



UG20001

BBS Reader iOS Generic Programming API User Guide

Rev 1.0 — July 24, 2025

Document information

Information	Content
Keywords	BBS, Bluetooth Connection, Linker Key, Terminal Key, AES-128, Soft PinPad, Session Key, PinBlock, Track2, PSE
Abstract	This iOS SDK user guide provides the details on the how to implement the Soft PinPad API .



Revision History

Revision history

Rev	Date	Description
1.0	July 24, 2025	1 st release of document on iOS Soft PinPad SDK API of BBS Reader

TABLE OF CONTENTS**Table of Contents**

1	Introduction	6
2	System Requirement	6
2.1	iOS SDK Folder Structure	6
2.2	Framework and Library Files	6
2.3	XCode Project Setup	7
2.3.1	Add static library, header, Resource Bundle file from SDK	7
2.3.2	Declare the delegate inside the ViewController.h	8
2.3.3	Delegate Initialization	9
2.3.4	Other Linker Flag under Build Settings	9
2.3.5	Link Binary with Libraries under Build Phases	9
2.3.6	Enable Bluetooth Permission in the Xcode project setting	9
2.3.7	Callback must be implemented to fulfil the Delegate requirements of SDK library.	10
3	Bluetooth Function Descriptions	11
3.1	Bluetooth API	11
3.1.1	(id) init	11
3.1.2	(void) BLEStartScan: (int) iTimeout	11
3.1.3	(int) IsDeviceConnect	11
3.1.4	(void) disconnectBTDevice	11
3.1.5	(void) didConnectReader: (int *)iConnectionStatus	12
3.1.6	(void) didBLEPowerOnState	12
3.1.7	(void) ConnectUUID: (NSString *)sUUID :(int)iTimeout	12
3.2	Device API	12
3.2.1	(NSString *) GetDeviceUUID	12
3.2.2	(NSString *) GetSDKVersion	13
3.2.3	(unsigned int) GetFWVersion: (uint_8 *)fwData :(int *)fwLen	13
3.2.4	(unsigned int) GetTID: (uint_8 *)tidData	13
3.2.5	(int) GetBatteryLevel	13
3.2.6	(unsigned int) Buzzer_Beep	14
4	Biometric Function Description	15
4.1	Biometric Connection API	15
4.1.1	(unsigned int) btConnectBio	15
4.1.2	(unsigned int) btDisconnectBio	15
4.2	Biometric Callback function	15
4.2.1	(void) VerifyStatusCallBack :(int *)Status: (int)iLen: (u_char *)data	15

4.3	Biometric Verification API	16
4.3.1	(unsigned int) btBioVerifyPKBothFPNoAsync:(uint8_t)templateType:(uint8_t *)TemplateData1:(int)iTemplateLength1:(uint8_t *)TemplateData2:(int)iTemplateLength2:(int)iTimeout:(int **)iMatchStatus	16
4.3.2	(unsigned int) btBioVerifyPKNoAsync:(uint8_t)templateType:(uint8_t *)TemplateData:(int)iTemplateLength:(int)iTimeout:(int **)iMatchStatus	17
4.4	Biometric Enrollment API	17
4.4.1	(unsigned int) btBioEnrollExportPKNoAsync:(int)iExportImage:(uint8_t)templateType:(int)iTimeout:(uint8_t *)oTemplateData:(int *)ioTemplateLength:(uint8_t *)oImageData:(int *)ioImageLength	17
5	Smartcard Functions Descriptions	18
5.1	Generic Smartcard Functions	18
5.1.1	(unsigned int) InitReader	18
5.1.2	(unsigned int) CloseReader	18
5.1.3	(unsigned int) DisconnectCard	18
5.1.4	(unsigned int) getSlotStatus:(int *)iCardStatus	18
5.1.5	(unsigned int) ConnectSCCard:(uint8_t)slot:(uint8_t *)pResponse:(uint8_t *)pResponseLen	19
5.2	Smartcard Reader APDU Transceive API	19
5.2.1	(unsigned int) SCTransmit:(uint8_t *)inCmdBuffer:(uint32_t *)inBufferLen:(uint8_t *)outCmdBuffer:(uint32_t *)oBufferLen	19
6	Soft PinPad Function Description	20
6.1	Xcode Project Setup Requirement.....	20
6.1.1	Project must add the Resource.bundle into the Workspace	20
6.1.2	Project must add the Resource.bundle into “Copy Bundle Resources” under Build Phase tab	21
6.1.3	Implement Callback function for Soft PinPad API	21
6.1.4	Soft PinPad Layout.....	22
6.2	Invoke Soft PinPad Layout with Session Key	24
6.2.1	(int) GetUserPin:(id)viewController:(Boolean)mode:(NSString *)sCardPan:(NSString *)sSessionKey	24
6.3	Invoke soft PinPad Layout without Session Key	24
6.3.1	(int) GetUserPinDirect:(id)viewController:(Boolean)mode:(NSString *)sCardPan	24
6.4	Reading EMV Debit Card Primary Account Number (PAN) and Equivalent Track2 Data	25
6.4.1	(int) GetPSE:(uint8_t *)oPANBuf:(uint8_t *)oTrack2ascii	25
7	Appendix 1	26
7.1	Error Code	26
7.2	Enrollment and Verification status code	28
7.3	Workflow for Bluetooth Connection Setup	30

- 7.3.1 Workflow for First Time Bluetooth Setup..... 30
 - 7.3.2 Workflow for Bluetooth Disconnection 31
 - 7.3.3 Workflow for Non-First Bluetooth Connection 31
- 7.4 Workflow for Smartcard Connection Setup32**
 - 7.4.1 Workflow for sending APDU command to Smartcard..... 32
- 7.5 Workflow for Biometric Connection Setup33**
 - 7.5.1 Workflow for Biometric Verification based on external minutiae..... 33
- 7.6 Workflow for Soft PinPad34**
- 8 Abbreviations 34**

1 Introduction

This document describes the Biometric library APIs support in iOS system via Bluetooth Low Energy (BLE) communication.

2 System Requirement

OS Supported: iOS SDK API version 10 onwards

IDE Environment: XCode

Hardware: iPhone and iPad OS operating system with BLE v4.0 onwards

2.1 iOS SDK Folder Structure

BBR10 iOS SDK release history.txt	Text file describes release history
MyKad_Programming_UserManual_iSDK_v1.5	User Guide of Biopad Programming API
BLEReaders.h	Header file
bbsrSdkPinDemo.zip	Demo Source Code
libBLEReaders.a	Library
Resources.bundle	PinPad Layout resource

2.2 Framework and Library Files

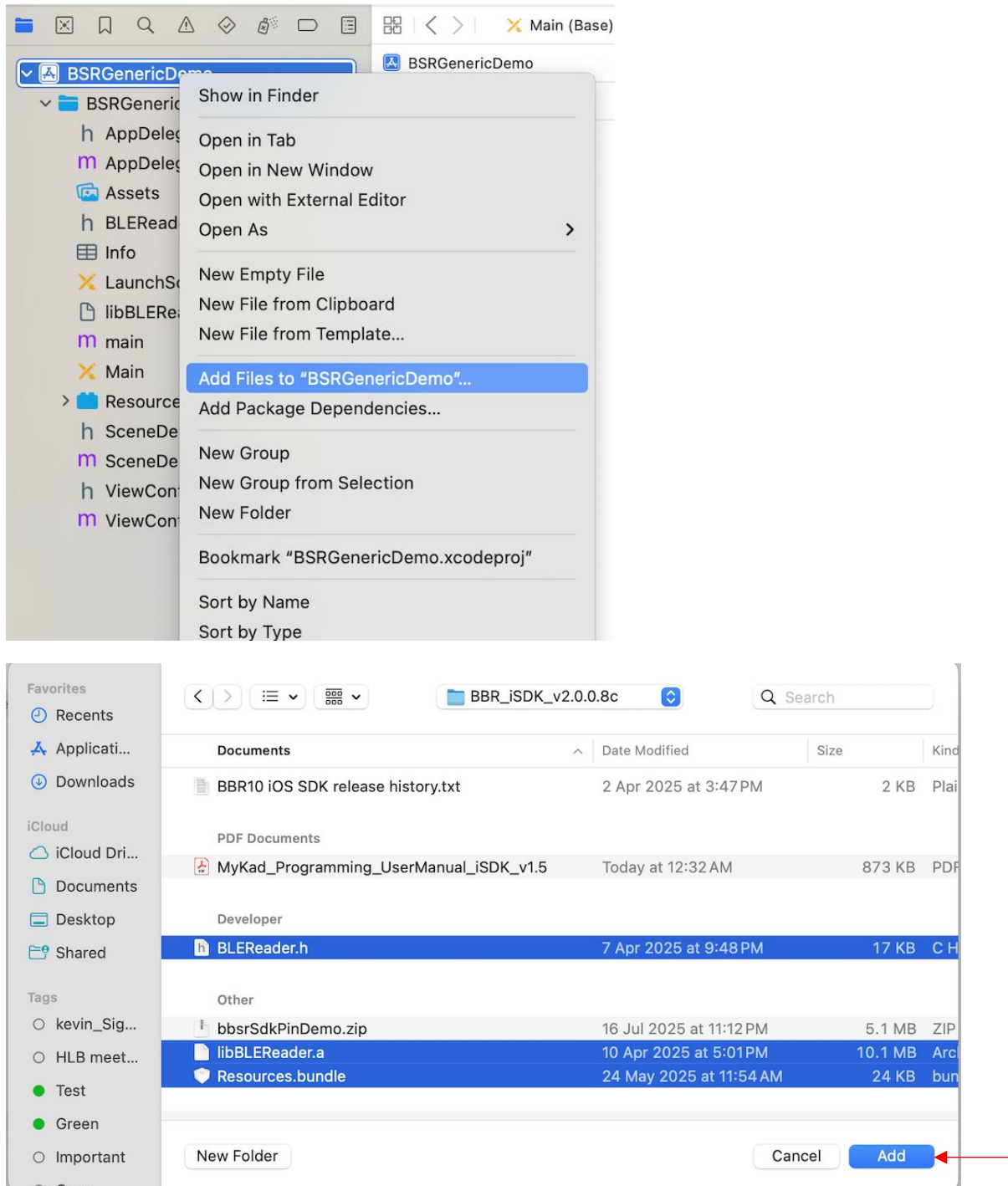
File name:

- a) libBLEReaders.a : Bluetooth Biometric Library
- b) BLEReaders.h : Library Header file
- c) Open SSL header files for cryptography
 - \third-party\OpenSSL\include\openssl\

2.3 XCode Project Setup

2.3.1 Add static library, header, Resource Bundle file from SDK

Developer could add the library and header file by right click on the Project root directory and then go to the “Add Files to **Your Project**”



Developer must choose “Copy files to destination” and check the Target as below

Choose options for adding these files:

Action: Copy files to destination

Groups: Create groups

Targets: ☒ BSRGenericDemo

Cancel Finish

2.3.2 Declare the delegate inside the **ViewController.h**

```
1 //
2 // ViewController.h
3 // BSRGenericDemo
4 //
5 // Created by kevin Chia on 23/7/25.
6 //
7
8 #import <UIKit/UIKit.h>
9 #import "BLEReader.h"
10
11 @interface ViewController : UIViewController<BLEReaderDelegate>{
12     BLEReader *blereader;
13 }
14
15
16 @end
17
18
```



2.3.3 Delegate Initialization

Developer must initialize the delegate object inside the **ViewController.m**,

```
blereader = [[BLEReader alloc] init];
[blereader setDelegate:self];
```

2.3.4 Other Linker Flag under Build Settings

▼ Linking - General

Setting	 bbsrDemo
Dynamic Library Install Name	
Dynamic Library Install Name Base	
Link With Standard Libraries	Yes
Mach-O Type	Executable
Other Librarian Flags	
Other Linker Flags	-ObjC
Quote Linker Arguments	Yes
Separately Edit Symbols	No

2.3.5 Link Binary with Libraries under Build Phases

▼ Link Binary With Libraries (4 items)

Name	Status
 CoreBluetooth.framework	Required
 UIKit.framework	Required
 libBLEReader.a	Required
 Foundation.framework	Required

2.3.6 Enable Bluetooth Permission in the Xcode project setting

Develop needs to add the Privacy – Bluetooth Always usage Description to enable the Bluetooth Permission in the plist as below.

Privacy - Bluetooth Always Usage Description	String	App needs to use Bluetooth to exchange
--	--------	--

2.3.7 Callback must be implemented to fulfil the Delegate requirements of SDK library.

```
// Verify user status called back
- (void)VerifyStatusCallBack :(int *)status :(int)iLen :(u_char *)data
{
    unsigned int value = 0;
    value = *status;

    NSLog(@"VerifyStatusCallBack:: %d", value);
    dispatch_async(dispatch_get_main_queue(), ^{
        switch (value) {
            case BT_OK:
                NSLog(@"Verify Match OK");
                //self->_lblStatus.text = @"Verify Match OK";

                break;
            case BIO_FINGER_NOT_MATCH:
                NSLog(@"Finger Not Match");
                //self->_lblStatus.text = @"Finger Not Match";
                break;

            default:
                break;
        }
    });
}
```

3 Bluetooth Function Descriptions

3.1 Bluetooth API

3.1.1 (id) init

This function initializes the device object.

Parameter : none

Return : device object

3.1.2 (void) BLEStartScan: (int) iTimeout

This function initializes and scans for peripheral devices.

Parameter : iTimeout – A BLE master scan timeout refers to the duration for which a Blue Low Energy (BLE) central device (master) scans for advertising peripheral device before automatically stopping the scan.

Return : void

3.1.3 (int) IsDeviceConnect

This function returns the central device connection status

Parameter : none

Return : 0 - no device connection has been found
1 - device connection has established

3.1.4 (void) disconnectBTDevice

This function is to disconnect the peripheral device (BBS Reader) from central device's (iPhone or iPad) Bluetooth connection

Parameter : none

Return : none

3.1.5 (void) didConnectReader: (int *)iConnectionStatus

This function needs to be implemented to receive the BLE device connection status callback. Upon device connection has been established, central device could start to initial any function API calls to peripheral device.

Parameter : none

Return : iConnectionStatus

- = 0 - no device connection has been found
- = 1 - client device connection has been established
- = 2 - device scan has timeout

3.1.6 (void) didBLEPowerOnState

This is delegate function returns central peripheral status when its BLE power stage in at ON.

Parameter : none

Return : none

3.1.7 (void) ConnectUUID: (NSString *)sUUID :(int)iTimeout

This function is to establishes central device Bluetooth connection with peripheral device via assigned UUID.

Parameter : (NSString *) sUUID – unique identifier for each peripheral.

: (int) iTimeout – A BLE master scan timeout refers to the duration for which a Blue Low Energy (BLE) central device (master) scans for advertising peripheral device before automatically stopping the scan.

Return : none

3.2 Device API

3.2.1 (NSString *) GetDeviceUUID

This function is to return BLE peripheral device's UUID.

Parameter : none

Return : Peripheral device UUID string

3.2.2 (NSString *) GetSDKVersion

This function returns the string of SDK version

Parameter : none

Return : string of SDK version

3.2.3 (unsigned int) GetFWVersion: (uint_8 *)fwData :(int *)fwLen

This function returns the peripheral device of BBS Reader's firmware version

Parameter : (uint_8 *)fwData – pointer for out buffer data of firmware version

: (int *)fwLen – pointer for out buffer length of firmware version

Return : BBS Reader firmware version data

3.2.4 (unsigned int) GetTID: (uint_8 *)tidData

This function returns the BBS Reader Terminal ID (TID)

Parameter : (uint_8 *)tidData – pointer for out buffer data of TID

Return : Reader TID data

3.2.5 (int) GetBatteryLevel

This function returns the current BBS Reader's battery data in integer value.

Parameter : none

Return : FW version < V3.0.0

return value: 0, 10, 20, 40, 60, 80, 100

: FW version > V3.0.0

return value : 0 - 100%

: 255(0xFF) means error battery reading.

3.2.6 (unsigned int) Buzzer_Beep

This function is to trigger buzzer sound from BBS Reader.

Parameter : none

Return : refer to Error Code in Appendix 1.

4 Biometric Function Description

4.1 Biometric Connection API

4.1.1 (unsigned int) btConnectBio

This function is to trigger BBS Reader to power on Biometric fingerprint scanner and to establish the connection.

Parameter : none

Return : FW version < V3.0.0,
return fails if battery level < 20%
: FW version > V3.0.0,
return fails if battery level < 1%

4.1.2 (unsigned int) btDisconnectBio

This function is to trigger BBS Reader to power off the Biometric fingerprint scanner.

Parameter : none

Return : refer to Error Code in Appendix 1.

4.2 Biometric Callback function

4.2.1 (void) VerifyStatusCallBack :(int *)Status: (int)iLen: (u_char *)data

This is Biometric API callback function that it receives return status of the biometrics enrollment or verification API. This callback is a must implementation to avoid compilation error.

Parameter : (int) iStatus - refer to Error Code in Appendix
: (int) iLen – Length of the return data, if any
: (u_char*) data – return data

Return : refer to Error Code in Appendix 1.

4.3 Biometric Verification API

4.3.1 (unsigned int)

```
btBioVerifyPKBothFPNoAsync :(uint8_t)templateType :(uint8_t
*)TemplateData1 :(int )iTemplateLength1 :(uint8_t
*)TemplateData2 :(int )iTemplateLength2 :(int)iTimeout :(int
**)iMatchStatus
```

This function captures a user live fingerprint, and it matches with provided both minutiae from the TemplateData1 and TemplateData2 parameter of API at once. Asynchronous messages have been enabled.

Parameter : (uint8_t) templateType (IN): Export fingerprint minutiae template format. (refer to Appendix)

: (uint8_t *) TemplateData1 (IN): input fingerprint minutiae data

: (int) iTemplateLength1 (IN): input fingerprint minutiae length

: (uint8_t *) TemplateData2 (IN): input fingerprint minutiae data

: (int) iTemplateLength2 (IN): input fingerprint minutiae length

: (int) iTimeout (IN): To specify the wait time of the fingerprint sensor

: (int **) iMatchStatus (OUT)

0 = finger has matched,

-8 = finger does not match

Return : refer to Error Code in Appendix 1.

4.3.2 (unsigned int) btBioVerifyPKNoAsync :(uint8_t)templateType :(uint8_t *)TemplateData :(int)iTemplateLength :(int)iTimeout :(int **)iMatchStatus

This function captures a user live fingerprint, and it matches with provided minutiae from the TemplateData parameter of API. Asynchronous messages have been enabled.

Parameter : (uint8_t) templateType (IN): Export fingerprint minutiae template format. (refer to Appendix)

: (uint8_t *) TemplateData (IN): input fingerprint minutiae data

: (int) iTemplateLength (IN): input fingerprint minutiae length

: (int) iTimeout (IN): To specify the wait time of the fingerprint sensor

: (int **) iMatchStatus (OUT)

0 = finger has matched,

-8 = finger does not match

Return : refer to Error Code in Appendix 1.

4.4 Biometric Enrollment API

4.4.1 (unsigned int)

btBioEnrollExportPKNoAsync :(int)iExportImage :(uint8_t)templateType :
(int)iTimeout :(uint8_t *)oTemplateData :(int
)ioTemplateLength :(uint8_t *)oImageData :(int *)ioImageLength

This function captures a user live fingerprint and export and returns into minutiae format with no asynchronous messages.

Parameter : (int) iExportImage (IN): RFU

: (uint8_t) templateType(IN): fingerprint minutiae template format. (Refer to Appendix)

: (int) iTimeout (IN): duration time in second to wait for user places finger on the sensor before its timeout. 0 value means wait forever.

: (uint8_t *) oTemplateData(OUT): fingerprint minutiae data.

: (int *) ioTemplateLength(OUT): fingerprint minutiae data length.

: (uint8_t *) oImageData(OUT): fingerprint raw image data (image width:256, height:400 pixels). (If iExportImage = 1)

: (int *) ioImageLength(OUT): fingerprint raw image length. (If iExportImage = 1)

Return : refer to Error Code in Appendix 1.

5 Smartcard Functions Descriptions

5.1 Generic Smartcard Functions

5.1.1 (unsigned int) InitReader

This function is to trigger power on the BBS Reader smartcard reader.

Parameter : none

Return : refer to error code

5.1.2 (unsigned int) CloseReader

This function is to trigger power off the BBS Reader smartcard reader.

Parameter : none

Return : refer to error code

5.1.3 (unsigned int) DisconnectCard

This function is to trigger BBS Reader to power off the inserted smartcard based on the slot number.

Parameter : none

Return : refer to error code

5.1.4 (unsigned int) getSlotStatus: (int *)iCardStatus

This function is to get the current connected slot of smartcard from the BBS Reader.

Parameter : none

Return : 0 – Card absents

: 1 – Card present

5.1.5 (unsigned int) ConnectSCCard :(uint8_t)slot :(uint8_t *)pResponse :(uint8_t *)pResponseLen

This function is to trigger BBS Reader to power on the inserted smartcard based on the slot number. smartcard must be inserted into the smartcard connector before calling this function.

Parameter : (uint8_t) slot
0: Smartcard slot,
2: SAM slot
:(uint8_t *) pResponse – return data
:(uint8_t *) pResponseLen – return data length
Return : refer to error code

5.2 Smartcard Reader APDU Transceive API

5.2.1 (unsigned int) SCTransmit :(uint8_t *)inCmdBuffer :(uint32_t *)inBufferLen :(uint8_t *)outCmdBuffer :(uint32_t *)oBufferLen

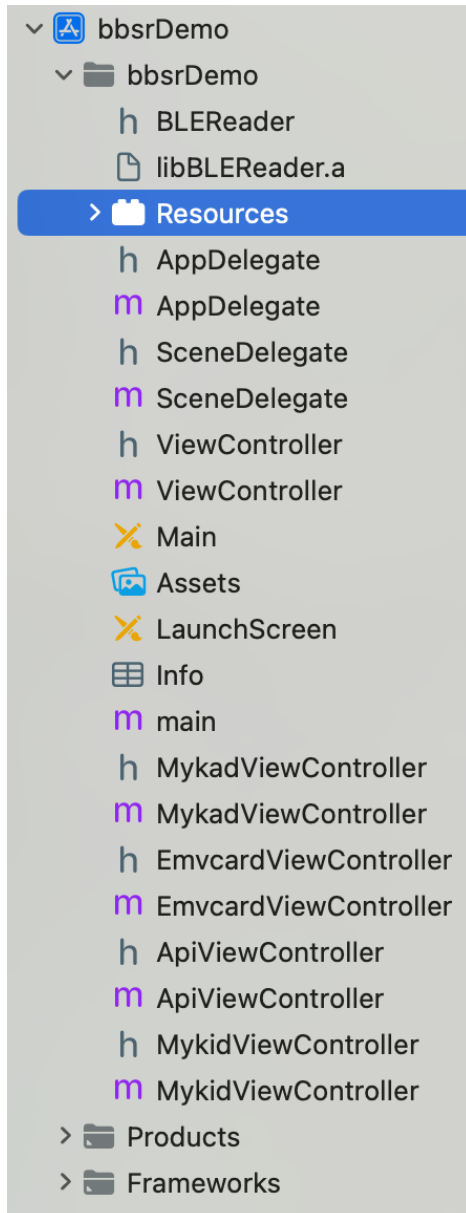
This API allows app to send raw APDU card command to the smartcard of BBS Reader

Parameter : (uint8_t *)inCmdBuffer (IN): input command byte array
:(uint32_t *)inBufferLen (IN): input command length
:(uint8_t *)outCmdBuffer (OUT): response buffer in byte array
:(uint32_t *) oBufferLen (OUT): response data length
Return : refer to error code

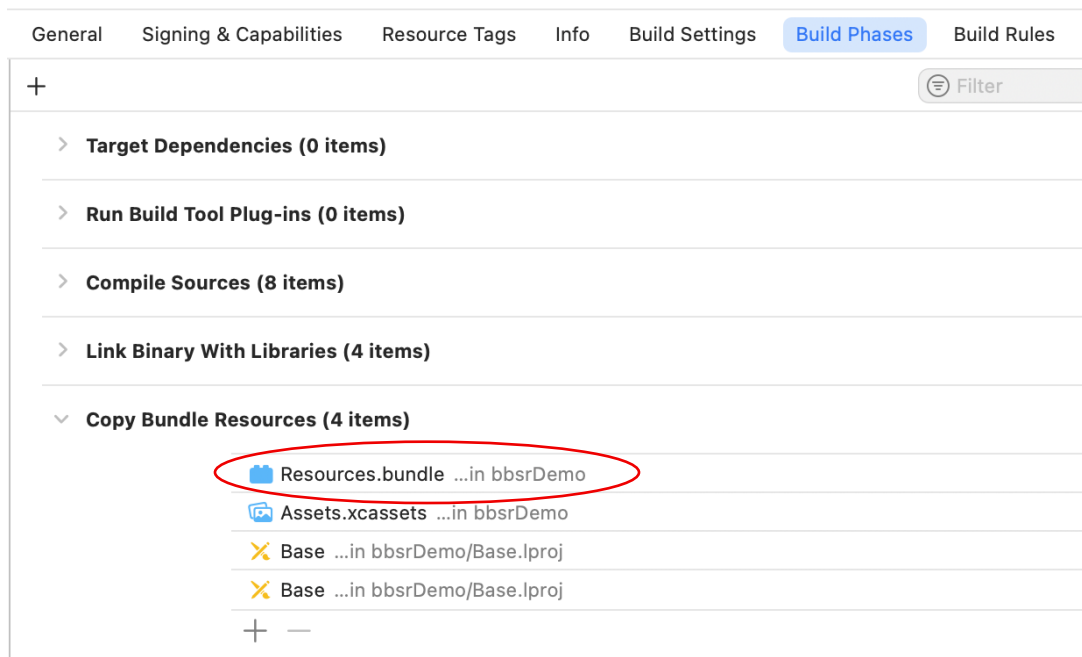
6 Soft PinPad Function Description

6.1 Xcode Project Setup Requirement

6.1.1 Project must add the Resource.bundle into the Workspace



6.1.2 Project must add the Resource.bundle into “Copy Bundle Resources” under Build Phase tab



6.1.3 Implement Callback function for Soft PinPad API

```

-(void) pinblockCallBack:(NSString *)data :(int*)pUiStatus{
    NSString * cardPinBlock = @"";
    NSString *testcardPinBlock = @"2560A46D8205AB3C";
    int value = *pUiStatus;

    if(value == 0){
        NSLog(@"PinBlockUI dismiss: %d",value);
    }else if (value == 1){
        NSLog(@"PinBlockUI show: %d",value);
        if(cardPinBlock != NULL){
            cardPinBlock = data;
            NSLog(@"pin block: %@", cardPinBlock);
            _lblpinblockdata.text = cardPinBlock;
            if([cardPinBlock isEqual:testcardPinBlock]){
                _lblpinblockdata.backgroundColor = [UIColor greenColor];
            }else{
                _lblpinblockdata.backgroundColor = [UIColor redColor];
            }
        }
    }
}

```

6.1.4 Soft PinPad Layout

Please enter 6 digits pin code

1	2	3
4	5	6
7	8	9
	0	
<	C	E

Please enter 6 digits pin code

1	2	3
4	5	6
7	8	9
	0	
<	C	E

Please enter same pin code to confirm

1	2	3
4	5	6
7	8	9
	0	
<	C	E

Please enter same pin code to confirm

Pin Code has Matched

1	2	3
4	5	6
	0	
<	C	E

Matched
Pin Code

[Exit](#)

6.2 Invoke Soft PinPad Layout with Session Key

6.2.1 `(int) GetUserPin:(id)viewController :(Boolean)mode :(NSString *)sCardPan :(NSString *)sSessionKey`

This API is to invoke the Soft PinPad layout as show in the 6.1.4 screenshots.

Parameters : `(id)viewController` – pass the current viewController context

: `(Boolean)mode`

-True means SessionKey is in the encrypted

-False means Session is in the plaintext

: `(NSString *)sCardPan` – EMV Debit Card Primary Account Number

: `(NSString *)sSessionKey` – The encrypted key from Backend Server

Return : refer to error code

6.3 Invoke soft PinPad Layout without Session Key

6.3.1 `int) GetUserPinDirect:(id)viewController :(NSString *)sCardPan`

This API is to invoke the Soft PinPad layout as show in the 6.1.4 screenshots.

Parameters : `(id)viewController` – pass the current viewController context

: `(NSString *)sCardPan` – EMV Debit Card Primary Account Number

Return : refer to error code

6.4 Reading EMV Debit Card Primary Account Number (PAN) and Equivalent Track2 Data

6.4.1 (int) GetPSE:(uint8_t *)oPANBuf:(uint8_t *)oTrack2ascii

Parameters : (uint8_t *)oPANBuf - Out Buffer to store Card PAN data

: (uint8_t *)oTrack2ascii – Out Buffer to store Track2 data

Return : refer to error code

7 Appendix 1

7.1 Error Code

Value	Definition	Description
Bluetooth Error Code		
0	BT_OK	No error
-100	BTERR	General Error
-101	BTERR_INVALID_PARAMETER	Invalid Parameter
-102	BTERR_NO_DEV	No BLE device
-103	BTERR_OPEN_DEV_FAIL	Connect BLE device error
-104	BTERR_DISABLED	Bluetooth disabled
-105	BTERR_NOT_CONNECTED	BLE device not connected
-106	BTERR_NO_IO_STREAM	I/O error
-107	BTERR_TIMEOUT	Timeout Error
-108	BTERR_IO	I/O operation error
-109	BTERR_NO_RESP	I/O no response
-110	BTERR_FRAME	I/O Frame error
-111	BTERR_BATT_LOW	Reader battery low
-112	BTERR_SAM_AUTH	SAM authenticate error
Biometric Error Code		
0	BIO_OK	No error
-1	BIOERR_INTERNAL	Biometric device performed an internal error
-2	BIOERR_PROTOCOLE	Internal Communication Protocol error
-3	BIOERR_CONNECT	Cannot connect biometric device
-5	BIOERR_BADPARAMETER	Invalid Parameter
-6	BIOERR_MEMORY_PC	Not enough memory
-7	BIOERR_MEMORY_DEVICE	Not enough memory for the creation of a database
-8	BIOERR_NO_HIT	Authentication or Identification failed
-9	BIOERR_STATUS	Unknown status error
-10	BIOERR_DB_FULL	The database is full
-11	BIOERR_DB_EMPTY	The Database is empty
-12	BIOERR_ALREADY_ENROLLED	User has already been enrolled
-13	BIOERR_BASE_NOT_FOUND	The specified base does not exist
-14	BIOERR_BASE_ALREADY_EXISTS	The specified base already exist

-17	BIOERR_INVALID_TEMPLATE	The template is not valid
-18	BIOERR_NOT_IMPLEMENTED	Command not yet implemented
-19	BIOERR_TIMEOUT	No response after defined time
-26	BIOERR_CMDE_ABORTED	Command has been aborted
-27	BIOERR_INVALID_PK_FORMAT	Invalid PK format
-28	BIOERR_SAME_FINGER	User gave twice the same finger
-30	BIOERR_INVALID_USER_ID	User ID is not valid
-31	BIOERR_INVALID_USER_DATA	The user data are not valid
-32	BIOERR_FIELD_INVALID	Additional field name length invalid
-33	BIOERR_USER_NOT_FOUND	User ID not found in database
-40	BIOERR_DEV_NOT_CONNECTED	Device not connected
-46	BIOERR_FFD	False Finger Detected
-47	BIOERR_MOIST_FINGER	The finger can be too moist or the scanner is wet
Smartcard Error Code		
-150	SCERR	Smart card general error
-151	SCERR_INVALID_PARAMETER	Invalid parameter input to smartcard functions
-152	SCERR_READER_NOT_AVAILABLE	Reader not available upon initialize reader
-153	SCERR_CARD_NO_RSP	Card not response during transceive
-154	SCERR_OPEN_READER_FAIL	Reader open failed during initialize
-155	SCERR_NO_DEV	No smartcard device
-156	SCERR_CARD_REMOVED	Card Removed
-157	SCERR_NO_CARD	Card Absent
-158	SCERR_DEV_NOT_CONNECT	Smart card device not connected
-159	SCERR_READER_UNAVAILABLE	No smartcard reader available
-160	SCERR_SHARING_VIOLATION	Smartcard reader sharing violation
-161	SCERR_TIMEOUT	Smartcard timeout
-162	SCERR_UNRESPONSIVE_CARD	Smartcard no response
-163	SCERR_UNSUPPORTED_CARD	Smartcard not supported
-164	SCERR_UNPOWERED_CARD	Smartcard not powered
-165	SCERR_RESET_CARD	Smartcard reset
-166	SCERR_INVALID_HANDLE	Invalid smartcard handle
-167	SCERR_AUTHENTICATION	Card Authentication failed

7.2 Enrollment and Verification status code

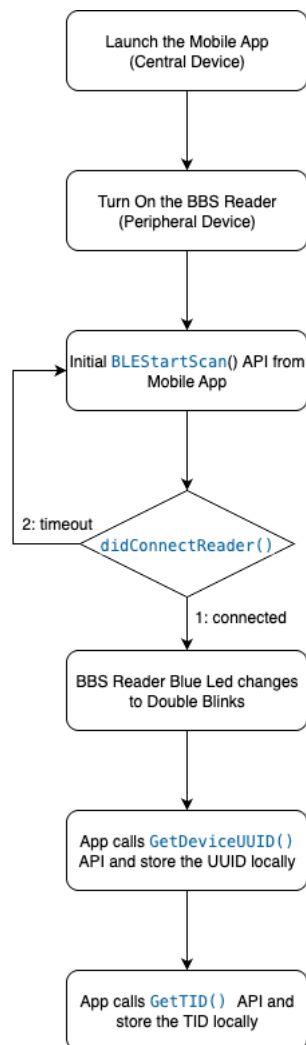
Value	Definition	Description
Biometric Status Code		
0x00	BIO_NO_FINGER	No finger
0x01	BIO_FINGER_UP	Move finger up
0x02	BIO_FINGER_DOWN	Move finger down
0x03	BIO_FINGER_LEFT	Move finger left
0x04	BIO_FINGER_RIGHT	Move finger right
0x05	BIO_FINGER_PRESS_HARDER	Press harder
0x06	BIO_FINGER_LATENT	Latent finger
0x07	BIO_FINGER_REMOVE	Remove finger
0x08	BIO_FINGER_OK	Finger Acquisition OK
0x09	BIO_FINGER_DETECTED	Finger detected
Biometric finger Matching return Code		
0	BIO_FINGER_MATCHED	Finger verification Success
-8	BIO_FINGER_NOT_MATCHED	Finger verification fail
Biometric Enrollment template format		
0x00	PK_COMP_V2	Morpho private format dedicated to terminals
0x02	PK_MAT	Morpho private format dedicated to AFIS
0x03	PK_MAT_NORM	Morpho private format
0x41	ANSI_INCITS_378_2004	ANSI INCITS 378-2004 public finger minutiae record format
0x4C	ISO_19794_2_FMR_2011	ISO/IEC 19794-2 Finger Minutiae Record version 2011
0x4D	ANSI_INCITS_378_2009	ANSI INCITS 378-2009 public finger minutiae record format
0x6C	ISO_19794_2_FMR_CS	ISO/IEC 19794-2 Finger Minutiae Card Record, Compact Size
0x6D	ISO_19794_2_FMR_NS	ISO/IEC 19794-2 Finger Minutiae Card Record, Normal Size
0x6E	ISO_19794_2_FMR	public fingerprint template format, defined by the ISO organization
0x6F	MINEX_A	public fingerprint template format, defined by the NIST organisation for MINEX testing
0x7D	DIN_V66400_FMR	public fingerprint template format, dedicated to use with Smart Cards to perform a Match On Card
0x7E	DIN_V66400_FMR_CS_AA	minutiae ordered by ascending angle

0x7F	ISO_19794_2_FMR_CS_AA	ISO/IEC 19794-2 Finger Minutiae Card Record, Compact Size, minutiae ordered by Ascending Angle

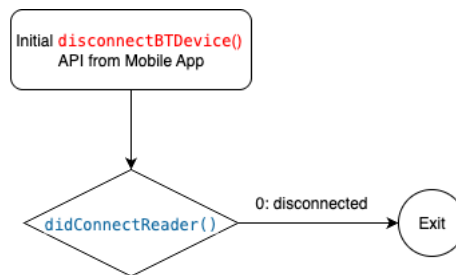
7.3 Workflow for Bluetooth Connection Setup

7.3.1 Workflow for First Time Bluetooth Setup

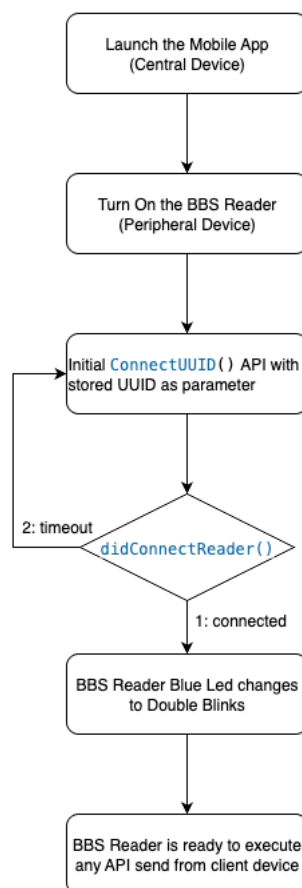
- App needs to store the assigned UUID and TID after central device has established the Bluetooth connection with peripheral device like BBS Reader.



7.3.2 Workflow for Bluetooth Disconnection



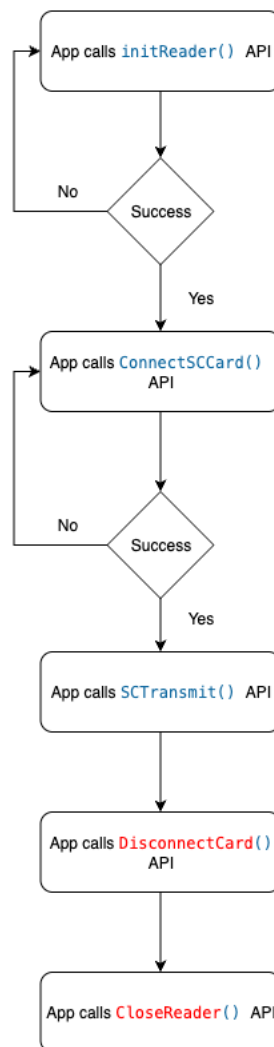
7.3.3 Workflow for Non-First Bluetooth Connection



7.4 Workflow for Smartcard Connection Setup

7.4.1 Workflow for sending APDU command to Smartcard

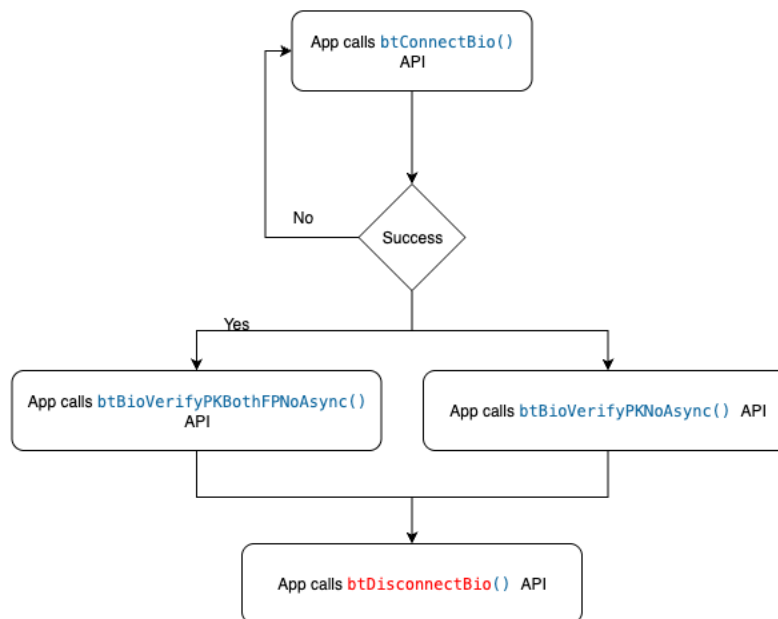
- Developer needs to assign card slot with value 0 for the ConnectSCCard() API
- Developer send APDU command via SCTransmit() API



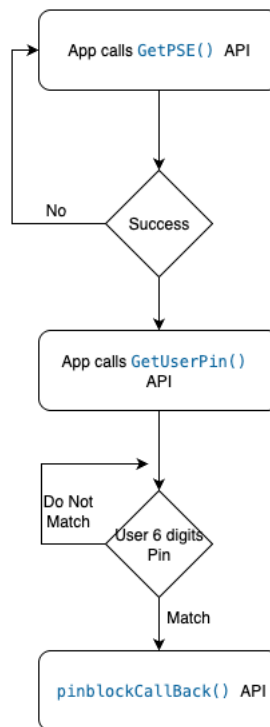
7.5 Workflow for Biometric Connection Setup

7.5.1 Workflow for Biometric Verification based on external minutiae

- Developer needs to know the external minutiae format such as ISO19794-2.
- Developer needs to get the complete external minutiae data via Server or Enrollment API.
- `btBioVerifyPKBothFPNoAsync` API is to verify user live fingerprint with 2 external minutiae at once.
- `btBioVerifyPKNoAsync` API is to verify user live fingerprint with single external minutiae at once



7.6 Workflow for Soft PinPad



8 Abbreviations

Acronym	Description
BLE	Bluetooth Low Energy
BLE Peripheral Device	Peripheral device is a device that advertises its presence and services to other BLE devices, typically in low-power, resource-constrained environment. It acts as a server in the BLE

	communication framework, waiting for a central device to connect to it. Example: BBS Reader
BLE Central Device	Central device acts as a “client” in a BLE connection, actively scanning for and connecting to nearby “peripheral” devices. Its responsible for initiating communication, managing connections, and typically receiving data from peripheral. Example: Smartphones, tables
UUID	In iOS when working BLE, the Bluetooth UUID is used to identify services and characteristics where Apple provides a unique identifier for each peripheral (like a device).