Advanced Card Systems Ltd.
Card & Reader Technologies

ACS Bluetooth Library Update Guide [iOS and Android]

User Guide V1.01

Subject to change without prior notice

info@acs.com.hk
www.acs.com.hk

Table of Contents

1.0. Introduction

This is the user guide for the procedure to update the ACS Bluetooth Library on the application created by client. Below are the general guidelines for updating the library. Please note that there may be incompatibility issues when updating the library, so additional modifications to the application may be required.

1.1. ACS Smart Card I/O Android Library

The library "acssmcio" provides classes and interfaces for communicating with ACS Bluetooth readers. It is based on the service provider interface (TerminalFactorySpi) from Java Smart Card I/O API defined by JSR 268 [1].

The Java Smart Card I/O API defines a Java API for communication with Smart Cards using ISO/IEC 7816-4 APDUs. It thereby allows Java applications to interact with applications running on the Smart Card, to store and retrieve data on the card, etc.

The library "smartcardio" imports Java packages from OpenJDK:

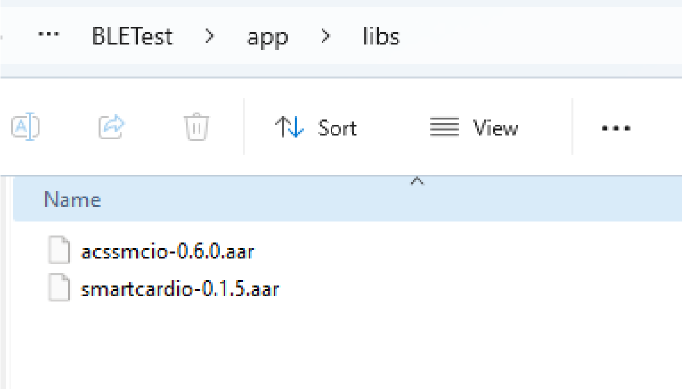
- javax.smartcardio
- sun.net.www
- sun.nio.cs
- sun.security.action
- sun.security.jca
- sun.security.util

[1] https://jcp.org/en/jsr/detail?id=268

1.1.1. Android library installation

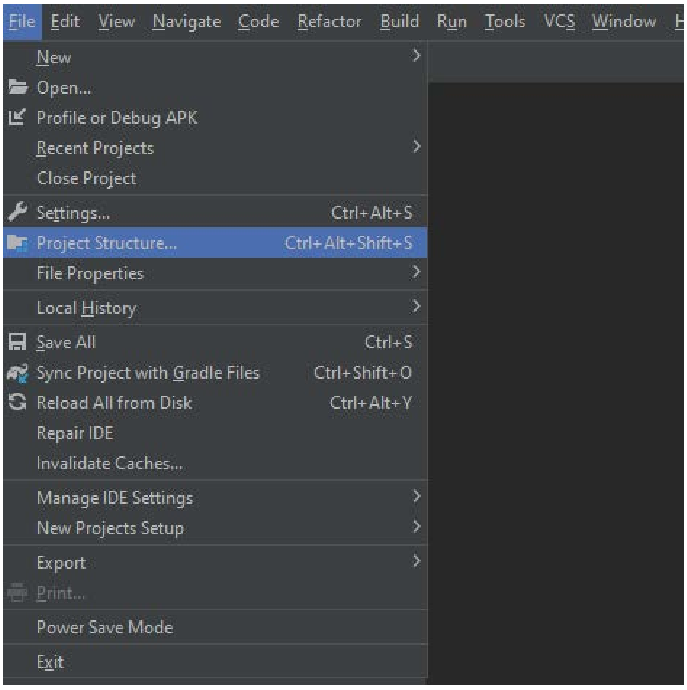
1) **Download the updated Android library and extract it.**

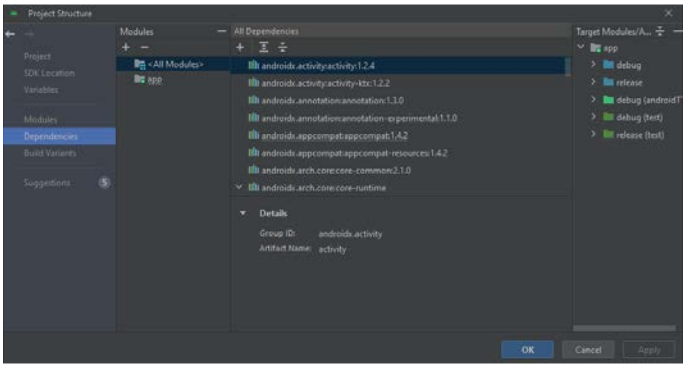
Locate the class library folder "C:\BLETest\app\libs\" you extracted, you can see the "acssmcio-x.y.z.aar" and "smartcardio-x.y.z.aar". Copy this two class library files to your project folder "app\libs"



acssmcio-0.6.0.aar
smartcardio-0.1.5.aar

2) **Go to File -> Project Structure -> Dependencies.**

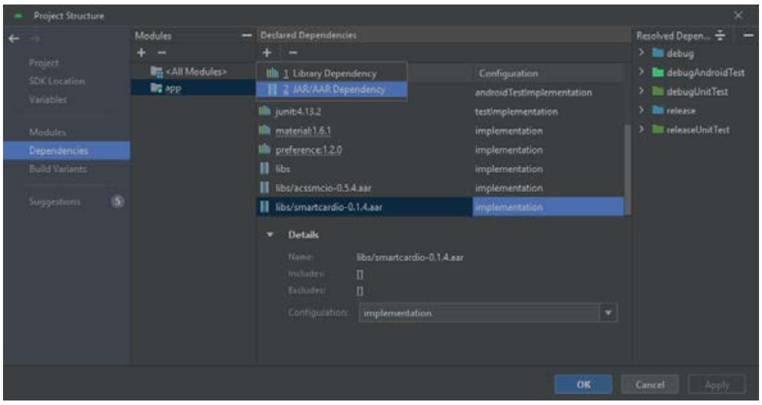




androidx.activity:activity:1.2.4
androidx.activity:activity-ktx:1.2.2
androidx.annotation:annotation:1.3.0
androidx.annotation:annotation-experimental:1.1.0
androidx.appcompat:appcompat:1.4.2
androidx.appcompat:appcompat-resources:1.4.2
androidx.arch.core:core-common:2.1.0
androidx.arch.core:core-runtime

Details
Group ID: androidx.activity
Artifact Name: activity

3) In the "Declared Dependencies" tab, click and select "Jar Dependency" in the dropdown.



	Configuration
	androidTestImplementation
junit:4.13.2	testImplementation
material:1.6.1	implementation
preference:1.2.0	implementation
libs	implementation
libs/acssmcio-0.5.4.aar	implementation
libs/smartcardio-0.1.4.aar	implementation

Details

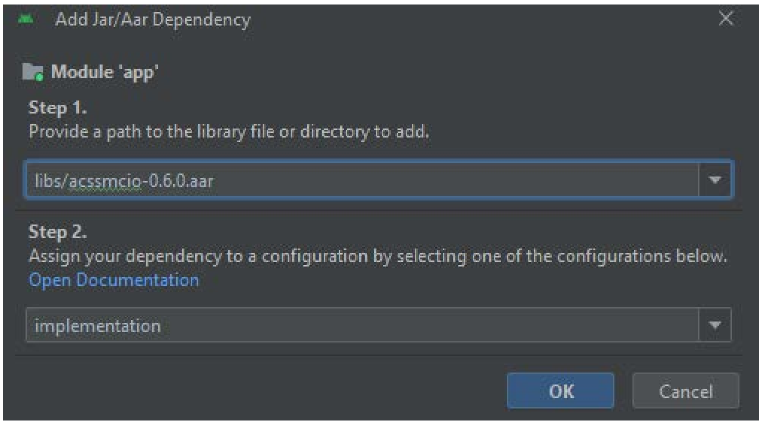
Name: libs/smartcardio-0.1.4.aar

Includes: []

Excludes: []

Configuration: implementation

4) In the "Add Jar/Aar Dependency" dialog, enter the path to "libs/acssmcio-x.y.z.aar" and select "implementation" configuration.



Module 'app'

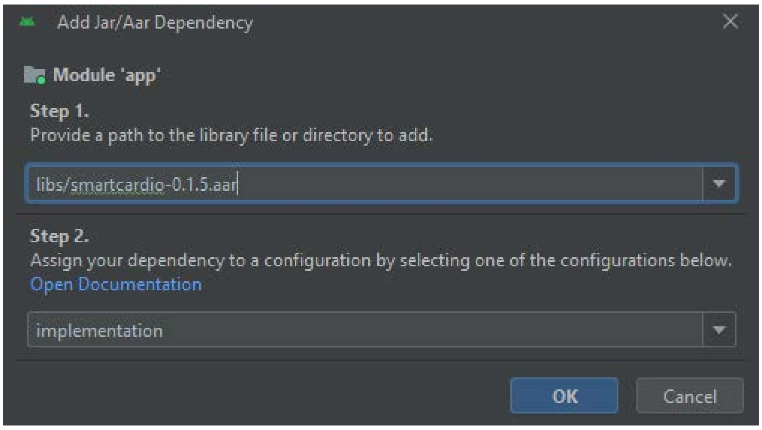
Step 1.
Provide a path to the library file or directory to add.

libs/acssmcio-0.6.0.aar

Step 2.
Assign your dependency to a configuration by selecting one of the configurations below.
Open Documentation

implementation

5) Follow the above steps to add "libs/smartcardio-x.y.z.aar".



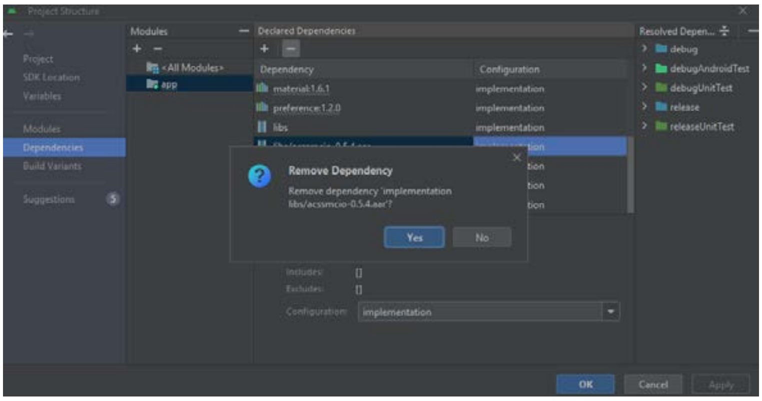
Module 'app'

Step 1.
Provide a path to the library file or directory to add.

libs/smartcardio-0.1.5.aar

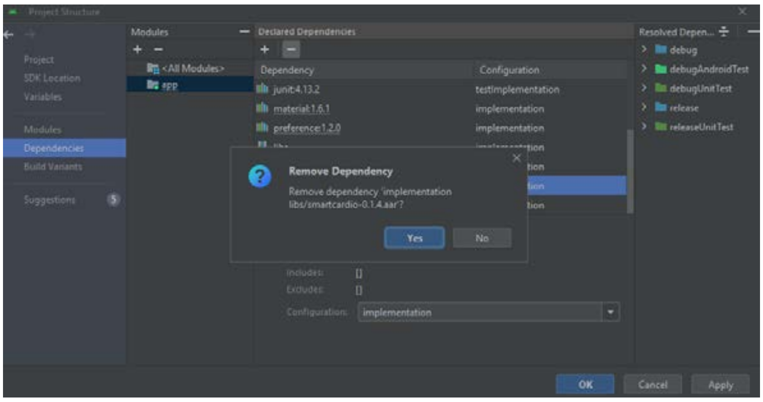
Step 2.
Assign your dependency to a configuration by selecting one of the configurations below.
Open Documentation

implementation



Remove Dependency

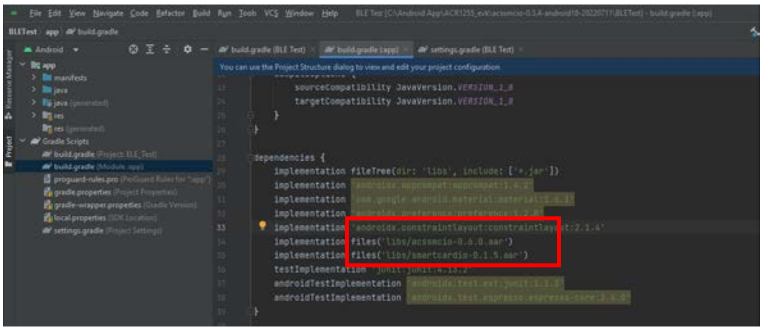
Remove dependency 'implementation libs/acssmcio-0.5.4.aar'?



Remove Dependency

Remove dependency 'implementation libs/smartcardio-0.1.4.aar'?

6) **You will see the following lines from your app's "build.gradle" file.**



You can use the Project Structure dialog to view and edit your project configuration

```
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'androidx.appcompat:appcompat:1.4.2'
    implementation 'com.google.android.material:material:1.6.1'
    implementation 'androidx.preference:preference:1.2.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    implementation files('libs/acssmcio-0.6.0.aar')
    implementation files('libs/smartcardio-0.1.5.aar')
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
}
```

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
BLETest app build.gradle
Android
build.gradle (BLE Test)
build.gradle (:app)
settings.gradle (BLE Test)
app
manifests
java
java (generated)
res
res (generated)
Gradle Scripts
build.gradle (Project: BLE_Test)
build.gradle (Module :app)
proguard-rules.pro (ProGuard Rules for "app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
local.properties (SDK Location)
settings.gradle (Project Settings)

implementation files('libs/acssmcio-x.y.z.aar')

implementation files('libs/smartcardio-x.y.z.aar')

7) **Instructions for Adding ACR1555U Terminal Type**

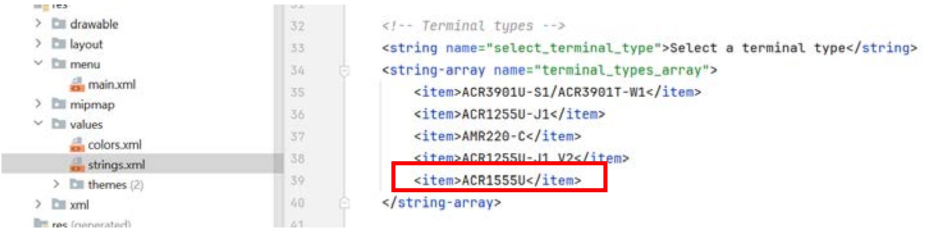
a. **Locate the strings.xml File:**

Open your project and navigate to the res/values/ directory to find the strings.xml file.

b. **Add Terminal Type:**

In the strings.xml file, insert the following line within the <string-array> tag to include the ACR1555U terminal type:

<item>ACR1555U</item>



```
<!-- Terminal types -->
<string name="select_terminal_type">Select a terminal type</string>
<string-array name="terminal_types_array">
    <item>ACR3901U-S1/ACR3901T-W1</item>
    <item>ACR1255U-J1</item>
    <item>AMR220-C</item>
    <item>ACR1255U-J1 V2</item>
    <item>ACR1555U</item>
</string-array>
```

a) **Build the Project and Testing:**

After adding the terminal type, proceed to build your project. This should enable the functionality to search for the ACR1555U terminal.

Please note that there may be incompatibility issues when updating the library, so additional modifications to the application may be required.

1.2. ACS Smart Card I/O iOS Framework

The framework "ACSSmartCardIO" provides classes and interfaces for communicating with ACS Bluetooth readers. It is based on the service provider interface (TerminalFactorySpi) from Java Smart Card I/O API defined by JSR 268 [1].

The Java Smart Card I/O API defines a Java API for communication with Smart Cards using ISO/IEC 7816-4 APDUs. It thereby allows Java applications to interact with applications running on the Smart Card, to store and retrieve data on the card, etc.

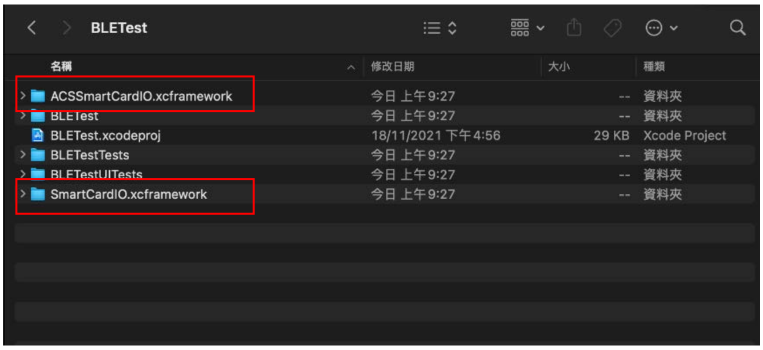
The framework "SmartCardIO" contains the following source code from OpenJDK which is ported to Swift:

- javax.smartcardio
- java.security.Provider

[1] https://jcp.org/en/jsr/detail?id=268

1.2.1. iOS library installation

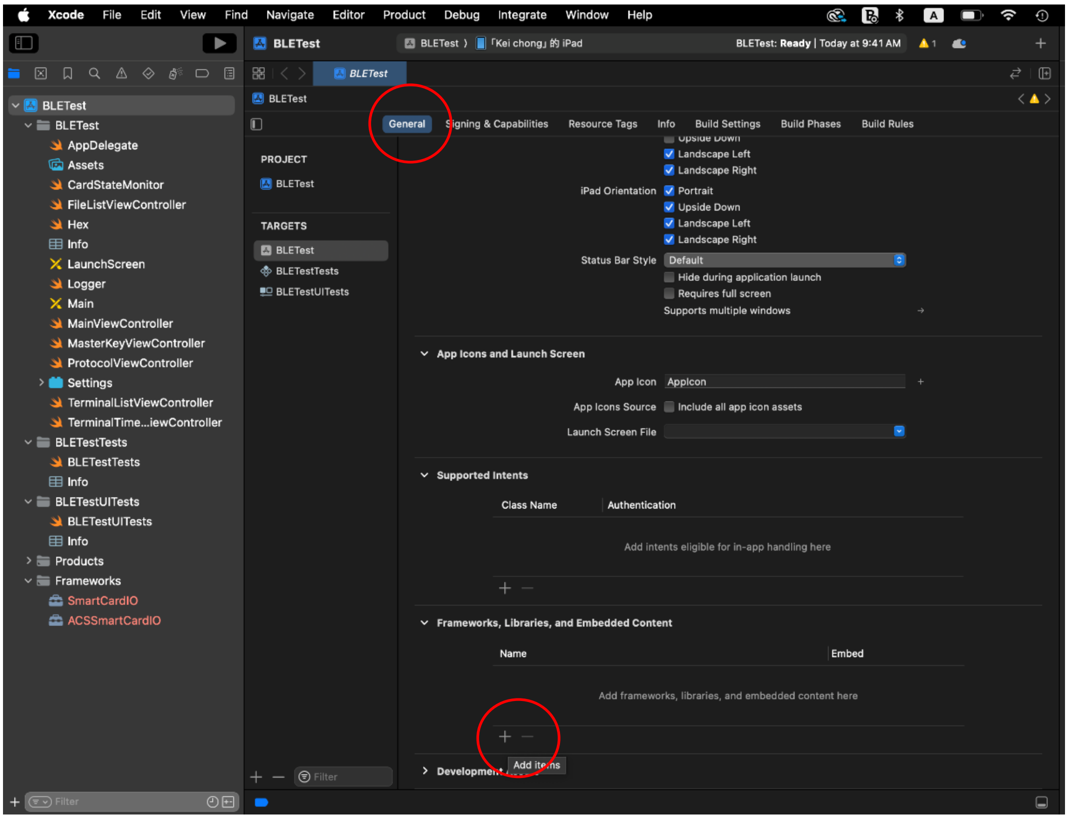
1. To use the framework to your project, copy the folder "BLETest\SmartCardIO.xcframework" and "BLETest\ACSSmartCardIO.xcframework" to your project folder.



BLETest

名稱	修改日期	大小	種類
ACSSmartCardIO.xcframework	今日 上午9:27	--	資料夾
BLETest	今日 上午9:27	--	資料夾
BLETest.xcodeproj	18/11/2021 下午4:56	29 KB	Xcode Project
BLETestTests	今日 上午9:27	--	資料夾
BLETestUITests	今日 上午9:27	--	資料夾
SmartCardIO.xcframework	今日 上午9:27	--	資料夾

2. Select General tab in the Targets and click "+" in Embedded Binaries.



☐ Upside Down
☑ Landscape Left
☑ Landscape Right

iPad Orientation ☑ Portrait
☑ Upside Down
☑ Landscape Left
☑ Landscape Right

Status Bar Style Default
☐ Hide during application launch
☐ Requires full screen
Supports multiple windows

App Icons and Launch Screen

App Icon AppIcon
App Icons Source ☐ Include all app icon assets
Launch Screen File

Supported Intents

Class Name	Authentication

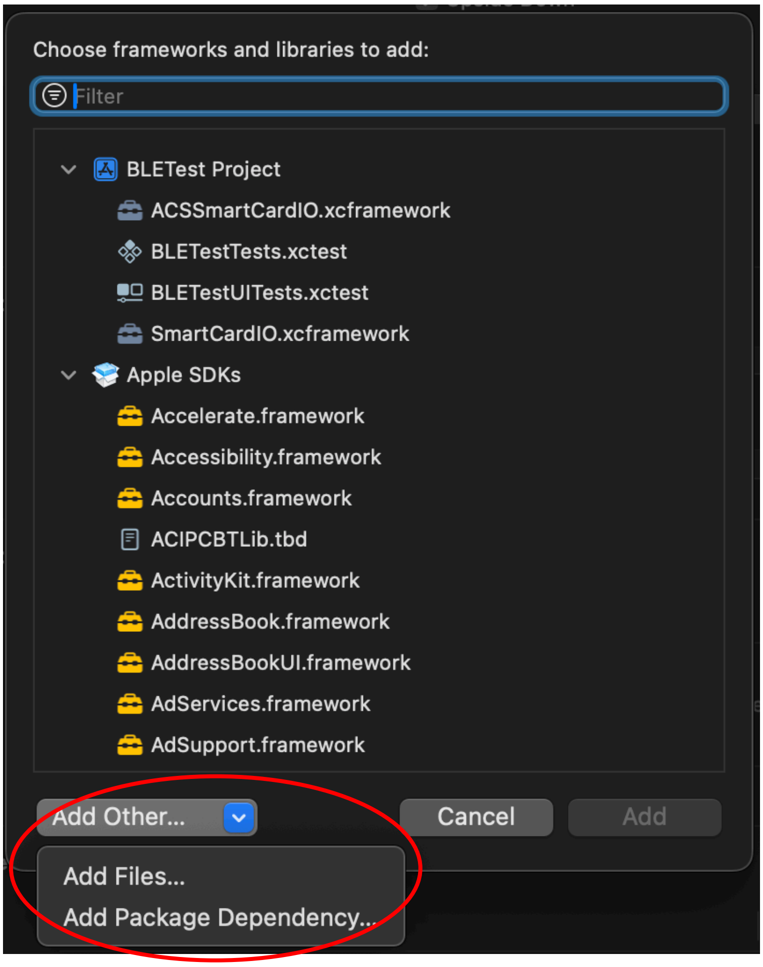
Add intents eligible for in-app handling here

Frameworks, Libraries, and Embedded Content

Name	Embed

Add frameworks, libraries, and embedded content here

3. When a dialog appears, click on "Add Other..." button to add frameworks to your project.

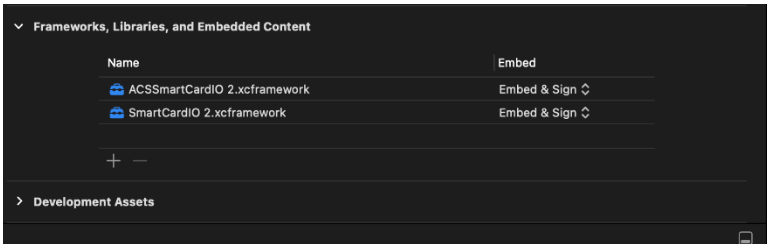


Choose frameworks and libraries to add:

Filter

- BLETest Project
  - ACSSmartCardIO.xcframework
  - BLETestTests.xctest
  - BLETestUITests.xctest
  - SmartCardIO.xcframework
- Apple SDKs
  - Accelerate.framework
  - Accessibility.framework
  - Accounts.framework
  - ACIPCBTLib.tbd
  - ActivityKit.framework
  - AddressBook.framework
  - AddressBookUI.framework
  - AdServices.framework
  - AdSupport.framework

4. You will see the following library added to your project



Frameworks, Libraries, and Embedded Content

Name	Embed
ACSSmartCardIO 2.xcframework	Embed & Sign
SmartCardIO 2.xcframework	Embed & Sign

Development Assets

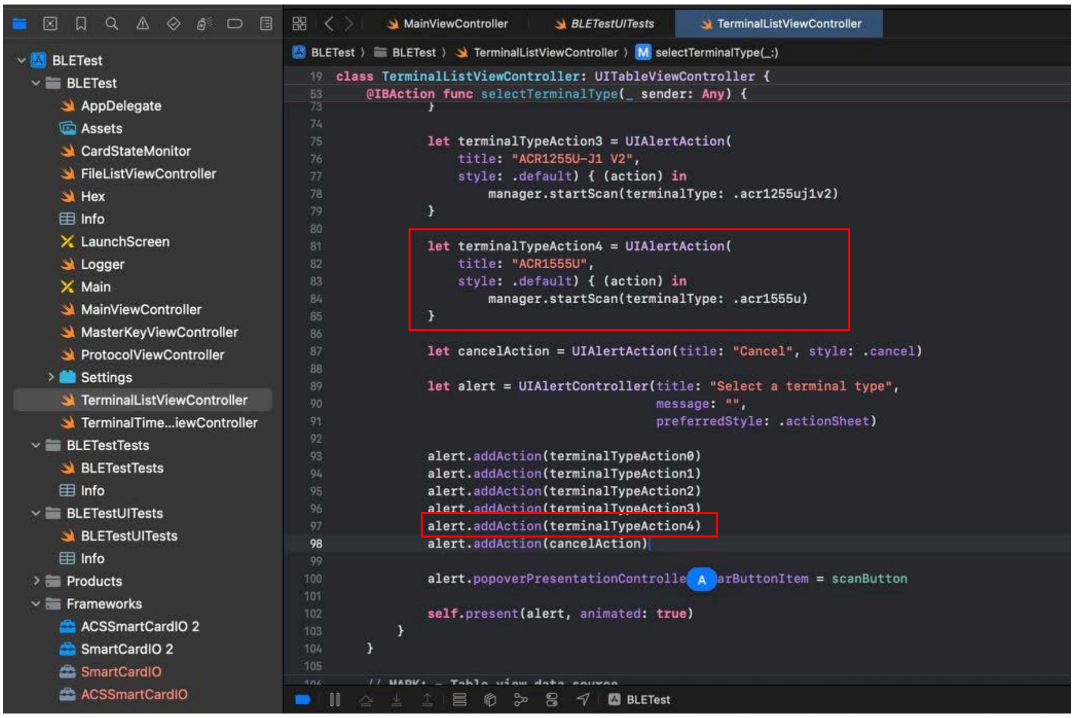
5) Instructions for Adding ACR1555U Terminal Type

Locate the TerminalListViewController and insert the following code after the section of “ACR1255-J1 V2”

```
let terminalTypeAction4 = UIAlertAction{
    title: “ACR1555”,
    style: .default) { (action) in
            manager.startScan(terminalType: .acr1555u)
}
```

And then insert the following code after the alert.addAction(terminalTypeAction3)

```
alert.addAction(terminalTypeAction4)
```



```
class TerminalListViewController: UITableViewController {
    @IBAction func selectTerminalType(_ sender: Any) {
        }

        let terminalTypeAction3 = UIAlertAction(
            title: "ACR1255U-J1 V2",
            style: .default) { (action) in
                manager.startScan(terminalType: .acr1255uj1v2)
        }

        let terminalTypeAction4 = UIAlertAction(
            title: "ACR1555U",
            style: .default) { (action) in
                manager.startScan(terminalType: .acr1555u)
        }

        let cancelAction = UIAlertAction(title: "Cancel", style: .cancel)

        let alert = UIAlertController(title: "Select a terminal type",
                                      message: "",
                                      preferredStyle: .actionSheet)

        alert.addAction(terminalTypeAction0)
        alert.addAction(terminalTypeAction1)
        alert.addAction(terminalTypeAction2)
        alert.addAction(terminalTypeAction3)
        alert.addAction(terminalTypeAction4)
        alert.addAction(cancelAction)

        alert.popoverPresentationControlle arButtonItem = scanButton

        self.present(alert, animated: true)
    }
}
```

6) **Build the Project and Testing:**

After adding the terminal type, proceed to build your project. This should enable the functionality to search for the ACR1555U terminal.

Please note that there may be incompatibility issues when updating the library, so additional modifications to the application may be required.