

iOS GlobalIQ SDK (v1.2.0) User Guide

Overview

`AAIGlobalIQASDK` contains two modules, the core module `AAIGlobalIQASDK.xcframework`, the network module `AAINetwork.xcframework`.

The actual total size of the SDK is about 3.6MB(armv7&arm64, disable bitcode).

- **AAIGlobalIQASDK.xcframework**

This is a core module.

The actual total size of the this module is about 2.5MB(armv7&arm64, disable bitcode). See [FAQ](#) for how to strip the bitcode.

- **AAINetwork**

A base network library that AAI SDKs rely on.

The actual total size of the this module is about 1.1MB(armv7&arm64, disable bitcode).

Migration Guides

1. When migrating from 1.0.x to 1.1.x or higher, you need to modify your `Podfile` to reintegrate the SDK(including SDK dependency `AAINetwork.xcframework`).

Run demo project

1. Download the [AAIGlobalIQASDK](#) and extract it, then navigate to the directory of `AAIGlobalIQASDKObjCDemo` or `AAIGlobalIQASDKSwiftDemo` project and install the dependencies:

```
pod install
```

2. Open xcworkspace file in Xcode.
3. Specify your `license`, and configure `region` and `cardType` and `cardSide`:
4. Run.

Installation

CocoaPods

1. Specify the SDK name and url in the podfile:

```
pod 'AAINetwork', :http => 'https://prod-guardian-cv.oss-ap-southeast-5.aliyuncs.com/sdk/iOS-libraries/AAINetwork/AAINetwork-V1.0.0.tar.bz2', type: :tbz
pod 'AAIGlobalIQ', :http => 'https://prod-guardian-cv.oss-ap-southeast-5.aliyuncs.com/sdk/iOS-global-IQA/iOS-GlobalIQ-SDK-V1.2.0.tar.bz2', type: :tbz
```

2. Install the dependencies in your project:

```
pod install
```

3. Add camera usage description in Info.plist as bellow. Ignore this step if you have added those.

```
<key>NSCameraUsageDescription</key>
<string>Use the camera to detect the card</string>
```

Manually

1. Download the [AAIGlobalIQASDK](#), then extract it and add `AAIGlobalIQASDK.xcframework` to your project and set embed as "Embed & Sign".
2. Choose "TARGETS -> General" add the following system libraries and frameworks in the `Frameworks, Libraries, and Embedded Content` section:
 - `libc++.tbd`
 - `libz.tbd`
 - `AVFoundation.framework`
 - `CoreGraphics.framework`
 - `MediaPlayer.framework`
 - `SystemConfiguration.framework`
 - `Accelerate.framework`
3. Download the [AAINetwork.xcframework](#) and extract it, then copy `AAINetwork.xcframework` to your project and add it in section "TARGETS -> General -> Frameworks,Libraries,and Embedded Content", then set Embed as "Embed & Sign".
4. Add camera usage description in `Info.plist` as bellow. Ignore this step if you have added those.

```
<key>NSCameraUsageDescription</key>
<string>Use the camera to detect the card</string>
```

Usage

1. Import SDK and initialize with `region` and `cardType` and `cardSide`:

```
@import AAIGlobalIQASDK;

AAIGlobalIQAConfig *config = [AAIGlobalIQAConfig initWithRegion:@"ID"
cardType:AAIIQACardTypeIDCard cardSide:AAIIQACardSideFront];
config.delegate = self;
// If you set YES, then the method `iqaOnDetectionComplete:` is called after the
IQA page is dismissed.
config.callbackAfterPageDismissed = YES;

/*
// Specify the operating mode of the SDK.
```

```

/// The SDK can have different operating modes, such as real-time scanning mode,
photo mode, default mode(Scanning + Photo). Default is
'AAIIQAOperatingModeDefault'.
// e.g.
config.operatingMode = AAIQAOperatingModeScanning;
*/

[AAIGlobalIQASDK initWithConfig:config];

```

2. User binding (Optional, but highly recommended).

You can use this method to pass us your unique user ID, and we will establish a mapping relationship based on that ID. It's easy to track logs with us in case of problems.

```
[AAIGlobalIQASDK bindUser: @"your-user-id"];
```

3. Configure SDK license and show SDK page:

```

// Note that the "setLicenseAndCheck:" method MUST BE CALLED before calling
"startWithRootVC:".
NSString *result = [AAIGlobalIQASDK setLicenseAndCheck:@"your-license-content"];
if ([result isEqualToString:@"SUCCESS"]) {
    [AAIGlobalIQASDK startWithRootVC:self];
} else {
    NSLog(@"%@", result);
}

```

4. Get detection results.

```

- (void)iqaOnDetectionComplete:(AAIGlobalIQAResult *)result
{
    if (result.success) {
        UIImage *cardImg = result.cardImg;
        NSString *idvid = result.IDVID;
        NSString *transactionId = result.transactionId;
        NSString *pictureType = result.pictureType;

        if (result.ocrResult) {
            NSDictionary *ocrResult = result.ocrResult;
        }

        if (result.idForgeryResult) {
            NSDictionary *idForgeryResult = result.idForgeryResult;
        }
    } else {
        NSString *errorCode = result.errorCode;
        if (result.errorMsg) {
            NSString *errorMsg = result.errorMsg;
        }
    }
}

```

```

    }

    if (result.transactionId) {
        NSString *transactionId = result.transactionId;
    }
}
}

```

Customizable UI

1. UI Customization.

Currently, the SDK only supports limited UI customization, see demo project for more details.

```

AAIGlobalIQAConfig *config = [AAIGlobalIQAConfig initWithRegion:@"ID"
cardType:AAIIQACardTypeIDCard cardSide:AAIIQACardSideFront];

// Whether to callback the AAIGlobalIQADelegate method `iqaOnDetectionComplete:`
// after the view controller is dismissed.
// Default is NO, which means that the `iqaOnDetectionComplete:` method will be
// called before the IQA page is closed,
// If you set YES, then the method `iqaOnDetectionComplete:` is called after the
// IQA page is dismissed.
config.callbackAfterPageDismissed = YES;

// Specify the operating mode of the SDK.
/// The SDK can have different operating modes, such as real-time scanning mode,
// photo mode, default mode(Scanning + Photo). Default is
// 'AAIIQAOperatingModeDefault'.
// e.g.
config.operatingMode = AAIIQAOperatingModeScanning;

// e.g.
config.uiConfig.flipCameraBtnVisible = NO;
config.uiConfig.lightBtnVisible = NO;

// Whether to show the timer label in the upper right corner of the view. Default
// is YES.
config.uiConfig.timerLabelVisible = NO;

config.uiConfig.navBarTitleTextColor = UIColor.redColor;
config.uiConfig.navBarBackgroundColor = UIColor.whiteColor;
config.uiConfig.statusBarStyle = UIStatusBarStyleDefault;
config.uiConfig.tipIconVisible = NO;
config.uiConfig.pageBackgroundColor = UIColor.redColor;
config.uiConfig.viewfinderColor = UIColor.greenColor;
config.uiConfig.primaryTextColor = UIColor.yellowColor;
config.uiConfig.toolButtonThemeColor = UIColor.blueColor;
config.uiConfig.orientation = AAIIQAOrientationPortrait;

```

```

config.uiConfig.takePhotoTipTextColor = UIColor.whiteColor;
config.uiConfig.takePhotoTipBackgroundColor = UIColor.blueColor;
config.uiConfig.previewPhotoTipBackgroundColor = UIColor.blueColor;
config.uiConfig.previewPhotoTipTextColor = UIColor.whiteColor;
config.detectionTimeoutInterval = 20;
config.returnEmptyOfOCR = NO;
config.languageLprojName = @"id";

// Set the left and right margin on x-axis of cameraView when device in portrait
mode.
config.uiConfig.cameraViewMarginLRInPortraitMode = @(12);

// The duration of the alert view displayed when SDK are about to enter the photo
mode,
// the default is 0s, which means the alert view will not be displayed.
config.takePhotoAlertViewTimeoutInterval = 0;

// If you want to add custom UI on the SDK page, just register the custom class to
the SDK, see demo project for more details.
/*
// You can customize the scan page UI by inheriting `AAIIQAScanController` class,
then register your custom class to the sdk. See demo project for more detail.
[config registerClass:[CustomScanController class]
forModule:AAIIQAModuleTypeScanController];
*/

/*
// You can customize the preview the photos taken page UI by inheriting
`AAIIQAPreviewImgController` class,
// then register your custom class to the sdk. See demo project for more detail.
[config registerClass:[CustomPreviewImgController class]
forModule:AAIIQAModuleTypePreviewImgController];
*/

/*
// You can customize the camera overlay view UI of `AAIIQAScanController` page by
inheriting `AAIIQAOverlayView` class,
// then register your custom class to the sdk. See demo project for more detail.
[config registerClass:[CustomOverlayView class]
forModule:AAIIQAModuleTypeOverlayViewOfScanVC];
*/

/*
// You can customize the camera overlay view UI of `AAIIQAPreviewImgController`
page by inheriting `AAIIQAOverlayView` class,
// then register your custom class to the sdk. See demo project for more detail.
[config registerClass:[CustomOverlayView class]
forModule:AAIIQAModuleTypeOverlayViewOfPreviewImgVC];
*/

```

```

/* imageName list:
    advance_iqa_scan: An image to show the scan animation.
    advance_iqa_tip_capture: The icon in the countdown pop-up view after the scan
times out.
    iqa_back: Icon of navigation bar back button.
    iqa_land_confirm: Confirm button icon in landscape take photo mode.
    iqa_land_retake: Retake button icon in landscape take photo mode.
    iqa_light_off: Icon of flashlight off button.
    iqa_light_on: Icon of flashlight on button.
    iqa_scan_processing: The icon displayed at the bottom of the vertical screen
(local string key is: "hold_phone").
    iqa_scan_successfully: The icon displayed at the bottom of the vertical screen
(local string key is: "iqa_scan_successfully").
    iqa_scan_warning: The icon displayed at the bottom of the vertical screen (local
string key is: "iqa_card_poor_quality").
    iqa_take_photo_tip: The tip icon displayed at the bottom of the viewfinder view
(local string key is: "iqa_take_photo_tips").
    iqa_take_photo: Icon of take photo button.
    iqa_transform_camera: Icon of switch camera button.
*/
// You can implement this block to customize the images used in the SDK.
config.uiConfig.loadImage = ^UIImage * _Nonnull(NSString * _Nonnull imgName,
UIImage * _Nonnull img) {
    if ([imgName isEqualToString:@"iqa_scan_successfully"]) {
        return [UIImage imageNamed:@"ok"];
    }
    return img;
};

/*
// You can implement this block to customize the localization string used in the
SDK.
// For all available keys, see
"AAIGlobalIQASDK.framework/AAIIQALanguageString.bundle/en.lproj/Localizable.strings
"
*/
config.uiConfig.loadLocalizedString = ^NSString * _Nonnull(NSString * _Nonnull key,
NSString * _Nonnull value, NSString * _Nonnull language) {
    if ([key isEqualToString:@"iqa_top_desc"]) {
        if ([language isEqualToString:@"en"]) {
            return @"My test string";
        }
    }

    return value;
};

```

2. Resources File Customization.

Currently, this SDK supports English (en), Simplified Chinese (zh-Hans), Indonesian (id).

The SDK will use the corresponding resource file according to the current language of the device. If this device language is not in the above languages, the SDK will use English by default.

If you want to support only a fixed language, you can set the `languageIprojName` property of `AAIGlobalIQAConfig` to specify the language you want the SDK to use.

The resource files corresponding to these languages are in the following bundles:

Name	Description	Customizable description
AAIQLanguageString	Muti-language bundle	Can be modified directly
AAIQAAudio.bundle	Audio bundle	Use your audio file to replace it. NOTE: the audio file name cannot be changed

3. Error code list.

Name	Description
USER_GIVE_UP	User tapped the back button
DEVICE_NOT_SUPPORT	This device is not supported
CAMERA_PERMISSION_DENIED	Permission to access the camera is not authorized
NETWORK_REQUEST_FAILED	Network request failed
CAMERA_OPEN_FAILED	Failed to open camera
MODEL_ERROR	Load model failed
SCAN_TIMEOUT	Scan timeout. Note this code appears only when you set the <code>operatingMode</code> of <code>AAIGlobalIQAConfig</code> to "AAIQAOperatingModeScanning".
Other error code	See document

FAQ

1. The earliest supported iOS version is iOS9, and support bitcode.

If you want to minimize the SDK's size, you can remove SDK's bitcode by using the following method:

```

# Strip bitcode (ios-arm64_armv7)
xcrun bitcode_strip AAIGlobalIQASDK.xcframework/ios-
arm64_armv7/AAIGlobalIQASDK.framework/AAIGlobalIQASDK -r -o
AAIGlobalIQASDK.xcframework/ios-
arm64_armv7/AAIGlobalIQASDK.framework/AAIGlobalIQASDK2
# Replace old SDK file
mv -f AAIGlobalIQASDK.xcframework/ios-
arm64_armv7/AAIGlobalIQASDK.framework/AAIGlobalIQASDK2
AAIGlobalIQASDK.xcframework/ios-
arm64_armv7/AAIGlobalIQASDK.framework/AAIGlobalIQASDK

# Strip bitcode (ios-arm64_i386_x86_64-simulator)
xcrun bitcode_strip AAIGlobalIQASDK.xcframework/ios-arm64_i386_x86_64-
simulator/AAIGlobalIQASDK.framework/AAIGlobalIQASDK -r -o
AAIGlobalIQASDK.xcframework/ios-arm64_i386_x86_64-
simulator/AAIGlobalIQASDK.framework/AAIGlobalIQASDK2
# Replace old SDK file
mv -f AAIGlobalIQASDK.xcframework/ios-arm64_i386_x86_64-
simulator/AAIGlobalIQASDK.framework/AAIGlobalIQASDK2
AAIGlobalIQASDK.xcframework/ios-arm64_i386_x86_64-
simulator/AAIGlobalIQASDK.framework/AAIGlobalIQASDK

```

After striped the SDK's bitcode, in your Xcode project, you must ensure that "ENABLE_BITCODE" is set to "NO", then clean build cache and rebuild your project.

Change logs and release history

See [iOS IQA SDK release history](#)