AsReaderDock React Native Wrapper Documentation

Table of Contents

Table of Contents 2
Release Versions........ 3
1.0 3
1. Setup 4
For AsReaderDock........ 4
2. Usage 5
3. RNReaderModule Events........ 6
4. RNReaderModule Methods 6
5. Errors........ 8
6. Example........ 9

Release Versions

1.0

Date of Release – 9/9/2022

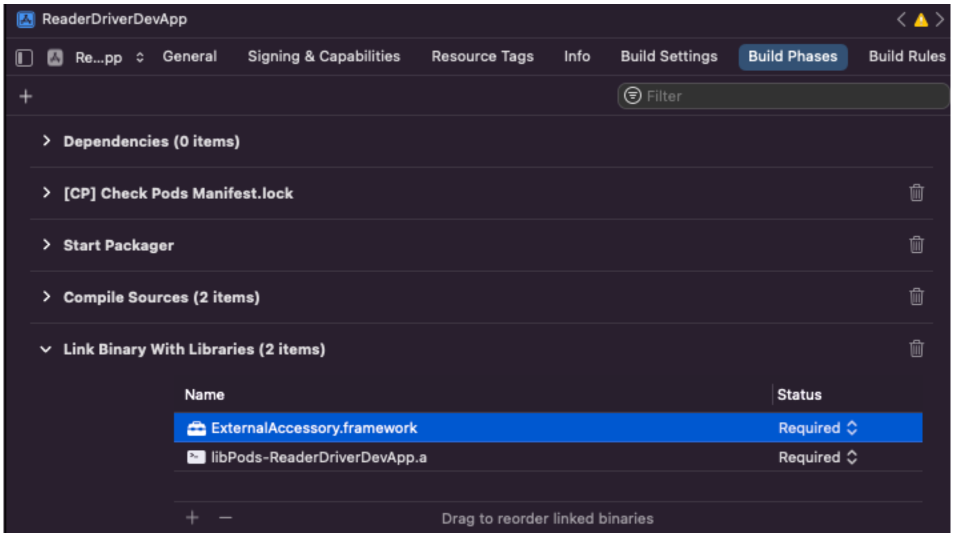
1. Initial release of the document.

1. Setup

1. In the React Native project, run “npm install @xemelgo/ios-edge-driver@1.1.7” or add "@xemelgo/ios-edge-driver": "1.1.7" in package.json
2. In the ios directory of the project, run “pod install”

For AsReaderDock

3. Open the project’s, .xcworkspace file in Xcode and go to the “Build Phases” page in Target settings. In “Link Binary with Libraries” add “ExternalAccessory.framework” (highlighted below).



> Dependencies (0 items)

> [CP] Check Pods Manifest.lock

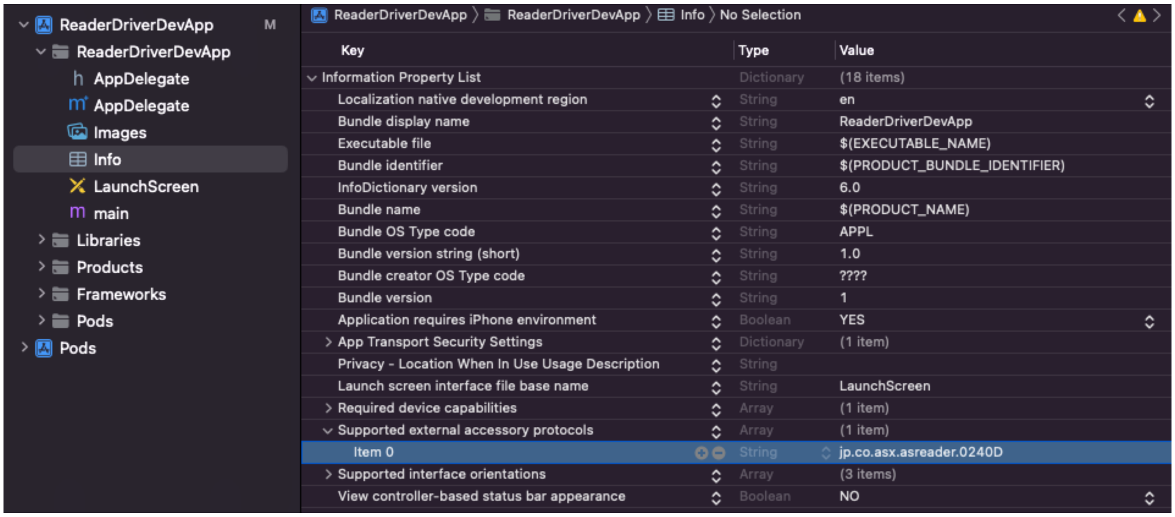
> Start Packager

> Compile Sources (2 items)

Link Binary With Libraries (2 items)

Name	Status
ExternalAccessory.framework	Required
libPods-ReaderDriverDevApp.a	Required

4. In Xcode, open the project’s info.plist file. For key “Supported external accessory protocols”, add “jp.co.asx.asreader.0240D” as an item (highlighted below):



Key	Type	Value
Information Property List	Dictionary	(18 items)
Localization native development region	String	en
Bundle display name	String	ReaderDriverDevApp
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle version string (short)	String	1.0
Bundle creator OS Type code	String	????
Bundle version	String	1
Application requires iPhone environment	Boolean	YES
App Transport Security Settings	Dictionary	(1 item)
Privacy - Location When In Use Usage Description	String	
Launch screen interface file base name	String	LaunchScreen
Required device capabilities	Array	(1 item)
Supported external accessory protocols	Array	(1 item)
Item 0	String	jp.co.asx.asreader.0240D
Supported interface orientations	Array	(3 items)
View controller-based status bar appearance	Boolean	NO

2. Usage

Accessing the Native Module:

```
import { NativeModules } from "react-native";
const { RNReaderModule } = NativeModules;

// Call methods on RNReaderModule, ie “RNReaderModule.getDeviceDetails();”
```

Creating event emitter:

```
import { NativeEventEmitter, NativeModules } from "react-native";
const { RNReaderModule } = NativeModules;

const events = new NativeEventEmitter(RNReaderModule);
```

Adding event listener example:

```
events.addListener("barcodeRead", (barcode) => {
  // process barcode
});
```

3. RNReaderModule Events

- **barcodeScan**
  - Sends event when a barcode is scanned
  - Body: string corresponding to the barcode that is scanned
- **rfidRead:**
  - Sends event when an RFID tag is read
  - Body: {
    "epc": the tag’s EPC,
    "rssi": the tag’s RSSI,
    "tagCount": the number of times this tag was read,
    "tid": the tag’s TID (currently always returns empty string for AsReaderDock)
    }

4. RNReaderModule Methods

- **getAvailable() => Promise<string[]>**
  - Returns a list of the readers that are plugged in or connected via Bluetooth and are available to be connected to
- **loadDriver(deviceName: string) => Promise<void>**
  - Loads the selected driver
  - deviceName: the identity of the driver to load. The driver identity must belong to one of the supported devices
  - throws LoadDriverError if unable to load driver because the device name was invalid
- **getDriverName() => Promise<string>**
  - Returns the name of the currently loaded driver
- **getDriverVersion() => Promise<{string: string}>**
  - Returns information about the version hierarchy of the driver
- **getDeviceDetails() => Promise<object>**
  - Returns JSON object with details of the device, like firmware, serial number, etc
- **connect(address: string) => Promise<void>**
  - Connects to the reader
  - address: the Bluetooth address of the reader if it is needed to connect. For non-Bluetooth readers, any string works
  - Throws ReaderNotFoundError
- **disconnect() => Promise<void>**
  - Disconnects from the reader
  - Throws ReaderNotFoundError
- **isConnected() => Promise<boolean>**
  - Returns the connection status of the reader
- **getBatteryPercentage() => Promise<number>**
  - Returns the battery percentage of the reader

  - Throws ReaderNotFoundError and ReaderConnectionError
- **switchToScanRfidMode() => Promise<void>**
  - Switches to RFID reading mode and starts reading tags if trigger pressed
  - Throws ReaderNotFoundError and ReaderConnectionError
- **startRfidScan() => Promise<void>**
  - Starts continuous RFID reading meaning that tags are read continuously without trigger needing to be pressed
  - Throws ReaderNotFoundError and ReaderConnectionError
- **stopRfidScan() => Promise<void>**
  - Stops continuous RFID reading
  - Throws ReaderNotFoundError and ReaderConnectionError
- **activateSearchMode(tagToFind: string) => Promise<void>**
  - Enables search mode with given tag EPC, so that only a tag with this EPC is sent
  - tagToFind: EPC of the tag to search for
  - Throws ReaderNotFoundError and ReaderConnectionError
- **stopSearchMode() => Promise<void>**
  - Stops search mode
  - Throws ReaderNotFoundError and ReaderConnectionError
- **switchToScanBarcodeMode() => Promise<void>**
  - Switches to barcode scanning mode
  - Throws ReaderNotFoundError and ReaderConnectionError

5. Errors

ReaderNotFoundError – This error occurs when the reader is not available, either because it is not plugged in or not connected via Bluetooth.
ReaderConnectionError – This error occurs if an operation which requires the reader to be connected (eg getDeviceDetails) is called before a successful call to connect.
NoDriverLoadedError – This error occurs if a method is called when no driver has been successfully loaded.
NoDriverLoadedError – This error there is an unsupported device name in call to loadDriver.

6. Example

The following example would connect to AsReaderDock, turn on RFID reading, and print any scanned tags

```
import { NativeEventEmitter, NativeModules } from "react-native";
const { RNReaderModule } = NativeModules;

const events = new NativeEventEmitter(RNReaderModule);
events.addListener("rfidRead", (tagData) => {
  console.log(tagData.epc, tagData.rssi, tagData.tagCount);
});

await RNReaderModule.loadDriver("AsReaderDock");
await RNReaderModule.connect("localhost");
await RNReaderModule.switchToScanRfidMode();
```