Zebra DevCon 2023

ZEBRA

Zebra Handheld Readers RFID SDK Overview

Gary Crean

Zebra Handheld Readers RFID SDK Overview

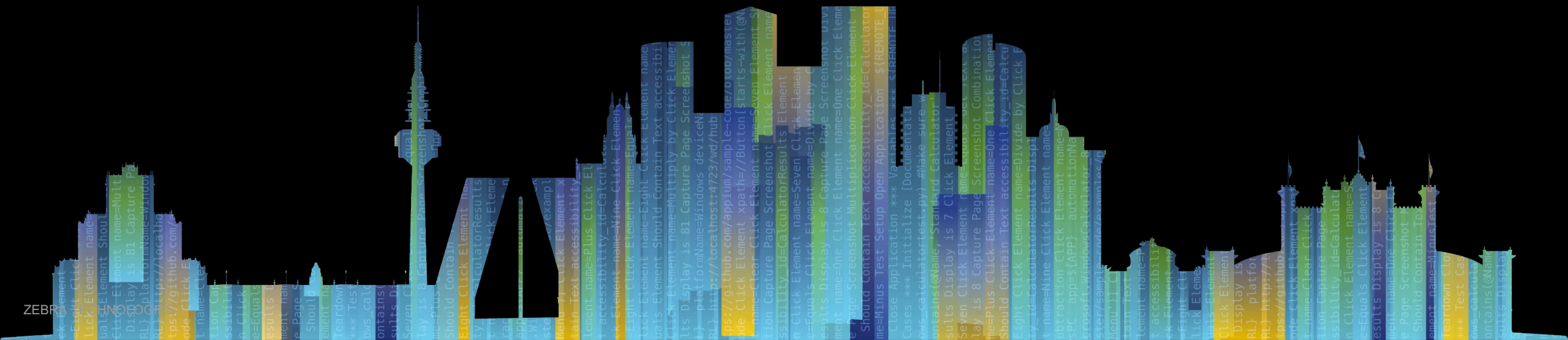
Zebra DevCon 2023

Agenda

- Hand held scanner portfolio
- Using the SDK

Zebra DevCon 2023

Zebra Handheld RFID Readers



Handheld Readers portfolio

RFD40/RFD90

- Sled Device
- Supported Platforms
  - Android
  - IOS
  - Windows
- Communication protocol
  - RFID ZETI
  - Barcode SSI
- Terminal Connectivity
  - Bluetooth
  - USB
- Device Supports
  - LED
  - Beep
- Software Tools support
  - 123RFID Mobile(IOS, Android)
  - 123RFID Desktop
  - IOS SDK
  - Windows SDK
  - Xamarin SDK
- Development Platform
  - Android
  - Windows
  - IOS

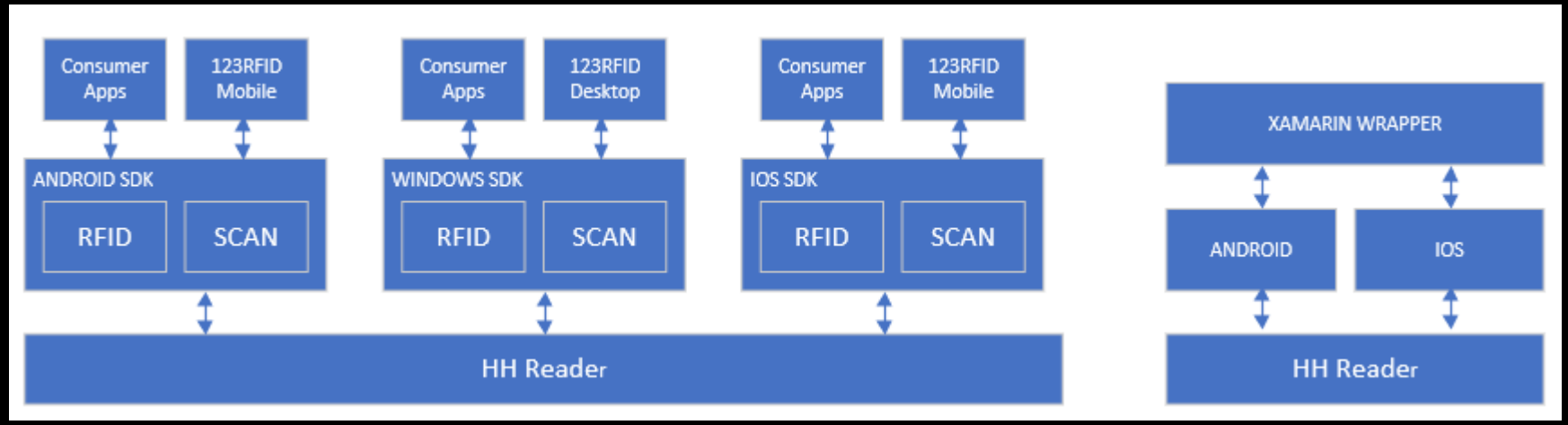
RFD8500

- Sled Device
- Supported Platforms
  - Android
  - IOS
  - Windows
- Communication protocol
  - RFID ZETI
  - Barcode SSI
- Terminal Connectivity
  - Bluetooth CDC
  - HID
- Software Tools support
  - RFID Manager
  - 123RFID Mobile
  - Scan control App
  - 123Scan
  - Android RFID/Scan SDK
  - IOS SDK
  - Windows SDK
  - Xamarin SDK
- Development Platform
  - Android
  - Windows
  - IOS

MC3300XR

- RFID Integrated Mobile computer
- QUALCOMM SDM660 chipset
- Supports integrated scanner
- Communication protocol
  - RFID ZETI
- Supported Platforms:
  - Android Q
  - Android R
- Software Tools support
  - RFID Manager
  - 123RFID Mobile
  - RFID Wedge
  - Android SDK
  - Xamarin SDK
- Development Platform
  - Android
  - Windows

Tools and SDK Support for HH Reader



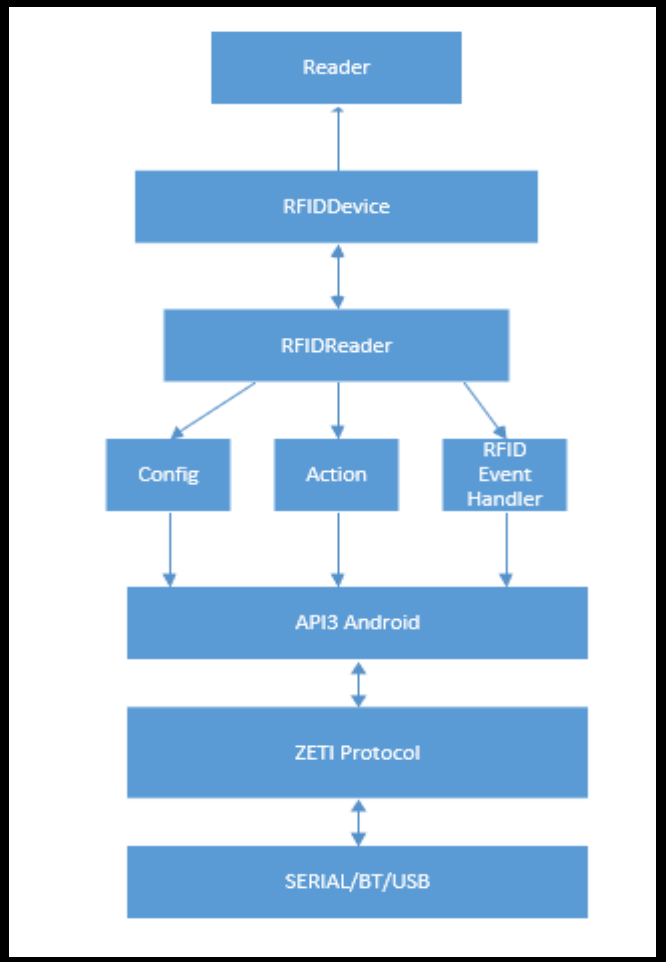
- SDK supported on following platforms
  - Android (RFD8500, MC33, RFD40)
  - IOS (RFD8500, RFD40)
  - Windows (RFD8500, RFD40)
- Xamarin wrapper for Windows development environment

Android SDK Architecture

Building Android RFID Application

Application development with Android SDK involves following main steps

1. Initialize transport type and get reader handle
2. Enumerate RFID Devices
3. Connect to the Reader
4. Configure/perform operations
5. Handle Status and data events
6. Disconnect the reader



Sample Code Snippet

```
/*Initialize and get the Reader handle */
Readers readers = new Readers (this, ENUM_TRANSPORT.ALL);

/*Discover RFID devices on all transport types*/
readerList = readers.GetAvailableRFIDReaderList();

/* define reader/transport available and not available notifications */
Readers.attach(this)

                              /* Connect to the reader */
try {
  readerList.get(0).getRFIDReader().connect();
} catch (InvalidUsageException e) {
  e.printStackTrace();
} catch (OperationFailureException e) {
  e.printStackTrace();
}
```

/* Define/override RFID Event listeners

```
@override
public void eventReadNotify(RfidReadEvents rfidData){
    /* Handle RFID Read Data and meta data
    Inventory/AccessData */
}

Public void eventStatusNotifications(RfidStatusEvents statusEvents){
    /*  Handle Status events/Notifications */
    /*Inventory, battery, trigger, etc */
}
```

/* disconnect the device */

```
try {
  readerList.get(0).getRFIDReader().disconnect();
} catch (InvalidUsageException e) {
  e.printStackTrace();
} catch (OperationFailureException e) {
  e.printStackTrace();
}
```

/* done with everything so dispose it off */

```
readers.Dispose();
```

Zebra Handheld Readers RFID SDK Overview

Singulation

Zebra DevCon 2023

Singulation refers to the method of identifying an individual/Group of Tags in a multiple-Tag environment.

```
// Get Singulation Control for the antenna 1
Antennas.SingulationControl singulationControl;
singulationControl =
reader.Config.Antennas.getSingulationControl(1);

// Set Singulation Control for the antenna 1
Antennas.SingulationControl singulationControl;
singulationControl.setSession(SESSION.SESSION_S0);
singulationControl.setTagPopulation((short) 30);
singulationControl.Action.setSLFlag(SL_FLAG.SL_ALL);
singulationControl.Action.setInventoryState(INVENTORY_STATE.I
NVENTORY_STATE_A);
reader.Config.Antennas.setSingulationControl(1,
singulationControl);
```

The function getSingulationControl retrieves the current settings of the singulation control from the reader for the given Antenna ID. To set the singulation control settings, the setSingulationControl method is used.

The following settings can be configured:

- Session: Session number to use for inventory operation.
- Tag Population: An estimate of the tag population in view of the RF field of the antenna.
- Tag Transit Time: An estimate of the time a tag typically remains in the RF field.
- State Aware Singulation Action: The action includes the Inventory state and SL flag. The action can be used if only the reader supports this capability.

Zebra Handheld Readers RFID SDK Overview

Inventory

Zebra DevCon 2023

1. Preparing

```
// tag event with tag data
reader.Events.setTagReadEvent(true); // application will collect tag
using getReadTags API
reader.Events.setAttachTagDataWithReadEvent(false);
 TriggerInfo triggerInfo = new TriggerInfo();
triggerInfo.StartTrigger.setTriggerType(START_TRIGGER_TYPE.START_TRIGGE
R_TYPE_IMMEDIATE);
triggerInfo.StopTrigger.setTriggerType(STOP_TRIGGER_TYPE.STOP_TRIGGER_T
YPE_IMMEDIATE);
 // set start and stop triggers
reader.Config.setStartTrigger(triggerInfo.StartTrigger);
reader.Config.setStopTrigger(triggerInfo.StopTrigger);
```

- A Simple continuous Inventory operation reads all tags in the field of view on all antennas of the connected RFID reader. It uses no filters (pre-filters or post-filters) and the start and stop trigger for the inventory is the default - immediate type. for example, start immediately when reader.Actions.Inventory.perform() is called, and stop immediately when reader.Actions.Inventory.stop() is called

2. Performing Inventory

```
try {
  // perform simple inventory
  reader.Actions.Inventory.perform();
   // Sleep or wait
   Thread.sleep(5000);
   // stop the inventory
   reader.Actions.Inventory.stop();
} catch (InvalidUsageException e) {
   e.printStackTrace();
 } catch (final OperationFailureException e) {
   e.printStackTrace();
 } catch (InterruptedException e) {
   e.printStackTrace();
 }
```

3. Processing the data

```
public void eventReadNotify(RfidReadEvents e) {
 TagData[] myTags =
reader.Actions.getReadTags(100);
        if (myTags != null) {
            for (int index = 0; index <
myTags.length; index++) {
            Log.d(TAG, "Tag ID " +
myTags[index].getTagID());
          }
        }
 }
```

Zebra Handheld Readers RFID SDK Overview

Tag Read and write

Zebra DevCon 2023

- Tag Access operations are performed on a specific tag or applied on tags that match a specific Access-Filter/Prefilter. If no Access-Filter/Pre-filter is specified, the Access Operation is performed on all tags in the field of view of chosen antennas.

Access Read

```
// Read user memory bank for the given tag ID
 String tagId = "1234ABCD00000000000025B1";
TagAccess tagAccess = new TagAccess();
TagAccess.ReadAccessParams readAccessParams = tagAccess.new
ReadAccessParams();
TagData readAccessTag;
readAccessParams.setAccessPassword(0);

 // read 4 words
 readAccessParams.setCount(4);
// user memory bank
readAccessParams.setMemoryBank(MEMORY_BANK.MEMORY_BANK_USER);

 // start reading from word offset 0
readAccessParams.setOffset(0);

 // read operation
 TagData tagData = reader.Actions.TagAccess.readWait(tagId,
readAccessParams, null);
```

Access Write

```
// write user memory bank data
TagData tagData = null;
String tagId = "1234ABCD00000000000025B1";
TagAccess tagAccess = new TagAccess();
TagAccess.WriteAccessParams writeAccessParams =
tagAccess.new WriteAccessParams();
String writeData = "11223344"; //write data in string
writeAccessParams.setAccessPassword(0);
writeAccessParams.setMemoryBank(MEMORY_BANK.MEMORY_BANK_USER
); writeAccessParams.setOffset(0); // start writing from
word offset 0 writeAccessParams.setWriteData(writeData); //
data length in words
writeAccessParams.setWriteDataLength(writeData.length() /
4); // antenna Info is null – performs on all antenna
reader.Actions.TagAccess.writeWait(tagId, writeAccessParams,
null, tagData);
```

Results and the operation details are populated into TagData structure

Zebra Handheld Readers RFID SDK Overview

Zebra DevCon 2023

```
// Add state aware pre-filter for given EPC or
Tag ID
private void addFilters(String tag) {
 // Add state aware pre-filter
 PreFilters filters = new PreFilters();
 PreFilters.PreFilter filter = filters.new
PreFilter();

// Set this filter for Antenna ID 1
filter.setAntennaID((short) 1);

filter.setTagPattern(tag);

// Tags which starts with passed pattern
filter.setTagPatternBitCount(tag.length() * 4);
filter.setBitOffset(32); // skip PC bits (always
it should be in bit length)
filter.setMemoryBank(MEMORY_BANK.MEMORY_BANK_EPC)
;
filter.setFilterAction(FILTER_ACTION.FILTER_ACTIO
N_STATE_AWARE);

// use state aware singulation
filter.StateAwareAction.setTarget(
        TARGET.TARGET_INVENTORIED_STATE_S1);
// inventoried flag of session S1 of matching
tags to B
filter.StateAwareAction.setStateAwareAction(

STATE_AWARE_ACTION.STATE_AWARE_ACTION_INV_B);

 // not to select tags that match the criteria
 try {
    reader.Actions.PreFilters.add(filter);
   } catch (Exception e) { e.printStackTrace(); }
 }
```

- pre-filters are applied prior to Inventory and Access operations.
- Pre-filters needs following actions
  - Add pre-filters
  - Set appropriate singulation controls
  - Perform Inventory or Access operation

Set Singulation

```
// Set the singulation control matching with prefilter
 private void setSingulationForFilter() {
  try {
     Antennas.SingulationControl s1_singulationControl
=reader.Config.Antennas.getSingulationControl(1);
      s1_singulationControl.setSession(SESSION.SESSION_S1);

s1_singulationControl.Action.setInventoryState(INVENTORY_STATE.INVENTORY_S
TATE_B);

s1_singulationControl.Action.setPerformStateAwareSingulationAction(true);
      reader.Config.Antennas.setSingulationControl(1,
s1_singulationControl);
 } catch (Exception e) { e.printStackTrace();
}
```

Perform inventory

```
try {

reader.Actions.Inventory.perform();
    Thread.sleep(5000);
    reader.Actions.Inventory.stop();
} catch (Exception e) {
e.printStackTrace(); }
```

Zebra Handheld Readers RFID SDK Overview

Antenna Settings

Zebra DevCon 2023

Add Prefilter

```
try {
// get the configuration
Antennas.AntennaRfConfig antennaRfConfig =
reader.Config.Antennas.getAntennaRfConfig(1);
antennaRfConfig.setrfModeTableIndex(0); antennaRfConfig.setTari(0);
antennaRfConfig.setTransmitPowerIndex(270);

 // set the configuration
reader.Config.Antennas.setAntennaRfConfig(1,antennaRfConfig);
 } catch (OperationFailureException ex)
 {
 Log.d(TAG, (" Antenna configuration failed " +
ex.getVendorMessage()));
}
```

The AntennaProperties is used to set the antenna configuration to individual antenna or all the antennas. The antenna configuration (SetAntennaConfig function) is comprised of Antenna ID, Transmit Power Index. These indexes are refers to the Transmit Power table, Tari Frequency Hop table, or Fixed Frequency table respectively

Zebra Handheld Readers RFID SDK

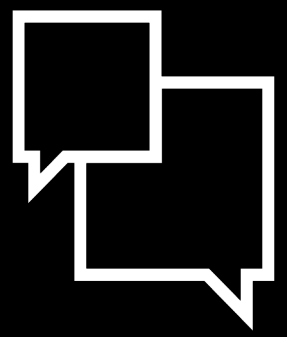
Resources

Zebra RFID Suppo rtand Downloads
https://www.zebra.com/us/en/support-downloads/rfid.html#text_e6a3

Zebra RFD40 Firmware, Utilities and Developer Tools
https://www.zebra.com/us/en/support-downloads/rfid/rfid-handhelds/rfd40.html

Zebra RFD90 Firmware, Utilities and Developer Tools
https://www.zebra.com/us/en/support-downloads/rfid/rfid-handhelds/rfd90.html

Zebra TechDocs (tutorials, samples, guides)
https://techdocs.zebra.com/dcs/rfid/



Questions

Zebra DevCon 2023

ZEBRA

Thank You