AMERICAS APPFORUM 2019 | ELEVATING ENTERPRISE INTELLIGENCE

ZEBRA CAPTURE YOUR EDGE

Using JavaScript Frameworks when developing for Zebra Mobile Computers

Darryn Campbell
SW Architect, Zebra Technologies
@darryncampbell
October 1st / 2nd 2019

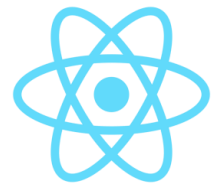
Using JavaScript Frameworks on Zebra Devices

Agenda

- Landscape of JavaScript development on Zebra Devices
- Some possible options for JavaScript developers on Zebra Devices

Using JavaScript Frameworks on Zebra Devices

Options for JavaScript developers



Using JavaScript Frameworks on Zebra Devices

Options for JavaScript developers

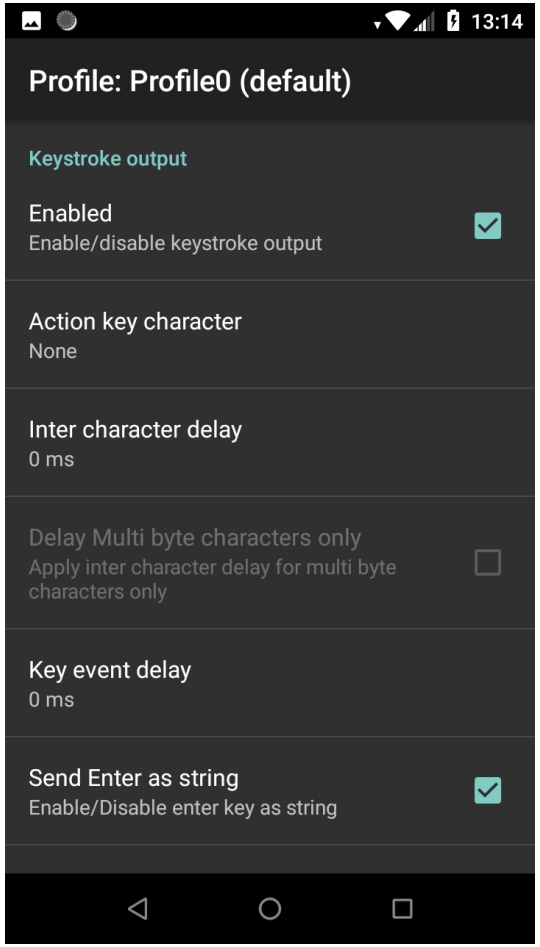
- **Different classes of “JavaScript frameworks” – nomenclature may vary**
- Run entirely within the Browser: Look and feel and perhaps MVC
  - E.g. Jquery, Angular, Vue.js, React.js
- Offer ‘Cross Platform’ functionality rendered in a webview
  - E.g. Enterprise Browser, Cordova (Phonegap), Ionic, Rho
- Offer ‘Cross Platform’ functionality with native controls
  - E.g. React native, NativeScript
- Some combination of the above
  - E.g. EB + Vue.js or Ionic v4 + React.js
- **Which of these does Zebra support?**

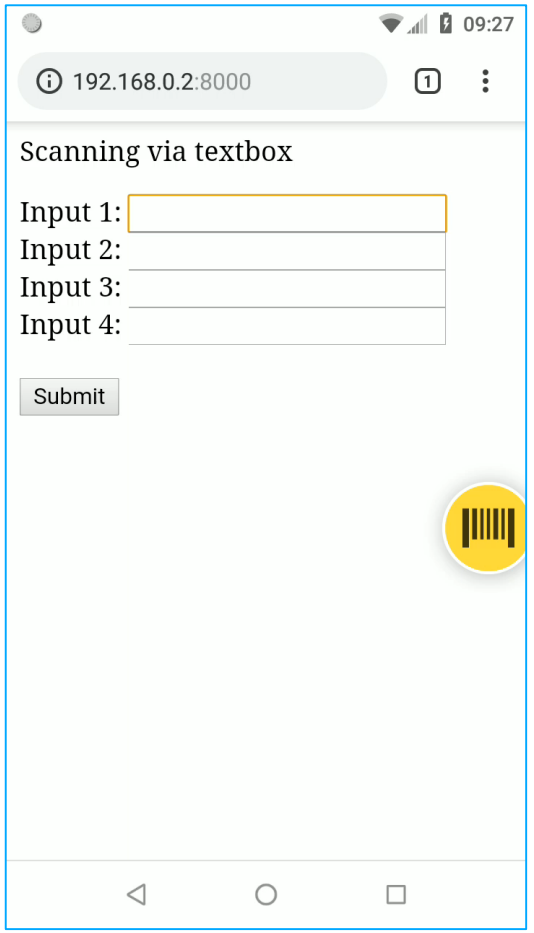
Using JavaScript Frameworks on Zebra Devices

Option 1: DataWedge Keystroke Output Plugin

- “Traditional” DataWedge with Keystroke output
- Append tab key with BDF
- Scanner works *like* a HID connected devices
- Text boxes must have focus, not tolerant of users clicking away from the intended textbox
- Keyboard will be visible when textbox gets focus

```
<form>
Input 1: <input type="text" id="input1"><br>
Input 2: <input type="text"><br><br>
<input type="submit">
</form>
```

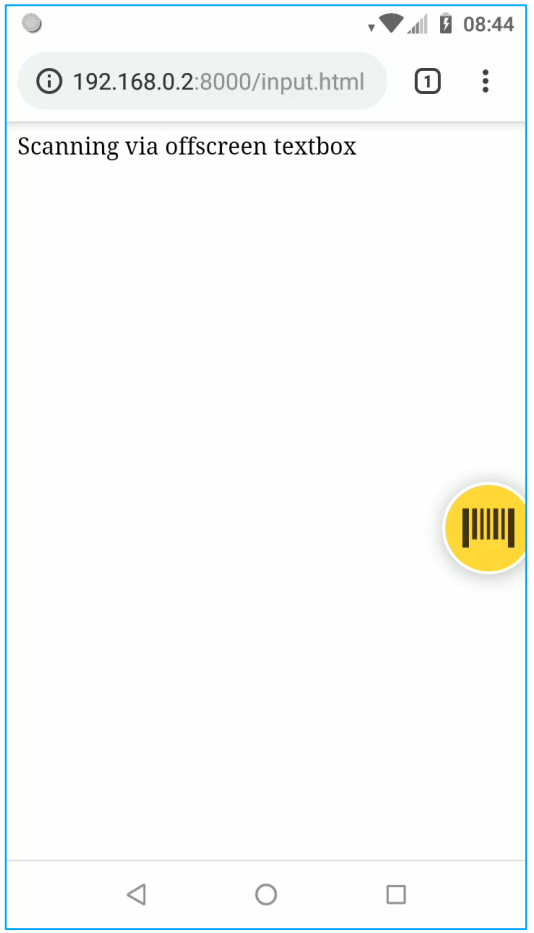




Using JavaScript Frameworks on Zebra Devices

Option 2: DataWedge Keystroke Output Plugin with offscreen text box

- Use DataWedge with Keystroke output plugin but **hide textbox off screen**
- Keyboard may still pop up
- Allows you to handle keyboard input in JavaScript, but a bit of a hack.



HTML:

```
<input type="text" class="scanner-input">
```

CSS:

```
.scanner-input{position:absolute;left:-10000px;}
```

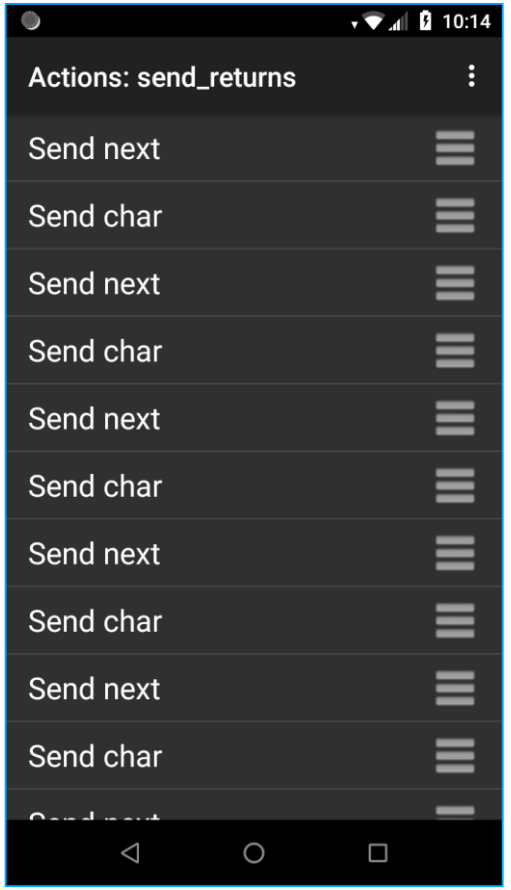
JS:

```
$(".scanner-input").focus().on("input",function(){
    let barcode = $(this).val();
    scanOutput.innerHTML = barcode;
    $(this).val("");
  }).blur(function(){
    $(this).focus();
});
```

Using JavaScript Frameworks on Zebra Devices

Option 3: DataWedge Keystroke Output Plugin with inter-character return

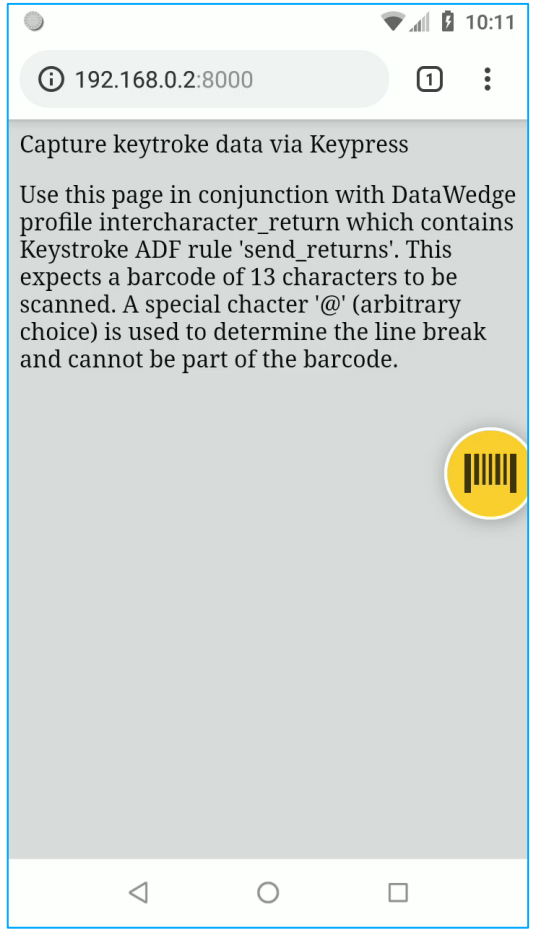
- Frequent question on the developer portal: “How can I use Chrome with DataWedge?”
- Frequent answer on the developer portal: Some amalgamation of
  - Use the Keypress event
  - Introduce an intercharacter delay
  - Use ADF to send a return (0x0D) after each char
- Keypress / intercharacter delay are not 100% reliable and may depend on your webview / browser version. Use of ADF will work on all devices
  - Root cause: DataWedge keys are not sent as key presses in the same manner as a HID device



Using JavaScript Frameworks on Zebra Devices

Option 3: DataWedge Keystroke Output Plugin with inter-character return

- Specify Advanced Data Formatting (ADF) for the Keystroke output plugin
- Create a rule to send an enter after each character in the barcode:
  - Criteria: All barcodes
  - Actions: Send next / Send char (0x0D) / Send next / Send char (0x0D) ……
- Downsides:
  - Requires foreknowledge of the *maximum* barcode size
  - UI can be quite fiddly to enter the entire rule – could use the DataWedge API or provision devices with a pre-configured profile
- Register for keypress event in JavaScript
- Does not show keyboard
- Still a bit of a ‘hack’



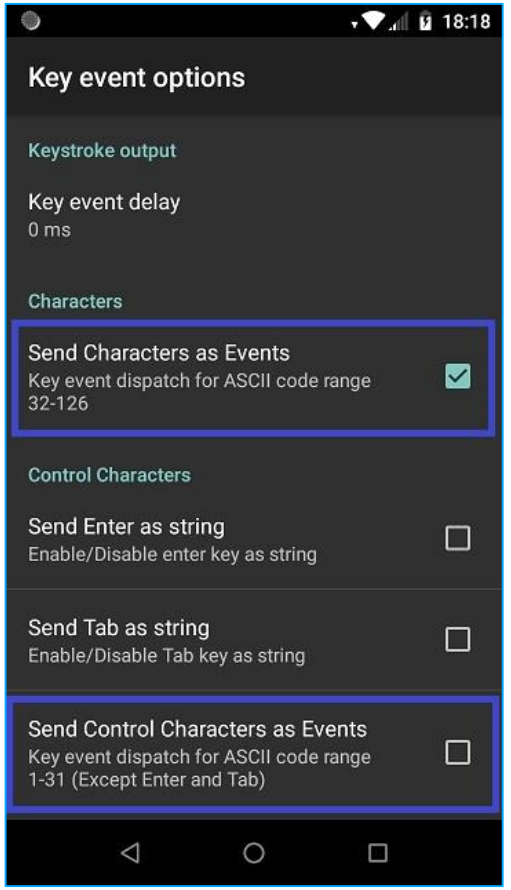
JS:

```
document.addEventListener('keypress’, handleKey);
```

Using JavaScript Frameworks on Zebra Devices

Option 4: DataWedge 7.3 Key Events and KeyPress

- Newly introduced feature in 7.3. *Frequent customer request*
- Keystroke output plugin → Key event options → Enable “Send Characters as Events”
- Enables you to receive data as though the user is typing it **without** using an input field
- Downsides:
  - Tricky to determine the barcode length
  - Only available in DataWedge 7.3 or higher
- Register for keypress event in JavaScript
- Does not show keyboard
- Same technique you would use for HID scanners



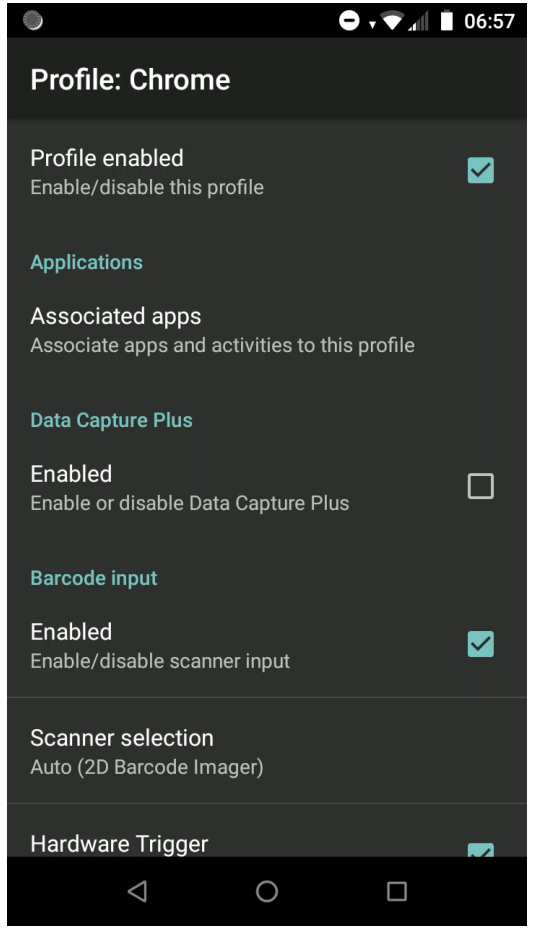
Using JavaScript Frameworks on Zebra Devices

Option 4: DataWedge 7.3 Key Events and KeyPress

- More information:
  - https://developer.zebra.com/blog/listening-keypress-events-datawedge
  - https://github.com/darryncampbell/DataWedge-KeyEvent-Options
- Remember: Android will not generate a KeyPress event for all keys (e.g. tab)

JS:

```
document.addEventListener('keypress', keypressHandler);
function keypressHandler(e) {
  const keypressoutput = document.getElementById('pressed_keys');
  if (e.keyCode == 13) // Enter key from DataWedge
    keypressoutput.innerHTML += "<BR>";
  else
     keypressoutput.innerHTML += e.key;
}
```



Using JavaScript Frameworks on Zebra Devices

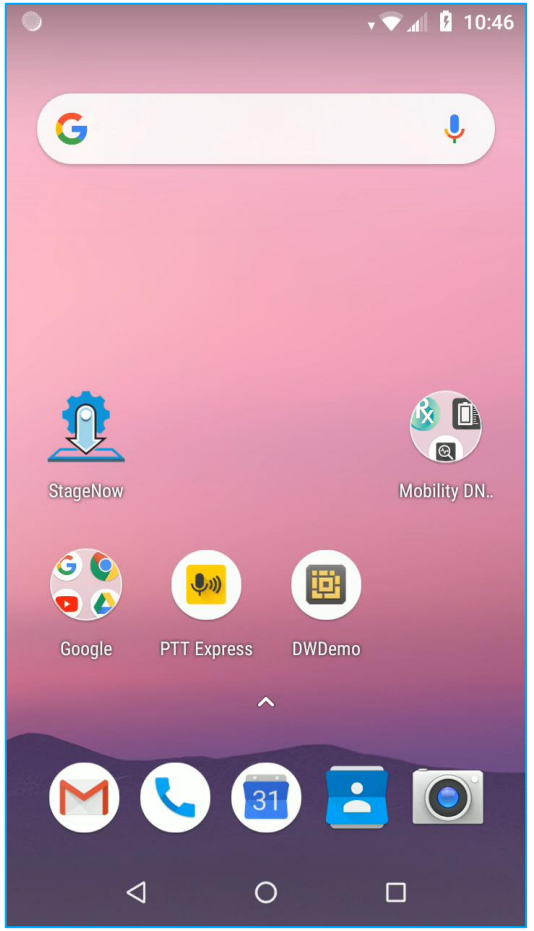
Option 5: Enterprise Browser

- Enterprise Browser is Zebra's recommended tool for developing applications with HTML, JavaScript and CSS
- EB exposes a significant JavaScript API set

JS:

```
EB.Barcode.enable({Props}, fnHandler());
```

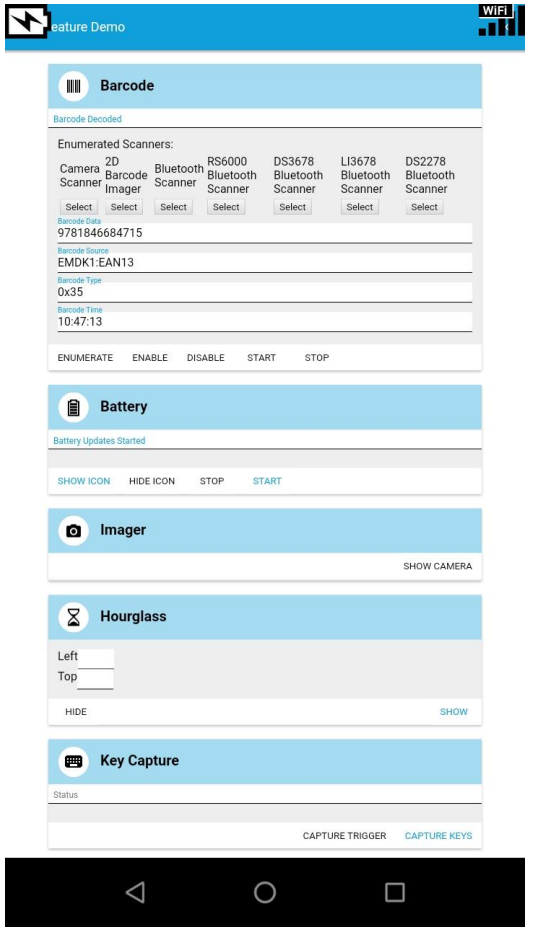
- Requires a license for production but is 100% functional for demo (with nag screen)
- Wraps the device WebView component so your application is rendered within that WebView
  - **If your application runs today in Chrome, it will run in Enterprise Browser but will have access to device hardware via the EB API**
- See other talks on Enterprise Browser



Using JavaScript Frameworks on Zebra Devices

Option 5: Enterprise Browser

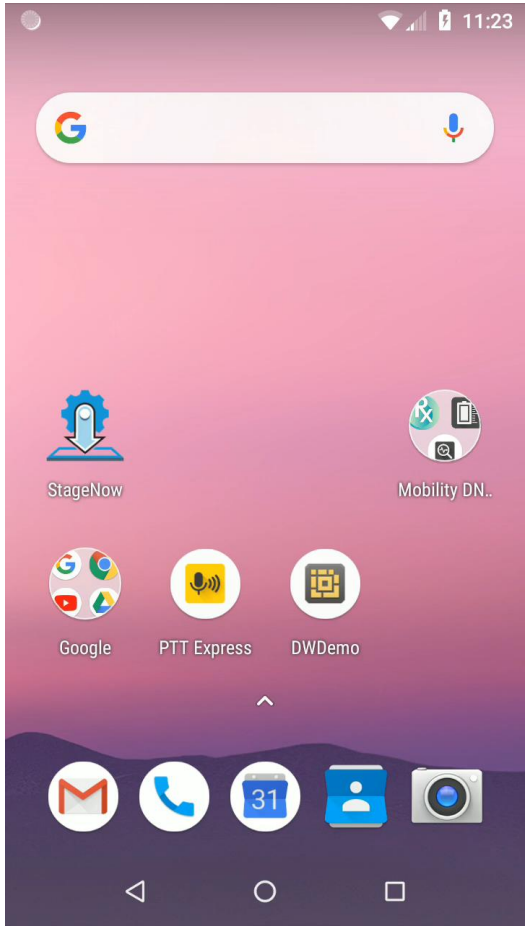
- Getting started:
  - Techdocs has a number of getting started guides.
  - There are a number of Sample Apps available
  - Download from the Zebra download portal.
  - Copy your config.xml file and you are ready to go
- Feature Demo (shown on previous slide and screenshot right):
  - Installed to your machine under C:\<installDir>\Feature-Demo
  - Detailed on techdocs but you can run from an external server:
    - Cd <install dir>\Feature-Demo
    - Start server (python –m SimpleHTTPServer 8081)
    - Push a config.xml with a start page pointing to your server and port.



Using JavaScript Frameworks on Zebra Devices

Option 6: Enterprise Browser with DataWedge Intent output plugin

- Very common question:
  - "Can I listen for DataWedge Intents (or any Intents) in Chrome?"
- The answer is "no" (or, more politely, "no, sorry")
- As detailed in the Chrome developer guide:
  - Chrome allows you to **send** highly configurable intents (and they give the example to invoke zxing through a hyperlink)
  - Chrome will **listen** for Intents sent with ACTION_VIEW with a uri schema
- But Chrome does NOT allow you to listen for arbitrary intents
- Although DataWedge can send an Intent with ACTION_VIEW action, it does not support encoding the data payload in a uri schema.
- You **CAN** however listen for DataWedge Intents with Enterprise Browser's Intent API



Using JavaScript Frameworks on Zebra Devices

Option 6: Enterprise Browser with DataWedge Intent output plugin

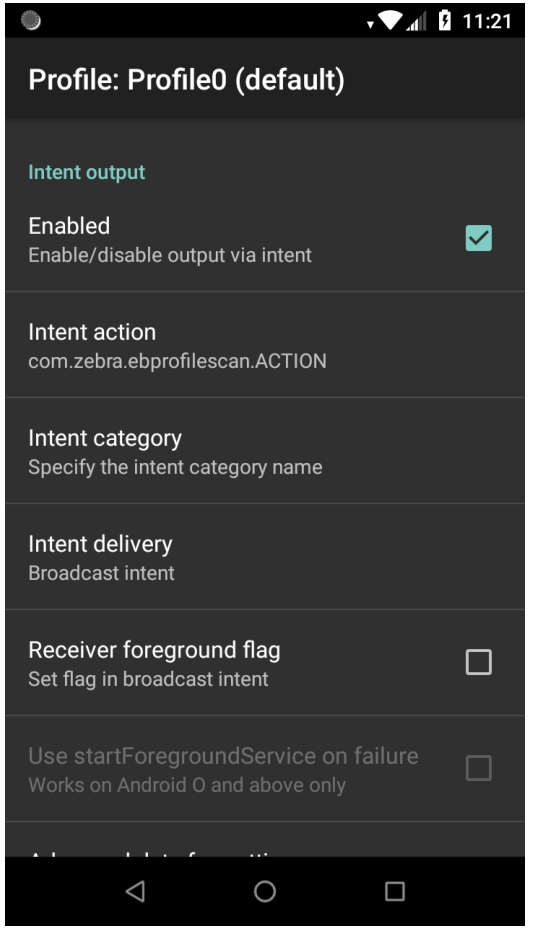
To receive Intents in Enterprise Browser:

1. Configure the DataWedge Profile to output with the Intent plugin and specify a known action
2. Specify the following in your Enterprise Browser config.xml:

```
<usedwforscanning value="1"/>
<IntentReceiver>
   <EnableReceiver value="1"/>
   <IntentAction value="com.zebra.ebprofilescan.ACTION"/>
   <IntentCategory value=""/>
</IntentReceiver>
```

3. Register for the Intent as follows

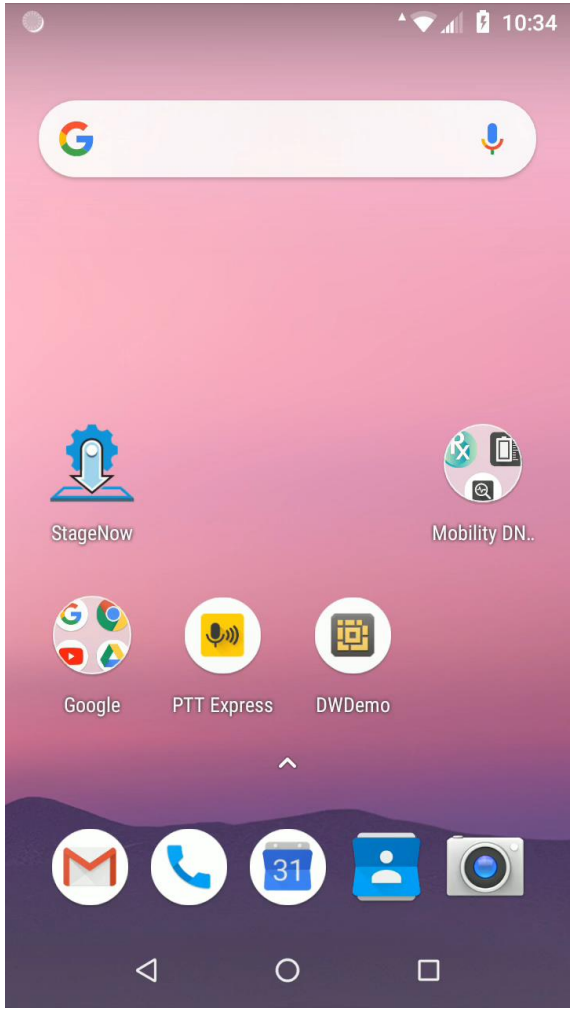
```
EB.Intent.startListening(function(intent) {
    console.log(intent.data;))}
```



Using JavaScript Frameworks on Zebra Devices

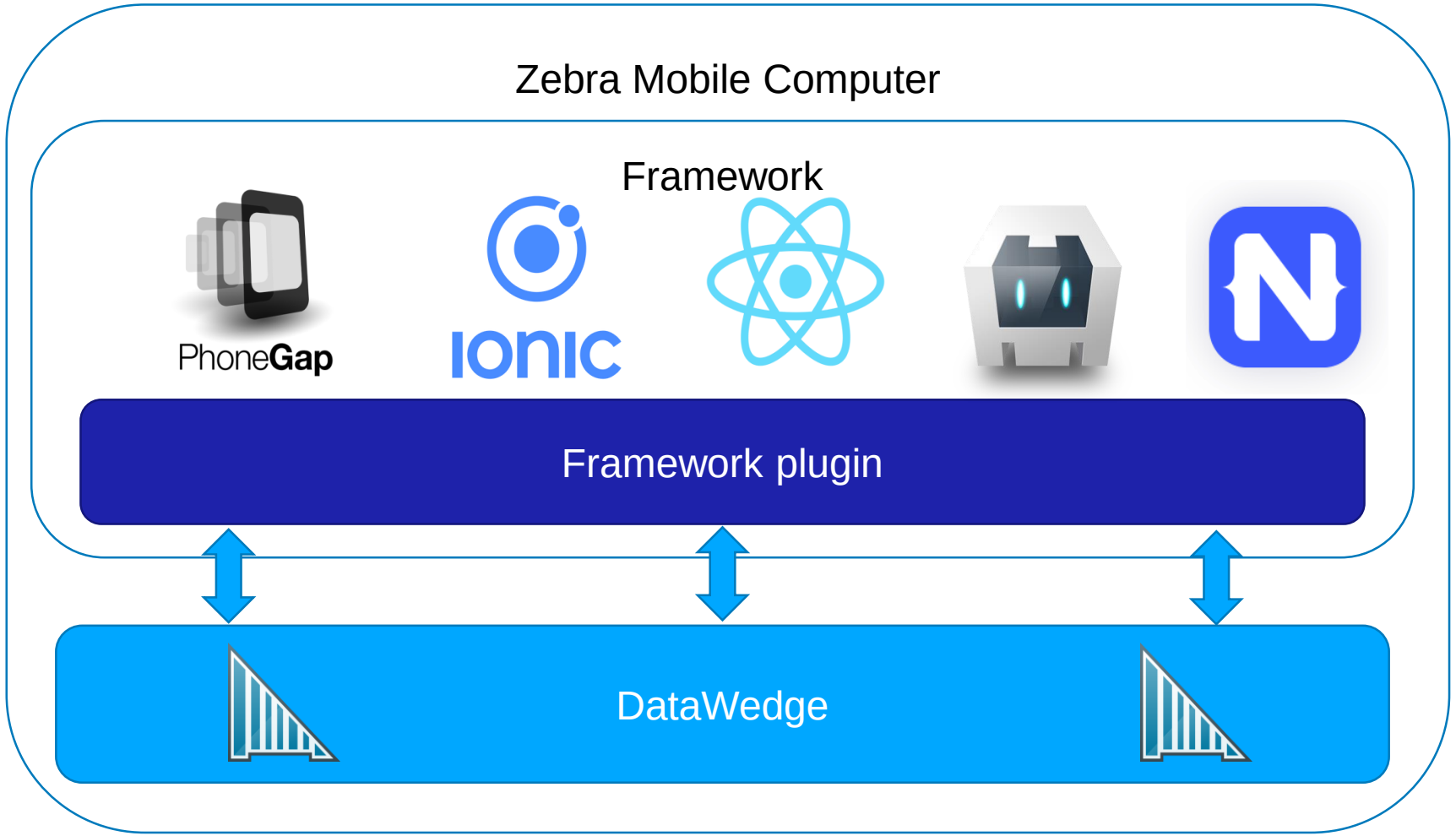
Option 7: Enterprise Browser with a browser-based framework

- JavaScript frameworks that “just work” in a browser can be run within Enterprise Browser to take advantage of the EB API set.
- See related developer articles on [Angular2, React, Vue.js | Framework 7].
- Example: Angular 2 todoMVC: https://github.com/tastejs/todomvc/tree/master/examples/angular2
  - cd angular2
  - npm i (install prerequisites)
  - python -m SimpleHTTPServer 8001 (serve the page from local machine)
  - Update start page:
    - e.g: <StartPage value="http://192.168.0.2:8001" name="Menu"/>
- The related article also covers examples for Vue.js, React & Framework7



Using JavaScript Frameworks on Zebra Devices

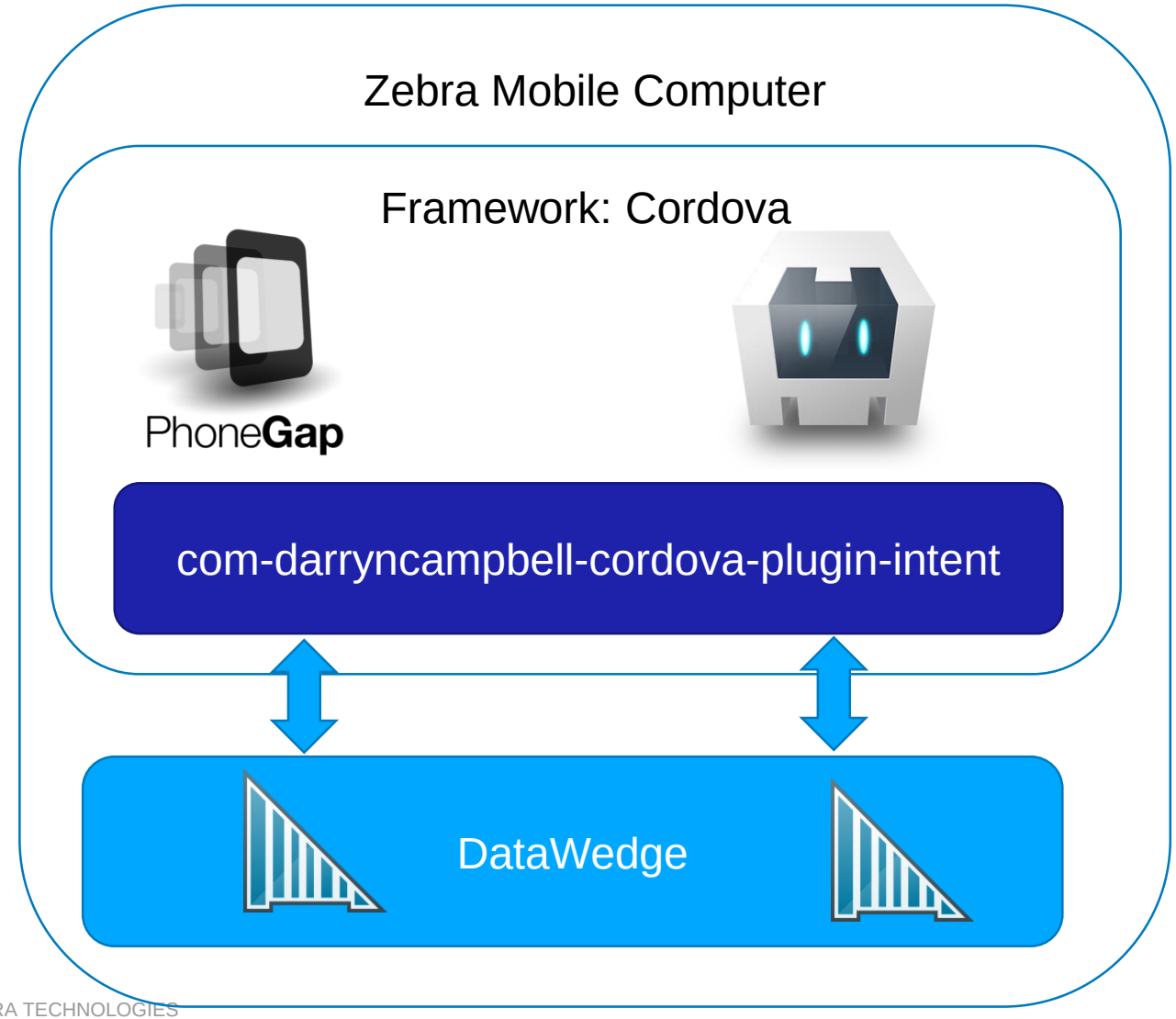
Options 8-10: Hybrid / Native-Hybrid Framework approach

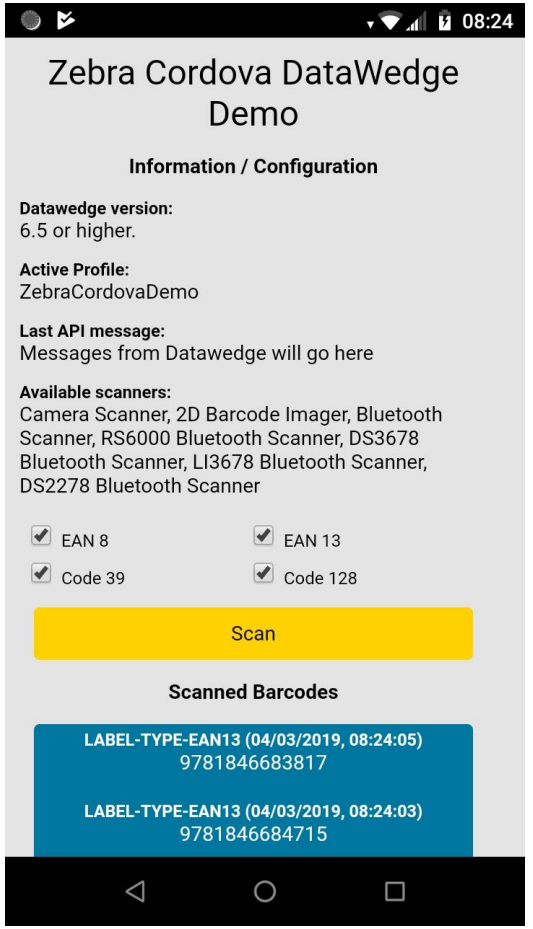


Note: Recommended approach but not formally supported

Using JavaScript Frameworks on Zebra Devices

Option 8: Cordova / PhoneGap

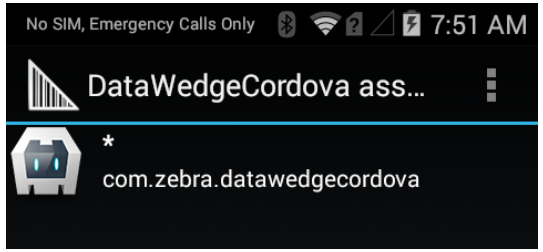




Using JavaScript Frameworks on Zebra Devices

Option 8: Cordova / PhoneGap

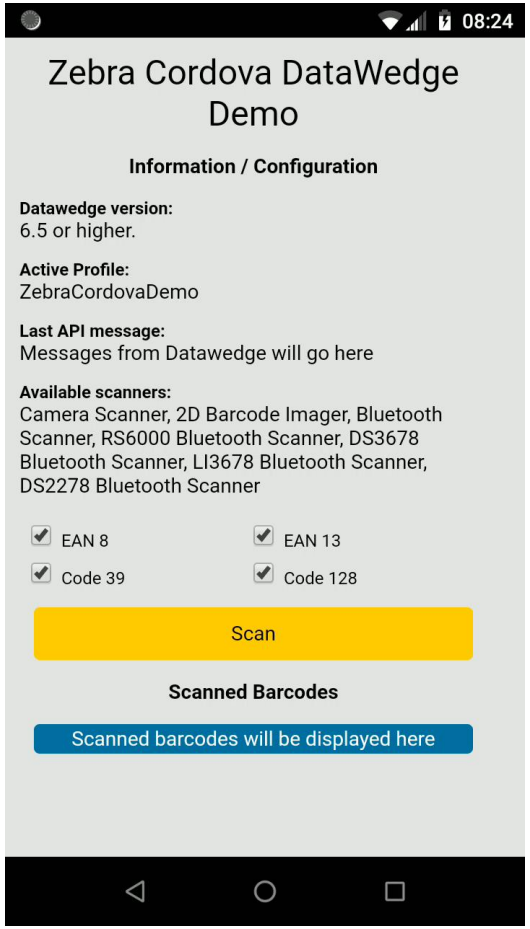
1. Add Cordova plugin to handle Intent communication with DataWedge
   - cordova plugin add com-darryncampbell-cordova-plugin-intent
2. Configure the DataWedge profile



3. Define callback in the application to handle barcode data

```
(<any>window).plugins.intentShim.registerBroadcastReceiver({
    filterActions: ['com.datawedgecordova.ACTION'],
    filterCategories: ['android.intent.category.DEFAULT']},
    function (intent) {console.log('Received Intent: ' + JSON.stringify(intent.extras));});
```

4. Optional: Further control DataWedge & the scanner with the DW API
   - See other presentations on the DataWedge API
5. Sample: https://github.com/darryncampbell/DataWedgeCordova
6. Developer article: https://developer.zebra.com/community/home/blog/2016/08/04/integrating-datawedge-into-your-cordova-application



Using JavaScript Frameworks on Zebra Devices

Option 8: Cordova / PhoneGap (JavaScript)

4. Optional: Further control DataWedge & the scanner with the DW API

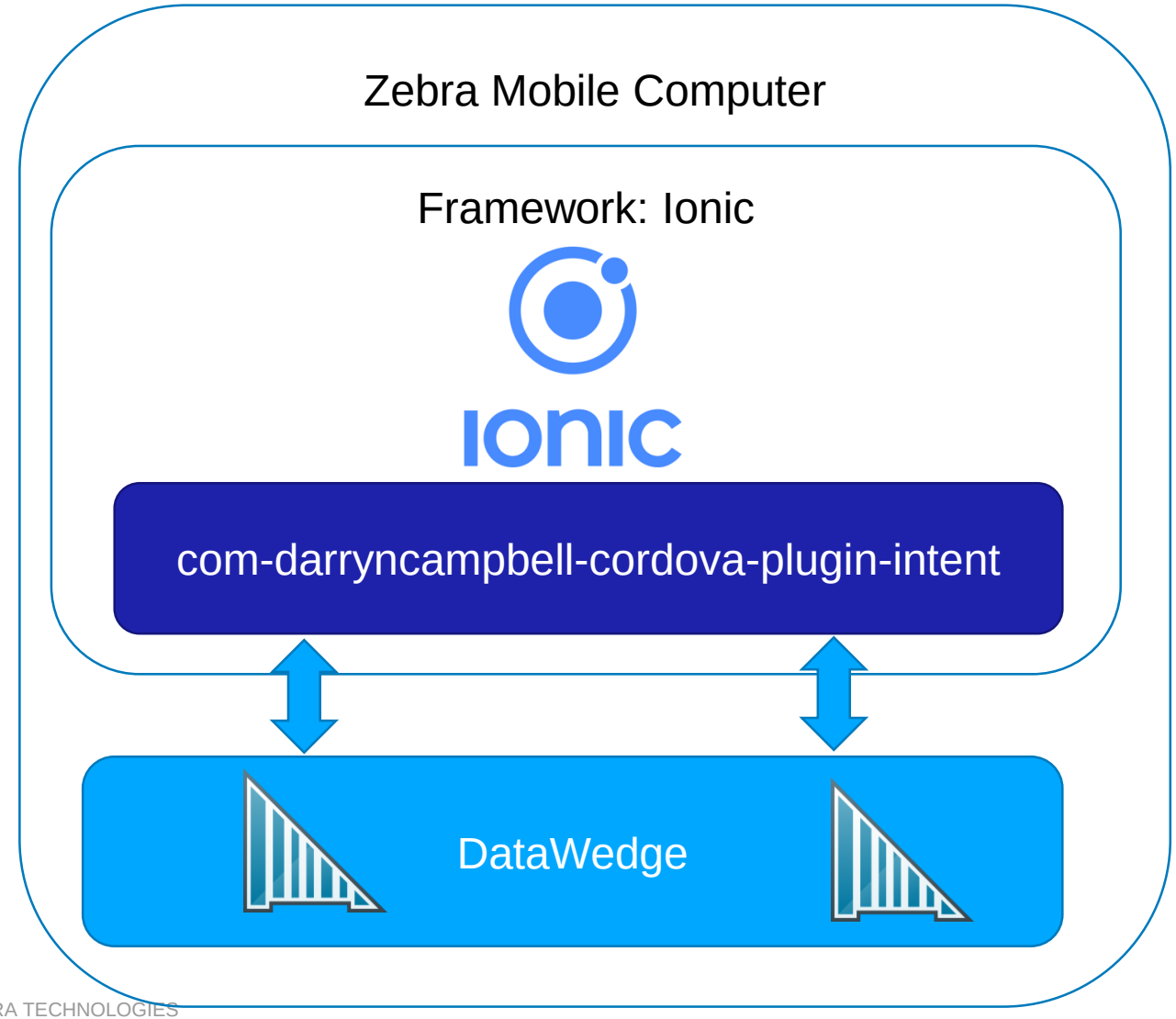
Example: Soft starting the scanner:

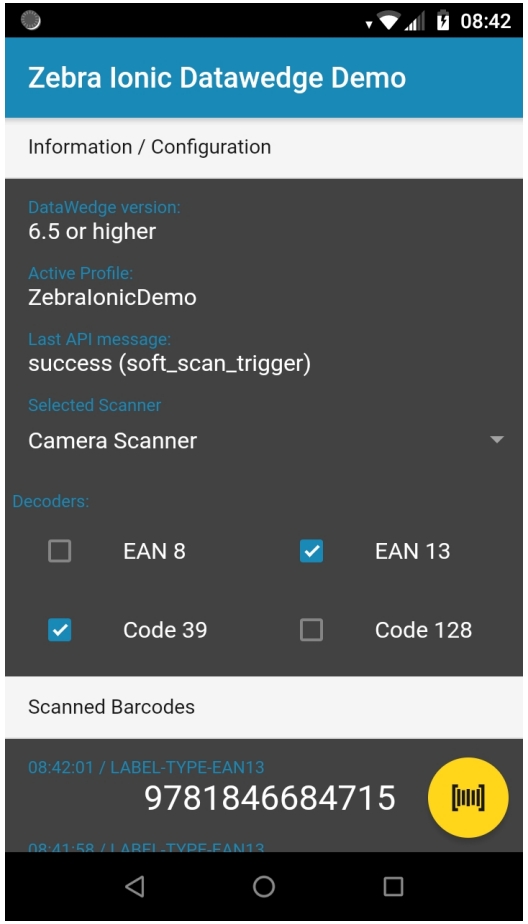
```
function sendCommand(extraName, extraValue) {
  var broadcastExtras = {};
  broadcastExtras[extraName] = extraValue;
  broadcastExtras["SEND_RESULT"] = sendCommandResults;
  window.plugins.intentShim.sendBroadcast({
    action: "com.symbol.datawedge.api.ACTION",
    extras: broadcastExtras
  },
function () { }, // Success
function () { });} // Failure

sendCommand("com.symbol.datawedge.api.SOFT_SCAN_TRIGGER", 'START_SCANNING');
```

Using JavaScript Frameworks on Zebra Devices

Option 9: Ionic

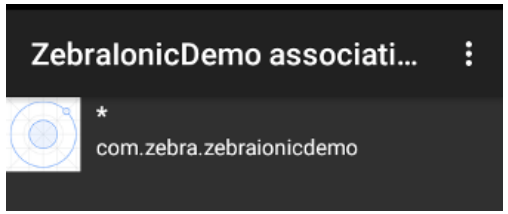




Using JavaScript Frameworks on Zebra Devices

Option 9: Ionic

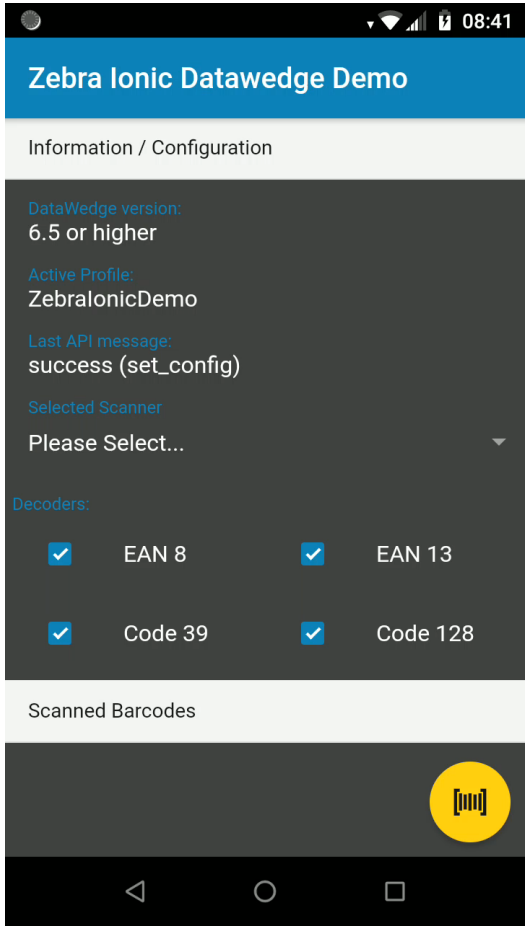
1. Add Cordova plugin to handle Intent communication with DataWedge
   - Ionic cordova plugin add com-darryncampbell-cordova-plugin-intent
2. Configure the DataWedge profile



3. Define callback in the application to handle barcode data

```
(<any>window).plugins.intentShim.registerBroadcastReceiver({
    filterActions: ['io.ionic.starter.ACTION'],
    filterCategories: ['android.intent.category.DEFAULT']},
    function (intent) {console.log('Received Intent: ' + JSON.stringify(intent.extras));});
```

4. Optional: Further control DataWedge & the scanner with the DW API
   - See other presentations on the DataWedge API
5. Sample: https://github.com/Zebra/ZebraIonicDemo
6. Developer article: https://developer.zebra.com/community/home/blog/2018/04/03/ionic-applications-on-zebra-devices



Using JavaScript Frameworks on Zebra Devices

Option 9: Ionic (TypeScript)

4. Optional: Further control DataWedge & the scanner with the DW API

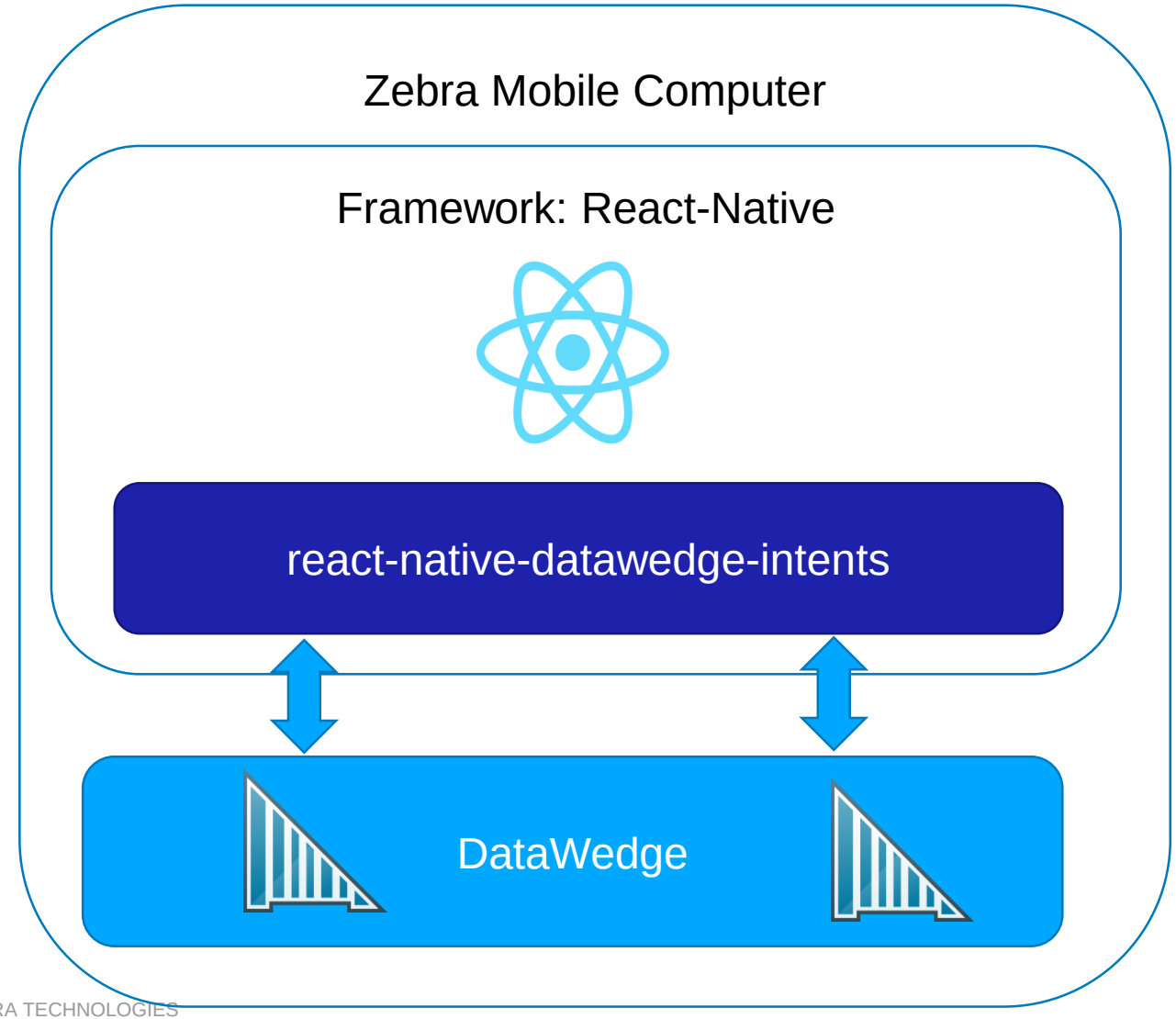
Example: Soft starting the scanner:

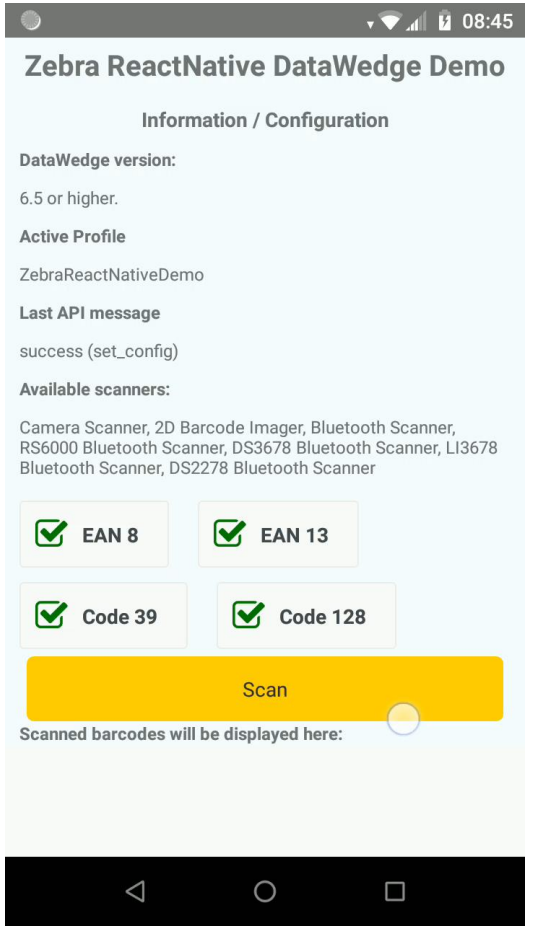
```
sendCommand(extraName: string, extraValue) {
  (<any>window).plugins.intentShim.sendBroadcast({
  action: 'com.symbol.datawedge.api.ACTION’,
  extras: {
    [extraName]: extraValue,
    "SEND_RESULT": "true"
  }},
function () { }, // Success.
function () { } // Failure.);}

sendCommand("com.symbol.datawedge.api.SOFT_SCAN_TRIGGER", "START_SCANNING");
```

Using JavaScript Frameworks on Zebra Devices

Option 10: React Native

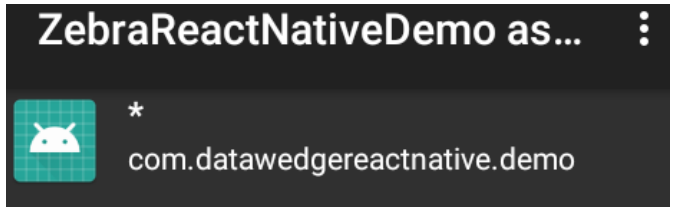




Using JavaScript Frameworks on Zebra Devices

Option 10: React Native

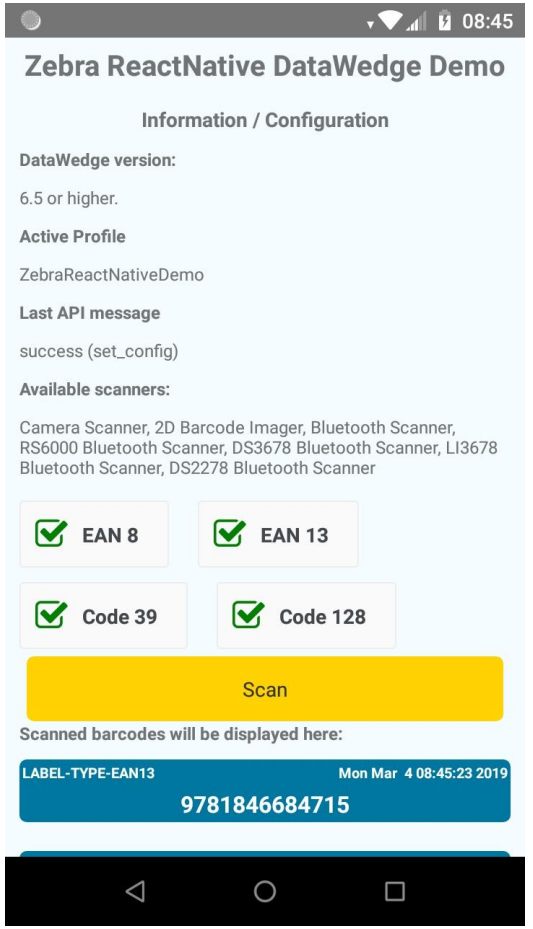
1. Add React-Native module to handle Intent communication with DataWedge
   - Package.json: “react-native-datawedge-intents”: “^0.1.1”
2. Configure the DataWedge profile



3. Define callback in the application to handle barcode data

```
DataWedgeIntents.registerBroadcastReceiver ({
     filterActions: [‘com.zebra.reactnativedemo.ACTION'],
     filterCategories: ['android.intent.category.DEFAULT’]});
     broadcastReceiver (intent) {console.log('Received Intent: ' + JSON.stringify(intent));});
```

4. Optional: Further control DataWedge & the scanner with the DW API
   - See other presentations on the DataWedge API
5. Sample: https://github.com/darryncampbell/DataWedgeReactNative
6. Developer article: https://developer.zebra.com/community/home/blog/2018/10/29/developing-react-native-applications-on-zebra-devices



Using JavaScript Frameworks on Zebra Devices

Option 10: React Native (JavaScript)

4. Optional: Further control DataWedge & the scanner with the DW API

Example: Soft starting the scanner

```
sendCommand(extraName, extraValue) {
  var broadcastExtras = {};
  broadcastExtras[extraName] = extraValue;
  broadcastExtras["SEND_RESULT"] = this.sendCommandResult;
  DataWedgeIntents.sendBroadcastWithExtras({
    action: "com.symbol.datawedge.api.ACTION",
    extras: broadcastExtras});
}

sendCommand("com.symbol.datawedge.api.SOFT_SCAN_TRIGGER", 'TOGGLE_SCANNING');
```

Using JavaScript Frameworks on Zebra Devices

Coming Soon

Using JavaScript Frameworks on Zebra Devices

Summary

- MANY options for JavaScript developers
- Zebra endeavours to enable JavaScript developers to create applications for our devices
  - Given the plethora or JavaScript frameworks available it is not possible to offer specific support for particular frameworks
- JS Frameworks are designed to run on Android therefore apps created with these frameworks will work on Zebra devices
  - The challenge is to interact with the device hardware
- Over the years, Zebra have published numerous recommendations on how best to work with the different frameworks

Using JavaScript Frameworks on Zebra Devices

Resources

- Resources used in this presentation:
  - Options 1 – 5 (DataWedge Keystroke options & EB options) (GitHub repo)
    - https://github.com/darryncampbell/Appforum_2019_Javascript_Frameworks
  - Option 6: Listening for KeyPress events with DataWedge (Developer post | Sample app)
    - https://developer.zebra.com/blog/listening-keypress-events-datawedge
    - https://github.com/darryncampbell/DataWedge-KeyEvent-Options
  - Option 7 (Blog posts on Angular2, React, Vue.js and Framework 7)
    - https://developer.zebra.com/community/home/blog/2019/03/05/javascript-frameworks-with-enterprise-browser
    - https://developer.zebra.com/community/home/blog/2018/05/21/using-framework7-with-enterprise-browser
  - Option 8: Integrating DataWedge into your Cordova application (Developer post | Sample app)
    - https://developer.zebra.com/community/home/blog/2016/08/04/integrating-datawedge-into-your-cordova-application
    - https://github.com/darryncampbell/DataWedgeCordova
  - Option 9: Ionic Applications on Zebra Devices (Developer post | Sample app) | DevTALK)
    - https://developer.zebra.com/community/home/blog/2018/04/03/ionic-applications-on-zebra-devices
    - https://github.com/Zebra/ZebraIonicDemo
    - https://www.youtube.com/watch?v=zuHkqGocOqE
  - Option 10: Developing React Native Applications on Zebra Devices (Developer post | Sample app)
    - https://developer.zebra.com/community/home/blog/2018/10/29/developing-react-native-applications-on-zebra-devices
    - https://github.com/darryncampbell/DataWedgeReactNative

Questions?

ZEBRA DEVELOPER PORTAL
http://developer.zebra.com

Sign up for news
Join the ISV program

Thank You

BACKUP