: "NOTIFICATION_TYPE");
                if (notificationType != null) {
                    switch (notificationType) {
                        case "SCANNER_STATUS":
                            // Change in scanner status occurred
                            isScannerIdle = extras.getString( key: "STATUS").equals("IDLE")
                                    && extras.getString( key: "PROFILE_NAME").equals(PROFILE_NAME);
                            //Handle other updates
                            break;
                        case "PROFILE_SWITCH":
                            // Received change in profile
                            break;
                        default:
                            // Handle default values
                            break;
                    }
                }

            }
        }
    }
};
```

```
private static final Object dwIntentCallLock=new Object();
1 usage
private static final String PROFILE_NAME="MyProfile";
2 usages
static boolean isScannerIdle=false;
2 usages
private final Handler intentHandler;

1 usage
public DWHandler(){
    HandlerThread handlerThread = new HandlerThread( name: "DWIntentThread");
    intentHandler = new Handler(handlerThread.getLooper());
}
```

```
public void switchScanParams(Intent intent){
    try {
        //Loop until scanner is IDLE
        while (!isScannerIdle){
            Thread.sleep( millis: 200L);
        }
        sendIntent(intent);
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }
}
1 usage
private void sendIntent(Intent intent){
    synchronized (dwIntentCallLock){
        App.getInstance().sendBroadcast(intent);
    }
}
```

Capture data efficiently using DW

How do I utilize a single device to meet both professional and personal needs?

DevCon 2025
Connect | Learn | Build

Use Case:

- How do I utilize a single device to meet both professional and personal needs while getting the flexibility to switch between work and personal profiles and ensuring that corporate data remains secure and separate from personal applications, thereby safeguarding my privacy?

Solution:

- Android Work Profile on mixed-use company-owned devices enables organizations to enforce corporate policies and restrictions while preserving the privacy of personal data. Formerly known as Corporate-Owned, Personally Enabled (COPE).

- Work Profile - Zebra Technologies TechDocs

Capture data efficiently using DW

How do I utilize a single device to meet both professional and personal needs? Contd.

DevCon 2025
Connect | Learn | Build

Customers can provide a Zebra Mobile Computer to their associates to do the enterprise work as well as use it for their personal use.

There will be two user profiles in the device:

- **Personal** – Associate can use for their personal use
- **Work** – Administrator will install all the enterprise applications.

DataWedge will be supported in the COPE mode but there are few things need to know:

- By default, DataWedge won't be enabled in the Work Profile, but it will be available in Personal.
- Administrator must enable the DataWedge application in the Work profile.
- When DataWedge is enabled in the Work Profile user will not be able to use DataWedge in personal user apps.

Capture data efficiently using DW

How do I utilize a single device to meet both professional and personal needs? Contd.

DevCon 2025
Connect | Learn | Build

Supported Features

- Barcode scanning via internal imager
- Keystroke data dispatching
- Intent data dispatching
- Dispatching data using IP Output
- Data Capture Plus
- Advanced data formatting
- Basic data formatting
- Import/Export DataWedge configurations
  Mass deployment

Limitations

- Cannot use SD card for configuration deployment
- Unsupported features
  - Voice Input
  - Serial Input
  - Enable scanning in Launcher screen

Capture data efficiently using DW

How do I temporarily suspend/resume scanning in my app?

DevCon 2025
Connect | Learn | Build

As a developer I need to enable the scanner only in scan fields which ready to accept scanned data and disable the scanner in manual entering field to avoid any accidental scans and filling unintended data. So, I should be able to quickly disable and enable the scanner. What is the most efficient way to achieve this?

- **Suspend the scanner** (temporarily inactive). This can be called only when the scanner is in the SCANNING or WAITING state. The scanner state can be retrieved using Get Scanner Status or Register for Notification.

```
private void suspendScanner() {
    Intent i = new Intent();
    i.setAction("com.symbol.datawedge.api.ACTION");
    i.putExtra("com.symbol.datawedge.api.SCANNER_INPUT_PLUGIN", "SUSPEND_PLUGIN");
    i.putExtra("SEND_RESULT", "true");
    i.putExtra("COMMAND_IDENTIFIER", "MY_SUSPEND_SCANNER");  //Unique identifier
    this.sendBroadcast(i);
}
```

Resume the scanner

```
private void resumeScanner() {
    Intent i = new Intent();
    i.setAction("com.symbol.datawedge.api.ACTION");
    i.putExtra("com.symbol.datawedge.api.SCANNER_INPUT_PLUGIN", "RESUME_PLUGIN");
    i.putExtra("SEND_RESULT", "true");
    i.putExtra("COMMAND_IDENTIFIER", "MY_RESUME_SCANNER");  //Unique identifier
    this.sendBroadcast(i);
}
```

Capture data efficiently using DW

Profile Stability in Multiple Applications:

DevCon 2025
Connect | Learn | Build

How can I ensure uninterrupted scanning functionality when user actions or EMM intermittently delete configured scanning profiles, causing applications to lose scanning capabilities?

Restrict DataWedge configuration deployment (EMM)

- DataWedge provides an option for admins to turn On/Off automatic importation of DataWedge configuration files when they're pushed to the /enterprise/device/settings/datawedge/autoimport folder on the device.
  Turn On/Off Auto Import Settings

Restrict user actions (UI)

- DataWedge provides an option for admins to control whether the device user has access to the DataWedge user interface and can change configuration settings on the device.
  Enable/Disable Device DW UI

Capture data efficiently using DW

How do I prevent an un-authorized app programmatically altering the scanning functionality and DataWedge configuration?

DevCon 2025
Connect | Learn | Build

Restrict access to intent API

- DataWedge provides an option for admins to control access to DataWedge intent APIs, allowing only approved apps to configure DataWedge. DataWedge intent APIs are categorized, allowing the administrator to grant DataWedge API access to specific apps based on category. By default, DataWedge accepts any intent API to avoid impact to existing applications.
  Control Access to DataWedge Intent APIs

Capture data efficiently using DW

How do I prevent an un-authorized app programmatically altering the scanning functionality and DataWedge configuration?

DevCon 2025
Connect | Learn | Build

Restrict access to intent API

- DataWedge provides an option for admins to control access to DataWedge intent APIs, allowing only approved apps to configure DataWedge. DataWedge intent APIs are categorized, allowing the administrator to grant DataWedge API access to specific apps based on category. By default, DataWedge accepts any intent API to avoid impact to existing applications.
  Control Access to DataWedge Intent APIs

DevCon 2025

Connect | Learn | Build

Capture data efficiently using Ent. Mobility Dev Kit (EMDK)

ZEBRA

Capture data efficiently using EMDK

How do I retrieve EMDK version on the device programmatically?

DevCon 2025
Connect | Learn | Build

- Zebra recommends to use VersionManager API to retrieve EMDK version pre-installed on the device.

- Note: Use of emdk service package name and PackageManager APIs to retrieve EMDK version is not recommended and will not provide required result on A14 onward.

```
// Get an instance of VersionManager:
VersionManager versionManager = (VersionManager) emdkManager.getInstance(EMDKManager.FEATURE_TYPE.VERSION);

// Use the getVersion method, passing in the VersionManager.VERSION_TYPE.EMDK:
String version = versionManager.getVersion(VersionManager.VERSION_TYPE.EMDK);
```

Techdocs: Can I determine programmatically which EMDK version is on a device?

Capture data efficiently using EMDK

How do I use EMDK API calls for an application after a reboot?

DevCon 2025
Connect | Learn | Build

- As kiosk application my application will be started on the device reboot. If my application needs to interact with the EMDK using the EMDK API what is the best way to find out whether EMDK is ready or not?
- Code snippet: TBD

DevCon 2025

Connect | Learn | Build

Capture data efficiently using Enterprise Browser (EB)

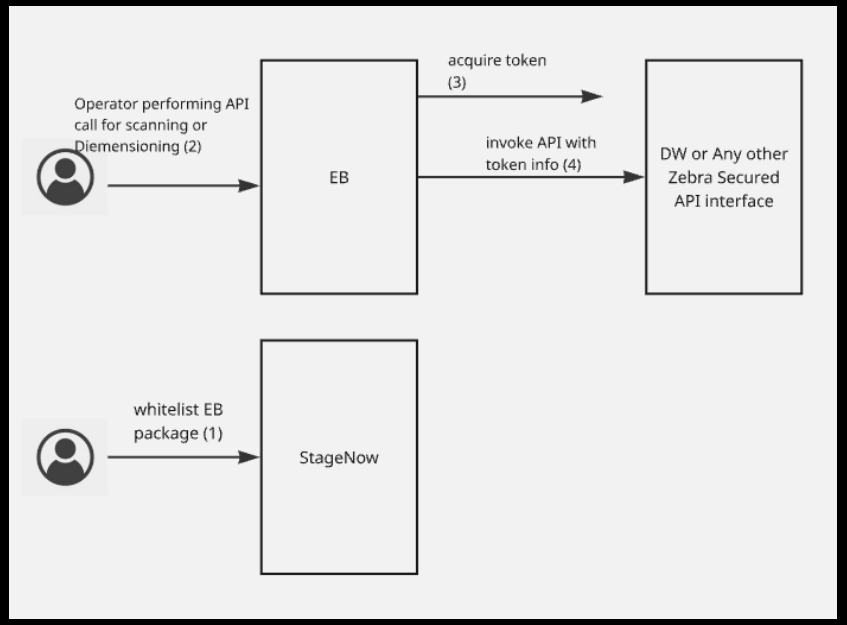
ZEBRA

Capture data efficiently using EB

How do I get gain permission to access DataWedge APIs?

DevCon 2025
Connect | Learn | Build

- Controlled Access to Datawedge




```
//set service identifier to be accessed from datawedge
var serviceIdentifierName = "delegation_scope_datawedge_query_api";
//get token from accessmgr
var token = EB.Accessmgr.acquireToken(serviceIdentifierName);
//pass the EB package name and token as extras in intent.
var extras = {
                "com.symbol.datawedge.api.GET_ACTIVE_PROFILE":"",
                "APPLICATION_PACKAGE": "com.zebra.mdna.enterprisebrowser",
                "TOKEN", token
            };

var params = {
                intentType: EB.Intent.BROADCAST,
                action: 'com.symbol.datawedge.api.ACTION',
                appName: 'com.symbol.datawedge',
                data: extraData
            };
EB.Intent.send(params);
```

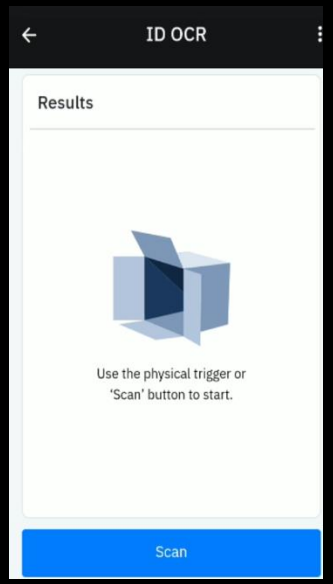
Capture data efficiently using EB

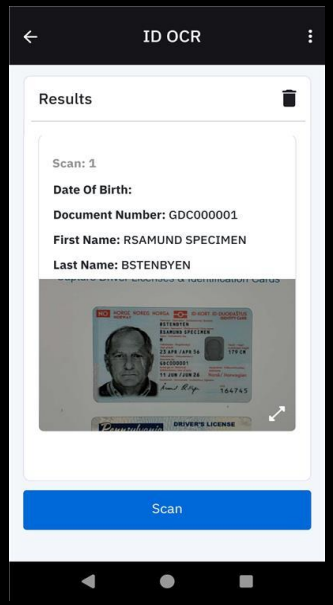
I would like perform OCR in my web app using DW

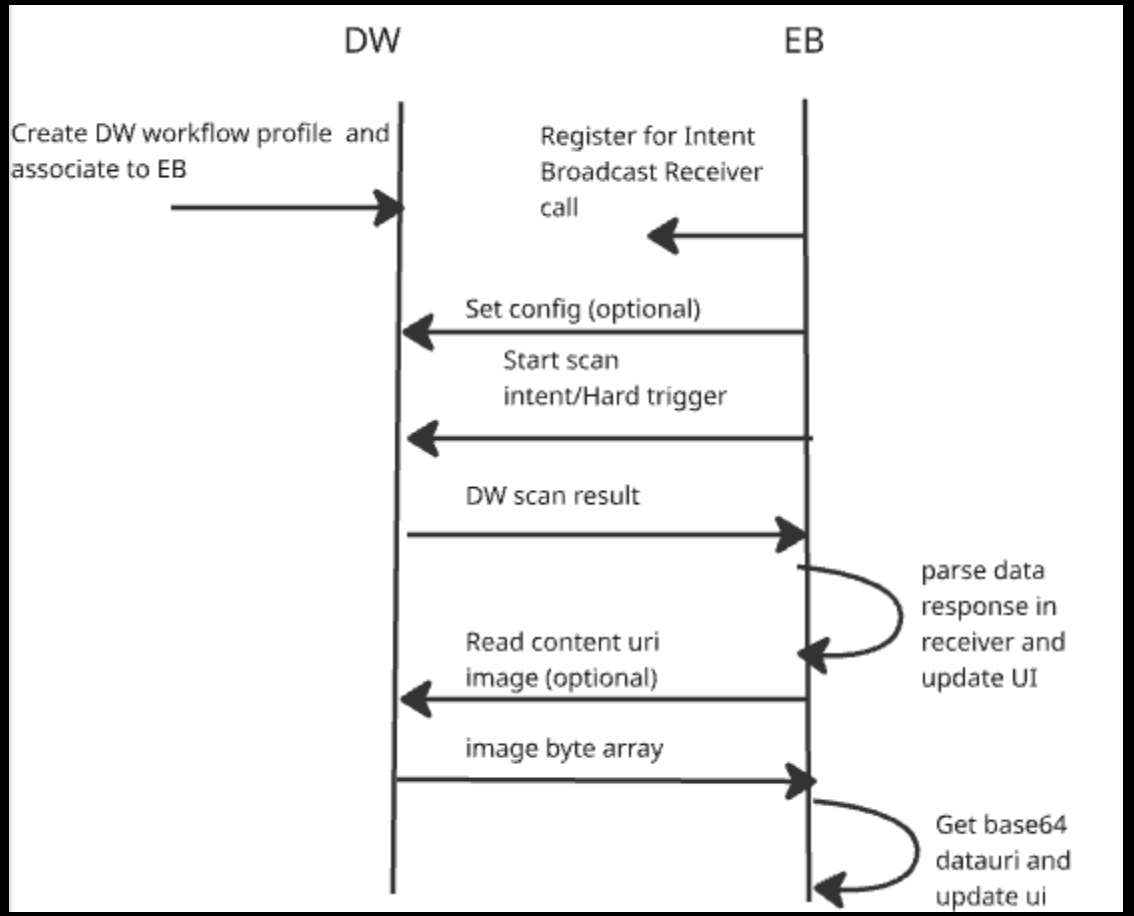
DevCon 2025
Connect | Learn | Build

DataWedgeIntentdemo.html

- License Plates (See supported countries/states.)
- Identification Documents (See supported document types by state, country, province.)
- Vehicle Identification Number (VIN)
- Tire Identification Number (TIN)
- Meter Reading







Capture data efficiently using EB

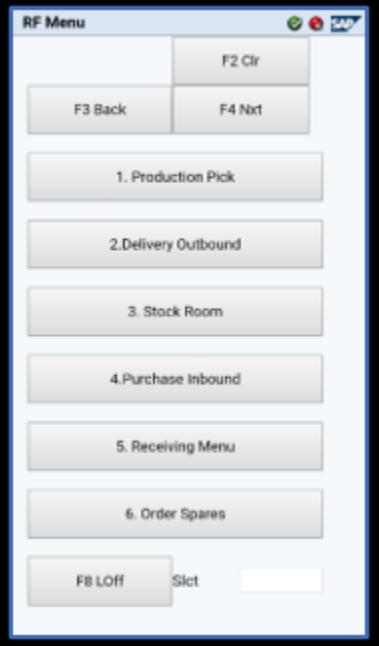
How can I create SQLITE database at a secured location on device for offline usage and then upload DB to server connectivity is back?

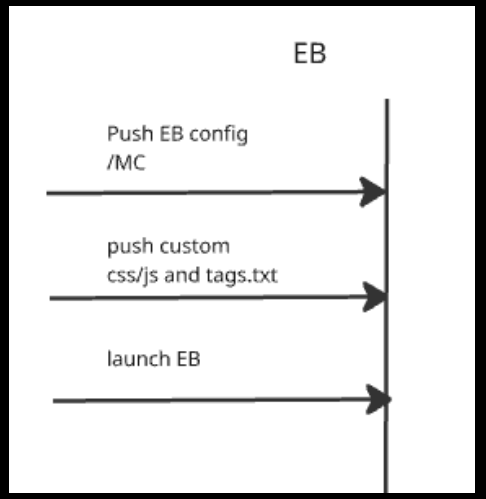

```
//create local db file
var db  = new EB.Database(EB.Application.databaseFilePath('testdb'), 'testdb');

//create a table in the db
db.executeSql('CREATE TABLE "' + tableName + '" (name TEXT,age INTEGER)');

//insert row
db.executeSql('INSERT INTO student (name, age) VALUES (?, ? );', ['sam', 20]);

db.close()

//check network and upload file

function checkNetworkAndUpload() {

        if( EB.Network.hasNetwork() )

            var f = EB.Network.uploadFile({
                  url: "http://10.233.82.114:8080/",
                  filename: "/data/user/0/com.zebra.mdna.enterprisebrowser/rhodata/db/syncdbtestdb.sqlite",
                  authType: "basic",
                  authUser: "admin",
                  authPassword: "password",
            },uploadFileCallback);

}
```

Capture data efficiently using EB

How do I change the style of my web page appearance on EB But I don't want modify the server code.

Customizing UI on EB
by injecting a stylesheet
to override.





Config.xml

```
<CustomDOMElements value="file://%INSTALLDIR%/mytags.txt
```

mytags.txt

```
<link rel='stylesheet' type='text/css' href='file://%INSTALLDIR%/mystyle.css'  pages='*' />
```

mystyle.css

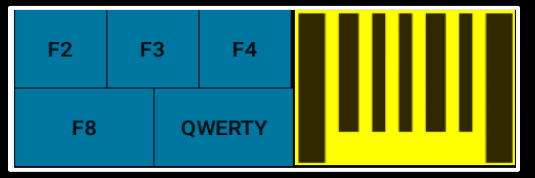
```
.MobileBody
{
  font-size:25px;
}
.MobileButton
{
 height: 100px;
}
```

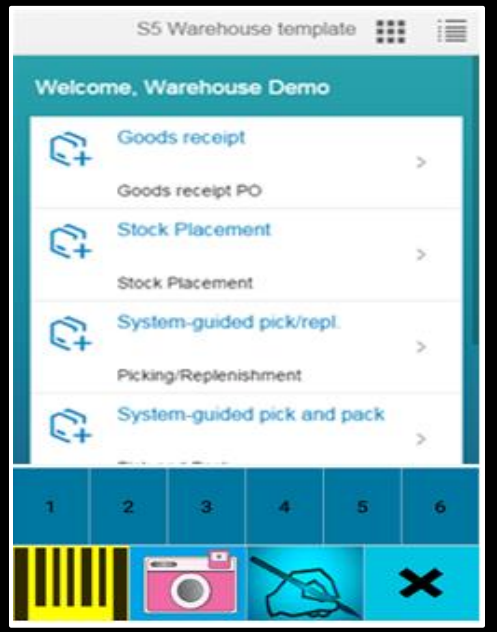
Capture data efficiently using EB

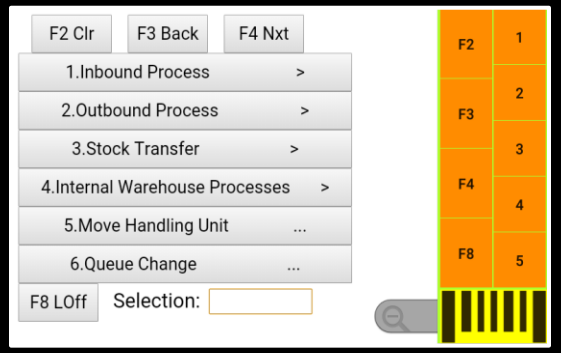
How can I use my custom Enterprise Keyboard (EKB) layouts in my web pages?

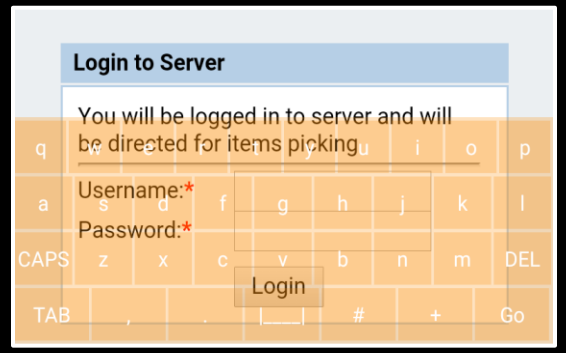
DevCon 2025
Connect | Learn | Build

```
function setlayout()
{
    var layoutGroupName = "mylayoutgroup";
     var layout = "FunctionKeyLayout";
    var bool = new Boolean(false);
    var data= {'CURRENT_LAYOUT_GROUP' : layoutGroupName, 'CURRENT_LAYOUT_NAME': layout};

    //create bundle with data
   var intentBundle = new createIntentBundle(EB.Intent.BROADCAST, "", "com.symbol.ekb.api.ACTION_UPDATE", "", "", "", "", "", data);

     //send intent to EKB
     EB.Intent.send(intentBundle, intentReceived);
}

var createIntentBundle = function (intentType, permission, action, categories, appName, targetClass, uri, mimeType, data)
  {
          var intentData = {};
          intentData.permission = permission;
          intentData.intentType = intentType;
          intentData.action = action;
          intentData.categories = categories;
          intentData.appName = appName;
          intentData.targetClass = targetClass;
          intentData.uri = uri;
          intentData.mimeType = mimeType;
          intentData.data = data;
         return intentData;
};
```





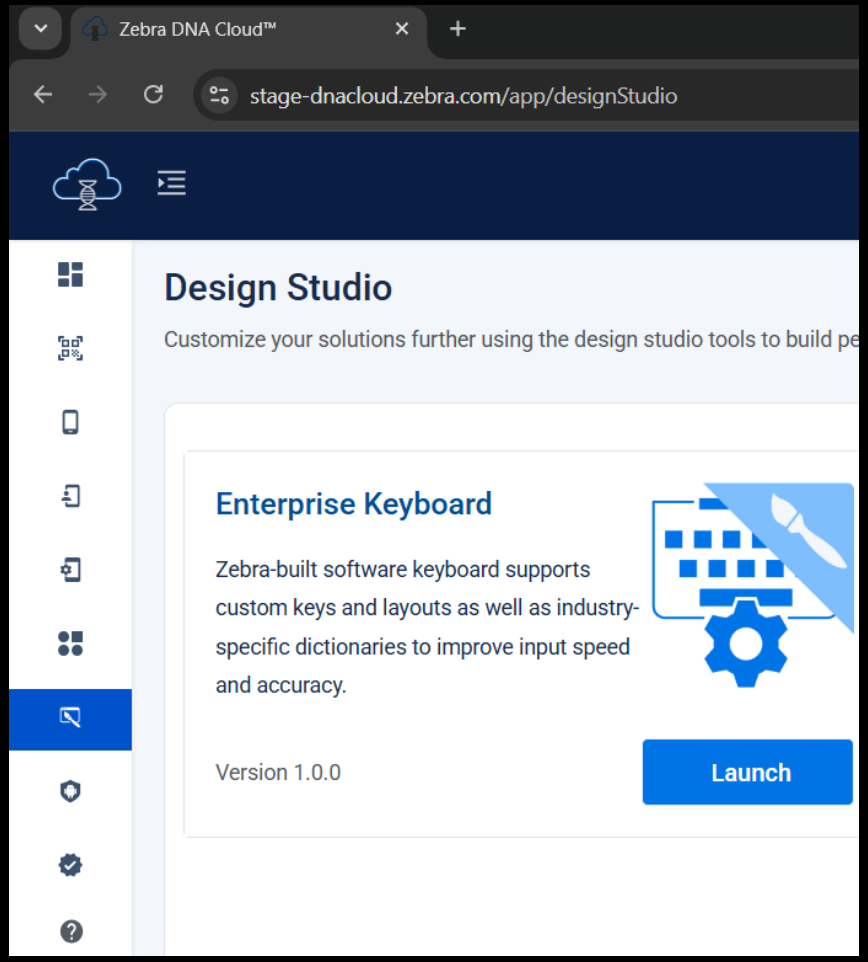


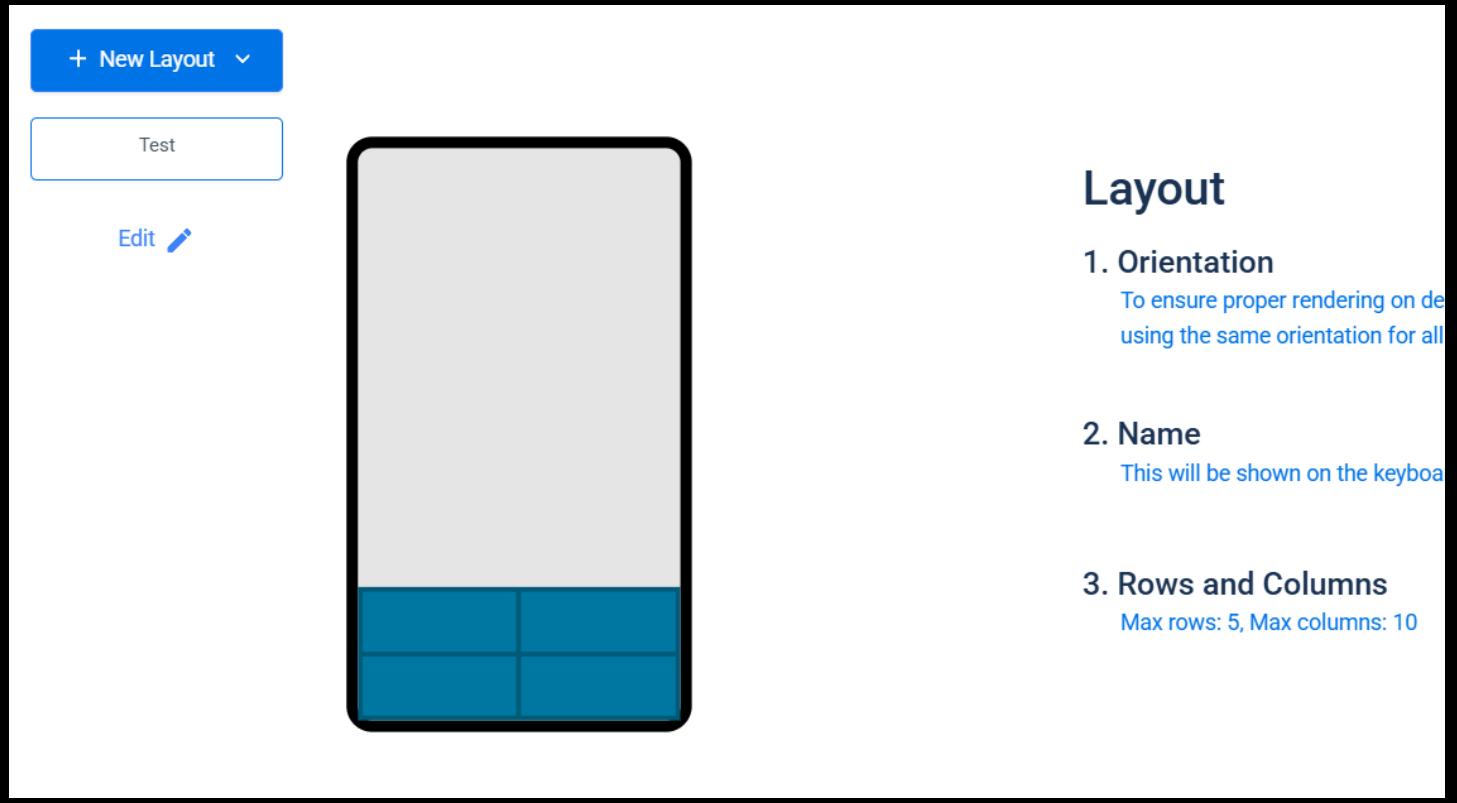


Capture data efficiently using EB

How to create customer EKB layouts? Use design studio in Zebra DNA cloud!

DevCon 2025
Connect | Learn | Build





DevCon 2025

Connect | Learn | Build

Some useful tips…

ZEBRA

Some useful tips …

I would like to auto grant some of the secure permissions so that my application will be able to perform required operation without prompting for “permission grating consents.”

Sensitive permissions - android.permission.MANAGE_EXTERNAL_STORAGE , android.permission.SYSTEM_ALERT_WINDOW , android.permission.PACKAGE_USAGE_STATS , android.permission.ACCESS_NOTIFICATIONS and all runtime permissions ex- READ_CONTACTS,READ_VIDEO etc

Zebra provides a mechanism for this purpose! Obtain permissions automatically

https://techdocs.zebra.com/mx/accessmgr/#permission-access-action

- An example showing auto-granting sensitive permission using Zebra Mx or Zebra EMDK SDK

```
<wap-provisioningdoc>
        <characteristic type="AccessMgr" >
        <parm name="PermissionAccessAction" value="1"/>
        <parm ame="PermissionAccessPermissionName"value="android.permission.MANAGE_EXTERNAL_STORAGE"/>
        <parm name="PermissionAccessPackageName" value="com.android.example"/>
        <parm name="PermissionAccessSignature" value="MIIDPjCCAiYCAQEwDQYJKoZIhvcNAQELBQAw" />
        </characteristic>
</wap-provisioningdoc>

Below XML to get all runtime permissions auto granted to an application

<wap-provisioningdoc>
        <characteristic type="AccessMgr" >
        <parm name="PermissionAccessAction" value="1"/>
        <parm name="PermissionAccessPermissionName" value="ALL_DANGEROUS_PERMISSIONS"/>
        <parm name="PermissionAccessPackageName" value="com.android.powerapplication"/>
        <parm name="PermissionAccessSignature" value="MIIDPjCCAiYCAQEwDQYJKoZIhvcNAQELBQAw" />
        </characteristic>
</wap-provisioningdoc>
```

Some useful tips …

I would like to start an activity/service from my application background components so that my application can do required operations from background;

- This is important because I want to launch an emergency alert
- This is important because I can perform action based on free fall detection , thread detection and Device vulnerability
- This is important because I can detect different events (such as SB card/USB insertion) and then perform action by displaying a message to user https://developer.android.com/develop/background-work/services/fgs/restrictions-bg-start

Solution - With the help of Zebra solution shown below, apps can bypass Android background restrictions (Restriction added from Oreo)

- https://techdocs.zebra.com/mx/intent/#action
- Sample Code – Start activity and broadcast from background using Zebra Mx XML interface

```
<wap-provisioningdoc>
        <characteristic type="Intent" version="10.5" >
        <parm name="Action" value="StartActivity"/>
        <parm name="ActionName" value="android.intent.action.MAIN"/>
        <parm name="Package" value="com.example.android"/>
        <parm name="Class" value="com.example.android.MainActivity"/>
        </characteristic>
</wap-provisioningdoc>
```

```
<wap-provisioningdoc>
        <characteristic type="Intent" version="10.5" >
        <parm name="Action" value="Broadcast"/>
        <parm name="ActionName" value="android.intent.action.RecieveData"/>
        <parm name="Package" value="com.example.android"/>
        <parm name="Class" value="com.example.android.Receiver"/>
        </characteristic>
</wap-provisioningdoc>
```

Some useful tips …

My app needs to share data and files with other applications in secure way

DevCon 2025
Connect | Learn | Build

Solution - With the help of Zebra secure storage manager, data/files can be shared between application securely.
https://techdocs.zebra.com/ssm/1-0/guide/use/

	Android Content provider	Secure storage manager
Standard content provider	Yes	Yes
Data sharing with other application	Yes	Yes
Data Sharing with authorized target by Admin	No	Yes
Data persistence	No	Yes
Secure data with encryption methodology	No	Yes
Automative data retrieval	No	Yes

Some useful tips …

Add Permissions in the Manifest

DevCon 2025
Connect | Learn | Build

- To insert / update / delete data

<uses-permission android:name="com.zebra.securestoragemanager.securecontentprovider.PERMISSION.WRITE"/>

- To query data

<uses-permission android:name="com.zebra.securestoragemanager.securecontentprovider.PERMISSION.READ"/>

- SDK 30 or above

<queries> <package android:name="com.zebra.securestoragemanager" /></queries>

- Authority URL for sharing data

"content://com.zebra.securestoragemanager.securecontentprovider/data“

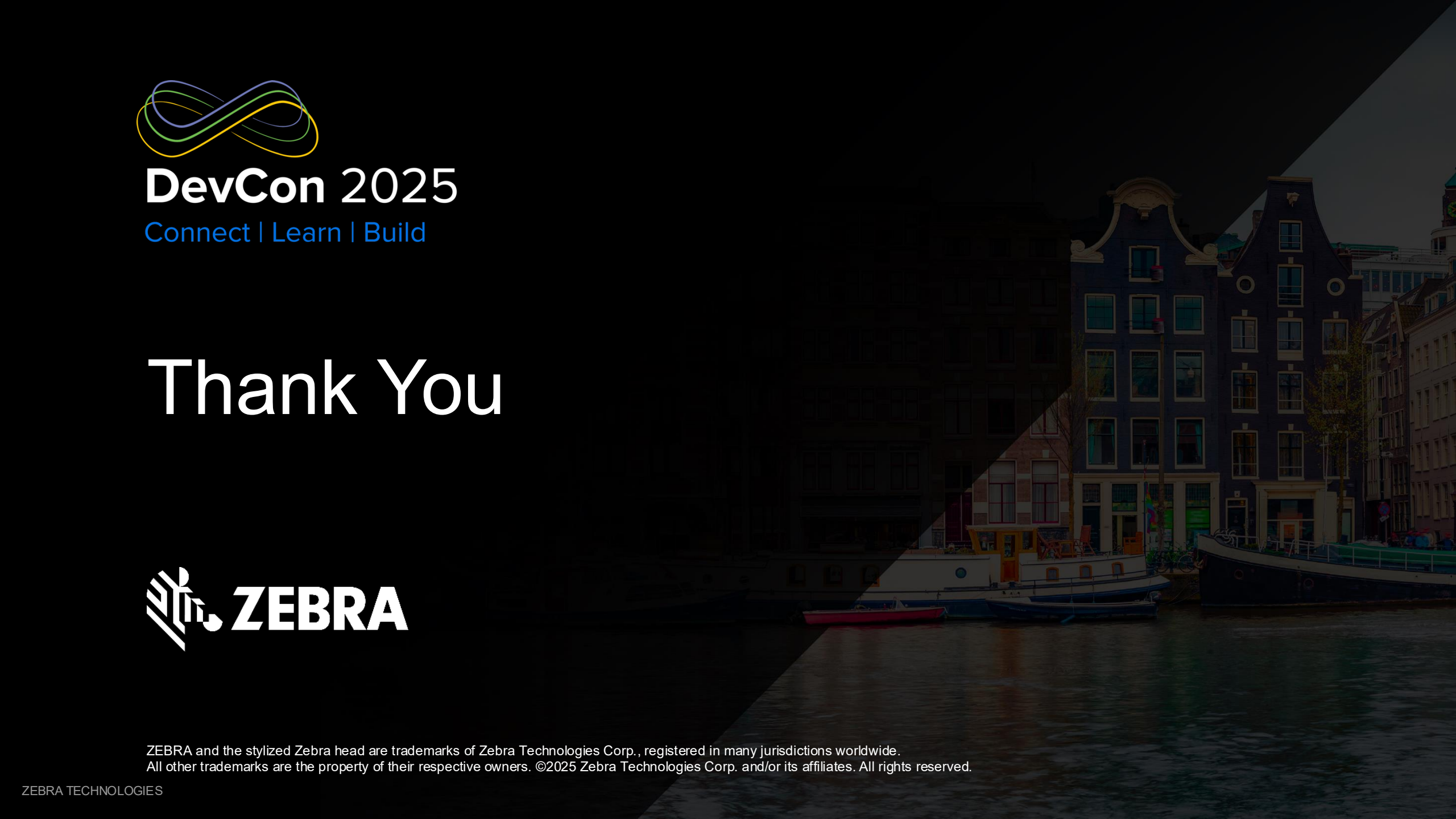
```
private String TARGET_APP_PACKAGE = "target_app_package";
private String DATA_NAME = "data_name";
private String DATA_VALUE = "data_value";
private String DATA_INPUT_FORM = "data_input_form";
private String DATA_OUTPUT_FORM = "data_output_form";
private String DATA_PERSIST_REQUIRED = "data_persist_required";
private String MULTI_INSTANCE_REQUIRED = "multi_instance_required";

AUTHORITY = "content://com.zebra.securestoragemanager.securecontentprovider/data";
Uri cpUri = Uri.parse(AUTHORITY);
ContentValues values = new ContentValues();

// TARGET_APP_PACKAGE gives both package name info and signature info in single entry. This can be either single target app or even mul
tiple target apps.

values.put(TARGET_APP_PACKAGE,
        "{\"pkgs_sigs\": [{\"pkg\":\"com.ztestapp.clientapplication\",\"sig\":\"ABSFFSDF… WREWED\"}]}");

// Dummy sig is placed here. use SigTool to get this base64 String.

values.put(DATA_NAME, “unique name to identify data in UTF-8 encoded format");
values.put(DATA_VALUE, “any string data/json data");
values.put(DATA_INPUT_FORM, "1"); //plaintext =1, encrypted=2
values.put(DATA_OUTPUT_FORM, "1"); //plaintext=1, encrypted=2, keystrokes=3
values.put(DATA_PERSIST_REQUIRED, "false");
values.put(MULTI_INSTANCE_REQUIRED, "false");
values.put(AUTO_DELETE_REQUIRED, "false");

Uri createdRow = getContentResolver().insert(cpUri, values);

Log.d(TAG, "Created row: " + createdRow.toString());
```


DevCon 2025
Connect | Learn | Build
Questions?

DevCon 2025

Connect | Learn | Build

Thank You



ZEBRA