

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

SMUFS-iOS

Application Program Interface (API)

Version 0.3

SMUFS Biometric Solutions Ltd.

www.smufsbio.com

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

Revision History

Date	Version	Description	Author
28/08/14	0.1	Initial draft	rishi@smufsbio.com
09/09/14	0.2	Adding Auto connect back feature, related properties and few new methods like controlling sensor power.	rishi@smufsbio.com
17/12/14	0.3	Added new functions and updated device properties with battery details	rishi@smufsbio.com

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

Contents

1	Introduction	5
1.1	Specification Objectives	5
1.2	Intended Audiences.....	5
1.3	Prerequisites.....	5
1.4	Configuring custom Xcode project with SMUFS SDK.....	5
2	Public Classes	8
2.1	SmufsDevice	8
2.2	FingerImage.....	8
2.3	DeviceProperties	8
2.4	ConnectionState	8
3	SmufsDevice	9
3.1	getSmufsDevices	10
3.2	initialize	11
3.3	getConnectionError	12
3.4	getFingerScan	14
3.5	cancelAsyncScan.....	15
3.6	getSDKVersion	15
3.7	getProperties	15
3.8	StartSensorPower	16
3.9	CloseSensorPower	16
3.10	setDeviceIdleTimeout.....	17
3.11	setDeviceConnectedTimeout	17
3.12	setAutoConnectFeature	18
3.13	setDeviceAutoConectCount	18
3.14	setDeviceAutoConnectFrequency	19
3.15	setDeviceAutoShutdown	19
3.16	setDevicePassword.....	20
3.17	disableDevicePassword	20
3.18	sleep	21
3.19	wakeup	21
3.20	disconnect	21
3.21	isConnected property	22
4	FingerImage.....	23
4.1	ImageCaptureStatus property	23
4.2	getBitmapImage	23
4.3	getWSQImage.....	23
4.4	getGrayScaleImageData	24
4.5	getImageHeight	24
4.6	getImageWidth.....	24
4.7	isValidImage	25

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

5	DeviceProperties	26
5.1	DeviceIdleTimeOutSec.....	26
5.2	DeviceConnectedTimeOutSec	26
5.3	BatteryCutOff_Percentage	26
5.4	DeviceVersion0	26
5.5	DeviceVersion1	26
5.6	ProductID	26
5.7	BootloaderVersion0.....	26
5.8	BootloaderVersion1.....	27
5.9	ProductSerialNumber	27
5.10	MAC_Address	27
5.11	manufacturerString	27
5.12	usbSerialNumber	27
5.13	deviceReleaseNumber	27
5.14	deviceVID	27
5.15	devicePID	27
5.16	AutoConnectBackFeature.....	27
5.17	AutoConnectBackFrequency	27
5.18	AutoConnectBackAttemptCount.....	28
5.19	Current_ConnectBack_MAC_Address.....	28
5.20	connectbackDeviceType.....	28
5.21	IsSensorPowered	28
5.22	ConnectionPasswordEnabled	28
5.23	stateOfCharge.....	28
5.24	Voltage.....	28
5.25	Current.....	28
5.26	Temprature.....	29
5.27	AbsoluteRemainingChargeValue	29
5.28	powerSource	29
5.29	batteryState.....	29
5.30	IsCharging	29
6	ConnectionState	30
6.1	connectionStatus property.....	30
7	Tutorials	31
7.1	SmufsBTSample	31

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

1 Introduction

SMUFS-iOS API provides an easy-to-use software library to communicate and handle the SMUFS mobile biometric products with iOS-based devices.

1.1 Specification Objectives

The objective of the API is to formally document all application programmer interfaces available in the library as well as to provide samples and explanation for easy integration.

1.2 Intended Audiences

The API has the following intended audiences:

- Architecture Team
- Development Team
- Test Team

1.3 Prerequisites

- OS X is required for all iOS development
- You need Xcode 5, you can download it from AppStore

1.4 Configuring custom Xcode project with SMUFS SDK

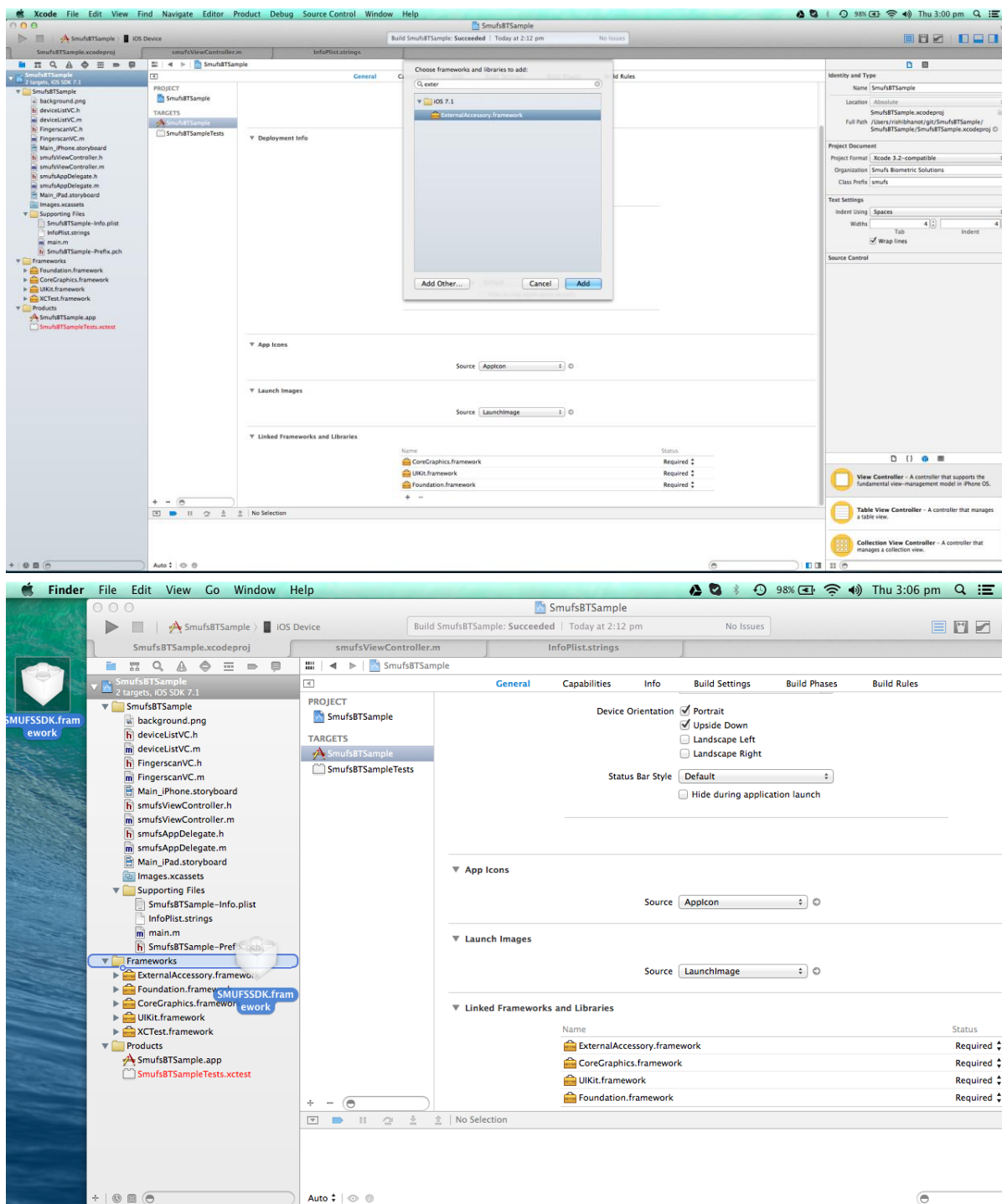
- You need to add the ExternalAccessory.framework to the project. Go to Application Target, Choose General then scroll down to frameworks and click +. Search by “external” and add ExternalAccessory.framework.

- In the SDK_Framework folder you will find the SMUFSSDK.framework. To use it in a custom application simply drag it to your Frameworks Folder in Xcode. Click checkbox for Copy items into destination group's folder (if needed) and finally click Finish.

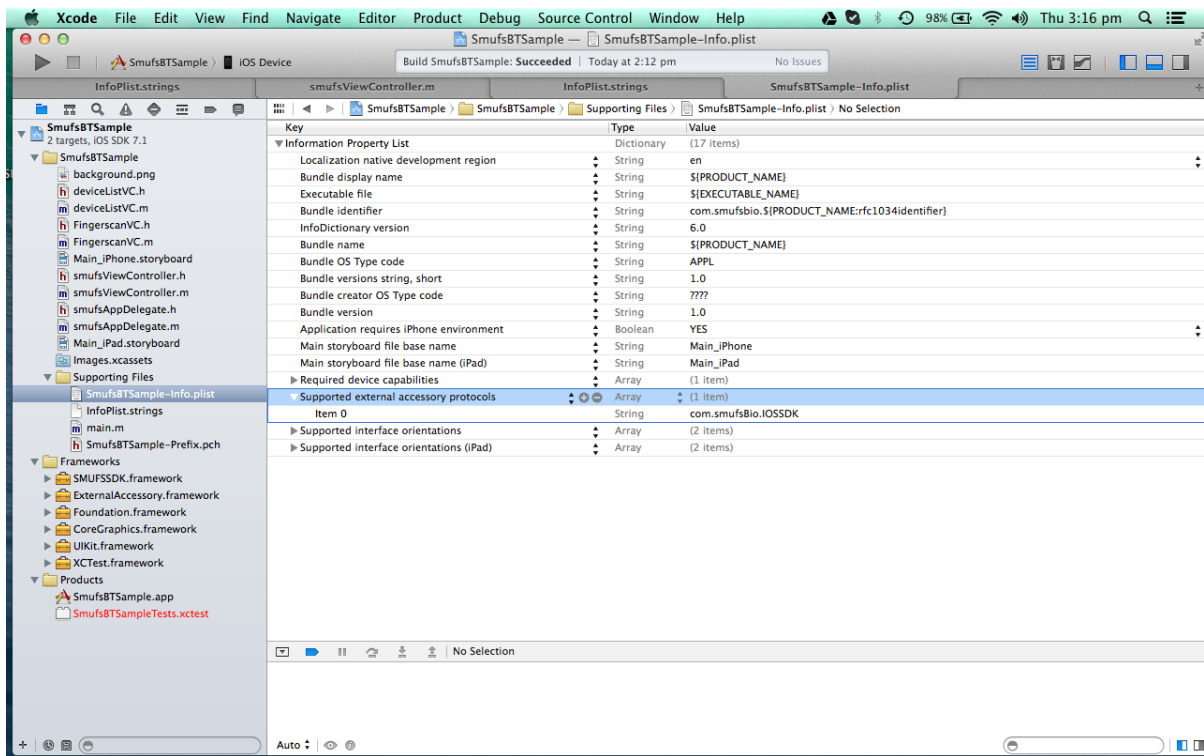
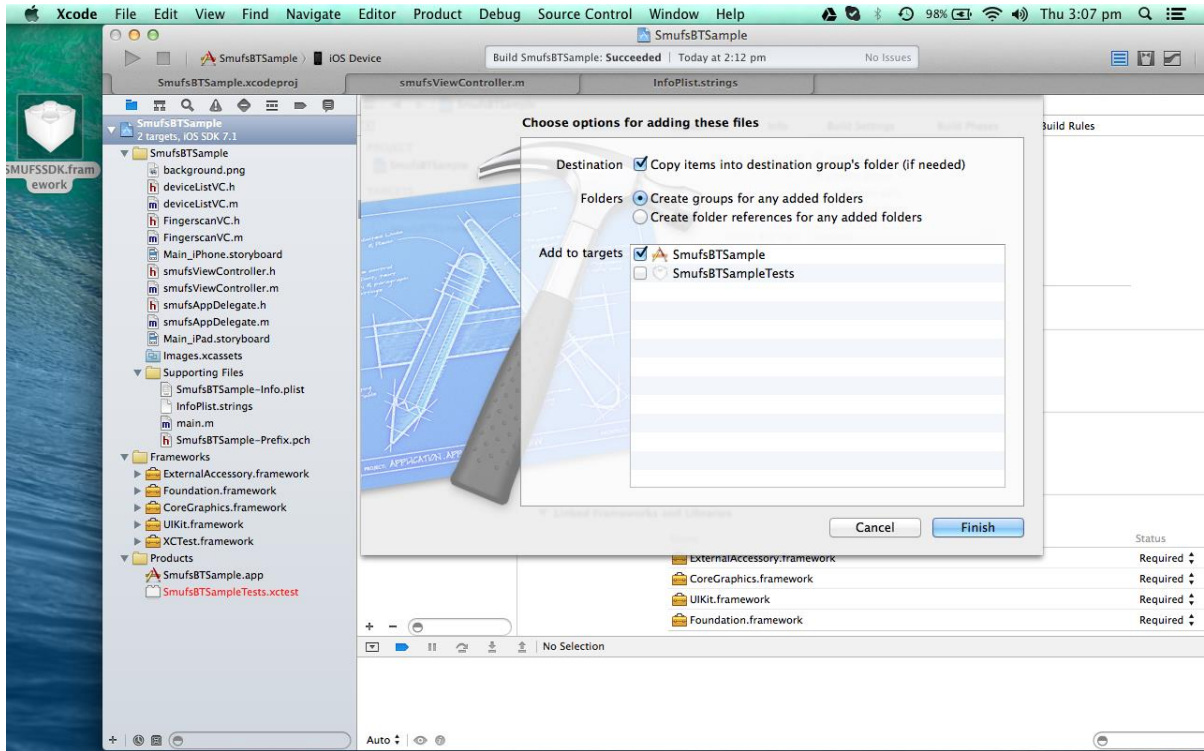
- Add Supported external accessory protocols to your Application info.plist file. Name of the protocol has to be “com.smufsBio.IOSSDK” (it's case sensitive).

Please refer to the following pictures for configuring custom application Xcode project:

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014



SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014



SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

2 Public Classes

Smuf's iOS SDK exposes the following public classes:

2.1 SmufsDevice

It's the main communication class to connect and talk to device. It provides all the required methods to get/set device properties and capture image scan.

2.2 FingerImage

This class holds the captured finger scan status, finger images in all the formats supported by this SDK (currently WSQ and BMP) and its associated properties.

2.3 DeviceProperties

This class holds all the device properties like battery data, timeouts, device access password setting etc

2.4 ConnectionState

This class have the property of Connection Status used in connection status callbacks.

Imports Required to access these classes

```
#import <SmufsSDK/SmufsDevice.h>
#import <SmufsSDK/FingerImage.h>
#import <SmufsSDK/DeviceProperties.h>
```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3 SmufsDevice

This is the main class providing all the communication methods with the Smuf's device. This section defines all the methods and properties provided by this class. Typical usage case will be as follows:

- Call **getSmufsDevices** to fetch the list of connected Smuf's devices.
- Initialize the SmufsDevice object by invoking **initialize** method.
- Get all device properties by invoking **getProperties** method.
- Start the sensor power by invoking **StartSensorPower** method.
- Capture the finger scans by invoking **getFingerScan** method.
- Close the sensor power by invoking **CloseSensorPower** method.
- When done with the device close the device connection by **disconnect** method.

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.1 **getSmufsDevices**

This method returns all the connected Smuf's Devices with iOS device. You first have to connect to the Smuf's device from Apple Device Bluetooth Settings Page. Then SDK can talk to connected device. Smuf's SDK CAN NOT pair to Smuf's device from within the application it has to be done through Apple Device | Bluetooth Settings Page.

getSmufsDevices

```
+ NSMutableArray* getSmufsDevices()
returns the list of connected SMUFS Bluetooth devices
Parameters:
    • None
Returns:
    • NSMutableArray of connected Smufs Devices.
```

Typical usage will be:

```
NSMutableArray* smufsDevices = [SmufsDevice getSmufsDevices];
```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.2 initialize

This method initializes the SDK and tries to connect to the desired Smufs Device. This should be the first method which user application calls after getSmufsDevices. If it successfully connects to the desired device it returns the SmufsDevice object otherwise it returns nil.

There is no way to detect if device is password protected or not before connecting to it. So first you can try with nil and if it fails with INVALID_PASSWORD then you can prompt the user to capture the device password.

Typical usage will be:

```
smufsDev = [SmufsDevice initialize:name :pwd :self :@selector(DeviceConectionStatusCallback:)];
```

initialize

```
+ (id)initialize:(NSString *)deviceName :(NSString*)devicePassword :(id)target :(SEL)sel;
```

Initializes and connects to SMUFS Bluetooth device

Parameters:

- deviceName – SMUFS Bluetooth device name to connect to
- devicePassword – device password if device is password protected else nil
- id:Sel – Connection status callback id and selector, where SDK delivers the device connection and disconnection notifications

Returns:

- SmufsDevice object for communicating with the device or nil if connection failed

ConnectionState Callback Signature

```
-(void)DeviceConectionStatusCallback :(ConnectionState*)state
{
    //Application handles device disconnection and connect back events
    //state.connectionStatus is DEVICE_CONNECTION_STATUS enumeration
}
```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.3 getConnectionError

This method returns the details of most recent connection attempt result. You should use this method if initialize call returns nil to get the reason of failure.

getConnectionError

+(Connection_Errors)getConnectionError;
Returns the connection error code.

```
typedef NS_ENUM(uint8_t, Connection_Errors)
{
    CONNECTION_SUCCESSFULL = 0,
    DEVICE_NOT_CONNECTED = 1,
    INVALID_FIRMWARE_VERSION = 2,
    COMMUNICATION_ERROR = 3,
    INVALID_PASSWORD = 4,
    SESSION_ALREADY_EXISTS = 5,
    UNKNOWN = 6
};
```

Typical Usage

```
smufsDev = [SmufsDevice initialize:devname :nil :self :@selector(DeviceConectionStatusCallback:)];

if (smufsDev == nil)
{
    NSLog(@"errCode = %@",[self ConnectionErrorString:[SmufsDevice getConnectionError]]);
}

- (NSString*)ConnectionErrorString:(Connection_Errors)err
{
    NSString *result = nil;

    switch(err) {
        case CONNECTION_SUCCESSFULL:
            result = @"CONNECTION_SUCCESSFULL";
            break;
        case DEVICE_NOT_CONNECTED:
            result = @"DEVICE_NOT_CONNECTED";
```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

```

        break;
    case INVALID_FIRMWARE_VERSION:
        result = @"INVALID_FIRMWARE_VERSION";
        break;
    case COMMUNICATION_ERROR:
        result = @"COMMUNICATION_ERROR";
        break;

    case SESSION_ALREADY_EXISTS:
        result = @"SESSION_ALREADY_EXISTS";
        break;

    case INVALID_PASSWORD:
        result = @"INVALID_PASSWORD";
        break;

    default:
        result = @"UNKNOWN";
    }

    return result;
}

```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.4 getFingerScan

The method captures the finger scan from the device. Before calling this method sensor needs to be powered, you can power up the sensor by StartSensorPower method. If you will try to capture image without powering up the sensor first it will fail with SENSOR_NO_POWER but then it will start the sensor power. Powering up the sensor takes around 1.6 seconds so second call to getFingerScan will return you the image. Based on the parameters it can capture synchronously or asynchronously. In the case of asynchronous mode user has to provide the callback method of their application where SDK can deliver the capture status messages. Please refer to the sample code in the iOS Sample application folder.

In the case of asynchronous scan you need to provide callback of the type SEL and you will get FingerImage object in this method.

Status of the finger scans and finger image can be retrieved from the [Finger Image class](#).

FingerImage Callback Signature

```
-(void) callbackMethod:(FingerImage*) fi
```

getFingerScan

```
- (FingerImage*) getFingerScan : (BOOL) AsyncScan : (int) timeout : (id)target : (SEL)sel ;
```

Commands the scanner to capture the finger scan and return to the application synchronously or asynchronously.

Parameters:

AsyncScan – Set to true if you want asynchronous capture and false for synchronous scan.

timeout – Timeout to wait for finger scan in seconds.

sel – Application defined callback method for asynchronous scan updates or nil for synchronous call.

Returns:

- FingerImage Object Containing image data, capture status messages and image properties

See Also:

- **FingerImage**

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.5 **cancelAsyncScan**

This method cancels the finger scan process if device is currently capturing. It's a Synchronous method, which blocks the control until it closes the scan process.

cancelAsyncScan

-(void) cancelFingerScan

Cancels the scanning process if currently capturing asynchronously.

3.6 **getSDKVersion**

This method returns the SDK version of the iOS Smufs SDK.

Version format is : MSB_B2_B1_LSB : MSB = MAJOR, Byte2 = MINOR and BYTE1 & LSB is 16 bit revision number.

getSDKVersion

-(int) getSDKVersion

Returns : The SDK version number in integer

3.7 **getProperties**

This method fetches all the device properties. Please refer to [DeviceProperties](#) Class for detailed properties.

getProperties

-(DeviceProperties*)getProperties;

Fetches the connected SMUFS device properties

Returns:

- DeviceProperties Object containing SMUFS device information, nil if there is any error.

See Also:

- [DeviceProperties](#)

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.8 StartSensorPower

This method will switch ON the sensor power if sensor is in deep sleep state. It can help your application to start the device power whenever really needed. It will return true immediately after instructing the device to start sensor power. On the device side it will take around 1.6 seconds to put the sensor in active state and ready to be used. After invoking this method you should make sure whether the sensor is ready or not by retrieving the device properties and checking for [IsSensorPowered](#) before invoking [getFingerScan](#) function. It's better to call [getProperties](#) at least after 1.5+ seconds as device will be busy powering up the sensor. If you will start calling [getProperties](#) earlier then it will delay the powering up process.

Alternatively you can also switch ON the sensor power by calling [getFingerScan](#) method directly, if sensor is in deep sleep and device gets a finger scan request, device will return with SENSOR_NO_POWER and will start the sensor power immediately so that device can be ready for the next finger scan request. Even in this case you should check [IsSensorPowered](#) after around 1.5+ seconds to make sure power is up before calling [getFingerScan](#) again.

StartSensorPower

-(BOOL) StartSensorPower;

Switches ON the sensor power and places it in active state.

Returns:

- true if successful else false

Typical usage

```
[smufsdev StartSensorPower];
[NSThread sleepForTimeInterval:2.00]; //or update UI
DeviceProperties* devPoperties = [smufsdev getProperties];
While(!devProperties.IsSensorPowered)
    devPoperties = [smufsdev getProperties];
```

3.9 CloseSensorPower

This method places the sensor in deep sleep state, which can save lot of battery power. In your application you should try to use this method when you are sure your application don't need scans for the next few minutes or so. Powering up sensor back will take around 1.4 seconds so optimize the battery power usage as applicable by the application scenario.

CloseSensorPower

-(BOOL) CloseSensorPower;

Switches OFF the sensor power and places it in deep sleep state.

Returns:

- true if successful else false

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.10 **setDeviceIdleTimeout**

This method sets the auto shutdown idle timeout in seconds for the device, when the device is in Bluetooth discoverable mode and it's idle for this given timeout period it will auto shutdown itself if running on battery. Valid range is 180 seconds (3 minutes) to 900 seconds (15 minutes). To disable the auto idle timeout feature you can set it to 0.

setDeviceIdleTimeout

```
-(BOOL) setDeviceIdleTimeout :(int)timeOutsecs;
```

Sets the device idle timeout in seconds.

Returns:

- true if successful else false

3.11 **setDeviceConnectedTimeout**

This method sets the auto shutdown connected timeout in seconds for the device, when the device is in Bluetooth connected mode and there is no communication on the connected channel for this given timeout period it will auto shutdown itself only if running on battery. Valid range is 180 seconds (3 minutes) to 900 seconds (15 minutes). To disable the auto connected timeout feature you can set it to 0.

setDeviceConnectedTimeout

```
-(BOOL) setDeviceConnectedTimeout :(int)timeOutsecs;
```

Sets the device connected timeout in seconds.

Returns:

- true if successful else false

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.12 **setAutoConnectFeature**

This method switches ON the Auto Connect feature. Once it's switched ON device will start remembering the last connected BT phone/tablet address. Then when Smufs device is switched ON or if it loses Bluetooth connection during normal operation it immediately starts trying to connect back to the last connected device for DeviceProperty.AutoConnectCount times at DeviceAutoConnectFrequency rate.

setAutoConnectFeature

-(BOOL) setAutoConnectFeature :(BOOL)enabled;

Sets the device connected timeout in seconds.

Parameters:

enabled – Set to true to enable AutoConnectFeature and set to false to disable.

Returns:

- true if successful else false

3.13 **setDeviceAutoConectCount**

This method sets the number of connection attempts while Smuf's device is trying to connect back. Valid Range is 1 to 50. If Smuf's device cannot connect back either when other BT device is not switched on or is out of range then after these many attempts device will go to normal discoverable mode. While trying to connect back Blue LED will be blinking fast, when in discoverable mode Blue LED will blink once a second.

setDeviceAutoConectCount

-(BOOL) setDeviceAutoConectCount :(int)count;

Sets the device auto connect back count

Parameters:

count – integer from 1 to 50.

Returns:

- true if successful else false

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.14 **setDeviceAutoConnectFrequency**

This is the time duration device waits while in auto connect mode before attempting the next auto connect. This time can be very crucial as Smuf's device will get a chance to accept any new incoming pairing or connection requests from other BT devices. Valid Range is 1 to 5 seconds.

setDeviceAutoConnectFrequency

-(BOOL) setDeviceAutoConnectFrequency :(int)frequencysecs;

Sets the device auto connect frequency in seconds

Parameters:

frequencysecs – integer from 1 to 5.

Returns:

- true if successful else false

3.15 **setDeviceAutoShutdown**

This method will set the battery threshold in percentage at which device should auto shutdown to protect the battery reaching drained state. Valid range is 5% to 20%

setDeviceAutoShutdown

-(BOOL)setDeviceAutoShutdown:(int) batteryPercent;

Sets the device battery percentage threshold to enable auto shutdown.

Parameters:

batteryPercent – integer from 5 to 20.

Returns:

- true if successful else false

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.16 setDevicePassword

This method enables the device access password. Once this password is set on the device, every initialize call will require this password or device connection will fail with Connection Error code INVALID_PASSWORD. Valid range is 6 to 16 characters.

setDevicePassword

```
-(BOOL) setDevicePassword :(NSString*)devicePassword;
```

Sets the device access password.

Parameters:

devicePassword – String from 6 to 16 characters.

Returns:

- true if successful else false

3.17 disableDevicePassword

This method disables the device access password. You have to provide the valid password previously set through setDevicePassword to disable the access password. Valid range is 6 to 16 characters.

disableDevicePassword

```
-(BOOL) disableDevicePassword :(NSString*)devicePassword;
```

Sets the device access password.

Parameters:

devicePassword – String from 6 to 16 characters.

Returns:

- true if successful else false

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.18 sleep

This method puts the SDK in sleep state. Actually SDK closes its active channel, saves the context and stop its all execution threads. This method is helpful when application has to manage going to background eg: when user presses the Quit button, call, SMS interruption. Application should call this method to put SDK to sleep from applicationWillResignActive or applicationDidEnterBackground application delegates.

sleep

```
-(BOOL)sleep;
```

Puts the SDK to sleep state.

Returns:

- true if successful else false

3.19 wakeup

This method wakes up the SDK from sleep state. In this method SDK attempts to connect back to the Smufs Device which was connected at the time of sleep with same parameters (like device password). This method is helpful when application has to manage coming to foreground state. Application should wakeup SDK from applicationWillEnterForeground or applicationDidBecomeActive application delegates.

wakeup

```
-(BOOL)wakeup;
```

Wakes up the SDK from sleep state.

Returns:

- true if successful else false

3.20 disconnect

This method disconnects from the connected Smuf's device and releases all the associated resources. This method should be the last method which user application calls after user application is done with the device. Optionally you can request device to shut down if it's running on battery.

disconnect

```
-(void)disconnect :(BOOL)shutdownDevice;
```

Disconnects from the connected device.

Parameters:

shutdownDevice : set to true to request device to shut down if running on battery else false

Returns: true if successful else false

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

3.21 isConnected property

This Boolean property value will indicate the current connection status. This property can be used when application is not using the connection status call backs. It's always better and strongly recommended to use callbacks instead of this property.

isConnected

```
@property (nonatomic,readonly) BOOL isConnected;
```

Typical Use:

```
- If(!smufsdevice_object.isConnected)
- {
-     //device not connected ...
- }
```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

4 FingerImage

4.1 ImageCaptureStatus property

ImageCaptureStatus property returns the current status of finger scan.

Values for ImageCaptureStatus can be as follows:

- GRABIMAGE_FAILED
- FINGER_NOT_ON_SENSOR
- SENSOR_NO_POWER
- CAPTURING_ASYNC
- IMAGE_CAPTURED
- ERROR_UNKNOWN
- INVALID_CALLBACK_PROVIDED
- FINGERIMAGE_TIMEDOUT
- ALREADY_CAPTURING
- USER_CANCELED_CAPTURE
- DEVICE_DISCONNECTED
- COMPRESSING_WSQ

4.2 getBitmapImage

This method returns the iOS UIImage Bitmap class object for the last captured image.

getBitmapImage

- (UIImage*) getBitmapImage;

Gets the RAW bitmap image

Returns:

- Bitmap of the grayscale image

4.3 getWSQImage

This method returns the byte buffer filled with WSQ image data for the last captured image.

getWSQImage

-(NSData*) getWSQImage;

Gets the WSQ image data

Returns:

- Byte array of the wsq image data

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

4.4 **getGrayScaleImageData**

This method returns the byte array for the last captured image or “nil” if image is not valid or other problem.

getGrayScaleImageData

```
- (NSData*) getGrayScaleImageData;
Gets the RAW Gray Scale Image
Returns:


- Byte Array of the grayscale image

```

4.5 **getImageHeight**

This method returns the Height for the last captured image in pixels.

getImageHeight

```
-(int) getImageHeight()
Gets the image height in pixels
Returns:


- The number of pixels of the image height

```

4.6 **getImageWidth**

This method returns the Width for the last captured image in pixels.

getImageWidth

```
-(int) getImageWidth()
Gets the image width in pixels
Returns:


- The number of pixels of the image width

```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

4.7 isValidImage

This method returns the validity status of image. Either true if image is valid or false otherwise.

isValidImage

-(BOOL) isValidImage;

Gets the image validity status

Returns:

- True if the image is valid, otherwise false

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

5 DeviceProperties

This class holds all the Properties of a connected Smuf's device.

5.1 DeviceIdleTimeOutSec

This integer property is auto shutdown idle timeout in seconds for the device, when the device is in Bluetooth discoverable mode and it's idle for this given timeout period then it will auto shutdown itself only if running on battery. 0 means never auto shutdown. Default is 180 seconds (3 minutes).

5.2 DeviceConnectedTimeOutSec

This integer property is auto shutdown connected timeout in seconds for the device, when the device is in Bluetooth connected mode and there is no communication on the connected channel for this given timeout period it will auto shutdown itself only if running on battery. 0 means never auto shutdown. Default is 180 seconds (3 minutes).

5.3 BatteryCutOff_Percentage

This integer property is battery threshold in percentage at which device should auto shutdown to protect the battery reaching drained state. Default is 5%. Device will auto shutdown only if running on Battery.

5.4 DeviceVersion0

This byte property returns the Device major firmware version.

5.5 DeviceVersion1

This byte property returns the Device minor firmware version.

5.6 ProductID

This byte property returns the Product Id of the device. Product IDs for the Smuf's devices are as follows:

- SMUFS BTWSQ, Product ID = 16

5.7 BootloaderVersion0

This byte property returns the Device Boot loader major firmware version.

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

5.8 **BootloaderVersion1**

This byte property returns the Device Boot loader minor firmware version.

5.9 **ProductSerialNumber**

This string property returns the unique serial number of the device.

5.10 **MAC_Address**

This string property returns the device Bluetooth MAC address.

5.11 **manufacturerString**

This string property returns the Manufacturer String.

5.12 **usbSerialNumber**

This string property returns the USB serial number.

5.13 **deviceReleaseNumber**

This integer property returns the device Release Number.

5.14 **deviceVID**

This integer property returns USB device Vendor ID.

5.15 **devicePID**

This integer property returns USB device Product ID.

5.16 **AutoConnectBackFeature**

This Boolean property returns whether Auto connect back feature is switched on or not.

5.17 **AutoConnectBackFrequency**

This integer property has the time duration device waits while in auto connect mode before attempting the next auto connect. This time can be very crucial as Smuf's device will get a chance to accept any new incoming pairing or connection requests from other BT devices. Default is 1.

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

5.18 **AutoConnectBackAttemptCount**

This integer property holds the number of connection attempts that Smuf's device will try to connect back before switching to discoverable mode. Default is 5.

5.19 **Current_ConnectBack_MAC_Address**

This string property returns the mac address of the last successfully connected Bluetooth device. Auto Connect feature will use this address while trying to auto connect.

5.20 **connectbackDeviceType**

This property returns the type of the last successfully connected Bluetooth device. Possible values are:

- Apple (iOS)
- SPP (Android or windows)
- NA (not applicable or unknown)

5.21 **IsSensorPowered**

This property returns the current status of the finger print sensor. It's true if sensor is in active state and ready to capture scans else false if sensor is in deep sleep state. You can use this property before going for finger capture.

5.22 **ConnectionPasswordEnabled**

This Boolean property returns the current state of Device access password. If device access password is enabled it's true otherwise will be false.

5.23 **stateOfCharge**

This float property returns the current battery level percentage.

5.24 **Voltage**

This integer property returns the current battery voltage level in millivolts.

5.25 **Current**

This integer property returns the battery current. It will be negative if battery is discharging and positive if battery is charging.

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

5.26 Temperature

This float property returns the temperature on the battery terminals in Celsius.

5.27 AbsoluteRemainingChargeValue

This integer property returns the absolute charge value remaining in battery.

5.28 powerSource

This property returns the current power source of the device. Possible values are:

- SMUFS_BATTERY = 1,
- SMUFS_USB_SRC = 2,
- SMUFS_EXTPWR = 3,
- SMUFS_PWR_UNKNOWN = 4

5.29 batteryState

This property returns the current state of the battery. Possible values are:

- BATTERY_CHARGING = 1,
- BATTERY_DISCHARGING = 2,
- BATTERY_LOW = 3,
- BATTERY_CRITICALLOW = 4,
- BATTERY_IDLE = 5,
- BATTERY_UNKNOWN = 6

5.30 IsCharging

This Boolean property returns true if battery is charging otherwise false.

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

6 ConnectionState

Object of this class is provided for device connection status callback.

6.1 connectionStatus property

This property returns the current connection state.

connectionStatus

```
@property (nonatomic,readonly) DEVICE_CONNECTION_STATUS connectionStatus;
```

Typical Use:

```
-(void)DeviceConnectionStatusCallback :(ConnectionState*)state
{
    //Application handles device disconnection and connect back events
    If(state.connectionStatus == DEVICE_CONNECTION_DROPPED)
        //handle connection dropped ...
}
```

Enumeration DEVICE_CONNECTION_STATUS

```
typedef NS_ENUM(uint8_t, DEVICE_CONNECTION_STATUS)
{
    DEVICE_CONNECTION_DROPPED = 0,
    CAN_NOT_RECONNECT_AS_CONNECTBACK_DISABLED =2,
    DEVICE_CONNECTEDBACK =3,
    DEVICE_RECONNECTION_FAILED = 4,
};
```

SMUFS-iOS	Document ID: SMUFS-005	Version: 0.3
Application Program Interface (API)		Version Date: Dec, 17,2014

7 Tutorials

This section will describe the example application(s) that are provided as part of the API.

7.1 SmufsBTSample

Please refer “SmufsBTSample” provided in Source folder of SDK package.