



# Kofax Express Developer's Guide

Version: 3.3.0

Date: 2022-06-21

**KOFAX**

© 2008-2021 Kofax. All rights reserved.

Kofax is a trademark of Kofax, Inc., registered in the U.S. and/or other countries. All other trademarks are the property of their respective owners. No part of this publication may be reproduced, stored, or transmitted in any form without the prior written permission of Kofax.

# Table of Contents

<b>Legal Notice</b> .....	<b>2</b>
Preface.....	5
Related Documentation.....	5
Offline Documentation.....	6
Accessing the SDK.....	6
Getting Help with Kofax Products.....	6
System Requirements.....	7
<b>Chapter 1: Overview</b> .....	<b>8</b>
API Libraries.....	8
<b>Chapter 2: Validation Scripts</b> .....	<b>9</b>
VB.NET Templates.....	9
VB.NET Script Generation.....	9
Compatibility with Kofax Capture.....	11
<b>Chapter 3: Script Classes</b> .....	<b>12</b>
DocumentValidationScript Class.....	13
FieldScript Class.....	14
Error Handling.....	15
<b>Chapter 4: Kofax Express Validation Library Classes</b> .....	<b>17</b>
Mapping Kofax Capture Validation API to Kofax Express.....	18
Suggested Value Support.....	20
Index Field Mapping.....	20
Index Field DefaultValue Property.....	21
Index Field Page Object.....	22
Date and Time Format.....	22
<b>Chapter 5: Document Level Validation Script Execution</b> .....	<b>23</b>
Script Storage.....	24
Sample Document Validation Script.....	25
<b>Chapter 6: Export Connectors</b> .....	<b>29</b>
Export Type Library.....	29
Kofax Express and the Export Process.....	30
Export Connector Setup.....	30
ReleaseSetupData Object.....	30
Requirements for the Export Connector Setup.....	31
Export Setup Data Object Interface - ReleaseSetupData.....	31

- IndexField..... 33
- BatchFields..... 34
- Export Connector Development..... 34
  - ReleaseData Object.....34
  - Requirements for the Export Connector..... 35
  - Export Data Object Interface - ReleaseData..... 35
  - Value.....36
  - ImageFile.....37
  - Other Export Connector API Objects..... 37
- Action Events..... 37
- Default Values in Kofax Export Connectors..... 39
- COM Servers: In-proc or Out-of-proc?..... 39
- Export Connector Registration.....40
- Differences Between Kofax Express and Kofax Capture Export Connectors..... 41


# Preface

This guide is intended for developers who are responsible for creating and customizing validation scripts and export connectors in the Kofax Express environment.

## Related Documentation

The documentation set for Kofax Express 3.3.0 is available here:<sup>1</sup>

<https://docshield.kofax.com/Portal/Products/Express/3.3.0-8vueggfgfv/Express.htm>

You can also access individual guides and online help directly from the Kofax Express 3.3.0 product. When you click the  Help button located in the top right corner of the Kofax Express window or press F1 on the keyboard, the online documentation appears in a new browser window.

 If the security policy for your organization restricts Internet access or the Internet connection is not stable, you can [access the documentation in offline mode](#) while using the product.

In addition to this guide, the documentation set includes the following items:

### ***Kofax Express Release Notes***

Contains late-breaking information that is not available in other Kofax Express documentation.

### ***Kofax Express Technical Specifications***

Contains information on supported operating systems and other system requirements.

### ***Help for Kofax Express***

Gives you online product assistance, including step-by-step procedures, description of the typical task workflow, and details about the user interface.

### ***Kofax Express Installation Guide***

Includes the information you need to successfully install Kofax Express and activate the product license.

### ***Kofax Express SDK***

Includes an API reference and the information about creating custom export connectors and validation scripts.

---

<sup>1</sup> You must be connected to the Internet to access the full documentation set online.

## Offline Documentation

To make the documentation available for use in offline mode, obtain the documentation files from the Kofax Express product package that you downloaded from the [Kofax Fulfillment Site](#). The product package includes the following documentation files for offline use:

- `KofaxExpressDocumentation_3.3.0_EN.zip`  
Contains the entire Kofax Express documentation set in English. This file is required for all users working in offline mode.
- Zip files that contain Kofax Express documentation translated to Arabic, Czech, German, Spanish, French, Italian, Japanese, Portuguese (Brazilian), Russian, and Chinese (Simplified). For example, the German .zip file name is `KofaxExpressDocumentation_3.3.0_DE.zip`.

The English .zip file includes the `Help` and `Print` folders:

- The `Print` folder contains the Installation Guide and the Developer's Guide.
- The `Help` folder contains Help for Kofax Express.

The `Print` and `Help` folders for the other languages contain only the translated Installation Guide and Help for Kofax Express, respectively.

After you install the product, extract the contents of the `Print` folder to a location on your computer. Make sure that you extract the `Print` folder from the .zip file for English and the .zip file for the other language that you require.

## Accessing the SDK

The Kofax Express SDK is available the [Kofax Product Documentation Site](#).

## Getting Help with Kofax Products

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

To access the Kofax Knowledge Base:

1. Go to the [Kofax website](#) home page and select **Support**.
2. When the Support page appears, select **Customer Support > Knowledge Base**.

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

The Kofax Knowledge Base 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.  
Scroll through the Kofax Knowledge Base home page to locate a product family. Then click a product family name to view a list of related articles. Please note that some product families require a valid Kofax Portal login to view related articles.

From the Knowledge Base home page, you can:

- Access the Kofax Community (for all customers).  
Click the **Community** link at the top of the page.
- Access the Kofax Customer Portal (for eligible customers).  
Click the **Support** link at the top of the page. When the Customer & Partner Portals Overview appears, click **Log in to the Customer Portal**.
- Access the Kofax Partner Portal (for eligible partners).  
Click the **Support** link at the top of the page. When the Customer & Partner Portals Overview appears, click **Log in to the Partner Portal**.
- Access Kofax support commitments, lifecycle policies, electronic fulfillment details, and self-service tools.  
Go to the **General Support** section, click **Support Details**, and then select the appropriate tab.

## System Requirements

- Adobe Reader (to view the *Developer's Guide*)
- HTML Help Viewer and Internet Explorer (to view the *API References*)

## Chapter 1

# Overview

This guide provides the information that you will need while working with the Kofax Express API to customize index validation scripts and export connectors.

## API Libraries

Kofax Express includes API libraries that you can use to create custom validation scripts or export connectors. See the table for details.

<b>Object and File Names</b>	<b>Description</b>
Kofax.Eclipse.Scripting (Kofax.Express.Script.Interface.dll)	Used to create custom VB.NET validation scripts
Kofax.EclipseModule (Kofax.Express.EclipseModule.dll)	Used to get information about batches and index fields
AscentRelease (AscRel.dll)	Used to create custom export connectors



## Chapter 2

# Validation Scripts

You can use document validation scripts to automate the process of populating and validating index field values. Validation scripts give you the ability to do the following:

- Populate index fields with custom default values
- Set up data validation rules to handle complex database lookups
- Perform validation checks across multiple fields in a document

A validation script is launched automatically when you enter indexing mode. You can create one validation script per job.

## VB.NET Templates

Your Kofax Express installation includes templates you can use to create validation script projects:

- VB2015Template
- VB2013Template
- VB2012Template

When you click New on the Index Setup tab to create a new validation script, Kofax Express opens the template applicable to the version of Visual Basic or Visual Studio installed on your computer. Once you install Kofax Express, the templates are available from the following location:

```
ProgramData\Kofax\Kofax Express 3.3\Templates\Scripts
```

## VB.NET Script Generation

This section gives you guidelines for generating a document validation script.

**i** Sample validation scripts are available from the `Sample Scripts` folder included in the product installation files.

A new class is derived from the `DocumentValidationScript` class. The name of the new class is based on the current job name, and any non-alphanumeric characters are replaced with underscore characters.

## Sample New Class Name Results

Current Job Name	New Class
My Scan Job	My_Scan_Job
My Scan.Job	My____Scan__Job
My.Scan.Job	My_Scan_Job

You can change the name of the new class. Note that changing the job name doesn't automatically update the generated validation script. The name of the FieldScript object is based on the index field name. As with the class name, any non-alphanumeric characters in the index field name are replaced with underscore characters.

**i** Changing the default name of the FieldScript object (created by Kofax Express) in the validation script causes the event handlers of the object to not be called.

## Sample - Not Called

```
<IndexFieldVariableAttribute("Unit Price")> _
Dim WithEvents UnitPrice As FieldScript
    Private Sub UnitPrice_FieldPreProcessing( _
        ByVal sender As Object, _
        ByVal e As PreFieldEventArgs) _
        Handles UnitPrice.FieldPreProcessing

        'Do something here...

    End Sub

    Private Sub UnitPrice_FieldPostProcessing( _
        ByVal sender As Object, _
        ByVal e As PostFieldEventArgs) _
        Handles UnitPrice.FieldPostProcessing

        'Do something here...

    End Sub
```

## Sample - Called

```
<IndexFieldVariableAttribute("Unit Price")> _
Dim WithEvents Unit_Price As FieldScript
    Private Sub Unit_Price_FieldPreProcessing( _
        ByVal sender As Object, _
        ByVal e As PreFieldEventArgs) _
        Handles Unit_Price.FieldPreProcessing

        'Do something here...

    End Sub

    Private Sub Unit_Price_FieldPostProcessing( _
        ByVal sender As Object, _
        ByVal e As PostFieldEventArgs) _
        Handles Unit_Price.FieldPostProcessing

        'Do something here...
```

```
End Sub
```

### Sample Generated Script

The following sample document validation script is based on a job named *Catalog Orders* with an index field named *Unit Price*.

```
Imports Kofax.Eclipse.Scripting
Imports Kofax.EclipseModule
Public Class Catalog_Orders
    Inherits DocumentValidationScript
    <IndexFieldVariableAttribute("Unit Price")> _
        Dim WithEvents Unit_Price As FieldScript

    ' TODO: Add event handlers from the event dropdown lists for the desired events.
End Class
```

## Compatibility with Kofax Capture

The Kofax Express scripting mechanism is similar to that used in Kofax Capture. You can easily copy and paste script source code between the two products and then make a few replacements.

For instance, when copying script source code from Kofax Capture to Kofax Express, you need to replace:

- References to the VB .NET script project
- Kofax.AscentCapture.Scripting with Kofax.Eclipse.Scripting
- Kofax.AscentCaptureModule with Kofax.EclipseModule

## Chapter 3

# Script Classes

This section describes information about script classes defined in a C# library called `Kofax.Express.ScriptInterface.dll`. The namespace of the library is `Kofax.Eclipse.Scripting`. Therefore, if you reuse a script from Kofax Capture, you need to replace "Imports Kofax.AscentCapture.Scripting" with "Imports Kofax.Eclipse.Scripting". If the script code contains any instances of "Kofax.AscentCapture.Scripting.xxx", use "xxx". See the examples for more details.

A document script is made up of a single `DocumentValidationScript` object.

**i** Structure the script code so that you modify the Imports statements to port a script from one environment to the other. The example code for Kofax Express shows the recommended approach.

Example: Kofax Capture script related to script classes:

```
Imports Kofax.AscentCapture.Scripting
Private Sub Catalog_Orders_DocumentPostProcessing( _
    ByVal sender As Object, _
    ByVal e As Kofax.AscentCapture.Scripting.PostDocumentEventArgs) _
    Handles Me.DocumentPostProcessing

    <IndexFieldVariableAttribute("Unit Price")> _
        Dim WithEvents Unit_Price As FieldScript

        Dim parseResult As Boolean
        Dim unitPrice As Double
        Dim quantity As Integer
        parseResult = Double.TryParse(Me.IsQuatityOK)
        If Not parseResult Then
            Throw New ValidationException("Unit Price is not a valid number.",
Me.Unit_Price.IndexField)
        End If
    End Sub
```

Example: Kofax Express script related to script classes:

```
Imports Kofax.Eclipse.Scripting
Private Sub Catalog_Orders_DocumentPostProcessing( _
    ByVal sender As Object, _
    ByVal e As PostDocumentEventArgs) _
    Handles Me.DocumentPostProcessing

    <IndexFieldVariableAttribute("Unit Price")> _
        Dim WithEvents Unit_Price As FieldScript

        Dim parseResult As Boolean
        Dim unitPrice As Double
```

```

Dim quantity As Integer
parseResult = Double.TryParse(Me.IsQuatityOK)
If Not parseResult Then
    Throw New ValidationException("Unit Price is not a valid number.",
Me.Unit_Price.IndexField)
End If
End Sub

```

## DocumentValidationScript Class

The DocumentValidationScript class contains the events that are available for use in a validation script. You add code for a selected event to perform a custom data validation routine. You can use the following events, event arguments, and their associated properties.

### DocumentValidationScript Class Events

Events	Description
BatchLoading	Called when entering indexing mode
BatchUnloading	Called when exiting indexing mode
DocumentPreProcessing	Called each time a new document is opened in indexing mode
DocumentPostProcessing	Called after each document is closed in indexing mode
PreDocumentEventArgs	Represents the event arguments for the DocumentValidationScript.DocumentPreProcessing event
PostDocumentEventArgs	Represents the event arguments for the DocumentValidationScript.DocumentPostProcessing event.

### DocumentValidationScript Class Properties

Property	Data Type	Access Type	Description
Application	ApplicationEnum	Read-only	Returns the application executing the script. 0=None 1=Scan 2=Index 3=Export 4=Operations 5=Administrator 6=Standalone
LocaleName	String	Read-only	Returns the locale name as a string value (such as en-US for English) that identifies the language currently used to display the user interface.
UserID	Called each time a new document is opened in the indexing mode	Read-only	Returns an empty string.

## FieldScript Class

Within the DocumentValidationScript, an instance of a FieldScript exists for each index field. FieldScript events are fired when the document validation script is run.

The FieldScript class contains two events that are available for use in a validation script. You add code for a selected event to perform a custom data validation routine. You can use the following events, event arguments, and their associated properties.

Events	Description
FieldPreProcessing	Called when the field is entered in indexing mode.
FieldPostProcessing	Called when the field is exited in indexing mode.
PreFieldEventArgs	Represents the event arguments for the FieldScript.FieldPreProcessing event.
PostFieldEventArgs	Represents the event arguments for the FieldScript.FieldPostProcessing event.

The FieldScript's events are needed for the following reasons:

- The FieldPreProcessing event is needed mainly used to automatically skip a field if OCR results are confident enough (or if you can validate that the results from OCR are valid in some other way, such as verifying in a database).
- The FieldPostProcessing event is critical, as it is fired when exiting a field. This event is used to immediately trigger an error message if the field format is not valid. It is also used to fill in values in other related fields. For example, if you enter a name in field 1, you may want to populate field 2 with the associated address. This is a common use case.

### FieldScript Class Properties

Property	Data Type	Access Type	Description
Application	ApplicationEnum	Read-only	Returns the application executing the script. 0=None 1=Scan 2=Index 3=Export 4=Operations 5=Administrator 6=Standalone
IndexField	Kofax.EclipseModule.IndexField	Read-only	Exposes the IndexField object from the Kofax Express module interface.

Property	Data Type	Access Type	Description
LocaleName	String	Read-only	Returns the locale name as a string value (such as en-US for English) that identifies the language currently used to display the user interface.
UserID	Called each time a new document is opened in indexing mode	Read-only	Returns an empty string.

Use the FieldScript Class methods to perform default field event processing in the overridden field event handlers.

### FieldScript Class Methods

Method	Parameters	Description
DefaultFieldFormatting	None Returns: String	Throw NotImplementedException()
DefaultFieldPostProcessing	None Returns: None	Throw NotImplementedException()

## Error Handling

A VB .NET script uses an exception during event handling to signal an error state. The Kofax Express .NET Scripting API provides the following exceptions.

Exception	Description
FatalErrorException	<p>This exception is used to signal a fatal error.</p> <p><b>Example</b></p> <p>Throw New FatalErrorException("Missing validation resource")</p> <p>If the exception is thrown from the .NET validation script, the following message box appears:</p> <p>You cannot continue indexing the batch due to the following error:</p> <p>Script Error: {0}.</p> <p>When the user clicks OK, the application exits indexing mode.</p>

Exception	Description
RejectAndSkipDocumentException	<p>This exception is used to reject and skip a document, and to advance to the next document. The DocumentPostProcessing event is called if the exception is thrown from DocumentPreProcessing.</p> <p><b>Example</b></p> <p>Throw New  RejectAndSkipDocumentException("Document does not have the correct format")</p> <p>If the exception is thrown from the .NET validation script, the following message box appears:</p> <pre>"The current document is rejected due to the following error: Script Error: {0}."</pre> <p>When the user clicks OK, the application advances to the next document. In case the current document is the last document in the batch, the application exits indexing mode.</p>
ValidationErrorException	<p>This exception is used to signal a validation error. There are two versions for this exception:</p> <ul style="list-style-type: none"> <li>• ValidationErrorException(ErrMsg): If this version is used, the error message appears and does not advance to the next field. The focus remains on the last field being validated.</li> <li>• ValidationErrorException (ErrMsg, IndexField): This version is similar to the prior version for ValidationErrorException, except that it allows the developer to set the focus on a particular field. IndexField specifies the field to get the focus. If the field does not exist or is null, then the last field being evaluated gets the focus. For example, if cross-field validation is being performed in the DocumentPostProcessing event and a computation routine determines that the third field does not match the sum of the first and second fields, an exception is thrown with the third field getting the focus.</li> </ul> <p>If the exception is thrown from the .NET validation script, a balloon message displays the error, similar to the validation error.</p>



## Chapter 4

# Kofax Express Validation Library Classes

Kofax Express uses a subset of the Kofax Capture scripting interface, which relies on a series of other classes known as the ACCI (Ascent Capture COM Interface) library.

The Kofax Express objects are extended to support all methods and properties of ACCI objects used in the Kofax Capture Document Validation script code. The namespace of the Kofax Express Validation Library is `Kofax.EclipseModule`. Its assembly name is `Kofax.Express.EclipseModule.dll`.

If you reuse a script from Kofax Capture, you need to replace the import "Imports Kofax.AscentCaptureModule" with Imports "Kofax.EclipseModule". If the script code contains any instances of "Kofax.AscentCaptureModule.xxx", use "xxx". See the following samples.

### Sample: Kofax Capture script related to validation library classes

```
Imports Kofax.AscentCaptureModule

Private Sub Catalog_Orders_DocumentPostProcessing( _
    ByVal sender As Object, _
    ByVal e As PostDocumentEventArgs) _
    Handles Me.DocumentPostProcessing

    <IndexFieldVariableAttribute("Unit Price")> _
        Dim WithEvents Unit_Price As FieldScript
        Dim parseResult As Boolean
        Dim unitPrice As Double
        Dim quantity As Integer
        parseResult = Double.TryParse(Me.IsQuatityOK)
        If Not parseResult Then
            Throw New ValidationErrorException("Unit Price is not a valid number.",
Me.Unit_Price.IndexField)
        End If

' Get the Document object
    Dim document As Kofax.AscentCaptureModule.Document
    document = e.Document
    ' do some thing here...

End Sub
```

### Sample: Kofax Express script related to validation library classes

```
Imports Kofax.EclipseModule

Private Sub Catalog_Orders_DocumentPostProcessing( _
    ByVal sender As Object, _
    ByVal e As PostDocumentEventArgs) _
    Handles Me.DocumentPostProcessing

    <IndexFieldVariableAttribute("Unit Price")> _
```

```

Dim WithEvents Unit_Price As FieldScript
Dim parseResult As Boolean
Dim unitPrice As Double
Dim quantity As Integer
parseResult = Double.TryParse(Me.IsQuantityOK)
If Not parseResult Then
    Throw New ValidationErrorException("Unit Price is not a valid number.",
Me.Unit_Price.IndexField)
End If

' Get the Document object
Dim document As Document
document = e.Document
' do some thing here...

End Sub

```

## Mapping Kofax Capture Validation API to Kofax Express

This section describes how the classes and objects for the Ascent Capture COM Interface used in Kofax Capture Document Validation Script code are mapped to Kofax Express Document Validation Script code.

ACCI objects	Properties/methods used in script	Value returned by Kofax Express
<b>Batch</b>	<b>Methods</b>	
	All methods	Throw NotImplementedException()
	Properties	
	ClassName	Job name
	CreateTime	Batch creation time
	Name	Batch name
	PublishTime	Batch last write time
	All other properties	Throw NotImplementedException()
<b>Document</b>	<b>Methods</b>	
	All methods	Throw NotImplementedException()
	<b>Properties</b>	
	ClassName	Job name
	IndexFieldCount	The number of index fields in the document
	IndexFields	Index fields for the document
PageCount	Throw NotImplementedException()	

	Pages	Throw NotImplementedException()
	All other properties	Throw NotImplementedException()
<b>Page</b>	<b>Methods</b>	
	All methods	Throw NotImplementedException()
	<b>Properties</b>	
	All properties	Throw NotImplementedException()
<b>Index Field</b>	<b>Methods</b>	
	All methods	Throw NotImplementedException()
	<b>Properties</b>	
	Batch	Points to the batch object that contains this index field
	DefaultValue	
	Document	Current document
	Length	
	Name	Index field label
	Numeric	
	NumericPrecision	
	NumericScale	
	Page	Throw NotImplementedException()
	Required	Same setting as Kofax Express
	SqlType	
	Sticky	Same setting as Kofax Express
	SuggestedValueCount	Number of suggested values
	SuggestedValues	
	ZoneHeight	Throw NotImplementedException()
	ZoneLeft	Throw NotImplementedException()
	ZoneTop	Throw NotImplementedException()
	ZoneWidth	Throw NotImplementedException()

	Confidence	Index field recognition confidence
	DisplayLabel	Index field label
	ReadOnly	Same setting as Kofax Express
	Value	Index field value
	All other properties	Throw NotImplementedException()
<b>SuggestedValues</b>	<b>Methods</b>	
	Add(object)	Add the specified SuggestedValue object to the collection
	Remove(object)	Remove the specified SuggestedValue object from the collection
	RemoveAll()	Remove all SuggestedValue objects from the collection
	<b>Properties</b>	
	Count	A number of SuggestedValue objects in the collection
<b>SuggestedValue</b>	<b>Properties</b>	
	OrderNumber	Order number of the value within the set of suggested values. It is 1-based.
	Value	String value

## Suggested Value Support

Kofax Capture supports the ability to dynamically populate combo boxes associated with each index field from the script. For example, if you enter a value and press Tab, the script performs a custom lookup and populates the index field.

In Kofax Express, a list of suggested values is exposed by the Index Field object, similar to Kofax Capture. If the number of suggested values of a Kofax Express index field (regardless of Single Line, Choice List, or Database Lookup field) is larger than zero, that index field is populated as a combo box with a list of values based on the suggested values.

## Index Field Mapping

Kofax Capture allows a validation script to update index field settings (if the access type of these settings is read-write). In this case, the index field controls are also updated accordingly. For example, if the ReadOnly setting for an index field is False prior to the DocumentPreProcessing call, it is updated to True by the script, and then the index field control is shown as read-only. The

change is only effective while Indexing mode is active; it has no effect on the index field settings for the application. This behavior is also applied to Kofax Express.

The `SqlType` value is an integer value, which is mapped to an enumeration in Kofax Capture as follows:

0 = `SQL_UNKNOWN_TYPE`

1 = `SQL_CHAR`

2 = `SQL_NUMERIC`

3 = `SQL_DECIMAL`

4 = `SQL_INTEGER`

5 = `SQL_SMALLINT`

6 = `SQL_FLOAT`

7 = `SQL_REAL`

8 = `SQL_DOUBLE`

9 = `SQL_DATETIME`

12 = `SQL_VARCHAR`


By default, Kofax Express does not have an exact mapping to these data types, but they can be inferred from the Validation setting.

If the Validation mask is blank, then assume that the `SqlType` is 12, `Numeric` is False, `NumericScale` is 0, `NumericPrecision` is 0, and `Length` is 25,000, similar to Kofax Capture.

If the Validation mask is purely composed of date and/or time elements, then `SqlType` is 9, `Numeric` is False, `NumericScale` is 0, `NumericPrecision` is 0. `Length` is 0.

If the Validation mask is purely composed of numbers (either explicit, or the N type mask), then `SqlType` is 4, `Numeric` is True, `NumericScale` is 0, `NumericPrecision` is 10, and `Length` is 0.

If the Validation mask is any other non-null mask, `SqlType` is 12, `Numeric` is False, `NumericScale` is 0, `NumericPrecision` is 0. `Length` should be the maximum number of characters allowed by the mask. For example, if the mask is "A(5)HHmmss", the `Length` is 11.

 The preceding content applies to both export connectors and validation.

## Index Field DefaultValue Property

Both Kofax Express and Kofax Capture always return the raw string typed in the configuration utility for the `DefaultValue` property and calculate the `Value` property based on the `DefaultValue` property prior to the `DocumentPreProcessing` call.

Kofax Express should return the raw default value as configured in `IndexSetup`.

## Index Field Page Object

The Page object is not implemented.

Kofax Express throws `NotImplementedException()` for all methods and properties of the Page object.

## Date and Time Format

Kofax Express and Kofax Capture use identical formats for date and time values, which are based on the date and time settings (such as "Short date" or "Long time") for your Windows operating system.

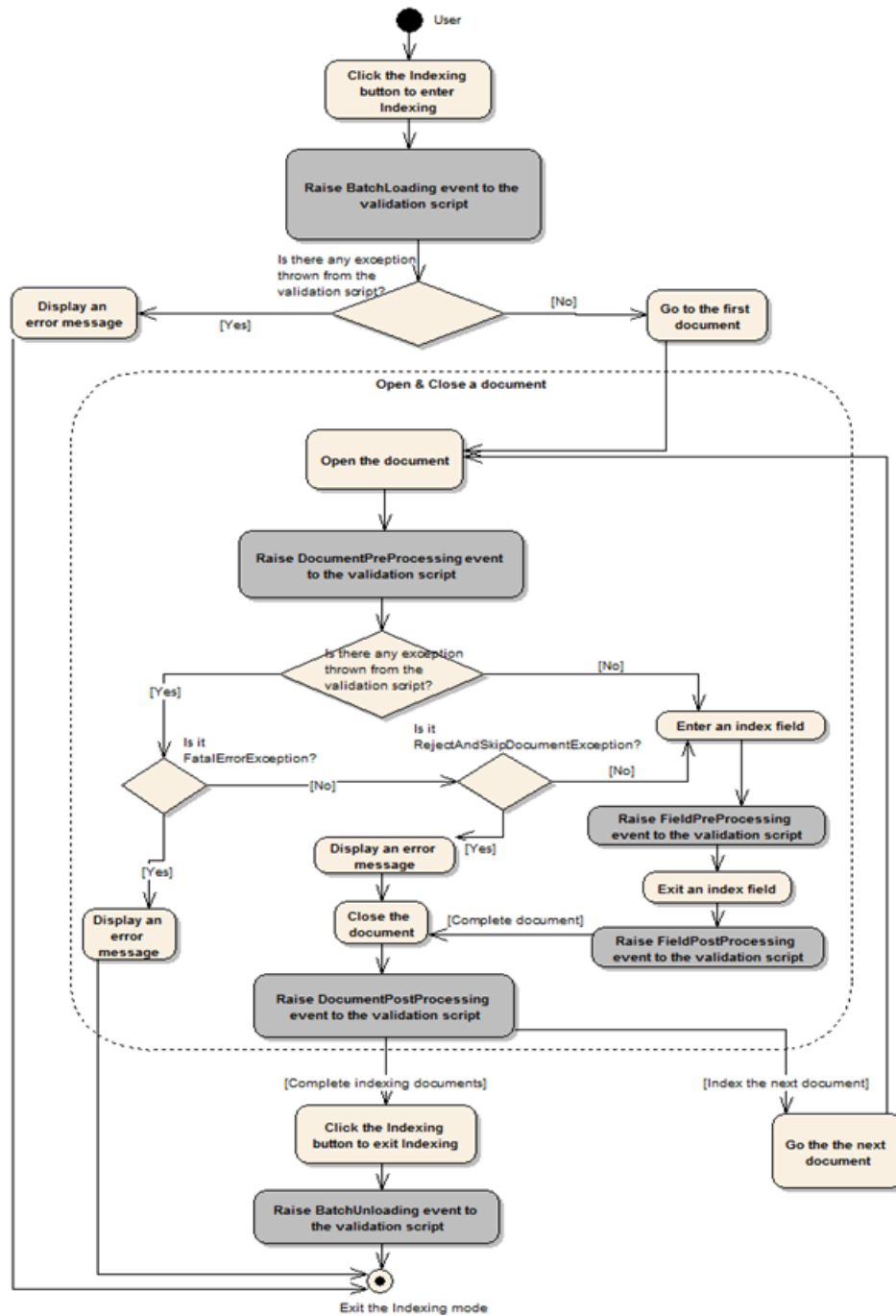
## Chapter 5

# Document Level Validation Script Execution

Refer to the illustration to see the basic flow for document level validation script execution.

By default, calculation of values for index fields is performed as in earlier versions of Kofax Express. However, when index values are updated by the validation script, they are used to fill in index fields when entering a document in indexing mode.

Kofax Capture allows a validation script to update index field settings (if their access type is read-write). In this case, the index field controls are also updated accordingly. The same behavior applies to Kofax Express. For example, if the ReadOnly setting of an index field is False prior to the DocumentPreProcessing call, it is updated to True by the script; then the index field control is shown as read-only. The change only affects indexing mode, and it does not change the global setting for the index field in the application.



**Basic Flow - Document Level Validation Script Execution**

Script Storage

The Kofax Express scripts use the folder hierarchy described here.



### Standalone

For standalone installations, the .NET script for each job is located here:

```
C:\ProgramData\Kofax\Kofax Express 3.3\Scripts\<<ScriptID>\<ProjectName>
```

### Server

On a server, the .NET script for each job is located here:

```
<Server Database Path>\Scripts\<<ScriptID>\<ProjectName> folder
```

### Client

On a client workstation, the .NET script for each job is located here:

```
C:\ProgramData\Kofax\Kofax Express 3.3\Scripts\<<ScriptID>\<ProjectName>
```



<ScriptID> is the unique ID given to the current script.

<ProjectName> is the name of the job.

## Sample Document Validation Script

This section shows a sample Kofax Express document validation script that is also supported for use with Kofax Capture.

The sample script assumes the following:

- Job name is Catalog Orders
- Job includes five index fields: Unit Price, Quantity, Subtotal, Item Number, and Description
- Script requires numeric values for Unit Price, Quantity, and Subtotal
- Script requires that the Subtotal be equal to the Unit Price multiplied by the Quantity

### Sample Script

```
.....  
' Copyright 2012, Kofax, Inc. All rights reserved.  
' Unauthorized use, duplication or distribution is strictly prohibited.  
.....  
Imports Kofax.Eclipse.Scripting  
Imports Kofax.EclipseModule  
  
Namespace Catalog_Orders  
  
    ' This is a simple example implementation class that performs form validation.  
    Public Class Catalog_Orders  
        Inherits DocumentValidationScript  
  
        Dim customers As Hashtable  
        Dim catalogItems As Hashtable
```

```
<IndexFieldVariableAttribute("Item Number")> _
Dim WithEvents Item_Number As FieldScript

<IndexFieldVariableAttribute("Description")> _
Dim WithEvents Description As FieldScript

<IndexFieldVariableAttribute("Unit Price")> _
Dim WithEvents Unit_Price As FieldScript

<IndexFieldVariableAttribute("Quantity")> _
Dim WithEvents Quantity As FieldScript

<IndexFieldVariableAttribute("Subtotal")> _
Dim WithEvents Subtotal As FieldScript

' In addition to initializing the sample data.
Public Sub New()
    ' Init test data
    InitRecords()
End Sub

Private Sub Catalog_Orders_BatchLoading( _
    ByVal sender As Object, _
    ByVal e As BatchEventArgs) _
    Handles Me.BatchLoading

    ' Throw a fatal error exception if the indexing batch does not belong to a
specific job
    If Not e.Batch.ClassName = "Catalog Orders" Then
        Throw New FatalErrorException(String.Format("The batch does not belong
to {0}.", "Catalog Orders"))
    End If

End Sub

Private Sub Catalog_Orders_DocumentPreProcessing( _
    ByVal sender As Object, _
    ByVal e As PreDocumentEventArgs) _
    Handles Me.DocumentPreProcessing

    ' Throw a fatal error exception if any index field is NULL
    If IsDBNull(Me.Item_Number.IndexField) Or
IsDBNull(Me.Description.IndexField) Or _
        IsDBNull(Me.Unit_Price.IndexField) Or IsDBNull(Me.Quantity.IndexField)
Or _
        IsDBNull(Me.Subtotal.IndexField) Then
        Throw New FatalErrorException("The batch contains index fields that are
missing or nonexistent data.")
    End If

End Sub

' This FieldPostProcessing procedure will be called whenever the Item Number
field is
' exited on the client. In this example the items number is used to fetch the
item
' record from the catalog and populate the item's description and price. If the
item is
' not found the field value is considered invalid and a
ValidationException to
' indicate this to the client.
```

```
Private Sub Item_Number_FieldPostProcessing( _
    ByVal sender As Object, _
    ByVal e As PostFieldEventArgs) _
    Handles Item_Number.FieldPostProcessing

    If catalogItems.ContainsKey(Me.Item_Number.IndexField.Value) Then
        Dim item As CatalogItem =
catalogItems.Item(Me.Item_Number.IndexField.Value)
        Me.Description.IndexField.Value = item.Description
        Me.Unit_Price.IndexField.Value = item.Price.ToString()
    Else
        Throw New ValidationErrorException("Item was not found in catalog.",
Me.Item_Number.IndexField)
    End If
End Sub

' This DocumentPostProcessing procedure will be called when the client tries to
submit
' the completed form. In this example validate that the unit price times the
amount
' equals the subtotal.
Private Sub Catalog_Orders_DocumentPostProcessing( _
    ByVal sender As Object, _
    ByVal e As PostDocumentEventArgs) _
    Handles Me.DocumentPostProcessing

    Dim parseResult As Boolean
    Dim unitPrice As Double
    Dim quantity As Integer

    parseResult = Double.TryParse(Me.Unit_Price.IndexField.Value, unitPrice)
    If Not parseResult Then
        Throw New ValidationErrorException("Unit Price is not a valid number.",
Me.Unit_Price.IndexField)
    End If
    parseResult = Integer.TryParse(Me.Quantity.IndexField.Value, quantity)
    If Not parseResult Then
        Throw New ValidationErrorException("Quantity is not a valid number.",
Me.Quantity.IndexField)
    End If

    Dim unitPriceTimesQuantity As Double = unitPrice * quantity
    Dim subtotal As Double
    parseResult = Double.TryParse(Me.Subtotal.IndexField.Value, subtotal)
    If Not parseResult Then
        Throw New ValidationErrorException("Subtotal is not a valid number.",
Me.Subtotal.IndexField)
    End If
    If Not Double.Parse(unitPriceTimesQuantity) = Double.Parse(subtotal) Then
        Throw New ValidationErrorException("Subtotal does not equal unit price
times quantity.", Me.Subtotal.IndexField)
    End If

End Sub

' Setup repository of test data representing customers and catalog items
Private Sub InitRecords()
    customers = New Hashtable()
    customers.Add("Bill Slater", New Customer("Bill Slater", "007846",
"Martin", "Janeway"))
    customers.Add("Janet Harvey", New Customer("Janet Harvey", "004018",
"Barb", "Harvey"))
    customers.Add("Bob Thompson", New Customer("Bob Thompson", "003310", "Kay",
"Brummel"))
```

```
        customers.Add("Art Hollingsworth", New Customer("Art Hollingsworth",
"007846", "Fred", "Wong"))

        catalogItems = New Hashtable()
        catalogItems.Add("638", New CatalogItem("638", "Lang Sterling Pie Server",
32.95))
        catalogItems.Add("4303", New CatalogItem("4303", "Flap Bags", 19.95))
        catalogItems.Add("5208", New CatalogItem("5208", "Stemware Protector",
19.95))
        catalogItems.Add("4308", New CatalogItem("4308", "Silvercare Package",
14.95))
    End Sub

    ' Data structure to represent a test customer
    Private Structure Customer
        Dim Name As String
        Dim SourceNumber As String
        Dim ShipToFirstName As String
        Dim ShipToLastName As String

        Sub New(ByVal Name As String, _
            ByVal SourceNumber As String, _
            ByVal ShipToFirstName As String, _
            ByVal ShipToLastName As String)
            Me.Name = Name
            Me.SourceNumber = SourceNumber
            Me.ShipToFirstName = ShipToFirstName
            Me.ShipToLastName = ShipToLastName
        End Sub
    End Structure

    ' Data structure to represent a test item in the catalog
    Private Structure CatalogItem
        Dim ItemNumber As String
        Dim Description As String
        Dim Price As Double

        Sub New(ByVal ItemNumber As String, _
            ByVal Description As String, _
            ByVal Price As Double)
            Me.ItemNumber = ItemNumber
            Me.Description = Description
            Me.Price = Price
        End Sub
    End Structure

End Class

End Namespace
```

## Chapter 6

# Export Connectors

*Export* is the process of exporting images and data to long-term storage after Kofax Express processing is finished. An *export connector* consists of two COM components that configure and execute this process. These two components could be in one or two `.dll` or `.exe` files.

Kofax Express includes the following standard export connectors:

- **Multipage:** Exports batches to a folder containing one file per document. Each file contains all the images that belong to a document.
- **SharePoint:** Exports batches to a SharePoint repository.  
Additional SharePoint export connectors are available from Kofax.
- **Single page:** Exports batches to a folder containing one file per scanned image.
- **Kofax Capture:** Exports batches to an XML auto import file for use in Kofax Capture.
- **Database:** Exports document index data to a Microsoft Access or ODBC-compliant database. Source code for this export connector, provided in Microsoft Visual Basic .NET, is available from your installation media in the `KEC-Database\Source` folder.  
Use this connector to export images and PDF files to the standard file system.

If you need to export document index data or files to other sources, you can modify one of the supplied connectors, create an entirely new connector, or purchase one from Kofax or other third-party entities.

Kofax export connectors are available for industry-standard applications developed by SharePoint, Documentum, IBM FileNet, and others. Contact your Certified Solution Provider (CSP) for details on availability.

Export connectors are typically written in Visual Basic .NET, but they can also be written with any tool that supports the development of COM servers.

## Export Type Library

Kofax Express and Kofax Capture share the same export type library, as documented in the *Kofax Capture API Reference Guide* provided in your product installation files.

1. In the SDK folder on your Kofax Express installation media, extract `KofaxExpressSDK.zip` and double-click `APIRef.chm`.  
The *Kofax Capture API Reference Guide* appears.
2. On the Contents tab, select and expand **Kofax Capture Export Type Library** in the table of contents.

**i** If the *API Reference* does not display properly, copy it to a local drive.

## Kofax Express and the Export Process

Kofax Express manages the export setup process. It loads the export connector setup when an export connector is selected for a job. The export connector setup must implement the `KfxReleaseSetupScript` COM interface required by Kofax Express.

Similarly, Kofax Express manages the export process. It loads the appropriate export connector when the batch starts the export process. The export connector must implement the `KfxReleaseScript` COM interface required by Kofax Express.

When writing an export connector, there are two basic functions you must provide:

- Export connector setup, which is the user interface for configuring the export process. The setup for the export connector is called by Kofax Express when the user selects an export connector to configure on the Job Setup tab.
- Export, which performs the export process that sends images and data to long-term storage. The export for the export connector is called by the export process when it is exporting batches.

## Export Connector Setup

There are two places in the export connector setup where you have to write a substantial amount of code:

- The `RunUI` event should load a form that presents a user interface. This form can be as simple or as complex as you wish.
- When the user clicks OK to finish export connector setup, you must save the user's settings. To do this, you must set up a links collection that specifies which index fields should be exported and then copy other settings as necessary into the `ReleaseSetupData` object. When this is finished, call the `Apply` method to save your changes.

### ReleaseSetupData Object

The `ReleaseSetupData` object is a top-level object used by both Kofax Express and the export connector setup to define the export process.

Some properties of this object are set up by Kofax Express when the `ReleaseSetupData` object is created. For example, the available index fields, image types, and storage types are set when the connector is loaded. The properties set up by Kofax Express are read-only.

The connector must set the properties that identify the external data repository and the target export folders. It must also set the properties that establish the links between the available data fields and the fields or columns in the external data repository that will receive the document data during the subsequent export process. The properties will be available to the export connector.

Refer to the *Kofax Capture API Reference Guide* for details on the specific properties and methods available in the `ReleaseSetupData` object.

## Requirements for the Export Connector Setup

The export connector setup component must define the KfxReleaseSetupScript COM interface.

The KfxReleaseSetupScript interface must include the following:

- One public object variable declared as ReleaseSetupData, which is used to expose and update export configuration for the document class.
- Four methods (OpenScript, RunUI, ActionEvent, and CloseScript), which are called to perform the various stages of the export setup process

When you configure an export connector the first time, Kofax Express does the following:

1. Fills in the document properties in the ReleaseSetupData variable. These properties include the available index fields and image file formats.
2. Calls the OpenScript method, which is used to perform any initialization required for the connector.
3. Calls the RunUI method. Use this method to load a form or window that allows the user to configure the export process. For example, the database export connector displays a form that allows setup of database links, image formats, file folders, and so forth. When the user is finished with setup, you store the user's selections in the ReleaseSetupData variable and call the ReleaseSetupData.Apply method to save them permanently.
4. Calls the CloseScript method after the RunUI method is completed and the connector is about to be unloaded. Use this method to perform any cleanup required for the connector.

If you change job settings after the initial export setup process, Kofax Express performs the same series of steps as before, with one exception: instead of calling the RunUI method, it calls the ActionEvent method with a set of parameters that indicate why it is being called. Use this method to determine whether a change in the export setup process is required. For example, if a new index field is added, you might want to call the RunUI method to provide an opportunity to save this new data in the external database.

## Export Setup Data Object Interface - ReleaseSetupData

As with Kofax Capture, the export setup data object Interface is used in Kofax Express to define export process for batches in a job. The ReleaseSetupData properties provide configuration information such as target export folders and the external data repository.

Property/Method	Description
BatchClassID	Returns the job ID, which is a unique number assigned to every job that Kofax Express creates or upgrades from a previous installation. Kofax Express is generates an incrementally unique number for standalone or client/server installations.
ExternalBatchClassID	Returns job ID (same value as for the preceding BatchClassID property).
BatchClassName	Returns job name.

Property/Method	Description
BatchClassDescription	Returns empty string because Kofax Express doesn't have a job description.
BatchClassSupportsNonImageFiles	Kofax Capture supports native storage of PDF, Word, Excel, or other non-image files (known as "eDocs"). Because Kofax Express does not support eDocs, it always returns False.
DocClassID	Returns job ID.
DocClassName	Returns job name.
_StartTime	This getter/setter property is kept in the API but not used in Kofax Express. The initial value is 00:00:00. Changes made to this property by export connectors are saved.
_EndTime	This getter/setter property is kept in the API but not used in Kofax Express. The initial value is 23:59:00. Changes made to this property by export connectors are saved.
_Priority	This getter/setter property is kept in the API but not used in Kofax Express, which does not support the concept of priority for a job or batch. The initial value is 0. Changes made to this property by export connectors are saved.
UserName Password ConnectionString	These properties are unchanged in Kofax Express. However, since Kofax Express saves its database to files on disk, this sensitive information is encrypted (FIPS compliant) in the database.
RowSeparator	Always returns ";".
SkipFirstPage	Returns the corresponding value.
BatchFields	Kofax Express doesn't support the BatchField concept; always returns an empty collection of BatchFields.
BatchVariableNames	Returns the list of default Kofax Values. See <a href="#">Default Values in Kofax Export Connectors</a> .
ImageTypes	Returns a list of supported image formats: <ul style="list-style-type: none"> <li>• Multipage TIFF - Kofax</li> <li>• Multipage TIFF - LZW Compression</li> <li>• Multipage TIFF - Technical Note 2</li> <li>• Multipage TIFF - Uncompressed</li> <li>• TIFF - Kofax</li> <li>• TIFF - LZW Compression</li> <li>• TIFF - Technical Note 2</li> <li>• TIFF - Uncompressed</li> <li>• JPG - JPEG Compression</li> <li>• BMP - Bitmap Image</li> </ul>



Property/Method	Description
TextFileEnabled	In Kofax Capture, this is True if OCR is enabled (exporting to Word, Excel and Text). In Kofax Express, this is not supported and always returns False.
KofaxPDFDocClassEnabled	Returns True if PDF output is selected for the job.
ServicePackVersion	Returns the service pack version of Kofax Express. This value is the third field in the 6-field version number stored in <code>KofaxReg.xml</code> . Returns 0 if the number is missing.
LogError	Sends error to Kofax Express log file for debugging purposes when "Log details" is selected on the Options window.
Apply()	Saves all setup data to the Kofax Express database for job settings.
New	Returns True if the export connector has never been launched and saved (apply is called) previously with this job.

## IndexField

IndexField consists of the IndexSetup information from the job setup in Kofax Express.

Property/Method	Description
Name	Returns the Index Field Label string that uniquely identifies a job index field in the IndexFields collection.
Type	Returns a <code>KfxIndexFieldType</code> value that indicates the data type index field for the job. Value is based on the index field's validation settings.
FieldID	Returns the unique numeric ID of the index field in the IndexFields collection for the job.
FieldTypeName	Not supported in Kofax Express. Always returns the index field label string.
FieldTypeDescription	Not supported in Kofax Express. Always returns an empty string.
FieldTypeLength	Returns a number that indicates the length of the index field type. This value is based on the index field's validation settings.
FieldTypePrecision	Value of "number digits" for the numeric data type in Kofax Capture. In Kofax Express, this is inferred from the validation mask for the index field. For the numeric type, it is either the value inferred from the mask, such as return 5 for the validation mask 9(5), or If the precision cannot be inferred from the mask, it is 0. For other data types, it returns -1.

Property/Method	Description
FieldTypeScale	The value of "decimal places" for numeric data type in Kofax Capture. In Kofax Express, this always returns 0 for numeric type and -1 for a non-numeric type.
TableName	Not supported in Kofax Express. Always returns an empty string.
FolderClassName	Not supported in Kofax Express. Always returns an empty string.
FolderIndexFieldName	Not supported in Kofax Express. Always returns an empty string.

## BatchFields

Kofax Express does not support the following:

- **BatchFields:** Although Kofax Express doesn't support a BatchFields concept, the Export Setup Data object still needs to contain an empty collection of BatchFields. Thus, the implementation of this BatchFields collection is unchanged from Kofax Capture.
- **BatchField:** This interface should never be used; if it is used, an exception occurs.

## Export Connector Development

The ReleaseDoc method is called every time Kofax Express is ready to export a new document. To export a document, you need to add code to this method to call the following methods:

1. **ReleaseData.ImageFiles.Copy:** Copies the images to a location specified by the ReleaseData object.
2. **ReleaseData.CopyKofaxPDFFile:** Copies the PDF file that belongs to a document into the export PDF path that is defined during export connector setup.

## ReleaseData Object

The ReleaseData object is a top-level object used by both Kofax Express and the export connector to access the batch information and perform the export operation.

All of the properties of this object are set up by Kofax Express. The properties that are common to all documents in the batch are set up when the ReleaseData object is initialized. Properties specific to an individual document are set up before the Kofax Express calls the ReleaseDoc method for that document.

The connector must use the ReleaseDoc method to read the properties and save the index data and related document information in the external data repository. Then, the connector must copy the image files, any Kofax PDF files to the target file system.

Refer to the *Kofax Capture API Reference Guide* for details on the specific properties and methods available in the ReleaseData object.

## Requirements for the Export Connector

The Export process uses a COM interface called KfxReleaseScript to communicate with the export connector.

The KfxReleaseScript interface must include the following:

- One public object variable declared as ReleaseData. The Export process uses this variable to expose export data.
- Three methods called OpenScript, ReleaseDoc, and CloseScript, which are called to perform various stages of the export process.

When a batch is ready for export processing, the export connector does the following:

1. Fills in the general batch properties in the ReleaseData variable.
2. Calls the OpenScript method. You should use this method to perform any initialization required for the connector.
3. Fills in the properties specific to the first document to be exported in the ReleaseData variable and calls the ReleaseDoc method for the document. Use this method to save the document data in the external database and copy the image files to the selected export folders.
4. Repeats the process for each remaining document to be exported.
5. Calls the CloseScript method after the last ReleaseDoc method is completed and the connector is about to be unloaded. You should use this method to perform any cleanup required for the connector.

## Export Data Object Interface - ReleaseData

ReleaseData is used by the export process and export connector to access the batch and document information, and to perform a document export. Use the ReleaseData object properties in your ReleaseDoc method to read and save data to a data repository.

Property/Method	Description
BatchClassID	Returns the job ID.
ExternalBatchClassID	Returns job ID.
BatchClassName	Returns job name.
BatchClassDescription	Returns empty string because Kofax Express doesn't have job description.
UserName Password ConnectionString	Returns the username/password/connection string, if any, provided during the export connector setup. These properties are unchanged in Kofax Express. However, since Kofax Express saves its database to files on disk, this sensitive information is encrypted (FIPS compliant) in the database.
RowSeparator	Always returns a semicolon (;).
UniqueDocumentID	Gets a unique number assigned to the document.

Property/Method	Description
Values	At export time, the calling application resolves all the links set up in the job file and computes the appropriate values for them. This typically involves copying the value of an index field, but may also involve converting Kofax Values. This is unchanged in Kofax Express. However, the links are resolved to proper Kofax Express values as indicated in <a href="#">Default Values</a> .
TextFiles	Kofax Express does not support OCR output. This is always an empty collection.
ImageFiles	Unchanged from Kofax Capture. For Kofax Express, contains a list of image files for the document being exporting.
Released	Always returns False.
KofaxPDFFileName	Returns the name of the PDF file.
DocumentCustomStorageString	As with Kofax Capture, during export an export connector can save/load custom key-value pair strings to a Kofax Express document. When the batch is saved, these strings are also saved to batch storage.
RepositoryDocumentID	Used for passing data between multiple export connectors for a given document. Kofax Express doesn't support this feature, so it always returns an empty string. However, the value set by the connector is still saved to the batch data via the DocumentCustomStorageString collection with a hardcode key (Kofax.AC.RepositoryDocumentID). This makes it possible to retrieve the value if support for multiple export connectors is added to Kofax Express in the future.
ServicePackVersion	Returns the service pack version of Kofax Express. This value is the third field in the 6-field version number stored in <code>KofaxReg.xml</code> . Returns 0 if that number is missing.
LogError	Sends error to Kofax Express log file for debugging purposes when "Log details" is selected on the Options window.

## Value

Value represents an individual data point in the Values collection of a ReleaseData object. Value identifies the data field (source) and the destination column or field in the external data repository.

Property/Method	Description
DataType	Inferred from the validation mask in Kofax Express. See <a href="#">Index Field Mapping</a> .

Property/Method	Description
TableName	Kofax Express does not support tables; always returns an empty string.
RowValue	Kofax Express does not support tables; always returns an empty string.
RowCount	Kofax Express does not support tables; always returns 0.
RowValues	Kofax Express does not support tables; always returns Null.

## ImageFile

ImageFile represents a single image (page) of a document in the ImageFiles collection. ImageFile contains the method for copying an image to the export folder. The Export process initializes the ImageFiles collection with data before calling ReleaseDoc.

Property/Method	Description
Copy	In client/server mode, extracts the image from the server and copies it to a local folder.
RawOriginalFileName	Returns the file name (without extension and path) of the imported file.

## Other Export Connector API Objects

- TextFiles: Intended for full text output (OCR output enabled), which Kofax Express does not support. These objects are implemented but have all methods/properties throw an exception.
- Tables: Kofax Express doesn't support tables. As indicated in the Value interface, it returns an empty collection of IableRows.
  - TableRows: Contained in Tables, which is always be empty collection. Inaccessible to the connector.
  - TableRowValues: Contained in TableRows and inaccessible to the connector.
  - RowValues: Value always returns NULL for RowValues. Inaccessible to the connector.

## Action Events

Kofax Express supports the action events listed in the table.

Event	Description
KFX_REL_INDEXFIELD_DELETE	<p>If a Kofax export connector is assigned to a Kofax Express job, this event is fired when the user clicks Delete in the Index Fields group on the Index Setup tab. The strData1 parameter contains the name of the index field to be deleted.</p> <p>A Kofax export connector interface may appear if a required field in the destination system is no longer linked to an index field.</p>
KFX_REL_INDEXFIELD_RENAME	<p>Fired when the user renames an index field. Most Kofax export connectors update internal data and do not appear in this situation.</p> <ul style="list-style-type: none"> <li>• The strData1 parameter contains the name of the original index field.</li> <li>• The strData2 parameter contains the name of the new index field.</li> </ul>
KFX_REL_INDEXFIELD_INSERT	<p>Adds a new index field. Most Kofax export connectors appear in this case. However, this would be a problem in Kofax Express since its user interface for entering index field is modeless. Kofax Express must send this event for every newly created index field. This event is never sent to a Kofax export connector.</p>
KFX_REL_DOCCLASS_RENAME KFX_REL_BATCHCLASS_RENAME	<p>Both of these events are fired in succession when the Kofax Express job name is changed. This is relatively unlikely as changes to job names are not allowed if the job contains batches. Most Kofax export connector interfaces are unlikely to appear in this case.</p> <ul style="list-style-type: none"> <li>• The strData1 parameter contains the name of the original job.</li> <li>• The strData2 parameter contains the name of the new job.</li> </ul>

Event	Description
KfxActionValue.KFX_REL_RELEASESETUP_DELETE	Fired when the user selects a different export connector for the job (whether it be another Kofax connector or a Kofax Express connector). Unlike Kofax Express connectors, which retain their settings even if not selected, only zero or one Kofax export connector configuration can be saved per job. Otherwise, Kofax Express would have to send events to many unselected export connectors, which would likely display configuration settings during job changes and cause confusion.

## Default Values in Kofax Export Connectors

The table lists default values or Kofax Values that Kofax Express returns to Kofax export connectors.

Default Values	Description
Job Name	Job name
Batch Name	Batch name
User Name	Current logged in user
Station ID	Current machine name
Document Number	Order of a document in the current batch
Document Number With Zeros[8]	The order with leading zeros
First Page Original File Name	Original file name of the first page when imported. Example: 00000001.tif
Document Count	Number of documents in the batch
Current Date	Current system date in current locale format
Current Time	Current system time in current locale format
UTC Offset	UTC offset of the current time zone. For example: (+7:00)
Batch Creation Date	Batch creation date in current locale format
Batch Creation Time	Batch creation time in current locale format

## COM Servers: In-proc or Out-of-proc?

Export connectors can be developed using any language and tool that can create COM components. If you are working with a 32-bit development environment, COM components for both export connector setup and export can be designed as in-process servers (ActiveX DLLs) or as out-of-process servers (ActiveX EXEs).

## Export Connector Registration

After completing your export connector, you must set up a registration `.inf` file.

The `.inf` file, which should be located in the same folder as your compiled export connector, has a format similar to that of an `.ini` file. A sample `.inf` file is shown here.

```
[Scripts]
Custom Script
[Custom Script]
SetupModule=Custom.dll
SetupProgID=Custom.kfxreleasesetupscript
SetupVersion=1.0
ReleaseModule=Custom.dll
ReleaseProgID=Custom.kfxreleasescript
ReleaseVersion=1.0
SupportsNonImageFiles=True
RemainLoaded=True
SupportsKofaxPDF=True
SupportsOriginalFileName=True
SupportsMultipleInstances=False
```

The `.inf` file must contain a `[Scripts]` section that includes the name (up to 255 characters) of the connector you are registering. For each entry in the `[Scripts]` section, there must be a corresponding section with the entries listed in the table.

### .Inf File Entries

.Inf File Entry	Description
ReleaseModule	Name of the compiled <code>.exe</code> or <code>.dll</code> containing the export COM component.
ReleaseProgID	Name of the export connector COM component. You can explicitly set the ProgID of your object using the ProgID Attribute. A ProgID is a unique identifier for every control, constructed through the combination of the control's project and object name (ProjectName.ObjectName).
ReleaseVersion	Version number assigned to the export connector COM component.
RemainLoaded	If True, the export connector will remain loaded until processing is complete.
SetupModule	Name of the compiled <code>.exe</code> or <code>.dll</code> containing the export connector setup COM component.
SetupProgID	Name of the export connector setup COM component. You can explicitly set the ProgID of your object using the ProgID Attribute. A ProgID is a unique identifier for every control, constructed through the combination of the control's project and object name (ProjectName.ObjectName).
SetupVersion	Version number assigned to the export connector setup COM component.



.Inf File Entry	Description
SupportsKofaxPDF	If True, the export connector supports the output of Kofax PDF files.
SupportsNonImageFiles	If True, the export connector supports eDocuments (non-image files).
SupportsOriginalFileName	If True, the export connector supports the use of the original name of the file.
SupportsMultipleInstances	If True, the export connector supports multiple instances of the Export service. Not supported with Kofax Express.

For export connector registration guidelines, see the *Kofax Express Installation Guide*.

## Differences Between Kofax Express and Kofax Capture Export Connectors

If you create an export connector for use with Kofax Capture, it is possible to use it with Kofax Express. Be sure to test the export connector carefully to ensure that it works properly, and take into account the following considerations:

- If the export connector has dependencies on Kofax Capture licensing or encryption libraries, they may not be available in a Kofax Express environment.
- If the export connector has dependencies on Kofax Capture registry entries, they may conflict with certain Kofax Express registry entries. The Kofax Express installation includes its own Kofax Capture registry entries that allow an export connector to function outside a Kofax Capture environment.

The following table outlines key differences between Kofax Express and Kofax Capture export connectors.

Feature	Kofax Express	Kofax Capture
Export a single batch to multiple export connectors	No	Yes
Upgrade to new export connectors	No; when switching from the default Kofax Express SharePoint connector to a Kofax connector, you must reconfigure the settings	Yes
Localized export connectors	No, English only	Yes for the Text and Database connectors; otherwise, English only
Support for custom image formats, such as the following for SharePoint: <ul style="list-style-type: none"> <li>• Multipage TIFF - G3</li> <li>• Multipage TIFF - G3/2D</li> <li>• Kofax PDF</li> </ul>	No, configuration is possible but an unsupported format may produce an error during export	Yes