Kofax Mobile ID Capture

Administrator's Guide

Version: 2.7.0

Date: 2023-08-10

Table of Contents

Preface

This guide provides instructions for developing applications for Kofax Mobile ID Capture.

Training

Kofax offers both classroom and online training to help you make the most of your product. To learn more about training courses and schedules, visit the Kofax Education Portal on the Kofax website.

Getting help with Kofax products

The Kofax Knowledge Portal repository contains articles that are updated on a regular basis to keep you informed about Kofax products. We encourage you to use the Knowledge Portal to obtain answers to your product questions.

To access the Kofax Knowledge Portal, go to https://knowledge.kofax.com.

> ⓘ The Kofax Knowledge Portal is optimized for use with Google Chrome, Mozilla Firefox, or Microsoft Edge.

The Kofax Knowledge Portal provides:

- Powerful search capabilities to help you quickly locate the information you need.

  Type your search terms or phrase into the **Search** box, and then click the search icon.
- Product information, configuration details and documentation, including release news.

  To locate articles, go to the Knowledge Portal home page and select the applicable Solution Family for your product, or click the View All Products button.

From the Knowledge Portal home page, you can:

- Access the Kofax Community (for all customers).

  On the Resources menu, click the **Community** link.
- Access the Kofax Customer Portal (for eligible customers).

  Go to the Support Portal Information page and click **Log in to the Customer Portal**.
- Access the Kofax Partner Portal (for eligible partners).

  Go to the Support Portal Information page and click **Log in to the Partner Portal**.
- Access Kofax support commitments, lifecycle policies, electronic fulfillment details, and self-service tools.

Go to the Support Details page and select the appropriate article.

Related documentation

In addition to this guide, refer to the following documentation:

- *Kofax QuickUpdater for Mobile Administrator's Guide* contains essential information about installing and configuring the Kofax QuickUpdater for Mobile.
- *Kofax Mobile Capture SDK Developer's Guide* contains essential information about installing and configuring Kofax Mobile Capture SDK.
- *Kofax Mobile ID Extracted Field Tables* is an HTML document with a complete listing of all currently supported ID fields by region and by country.
- *Real-Time Transformation Interface Administrator's Guide* contains essential information about installing and using the Real-Time Transformation Interface.

Online documentation

Product documentation for Kofax Mobile ID Capture is available at the following location.

https://docshield.kofax.com/Portal/Products/en_US/SMC/2.7.0-pm2voa50k0/KMID.htm

Chapter 1

Overview

Kofax Mobile ID Capture contains a Kofax Transformation project for extracting information from images of driver licenses or passports.

Kofax Mobile ID Capture includes the following:

- KTM Project - this folder contains the installers of Kofax Transformation project for use with Real-Time Transformation Interface.
- KTA Package - this folder contains a Kofax TotalAgility package for use with TotalAgility.

The following key features are supported:

- Identity Card, Driver License OCR data extraction.
- Duplex (front and back) ID card data extraction.
- 2D bar code (PDF417) and QR code parsing.
- MRZ parsing

Kofax Mobile ID Capture requires color tiff or jpeg images.

Usage tips

Processing passports

When extracting data from passports, keep in mind that the confidence value for the MRZ1 and MRZ2 fields is always shown as 81%, even when the correct value is extracted in the Passport Extracted Results.

There is no checksum bit for gender in the MRZ line. Therefore, the confidence for gender from MRZ will be 0.95 if the confidence of an OCR value (or any other MRZ value where a checksum is not present or fails) has been validated.

The MRZ value takes precedence over the VIZ value because the OCR for MRZ characters is more reliable. Consequently, when OCR for MRZ characters is used the value 'M' is validated because it satisfies the validation rule (M, F) 0.05 for the confidence value.

Fuzzy database lookup for zip/city/state

For variants that support it, the ODE extractor will automatically do a fuzzy search of a zip/city/state database to match OCR results to known zip codes, cities, and states. This is done in an effort in increase extraction accuracy results.

The zip/city/state fuzzy database lookup currently works with the US, Argentina, Germany, Poland and UK Regions, and is dependent on those fields being available on the corresponding ID.

Handling classification results

The score is a float, with negative values indicating low probability, while positive values indicate high confidence. The higher the positive score, the more confident the classification. A lower score means less confident. The value is unbounded.

Considering a distribution of scores, most will be within a range of [-2;2] but some can be significantly outside of that distribution.

The application can use the classification result data to perform judgments on how confident the software is of the extraction results of the given image. For example, if the confidence score is 1.5, the given image was classified and extracted with high confidence. If the confidence value is 4.7, the software gives estimate that classification was done correctly with extremely high confidence, thus the extracted data is likely very accurate.

If the classification confidence for an image is too low, it may be rejected. In that case, an exception will be returned, and no extraction will take place.

Extraction of encrypted bar codes

On-device extraction does not support encrypted bar codes. If a bar code is encrypted, the extractor will continue extraction without the bar code. It also will return an error code that notifies the user of the encrypted bar code. See Error codes for a list of codes.

For a list of supported and unsupported bar codes, see the *Kofax Mobile ID Extracted Field Tables*.

MRZ, OCR, and bar code interaction

Data extracted from MRZ fields or bar codes will overwrite corresponding values extracted by OCR fields, except in the following cases:

- When the extracted field value has higher confidence than the MRZ field value.
- When the MRZ field value is truncated.
- When the extracted field value has special characters and the extracted value (with substituted values for the special characters) matches the MRZ value for the corresponding field.

  For example, if the extracted value is "KNIEß" and the corresponding MRZ value is "KNIESS", the special characters are compared and replaced in accordance with the relevant ICAO standards for machine readable documents. In this example, the specified transliteration for ß is SS, so the original extracted value would become "KNIESS", which matches the MRZ value of "KNIESS". In this case the original OCE value (KNIEß) is preferred over the MRZ value.

This behavior should be applicable for all IDs where a field value exists in both MRZ and VIZ (OCR region or bar code) sections of the ID.

ⓘ Not all text from an MRZ will include a checksum. Since our field confidence rules place a lower confidence value on fields that do not have checksums, it is possible to have some fields returned from the MRZ with a high confidence while other fields returned from the same MRZ may have a low confidence. See the *API Reference Guide* for more details.

Chapter 2

Installation and configuration

Upgrade with Kofax TotalAgility

To upgrade Kofax Mobile ID Capture with Kofax TotalAgility, do the following.

1. Extract KofaxMobileIDCapture-2.7.0.0.zip to a folder of your choice. All instructions and examples in this guide assume the folder name is `C:\MobileIDCapture`.
2. Import the KTA Package and follow the prompts on the screen.
3. After successfully importing the KTA Package, restart the Kofax Transformation server.

Upgrade with Real-Time Transformation Interface

To upgrade Kofax Mobile ID Capture with Real-Time Transformation Interface, do the following.

1. Extract KofaxMobileIDCapture-2.7.0.0.zip to a folder of your choice. All instructions and examples in this guide assume the folder name is `C:\MobileIDCapture`.
2. To install Kofax Mobile ID Capture for use with Real-Time Transformation Interface, run `KofaxMobileIDCapture.exe` from the `KTM Project` subfolder of the folder above.
3. When you are prompted if you want to upgrade the current installation, click **Yes**.

   The Installation Wizard opens.
4. Click **Next** and follow the instructions in the Installation Wizard to upgrade the installation.

Install with Kofax TotalAgility On-Premise Multi-Tenant

If you are using Kofax Mobile ID Capture with TotalAgility On-Premise Multi-Tenant, see the *Kofax TotalAgility On-Premise Multi-Tenant Installation Guide* for installation instructions. You must use the Real Time Transformation Server when running On-Premise Multi-Tenant. Note that at least one Real Time Transformation Server is required for each tenant.

Install with Kofax TotalAgility and Docker

If you are using Kofax TotalAgility 7.9 or later, you can install Kofax Mobile ID Capture with Docker using Intelligent Automation (IA). Follow these steps and refer to the Kofax TotalAgility documentation for full information.

1. Make sure the container host with the following Docker base container is specified in the Docker file: `mcr.microsoft.com/windows:1809`
2. Install .NET Framework 4.8 if it is not already installed.
3. Follow the instructions in the *Kofax TotalAgility Installation Guide* to use Kofax Mobile ID Capture with Docker.

Uninstallation from Kofax TotalAgility

1. Log in to the TotalAgility Designer.
2. Navigate to the **Process Designer**.
3. Remove **KofaxMobileIDCaptureSync**.
4. Navigate to the **Data Designer**.
5. In the **Data Designer** tab, click **Extraction Groups**, and then delete the **KofaxMobileIDCapture** group.
6. Navigate to **System Settings** > **Scan/VRS Profiles**.
7. Delete the KofaxMobileIDCapture profile.
8. Delete the KTA Package.

Chapter 3

Extraction

High-level flow

When extracting information from an ID document, the following process occurs:

1. The color image of the ID, ideally on a solid color background, is deskewed and cropped to just the document itself.
2. The type of the cropped image is classified based on the specific issuing authority and layout.
3. The cropped image is cleaned up with an image cleanup profile specific to classification.
4. Extraction is performed on the cleaned up image.

Settings and parameters

The following settings and parameters can be used:

Input parameters

The following parameters and their valid values are described below.

Note the following:

- When using Real-Time Transformation Interface, all parameters must be prefixed with an x. The LogFolder parameter should be sent to the server as xLogFolder. When using TotalAgility, the parameter names do not require this prefix.
- When using a PUT request with Real-Time Transformation Interface, any parameter value containing a space that is passed in the URL must have "%20" in place of the space. For example, if the xLogFolder is `C:\Path To Log Folder`, use the following URL, `http://servername/mobilesdk/api/mobileid?xlogfolder=C:\Path%20To%20Log%20Folder`.

Barcode

This parameter is used to extract an ID using the image from the front side, and the encoded content from the PDF417 bar code on the back side.

ⓘ ID bar codes often contain non-printable or otherwise special characters. Non-printable characters other than carriage returns and newlines should be replaced with spaces, and JSON special characters such as the double quote (") or backslash (\) should be escaped with a backslash.

Region

This is the parameter your mobile application can use to pass the global region to the Kofax Mobile ID Capture project. This parameter is now optional.

Region
Africa
Asia
Australia
Canada
Europe
Latin America
United States

IdType

Your mobile application can use this parameter to pass an alternative ID type if the submitted document does not contain a standard ID card front. The default value is ID. When the front and back of a United States ID are sent for extraction, the back idType is set to Barcode by default. Barcode should also be used when sending only the back of United States IDs.

idType	Description
ID	Standard ID-1 size. For all non-passports and IDs without bar codes.
Barcode	PDF417 bar code type.
Passport	Standard ID-3 size.

For ID2 and other non-standard size documents, send the uncropped image to the server and set the CropImage parameter to True. This enables Kofax Mobile ID Capture to detect and crop the image to the correct dimensions.

IDSize

This parameter is only for ID Verification. Possible values are ID1, ID2, Passport, or GreenID.

CropImage

This parameter enables or disables server image processing, such as cropping, rotating, and scaling. Set the parameter to True to enable, or False to disable. The default is False.

We recommended setting the CropImage to True when performing image processing on the server. In this case, the ProcessImage parameter in Real-Time Transformation Interface should not be used or set to False. For TotalAgility, the ProcessImage parameter is obsolete and should not be used.

EnableClarityEngine

Kofax Clarity is a cloud-based OCR engine that results in improved extraction accuracy on documents captured from mobile devices, particularly in cases with glare, holograms, or other similar artifacts. This setting enables or disables the use of Kofax Clarity for OCR and is turned on by default. When disabled or when the corresponding Kofax Clarity license is not available, an on-premise OCR engine is used instead. Set it to false to disable this feature.

> ℹ This feature requires Kofax Clarity volume license. Contact Kofax Support to purchase a license if necessary.
>
> Kofax Clarity requires an HTTPS connection to external endpoints over port 443. Ensure that your firewall allows access to these URLs: `licenseapi.kofax.com` and `content-vision.googleapis.com`.

EnableLogging

This parameter can be used to troubleshoot extraction issues. The default is FALSE. When set to TRUE, it saves the following to the location specified in LogFolder (see below):

- Original image.
- Cropped image as *[input image name]-out.[input image extension]*.
- Extraction result in `ocrFields.txt` and `barcodeFields.txt` if the fields are available.

Note the following:

- When EnableLogging is set to TRUE, Kofax Mobile ID Capture response times are stored in the MIDDebugTimeLogs.csv file in the default logging folder.
- Once troubleshooting is done, this parameter should be set to FALSE to allow optimal performance.

ExtractFaceImage

This parameter can be used to request the face image of the person whose ID is being processed. The returned value is a base64-encoded string in the `FaceImage64` field. Set this parameter to True to return the face image--note that this may increase processing time. The default value is False.

ExtractSignatureImage

This parameter can be used to request the signature image of the person whose ID is being processed. The returned value is a base64-encoded string in the `SignatureImage64` field. Set this parameter to True to return the signature image--note that this may increase processing time. The default value is False.

LicenseSystemID

For use in conjunction with Kofax Clarity on non-production systems such as backup or test servers that use a secondary serial number. This setting is empty and not used by default. When using Kofax Clarity on a non-production system, you need to change it to the value of the System ID provided in your original order email as follows.

For KTM projects:

1. Open the `KofaxMobileIDCapture.fpr` transformation project.
2. Open the Script Variables window.
3. Change the value of `LicenseSystemID` to the System ID mentioned above.
4. Save the project and restart the Real-Time Transformation Interface Web site or IIS for the change to take effect.

For KTA packages:

1. Navigate to the Data page in Kofax TotalAgility Designer.
2. Open the Server Variables page and filter for the KofaxSMCs category.
3. Select `SystemLicenseID` and update its value to the System ID mentioned above.

LogFolder
This parameter specifies the root directory where logging items are saved when EnableLogging is TRUE. The default value is `C:\ProgramData\Kofax\MobileIDCapture\Logs`. The folder structure is similar to that of Real-Time Transformation Interface:

```
[LogFolder path]
>YYYYMMDD
>>HH
>>>[Input image name]
>>>>Images and result text files
```

ReadMRZOnly

Kofax Mobile ID Capture returns parsed MRZ data only. When the ReadMRZOnly parameter when set to TRUE, Kofax Mobile ID Capture does not consider Extraction and Verification parameters.

BarcodeTimeout

Sets the default timeout for reading the bar code in seconds. The default value is 10. User can change this parameter based on project need.

AutoCorrectMRZ

Specifies whether OCR replacement is done with alternate characters. By default, this is set to FALSE so that Kofax Mobile ID Capture does not perform OCR replacement. Set to TRUE to enable OCR replacement with alternate characters.

Return version number

A `ProductVersion` field is available that returns version information to a global variable when the project is loaded into the process at runtime. This variable is populated only once per project session.

Image processing profiles

Several image processing profiles are provided with Kofax Mobile ID Capture. These files can be found in the `Samples` folder where Kofax Mobile ID Capture is installed.

> ℹ Image perfection profiles are supported only for Real-Time Transformation Interface 2.0.0.1 and above.

The following text files are included.

IPProfile_MobileIDCapture_ID.txt

For use with the image of an ID-1 sized card / driver license (or any image containing MRZ lines of a supported format) captured by a mobile camera as a single page (front or back) to deskew and crop to an appropriate width and height to maximize image quality.

IPProfile_MobileIDCapture_ID2.txt

For use with the image of an ID-2 sized card / driver license (or any image containing MRZ lines of a supported format) captured by a mobile camera as a single page (front or back) to deskew and crop to a width of an appropriate width and height to maximize image quality.

IPProfile_MobileIDCapture_Passport.txt

For use with an image of an ID-3 size passport (or any image containing MRZ lines of a supported format) captured by a mobile camera as a single page to deskew and crop to an appropriate width and height to maximize image quality.

ⓘ The _ProcessMrzDoc_ operation string keyword enables the page detection algorithm specific to documents containing MRZ lines. The following MRZ document formats are recognized:

- TD1 size: 2.130" x 3.370"

  3 MRZ lines of 30 characters. Top line begins with character A, C, or I.
- TD2 size: 2.910" x 4.130"

  2 MRZ lines of 36 characters. Top line begins with character A, C, or I.
- TD3 size: 3.460" x 4.920"

  2 MRZ lines of 44 characters. Top line begins with character P.
- MRV-A size: 3.150" × 4.720"

  2 MRZ lines of 44 characters. Top line begins with character V.
- MRV-B size: 2.910" × 4.130"

  2 MRZ lines of 36 characters. Top line begins with character V.

Kofax Mobile ID Capture does not perform autocorrect of OCR data in MRZ by default.

Supported fields

The following fields are supported:

- **IsBarCodeRead**: Indicates whether or not a barcode was successfully detected and read.
- **IsOcrRead**: Indicates whether or not data was extracted as a result of OCR on the submitted image(s).
- **IsIdVerified**: Indicates whether or not the `DocumentVerificationConfidenceRating` is greater than or equal to 75. Only valid when front and back images are sent.
- **DocumentVerificationConfidenceRating**: Indicates the confidence level that the front and back of an ID is of the same document. This value is based on comparing textual OCR data to data contained within bar codes on the document. The value may range from 0-100 and is only meaningful when both the front and back images are submitted and when there is a bar code on the submitted document. The value 50 is returned when there is no bar code.

> ℹ The above two field values are set when extraction is run on an ID document and provides a basic ID check by comparing information on the front and back of an ID document. For information on the advanced ID verification feature, please refer to the *Kofax Mobile ID Verification and Facial Recognition Administrator's Guide*.

- **DocumentState**: This is the issuing state of the ID, for example any of the fifty states in the US.
- **DocumentType**: Indicates the type of ID, for example an ID.
- **ProductVersion**: Returns the version including the specific build number.
- **MRZCompositeCheckDigitValidity**: Returns Valid or Invalid value to determine the validity of the composite check digit. For individual MRZ field check validation, there is a valid key in the field for checksum validation. It returns either true or false. The checksum is supported for these fields:
  - PassportNumber
  - PersonalNumber
  - DateOfBirth
  - ExpirationDate
  - IDNumber

Field property restrictions

Some restrictions apply to the returned values for certain properties of the above fields.

- **Confidence**: The value for the confidence field can be from 0.0 to 1.0, with 1.0 being certain and 0.0 means the value is either not applicable for that ID type or not extracted.
- **FormattingFailed**: The values for the `FormattingFailed` property is always false.

Example URL calls

Samples of URL POST requests for the front and back of a driver license are provided in the `Samples` folder where Kofax Mobile ID Capture is installed. These samples are described in this section.

Real-Time Transformation Interface

The Real-Time Transformation Interface sample requires the following:

- Kofax Transformation 6.4.0 or later is installed.
- Real-Time Transformation Interface is configured at `http://localhost/MobileSDK`.
- The Transformation project is mapped as IDCapture.

RequestURL

```
http://<servername>/mobilesdk/api/IDCapture
```

Request Headers

```
Accept: application/json
Content-Type: multipart/form-data; boundary=-------------------------acebdf13572468
```

Request Body

Open RTTI_Request.txt from the `Samples` folder.

Request Response

Open RTTI_Response.txt from the `Samples` folder.

Kofax TotalAgility

The request body and request response may take a moment to open.

Request URL

`http://<servername>/TotalAgility/Services/SDK/JobService.svc/json/CreateJobSyncWithDocuments`.

Request Headers

```
Accept: application/json
Host: <servername>
Content-Type: application/json
```

Request Body

Open KTA_Request.txt from the `Samples` folder.

Response

Open KTA_Response.txt from the `Samples` folder.

Chapter 4

Update classification and extraction models

Kofax Mobile ID Capture's classification and extraction models can be automatically updated using Kofax QuickUpdater for Mobile product. Refer to the *Kofax QuickUpdater for Mobile Adminstrator's Guide* for installation and configuration instructions.

Once the Kofax QuickUpdater for Mobile has been installed and configured, follow these steps to set up automatic model updates in Kofax Mobile ID Capture. The instructions assume the Kofax QuickUpdater for Mobile Web service is accessible at `http://localhost/mobileupdater/api/ExtractionModelService/`.

Configuration with Real-Time Transformation Interface for Kofax Mobile ID Capture

> ⓘ Follow these steps only if Kofax QuickUpdater for Mobile information was not entered during the installation.

1. Open the `KofaxMobileIDCapture.fpr` transformation project.
2. Open the Script Variables dialog.
3. Change the value of the `ModelUpdateWebService` variable to `http://localhost/mobileupdater/api/ExtractionModelService/`.
4. If desired, change the value of the `UpdateCheckInterval` variable to the number of minutes between each check for updates. The default is 43,200 minutes (equivalent to 30 days).
5. Restart the Real-Time Transformation Interface Web site or IIS for the change to take effect.

Configuration with Kofax TotalAgility for Kofax Mobile ID Capture

1. Log in to TotalAgility Designer.
2. Navigate to Process Designer.
3. Open the KofaxMobileIDCaptureSync process under the KofaxSmcs\KofaxMobileID category.
4. Open the Variables view.

5. Change the value of the `ModelUpdateWebService` variable to `http://localhost/mobileupdater/api/ExtractionModelService/`.
6. If desired, change the value of the `UpdateCheckInterval` variable to the number of minutes between each check for updates. The default is 43,200 minutes (equivalent to 30 days).
7. Click **Release** to save and release the change.

Chapter 5

On-Device Extraction

On-device extraction affords the app developer the choice of extracting data from an image of an ID on the device instead of sending large images to a server for extraction. Extraction is done entirely on the device, asynchronously; however, the component does periodically communicate with a Real-Time Transformation Interface or TotalAgility server for license accounting purposes. See the *Kofax Mobile SDK API Reference Guide* for information about loading licenses onto the device for offline use.

Scripting for ID extraction

You can now customize on-device extraction logic through scripting. Field data can now by manipulated by using scripts. This document shows how to update the Customization.script file in the `Models` folder.

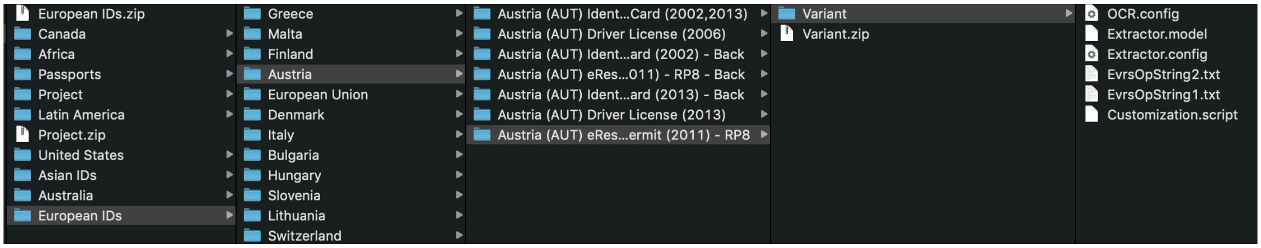
These scripts use the LUA scripting language. This document does not provide instructions on using LUA. For information about LUA, see www.lua.org.

Specifying the script

The CombinedIDs file contains models for on-device extraction. This can either be downloaded from the server or pre-complied into the app. Use the method you choose to update the file and incorporate it into the app.

Get the CombinedIDs file first. The CombinedIDs model contain different regions (such as Canada, the United States, and Australia) with each region containing different variants. Each variant contains a Customization.script file. This script is used while performing validation.

SDK looks for these variant level script file paths in validation stage. The following figure shows the variant level script that users can apply to their own scripts.



When adding a customized script to your models, name it Customization.script.

Methods used in Customization.script

The Customization.script may contain the following methods for each field:

- preValidate<FieldName>
- validate<FieldName>
- postValidate<FieldName>

The field name is case-sensitive.

For example, for DateOfIssue, following methods can be used:

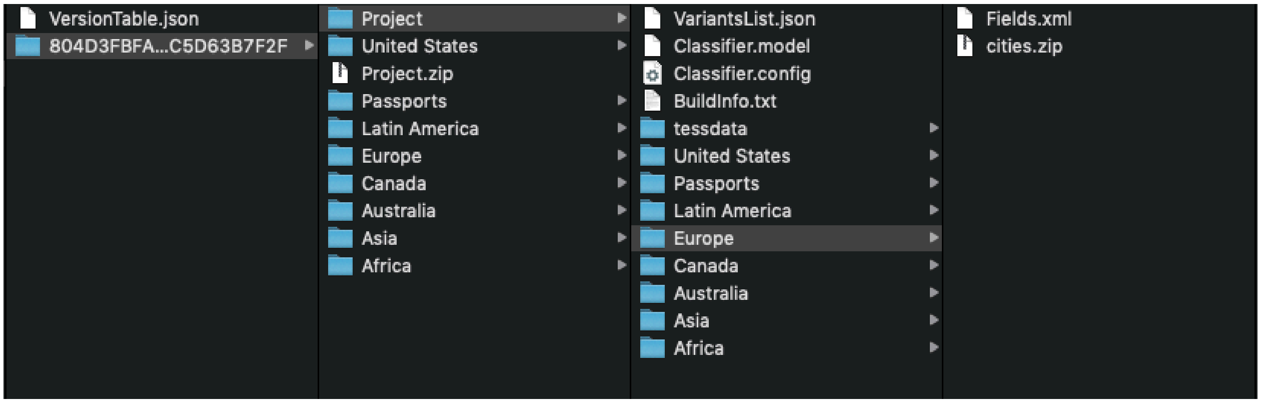
- preValidateDateOfIssue
- validateDateOfIssue
- postValidateDateOfIssue

Field names available in the variant

Users cannot define any new field name for output. Instead pre-existing fields must be used. Below are some of the available fields, which are common in every variant.

- FirstName
- MiddleName
- LastName
- FullName
- Gender
- ExpirationDate
- IssueDate
- DateOfBirth
- IDNumber
- License
- AddressLine1
- AddressLine2
- AddressLine3
- AddressLine4
- City
- State
- ZIP

Users can also check these fields from the Fields.xml file which is available in Project.zip file. The following figure shows the Field.xml file.



The SDK looks for preValidation, validation and postValidation methods in the script for each field.

These methods will be helpful for the parsing the fields like address, date, and name.

Order of execution of the methods in a script

The SDK does the following when an image is received:

1. Processing: The SDK processes the image in this stage.
2. Classification: This stage classifies the variant type (such as WI_2015_u21, WI_2015_id and TN_2012.).
3. Extraction: This stage extracts the fields available on the ID card.
4. Validation: At this stage, the user has access to the fields available on the ID. The SDK looks for the Customization.script file, which is available at the level of the classified variants.

In the validation stage, events are executed as follows:

1. All pre-Validation field methods will be triggered before Validation stage.
2. Validation field methods will be executed.
3. Post-validation field methods will be executed.

For example, in the addressLine1 field, the validateAddressLine1 method may have parsing logic, parsing addressline into city, state and ZIP fields. The postValidateAddressLine1 method may have logic of boosting confidence logic.

Script example

The following example shows how to write a script to prepare IssueDate format. The format present on the document is DD.MM.YYYY, and we want to convert this into YYYY-MM-DD. The following validateIssueDate function uses a parameter called fields. This is a data type table used to parse the IssueDate. The script converts the fields into the desired format.

```
function validateIssueDate (fields)
 local issueDateIndex = -1
 for key, value in pairs(fields) do
  if value.label == "IssueDate" then
   issueDateIndex = key
   break
  end
 end

 if issueDateIndex >= 0  then
  local issueDateValue = fields[issueDateIndex].value
  print(issueDateValue)
  date, month, year = string.match(issueDateValue, '(%d%d).(%d%d).(%d%d%d%d)')  --

  if year ~= nil and month ~= nil and date ~= nil then

   formattedDate = string.format("%s-%s-%s", year,month,date)
   return {[0] = {index = issueDateIndex, label = "IssueDate", value = formattedDate}}
  end
 end
end
```

Full Customization.script example

The following is a sample Customization.script in its entirety.

```
function validateFirstName(fields)
 --Get FirstName field
 local firstNameFieldIndex = -1

 for key, value in pairs(fields) do
  if value.label == "FirstName" then
   firstNameFieldIndex = key
   break
  end
 end

 if firstNameFieldIndex >= 0 then
  local firstNameValue = fields[firstNameFieldIndex].value
  local replaceSmall = string.gsub(firstNameValue, "a", "*")
  local replaceCapital = string.gsub(replaceSmall, "A", "*")
  local finalFirstName = {index = firstNameFieldIndex, label = "FirstName", value =
 replaceCapital}
  return {[0] = finalFirstName}
 end
end

function validateLastName(fields)
 local LastNameFieldIndex = -1

 for key, value in pairs(fields) do
  if value.label == "LastName" then
   LastNameFieldIndex = key
   break
  end
 end

 if LastNameFieldIndex >= 0 then
  local lastNameValue = fields[LastNameFieldIndex].value
  local replaceSmall = string.gsub(lastNameValue, "a", "*")
  local replaceCapital = string.gsub(replaceSmall, "A", "*")
  local finalLastName = {index = LastNameFieldIndex, label = "LastName", value =
 replaceCapital}
  return {[0] = finalLastName}
 end
end

function validateState(fields)
```

```
 local stateFieldIndex = -1

 for key, value in pairs(fields) do
  if value.label == "State" then
   stateFieldIndex = key
   break
  end
 end

 if stateFieldIndex >= 0 then
  local stateValue = fields[stateFieldIndex].value
  stateTable = {}
  stateTable["AL"] = "Alabama"
  stateTable["AK"] = "Alaska"
  stateTable["AZ"] = "Arizona"
  stateTable["AR"] = "Arkansas"
  stateTable["CA"] = "California"
  stateTable["CO"] = "Colorado"
  stateTable["CT"] = "Connecticut"
  stateTable["DE"] = "Delaware"
  stateTable["FL"] = "Florida"
  stateTable["GA"] = "Georgia"
  stateTable["HI"] = "Hawaii"
  stateTable["ID"] = "Idaho"
  stateTable["IL"] = "Illinois"
  stateTable["IN"] = "Indiana"
  stateTable["IA"] = "Iowa"

  if tonumber(string.len(stateValue)) >= 2 then

   stateCode = string.sub(stateValue, 1, 2)
   print(stateCode)
   fullStateName = stateTable[stateCode]

   if fullStateName == nil  then
    print("state code not found")
   else
    local stateFieldTable = {index = stateFieldIndex, label = "State", value =
 fullStateName}
    return {[0] = stateFieldTable}
   end
  end
 end
end


function postValidateSponsor(fields)  --fields will be passed in the form of table

 --Get FirstName field
 local firstNameFieldIndex = -1

 for key, value in pairs(fields) do
  if value.label == "FirstName" then
   firstNameFieldIndex = key
   break
  end
 end

 --Get LastName field

 local LastNameFieldIndex = -1
 for key, value in pairs(fields) do
  if value.label == "LastName" then
   LastNameFieldIndex = key
   break
  end
 end


 if firstNameFieldIndex >= 0 and LastNameFieldIndex >= 0 then

  local firstNameValue = fields[firstNameFieldIndex].value
  local LastNameValue = fields[LastNameFieldIndex].value
  print(firstNameValue)
        print(LastNameValue)
  sponserName = firstNameValue .. " " .. LastNameValue
        print(sponserName)
  local sponserName = {index = firstNameFieldIndex, label = "Sponsor", value =
 sponserName}
  return {[0] = sponserName}
 end
end

function postValidateSponsorBranch(fields)  --fields will be passed from the SDK in the
 form of table

 --Get LastName field

 local LastNameFieldIndex = -1
 for key, value in pairs(fields) do
  if value.label == "LastName" then
   LastNameFieldIndex = key
   break
  end
 end

 if LastNameFieldIndex >= 0 then

  local LastNameValue = fields[LastNameFieldIndex].value
  SponsorBranch = string.len(LastNameValue)
        print(SponsorBranch)
  local SponsorBranch = {index = LastNameFieldIndex, label = "SponsorBranch", value =
 SponsorBranch}
  return {[0] = SponsorBranch}
 end
end
```

Chapter 6

Error codes

Error Code	Error Type	Message	Description
100	License	License <name> is not available.	The license indicated in the message is missing or expired.
300	Classification	Unsupported Document/ Classification failed	The document fails validation or is a type not supported by Kofax Mobile ID Capture.
300	Classification	Classification confidence is too low to continue	The document does not match the closest document type with the required degree of confidence.
500	Extraction	Unable to read the MRZ from the document	The MRZ on the document is unclear, damaged, or otherwise unreadable.
500	Extraction	Passport MRZ data not found	IDType is set to Passport, but the MRZ is missing from the image, or the document is not a passport.
500	Extraction	IDType parameter is mismatched. Please send correct IDType	The document does match the type specified in the IDType parameter.
500	Extraction	Extraction failed	OCR fails when trying to read the document.