Kofax Kony FFI

Atalasoft MobileImage Kony FFI Developer's Guide

Version: 3.2.0

Date: 2017-05-23

Atalasoft
A Kofax Company

©2013- 2017 Atalasoft, 116 Pleasant St, Suite 321, Easthampton, MA 01027, U.S.A. All rights reserved.
Use is subject to license terms.

Third-party software is copyrighted and licensed from Atalosoft's suppliers.

THIS SOFTWARE CONTAINS CONFIDENTIAL INFORMATION AND TRADE SECRETS OF ATALASOFT, INC. USE, DISCLOSURE OR REPRODUCTION IS PROHIBITED WITHOUT THE PRIOR EXPRESS WRITTEN PERMISSION OF KOFAX.

Atalasoft, the Atalasoft logo, and the Atalasoft product names stated herein are trademarks or registered trademarks of Atalasoft, Inc. in the U.S. and other countries. All other trademarks are the trademarks or registered trademarks of their respective owners. U.S. Government Rights Commercial software. Government users are subject to the Kofax. standard license agreement and applicable provisions of the FAR and its supplements.

You agree that you do not intend to and will not, directly or indirectly, export or transmit the Software or related documentation and technical data to any country to which such export or transmission is restricted by any applicable U.S. regulation or statute, without the prior written consent, if required, of the Bureau of Export Administration of the U.S. Department of Commerce, or such other governmental entity as may have jurisdiction over such export or transmission. You represent and warrant that you are not located in, under the control of, or a national or resident of any such country.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Table of Contents

Preface

This guide includes the information you need to successfully integrate Atalasoft MobileImage Kony FFI into your mobile project.

Getting help for Kofax products

Kofax regularly updates the Kofax Support site with the latest information about Kofax products.

To access some resources, you must have a valid Support Agreement with an authorized Kofax Reseller/Partner or with Kofax directly.

Use the tools that Kofax provides for researching and identifying issues. For example, use the Kofax Support site to search for answers about messages, keywords, and product issues. To access the Kofax Support page, go to www.kofax.com/support.

The Kofax Support page provides:

- Product information and release news

  Click a product family, select a product, and select a version number.
- Downloadable product documentation

  Click a product family, select a product, and click **Documentation**.
- Access to product knowledge bases

  Click **Knowledge Base**.
- Access to the Kofax Customer Portal (for eligible customers)

  Click **Account Management** and log in.

  To optimize your use of the portal, go to the Kofax Customer Portal login page and click the link to open the *Guide to the Kofax Support Portal*. This guide describes how to access the support site, what to do before contacting the support team, how to open a new case or view an open case, and what information to collect before opening a case.
- Access to support tools

  Click **Tools** and select the tool to use.
- Information about the support commitment for Kofax products

  Click **Support Details** and select **Kofax Support Commitment**.

Use these tools to find answers to questions that you have, to learn about new functionality, and to research possible solutions to current issues.

Chapter 1

Overview

The *Atalasoft MobileImage Kony FFI Developer's Guide* provides an overview of developing Atalasoft MobileImage Kony FFI implementations. This document includes information on creating a sample app that can be used to demonstrate the functionality of the SDK.

Note Atalasoft MobileImage Kony FFI will not be supported from version of the Mobile SDK release.

Recommendations

- Use the getImageBase64 API for getting image data to display in any image widget.
- Use the getImageRawBytes API to send/upload an image to any Web server.

Known limitations

- Kony applications do not support 64-bit applications.
- For Android, the response object from any method needs to be formatted as a JSON object in JavaScript, since the response from the Android FFI is returned as an Object (String).
- For Android, the application state is not maintained. This is a Kony limitation.
  - Whenever an app is launched by clicking the app icon, by default the launcher activity is launched.
  - This launcher activity is created dynamically while the application is being built.
- For iOS 9.0 and above:
  - Please set "Enable Bitcode" to NO.
  - If there are any existing `dylib` files, remove them and add tbd equivalents.
  - To the `info.plist` add a dictionary named "App Transport Security Settings" and add a key value pair of "Allow Arbitrary Loads" and "YES".
  - For Android 6.0 and above, please upgrade the Kony Studio to 6.5.2 GA version and build the sample app.
  - Kony studio upgrade plugin: `http://download.kony.com/studio/65/hotfixsite.xml`.
  - Please upgrade the `libkonyjsvm.so` file with the latest which is generated by Kony studio 6.5.2 GA version
- `libkonyjsvm.so` will cause an unauthorized access to `libssl.so` and `libcrypto.so` warning on Android 7.X devices, when built against Kony Visualizer Enterprize 7.1.

Chapter 2

The Foreign Function Interface

A Foreign Function Interface (FFI) provides a mechanism for a program written in one language to call routines or use services in some other language. The Atalasoft MobileImage Kony FFI has four classes: KofaxUtilities, KofaxUIControls, KofaxImageArray, and KofaxImageProcessor. The following sections describe these classes and their associated APIs.

The Utilities class

This class handles all the license and error-related APIs.

Licenses

An instance of this class contains the properties and methods used to set license usage. Once you set the license, the other library API methods work as designed. The license remains valid while the app is still running. Setting the license is required to unlock the library's features.

Native Android license

Package name: `com.kofax.utilities`

Class Name: License

Native iOS license

Class Name: KofaxLicense

JavaScript constructor

```
var License = Kofax.Utilities.License();
```

License API (FFI mappings)

getSDKVersion()

Parameters

None

Return value

JSON object

Description
Gets the version of the SDK being used.

Android native mapping
ava.lang.Object getSDKVersion ()

iOS native mapping
-(NSDictionary*) getSDKVersion

setMobileSDKLicense()

Parameters
License String

Return value
JSON object

Description
Method to set a valid license obtained from Kofax.

Android native mapping
ava.lang.Object setMobileSDKLicense (java.lang.String license)

iOS native mapping
-(NSDictionary*) setMobileSDKLicense:(NSString*) license

getDaysRemaining ()

Parameters
None

Return value
JSON object

Description
Gets the number of days remaining before the license expires.

Android native mapping
java.lang.Object getDaysRemaining ( )

iOS native mapping
-(NSDictionary*) getDaysRemaining

Sample code

This method returns a response dictionary that contains the status of the action performed.

```
//Creates an object of class 'License'
var LicenseObject = new Kofax.Utilities.License();
//Invokes method 'setMobileSDKLicense' on the object
var returnedValue = LicenseObject.setMobileSDKLicense(
  /**String*/ license);
```

The UI Controls class

This class contains the Camera Control, Bar Code Control, Credit Card Control, and the Image Review Edit Control classes.

The Camera Control class

This class handles all the capture-related APIs. Use this class to invoke the appropriate capture experience by passing the type of experience in the constructor. The JSON definition for the camera includes the `captureexperience` key, which specifies the type of experience.

There are two types of experiences: the camera JSON definition "0-CheckCapture Experience" and the "1-Uniform Guidance Experience" JavaScript Constructor:

Native Android

Package name: `com.kofax.capture`

Class Name: Camera

Native iOS

Class name: `KofaxCaptureControl`

Camera API

show(Options)

Parameters

JSON

Return value

None

Description

Adds the Camera view depending on the type passed. The default type is Uniform Guidance.

Android native mapping

show

iOS native mapping
showCamera

set(Options)

Parameters
JSON

Return value
JSON Object

Description
The Camera Options have both a type and set of options. Options for the default camera would be `{type: default, Options:{levelindicator:false}}`.

Android native mapping
setOptions

iOS native mapping
setOptions

get(Options)

Parameters
JSON

Return value
JSON Object

Description
Retrieves the options on the current camera depending on the type. For example, if the camera currently shows animated guides, the result will contain the type and options related to that status.

Android native mapping
getOptions

iOS native mapping
getOptions

takePicture()

Parameters
NONE

Return value
NONE

Description

Takes picture on the active camera and triggers the callback specified in `addImageCapturedListener()` with the captured image.

Android native mapping

takePicture

iOS native mapping

takePicture

takePictureContinually()

Parameters

NONE

Return value

NONE

Description

Takes pictures continually on the active Camera, and triggers a call back with the captured images.

Android native mapping

takePictureContinous

iOS native mapping

takePictureContinually

addFocusListener()

Parameters

focusListener - function

Return value

NONE

Description

Function called on focus events.

Android native mapping

addFocusListener

iOS native mapping

addFocusListener

addStabilityListener()

Parameters

stabilityListener - function

Return value

NONE

Description

Function called on stability events.

Android native mapping

addStabilityListener

iOS native mapping

addStabilityListener

addPagedetectionListener()

Parameters

pageDetectionListener - function

Return value

NONE

Description

Function called on page detection events.

Android native mapping

addPageDetectionListener

iOS native mapping

addPageDetectionListener

addLevelnessListener()

Parameters

levelnessListener - function

Return value

NONE

Description

Function called on levelness events.

Android native mapping

addLevelnessListener:

iOS native mapping
addLevelnessListener:

addImageCapturedListener()
Parameters
imageCapturedListener - function

Return value
NONE

Description
Function called on image captured event.

Android native mapping
addImageCapturedListener

iOS native mapping
addImageCapturedListener

addCameraInitialisationListener()
Parameters
Function cameraInitialisationCallback

Return value
NONE

Description
Function which gets called when the camera is added to the view and initialized.

Android native mapping
addCameraInitialisationListener

iOS native mapping
addCameraInitialisationListener

addCameraInitialisationFailedListener()
Parameters
Function cameraInitialisationFailedCallback

Return value
NONE

Description
Function which gets called when the camera initialization failed when it is added to the view.

Android native mapping
addCameraInitialisationFailedListener

iOS native mapping
addCameraInitialisationFailedListener

removeFocusListener()

Parameters
NONE

Return value
NONE

Description
Removes the listener added for focus events.

Android native mapping
removeFocusListener

iOS native mapping
removeFocusListener

removeStabilityListener()

Parameters
NONE

Return value
NONE

Description
Removes the listener added for stability events.

Android native mapping
removeStabilityListener

iOS native mapping
removeStabilityListener

removePagedetectionListener()

Parameters
NONE

Return value
NONE

Description
Removes the listener added for Page detection events.

Android native mapping
removePageDetectListener

iOS native mapping
removePageDetectListener

removeLevelnessListener()
Parameters
NONE

Return value
NONE

Description
Removes the listener added for level events.

Android native mapping
removeLevelnessListener

iOS native mapping
removeLevelnessListener

removeImageCapturedListener()
Parameters
NONE

Return value
NONE

Description
Removes the listener added for image captured events.

Android native mapping
removeImageCapturedListener

iOS native mapping
removeImageCapturedListener

removeCameraInitialisationListener()
Parameters
NONE

Return value
NONE

Description
Removes the listener for camera initialization event.

Android native mapping
removeCameraInitialisationListener

iOS native mapping
removeCameraInitialisationListener

Sample code

```
 //Create an object of Camera class.

      camera = new Kofax.UI.Camera(cameraOptions.cameraoptions.captureexperience);

//Launch the camera with set of options Here cameraOptions is the JSON definition for the
 camera.
  camera.

      camera.show(cameraOptions);

//Add camera initialization listener.This call back is received when camera is
  initialized.Any actions to be performed after the initialization can be executed
  in this listener viz. The TakePicture method can be called as the call back is received.

     camera.addCameraInitialisationListener(cameraInitialized);

//Add image captured listener.This call back is received when image is captured.
  Any actions to be performed after the capture can be executed in this listener.

   camera.addImageCapturedListener(imagecaptured);
```

The Bar Code Control class

This class handles all the bar code related APIs.

Native Android
Package name: `com.kofax.capture`
Class Name: Barcode

Native iOS
Class Name: KofaxBarcodeControl

JavaScript Constructor

```
var barcode = new Kofax.UI.BarcodeControl();
```

This view provides visual guidance to the end user to maximize the chances of correctly decoding a bar code. Once the view has been asked to read a bar code, it will search continuously until one is found.

Bar Code Control API

show(Options)

Parameters

Options -- JSON Object

Return Type

NONE

Description

Shows bar code view.

Android native mapping

void show (java.util.Hashtable options)

iOS native mapping

show: (NSDictionary*) options

setOptions(Options)

Parameters

Options - JSON Object

Return Type

JSON object

Description

Options to set on bar code.

Android native mapping

void setOptions (java.util.Hashtable barcodeOptions)

iOS native mapping

setOptions: (NSDictionary*) options

getOptions()

Parameters

NONE

Return Type

JSON object

Description

Retrieves the properties of the Barcode Capture Control and sends them to the callback function.

Android native mapping
java.lang.Object getOptions ( )

iOS native mapping
getOptions

readBarCode()

Parameters
NONE

Return Type
JSON object

Description
Reads a single bar code from the camera view.

Android native mapping
void readBarcode ( )

iOS native mapping
readBarcode

addBarCodeFoundListener()

Parameters
barcodeFoundListener - function

Return Type
NONE

Description
Function called when a bar code is found.

Android native mapping
void addBarCodeFoundListener (com.konylabs.vm.Function barcodeFoundListener)

iOS native mapping
addBarCodeFoundListener:(id *)callBack

removeBarCodeFoundListener()

Parameters
NONE

Return Type
NONE

Description
Removes the listener added for the bar code found event.

Android native mapping
void removeBarCodeFoundListener ( )

iOS native mapping
removeBarCodeFoundListener

remove()

Parameters
NONE

Return Type
NONE

Description
Removes the bar code control.

Android native mapping
void remove ( )

iOS native mapping
remove

Sample code

```
//Creates an object of class 'BarcodeControl'
var  barcode = new Kofax.UI.BarcodeControl();

//Add bar code found listener.Called when barcode is found
barcode.addBarCodeFoundListener(barcodeFoundListener);

//Shows the barcodes with the specified options
barcode.show(barcodeOptions);

//Reads the barcode and triggers barcode found event
barcode.readBarcode();
```

The Credit Card Control class

This class handles credit card related APIs.

Native Android
Package name: `com.kofax.capture`
Class Name: CreditCardWrapper

Native iOS
Class Name: KofaxCreditCardControl

JavaScript Constructor

```
var creditcard = new Kofax.UI.CreditCardControl();
```

Credit Card Control API

show()

Parameters
NONE

Return Type
NONE

Description
Shows credit card view.

Android native mapping
void show ()

iOS native mapping
-(void) show

addCardExtractedListener()

Parameters
cardExtractedListener-Function

Return Type
NONE

Description
Function called when credit card data is extracted.

Android native mapping
void show () void addCardExtractedListener (com.konylabs.vm.Function cardExtractionListener)

iOS native mapping
addCardExtractedListener:(id)callBack

addCreditCardExtractionFailedListener()

Parameters
creditCardExtarctionFailedListener-Function

Return Type
NONE

Description
Function called when credit card extraction fails.

Android native mapping
void addCreditCardExtractionFailedListener (com.konylabs.vm.Function creditCardExtractionFailedListener)

iOS native mapping
addCreditCardExtractionFailedListener:(id)callBAck

removeCardExtractedListener()

Parameters
NONE

Return Type
NONE

Description
Removes the listener for the credit card extracted event.

Android native mapping
void removeCardExtractedListener ( )

iOS native mapping
removeCardExtractedListener

removeCreditCardExtractionFailedListener()

Parameters
NONE

Return Type
NONE

Description
Removes the listener for the credit card extraction failed event.

Android native mapping
void remove ( )

iOS native mapping
removeCreditCardExtractionFailedListener

Sample code

```
//Creates an object of class 'CreditCardControl'
var creditCard = new Kofax.UI.CreditCardControl();

//Add card extracted listener.Called when credit card extraction is successful
creditCard.addCardExtractedListener(cardExtractionListener);

//Invokes method 'show' on the object
creditCard.show();
```

The Image Review Edit Control class

This class handles ImageEdit related APIs.

Native Android

Package name: `com.kofax.preview`

Class Name: ImageReviewEditControl

Native iOS

Class Name: KofaxImageReviewEdit

JavaScript Constructor

```
var imageReviewEdit = new Kofax.UI.ImageReviewEdit();
```

Image Review Edit API

show(Options)

Parameters

Options-JSONObject

Return Type

NONE

Description

Shows the Image Review and Edit.

Android native mapping

void show java.util.Hashtable imageEditReviewOptions)

iOS native mapping

showImageReviewEdit:(NSDictionary *)options

setOptions(Options)

Parameters

Options-JSONObject

Return Type

JSON object

Description

Options to set on the Image Review Edit.

Android native mapping

void setOptions (java.util.Hashtable ierOptions)

iOS native mapping

setImageReviewEditOptions:(NSDictionary*)options

getOptions()

Parameters

NONE

Return Type

JSON object

Description

getImageReviewEditOptions returns the options on the Image Review Edit.

Android native mapping

java.lang.Object getOptions ( )

iOS native mapping

getImageReviewEditOptions

setImage ()

Parameters

mage ID - String

Return Type

JSON object

Description

Displays the Image Object in Image Review Edit.

Android native mapping

java.lang.Object setImage (java.lang.String imageID)

iOS native mapping
setImageToImageReviewEdit:(NSString*)imageId

getImage ()
Parameters
NONE

Return Type
JSON object

Description
getImageOfImageReviewEdit returns the image of Image Review Edit.

Android native mapping
java.lang.Object getImage ( )

iOS native mapping
getImageOfImageReviewEdit()

clearImage ()
Parameters
NONE

Return Type
NONE

Description
Clears the image in Image Review Edit.

Android native mapping
void clearImage ( )

iOS native mapping
clearImageOfImageReviewEdit()

remove ()
Parameters
NONE

Return Type
NONE

Description
Removes the Image Review Edit.

Android native mapping

void remove ( )

iOS native mapping

removeImageReviewEdit()

addCropTetragonListener()

Parameters

Function cropTetragonListnerCallback

Return Type

NONE

Description

Function which gets called when Tetragon is selected, and the user clicks Done in the ImageEditReview control.

Android native mapping

void addCropTetragonListener (com.konylabs.vm.Function cropTetragonListener)

iOS native mapping

addCropTetragonListener:(id)callBack

Sample code

```
//create an object of class ImageReviewEdit
 var  imageReviewEdit = new Kofax.UI.ImageReviewEdit();

//Set the capturedImage to ReviewEdit
 imageReviewEdit.setImage(capturedImageID);

//Show crop rectangle
 imageReviewEdit.showCropRectangle(true);

//Add crop tetragon listener.Called when cropping is done
imageReviewEdit.addCropTetragonListener(cropTetragon);

//show EditReview with the specified options
imageReviewEdit.show(options);
```

The Image Array

This section describes the ImageArray class in detail.

ImageArray

This class handles all image related operations such as getting base64 from an image, "image write to file" and so on.

Native Android
Package name: `com.kofax.engines`
Class Name: ImageArray

Native iOS
Class Name: KofaxImageArray

JavaScript Constructor

```
var imageArray = Kofax.ImageArray.ImageArray();
```

ImageArray API

ImageArray()

Parameters
NONE

Return value
Default constructor

Description
ImageArray().

Android native mapping
ImageArray

iOS native mapping
-(id) init

getImageIDs()

Parameters
NONE

Return value
JSON Array

Description
Returns all Image IDs as JSON array.

Android native mapping
java.lang.Object getImageIDs ( )

iOS native mapping
-(NSDictionary*) getImageIDs

removeImages(JSONArray)

Parameters

imageIDs- JSONArray

Return value

NONE

Description

Removes the images for given IDs.

Android native mapping

void removeImages (java.util.Hashtable imageIDs)

iOS native mapping

-(void) removeImages:(NSDictionary*) imageIDs

removeAllImages()

Parameters

NONE

Return value

NONE

Description

Clears the ImageArray.

Android native mapping

void removeAllImages ( )

iOS native mapping

-(void) removeAllImages

writeImageToFile(JSONObject)

Parameters

JSON Object

Return value

NONE

Description

Writes image to a specific file path.

Android native mapping

java.lang.Object writeImageToFile (java.util.Hashtable filePathDetails)

iOS native mapping
-(NSDictionary*) writeImageToFile:(NSDictionary*) filePathDetails

readImageFromFile(ImageID)

Parameters
imageID-String

Return value
NONE

Description
Reads the image bitmap from a file.

Android native mapping
java.lang.Object readImageFromFile (java.lang.String imageID)

iOS native mapping
-(NSDictionary*) readImageFromFile:(NSString*) imageID

clearImageBitmap(ImageID)

Parameters
imageID-String

Return value
NONE

Description
Clears the bitmap for the given image ID.

Android native mapping
java.lang.Object clearImageBitmap (java.lang.String imageID)

iOS native mapping
-(NSDictionary*) clearImageBitmap:(NSString*) imageID

deleteImageFromDisk(ImageID)

Parameters
imageID-String

Return value
NONE

Description
Deletes the image from the disk for the given image ID.

Android native mapping
java.lang.Object deleteImageFromDisk (java.lang.String imageID)

iOS native mapping
-(NSDictionary*) deleteImageFromDisk:(NSString*) imageID

setImageProperties(Options)

Parameters
Options - JSON Object

Return value
NONE

Description
Sets the properties for the given image ID.

Android native mapping
java.lang.Object setImageProperties (java.util.Hashtable jsonProperties)

iOS native mapping
-(NSDictionary*) setImageProperties:(NSDictionary*) jsonProperties

getImageProperties(imageID)

Parameters
imageID-String

Return value
JSON Object

Description
Gets the properties of the given Image ID in JSON format.

Android native mapping
java.lang.Object getImageProperties (java.lang.String imageID)

iOS native mapping
-(NSDictionary*) getImageProperties:(NSString*) imageID

getImageRawBytes(imageID)

Parameters
imageID-String

JSON Array with raw bytes object

Description

Returns the image raw bytes(blob) for the given image ID.

Android native mapping

java.lang.Object getImageBase64 (java.lang.String imageID)

iOS native mapping

-(NSDictionary*) getImageRawBytes:(NSString*) imageID

getImageBase64(imageID)

Parameters

imageID - String

Return value

String

Description

Returns the base64 string of the given image ID.

Android native mapping

java.lang.Object getImageBase64 (java.lang.String imageID)

iOS native mapping

-(NSDictionary*) getImageBase64:(NSString*) imageID

getFullBase64ImageData(imageID)

Parameters

imageID - String

Return value

String

Description

Returns the base64 string of the given image ID.

Returns the raw bytes as encoded base64 string of the given image ID. The application can use the kony.convertToRawBytes(base64String) API to get the actual rawBytes from the base64 string and use this accordingly.

getFullBase64ImageData is generally used on processed image in order to pass obtained rawBytes for extraction/upload to custom server. Using this on captured image will take more than 20 seconds depending on the captured image size, where as for processed image this is around 1 second. We suggest using getImageBase64 API's base64 for display purposes and getFullBase64ImageData API's base64 for extraction/upload to custom server.

Android native mapping

java.lang.Object getFullBase64ImageData (java.lang.String imageID)

iOS native mapping
-(NSDictionary*) getFullBase64ImageData:(NSString*) imageID

getImageFromBase64(base64)

Parameters
Base64 - String

Return value
JSON Object

Description
Returns the image object in JSON format.

Android native mapping
java.lang.Object getImageFromBase64 (java.lang.String strBase64)

iOS native mapping
-(NSDictionary*) getImageFromBase64:(NSString*) strBase64

getImageFromFilePath(Options)

Parameters
Options -- JSON Object

Return value
JSON Object

Description
Returns the image object in JSON format.

Android native mapping
java.lang.Object getImageFromFilePath (java.util.Hashtable filePathDetails)

iOS native mapping
-(NSDictionary*) getImageFromFilePath:(NSDictionary*) filePathDetails

Sample Code

```
//Creates an object of class 'ImageArray'
var ImageArrayObject = new Kofax.ImageArray.ImageArray();
//Invokes method 'getImageIDs' on the object
var returnedValue = ImageArrayObject.getImageIDs();
```

The Image Processor

The following section describes the ImageProcessor object in detail.

ImageProcessor

The ImageProcessor object is an image processing engine that can process a single image according to settings contained in either an ImagePerfectionProfile object, or a BasicSettingsProfile object, but not both simultaneously.

Native Android

Package name: `com.kofax.engines`

Class Name: Processor

Native iOS

Class Name: KofaxImageProcessor

JavaScript Constructor

```
var ImageProcessorObject = new Kofax.ImageProcessor.ImageProcessor();
```

ImageProcessor API

getOptions()

Parameters

NONE

Return value

JSON object containing options

Description

getImageProcessorOptions retrieves the options for the Image Processor.

Android native mapping

java.lang.Object getOptions ( )

iOS native mapping

-(NSDictionary*) getOptions

specifyProcessedImageFilepath()

Parameters

Filepath - String

Return value
JSON object

Description
Specifies the file path for the Image Processor.

Android native mapping
java.lang.Object specifyProcessedImageFilepath (java.lang.String filePath)

iOS native mapping
-(NSDictionary*) specifyProcessedImageFilepath:(NSString*) filePath

getProcessedImageFilepath()

Parameters
NONE

Return value
JSON object

Description
Retrieves the file path of the Image Processor.

Android native mapping
j ava.lang.String getProcessedImageFilepath ( )

iOS native mapping
-(NSString*) getProcessedImageFilepath

processImage(imageID,Options)

Parameters
Image ID - String, Options --JSONObject

Return value
JSON object

Description
Processes the specified image.

Android native mapping
java.lang.Object processImage (java.lang.String imageID, java.util.Hashtable imageProcessorOptions)

iOS native mapping

-(NSDictionary*) processImage:(NSString*) imageID withOptions:(NSDictionary*) imageProcessorOptions

cancelProcessing()

Parameters

NONE

Return value

NONE

Description

Cancels the image processing.

Android native mapping

void cancelProcessing()

iOS native mapping

void cancelProcessing()

doQuickAnalysis(imageID,Options)

Parameters

ImageID - String, Options -JSONObject

Return value

JSON object

Description

Checks the image quality and returns the QuickAnalysis Feedback.

Android native mapping

void doQuickAnalysis (java.lang.String imageID, java.util.Hashtable options)

iOS native mapping

-(void) doQuickAnalysisForImage:(NSString*) imageID withSettings:(NSDictionary*) options

getQuickAnalysisOptions()

Parameters

NONE

Return value

JSON object containing options

Description
Returns a JSON object.

Android native mapping
java.lang.Object getQuickAnalysisOptions ( )

iOS native mapping
-(NSDictionary*) getQuickAnalysisOptions

addImageOutEventListener()

Parameters
imageOutEventListner - function

Return value
NONE

Description
Function called when an image is processed.

Android native mapping
void addImageOutListener (com.konylabs.vm.Function imageOut)

iOS native mapping
-(void) addImageOutEventListener:(Callback*) imageOut

addProcessProgressListener()

Parameters
processProgressListener - function

Return value
NONE

Description
Function called for "image processing progress."

Android native mapping
void addProcessProgressListener (com.konylabs.vm.Function processProgressCallback)

iOS native mapping
-(void) addProcessProgressListener:(Callback*) processProgressCallback

addAnalysisCompleteListener()

Parameters
analysisCompleteListener - function

Return value
NONE

Description
Function called for "image analysis progress."

Android native mapping
void addAnalysisCompleteListener (com.konylabs.vm.Function analysisCompleteCallback)

iOS native mapping
-(void) addAnalysisCompleteListener:(Callback*) analysisCompleteCallback

removeImageOutEventListener()

Parameters
NONE

Return value
NONE

Description
Removes the listener added for the image processed.

Android native mapping
void removeImageOutListener ( )

iOS native mapping
-(void) removeImageOutEventListener

removeProcessProgressListener()

Parameters
NONE

Return value
NONE

Description
Removes the listener added for "image process progress."

Android native mapping
void removeProcessProgressListener ( )

iOS native mapping
-(void) removeProcessProgressListener

removeAnalysisCompleteListener()

Parameters

NONE

Return value

NONE

Description

Removes the listener added for "image analysis progress."

Android native mapping

void removeAnalysisCompleteListener ( )

iOS native mapping

-(void) removeAnalysisCompleteListener

Sample Code

```
//Creates an object of class 'ImageProcessor'
var ImageProcessorObject = new Kofax.ImageProcessor.ImageProcessor();
//Invokes method 'doQuickAnalysis' on the object
ImageProcessorObject.doQuickAnalysis(imageID, options);
```

JSON definitions

This section contains definitions for JSON formatted items.

General response structure

All JSON objects returned by the APIs have the structure defined here. Only if success is TRUE does the response contain valid data.

```
{
success:true/false,
errorInfo:{
      ErrorMsg: String,
      ErrorDesc: String
},
response:{
      method specific key: value
      }
}
```

Camera options

The JSON definitions in this section consist of three main keys used to configure the SDK camera:

- Cameraoptions

- checkcaptureoptions
- documentcaptureoptions

Cameraoptions

These options are common irrespective of the selected type of experience. The following JSON structure contains default values that can be configured in the application. These values are set to properties in the kfxKUIImageCaptureControl class in the Atalasoft MobileImage Kony FFI.

iOS

Framework: kfxLibUIControls

Class: kfxKUIImageCaptureControl

Android

Package: com.kofax.kmc.kui.uicontrols

Class: ImageCaptureView

Check Capture experiences

There are two types of experience: UniformGuidance and Check Capture Experience. The experience shown to the user depends on the value of the JSON capture experience key.

CheckCaptureExperience(captureexperience:0)

If the selected experience is "checkcapture," values from checkcaptureoptions are set as properties in the kfxKUICheckCaptureExperience class in the Mobile SDK.

iOS

Framework: kfxLibUIControls

Class: kfxKUICheckCaptureExperience

Android

Package: com.kofax.kmc.kui.uicontrols.captureanimations

Class: CheckCaptureExperience

CheckCaptureExperience(captureexperience:1)

If the selected experience is "uniform guidance," values from checkcaptureoptions are set as properties in the kfxKUIDocumentCaptureExperience class in the Mobile SDK.

iOS

Framework: kfxLibUIControls

Class: kfxKUIDocumentCaptureExperience

Android

Package: com.kofax.kmc.kui.uicontrols.captureanimations

Class: DocumentCaptureExperience

Check capture options

```
cameraoptions :     {
        flashmode : "OFF",
        devicedeclinationpitch : 0,
        deviceDeclinationRoll : 0,
        levelthresholdpitch : 7,
        levelthresholdroll : 7,
        stabilitydelay : 75,
        videoframe : false,
        continuouscapturemode:false,
        captureexperience:1
    },
checkcaptureoptions: {
tutorialenabled: true,
vibrationenabled:true,
outerviewfindercolor:"#00000000",
checkdetectionsettings:
{
checkSide:0,
targetframeaspectratio:2.18,
targetframepaddingpercent: 5,
targetframecenter: {
x: null,
y: null
},
zoomminfillfraction:0.65,
zoommaxfillfraction:1.5,
turnskewangletolerance:10,
tolerance:0.15
},
capturecriteria: {
stabilitythresholdenabled: true,
stabilitythreshold: 95,
focusenabled: true,
rollthresholdenabled: false,
rollthreshold: 7,
pitchthreshold: 7,
pitchthresholdenabled: false
},
capturedmessage :        {
backgroundcolor : "#70000000",
textcolor : "#01DF01",
message: "Done",
visible :true
},
centermessage :        {
backgroundcolor : "#70000000",
textcolor : "#FFFFFFFF",
message:"Center the Check",
visible : true
},
userinstructionmessage: {
backgroundcolor: "#00000000",
textcolor: "#FFFFFFFF",
message: "Fill viewable area with check",
visible: true
},

zoominmessage : {
backgroundcolor : "#70000000",
textcolor : "#FFFFFFFF",
message: "Move Closer",
visible : true
},
```

```
zoomoutmessage :         {
backgroundcolor : "#70000000",
textcolor : "#FFFFFFFF",
message: "Move Back",
visible : true
},
holdsteadymessage: {
backgroundcolor: "#70000000",
textcolor: "#FF0000",
message: "Hold Steady",
visible: true
}
}
```

Uniform guidance options

```
cameraoptions :     {
        flashmode : "OFF",
        devicedeclinationpitch : 0,
        deviceDeclinationRoll : 0,
        levelthresholdpitch : 7,
        levelthresholdroll : 7,
        stabilitydelay : 75,
        videoframe : false,
        continuouscapturemode:false,
        captureexperience:1
    },
documentcaptureoptions :     {
        capturecriteria :       {
            focusenabled : true,
            pitchthreshold : 15,
            pitchthresholdenabled : true,
            rollthreshold : 15,
            rollthresholdenabled : true,
            stabilitythreshold : 95,
            stabilitythresholdenabled : true
        },
        capturedmessage :        {
            backgroundcolor : "#70000000",
            textcolor : "#01DF01",
            message: "Done",
            visible :true
        },
        centermessage :        {
            backgroundcolor : "#70000000",
            textcolor : "#FFFFFFFF",
            message:"Center the document",
            visible : true
        },
        documentdetectionsettings :         {
            targetframecenter :             {
                x : 0.00,
                y : 0.00
            },
            longaxisthreshold : 85,
            shortaxisthreshold : 85,
            targetframepaddingpercent : 5,
            targetframeaspectratio:0.75,
            tolerance : 0.15,
            turnskewangletolerance : 10,
            zoommaxfillfraction : 1.5,
            zoomminfillfraction : 0.2

        },
        holdsteadymessage :         {
            backgroundcolor : "#70000000",
            textcolor : "#FF0000",
            message: "Hold Steady",
```

```
            visible : true
        },
        userinstructionmessage :  {
            backgroundcolor : "#00000000",
            textcolor : "#FFFFFFFF",
            message: "Fill the area with the document",
            visible : true
        },
        zoominmessage :         {
            backgroundcolor : "#70000000",
            textcolor : "#FFFFFFFF",
            message: "Move Closer",
            visible : true
        },
        zoomoutmessage :         {
            backgroundcolor : "#70000000",
            textcolor : "#FFFFFFFF",
            message: "Move Back",
            visible : true
        },
        outerviewfindercolor : "#70000000",
        tutorialenabled : false,
        vibrationenabled : true
    }
```

Image review edit options

The following JSON definition specifies the tetragon corners used to crop the image. Crop line, crop corner color, and crop style can also be configured.

iOS

Framework: kfxLibUIControls

Class: kfxKUIImageReviewAndEdit

Android

Package: com.kofax.kmc.kui.uicontrols

Class: ImgReviewEditCntrl

```
{
        Tetragon : {
            show : true,
  TopLeft :{
                x : 0,
                y : 0
            },
  TopRight :{
                x : 0,
                y : 0
            },
  BottomLeft : {
                x : 0,
                y : 0
            },
  BottomRight : {
                x : 0,
                y : 0
            }
        },
        Crop :{
   lineColor : "#AABBCCDD",
```

```
   cornerColor : "#AABBCCDD",
   lineStyle : "LINE_STYLE_SOLID"
```

Image array options

```
writeImageToFile method JSON is
{
 filePath : "Give Filepath Here",
 imageID: "Give Image ID Here"
}
getImageFromFilePath method JSON is
{
 filePath: "Give Filepath Here",
 mimeType: "MIMETYPE_JPG"
}
setImageProperties method JSON is
{
 imageID: "Give imageID Here",
 mimeType: "MIMETYPE_JPG",
 dpi: 75,
 tag: 1,
 filePath: "Give Filepath Here",
 createDateTime: "Give ISO 8601 Date Format String Here",
 jpegQuality: 75
}
```

Barcode capture control options

The following options are used to configure BarCodeCaptureControl.

iOS

Framework: kfxLibUIControls

Class: kfxKUIBarCodeCaptureControl

Android

Package: com.kofax.kmc.kui.uicontrols

Class: BarCodeCaptureView

```
    {
   searchDirection:["HORIZONTAL","VERTICAL"],
   symbologies:["PDF417","CODE39"],
   guidingLine:"LANDSCAPE"
    }
```

Quick analysis options

Use the quickanalysis settings to get feedback on image quality.

iOS

Framework: kfxLibEngines

Class: kfxKEDQuickAnalysisSettings

Android

Package: com.kofax.kmc.ken.engines.data

Class: QuickAnalysisSettings

```
{
    QuickAnalysisSettings =     {
        enableBlurDetection = true;
        enableGlareDetection = true;
        enableSaturationDetection = true;
        generateImage = true;
        glareDetectedThreshold = "0.02";
        glareDetectionIntensityFraction = "0.03";
        glareDetectionIntensityThreshold = 230;
        glareDetectionMinimumGlareAreaFraction = "0.01";
        glareDetectionNumberOfTiles = 100;
    };
}
```

Kony Studio

The following sections contain informatin about building the sample app with Kony Studio.

Note Kony has dropped support for the Kony Studio IDE, therefore, as of the next release, support for Kony Studio will be dropped from this product as well.

Bulding the sample app

To build and use the sample application with the integrated FFI, follow these instructions. Atalasoft MobileImage Kony FFI Mobile SDK

1. Import the sample application into Kony Studio.
2. Import `Kofax-Kony-FFI.zip` file (available in the SDK release folder) into the Atalasoft MobileImage Kony FFI sample app.
    a. Right-click the sample application in Kony Studio and import a custom library.
    b. Select `Kofax-Kony-FFI.zip`.
3. Build the Kony application at least once by selecting a supported platform (Android or iOS).
4. Go to the Navigator tab in Kony Studio.
5. Select the Kony application.
6. Refresh the Kony application once.
7. Go to the `Resources>Customlibs>lib>Android` (or iOS) folder.
8. Paste the SDK libs from the `Jar` folder (or framework files for iOS).

Note For Android do not copy `repackaged.jar`.

9. For Android, copy the values folders (e.g. `values-de`) from the localization folder (`Android\MobileSDK_libs\jar\localization`) of the Mobile SDK to this path: `resources/mobile/native/android/` in the Navigator tab. Then, edit the `strings.xml` file in all the values folders to add a string by including this line: `<string name="app_name">Your app name</string>`.
10. For Android, copy or merge the `drawables` folders (e.g. `drawable-xhdpi`) from the `drawables` folder (`Android\MobileSDK_libs\jar\drawables`) of the Mobile SDK to this path: `resources/mobile/native/android/` in the Navigator tab.
11. A `Libkonyjsvm.so` file for each platform is bundled with the FFI. Replace the `Libkonyjsvm.so` file originally included in Kony Sudio with the appropriate platform specific file that came with the Atalasoft MobileImage Kony FFI .
12. Make a clean build of the Kony app. (For iOS, if the app is not launched automatically in Xcode, directly launch the project from the Kony workspace.)

Using the Atalasoft MobileImage Kony FFI in other apps

In order to create a new Kony application and use/integrate the Atalasoft MobileImage Kony FFI into it, follow these instructions.

1. Create an application using Kony Studio.
2. Import the `Kofax-Kony-FFI.zip` file (available in the SDK release folder) into the Atalasoft MobileImage Kony FFI sample app.
   a. Right-click the sample application in Kony Studio and import a custom library.
   b. Select the `Kofax-Kony-FFI.zip` file.
3. Build the Kony application at least once by selecting a supported platform (Android or iOS).
4. Go to the Navigator tab in Kony Studio.
5. Select the Kony application.
6. Refresh the Kony application once.
7. Go to the `Resources>Customlibs>lib>Android` (or iOS) folder.
8. Paste the SDK libs from the `Jar` folder (or framework files for iOS).

Note For Android do not copy `repackaged.jar`.

9. For Android, Copy the values folders (e.g. `values-de`) from the localization folder (`Android\MobileSDK_libs\jar\localization`) of the Mobile SDKto this path: `resources/mobile/native/android/` in the Navigator tab. Then, edit the `strings.xml` file in all the values folders to add a string by including this line: `<string name="app_name">Your app name</string>`.
10. For Android, copy or merge the `drawables` folders (e.g. `drawable-xhdpi`) from the `drawables` folder (`Android\MobileSDK_libs\jar\drawables`) of the Mobile SDK to this path: `resources/mobile/native/android/` in the Navigator tab.

11. A `Libkonyjsvm.so` file for each platform is bundled with the Atalasoft MobileImage Kony FFI . Replace the `Libkonyjsvm.so` file originally included in Kony Sudio with the appropriate platform specific file that came with the Atalasoft MobileImage Kony FFI .
12. Make a clean build of the Kony app. (For iOS, if the app is not launched automatically in Xcode, directly launch the project from the Kony workspace.)

Manually integrate the FFI library and mappings into the Kony Sample App

1. Create an application using Kony Studio.
2. In the applications tab, select your Kony application and select **Integrate Third Party** > **Manage Custom Libraries**.
3. Right-click the JavaScript FFI Definition and give it a valid Javascript namespace.
4. Copy the Mobile SDK libraries into the platform appropriate folders.
5. Right-click the Javascript name space and create a Javascript class.
6. In the Native Mappings tag, click the import button to import the required native FFI library.
7. Map the Javascript class names to the native FFI class names as well as the Javascript method names with the FFI method names.
8. Clean up the application as needed.
9. Write the Javascript file in the `modules/JS` folder of the application with the business logic.
10. Run the application. (For iOS, if the app is not launched automatically in xCode, directly launch the project from the Kony workspace.)
11. Repeat steps 3-10 as needed to update the FFI integrations.

Create your own FFI on top of the SDK

To create your own FFI (iOS or Android) on top of the Atalsoft MobileImage SDK and use it in a Kony application, refer to the following Kony documentation.

URL: http://docs.kony.com/konyonpremises/#../Subsystems/Studio_User_Guide/Content/Foreign_Function_Interface_JS.htm%3FTocPath%3DKony%2520Studio|Studio%2520User%2520Guide|Integrating%2520with%2520Third-party%2520Libraries%2520Using%2520FFI|_____0

Kony Visualizer Enterprise

Use the following information to build the sample app using Kony Visualizer Enterprise.

Building the Sample app

To build and use the sample application with the integrated FFI, follow these instructions. For Android, make sure the built-in sample app is working in your workspace.

1. Unarchive the Kony sample application, which is available in the Atalsoft MobileImage release build and import it into Kony Visualizer Enterprise.

Note For Android, an error dialog "There are compatibility issues in project and the project should be upgraded. It is …." with Cancel and Upgrade buttons will appear. Click **Upgrade**.

2. For Android, give environment specific details like the Android Home Location, change or download the SDK version specific to the project (based on the available SDK in your environment), and login into your Kony account.
3. Open Package Explorer (**Window** > **Open Perspective** > **Java**).
4. Unarchive `kofax_kony_ffi.zip` and copy `jsFfiMappings.xml` to the project root path.
5. Import `Kofax-iOS-FFI.zip/Kofax-Android-FFI.jar` from **Edit** > **Integrate Third Party** > **Manage Custom Libraries** > **Java Script FFI Definition** > **<choose one of FFI classes (License)>** > **Native Mappings** > **Mobile <Platform>** > **Library**

   Then click **Import icon** > **Import** > **<select platform specific file>** > **"OK"** > **OK** to generate the FFI generated code.
6. For Android, copy all SDK jars except the `localization` and `drawables` folders (<Downloaded `Kofax SDK Folder>/Android/MobileSDK_libs/jar`) into `/resources/customlibs/lib/Android`.
7. For Android, copy all `values` folders (`values-de`, `values-es`, etc.) from the localization folder (`Android\MobileSDK_libs\jar\localization`) of the SDK to `resources/mobile/native/android/` in the package explorer.

   Now edit `strings.xml` in the default values folder by adding the following lines.

```
<string
name="app_name">KonySampleApp</string>
<string name="search_hint"> </string>
<string name="search_label"> </string>
```

8. For Android, copy all `drawables` and merge into `resources/mobile/native/android/`.
9. For Android, go to "Kony perspective" then double click on Project Settings and Navigate to the Native tab.

   Select the Android tab, then select the Tags tab at the bottom. Do the following:

   a. Go to "Manifest Tag Attributes" and remove `xmlns:tools=http://schemas.android.com/tools`.

   b. Go to the Child tag entries under <application> tag

      Scroll to the end and remove:

```
<meta-data
tools:replace="android:name,android:value"
android:name="roboguice.annotations.packages"
android:value="com.kofax" /> <meta-data
tools:replace="android:name,android:value"
android:name="roboguice.modules"android:value="com.kofax.mobile.sdk.capture.MainModule"/
>
```

   c. Also change SDK versions, depending on your environment.
10. For Android, to make the app work in release mode: copy all Kony generated value files except `strings.xml` from the workspace `/temp/<app>/build/luaandroid/dist/<app>/res/ values/` and paste into all the `values` folders in `resources/mobile/native/android/`.

11. Go to the sample app (**Window** > **Open Perspective** > **Kony**) and Refresh the application.
12. Run the Kony app, select the platform and launch the app.

    Select Continue to build without Services.
13. For iOS, if the app is not launched automatically in XCode, directly launch the project from the Kony workspace.
14. For iOS, when you try to run the application on iOS10 devices, if you get an error like " [TabSkinModel setValue: forKey:]: attempt to set a value for a nil key," then you have to add the plugin (`http://download.kony.com/studio/70/hotfixsite.xml`) to the Kony Visualizer Enterprise. Click "Install New Software" in the Help menu item to open the installation window, and then enter the plugin url. Select "Kony Platform 7.0 - Mobile Native Channel" and continue the installation process.

    Select the release mode while building the app.
15. For more information on building and migrating the application, including Kony recommendations and documentation, see "Building and Viewing an Application" or "Migrate a Project from an Earlier Version of Kony Studio or Visualizer" in the *Kony Visualizer User Guide*:

    `http://docs.kony.com/konylibrary/visualizer/visualizer_user_guide/`

Manually import Kony-FFI in other Kony apps

Follow these instructions to create a new Kony application to use or integrate the FFI.

1. Create a Kony application using Kony Visualizer Enterprise.
2. Open Package Explorer (**Window** > **Open Perspective** > **Java**)
3. Unarchive `kofax_kony_ffi.zip` and copy `jsFfiMappings.xml`, select the Kony application in Package Explorer and paste the copied file.
4. Copy the `customlibs` folder (which is located at `kofax_kony_ffi/resources/customlibs` and which contains `Kofax-iOS-FFI.zip` and `Kofax-Android-FFI.jar`), select the Kony application in Package Explorer and paste the copied folder into the resources folder of the application.
5. For Android, copy all the SDK jars except the `localization` and `drawables` folders (`<Downloaded Kofax SDK Folder >/Android/MobileSDK_libs/jar`) into custom libs.
6. For Android, Copy the values folders (e.g. values-de) from the localization folder (`Android \MobileSDK_libs\jar\localization`) of the Mobile SDK to this path: `resources/mobile/native/android/` in the Navigator tab. Then, edit the `strings.xml` file in all the values folders to add a string by including this line:

    ```
    <string name="app_name">Your app name</string> <string name="seatch_hint"> </string> <string name="search_label"> </string>
    ```
7. For Android, copy or merge the `drawables` folders (e.g. `drawable-xhdpi`) from the `drawables` folder (`Android\MobileSDK_libs\jar\drawables`) of the Mobile SDK to this path: `resources/mobile/native/android/` in the Navigator tab.
8. For Android, go to "Kony perspective" then double click on Project Settings and Navigate to the Native tab. Select the Android tab, then select the Tags tab at the bottom. Do the following:
    a. Go to Manifest Tag Attributes and remove `xmlns:tools=http://schemas.android.com/tools`.

b. Go to the Child tag entries under <application> tag and add below android activities information:

```
<activity android:name="com.kofax.capture.CreditCardActivity"
   android:screenOrientation="portrait" >
   <intent-filter>
   <action android:name="com.kofax.android.action.CREDITCARDCAPTURE" />
   <category android:name="android.intent.category.DEFAULT" />
   </intent-filter>
</activity>
<activity
 android:name="com.kofax.mobile.sdk.capture.creditcard.CreditCardWorkflowActivity" />

<activity
 android:name="com.kofax.mobile.sdk.capture.creditcard.CardIoWrapperActivity"/>
<activity android:name="io.card.payment.CardIOActivity" android:configChanges=
   "keyboardHidden|orientation" android:hardwareAccelerated="true" />
<activity android:name="com.kofax.capture.BarcodeCaptureActivity"
   android:screenOrientation="portrait" >
   <intent-filter>
   <action android:name="com.kofax.android.action.BARCODECAPTURE" />
   <category android:name="android.intent.category.DEFAULT" />
   </intent-filter>
</activity>
<activity android:name="com.kofax.capture.CheckExperienceActivity"
   android:screenOrientation="portrait">
   <intent-filter>
   <action android:name="com.kofax.android.action.CHECKCAPTURE" />
   <category android:name="android.intent.category.DEFAULT" />
   </intent-filter>
</activity>
<activity android:name="com.kofax.capture.UniformGuidenceActivity"
   android:screenOrientation="portrait" >
   <intent-filter>
   <action android:name="com.kofax.android.action.UNIFORMCAPTURE" />
   <category android:name="android.intent.category.DEFAULT" />
   </intent-filter>
</activity>
<activity android:name="com.kofax.preview.ImageReviewEditActivity"
    android:screenOrientation="portrait" >
    <intent-filter>
    <action android:name="com.kofax.android.action.SHOWIMAGEEDITREVIEW" />
    <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

c. Go to the Application tag Attributes and make sure `android:largeHeap="true`" is specified.

d. Also change SDK versions, depending on your environment.

9. For Android, to make the app work in release mode: copy all Kony generated value files except `strings.xml` from the workspace `/temp/<app>/build/luaandroid/dist/<app>/res/ values/` and paste into all the values folders in `resources/mobile/native/android/`.

10. Go to Mange Custom Libraries (**Edit** > **Integrate Third Party Libraries** > **Manage Custom Libraries**).

11. Find the JavaScript FFI Definitions and native mappings.

12. Run the Kony app, select a platform and launch the app.

13. For iOS, if the app is not launched automatically in XCode, directly launch the project from the Kony workspace.

Manually integrate the FFI library and mappings into the Kony sample app

1. Create a Kony application in Kony Studio.
2. Go to Mange Custom Libraries (**Edit** > **Integrate Third Party Libraries** > **Manage Custom Libraries**).
3. Right click **JavaScript FFI Definition** and give a valid JavaScript namespace to it, for example: `Kofax.UI` or `Kofax.utilities`
4. Right click the JavaScript name space and create a JavaScript class.
5. Go to the Native Mappings tag and click on the Import button to import the required native FFI library (`.zip` files for iOS, `.jar` files for Android.)
6. Map the JavaScript class names to native FFI class names as well as JavaScript method names to FFI method names with valid parameter types.
7. Write a JavaScript file in the `modules/JS` folder of the application with business logic.
8. Run the application.
9. For iOS, if the app is not launched automatically in XCode, directly launch the project from the Kony workspace.
10. Repeat steps 3 to 7 to update the FFI integrations.

Create Your Own FFI on Top of the SDK

In order to create your own FFI (native iOS or Android) on top of the SDK and use it with any Kony application, follow the steps mentioned in the Kony documentation located here:

```
http://docs.kony.com/konyonpremises/#../Subsystems/Studio_User_Guide/Content/
Foreign_Function_Interface_JS.htm%3FTocPath%3DKony%2520Studio|Studio%2520User
%2520Guide|Integrating%2520with%2520Thirdparty% 2520Libraries%2520Using%2520FFI|_____0
```