RapidScan

Quick Start Guide

May 2023

Table of Contents

Introduction

This guide describes how to use the RapidScan application, built specifically to run on the HaloRing. This application utilizes the RISL programming language which allows a remote system to display custom messages onto the HaloRing's screen. RapidScan currently supports receiving these messages using the following: over Wi-Fi using REST or MQTT and over Bluetooth using BLE or SPP.

If you do not have a remote system set up, we provide a couple out-of-the-box options meant for testing and demo purposes. These will be covered at the end of this document.

Note: This document is meant to cover the basics of RapidScan. For a more in-depth look, there are additional documents that cover the RISL language, the supported connection types (REST, MQTT, BLE, SPP) and any available demos.

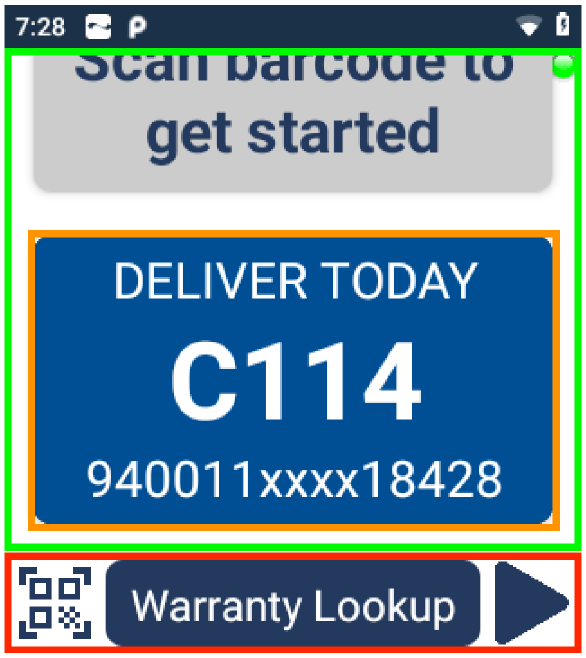
Pre-Requisites

- HaloRing with the RapidScan application installed
- Connection to a supported remote system. This can be either:
    - REST server via Wi-Fi
    - MQTT server via Wi-Fi
    - GATT server via BLE
    - iOS or Android application w/ companion framework
    - SPP server via Bluetooth
- Alternatively, connection to one of the following demo systems:
    - MQTT dashboard via Wi-Fi **(demo purposes only)**
    - iOS/Android companion demo via BLE **(demo purposes only)**

Application Layout

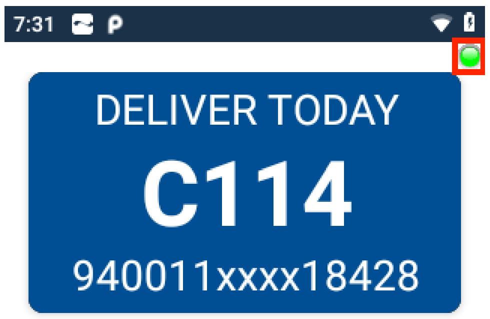
The application UI can be broken into 2 main sections. The bottom section highlighted below in red is the action menu and the upper section highlighted in green is the card section.

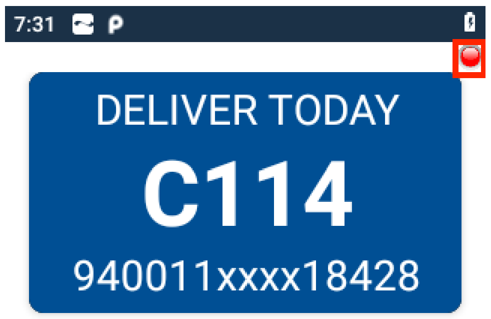
Any RISL messages you receive will be shown in the card section by being inserted at the bottom of a scrollable list. This list allows you to scroll up to see past messages if necessary. The area highlighted in orange below is a single RISL card.



Connection Status

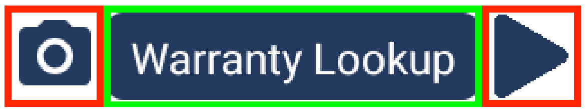
The connection status indicator is shown below highlighted in red. When the indicator is green it means RapidScan is currently connected. Depending on the configuration this could mean to a Bluetooth device, MQTT server or just the internet. Red indicates that RapidScan is not connected.





Action Menu

The action menu can be further broken down into 2 sections. The area highlighted in green indicates the current action and the icons highlighted in red indicate what the HaloRing's hardware buttons will do in the current state.

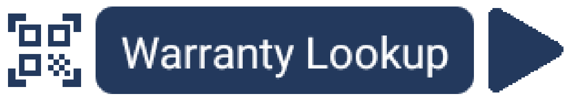


Hardware Buttons

The HaloRing has 3 hardware buttons that are used to easily interact with and traverse the action menu. The icon on the left side indicates what the left hardware button will do, and the right icon indicates what the right hardware button will do. The center hardware button is used for scanning, picture taking and menu selection.

The action menu above is an example of what you would see on the main screen. The right arrow icon indicates that the right hardware button will iterate to the next action menu item. The left camera icon is showing that the current mode is "camera mode". This means that if you press the center button in this mode the camera will be activated.

To toggle to "scan mode" simply press the left button so that the icon now shows the barcode icon like in the image below. Now the middle button will activate the scanner instead.

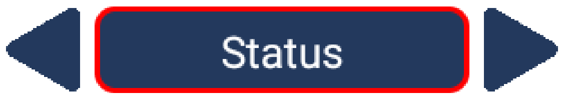


NOTE:

When adding actions to RapidScan you can specify whether the action is Scan Only, Camera Only, or Both. If the action can do Both, then the ability to toggle between modes acts the way that is described above. If the action is Scan or Camera only, then there will be no toggle and the left button will just be an arrow.

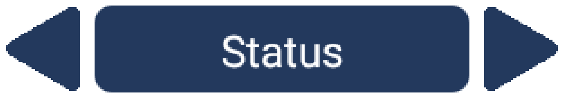
Changing Actions

As stated earlier, traversing the action menu is done by clicking the buttons with arrows over them. When pressing one of these buttons you will scroll one step in that direction and enter "action selection mode". This is indicated with a red ring around middle box like in the image below.



When in this mode the left icon becomes an arrow allowing you to use both the left and right hardware buttons to scroll freely in any direction between actions.

When you are ready to select an action click the middle button and the red ring will disappear and action shown will become your "current action" like in the image below.



What Makes Actions Powerful?

By adding the current action to any barcode scan or image capture payload, actions can effectively allow you to scan in different "modes".

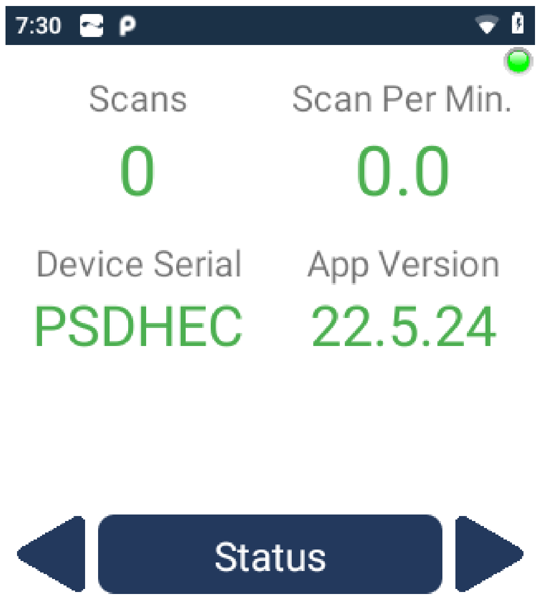
Example:

Working the entrance of a concert you scan either patron tickets or driver's licenses. When scanning a ticket, the "Ticket" action is active so the backend system knows the barcode is supposed to be a ticket and can verify it accordingly. When you need to scan a license, you switch to the "License" action. Now, the backend can run a different set of business logic to confirm the license is valid and give you the appropriate RISL response.

Status & Settings Screens

Aside from the “main” screen there are two other screens that can be accessed via the action menu. They are the Status and Settings screens. Your app will automatically navigate to these screens when their corresponding action is selected in the action menu. To go back to the main screen, simply select another action.

Note: By default, these screens are not exposed. If you want them to be exposed to your users, you need to add their actions to your app’s action menu via app configuration. This will be covered in the following section.

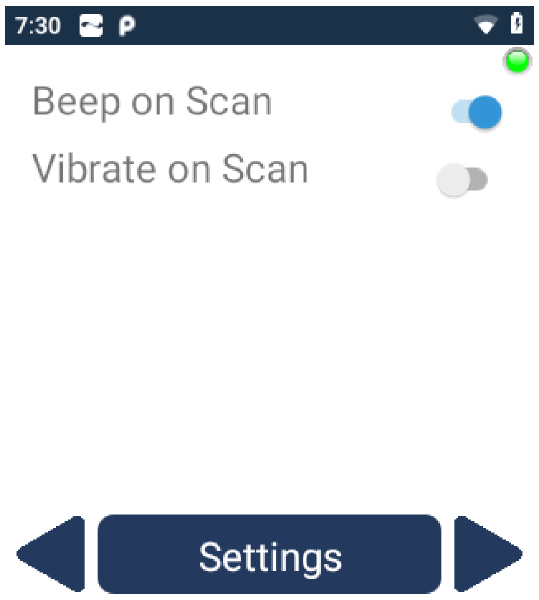

7:30

Scans
0

Scan Per Min.
0.0

Device Serial
PSDHEC

App Version
22.5.24


7:30

Beep on Scan: on

Vibrate on Scan: off

The image to the left is the Status screen. In this screen you can find some useful information such as your device’s serial, the app’s version and some scanning statistics.

The image to the right is the settings screen. Here you will be able to toggle some app or hardware settings. This will likely change in the future to expose more app settings.

Exit

The last special action is the “Exit” action. When selected the application will close.

Note: Just like the Status and Settings actions above, the Exit action is not exposed by default. It needs to be added to the action menu via app configuration.

App Configuration

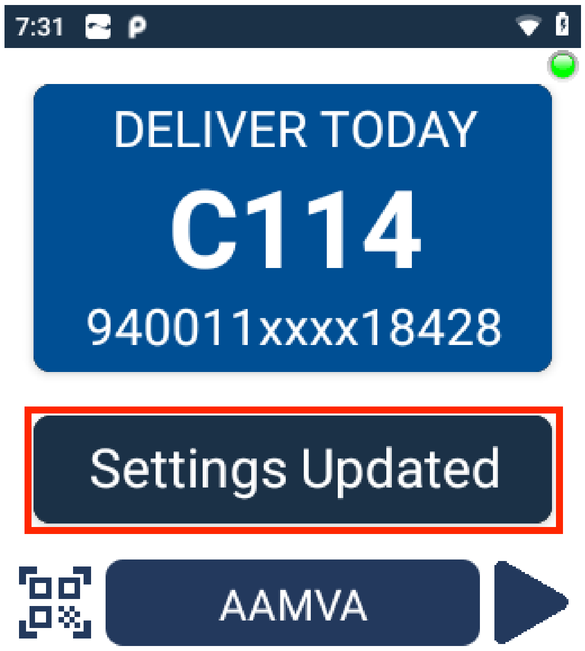
To configure RapidScan's settings, you will need have your app consume an app configuration payload which is in the standard JSON format. This can be accomplished by either scanning a QR code containing the payload or over WiFi/Bluetooth. The format for either method is the same.

For QR code: take the text of your configuration JSON and create a QR code using that text. Scan it with RapidScan in the foreground and your settings should be set.

```
{
    "action": "config",
    "command": {
        // App settings go here
    }
}
```

For REST/MQTT/BLE/SPP: simply send your configuration JSON to RapidScan as a string.

For either method you will know your settings were consumed if you get a message that looks like the following highlighted in red below.



Configuration JSON Format

The configuration payload follows standard JSON format as shown below. You are free to add as many or as few of the settings as you want. Just make sure you have the outer JSON structure.

```json
{
    "action": "config",
    "command": {
        "mode": "mqtt"
        "mqttServer": {
            "host": "192.168.0.1",
            "username": "relay",
            "password": "relay",
            "port": 1883,
            "topic": "risl"
        },
        "verbs": [
            {
                "name": "Scan Package",
                "filter": "trackingNum",
                "mode": 1
            },
            {
                "name": "Package Damage",
                "mode": 2
            },
            {
                "name": "Status"
            },
            {
                "name": "Settings"
            },
            {
                "name": "Exit"
            }
        ],
        "maxCards": 10,
        "scanBeep": false,
        "scanVibrate": true
    }
}
```

Configuration Fields

mqttServer

This object has all MQTT related settings. Host, username, password and port all correspond to the MQTT server credentials. The “topic” will be the prefix for the topics you subscribe and publish to. If this is not defined, the default topic prefix is “risl”.

So, if using MQTT, your device will be subscribed to the following topics:

{topic}/{serial}/in
{topic}/broadcast/in

And your device would publish to the following topic:

{topic}/{serial}/out

Example of this:

Subscribe ->	risl/V3WJSD/in
	risl/broadcast/in
Publish ->	risl/V3WJSD/out

verbs

The verbs array is your action list. Every action that you want in your action menu needs to be added to this array. Each verb object can have a name, filter, and mode field with filter and mode being optional. Details on the fields are below:

name
Name of action

filter (optional)
Allows you to specify a barcode filter which allows you to restrict the barcodes scanned to those that pass a specified format/type ruleset. These barcode filters are set through Q Wedge. If you want more info on how barcode filters work, be sure to check the Q Wedge documentation.

mode (optional)
Allows you to specify RapidScan capabilities on a per action basis. Possible values are 1 = Scan Only, 2 = Camera Only and 3 = Both. The default is Scan Only.

Reminder: Earlier in this document we mentioned the special actions Status, Settings, and Exit. If you would like any of these actions present remember to add them to your verb array. Below is an example of an app config JSON that sets a full list of actions.

```
{
    "action": "config",
        "verbs": [
            {
                "name": "Scan Package",
                "filter": "trackingNum", // Use specified barcode filter
                "mode": 1 // Scan only
            },
            {
                "name": "Package Damage",
                "mode": 2 // Camera only
            },
            {
                "name": "AAMVA",
                "mode": 3 // Both Scan and Camera
            },
            {
                "name": "Status" // Allows you to navigate to the Status screen
            },
            {
                "name": "Settings" // Allows you to navigate to the Settings screen
            },
            {
                "name": "Exit" // Selecting this exists the app (useful in kiosk mode)
            }
        ]
    }
}
```

currentVerb

currentVerb is used to set your current action. This is a convenience method that allows you to choose an action without needing to scroll through the action menu. The main purpose of this is to use barcodes or MQTT/SPP messages to streamline workflows.

Example: The QR code above contains the following app configuration JSON. Remember since it's a barcode we prefix with "RapidScan".

```
RapidScan {
    "action": "config",
    "command": {
        "currentVerb": "AAMVA"
    }
}
```

If scanned, RapidScan will check to see if you have an action in your action menu with the name "AAMVA". If you do, it will automatically switch your current action to that.

maxCards

This allows you to define the maximum number of cards RapidScan can display at once. Remember, the RISL cards appear at the bottom of a scrollable list view. This setting essentially allows you to put a card limit to that list. The default value for this is 50.

scanBeep & scanVibrate

When toggled on these will make RapidScan beep or vibrate respectively when a barcode is scanned. There may be situations where you would want one or both settings off, such as when you are relying on RISL messages to give your user audio or haptic feedback.

Example:

You only want to vibrate your Halo when a “bad” scan takes place. So, you turn off scanVibrate so you can rely on the RISL message to provide that haptic feedback.

Outgoing MQTT Payloads

The following is a list of all possible outgoing MQTT payloads. Lines 2-3 are saying that the 4 fields listed (serial, device, bundle and utcTime) will be in every outgoing payload.

```
1.  /**
2.   * Any of the following payloads will have these
3.   * 4 fields in addition to those listed below
4.   **/
5.  {
6.      "serial": "V3WJSD",
7.      "device": "DRS-50",
8.      "bundle": "com.ipc.haloring.rapidscan",
9.      "utcTime": "2022-05-09T09:15:03-0700"
10. }
11.
12. // RapidScan Device Connected (device successfully connected to server)
13. {
14.     "event": "imAlive"
15. }
16.
17. // Barcode Scan
18. {
19.     "event": "barcode",
20.     "value": {
21.         "barcode": "123456",
22.         "type": 1
23.     },
24.     "verb": "Warranty Lookup"
25. }
26.
27. // Image Capture (user captures an image)
28. {
29.     "event": "image",
30.     "value": {
31.         "base64": "base64Image"
32.     },
33.     "verb": "Damaged Package"
34. }
35.
36. // Action Selection (user selects an action from action menu)
37. {
38.     "event": "verbSelected",
39.     "value": {
40.         "name": "Warranty Lookup"
41.     }
42. }
43.
44. // RISL Button Response (user presses a button from a RISL message)
45. {
46.     "event": "buttonPress",
47.     "value": {
48.         "button": "msgConfirm" // Key of the button that was pressed
49.     }
50. }
51.
52. // RISL Card Exists Response (RISL command to check if card is saved to device)
53. {
54.     "event": "cardExists",
55.     "value": {
        "exists": true // True if the saved card exists on the device
    }
}
```

Incoming MQTT Payloads

There are currently only 2 possible incoming messages: “config” which was covered earlier in the App Configuration section and “risl” which is how RapidScan receives RISL cards.

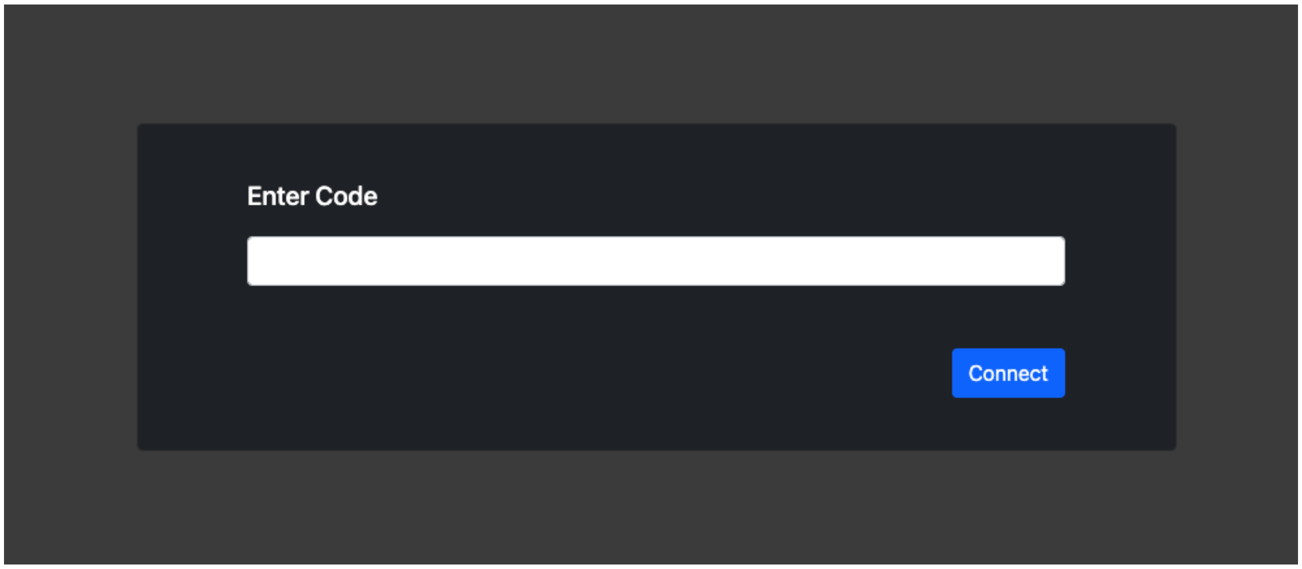
```
// RISL Action
{
    "action": "risl",
    "command": "^StartCard|290|70^TextC|4|Hello World!^ShowCard"
}

// Config Action
{
    "action": "config",
    "command": {
        // App settings go here
    }
}
```

MQTT Dashboard

If you do not have a MQTT or SPP server set up yet a good place to start is with our MQTT dashboard. This tool allows you to demo RapidScan functionality out of the box with almost no setup. Just make sure your HaloRing is connected to Wi-Fi and you should be good to go.

The MQTT dashboard is currently hosted here: https://rapidscan.ipcmobile.com/

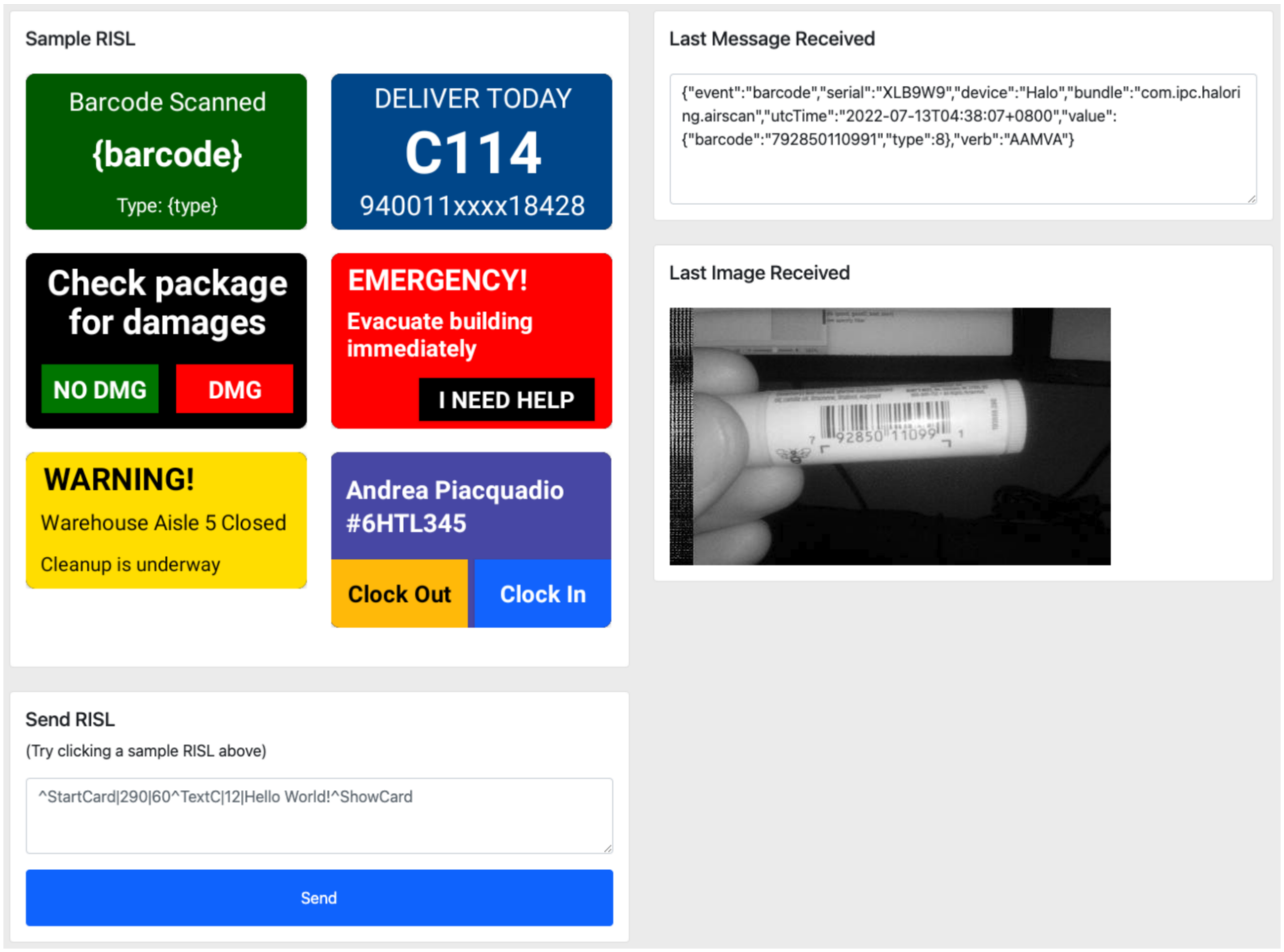


When you first navigate to the dashboard you should see something like the image above. The code the dashboard is asking for is your device's serial number which can be found by navigating to the "Status" screen on RapidScan.

Remember this screen isn't enabled by default so you will need to add it as an action to your action list. You can use the following app config barcode to set an action list with "Status" exposed.

Using the Dashboard

Once you type your device's serial and press "Connect" you will be presented with the main dashboard shown below.



Sample RISL

Barcode Scanned
{barcode}
Type: {type}

DELIVER TODAY
C114
940011xxxx18428

Check package for damages
NO DMG
DMG

EMERGENCY!
Evacuate building immediately
I NEED HELP

WARNING!
Warehouse Aisle 5 Closed
Cleanup is underway

Andrea Piacquadio
#6HTL345
Clock Out
Clock In

Last Message Received

```
{"event":"barcode","serial":"XLB9W9","device":"Halo","bundle":"com.ipc.halori
ng.airscan","utcTime":"2022-07-13T04:38:07+0800","value":
{"barcode":"792850110991","type":8},"verb":"AAMVA"}
```

Last Image Received

Send RISL
(Try clicking a sample RISL above)

```
^StartCard|290|60^TextC|12|Hello World!^ShowCard
```

Send

The right side of the dashboard is where your received MQTT messages are shown. Images well be displayed in the "Last Image Received" section and all other messages like barcode scans and verb selections will be displayed in the "Last Message Received" section.

On the left side of the dashboard is a list of sample RISL cards with images of what each card will look like in RapidScan. When you click on one of these images the sample RISL code used to create the card will be populated into the text field below. This allows you to see the commands used to make each card. Before you click "Send" you can edit the commands or even make a completely new RISL from scratch! This is a great sandbox feature you can use to play with the language.

Once you click "Send" the RISL message will be sent over MQTT. If your device and the dashboard are both connected MQTT and you used the correct serial, your RapidScan app should receive your message!