floatingDataExtract Setup Guide

Code-Free Floating Data Extraction Utility for Kofax Capture

Contents

1 Overview

This guide is intended to show how to install and configure the **floatingDataExtract** workflow agent for Kofax Capture.

It assumes a working knowledge of Kofax Capture batch class setup and an understanding of the standard Kofax workflow process.

floatingDataExtract is supported for Kofax Capture version 9, 10, and 11.

2 Installation

To install **floatingDataExtract** follow the steps below:

-Unzip the **floatingDataExtract.zip** file and copy **floatingDataExtract_Setup.dll, floatingDataExtract_Runtime.dll** and **floatingDataExtract.AEX** to the Kofax Capture Install Path …\Bin folder

-In the Administration module, select Tools>Workflow Agent Manager and Add

-Browse to the **floatingDataExtract.AEX** file and select it

-Select **floatingDataExtract** and Install

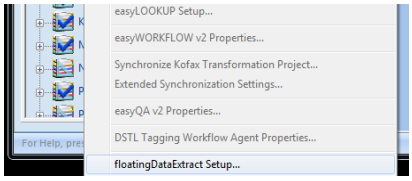
floatingDataExtract should now be installed and ready for use.

3 Configuration

3.1 Adding floatingDataExtract to a Batch Class

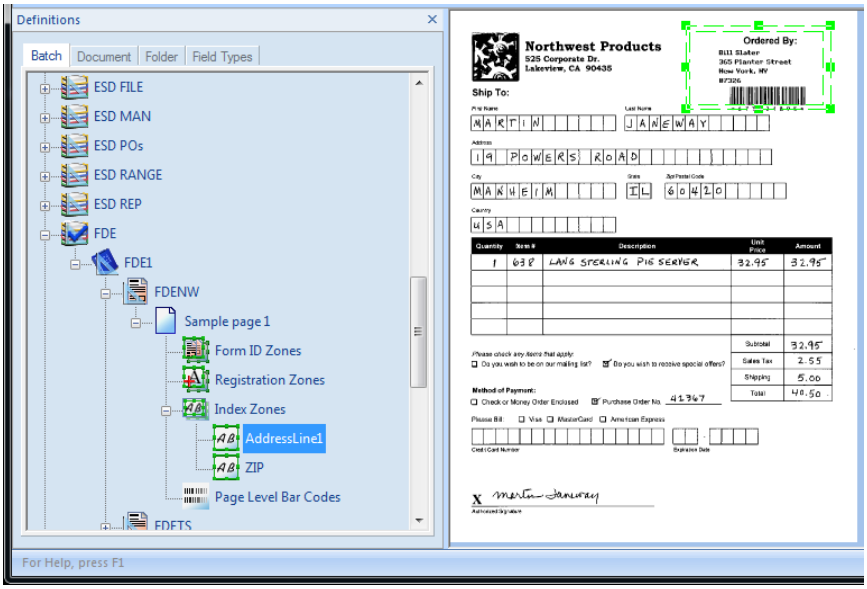
To add **floatingDataExtract** to a Batch Class, select the Workflow Agents tab from the Batch Class properties, and add **floatingDataExtract** to the Selected Workflow Agents list.

Once **floatingDataExtract** has been added to a Batch Class, right-clicking on the Batch Class will show the **floatingDataExtract** Setup menu item.

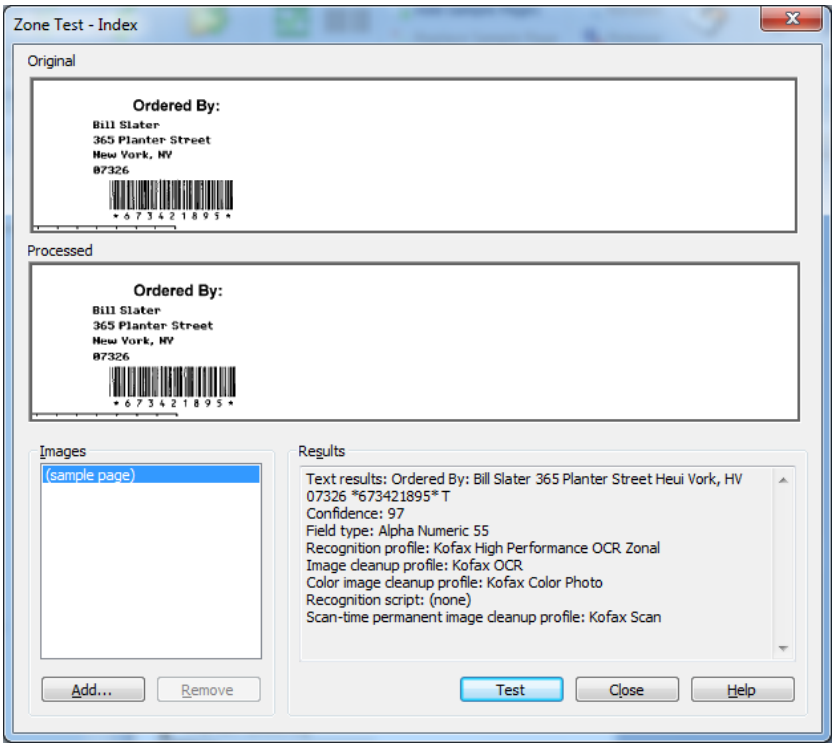


3.2 Settings

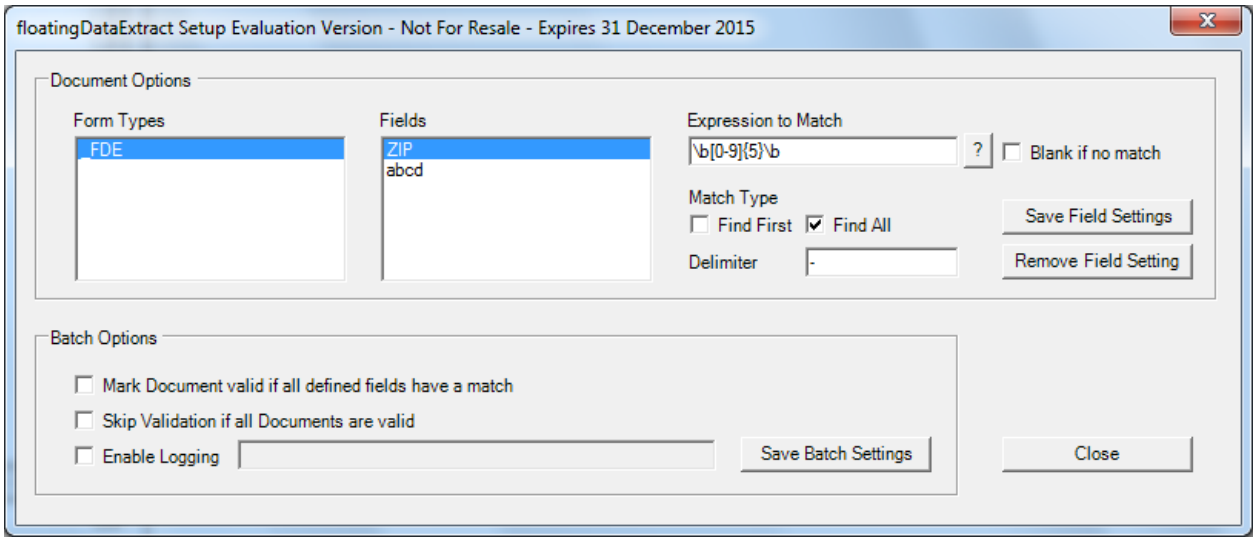
To use **floatingDataExtract** processing for an Index Field, you first configure the Index Field Recognition Profile as you would normally, but draw an area large enough to cover the area where the value is expected to be found. In the example below we are configuring a zone to find Address Line 1.



Test the zone to make sure that all the relevant data will be read correctly.



The section below details how to configure **floatingDataExtract** to extract the relevant data from this larger amount of raw data from the configured Index Zone.



3.2.1 Document Options

For each Form Type and Index Field combination, you can configure settings to determine what data should be extracted from the larger data zone which has been defined for that Index Field in Administration.

3.2.1.1 Expression to Match

Enter a regular expression which defines the data you want to extract. See *Section 5 – Regular Expressions* for more detail on how to define these.

In the example above the expression \b[0-9]{5}\b is used. This will search for 5 digit blocks which form whole words and not parts of words.

\b	word boundary
[0-9]	character set
{5}	number of characters to search
\b	word boundary

For the Address Line 1 field we could use something like:

\b[0-9]{1,3}\s[A-Z,a-z]{2,25}\s[A-Z,a-z]{2,25}\b

This would search for:

\b	word boundary
[0-9]{1,3}	1-3 numbers
\s	space
[A-Z,a-z]{2,25}	2-25 letters
\s	space
[A-Z,a-z]{2,25}	2-25 letters
\b	word boundary

This would match '123 High Street' for example, but not 'Manchester'

Click the **?** to the right of Regular Expression to open a Regular Expression Builder utility. See Section *3.3 Regular Expression Builder* for more details.

3.2.1.2 Match Type and Delimiter

It may be the case that the larger area contains more than one matching piece of data. In this case you can select **Find All** and optionally enter a delimiter.

For example if you used the expression above for 5 digit whole words, selected **Find All** and entered '#' as delimiter, and the large data zone had the following:

> *Ordered By: Bob Thompson 956 South Street San Diego, Cn 92758 Tri-Spectrum Solutions 21 Tylor St., Suite 100 New York, NY 02395*

The resulting match would be '92758#02395'

3.2.1.3 Blank if no Match

Select Blank if no Match to remove all data from the Index Field if no matches for the regular expression are found.

3.2.1.4 Save/Remove Field Settings

For each Form Type and Index Field pair, you need to click Save Field Settings to store the new/updated settings, or Remove Field Settings to clear them.

To make the changes live, you must then re-publish the batch class.

3.2.2 Batch Options

Mark document valid if all defined fields have a match

> Select this option if the document should be marked Valid if each of the fields for which there are floatingDataExtract settings find a match.

Skip Validation if all Documents are Valid

> If all documents in the batch have been marked as valid, skip the Validation module for the entire batch.

Enable Logging

> Enable logging, and select a folder for the log files to be written. This will produce quite verbose logging so if you process high volumes, you may want to only use this during testing phases.

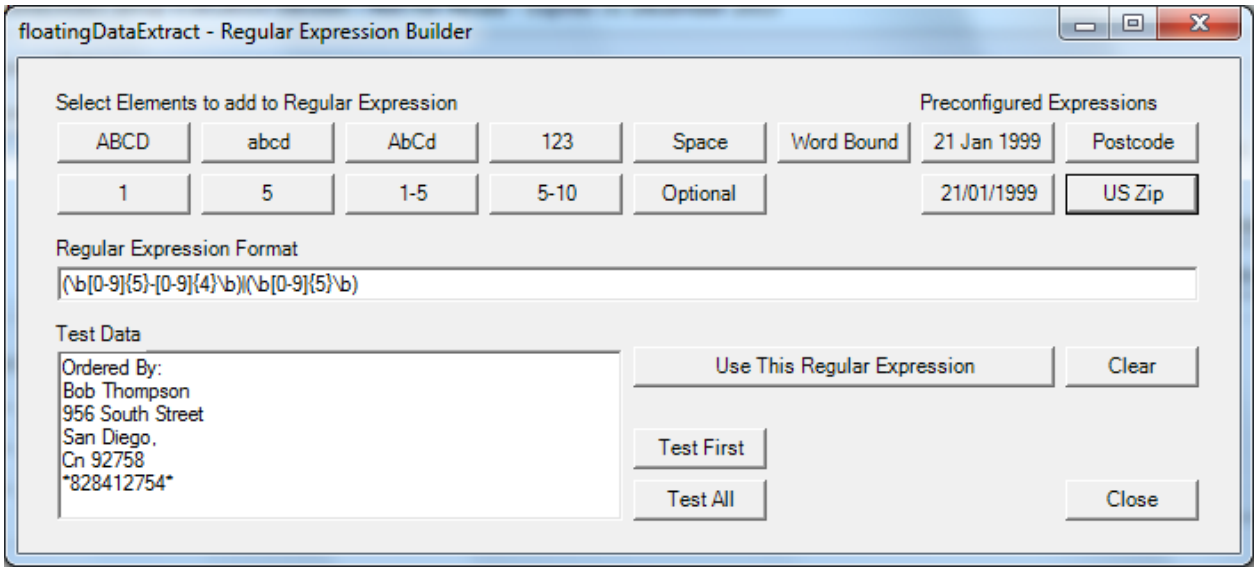
Save Batch Settings

> Click Save Batch Settings to update the current configuration. These settings are ignored if no Index Field options are defined.

3.3 Regular Expression Builder

A Regular Expression Builder is available to help design the expressions required to extract data.

Clicking Elements will add them to the Regular Expression, and they can be edited to your specific requirements. Or you can use preconfigured expressions.



Preconfigured Expressions

21 Jan 1999	day part	1 / 01 / 1st
	month part	Jan / January
	year part	99 / 1999
21/01/1999	day part	1 / 01
	month part	01
	year part	99 / 1999
	delimiter	/\,.- <space>
Postcode	UK Postcode formats	
US ZIP	US Zip Code formats	

Use the Test First or Test All options to test the current expression against the text in the Test Data area.

Click Use This Expression to set the current expression as the expression for the currently selected Form Type and Index Field.

4 General Considerations

4.1 Skip If Cannot Load

An option exists in Kofax workflow agents to skip if not loaded. This option can be set to TRUE or FALSE.

If set to TRUE, if a workstation processes a batch but does not have **floatingDataExtract** installed, it will just ignore the settings.

If set to FALSE, if a workstation processes a batch but does not have **floatingDataExtract** installed, it will cause an error and the batch will be sent to Quality Control.

By default **floatingDataExtract** is set to TRUE. If it is required to set this to FALSE, then the **floatingDataExtract.AEX** file needs to be edited so that the highlighted line reads FALSE instead or TRUE. **floatingDataExtract** then has to be re-registered via Workflow Agent Manager in the Administration module. **No other changes must be made to floatingDataExtract.AEX**

[Workflow Agents]
floatingDataExtract

[floatingDataExtract]
WorkflowAgentID=floatingDataExtract
WorkflowAgentProgID=floatingDataExtract.Runtime
WorkflowAgentFile=floatingDataExtract_Runtime.dll
Description=Floating Data Extraction Utility for Kofax Capture
Version=1.0
SupportsNonImageFiles=True
SetupProgram=floatingDataExtract Setup
WorkflowAgentSkipIfCantLoad=False

[floatingDataExtract Setup]
OCXFile=floatingDataExtract_Setup.dll
ProgID=floatingDataExtract.WorkflowCtrl
Visible=0
MinSizeX=300
MinSizeY=150
BatchClassMenus=floatingDataExtract Setup Menu

[floatingDataExtract Setup Menu]
MenuText=floatingDataExtract Setup...
[easyQA Menu]
MenuText=easyQA v2 Properties...

4.2 Processing

floatingDataExtract runs at the close of each module in the Kofax workflow. Only after Recognition Server does it process Index Fields.

For this reason **floatingDataExtract** must be installed on the Kofax station that will run Recognition Server.

4.3 Running Processes Multiple Times

It is possible that other workflow agents, or manual actions can force a batch into the same module more than once.

When this happens, **floatingDataExtract** may process the fields more than once. In most cases this will not change the output unless the Index Field value was changed by some other process.

4.4 Setting Documents Valid

When setting documents valid, they will not be automatically opened in Validation.

This is usually the desired behaviour, but it is important to check that no other processes are set to run in Validation, as these may not run for the documents which have been marked valid.

5 Regular Expressions

Regular expressions are a way of defining patterns. The values of Index Fields will be evaluated against the defined regular expression. Where a valid match is found, it will be used to populate the Index Field.

A full discussion of regular expression definition is beyond the scope of this document – an online search will provide further details if required.

Some basics of regular expressions:

Character class	Description	Pattern	Matches
[*character_group***]**	Matches any single character in *character_group*. By default, the match is case-sensitive.	`[ae]`	"a" in "gray" "a", "e" in "lane"
[^*character_group***]**	Negation: Matches any single character that is not in *character_group*. By default, characters in*character_group* are case-sensitive.	`[^aei]`	"r", "g", "n" in "reign"
[*first* **-** *last* **]**	Character range: Matches any single character in the range from *first* to *last*.	`[A-Z]`	"A", "B" in "AB123"
.	Wildcard: Matches any single character except \n. To match a literal period character (. or **\u002E**), you must precede it with the escape character (**\.**).	`a.e`	"ave" in "nave" "ate" in "water"
\w	Matches any word character.	`\w`	"I", "D", "A", "1", "3" in "ID A1.3"
\W	Matches any non-word character.	`\W`	" ", "." in "ID A1.3"
\s	Matches any white-space character.	`\w\s`	"D " in "ID A1.3"
\d	Matches any decimal digit.	`\d`	"4" in "4 = IV"
\b	Matches a word boundary	`\b[A-`	Matches 'CAT' in

		Z]{3}\b	'A BLACK CAT RUNS' but not in 'SCATTERED'

Quantifier	Description	Pattern	Matches
*	Matches the previous element zero or more times.	\d*\.\d	".0", "19.9", "219.9"
+	Matches the previous element one or more times.	"be+"	"bee" in "been", "be" in "bent"
?	Matches the previous element zero or one time.	"rai?n"	"ran", "rain"
{ *n* }	Matches the previous element exactly *n* times.	",\d{3}"	",043" in "1,043.6", ",876", ",543", and ",210" in "9,876,543,210"
{ *n* ,}	Matches the previous element at least *n* times.	"\d{2,}"	"166", "29", "1930"
{ *n* , *m* }	Matches the previous element at least *n* times, but no more than *m* times.	"\d{3,5}"	"166", "17668" "19302" in "193024"

Regular expressions will match when a field's value contains the pattern. So for example looking for 'BC123' will match in 'ABC1234'. To avoid this, it is possible to use \b at the beginning or end of a regular expression.

\b[A-Z]{1}[0-9]{2}

A12	TRUE
A123	TRUE
1234	FALSE

\b[A-Z]{1}[0-9]{2}\b

A12	TRUE
A123	FALSE
AB1234	FALSE

6 Contact

For support and assistance please contact **mail@davidcrewe.com**