TransUnion

HKID Card Identification SDK-IOS Documentation

Aug 2019

Version 1.1.9

Document History

Version	Description	Prepared By	Date
1.0	Initial version	TU	Jun 5, 2019
1.1	Added description to system and CPU architecture support Added description to code structure Added explanation to code logic	TU	July 23, 2019
1.1.6	Updated detection status list and handling	TU	Aug 6, 2019
1.1.8	Updated detection status, result list and handling	TU	Aug 15, 2019
1.1.9	Updated SDK Header file description	TU	Aug 26, 2019

Table of Contents

1. Introduction

1.1. Objective

Describe the details and usage of HKID Card Identification SDK for third parties.

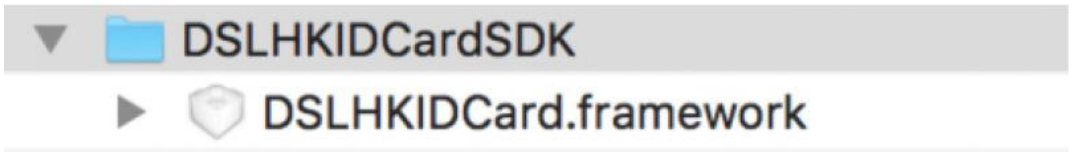
1.2. Version Support

iOS 8.0 or above.

2. SDK Integration

2.1. Extract SDK

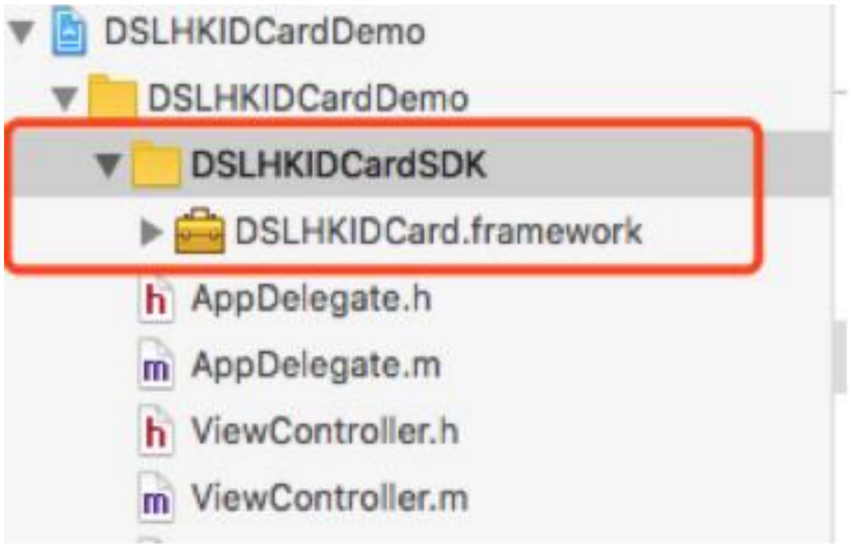
- Unzip DSLHKIDCardSDK.zip:



DSLHKIDCardSDK
DSLHKIDCard.framework

2.2. Importing the SDK

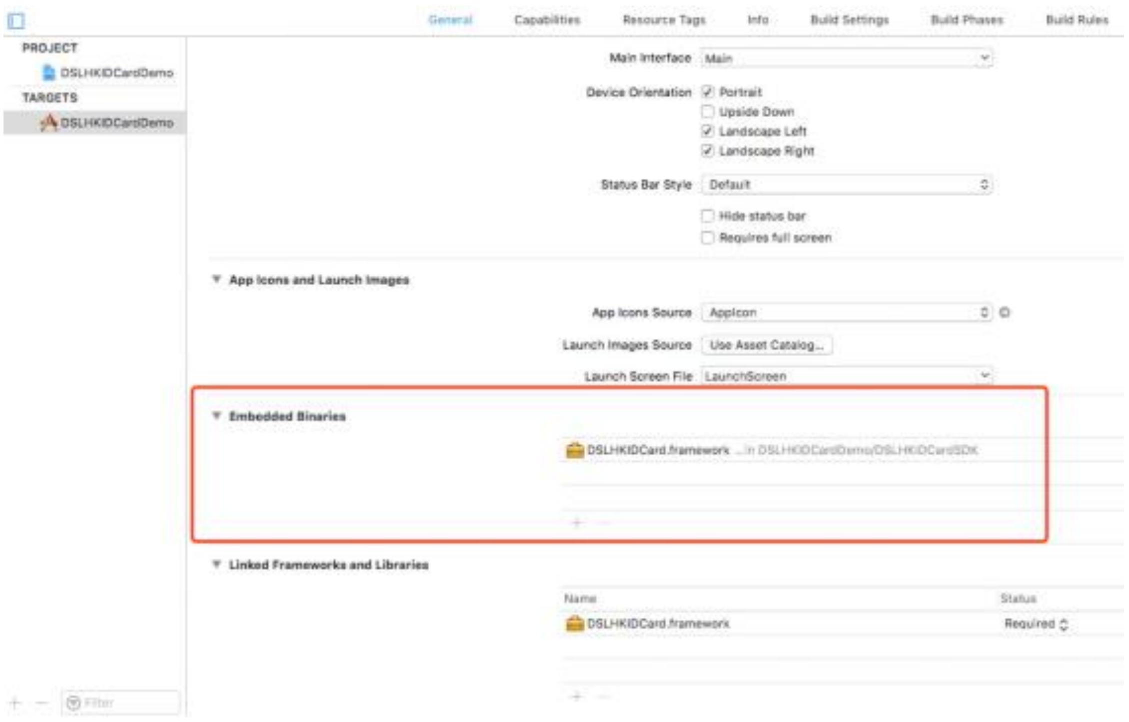
- Import the unzipped DSLHKIDCard.framework by dragging it into your Xcode project:



DSLHKIDCardDemo
DSLHKIDCardDemo
DSLHKIDCardSDK
DSLHKIDCard.framework
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m

2.3. Adding necessarily frameworks

- In General, add DSLHKIDCard.framework into Embedded Binaries:



Main Interface: Main

Device Orientation: [✓ Portrait] Upside Down [✓ Landscape Left] [✓ Landscape Right]

Status Bar Style: Default

Hide status bar

Requires full screen

▼ App Icons and Launch Images

App Icons Source: AppIcon

Launch Images Source: Use Asset Catalog...

Launch Screen File: LaunchScreen

▼ Embedded Binaries

DSLHKIDCard.framework ...in DSLHKIDCardDemo/DSLHKIDCardSDK

▼ Linked Frameworks and Libraries

Name	Status
DSLHKIDCard.framework	Required

3. SDK Description

3.1. Header file description

3.1.1. DSLHKIDCardSDK.h

All SDK external control methods are included in this class

- Interface description:

```
/**
 Start detection
 */
+(void)startDetect;


/**
 Reset status when detection is ended
 Alert: Right now SDK will reset status internally and no need to be called
 from outside
 */
+(void)resetStatus;


/**
 Used when quitting detection screen for destroying the object
 */
+(void)destroyOCR;

/**
 Register callback observer object
 @param observer callback object
 */
+(void)registerObserver:(id<DSLHKIDCardDetectObserverDelegate>)observer;

/**
 Unregister callback observer object
 @param observer callback object
 */
+(void)unregisterObserver:(id<DSLHKIDCardDetectObserverDelegate>)observer;

/**
 Push image data to SDK for detection
 @param imageData imageData Image data
 @param orient Image orientation
 @param lensPosition lensPosition Lens position
 */
+(void)addDetectBufferData:(NSData *_Nonnull)imageData Orient:(int)orient
LensPosition:(float)lensPosition;
/**
 Sets ID card type
 @param isNewIDCard YES: New card； NO: Old card
 */
+(void)setIDCardType:(BOOL)isNewIDCard;

/**
 Retrieve current SDK version number
 @return Current SDK version number
 */
+(NSString *)getVersion;

/**
 Set detection time limit
 @param times Unit:Second; Default is 30s
 */
+ (void)setOverTimes:(int)times;


/**
 Start counting time
 */
+ (void)startOverTime;

/**
 End counting time
 */
+ (void)stopOverTimer;


/**
 Video integrity signature
 @param videoPath Path of saved video file
 @param imgIDCard Static ID card Image (After successful detection, returned
 DSLHKIDCardResult object will contains static ID card image)
 @param videoTimeStamp Concatenated string with video timestamp (example: video
 start time | First motion detected time | Second motion detected time (time
 when flash light is on) | video stop time)
 @return A dictionary object, including signature and custom
 */
+ (NSMutableDictionary* _Nonnull)signatureWithVideoFile:(NSString* _Nonnull)videoPath
TemplateImage:(NSString* _Nonnull) imgIDCard VideoTimeStamp:(NSString* _Nonnull)videoTimeStamp;


/**
 Static image integrity signature
 @param imgIDCard Static ID card Image (After successful detection, returned
 DSLHKIDCardResult object will contains static ID card image)
 @return A dictionary object, including signature and custom
 */
+ (NSMutableDictionary* _Nonnull)signatureWithImg:(NSData* _Nonnull)imgIDCard;


/**
```

```
 Set whether SDK should output log
 Default value: YES for Debug mode; NO for Release mode
 @param enable YES: Export log; NO: Don't export log
 */
+(void)printLog:(BOOL)enable;
```

3.1.2. DSLHKIDCardResult.h

Return object of detection result from detection process

- Interface description:

```
// Is detection success
@property(nonatomic,assign) BOOL success;

/**
   retCode and message possible values:

   retCode    message      Description

   0                       Detection success
   1          first        Returns static image of first motion
      2          exposure     Represents reflection is too strong, need to
                                        restart detection
   3          lost          Represents target (ID card) lost, or during
                                        detection, keep failing to capture some
                                        feature on the ID card and cause detection
                                        failure
      4          overtime     Represents detection time is over. Can set
                                         detection time limit
 */
@property(nonatomic,copy) NSString *message;
@property(nonatomic, assign) int retCode;

// Detection data, reserved and no need to handle
@property(nonatomic,copy)NSString *delta;

// Image data
//1. When success is NO, retCode is 1, first element of the array will be the static image of the ID card, in
other cases, array will be empty
```

```
//2. When success is YES, retCode is 0, first element of the array will be the static image of the ID card, and
last element of the array will be head image from the ID card

@property(nonatomic,copy)NSArray<NSData *> *imageDataArray;
```

3.1.3. DSLHKIDCardNextOperation.h

Return object of detection status during detection process

- Interface description:

```
typedef NS_ENUM(NSInteger,DSLHKIDCardOperationStatus){
  DSLHKIDCardOperation_BEGIN, // Detection starts
      DSLHKIDCardOperation_ORTH,      // Up for 2003 version ID card (Right for
      2018 version ID Card) Start detect rotation
      DSLHKIDCardOperation_LEFTDOWN,  // Down for 2003 version ID card (Left for 2018 version ID
      Card) Start detect rotation
  DSLHKIDCardOperation_COMPLETE, // Detection completes
  DSLHKIDCardOperation_RESET      // Detected card reset to front face
  position
  DSLHKIDCardOperation_FAR,        // Detected distance is too far (Only
  for alert, need to continue detection when distance returns to normal)
 DSLHKIDCardOperation_NEAR,       // Detected distance is too close (Only
  for alert, need to continue detection when distance returns to normal)
 DSLHKIDCardOperation_OVERSPEED,  // Detected rotation is too fast (Only
  for alert, need to continue detection when speed returns to normal)
  DSLHKIDCardOperation_VALID       // Fire when FAR, NEAR, OVERSPEED
   returns to normal, UI should display message to continue detection
};
```

- Result description:

```
// Current detection status
@property(nonatomic,assign)DSLHKIDCardOperationStatus currentStatus;

//  Can ignore if value is nil; Used for better management and support localization, which is placed outside
SDK
@property(nonatomic,copy)NSString *nextOperationHint;
```

3.1.4. DSLHKIDCardDetectObserverDelegate.h

Observer object to receive detection status and result

- Delegate method description:

```
/**
 Must implement this method to receive current detection motion
 @param command DSLHKIDCardNextOperation object, the following are the possible status
   DSLHKIDCardOperation_BEGIN, // Detection starts
       DSLHKIDCardOperation_ORTH,      // Up for 2003 version ID card (Right for
       2018 version ID Card) Start detect rotation
       DSLHKIDCardOperation_LEFTDOWN,  // Down for 2003 version ID card (Left for 2018 version ID
       Card) Start detect rotation
   DSLHKIDCardOperation_COMPLETE, // Detection completes
   DSLHKIDCardOperation_RESET      // Detected card reset to front face
   position
   DSLHKIDCardOperation_FAR,        // Detected distance is too far (Only
   for alert, need to continue detection when distance returns to normal)
  DSLHKIDCardOperation_NEAR,       // Detected distance is too close (Only
   for alert, need to continue detection when distance returns to normal)
   DSLHKIDCardOperation_OVERSPEED,   // Detected rotation is too fast (Only
   for alert, need to continue detection when speed returns to normal)
       DSLHKIDCardOperation_VALID       // Fire when FAR, NEAR, OVERSPEED returns
       to normal, UI should display message to continue detection

*/
-(void)didUpdateOperationCommand:(DSLHKIDCardNextOperation *)command;

/**
 Must implement this method to receive detection result

 @param result Detection result, the following are possible code and message
 retCode and message values:

   retCode    message     Description

   0                      Detection success
   1          first       Returns static image of first motion
       2          exposure     Represents reflection is too strong, need to
                                               restart detection
   3          lost        Represents target (ID card) lost, or during
                                               detection, keep failing to capture some
                                               feature on the ID card and cause detection
```

```
                                   failure
         4         overtime     Represents detection time is over. Can set
                                    detection time limit
    */

    -(void)didDetectResult:(DSLHKIDCardResult *)result;
```

4. SDK Usage

4.1. SDK Initialization

Only need to initialize once.

- Initialize SDK in viewDidLoad method as follows:

```
/**
 Initialize SDK
 */
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view.

    //SDK initialize
    [self initHKIDCardSDK];

   …

}

- (void)initHKIDCardSDK
{
    // Must have, sets type of ID card (2003 version or 2018 version)
    [DSLHKIDCardSDK setIDCardType:!self.recType];
    // Must have, register observer object
    [DSLHKIDCardSDK registerObserver:self];

    // Optional, set detection time limit
    [DSLHKIDCardSDK setOverTimes:30];
 }
```

4.2. SDK Start Detection

Need to call start detection when detection starts in the first time. For example inside viewDidLoad method when entering ViewController for the first time; After detection failed (Lost card, overtime etc) and need to restart detection. Just as the following:

```
- (void)viewDidLoad {
   [super viewDidLoad];
   // Do any additional setup after loading the view.
    ...
   // Start detection
   [self startRecognize];
}

- (void)startRecognize
{
   ...
   [DSLHKIDCardSDK startDetect];
}
```

4.3. SDK Video Frame Data Handling

In system camera's callback method, calls SDK's addDetectBufferData method to pass video frame data for detection, just as the following:

```
#pragma mark -- AVCaptureVideoDataOutputSampleBufferDelegate
- (void)captureOutput:(AVCaptureOutput *)output didOutputSampleBuffer:(CMSampleBufferRef)sampleBuffer
fromConnection:(AVCaptureConnection *)connection {

   if(ALL_Save_Video)
   {
      [self recordingCompressionSession:sampleBuffer];

      if(self.bStopRecording || !self.bStartRec)
      {
         return;
      }
```

```
    }
    else
    {
        if(self.bStopRecording || !self.bStartRec)
        {
            return;
        }

        if(self.bRecStaticImgOk)
        {
            [self recordingCompressionSession:sampleBuffer];
        }
    }

    self.curImgData = [self imageFromSampleBuffer:sampleBuffer];

    dispatch_async(self.videoProcessQueue, ^{

        // Perform data analysis

        static NSInteger nProcessFrame = 0;
        if (nProcessFrame > 0 || !self.bStartRec) {
            return;    //last frame not finished
        }
        nProcessFrame ++;

        // Must call, push video frame data to SDK method (addDetectBufferData)
            // for SDK to analyze detection
        // Alert: Need to set ContinuousAutoFocus mode when initialize, just as
            // follows:
        /**
         // Set ContinuousAutoFocus
         if([_inputDevice isFocusModeSupported: AVCaptureFocusModeContinuousAutoFocus])
         {
         [_inputDevice setFocusMode:AVCaptureFocusModeContinuousAutoFocus];
         }
         */

        [DSLHKIDCardSDK addDetectBufferData:self.curImgData Orient:1
  LensPosition:self.inputDevice.lensPosition];
```

```
        nProcessFrame--;

      });

    }
```

4.4. SDK Return Result Handling

Description:

- Processing message (didUpdateOperationCommand:), current operation motion return message during detection
- Result message (didDetectResult:), return result and details of detection fails (Due to overtime, card lost, too bright, rotation too fast) or detection success. Must be implemented just as the following:
- Message callback:

```
-(void)didUpdateOperationCommand:(DSLHKIDCardNextOperation *)command {

    NSLog(@"operation command = %@, LensPosition: %f",@(command.currentStatus),
self.inputDevice.lensPosition);

    dispatch_main_safe(^{

        [self.nextOperationView setNextOpStatus:command.currentStatus Text:[self
getOpTip:command.currentStatus] IDCardType:self.recType];

        if(command.currentStatus == DSLHKIDCardOperation_ORTH)
        {
            [self setRecStatus:1];
        }
        else if(command.currentStatus == DSLHKIDCardOperation_LEFTDOWN)
        {
            // Detected front ID card, update rect image view to detected state
            self.effectiveRectImgView.image = [UIImage imageNamed:@"rec_new_card_bk"];

            [self setRecStatus:2];
        }
        else if(command.currentStatus == DSLHKIDCardOperation_RESET)
        {
            // Ask user to reset ID card position, update rect image view to
                    // undetected state
            self.effectiveRectImgView.image = [UIImage imageNamed:@"rec_card_un_bk"];

            // Open flash light when receive reset state
            if(self.dRecStartExposureTimeStamp < 0.01)
            {
                self.dRecStartExposureTimeStamp = [[NSDate date] timeIntervalSince1970] * 1000;
            }

            [self setRecStatus:3];
        }
        else if(command.currentStatus != DSLHKIDCardOperation_FAR
                &&
                command.currentStatus != DSLHKIDCardOperation_NEAR
                &&
                command.currentStatus != DSLHKIDCardOperation_OVERSPEED
                &&
                command.currentStatus != DSLHKIDCardOperation_VALID)
        {
            [self setRecStatus:0];
        }
    });
}
```

- Detection result callback:

```
-(void)didDetectResult:(DSLHKIDCardResult *)result
{
    NSLog(@"didDetectResult result = %@, retCode=%i, message=%@, 焦距: %f",@(result.success),
result.retCode, result.message, self.inputDevice.lensPosition);
    dispatch_main_safe((^{
        if (result.retCode == 1) {
            self.bRecStaticImgOk = YES;

            // This two time stamp is the same for now
            self.dRecStartTimeStamp = [[NSDate date] timeIntervalSince1970] * 1000;
            self.dORTHTimeStamp = self.dRecStartTimeStamp;

            // Detected front ID card, update rect image view to detected state
            self.effectiveRectImgView.image = [UIImage imageNamed:@"rec_new_card_bk"];
            // Receive first action's static image
            // Can submit to server if needed
            if (!Is_Local_Save) {
                [self uploadStaticImg:[result.imageDataArray firstObject]];
            }
        } else {
            self.bStopRecording = YES;
            if (!ALL_Save_Video) {
                if ([self.captureSession isRunning]) {
                    [self.captureSession stopRunning];
                }
            }

            if (self.recType == 0) {
                [self turnTorchOn:NO];
            }

            if (result.success == NO) {
                // Detection failed
                [self proccessRecFail:result];
                self.curIDCardResult = nil;
            } else {
                // Detection success and stop video recording
                if (ALL_Save_Video) {
                    if ([self.captureSession isRunning]) {
                        [self.captureSession stopRunning];
                    }
                }

                self.dRecEndTimeStamp = [[NSDate date] timeIntervalSince1970] * 1000;
                self.curIDCardResult = result;
                NSLog(@"delta string = %@",(result.delta));

                // Detection success and stop video recording
                if (ALL_Save_Video) {
                    [self finishVideoWriter];
                }
            }
            if (!ALL_Save_Video) {
              [self finishVideoWriter];
            }

            if (result.success) {
              // save image
              if (Is_Save_Video_Picture) {
                for (int i = 0; i < (int)[result.imageDataArray count]; ++i) {
                  UIImage *image = [UIImage imageWithData:[result.imageDataArray objectAtIndex:i]];
                  UIImageWriteToSavedPhotosAlbum(image, self, nil, nil);
                }
              } else if(Is_Local_Save) {
                 UIImage *image = [UIImage imageWithData:[result.imageDataArray firstObject]];
                __block ALAssetsLibrary *lib = [[ALAssetsLibrary alloc] init];
                [lib writeImageToSavedPhotosAlbum:image.CGImage metadata:nil completionBlock:^(NSURL
      *assetURL,NSError *error) {
                  ALAssetsLibraryAssetForURLResultBlock resultblock = ^(ALAsset *imageAsset) {
                    ALAssetRepresentation *imageRep = [imageAsset defaultRepresentation];
                    self.imgSaveAlbumFileName = [imageRep filename];
                    if (![self.imgSaveAlbumFileName isEqualToString:@""] && ![self.videoSaveAlbumFileName
      isEqualToString:@""]) {
                      [self saveRequestData];
                      self.imgSaveAlbumFileName = @"";
                    }
                    NSLog(@"[imageRep filename ] : %@", [imageRep filename]);
                  };
                  ALAssetsLibrary* assetslibrary = [[ALAssetsLibrary alloc] init];
                  [assetslibrary assetForURL:assetURL resultBlock:resultblock failureBlock:nil];

                  NSLog(@"save image assetURL = %@, error = %@", assetURL, error);
                  lib = nil;
                }];
              }
            }
          }
        }));

      }
```

4.5. SDK Destruction

When exiting detection screen or the screen disappears, destroyOCR method needs to be called to release internal resource used by SDK

```
-(void)viewWillDisappear:(BOOL)animated{
   [super viewWillDisappear:animated];


       …


   if ([self.captureSession isRunning])
   {
      [self.captureSession stopRunning];
   }
   // Stop detection
   self.bStartRec = NO;
   // Stop recording video
   [self stopMovieRecorder];

   // Must have, when detection screen disappear, calling destroyOCR method is needed to release internal
   // resources used by the SDK
   [DSLHKIDCardSDK destroyOCR];


       …


}
```

4.6. SDK Result Observer Unregister

When ViewController is deallocated (calls dealloc method), observer object needs to be unregistered just as the following:

```
- (void)dealloc
{
   NSLog(@"%@ dealloc!",NSStringFromClass([self class]));
   // Must have, unregister observer object
   [DSLHKIDCardSDK unregisterObserver:self];

}
```

5. Sample Project Description

5.1. Base class: DSLHKBaseIDCardViewController

Description:

- Video capture and UI of motion instruction animation shown during detection process are implemented inside DSLHKBaseIDCardViewController class, exposed necessary methods and properties, and supply video frame data to SDK in - (void)captureOutput:(AVCaptureOutput *)output didOutputSampleBuffer :(CMSampleBufferRef)sampleBuffer fromConnection:(AVCaptureConnection *)connectionby calling SDK's addDetectBufferData method, for SDK to analyse and perform detection
- Third party needs to inherit this class to implement following functions:
    - Calling related SDK method and callback handling (Except addDetectBufferData method)
    - Making request to server
    - Display request result from server
    - Personalized UI

5.2. Configuration file

In DSLHKIDCardConfig.h, can set the following:

- Connecting server
- Display instruction of holding ID card
- Is_Save_Video_Picture(Whether saving video and images)
- ALL_Save_Video(Whether saving video for failed results)
- Is_Local_Save(Save video and image inside the device, no need to submit to server. Used for gathering result in no network environment)
- Server address depends on actual development

5.3. Localization

Description:

- Support localization base on iOS's development standard, placed inside Localizable.strings file
- Currently there are only Traditional Chinese language (Hong Kong) (Simplified Chinese, English file still contains Traditional Chinese language)，Actual text displayed can be edited by users
- The following are the current text to be localized:

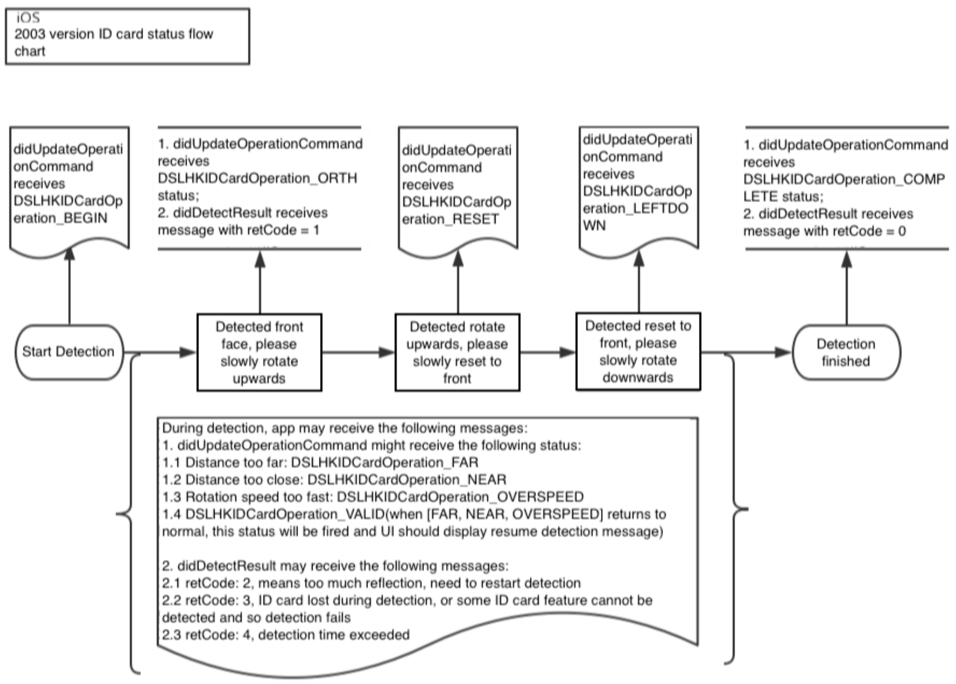
```
// Main page related
```

```
“main_vc_title1” = “手機IP地址 請稍等”;
“main_vc_title2” = “進入掃描”;
“main_vc_title3” = “重播”;
“main_vc_title4” = “設定”;
“main_vc_title5” = “新版證件”;
“main_vc_title6” = “舊版證件”;
“main_vc_title7” = “退出”;
“main_vc_title8” = “跳過”;

// Decide upload page related
"upload_video_tip1" = "上傳視頻可以讓檢測更加準確！\n我們將保障您的個人信息安全。";
"upload_video_tip2" = "沒問題，同意上傳";
"upload_video_tip3" = "殘忍拒絕";
"upload_video_tip4" = "不再提示";
"upload_video_tip5" = "確定";

// Setting page related
"setting_view_title1" = "轉速檢測";
"setting_view_title2" = "光亮檢測";
"setting_view_title3" = "播放視頻";

// Detection process related
"idcard_operation_title1"  = "請將身份證正面置於掃描框內";
"idcard_operation_title2"  = "請緩慢向右翻轉";
"idcard_operation_title3"  = "請緩慢向左翻轉";
"idcard_operation_title4"  = "請緩慢向上翻轉";
"idcard_operation_title5"  = "請緩慢向下翻轉";
"idcard_operation_title6"  = "光線過亮，請調整";
"idcard_operation_title7"  = "識別完成";
"idcard_operation_title8"  = "向右翻識別成功，請把證件復位";
"idcard_operation_title9"  = "上翻識別成功，請緩慢把證件復位";
"idcard_operation_title10"  = "證件捕捉成功，開始識別";
"idcard_operation_title11"  = "超時，請重新開始";
"idcard_operation_title12"  = "證件復位成功";
"idcard_operation_title13"  = "距離過遠，請調整";
"idcard_operation_title14"  = "距離過近，請調整";
"idcard_operation_title15"  = "轉動速度過快，請慢點";
```

```
// Holding ID Card related
"hand_idcard_tip1"  = "手持證件，將證件置於掃描框內\n按照提示完成翻轉動作";
"hand_idcard_tip2"  = "證件識別";
"hand_idcard_tip3"  = "識別規範";
"hand_idcard_tip4"  = "開始掃描";

"result_fail_title1"  = "再試一次";

// Wait for result related

"loading_rec_title1"  = "識別中，請稍等";
```

6. ID Card Detection Flow Chart

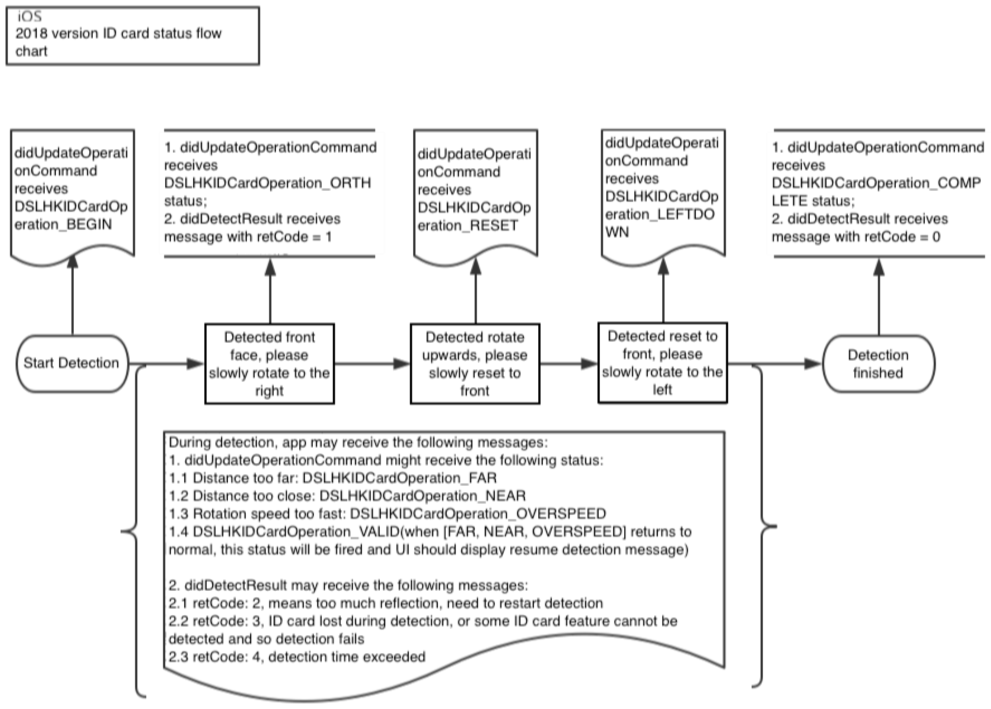
6.1. 2003 Version


iOS
2003 version ID card status flow chart
didUpdateOperationCommand receives DSLHKIDCardOperation_BEGIN
1. didUpdateOperationCommand receives DSLHKIDCardOperation_ORTH status;
2. didDetectResult receives message with retCode = 1
didUpdateOperationCommand receives DSLHKIDCardOperation_RESET
didUpdateOperationCommand receives DSLHKIDCardOperation_LEFTDOWN
1. didUpdateOperationCommand receives DSLHKIDCardOperation_COMPLETE status;
2. didDetectResult receives message with retCode = 0
Start Detection
Detected front face, please slowly rotate upwards
Detected rotate upwards, please slowly reset to front
Detected reset to front, please slowly rotate downwards
Detection finished

During detection, app may receive the following messages:
1. didUpdateOperationCommand might receive the following status:
1.1 Distance too far: DSLHKIDCardOperation_FAR
1.2 Distance too close: DSLHKIDCardOperation_NEAR
1.3 Rotation speed too fast: DSLHKIDCardOperation_OVERSPEED
1.4 DSLHKIDCardOperation_VALID(when [FAR, NEAR, OVERSPEED] returns to normal, this status will be fired and UI should display resume detection message)

2. didDetectResult may receive the following messages:
2.1 retCode: 2, means too much reflection, need to restart detection
2.2 retCode: 3, ID card lost during detection, or some ID card feature cannot be detected and so detection fails
2.3 retCode: 4, detection time exceeded

6.2. 2018 Version


iOS
2018 version ID card status flow chart
didUpdateOperationCommand receives DSLHKIDCardOperation_BEGIN
1. didUpdateOperationCommand receives DSLHKIDCardOperation_ORTH status;
2. didDetectResult receives message with retCode = 1
didUpdateOperationCommand receives DSLHKIDCardOperation_RESET
didUpdateOperationCommand receives DSLHKIDCardOperation_LEFTDOWN
1. didUpdateOperationCommand receives DSLHKIDCardOperation_COMPLETE status;
2. didDetectResult receives message with retCode = 0
Start Detection
Detected front face, please slowly rotate to the right
Detected rotate upwards, please slowly reset to front
Detected reset to front, please slowly rotate to the left
Detection finished

During detection, app may receive the following messages:
1. didUpdateOperationCommand might receive the following status:
1.1 Distance too far: DSLHKIDCardOperation_FAR
1.2 Distance too close: DSLHKIDCardOperation_NEAR
1.3 Rotation speed too fast: DSLHKIDCardOperation_OVERSPEED
1.4 DSLHKIDCardOperation_VALID(when [FAR, NEAR, OVERSPEED] returns to normal, this status will be fired and UI should display resume detection message)

2. didDetectResult may receive the following messages:
2.1 retCode: 2, means too much reflection, need to restart detection
2.2 retCode: 3, ID card lost during detection, or some ID card feature cannot be detected and so detection fails
2.3 retCode: 4, detection time exceeded