What is Extraction?

- The ability to electronically locate and extract pieces of data from documents that have been scanned or electronically produced
Extraction
Step through your presorted documents and try to find your important data, enter all values into an excel sheet and make notes like:

- Is the data always in a specific region?
- Does the data have a specific format?
- Are there some keywords?
- Is your value close to another value?
- Is your value already stored in a database?
- Is the data hand print, hand written, or machine print

In the Essentials class we will cover Position based extraction

Other extraction methods are covered in other courses
Position Based Extraction

- Sometimes information is always in the same area on a document. This is mostly the case with fixed layout forms.

```
<table>
<thead>
<tr>
<th>Title</th>
<th>First Name(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mr.</td>
<td>A.</td>
</tr>
<tr>
<td>Miss</td>
<td></td>
</tr>
<tr>
<td>Mrs.</td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td></td>
</tr>
<tr>
<td>Surname</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Address 1</td>
<td></td>
</tr>
<tr>
<td>Address 2</td>
<td></td>
</tr>
<tr>
<td>Zip code</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td></td>
</tr>
<tr>
<td>Date of Birth</td>
<td></td>
</tr>
<tr>
<td>Telephone</td>
<td></td>
</tr>
<tr>
<td>Email</td>
<td></td>
</tr>
<tr>
<td>Occupation</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>Total Income</td>
<td></td>
</tr>
<tr>
<td>Financial Status</td>
<td>Yes</td>
</tr>
<tr>
<td>Loan Purpose</td>
<td></td>
</tr>
<tr>
<td>Loan Amount</td>
<td></td>
</tr>
</tbody>
</table>
```
Position Based Extraction
Position Based Extraction

- Position based extraction is performed using an **Advanced Zone Locator**
- Uses **Zonal OCR** (Optimal Character Recognition)
- Used on documents with **Fixed** layouts
- Used on Hand Printed documents
- Draw **Zones** around the area to extract on a **Reference Document**
  - Text Zone
  - OMR/OMR Group Zone (optical marks – check boxes)
- Select a **Recognition Engine**
  - FineReader
  - RecoStar
Position Based Extraction

- **Image Cleanup Profiles**
  - Box or Line Removal

- **Anchors**
  - Can improve extraction by specifying anchor points in case of document skew (It is best to use words as anchor points - Machine Print)

- **Map Locator to a Field**

- **Test at design time by viewing Extraction Results**
You can test extraction at **run time** by creating a **job** in the **xyzbank website**.

The **Validation** manual activity allows users to review (and modify) extraction results.
Additional Slides
Overview

- Extraction
  - How do I extract information from a document
  - Formatters - Input and output formats and values
  - Validation rules – How to make values valid or invalid

- The beginning
  - How should I start?
  - Analyze documents
  - Analyze documents – lets do it together on ONE example
  - Analyze documents – now it is your turn
Extraction

Formatters - Input and output formats and values
Lets check our first invoice and define basic data. Where can you find data?
Let's check our first invoice and define basic data. Where can you find data?

**Invoice Nr**: 310254 RI  
**Invoice Date**: 05/21/03  
**PO Number**: 20030032

- **SUBTOTAL**: 21,553.42 USD  
- **SALES TAXES**: 0.00 USD  
- **AMOUNT DUE**: 21,553.42 USD
Data on a document and data in your backend system (DB) are 2 different things…

<table>
<thead>
<tr>
<th>Name</th>
<th>Value on doc</th>
<th>Value in DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoice Nr</td>
<td>310254 RI</td>
<td>310254RI</td>
</tr>
<tr>
<td>Invoice Date</td>
<td>05/21/03</td>
<td>05/21/2003</td>
</tr>
<tr>
<td>PO Number</td>
<td>20030032</td>
<td>20030032</td>
</tr>
<tr>
<td>Subtotal</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
<tr>
<td>Sales Taxes</td>
<td>0.00 USD</td>
<td>0.00</td>
</tr>
<tr>
<td>Amount Due</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
</tbody>
</table>

**SUBTOTAL** 21,553.42 USD
**SALES TAXES** 0.00 USD
**AMOUNT DUE** 21,553.42 USD
In most cases you want to convert values to a single unique format

<table>
<thead>
<tr>
<th>Name</th>
<th>Value on doc</th>
<th>Value in DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoice Nr</td>
<td>310254 RI</td>
<td>310254RI</td>
</tr>
<tr>
<td>Invoice Date</td>
<td>05/21/03</td>
<td>05/21/2003</td>
</tr>
<tr>
<td>PO Number</td>
<td>20030032</td>
<td>20030032</td>
</tr>
<tr>
<td>Subtotal</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
<tr>
<td>Sales Taxes</td>
<td>0.00 USD</td>
<td>0.00</td>
</tr>
<tr>
<td>Amount Due</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
</tbody>
</table>

- No spaces in invoice numbers
- Date format is always MM/DD/YYYY
- Currency USD removed
- Decimal symbol always “.”
- 3 decimal places symbol “,” removed
Extraction

Validation rules – How to make values valid or invalid
We want to read data from a document and often we can build a list of exceptions – when we know a value is invalid.

For example we could define the following rules:

- Invoice dates can not be older than one year (365 days)
- PO Numbers are always 8 digits long
- Total amount must be somewhere between 100 and 10000 USD
If we apply our rules to values some of them become invalid

- Invoice dates cannot be older than 1 year (365 days)
- PO Numbers are always 8 digits long
- Amount must be somewhere between 100 and 10000 USD

<table>
<thead>
<tr>
<th>Name</th>
<th>Value on doc</th>
<th>Value in DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoice Nr</td>
<td>310254 RI</td>
<td>310254RI</td>
</tr>
<tr>
<td>Invoice Date</td>
<td>05/21/03</td>
<td>05/21/2003</td>
</tr>
<tr>
<td>PO Number</td>
<td>20030032</td>
<td>20030032</td>
</tr>
<tr>
<td>Subtotal</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
<tr>
<td>Sales Taxes</td>
<td>0.00 USD</td>
<td>0.00</td>
</tr>
<tr>
<td>Amount Due</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
</tbody>
</table>
Our defined rules are very poor, so let’s rethink….  

- Invoice dates can not be older than 1 year (365 days)
- PO Numbers are always 8 digits long
- Amount must be somewhere between 100 and 10000 USD

<table>
<thead>
<tr>
<th>Name</th>
<th>Value on doc</th>
<th>Value in DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoice Nr</td>
<td>310254 RI</td>
<td>310254RI</td>
</tr>
<tr>
<td>Invoice Date</td>
<td>05/21/03</td>
<td>05/21/2003</td>
</tr>
<tr>
<td>PO Number</td>
<td>20030032</td>
<td>20030032</td>
</tr>
<tr>
<td>Subtotal</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
<tr>
<td>Sales Taxes</td>
<td>0.00 USD</td>
<td>0.00</td>
</tr>
<tr>
<td>Amount Due</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
</tbody>
</table>
Formatters - Input and output formats and values

- Our defined rules are very poor, so let’s rethink….  
  - Invoice dates can be older than 1 year (365 days), but they cannot be future dates  
  - PO Numbers are always 8 digits long  
  - Amount must be between 0 and 50000 USD

<table>
<thead>
<tr>
<th>Name</th>
<th>Value on doc</th>
<th>Value in DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoice Nr</td>
<td>310254 RI</td>
<td>310254RI</td>
</tr>
<tr>
<td>Invoice Date</td>
<td>05/21/03</td>
<td>05/21/2003</td>
</tr>
<tr>
<td>PO Number</td>
<td>20030032</td>
<td>20030032</td>
</tr>
<tr>
<td>Subtotal</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
<tr>
<td>Sales Taxes</td>
<td>0.00 USD</td>
<td>0.00</td>
</tr>
<tr>
<td>Amount Due</td>
<td>21,553.42 USD</td>
<td>21553.42</td>
</tr>
</tbody>
</table>
The beginning

How should I start?
How should I start?

- As you have seen you need “to know” your documents first
- You have to start with document analysis
  - If there is a “document expert” in your company, use him to analyze the documents
  - If there isn’t anybody who can help, find someone
  - If there still isn’t anybody, then ask yourself: How did you process these documents in the past?
How should I start?

![Image](image_url)

**Who is a document expert?**

- It’s NOT an IT guy
- It’s NOT a manager
- It’s the guy who processes documents on a daily basis
- It is YOU when you have finished your project
How should I start?

- In any case:

YOU have to get familiar with the documents now!
The beginning

Analyze documents
Analyze documents

STEP 1 – Classification:
Get a large stack of paper and try to sort it. When you do so, think about the technique that you use

- Have you recognized it by a specific layout?
  ➔ Layout classification

- Have you read the whole document?
  ➔ Content classification (Adaptive Feature classifier)

- Have you read just a phrase or a word?
  ➔ Content classification (Instruction)
STEP 2 – Extraction:
Step through your presorted documents and try to find your important data, enter all values into an excel sheet and make notes like:

- Is it always in a specific region?
- Does it have a specific format?
- Are there some keywords?
- Is your value close to another value?
- Is your value already stored in a database?
STEP 3 - Formatters:
Now let's check your values in your excel sheet. Create a new column for converted values

Do you want to have all values in OCR format?
- E.g. $12,345.99 USD

Does your backend system (DB) allow these values?

Do you want to have a specific unique format applied to the OCR result?
- E.g. $12,345.99 USD ➔ 12345.99
Analyze documents

- STEP 4 – Validation rules:
  Now let’s check the values in your excel sheet once more:
  Create a new column for valid and invalid data:
  - Digits / Characters or both?
  - Length of the value?
  - Is it in a specific format (regular expression)?
  - Is your data known (e.g. your product names)
  - Date:
    Can it be older than a number of days (365) and can it be in future?
  - Amounts:
    Do you use VAT (and if so, which percentages)?
The beginning

Analyze documents – let’s do it together on ONE example
Analyze documents – lets do it together on ONE example

How many different document types do you see?
How many different document types do you see?
How many different document types do you see?
Analyze documents – let's do it together on ONE example

How many different document types do you see?
Analyze documents – lets do it together on ONE example

Let’s check invoice numbers on one document / form type
Analyze documents – let’s do it together on ONE example

Let’s check invoice numbers on one document / form type

- Always on same position
  - Values are 3090, 3101, 3112, 3117
  - Numbers are always four digits and they always start with “3”
  - Keyword is always INVOICE NO
Analyze documents – let's do it together on ONE example

Let's check invoice date
Analyze documents – let's do it together on ONE example

Let's check invoice date

Always on same position

Always to the left of the invoice number

Values 03/24/2003, 04/08/2003, 04/16/2003, 04/24/2003

Keyword is always DATE
Findings:

- Invoice numbers and invoice dates are always in the same area
- Specific format for invoice number is 3xxx
- Good keyword for invoice number
- Bad keyword for invoice date
  - Date is part of every date
    (invoice date, order date, delivery date)
- Relation between invoice number and date
  - Date is always to the left of the invoice number
- Specific format for date is MM/DD/YYYY
- Invoice date is always in past, can’t be in future and it is very old
The beginning

Analyze documents – now it is your turn
Analyse documents – now it is your turn

- You can find some documents in front of you, try to do the same like we have done
  1. Try to sort them
  2. Try to find important data:
     Invoice number, invoice date, order number, total amount
  3. Try to find some further info about your data
     1. Always in same area? Specific format?
        Keywords? Relation?
     2. Is the OCR format also used by your backend system?
     3. Do you want to convert the format?
     4. Can you apply some logic to values (validation rules)?
Document sets
If you want to create a Transformation Designer project you need a lot of documents.

Best practice:
- If possible you should have 50 to 150 documents available for each class
- You can split your document set in different sets later

We use several document sets
- Training sets
  - For classification
  - For extraction
- Reference sets
- Test sets
What is a test set of documents?

- A test set of documents is a windows folder with tiff, pdf, xdc or text files in it.
- It is used to play around with these files in project builder and to test your current project settings.
- You can test your classification, separation and extraction.
- You have to manually review the results.
- There is no automatic information available if documents was processed correctly.
- A test set does NOT contain documents from other sets.
- Documents can be used for other sets, but if you want to do so, you should delete them from your test set.
What is a training set of documents?

A training set is used to train classification or extraction.

Before you can add a document to your training set, you need to know what you want to do with it.

Classification training set

- Layout classification
  - You need at least one image for every class

- Content classification
  - You need several text files for every class

Extraction training set

- Every time you are using group locators, you can add a document to your train set for extraction. This will improve extraction quality.
Document sets – Reference set

◆ What is a reference set of documents?
  ◆ A reference set is used to test your current project against it and to obtain information about if it was processed correctly or not. You can also have benchmarking (statistics) for it. It is always a prepared set of documents.

◆ Reference set for classification
  ◆ Are used to benchmark your classification

◆ Reference set for extraction
  ◆ Are used to benchmark your extraction

◆ Reference set for separation
  ◆ Are used to benchmark your separation
A reference set for classification is a windows folder with some
subfolders in it

Every subfolder must have the same name as the class in your
project

It must be exactly the same name (Case sensitive)

Every subfolder contains only files from a specific class

For example:
If you have classes
Oki, Fedex and Others in your project
you will have a windows folder RefSet with subfolders
Oki, Fedex and Others in it

Transformation Designer will try to match a subfolder to a class
and show correct and incorrect classification results
A reference set for extraction is a windows folder with xdc files in it. (XDC files are internal Transformation Server files similar to XML files)

Every field and its expected value is stored in such an xdc file.

You can test your project against your reference set.

As a result you can see which documents were extracted correctly or incorrectly.

You can also see if it will be shown or skipped in Validation.

An XDC file from a reference set for extraction is also called “Golden file.”

Golden Files can be created by a validation operator or you can build them in Project Builder during a “Test Validation.”
A reference set for separation is again a Windows folder with xdc files in it.

Here we don’t care about fields and values for separation.

We care about separation only.

You can test your project against your reference set.

You will see if documents were separated correctly OR if documents were separated incorrectly.

You can get a benchmark from it.
First project – Build sets
First project – Build sets

Test set, Classification training set & Classification reference sets
First project – Build sets
First project – Build sets

Right click on project class and create a new class called “Invoices”
First project – Build sets

Right click on Invoices and create new classes Dell, FedEx and Oki
First project – Build sets

Open document set

Select C:\Documents\unsorted
First project – Build sets

Double click on a file to show document viewer
First project – Build sets

This is a FedEx document
Close the document viewer
First project – Build sets

Right click on your document and select Add to Training Set of Selected class (Classification). Focus on Class FedEx.
First project – Build sets

Focus on Class Dell

Right click on a Dell document and select Add to Training Set of Selected class (Classification)
First project – Build sets

Focus on Class Oki

Right click on a Oki document and select Add to Training Set of Selected class (Classification)
First project – Build sets

- We have one document assigned to every class.
- We can review our training set easily at any time:
  - Select class to review first (focus it).
  - Click on Classification Set in your documents panel.
First project – Build sets

- You can review all classification documents if you change the filter to none
First project – Build sets

- Let’s test our classification

1. Switch back to your test documents
2. Switch to the Process tab of the ribbon
3. Select Classification from Train Area
4. Select All Documents in Document panel
5. Click on Classify in Test Panel
First project – Build sets

This column shows you the classification result for every document
First project – Build sets

Double click opens document viewer. Review your document and its classification result.
First project – Build sets

- In document viewer you have to review every document now
- You don’t know if a document was classified correctly or incorrectly
- To avoid manual review you can work with classification reference sets
First project – Build sets

- Lab
  - Create your very first project
  - Add documents to the classification set
  - Classify your test set
  - Review results
First project – Build sets

Classification reference sets
First project – Build sets

- Let's create a classification reference set
- Copy the folder called “unsorted” and rename it to “sorted”
- Create 3 subfolder (Oki, Dell and FedEx)
- Move images to correct folders
First project – Build sets

1) Click on Open document set

2) Select sorted folder
First project – Build sets

3) Include subdirectories

4) Assign subdirectory as class
First project – Build sets

5) Right click on new test set

6) Use it as Benchmark set
First project – Build sets

7) Select All Documents

8) Click on Classification Benchmark
First project – Build sets

◆ Lab
  ◆ Create a new reference set
  ◆ Move all documents to reference sets
  ◆ Run Classification benchmark
  ◆ Review your results
First project – Build sets

Extraction reference set
First project – Build sets

- Based on our layout classification project you can build a reference set for extraction now.
- Keep in mind that we need formatters and validation rules which will be applied to a field.
- You have already some knowledge about:
  - Invoice Number
  - Invoice Date
  - Order Number
  - Total Amount
- Let’s start with these fields.
First project – Build sets

First step is to add a new field to our main class Invoices.

1. Expand invoices by clicking on the eyeball symbol
2. Right click on Fields
3. Select Add Field
4. Name it InvoiceNumber
First project – Build sets

Field properties appears in Panel Details

You want to have a formatter which removes spaces on this field. Click on Formatters
Two default formatters are available already but you have to review them. Review and maybe change it by double clicking on it.
First project – Build sets

Default Date Formatter uses input formats to search on a document. Input format is also used to convert keying from validator operators. There are several formats available and you can select them.
First project – Build sets

If you process US invoices you have to use **MM_DD_YYYY**

Month is always first to digits

Date Output Format is target format.
You expect to receive data in this format in your backend system.
First project – Build sets

Amount formatting is used to convert amounts. Enter all expected currencies and use a correct decimal symbol.
First project – Build sets

Add more fields to your project
First project – Build sets

Assign DefaultDateFormatter for Invoice Date
Add field OrderDate
First project – Build sets

Assign DefaultAmountFormatter for total amount
First project – Build sets

- All fields are created
- Most of them have a link to formatters
- We don’t have validation rules
  (Rules to make values valid or invalid based on logic)
- This scenario can happen in productive environment when we face a new document
- Let’s try to validate all our documents like a validation operator
- It is important that you will find every value on all documents
- Later we want to store our values in xdc files, because then we can test our project settings against it
Why is validation necessary and important?

- There will always be documents with OCR errors on them
- A validation operator lives in the KTA Validation module for 8 hours each day
- Do your best to make him/her happy!
- Speak to them
- Improve the validation forms
- Validation is the most important part and you need to get your operators satisfied and highly productive
First project – Build sets

- Why not using backend system to enter values?
  - KTA validation is optimized to get values from a document
  - Compare KTA Validation with entering values directly into the backend system

- In our case processing time for one document is:
  - KTA validation: 1 minute
  - Backend system: 10 minutes
    (load document, load backend system form, enter values, switch between document and backend system and so on)

- If you can save 9 minutes for each document, think about 100000 documents
  You will save 900000 min ➔ 15000 hrs ➔
  1875 working days
First project – Build sets

You can save your new field values in the xdc files by clicking on

Save All Documents

Or if you want to save only one document by clicking on

Save Selected Document
First project – Build sets

Xdocs contain target values.

You can use this folder as a benchmark set by
Right clicking your test set and by selecting Use as Benchmark Set
First project – Build sets

Benchmarking gets enabled.

Select Extraction to run your automatic extraction against your benchmark set.
First project – Build sets

Extraction benchmark runs

You will see that nothing was extracted.
REASON: We are not using extraction methods right now.

BUT
You can see all your entered values already in brackets [ ].
First project – Build sets

- Lab
  - Create formatters
  - Link formatters to fields
  - Process documents
  - Validate documents
  - Store validated data in xdocs
  - Run extraction benchmark
  - Review your results
First project – Position based extraction

Add extraction to your project
Now it’s time to add automatic extraction to our project.

Position based extraction is performed using an Advanced Zone Locator in Transformation Designer.

Keep in mind that it is a position based extraction, so you can use it only on one specific structured form.
First project – Position based extraction

1) Double click on Oki and expand it

2) Select an Oki document

3) Right click on Locators and add a new one
First project – Position based extraction

4) Name it OKI_AZL

5) Select Advanced Zone Locator

6) Click on Locator properties
AZL needs sample images.
If you click on it, then it will use the selected image from test set.
Always use good quality images.
If you add 5 different samples it is possible to use automatic background removal.
First project – Position based extraction

Sample images will be renamed to Sample0, Sample1 and so on.
Make notes about which test documents were used.
Double click on Sample0.xdc (Reference) to draw extraction zones.
First project – Position based extraction

1) Select T to extract text values

2) Draw a zone around your text value
First project – Position based extraction

3) Name it

4) Choose a recognition engine

5) Test it
First project – Position based extraction

Well done!
Text value is fine
If you have any error try to change recognition engine, zone position, image cleanup and anchors.
You need a zone for every value you want to extract.
See yellow highlights in document viewer.
First project – Position based extraction

Test your Advanced Zone Locator.
You will see results highlighted in document viewer.
First project – Position based extraction

Select another document from your test set and click on test again to show results for another document.
First project – Position based extraction

Double click on fields to show fields of Oki class

Assign your field to a subfield from AZL to populate it
First project – Position based extraction

Every field is linked to a locator now
First project – Position based extraction

Run extraction benchmark again to get statistics
Results are not perfect, but at least we get some values now. There is plenty of space for improvement. We will speak about AZL optimization in a later module.
First project – Position based extraction

Lab
- Add an advance zone locator (AZL) to your class Oki
- Link Oki fields to AZL
- Process documents
- Run Extraction Benchmark
- Review your results
First project – Extraction by formats & keywords

Always try to be generic in first step
First project – Extraction by formats & keywords

- We have created a main class called invoices and some subclasses
- Invoices may be sent by several thousands of vendors and perhaps you don’t want to have a subclass and an Advanced Zone Locator for each of those invoices…
- Let’s suppose that we want to set up the project in a more generic manner and extract values for all these documents in the same way
- Remember that we may not get perfect results
First project – Extraction by formats & keywords

To be generic you need to know:

- Which values do you want to extract?
- Which keywords are accompanying your values?
- Is there a relation between the values?
- Do I know the values already? (Can I use a database?)

- We made some notes earlier on…
- Let’s review our current knowledge of the documents
First project – Extraction by formats & keywords

PONumber is always 8 digits
Keyword is Purchase Order

Invoice Number is always 9 digits
Keyword is Invoice Number

Invoice Date is always MM/DD/YY
Keyword is Invoice Date

Total amount is always in US Format
Keyword is Invoice Total

PO Number: 20026306

Invoice Number: 196328182

Invoice Date: 12/26/02

Invoice Total: $2,038.66
## First project – Extraction by formats & keywords

<table>
<thead>
<tr>
<th>Invoice Number</th>
<th>Keyword</th>
<th>Invoice Date</th>
<th>Keyword</th>
<th>PONumber</th>
<th>Keyword</th>
<th>Total amount</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>\d{9}</td>
<td>Invoice Number</td>
<td>US DATE</td>
<td>Invoice Date</td>
<td>\d{8}</td>
<td>Purchase Order</td>
<td>US Amount</td>
<td>Invoice Total</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notes:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
First project – Extraction by formats & keywords

- **PONumber** is sometimes empty, 6, 8 or 12 digits or like A55611
  - **Keyword** is **P.O. NUMBER**

- **Invoice Number** is always 9 digits
  - **Keyword** is **Freight Bill Number**

- **Invoice Date** is always MM/DD/YY
  - **Keyword** is **Date**
  - **To be found below invoice number**

- **Total amount** is always in US Format
  - **Keywords are** Please pay this amount

- **PLEASE PAY THIS AMOUNT** 62 76
First project – Extraction by formats & keywords

<table>
<thead>
<tr>
<th>Invoice Number</th>
<th>Keyword</th>
<th>Invoice Date</th>
<th>Keyword</th>
<th>PONumber</th>
<th>Keyword</th>
<th>Total amount</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>\d{9}</td>
<td>Invoice Number</td>
<td>US DATE</td>
<td>Invoice Date</td>
<td>\d{8}</td>
<td>Purchase Order</td>
<td>US Amount</td>
<td>Invoice Total</td>
</tr>
<tr>
<td>\d{9}</td>
<td>Freight Bill Number</td>
<td>US DATE</td>
<td>Date</td>
<td>\d{6} \d{8} \d{12} A\d{4}I</td>
<td>P.O. Number</td>
<td>US Amount</td>
<td>Please pay this amount</td>
</tr>
</tbody>
</table>

Notes:
Fedex: Poor Keyword for Invoice Date, but it is below Invoice number. Keywords ORIGIN and DEST are always below Invoice Date. PO Number can be empty
First project – Extraction by formats & keywords

**Invoice Date** is always MM/DD/YY
Keyword is Invoice Date

**PONumber** is 8 digits
Keyword is Customer PO NO

**Invoice Number** is always 6 digits and 2 chars
Keyword is Invoice Number

**Total amount** is always in US Format
Keyword is AMOUNT DUE
### First project – Extraction by formats & keywords

<table>
<thead>
<tr>
<th>Invoice Number</th>
<th>Keyword</th>
<th>Invoice Date</th>
<th>Keyword</th>
<th>PONumber</th>
<th>Keyword</th>
<th>Total amount</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>\d{9}</td>
<td>Invoice Number</td>
<td>US DATE</td>
<td>Invoice Date</td>
<td>\d{8}</td>
<td>Purchase Order</td>
<td>US Amount</td>
<td>Invoice Total</td>
</tr>
<tr>
<td>\d{9}</td>
<td>Freight Bill Number</td>
<td>US DATE</td>
<td>Date</td>
<td>\d{6} \d{8} \d{12} A\d{4}I</td>
<td>P.O. Number</td>
<td>US Amount</td>
<td>Please pay this amount</td>
</tr>
<tr>
<td>\d{6}[A-Z]{2}</td>
<td>Invoice NO</td>
<td>US DATE</td>
<td>Invoice Date</td>
<td>\d{8}</td>
<td>Customer PO NO</td>
<td>US Amount</td>
<td>Amount Due</td>
</tr>
</tbody>
</table>

### Notes:

**Fedex:**

Poor Keyword for Invoice Date, but it is below Invoice number
PO Number can be empty

**Oki:**

Poor Keyword for Total Amount, because it has white text on a black background
Total Amount is always below Sales Taxes
Based on this knowledge we can create some generic locators for your project.

Because it is generic knowledge we want to create them on the main Invoices class.

Every class below Invoices will inherit settings from this main class.

Sometimes main classes are called parent classes.

Sometimes subclasses are called child classes.
First project – Extraction by formats & keywords

1) Add a new locator to your system and name it FL_InvoiceNumber
First project – Extraction by formats & keywords

2) Use Format Locator as Locator Method

3) Click on locator properties
First project – Extraction by formats & keywords

4) Add a new regular expression to it
First project – Extraction by formats & keywords

4) Enter regular expression

5) Switch to Evaluation Settings
First project – Extraction by formats & keywords

6) Enter your keyword

7) Select “Match all words as a phrase” if you have a space between your keywords and if you expect them to be one line

8) Add it to list
Keyword list:
Turn keywords on or off for testing
Just click on use

9) Enter your second keyword

10) Add it to list
First project – Extraction by formats & keywords

All Keywords are in the list now

11) Test your settings on the document which is selected in test docs
First project – Extraction by formats & keywords

129

Test results in document viewer

Test results in locator properties

12) Select another document from test set and try it again
First project – Extraction by formats & keywords

Too many results with a high confidence for Dell documents
Let's have a closer look
First project – Extraction by formats & keywords

First result comes from page 2 and it is waybill number
First project – Extraction by formats & keywords

Second result comes from page 1 and it is invoice number
First project – Extraction by formats & keywords

Third result comes from page 1 and it is waybill number again

NOTICE: GREEN HIGHLIGHTS ARE ALWAYS SELECTED RESULTS
First project – Extraction by formats & keywords

Keyword relation becomes important now.

Keyword “Invoice Number” is to the West of the target.

Deselect NW and N and click on Modify.
First project – Extraction by formats & keywords

Relation is west only now
If you test it again, you will see…
You have only one result with a high confidence.

GREAT!
First project – Extraction by formats & keywords

Assign your field to a locator

1) Double click on Fields below Invoices

2) Change column “Field populated by” to FL_InvoiceNumber
First project – Extraction by formats & keywords

3) Benchmark your new extraction settings
First project – Extraction by formats & keywords

Significant improvement
First project – Extraction by formats & keywords

- Settings for FL_InvoiceDate are:
  - Use a predefined regular expression
  - Use keywords
First project – Extraction by formats & keywords

- Settings for FL_OrderNumber are:

Based on your documents

Use your keywords
First project – Extraction by formats & keywords

Settings for FL_OrderNumber are:

- Predefined regular expression for US Amounts available
  - Use Amount with Dot and Amount with Blank

Use your keywords
First project – Extraction by formats & keywords

Double click on Fields below Class Invoices
Link all your fields to locators
First project – Extraction by formats & keywords

Amazing results
But it is not good enough 😊

Run extraction benchmark again
First project – Extraction by formats & keywords

- **Lab**
  - Add Format Locators to the Invoices class
  - Link your new locators to fields
  - Unlink fields from AZL in class Oki (Set it back to inherited)
  - Run extraction benchmark
  - Review your results
First project – *Improvements*

Always improve your project
Always keep the focus on your operator
First project – *Improvements*

- First project has good recognition rates, but there is still space for further improvement.
- If you want to optimize stuff, a good way is to check your benchmark first.
- Another way to get further knowledge is to validate a whole bunch of documents and make some notes to it.
- Let’s have a closer look at extraction benchmarks.
First project – *Improvements*

Green fields are correct (fields will be skipped in validation)

Green check in validation
Value is correct
Nobody will review this value
First project – Improvements

Yellow fields are fields which are incorrect BUT they are displayed in validation and somebody has to confirm their values.

Exclamation mark
Field value is empty
Somebody has to enter field value.
First project – *Improvements*

Red fields are fields which are incorrect AND nobody will review them because TotalAgility is sure that their value is correct.

Green check in front Value is incorrect.

The amount field does not contain a valid amount.
First project – Improvements

Blue fields are fields which are correct
BUT somebody has to confirm them

Red exclamation mark
Value is incorrect

Unnecessary keystrokes
First project – *Improvements*

Save results before trying to improve your project...
First project – *Improvements*

...because later you can compare your benchmarks.
First project – *Improvements*

- Why is a field green or red?
  - First check is your extraction confidence
  - Second check is distance between first and second result
  - Third check is your formatting
  - Fourth check is your validation rule

- What is a validation rule?
  - We will discuss it later
First project – Improvements

First check: Validation threshold

Second check: Distance between first two results

Third check: Formatter
First project – Improvements

Fourth check
Validation rules linked to fields
First project – *Improvements*

Because we don’t have formatters or validation rules applied to order number, field state is red or green based on confidence.
First project – *Improvements*

Test your locator FL_OrderNumber
You will see two results with 83,33 % confidence

Test it in document viewer
You will see two keywords highlighted in orange and values next to it
First project – *Improvements*

Customer PO NO was searched fuzzy

System gets Customer NO, this is 83,33% of your keyword Customer PO NO
First project – Improvements

Customer PO NO was search fuzzy
System gets Customer NO, this is 83.33% of whole keyword
Minimum distance is set to 10%

In our case distance is 83,33 – 83,33 = 0 %

Field status becomes red
First project – *Improvements*

*Let's start easy*

We have a lot of blue documents for Dell invoices.

There is space for improvement.

Double click on first Dell to open document viewer and close extraction benchmark.
First project – **Improvements**

Double click on locator FL_OrderNumber

Test it
First project – Improvements

Second alternative is in north west of Purchase order number

Alternative 1 is correct invoice number
First project – Improvements

Review your keyword settings. Relation is W only, don’t use W, N, NW (default setting).
First project – Improvements

New test shows that only PONumber gets a confidence of 100%. Because order number is printed twice on the document, you see the first two results with the same value.
First project – Improvements

Run extraction benchmark again

Purchase Order number for Dell is optimized! 😊

Scroll down to find next issues, don’t worry about minor stuff in first run
First project – *Improvements*

1. Double click on first document (Document viewer appears)
2. Save extraction benchmark
3. Close extraction benchmark
4. Double click on FL_InvoiceDate (Locator properties appears)
5. Test it
Confidence of alternative is too low
Lets check keywords again
First project – Improvements

Keyword DEST and ORIGIN was set to south east and south west. Relation in OCR (xdoc) is not always the same as what you see on your screen. Add South for both and test it again.
First project – Improvements
First project – *Improvements*

Run extraction benchmark again.

Look at your extraction benchmark summary: **88.72%** are correct and skipped!
Some documents don’t have an order number (and this is correct).

BUT no order number means no confidence, so this will stay red until we use a validation rule.
First project – *Improvements*

Process your documents again (click on arrow next to Process and process the whole batch)

Click on validate and try to validate all documents.

Check the time you need for this.
First project – *Improvements*

If you confirm a value, system will jump to next invalid (red) field and skip all green fields. Even if next invalid field is more than 100 documents away, the system skips all documents that are between them.

At first, it took you more than 15 minutes to Validate this batch. Now it has taken you 2 minutes!

13 MINUTES SAVED
First project – *Improvements*

- LAB
  - Use extraction benchmark to find errors
  - Make improvements to your locators
  - Rerun extraction benchmark
  - Review your results
First project – Green & red fields
(Validation Rules)

How to make values green or red automatically
First project – Green & red fields (Validation Rules)

- Fields can be valid or invalid
  - As already discussed in improvements
- Why is a field green or red?
  - First check your extraction confidence
  - Second check the distance between first and second result
  - Third check is your formatting
  - Fourth check is your Validation Rule
- What is a Validation Rule?
  - Now we will discuss it…
First project – Green & red fields (Validation Rules)

- A validation method is a project setting and can be used
  - From a locator to search for a value (we will discuss this in a later module)
  - Also it can be linked to a field to change the state of a field to red or green.

- This is called a **VALIDATION RULE**

- Use existing knowledge for your validation methods
- Available methods are: Standard, Date, Regular Expression, Single Field and Multi Field Script Validation and Invoice Validation
First project – Green & red fields (Validation Rules)

<table>
<thead>
<tr>
<th>Invoice Number</th>
<th>Keyword</th>
<th>Invoice Date</th>
<th>Keyword</th>
<th>PONumber</th>
<th>Keyword</th>
<th>Total amount</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>\d{9}</td>
<td>Invoice Number</td>
<td>US DATE</td>
<td>Invoice Date</td>
<td>\d{8}</td>
<td>Purchase Order</td>
<td>US Amount</td>
<td>Invoice Total</td>
</tr>
<tr>
<td>\d{9}</td>
<td>Freight Bill Number</td>
<td>US DATE</td>
<td>Date</td>
<td>\d{6} \d{8} \d{12} A\d{4}I</td>
<td>P.O. Number</td>
<td>US Amount</td>
<td>Please pay this amount</td>
</tr>
<tr>
<td>\d{6}[A-Z]{2}</td>
<td>Invoice NO</td>
<td>US DATE</td>
<td>Invoice Date</td>
<td>\d{8}</td>
<td>Customer PO NO</td>
<td>US Amount</td>
<td>Amount Due</td>
</tr>
</tbody>
</table>

Notes:

Fedex: Poor Keyword for Invoice Date, but it is below Invoice number
PO Number can be empty

Oki: Poor Keyword for Total Amount, because it has white text on black background
Total Amount is always below Sales Taxes
First project – Green & red fields (Validation Rules)

1) Click on Project setting (part of ribbon under Project Tab)
2) Switch to tab Validation
3) Add a new validation method
First project – Green & red fields (Validation Rules)

4) Give it a name

5) Select regular expression for invoice numbers

Current knowledge is like \d{8}
First project – Green & red fields (Validation Rules)

6) Add your regular expression (similar to format locator)

List of all added formats

Test your regular expressions
First project – Green & red fields (Validation Rules)

7) Create another Validation Method for invoice dates

8) Select Date Validation for invoice dates
First project – Green & red fields (Validation Rules)

9) Use DefaultDateFormatter

10) Invoice date can be really old but it is never in the future
   - Uncheck first option (Check for period before ref date)
   - Check second (Check for period after ref date)
   - Uncheck allow empty fields
First project – Green & red fields (Validation Rules)

11) Create a Validation Method for order numbers

12) Add your regular expressions
   Tick allow empty fields, because we have documents without a PONumber
First project – Green & red fields (Validation Rules)

All validation methods are created

Let’s link them to fields
First project – Green & red fields (Validation Rules)

1) Right click on Validation Rules (below class invoices)
2) Add a Single Field Validation Rule
3) Name it InvoiceNumber
First project – Green & red fields (Validation Rules)

4) Select Field InvoiceNumber

5) Add a validation method

6) Add your method InvoiceNumber
First project – Green & red fields (Validation Rules)

Create a new rule for invoice date
First project – Green & red fields (Validation Rules)

Last one for OrderNumber

Check “Validation rule makes uncertain results valid”

→ This is useful for empty order numbers: even if the field is empty our rule will display values in green
First project – Green & red fields (Validation Rules)

**Run extraction benchmark again**

Validation rule Order Number makes empty fields valid for order numbers. No one has to confirm empty fields.

**Save your benchmark and compare it**
First project – Green & red fields (Validation Rules)
First project – Green & red fields (Validation Rules)

- Lab
  - Create Validation Methods
  - Create Validation Rules
    (Link a field to a method)
  - Run Extraction Benchmark
  - Review your results
First project – What is next?

- More basic locators
- More advanced locators
- Trainable group locators (IGL, OGL, AGL, TGL)
- Evaluators – use to compare locators
- All Locators in detail
- All Evaluators in detail
- Optimize forms
- Best practices
- Scripting
- Licensing
### First project – What is next?

<table>
<thead>
<tr>
<th>Basic</th>
<th>Advanced</th>
<th>Trainable</th>
<th>Legacy</th>
<th>Evaluators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bar Code Locator</td>
<td>Database Locator</td>
<td>Amount Group Locator</td>
<td>Invoice Header Locator</td>
<td>Standard Evaluator</td>
</tr>
<tr>
<td>Advanced Zone Locator</td>
<td>Vendor Locator</td>
<td>Invoice Group Locator</td>
<td></td>
<td>Relation Evaluator</td>
</tr>
<tr>
<td>Format Locator</td>
<td>Table Locator</td>
<td>Order Group Locator</td>
<td></td>
<td>OCR Voting Evaluator</td>
</tr>
<tr>
<td>Database Locator</td>
<td>Line Item Matching Locator</td>
<td>Trainable Group Locator</td>
<td></td>
<td>Database Evaluator</td>
</tr>
<tr>
<td>Classification Locator</td>
<td>Text content locator</td>
<td></td>
<td></td>
<td>Address Evaluator</td>
</tr>
<tr>
<td>A2iA Zone Locator</td>
<td></td>
<td></td>
<td></td>
<td>Advanced evaluator (Invoice Evaluator)</td>
</tr>
</tbody>
</table>