

Version 2.1

Transform[®]

Plain Text Exercise

Plain Text Exercise

Transform Project Development Training Manual

Legal Statement

© Copyright CTA Consultancy Ltd All rights reserved.

Reproduction of this document and use of its contents is prohibited unless expressly approved.

Amendments and Changes

CTA Consultancy Ltd reserves the right to modify, update, or amend this document and its contents at any time without prior notice. Any changes made to this document will supersede previous versions. It is the responsibility of the recipient to ensure they are working with the most current version of this document. CTA Consultancy Ltd shall not be liable for any consequences arising from the use of outdated or superseded versions of this document.

Compliance

All recipients and users of this document must comply with applicable laws, regulations, and industry standards in their jurisdiction. The recipient acknowledges their responsibility to ensure that their use of this document and its contents complies with all relevant legal, regulatory, and contractual obligations. CTA Consultancy Ltd makes no warranty regarding compliance with specific regulatory requirements and disclaims any liability for non-compliance by the recipient. Any breach of these terms or applicable compliance requirements may result in immediate termination of access to this document and potential legal action.

This document and all information contained herein remains the exclusive property of CTA Consultancy Ltd. Any unauthorized disclosure, distribution, or use of this document or its contents may result in legal action. By accessing this document, the recipient acknowledges and agrees to maintain the confidentiality of all information contained within.

Plain Text Exercise Resources

Download Resources

The resources necessary to complete the Plain Text Exercise are available on Dropbox, link below.



Download and extract the ZIP file to a local folder.

Contents include:

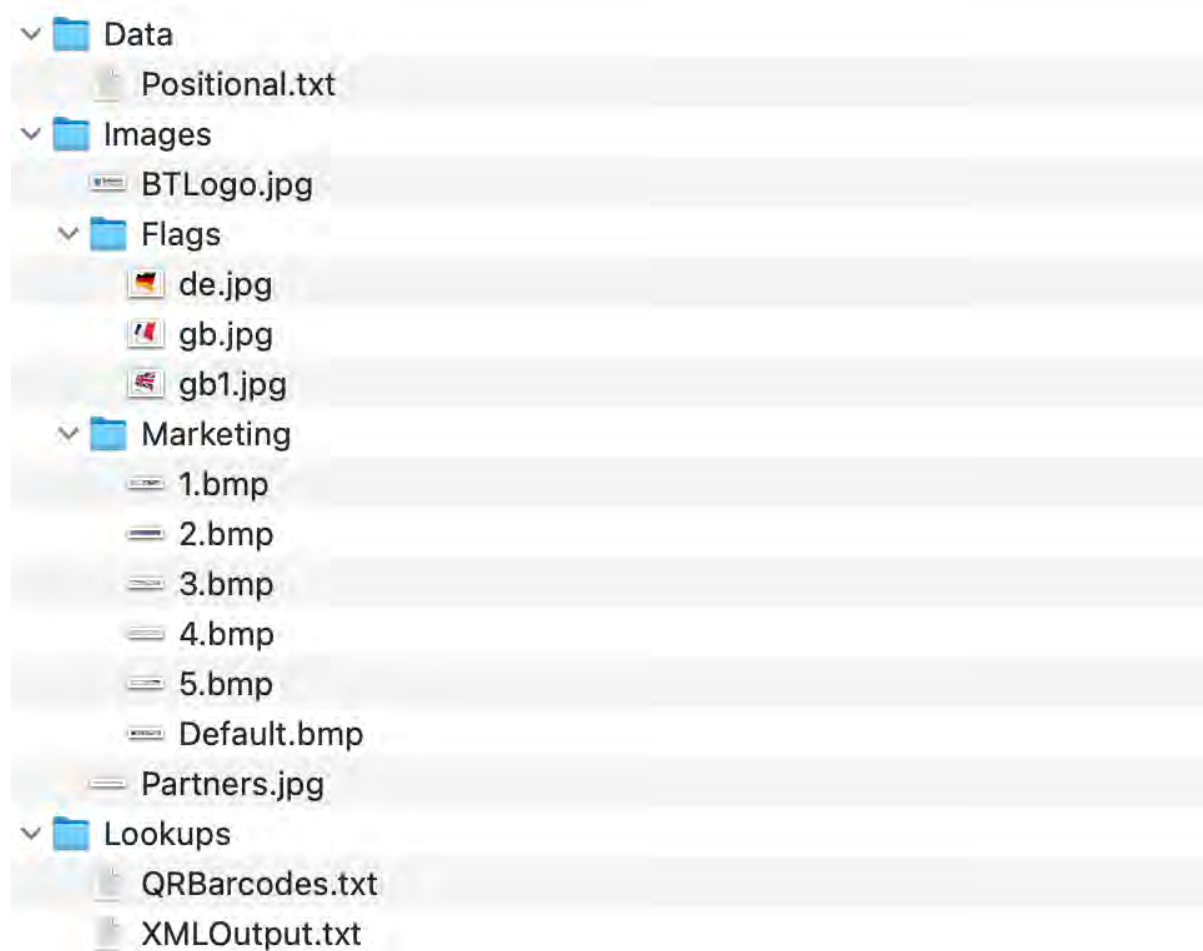




Table of Contents

Project Scope

Introduction	A-1
Project Scope	A-2
Input	A-4
Document Layout	A-10
Output	A-18

Walk Through Guide

Introduction	B-1
Repository Designer	
Branch Relationship Diagram	B-3
Branch Files Required	B-4
Repository Structure	B-5
Create Local Repository	B-6
Open Synchronizer program	B-7
Create Repository	B-7
Create Packages	B-10
Create Subfolders	B-12
Create Language Label Branches	
Draw Process	B-15
Label Values	B-16
Open Simple Action Template	B-17
Define Parameters	B-18
Add Memory Objects	B-20
Empty Container	B-20
Copy Container	B-22
Add Unicode Text Objects	B-24
Add Log Event	B-27
Save in Repository Package	B-29
Global Resources	
Global Constants	
Open Simple Action Template	B-31
Replace Root object with Items Folder	B-31
Create Input path	B-32
Create Output path	B-33
Create Image path	B-36
Scheduled Process Branch	
Open Simple Action Template	B-38
Add Scheduler	B-39
Create Subfolder Array	B-39
Branch Shortcut	B-40
Add Scan object	B-41
Tokenizer	B-41
Shell Command	B-42
Test	B-44
Plain Text Template	
Open Plain Text Template	B-46
Choose Project type	B-46
Browse for Sample Data	B-47
Select Output Type	B-47



Table of Contents

Plain Text To Container

Define Plain Text to Container Properties	B-49
Edit Standard Sections	B-50
Define Header Section	B-51
Create Header Blocks	B-52
Document Grouping	B-54
Define Custom Sections	B-55
Footer Section	B-55
Build Text Expression	B-57
Change Section Type	B-60
Item Lines Section	B-61
Description Lines Section	B-63
Update Container	B-64

Modify Branch Structure

GlobalConstants Branch Shortcut	B-67
Input Method	B-68
Scan Through Each Document	B-70
Delivery Method	B-83

Dynamic Branch Shortcut

Container Memory Item	B-91
Create Dynamic Branch Shortcut	B-92
Parameters	B-93

Document Organizer

Move Document Organizer on Branch	B-98
Repeated Section Container Shortcut	B-99
Add Page Types	B-100
Define Page Settings	B-101
Repeating Detail Configuration	B-105
Map Data onto Sortable Table Row	B-107
Create Background Forms	B-109
First Page	B-110
Continuation Page	B-111
Last Page	B-112
Single Page	B-113
Gradient Fill	B-114
Map Language Labels	B-118
Build Text Expression	B-121
Map Header Data	B-129
Page N of M	B-130
Dynamic Flag Image	B-133
QR Barcode	B-136
Map Footer Data	B-138
Add Subtotal	B-138
Date Difference	B-139
Dynamic Marketing Image	B-142

Synchronize and Deploy

Synchronize with Deployment Server	B-149
Deploy with Runtime Server	B-150
Package Watch List	B-151

Exercise overview

- Introduction
- Project Scope
- Input
- Data Specification
- Document Layout
- Output
- Repository Design
- Create Local Repository
- Plain Text Template Wizard
- Expression Editor
- Document Organizer
- Dynamic Images
- Print to PDF
- Output Culture settings
- Dynamic Branch Shortcut



Introduction

Bottomline Education Supplies provides a one-stop shop of education supplies including curriculum resources, exercise books, consumables, furniture, stationery, cleaning and janitorial supplies to education providers across the world.

They work with 800 audited suppliers and manage over 95,000 lines to provide the range and choice their customers require. Their extensive warehousing facilities, together with their expertise in supply chain management and logistics mean their customers can rely on minimal stock levels and just one supplier with fast, reliable delivery to meet the needs of their business.

As part of their continuous investment in innovation they have recently purchased Bottomline Technologies Transform Foundation Server to manage the delivery of outbound documents to their worldwide client base.

This exercise will teach you the techniques employed to deliver their Invoice document, we will explore the requirements and plan our approach before we embark on our Transform design.

Techniques covered in exercise.

- **New Template Plain Text Wizard** – Create a starting point for our project
- **Filter Process** – Plain Text to Container
- **Queue Process** – File Queue
- **Document Organizer** – Create dynamically paginated output
- **Culture** – Change culture for currency and date formats based on variable
- **Dynamic images** – Pull images from a library based on a variable
- **Scan object** – Loop through each document so the output can be conditional
- **Dynamic Branch Shortcut** – Target branches used for Language Labels
- **Print to PDF** – Produce PDF output with specific naming convention
- **XML output** – Conditionally produce XML based on variable content

Project Scope

The Transform process will be responsible for accepting Invoice data from the Host system via a file queue compose paginated documents and route them to the designated output. The project has three distinct elements.

Input

- Input Folder accept plain text positional invoice data from a UNC path.
- Archive original input spool files.
- Split Invoice batch files so each document can be autonomous.

Document Layout

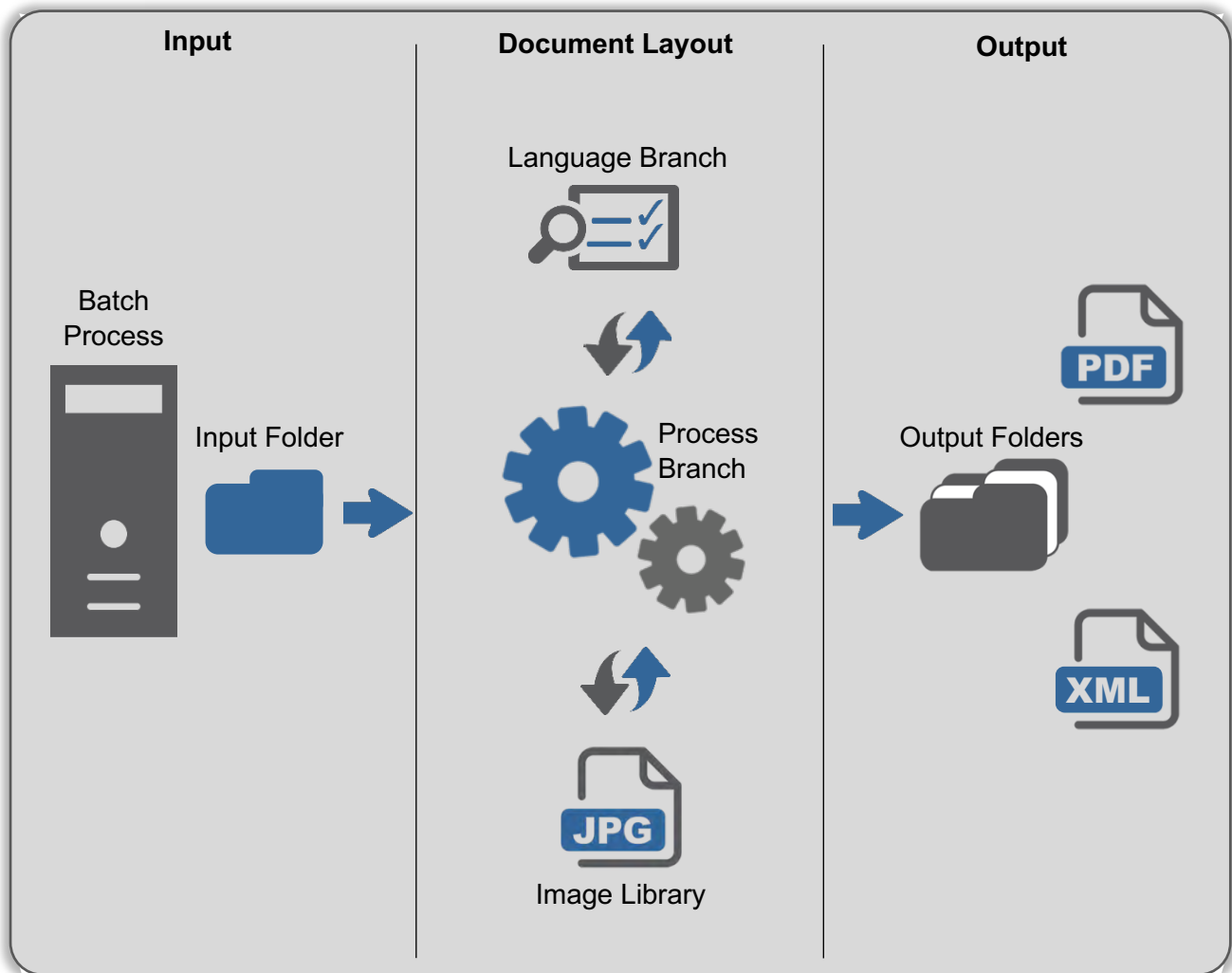
- Repaginate source data.
- Subtotals on page.
- Conditional language labels.
- Dynamic images.
- QR Barcode.
- Payment terms in footer.
- Optical mark recognition (OMR) marks.

Output

- Output should be in PDF format written to specified folders with unique naming convention.
- Conditional XML output is required for account numbers maintained outside the host data stream in an external CSV file.
- Audit of each invoice processed appended to daily log file.

Workflow diagram

The following high level workflow diagram depicts the required workflow.



Input

The Invoice data is generated from an IBM I-Series running Infor System 21 Enterprise Resource Planning (ERP). The invoices can be run as a batch containing multiple documents, or as individual on demand invoice reprints. The project must meet the following input requirements.

- Input Folder accept plain text positional invoice data from a UNC path.
- Archive original input spool files.
- Split Invoice batch files so each document can be autonomous.

Input Folder

Spool files will be written to a UNC path, where the Transform process will pick them up compose and route them to the desired output.



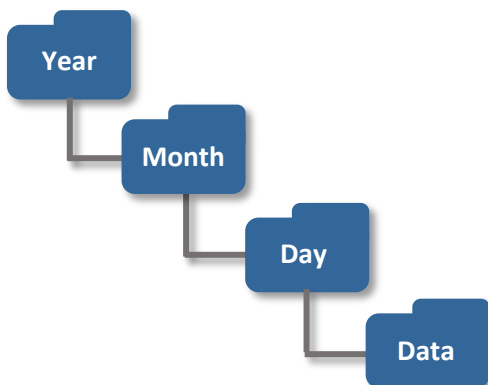
Important

When using a file queue process the Transform Foundation Service must be logged on with permissions to remove files from the input folder. The queue process deletes the original spool file from the source folder.

Archive Original Spool Files

As the host system only keeps the spool files for 7 days there is a request for the Transform Project to archive the original spool files.

The archive directory will need to be created by the Transform process with the folder structure illustrated below.



The archive filename convention will be based on the following

Original Input Filename_YYYYMMdd_HHmmss.txt

Split Invoices

The plain text positional source file can be produced via a batch process on the host containing multiple documents. It is imperative that the batch is split by document so it can be processed independently applying local conditions.

Data Specification

The plain text Positional data is structured by its position on a page, so it is important that we spend time to review the data so that we can identify the following.

- **Page definition** – Page break and line termination.
- **Documents** – Grouping key for each transaction within the spool file
- **Sections** – Fixed and repeating sections
- **Fields** – Field definitions



Important

It is imperative to have a comprehensive input data sample when completing the development, which contains multiple records and multiple pages containing all possible sections and fields.

Page Definition

The following table describes the page definition parameters.

Event	Hex Code	Decimal Code	Setting
Form Feed Causes Carriage Return	0C0D	1213	YES
Form Feed ends page	0C	12	YES
Insert Carriage Return after Line Feed	0A0D	1013	YES
Limit Line Length	-	-	NO
Limit Page Length	-	-	NO

Documents

It has been agreed that we use the **Invoice Number** as the grouping key to identify the transactions within a batch.

Fields

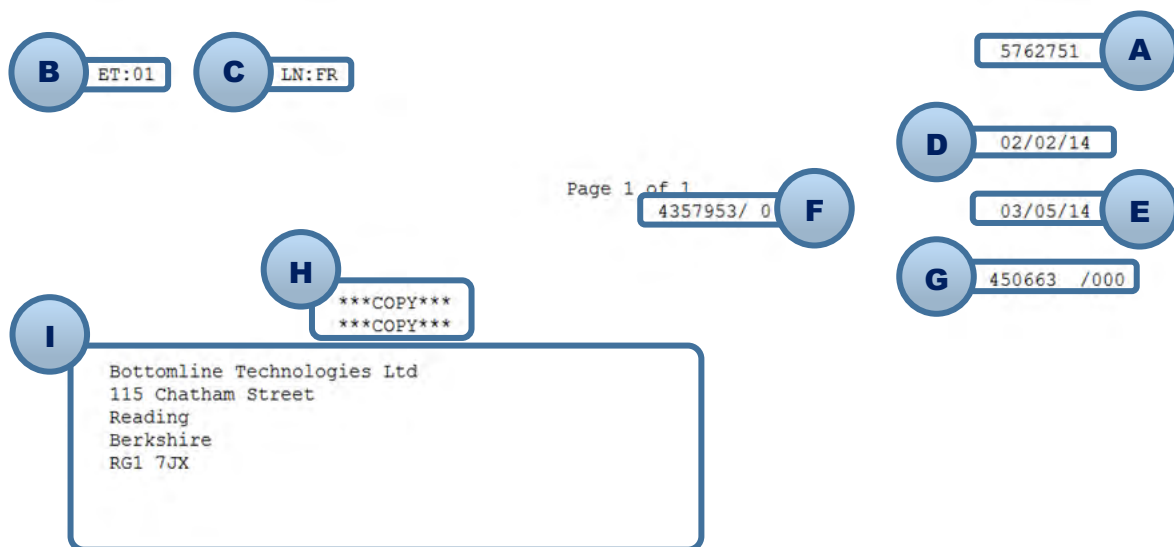
If possible ask the ERP team to supply a definition of the fields within the positional data containing.

- Field Name
- X position column
- Y position line
- Height
- Width

The following defines the fields used within the each section.

Header Section

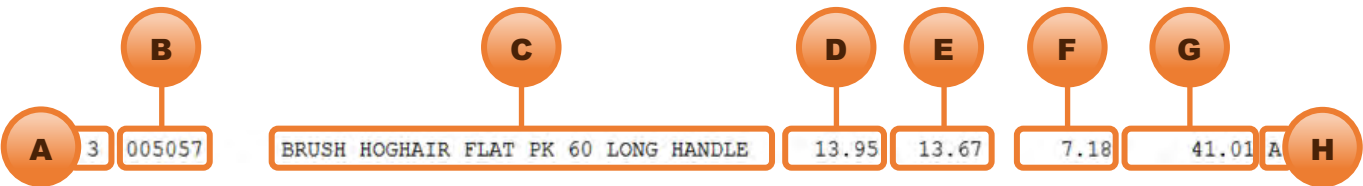
Fixed on every page of the positional data for the first 28 lines.



Reference	Field Name	X position Column	Y position Line	Width	Height
A	InvoiceNo	82	2	8	1
B	EstType	6	3	2	1
C	Language	22	3	2	1
D	InvoiceDate	82	6	8	1
E	PaymentDate	82	9	8	1
F	CustRef	52	9	12	1
G	AccountNo	81	12	12	1
H	Copy	24	13	10	2
I	NameAddress	4	16	50	8

Item Lines

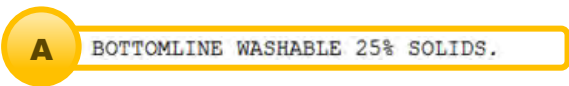
Custom repeated section as an Invoice can contain multiple item lines that span over pages.



Reference	Field Name	X position Column	Y position Line	Width	Height
A	QTY	3	0	4	1
B	ItemCode	8	0	7	1
C	Description	20	0	38	1
D	UnitPrice	59	0	7	1
E	DiscountPrice	67	0	7	1
F	Tax	75	0	9	1
G	LineTotal	85	0	10	1
H	TaxPoint	96	0	1	1

Description Lines

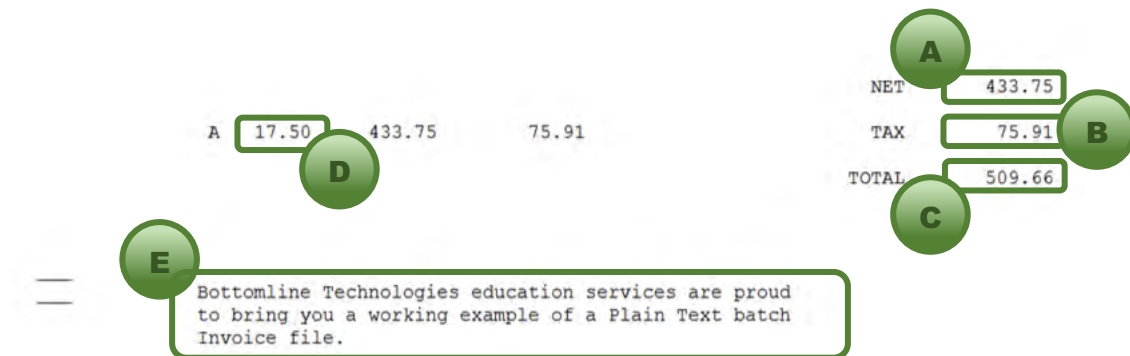
Custom repeated section as extended description lines, description lines can span multiple lines.



Reference	Field Name	X position Column	Y position Line	Width	Height
A	Description	20	0	38	1

Footer

Custom Footer section only appears on final page of each Invoice (document).



Reference	Field Name	X position Column	Y position Line	Width	Height
A	Net	84	2	12	1
B	Tax	84	4	12	1
C	Total	84	6	12	1
D	TaxRate	24	6	6	1
E	StrapLine	20	11	70	3

Document Layout

The document layout composed from the plain text positional source data must meet the following requirements.

- Repaginate source data.
- Sub totals on page.
- Conditional language labels.
- Dynamic images.
- QR Barcode.
- Payment terms in footer.
- Optical mark recognition (OMR) marks.

Repaginate Source Data

There is a desire to use the Invoice as a vehicle to communicate with the existing customer base by adding targeted marketing messages to the document. In order to maximise the space available the original source data needs to be repaginated to accommodate the messages. The following depicts the desired output once repaginated.

First Page

Continuation Page

Last Page

Invoice									
To: Bottomline Technologies Ltd 115 Chatham Street Reading RG2 1JH		Bottomline Technologies 115 Chatham Street Reading RG2 1JH Tel: +44 (0)118 220000		Invoice No: 45840-009 Invoice Date: 01/03/2014 Invoice To: 45840-009		Page 1 of 3			
Item Code	Item Description	Quantity	Unit Price	Unit	Total Price	VAT Rate	VAT	Net Total	Net Total
000018	CONCORDIUM BLUE STICKER PK 12	8	8.50	EA	68.00	0%	0.00	68.00	68.00
010005	CONCORDIUM PENCIL BRANDER 40 PER BOX	5	2.00	1/2	9.90	0%	0.00	9.90	9.90
010042	PAPERDIP PENCIL SHARPENER	5	8.50	EA	42.50	0%	0.00	42.50	42.50
010036	PENCIL SHARPENER METAL 1/2 SIZE BOX 20	3	8.50	EA	25.50	0%	0.00	25.50	25.50
010047	300M CLEAR RUBBER PICK UP 100	2	13.50	EA	27.00	0%	0.00	27.00	27.00
010060	TAPE BRUSH GUARDED PAPER 400M X 200M	5	2.15	EA	10.75	0%	0.00	10.75	10.75
010042	BRANDER TAPE 200M X 200M PICK UP	3	7.50	EA	22.50	0%	0.00	22.50	22.50
010024	TAPE CLEAR ADHESIVE CONCORDIUM PACK 6	8	1.75	EA	14.00	0%	0.00	14.00	14.00
010009	SCHOOL SCISSORS LEFT/RIGHT 3 1/2 INCH	1	33.75	EA	33.75	0%	0.00	33.75	33.75
010056	CLASP RINGERS DISPOSABLE 800 OF 10	1	2.40	EA	2.40	0%	0.00	2.40	2.40
010030	FLOURET WIRE 2 INCH PAPER	4	7.50	EA	30.00	0%	0.00	30.00	30.00
010051	LOCITE HOT MELT GLUE GUN	3	7.00	EA	21.00	0%	0.00	21.00	21.00
010052	LOCITE HOT MELT GLUE STICKS PK 10 300M X 300M	10	1.80	EA	18.00	0%	0.00	18.00	18.00
010110	CONCORDIUM PVA ADHESIVE GEL PUMP 5L PVA ADHESIVE - REMAINS OUT OF CLOTHING (WASHABLE) - VERBODEN RENEVAL INVERBODEN - GEBEIDEN INVERBODEN BESTE PRAKTIJKE	1	5.40	EA	5.40	0%	0.00	5.40	5.40
010006	CONCORDIUM READY MIXED PASTE PK 20	5	14.85	EA	74.25	0%	0.00	74.25	74.25
010035	MOUNTING PAPER A4 1000MM X 700MM	3	12.90	EA	38.70	0%	0.00	38.70	38.70
010002	ARISTO GARTHOPE A1 1000MM X 700MM	5	20.95	EA	104.75	0%	0.00	104.75	104.75
010002	ARISTO GARTHOPE A2 1000MM X 700MM	10	11.95	EA	119.50	0%	0.00	119.50	119.50
010004	ARISTO GARTHOPE A3 1000MM X 700MM	15	5.90	EA	88.50	0%	0.00	88.50	88.50
010007	QUALITY TRACKING PAPER A4 1000MM X 700MM	2	16.45	EA	32.90	0%	0.00	32.90	32.90
010003	TRACKING PAPER GREENPROOF A4 1000MM X 700MM	2	4.95	EA	9.90	0%	0.00	9.90	9.90
Sub Total								£721.84	£721.84

Continuation Page									
Transaction Details		Page 2 of 3							
Item Code	Item Description	Quantity	Unit Price	Unit	Total Price	VAT Rate	VAT	Net Total	Net Total
010005	PAPERDIP PENCIL SHARPENER	5	8.50	EA	42.50	0%	0.00	42.50	42.50
010036	PENCIL SHARPENER METAL 1/2 SIZE BOX 20	3	8.50	EA	25.50	0%	0.00	25.50	25.50
010047	300M CLEAR RUBBER PICK UP 100	2	13.50	EA	27.00	0%	0.00	27.00	27.00
010060	TAPE BRUSH GUARDED PAPER 400M X 200M	5	2.15	EA	10.75	0%	0.00	10.75	10.75
010042	BRANDER TAPE 200M X 200M PICK UP	3	7.50	EA	22.50	0%	0.00	22.50	22.50
010024	TAPE CLEAR ADHESIVE CONCORDIUM PACK 6	8	1.75	EA	14.00	0%	0.00	14.00	14.00
010009	SCHOOL SCISSORS LEFT/RIGHT 3 1/2 INCH	1	33.75	EA	33.75	0%	0.00	33.75	33.75
010056	CLASP RINGERS DISPOSABLE 800 OF 10	1	2.40	EA	2.40	0%	0.00	2.40	2.40
010030	FLOURET WIRE 2 INCH PAPER	4	7.50	EA	30.00	0%	0.00	30.00	30.00
010051	LOCITE HOT MELT GLUE GUN	3	7.00	EA	21.00	0%	0.00	21.00	21.00
010052	LOCITE HOT MELT GLUE STICKS PK 10 300M X 300M	10	1.80	EA	18.00	0%	0.00	18.00	18.00
010110	CONCORDIUM PVA ADHESIVE GEL PUMP 5L PVA ADHESIVE - REMAINS OUT OF CLOTHING (WASHABLE) - VERBODEN RENEVAL INVERBODEN - GEBEIDEN INVERBODEN BESTE PRAKTIJKE	1	5.40	EA	5.40	0%	0.00	5.40	5.40
010006	CONCORDIUM READY MIXED PASTE PK 20	5	14.85	EA	74.25	0%	0.00	74.25	74.25
010035	MOUNTING PAPER A4 1000MM X 700MM	3	12.90	EA	38.70	0%	0.00	38.70	38.70
010002	ARISTO GARTHOPE A1 1000MM X 700MM	5	20.95	EA	104.75	0%	0.00	104.75	104.75
010002	ARISTO GARTHOPE A2 1000MM X 700MM	10	11.95	EA	119.50	0%	0.00	119.50	119.50
010004	ARISTO GARTHOPE A3 1000MM X 700MM	15	5.90	EA	88.50	0%	0.00	88.50	88.50
010007	QUALITY TRACKING PAPER A4 1000MM X 700MM	2	16.45	EA	32.90	0%	0.00	32.90	32.90
010003	TRACKING PAPER GREENPROOF A4 1000MM X 700MM	2	4.95	EA	9.90	0%	0.00	9.90	9.90
Sub Total								£909.80	£909.80

Last Page									
Transaction Details		Page 3 of 3							
Item Code	Item Description	Quantity	Unit Price	Unit	Total Price	VAT Rate	VAT	Net Total	Net Total
010005	PAPERDIP PENCIL SHARPENER	5	8.50	EA	42.50	0%	0.00	42.50	42.50
010036	PENCIL SHARPENER METAL 1/2 SIZE BOX 20	3	8.50	EA	25.50	0%	0.00	25.50	25.50
010047	300M CLEAR RUBBER PICK UP 100	2	13.50	EA	27.00	0%	0.00	27.00	27.00
010060	TAPE BRUSH GUARDED PAPER 400M X 200M	5	2.15	EA	10.75	0%	0.00	10.75	10.75
010042	BRANDER TAPE 200M X 200M PICK UP	3	7.50	EA	22.50	0%	0.00	22.50	22.50
010024	TAPE CLEAR ADHESIVE CONCORDIUM PACK 6	8	1.75	EA	14.00	0%	0.00	14.00	14.00
010009	SCHOOL SCISSORS LEFT/RIGHT 3 1/2 INCH	1	33.75	EA	33.75	0%	0.00	33.75	33.75
010056	CLASP RINGERS DISPOSABLE 800 OF 10	1	2.40	EA	2.40	0%	0.00	2.40	2.40
010030	FLOURET WIRE 2 INCH PAPER	4	7.50	EA	30.00	0%	0.00	30.00	30.00
010051	LOCITE HOT MELT GLUE GUN	3	7.00	EA	21.00	0%	0.00	21.00	21.00
010052	LOCITE HOT MELT GLUE STICKS PK 10 300M X 300M	10	1.80	EA	18.00	0%	0.00	18.00	18.00
010110	CONCORDIUM PVA ADHESIVE GEL PUMP 5L PVA ADHESIVE - REMAINS OUT OF CLOTHING (WASHABLE) - VERBODEN RENEVAL INVERBODEN - GEBEIDEN INVERBODEN BESTE PRAKTIJKE	1	5.40	EA	5.40	0%	0.00	5.40	5.40
010006	CONCORDIUM READY MIXED PASTE PK 20	5	14.85	EA	74.25	0%	0.00	74.25	74.25
010035	MOUNTING PAPER A4 1000MM X 700MM	3	12.90	EA	38.70	0%	0.00	38.70	38.70
010002	ARISTO GARTHOPE A1 1000MM X 700MM	5	20.95	EA	104.75	0%	0.00	104.75	104.75
010002	ARISTO GARTHOPE A2 1000MM X 700MM	10	11.95	EA	119.50	0%	0.00	119.50	119.50
010004	ARISTO GARTHOPE A3 1000MM X 700MM	15	5.90	EA	88.50	0%	0.00	88.50	88.50
010007	QUALITY TRACKING PAPER A4 1000MM X 700MM	2	16.45	EA	32.90	0%	0.00	32.90	32.90
010003	TRACKING PAPER GREENPROOF A4 1000MM X 700MM	2	4.95	EA	9.90	0%	0.00	9.90	9.90
Sub Total								£909.80	£909.80
Grand Total								£2,401.44	£2,401.44

First Page

Comprised with a large header section to include the customer name and address plus invoice details and a small footer that contains the subtotal and the marketing message.

Continuation Page

Comprised with a small header and a small footer

Last Page

Comprised of a small header and a large footer containing invoice total information.

Single Page

If the source data only contains a single page of item lines then the page will contain a large header and footer.

Subtotals on Page

At the bottom of each page that flows onto another page then there must be a subtotal displayed. The example below illustrates the requirement.

003397	QUALITY TRACING PAPER A2 63GSM PK250	2	19.45	6.67	38.12	A
003455	TRACING PAPER GREASEPROOF A4 PK500	2	4.65	1.59	9.11	A
					Sub Total	
					£721.84	

Continued



As the source data is repaginated the subtotal must be controlled by the Transform process, taking care to ensure the sub totals are contained in each Invoice within a batch.

In addition Continued must appear to the left of the subtotal page on all but the last or single pages.

Conditional Language Labels

As the company trades internationally there is a requirement to produce the invoices in the language of the target audience. Whilst the content of the source data remains in English the labels on the document need to be conditional, based on a language code in the source file.

Within a batch the language can change so it is essential that the language changes with each document. If a language code is missing in the source data or the translation has not been completed within the Transform process then the default language should be English.

Label Values

The following table shows the labels and values that need to be created for each language.

Label	English	French	German
Title	Invoice	Facture	Rechnung
InvoiceTo	Invoice To	Facture Pour	Rechnung an
Invoice No	Invoice No	No de Facture	Rechnungsnummer
InvoiceDate	Invoice Date	Date de la Facture	Rechnungsdatum
AccountNo	Account No	No de compte	Kontonummer
OurOrderRef	Our Order Ref	Notre Ordre ref	Unsere Auftrags
TransactionDetails	Transaction Details	Détails de la Transction	Details der Transaktion
Page	Page	Page	Seite
Copy	Copy	Copie	Kopie
Of	of	sur	von
ItemCode	Item Code	Code de l'article	Artikel-Code
ItemDescription	Item Description	Description de l'objet	Arikel Beschreibung
Quantity	Quantity	Quantité	Menge
UnitPrice	Unit Price	Prix Unitaire	Einheitspreis
Tax	VAT	TVA	Mwst
Total Value	Total Value	Valeur Totale	Gesamtwert
Tax Rate	VAT Rate	Taux de TVA	Steuersatz
Continued	Continued	Continuer	Fortsetzen
SubTotal	Subtotal	Sous-Total	Zwischensumme
PaymentTerms	Payment Terms	Conditions de paiement	Zahlungsbedingungen
Days	Days	Jours	Tage
PleasePayBy	Please pay by	S'il vous plaît payer par	Bitte achten Sie durch
NetTotal	Net Total	Total net	Gesamt keine MwSt
LanguageCode	En-GB	fr	de


Results

The image below illustrate the header with required language labels.


Invoice

Invoice To

Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX



Bottomline Technologies®
Bottomline Technologies
115 Chatham Street
Reading
Berkshire
RG1 7JX
Tel: +44 (0)118 222000




Invoice No
5762746

Invoice Date
02/02/14

Account No
450663 /000

Our Order Ref
4357953/ 0



Transaction Details


Page 1 of 3

Item Code	Item Description	Quantity	Unit Price	VAT	Total Value	VAT Rate
-----------	------------------	----------	------------	-----	-------------	----------

Rechnung

Rechnung an

Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX



Bottomline Technologies®
Bottomline Technologies
115 Chatham Street
Reading
Berkshire
RG1 7JX
Tel: +44 (0)118 222000




Rechnungsnummer
5762746

Rechnungsdatum
02/02/14

Kontonummer
450663 /000

Unsere Auftrags
4357953/ 0



Details der Transaktion


Seite 1 of 2

Artikel-Code	Artikel Beschreibung	Menge	Einheitspreis	Steuer	Gesamtwert	Steuersatz
--------------	----------------------	-------	---------------	--------	------------	------------


Facture

Facture Pour

Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX



Bottomline Technologies®
Bottomline Technologies
115 Chatham Street
Reading
Berkshire
RG1 7JX
Tel: +44 (0)118 222000




No de Facture
5762746

Date de la Facture
02/02/14

No de compte
450663 /000

Notre Ordre ref
4357953/ 0



Détails de la Transction

Page 1 sur 2

Code de l'article	Description de l'objet	Quantité	Prix Unitaire	TVA	Valeur Totale	Taux de TVA
-------------------	------------------------	----------	---------------	-----	---------------	-------------

Currently there is a requirement for three languages to be supported by the Transform process.

- English
- French
- German

Dynamic Images

There is a requirement to make use of external images applied to the document layout according to the conditions current at run time, project logic will be used to test the prevailing condition, and then call the appropriate image. There are two areas where the images will be applied to the document.

Marketing Messages

On each page of the document a footer marketing message must be applied. The messages will change on each page of the document to a maximum of four pages these message will be supplied by the art department and published to an images directory. If an invoice exceeds four pages then the Default image should be used.

The image naming convention will be based on page number where the message will be placed.

- 1.bmp
- 2.bmp
- 3.bmp
- 4.bmp
- Default.bmp

National Flag

In the header to identify the output language the national flag image will be placed beneath the corporate details. The naming convention of the images will match the language code read from the header section of the source data.

```

ET:01      LN FR      Language      5762751
                                     02/02/14
                                     03/05/14
Page 1 of 1      4357953/ 0      450663 /000
***COPY***
***COPY***
Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX
  
```

Result

Pick up the following image **FR.jpg**

Facture		Bottomline Technologies®		No de Facture	
Facture Pour		Bottomline Technologies		5762746	
Bottomline Technologies Ltd		115 Chatham Street		Date de la Facture	
115 Chatham Street		Reading		02/02/14	
Reading		Berkshire		No de compte	
Berkshire		RG1 7JX		450663 /000	
RG1 7JX		Tel: +44 (0)118 222000		Notre Ordre ref	
				4357953/ 0	
Détails de la Transction					
Code de l'article	Description de l'objet	Quantité	Prix Unitaire	TVA	Valeur Totale Taux de TVA

QR Barcode

There is a requirement from marketing to place a Quick Response (QR) Barcode on the header of each invoice, the content of the barcode will be taken from an external lookup text file to return a web address (uniform resource locator). The key for the lookup will be the EstType field in the header section of the source input file.

```

ET 01 LN:FR 5762751
                                     02/02/14
                                     03/05/14
                                     450663 /000

***COPY***
***COPY***

Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX
  
```

Diagram: A box labeled 'EstType' is connected by a line to the '01' in the 'ET 01' field of the text above.

Page 1 of 1
4357953/ 0

The external lookup file will be maintained by the marketing department. The format of the file is explained below.

ESTTYPE = Uniform resource locator (URL)

Example

01=http://www.bottomline.co.uk/document_management/transform.html

Result

Facture		Bottomline Technologies®		No de Facture	
Facture Pour Bottomline Technologies Ltd 115 Chatham Street Reading Berkshire RG1 7JX		Bottomline Technologies 115 Chatham Street Reading Berkshire RG1 7JX Tel: +44 (0)118 222000		5762746	
				Date de la Facture	
				02/02/14	
				No de compte	
				450663 /000	
				Notre Ordre ref	
				4357953/ 0	
Détails de la Transction					
Code de l'article	Description de l'objet	Quantité	Prix Unitaire	TVA	Valeur Totale Taux de TVA

Page 1 sur 2

Payment Terms

Currently the source data contains the invoice date and payment due date, however the accounts department want the payment terms to be displayed on the final page footer.

Payment Terms 90 days Please pay by 03/05/14		Net Total £2,045.82
Bottomline Technologies education services are proud to bring you a working example of a Plain Text batch Invoice file.		VAT £358.04
   		Total £2,403.86

Make it Earth Day *Every Day* with Bottomline Technologies

Learn more about Bottomline Technologies' earth-friendly solutions

[LEARN MORE](#)

The Transform process will need to calculate the number of days between the invoice date and payment due date and return the result to a string to present on the footer.



OMR Marks

The central print facility makes use of an automated envelope inserter. The inserter requires Optical Mark Recognition (OMR) marks on the documents to differentiate between invoices in a batch.

The Transform process will need to adhere to the following specification.

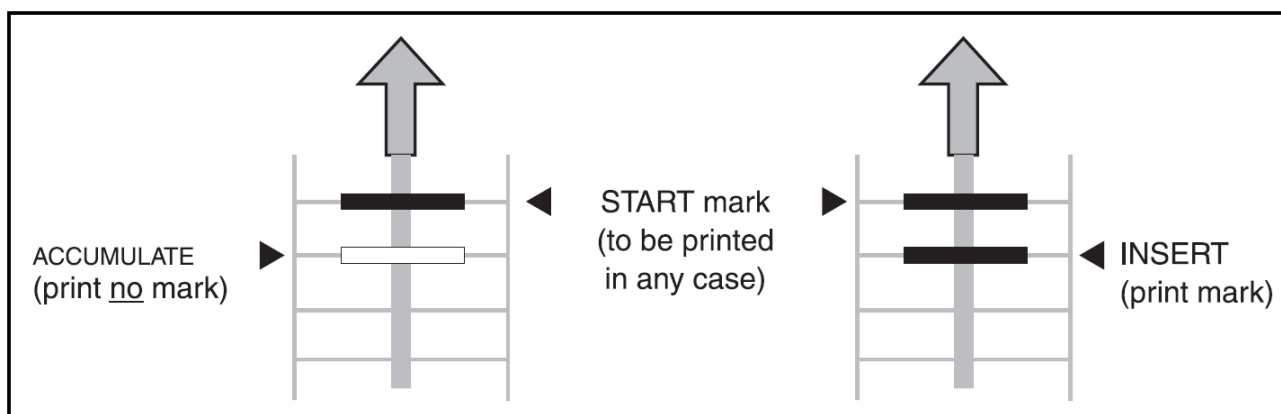
The minimum code comprises two commands: START and INSERT.

START

The “Start” command begins the read activity. This mark must be printed on every sheet. The distance of this mark from the top edge of the sheet must be 3.5 cm

INSERT

If there is no mark printed in this line of code, the sheet in question will be collated. If the mark is set, the sheets so far will be folded and inserted. The INSERT command also marks the end of the document in the paper stack.



START (always print mark in the first code line)

ACCUMULATE (do not print mark in the second code line)

INSERT (print mark in the second code line)

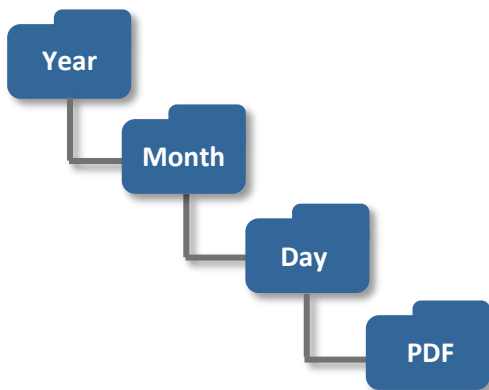
Output

The project must meet the following delivery requirements.

- PDF format written to specified folders with unique naming convention.
- Conditional XML output is required for certain account numbers maintained outside the host data stream in an external CSV file.
- Audit of each invoice processed appended to daily log file.

PDF Format

All documents are to be produced in PDF format and written to the following directory structure.

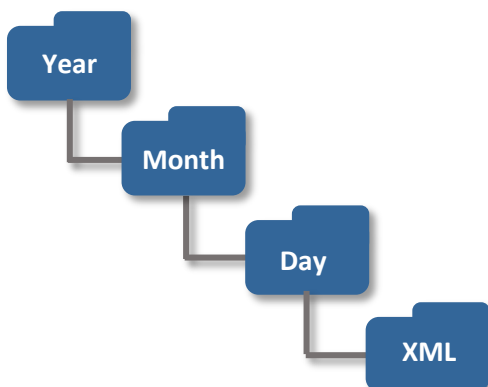


The PDF filename convention will be constructed as follows

Language_AccountNo_InvDate_YYYYMMdd_HHmmss.pdf

Conditional XML

Conditional XML output is required for account numbers maintained outside the source data stream as an external list containing the account numbers that require XML output. The files will be written to the following directory structure.



The XML filename convention will be constructed as follows.

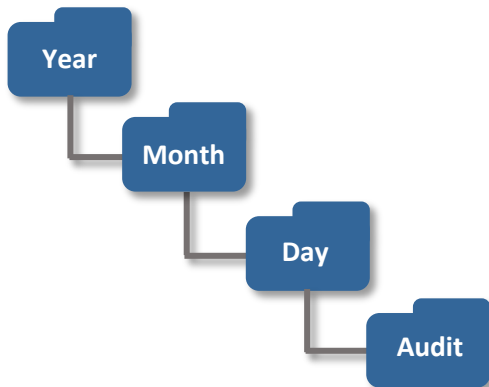
AccountNo_InvDate_YYYYMMdd_HHmmss.xml

Audit

The accounts department want to receive a daily audit of the invoices processed. The audit file will be produced in comma separated values, format described in the example below.

	A	B	C	D	E	F	G
1	Invoice Date	InvoiceNo	AccountNo	NetTotal	Tax	Total	EstType
2	02/02/2014	5762745	450663/000	2709.64	474.19	3183.83	1
3							

The files will be written to the following directory structure.



The Audit filename convention will be constructed as follows.

InvoiceAudit_YYYYMMdd.csv

Version 2.1

Transform[®]

*Plain Text Exercise
Walk Through Guide*



Introduction

This guide is designed to accompany the Transform Practical Introduction to Project Development Plain Text Exercise. It provides step by step instructions on how to build the project covered in the training session.

Each element is introduced with a Project Checklist to identify the steps required to complete the process.

In some cases the guide will explore alternate techniques to demonstrate different ways to achieve the same result.

The goal of the exercise is to introduce you to different techniques whilst building a complete Transform project.



Project Checklist

1. Repository Design

Branch Relationship Diagram

Branch Files Required

Repository Structure

2. Create Local Repository
3. Language Labels
4. Global Resources
5. Plain Text Template Wizard
6. Plain Text to Container
7. Modify Branch Structure
8. Dynamic Branch Shortcut
9. Document Organizer
10. Synchronize and Deploy



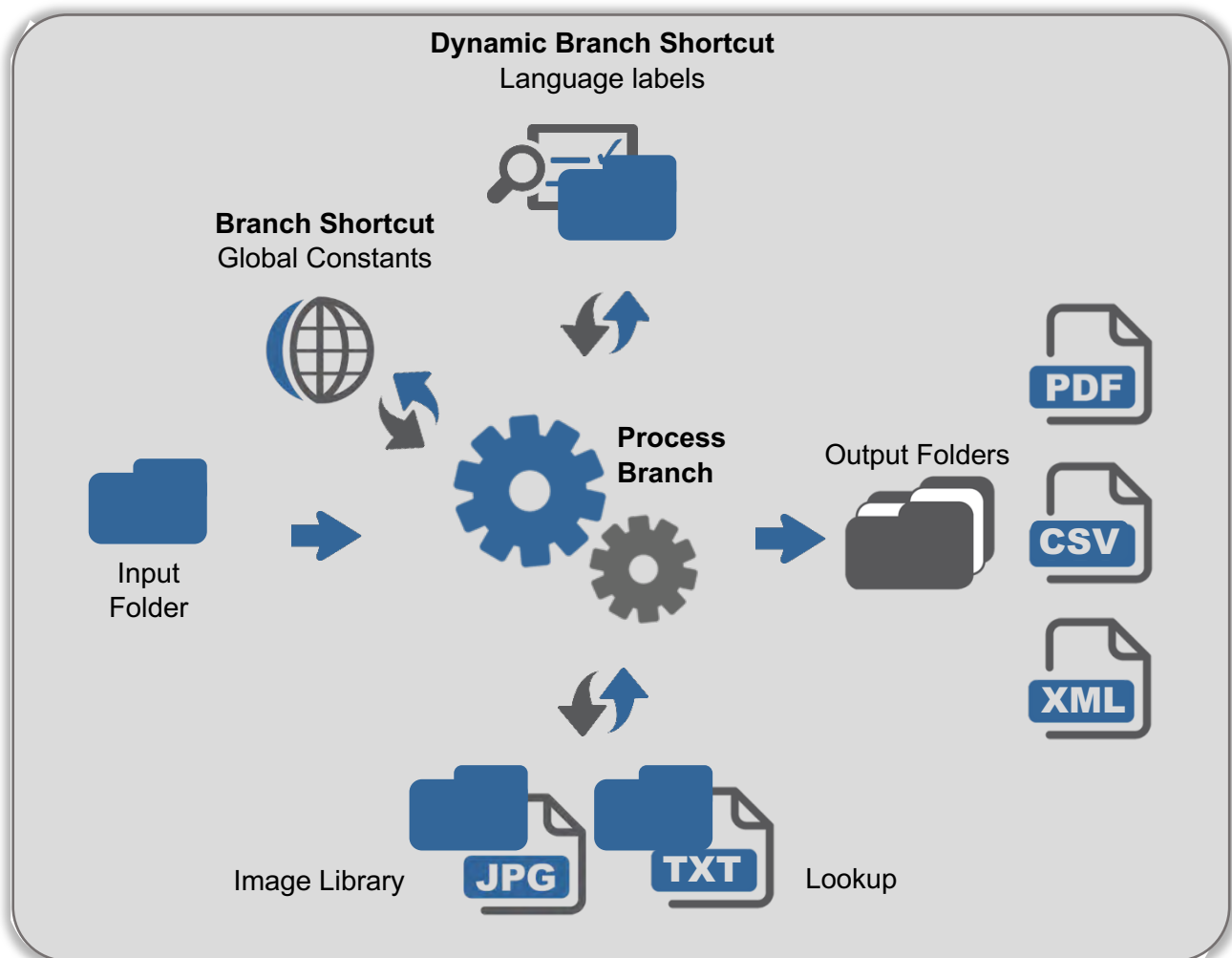
Repository Design

Now we understand the requirements we need to first consider our Repository design. A repository is nothing more than a standard directory except that the last four characters of the repository directory's name is **".fsr"**. Although this may look like a file extension, it is not since directories do not have extensions. The **".fsr"** is used to indicate to the Transform that this directory is the root directory of a repository directory structure. A local repository can appear anywhere within a system's file system and is not limited to any particular parent directory. This becomes a developer's local copy of an application and is where all development and testing can be performed.

A subdirectory immediately beneath a repository directory is called a package directory, usually referred to simply as a **"package"**. Packages provide a way to keep branch files and their associated files organized in a manageable structure. What is actually placed in these packages is up to the developer. A branch file cannot be deployed to the Runtime Server unless it resides inside a package directory.

Package directories may have subdirectories depending on how the developer designs the application. There is a special class of branch file, a Process branch, which is started automatically by the Runtime Server, that branch **must** reside in a subdirectory of a package directory and that subdirectory must be named **"processes"**.

The following diagram illustrates the branch relationships required to complete our project.



Branch Files Required

When designing the project we should always keep one eye on the future so that the solution can meet the demands of the business moving forward. Planning and then implementing global or dynamic branches for the first project installed can make branch file development easier because you don't have to create the same objects and enter the same data for each branch you create. The biggest advantage to using these is what happens when a declared element value must change. Change the declaration within a global branch and then deploy that branch to change every branch file that uses that declaration.

For this project we are going to utilise the following branches.

- **Scheduled Process Branch** – Build output directory structure
- **Queue Process Branch** – Invoice branch
- **Dynamic Branch Shortcuts** – Language labels
- **Branch Shortcut** - Global constants

Scheduled Process Branch

To create the output windows folder structure a scheduled process branch will run at 00:01 every day. As a process branch once deployed to the Runtime server will run as a Windows task. All process branch files must be saved in the \Processes subfolder of a package.

Queue Process Branch

The queue process branch will use a file queue to watch a folder for the plain text positional source files. As a process branch once deployed to the Runtime server will run as a Windows task and will be responsible for the data acquisition step of our workflow.

Dynamic Branch Shortcut

In order to produce conditional language labels we are planning to make use of Dynamic Branch Shortcuts. The input data source includes a language field that will be used to call the appropriate branch containing the document labels for the required language.

All the branches that can be called by a single Dynamic Branch Shortcut must be saved in the same package subfolder. This subfolder must have a name with the extension **.fdbbs**.

The **.fdbbs** subdirectory must also contain a default branch, which is called if the dynamically called branch cannot be found. The input and output parameters that are available to this default branch are also those that are available to the branches that can be called dynamically from the same Dynamic Branch Shortcut.

In our example we need to create four Simple Action branches.

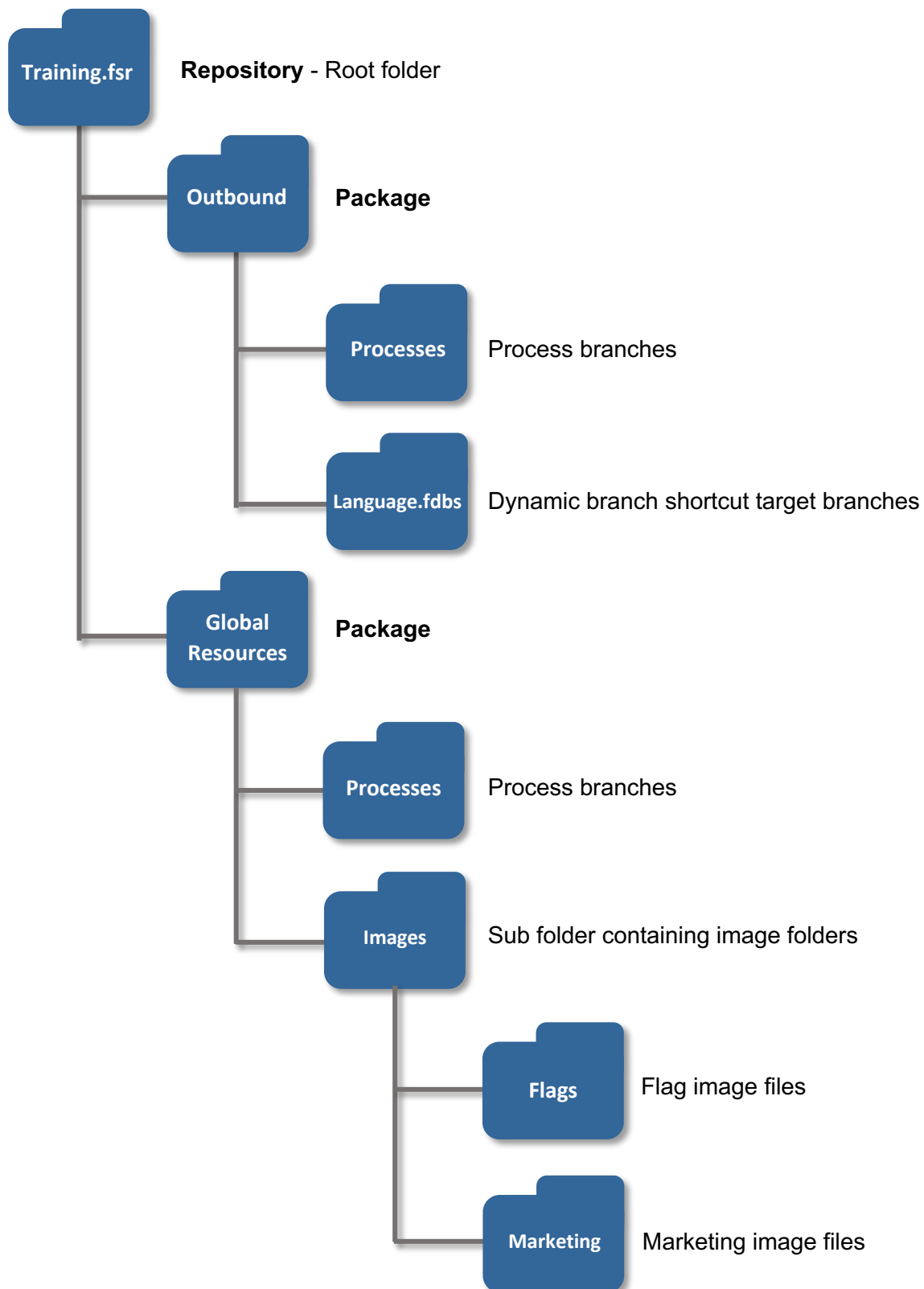
- Default.fsp - Default
- GB.fsp - English
- FR.fsp - French
- DE.fsp - German

Global Constants

The project will require external resources to read from and write to. It makes sense to create a project to store the paths to these resources so they can be maintained centrally. This branch will be called via a branch shortcut from the process branch.

Repository Structure

The Repository structure we are going to create for our project is illustrated below.



Project Checklist

1. ~~Repository Design~~
2. **Create Local Repository**
 - Open Synchronizer program**
 - Create Repository**
 - Create Packages**
 - Create Subfolders**
3. ~~Language Labels~~
4. ~~Global Resources~~
5. ~~Plain Text Template Wizard~~
6. ~~Plain Text to Container~~
7. ~~Modify Branch Structure~~
8. ~~Dynamic Branch Shortcut~~
9. ~~Document Organizer~~
10. ~~Synchronize and Deploy~~



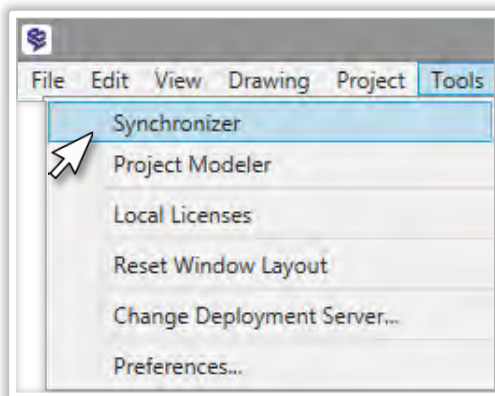
Create Local Repository

To create and manage the repository structure Transform makes use of the Synchronizer program.

The Synchronizer is a flexible file management program whose focus is to make management of a local repository and the transfer of files into a Deployment Server's repository as easy as possible. This is a program that does not automatically select the files to be synchronized. Instead, it places total control into your hands and you decide exactly which files you want to copy between sources.

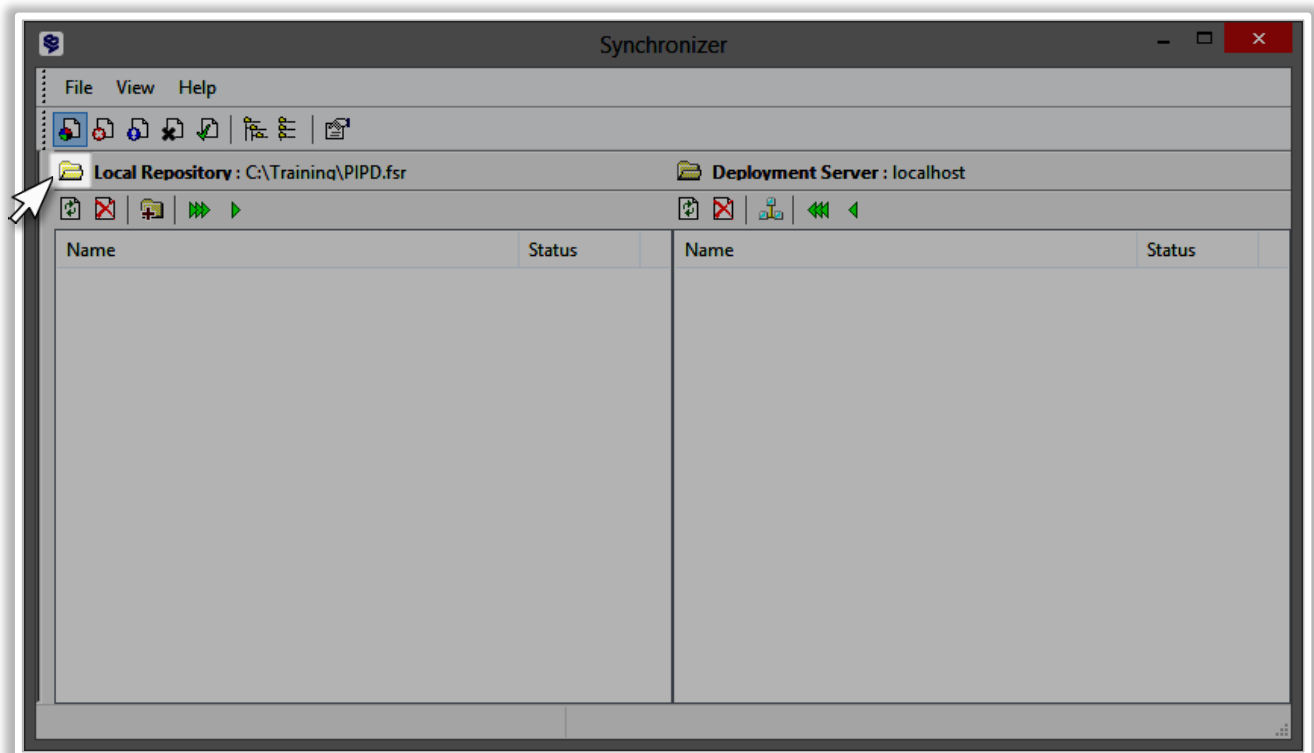
To open the Synchronizer from **Start > All Programs > Bottomline Technologies > Transform Client Tools > Transform Synchronizer**

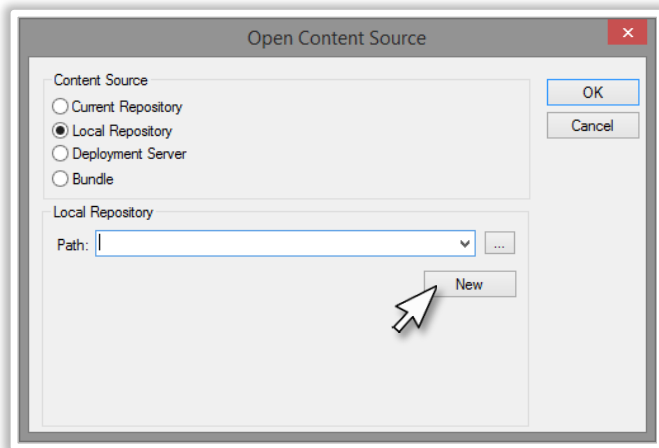
Alternatively you can open the Synchronizer from within the Transform Designer menu bar select **Tools > Synchronizer**.



Step 1 – Open Content Source

Select Open Content Source, click  on the toolbar in the left pane





Step 2 – New Local Repository

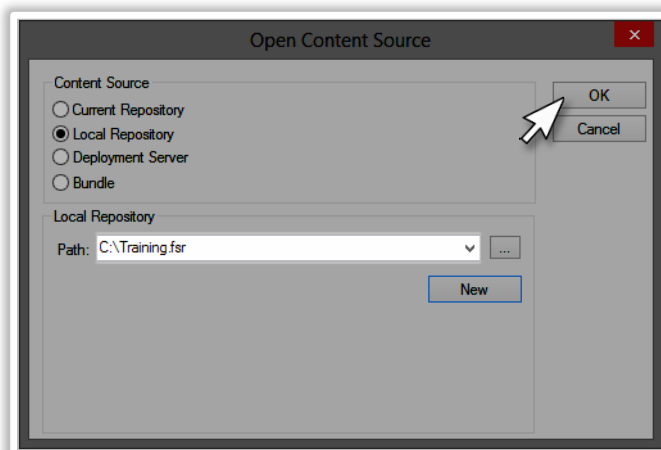
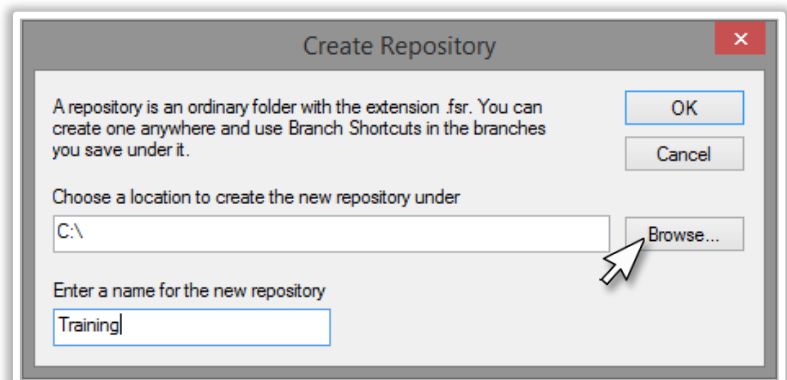
Select Local Repository radio button then click **New** button.

Step 3 – Create Repository

To Choose a location to create the new repository under click **Browse...**

Then enter a name for the new repository as shown.

Click **OK** to create the repository.



The Local Repository Path now displays the new Repository created.

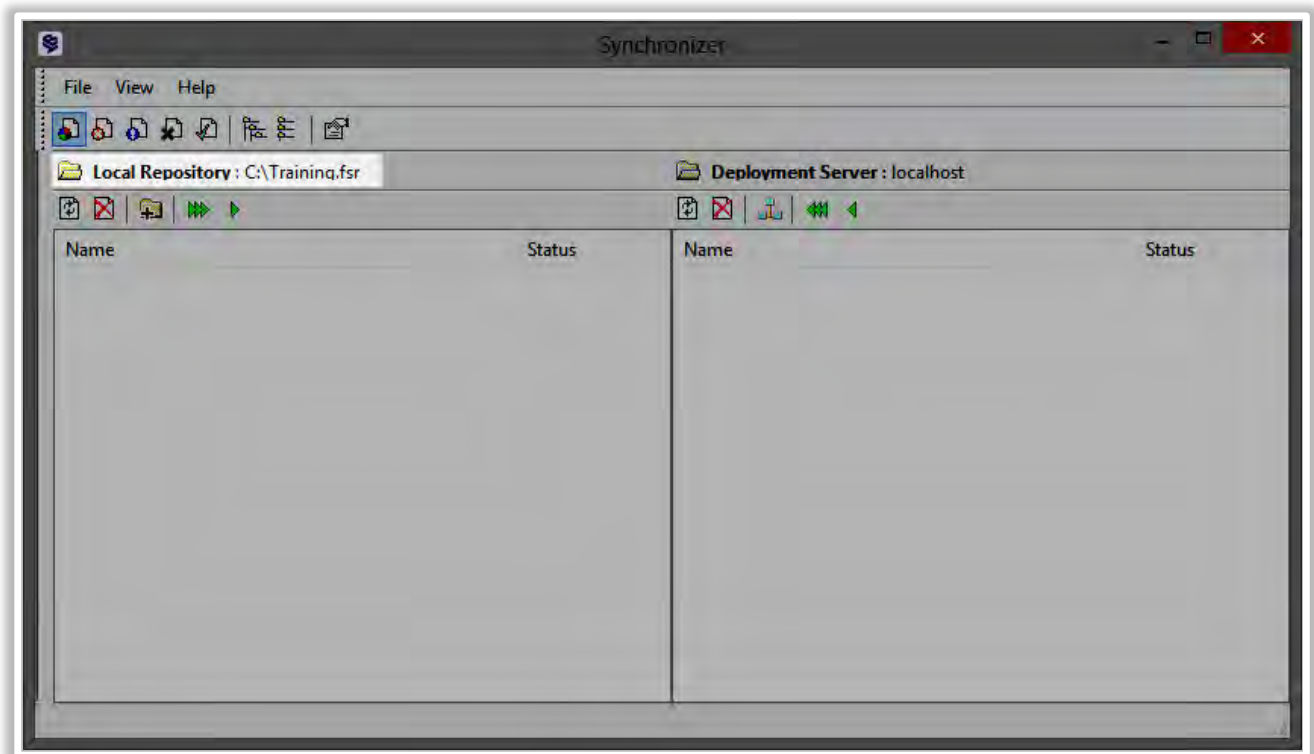
Click **OK** to close the Open Content Source dialog box.



Note

The Repository, Packages and files in the Synchronizer can be located and modified with Windows File Explorer.

The new Repository created is displayed as the current Local Repository in the Synchronizer window.

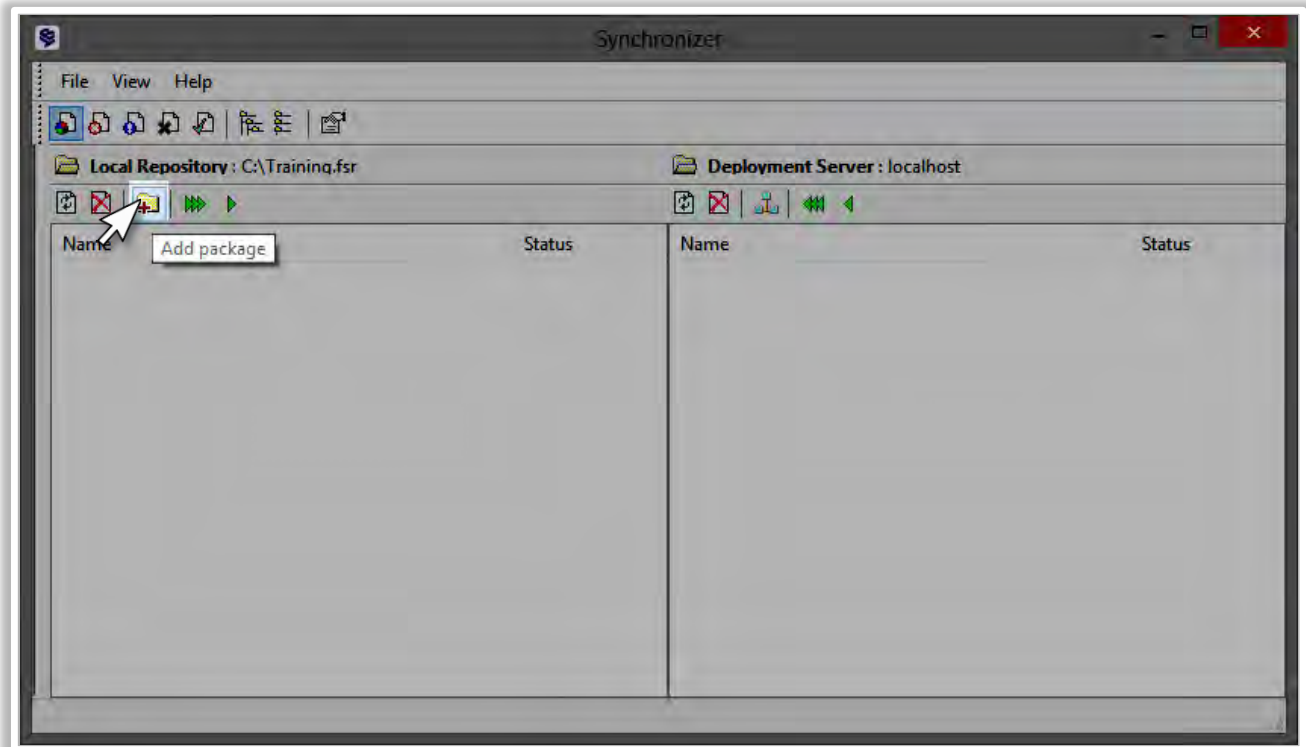


Create Packages

Now we have a Repository we need to create the following packages required for our project

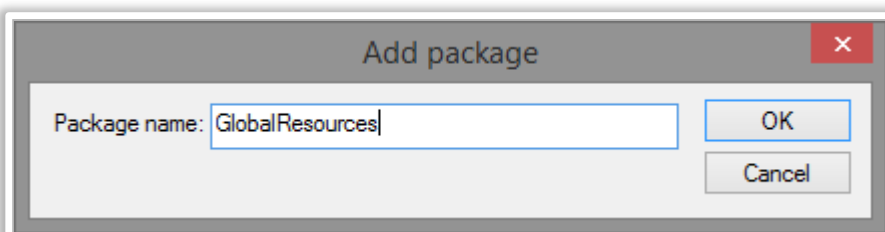
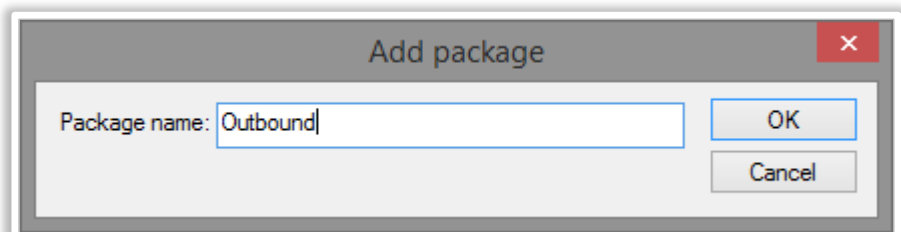
Step 1 – Add Package

To add a Package to the Repository click  on the toolbar.




Step 2

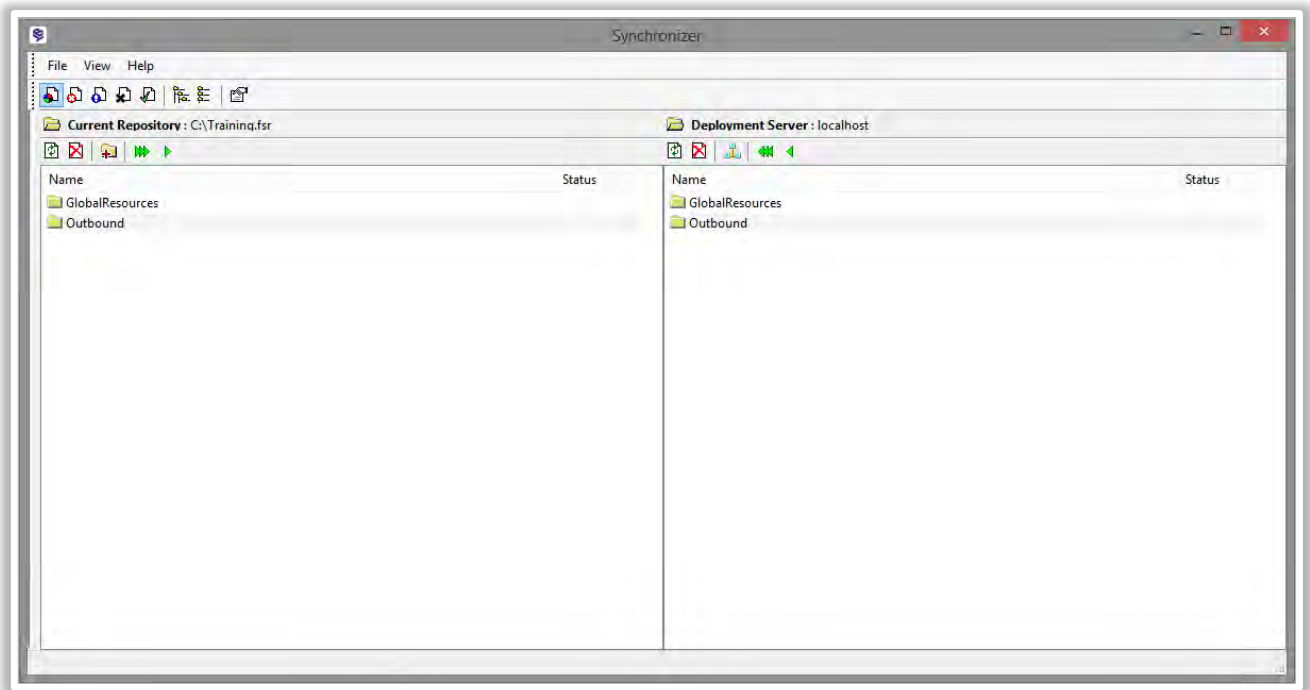
Type the new name in the Package name field within the Add package dialog box and click **OK**.



Step 3

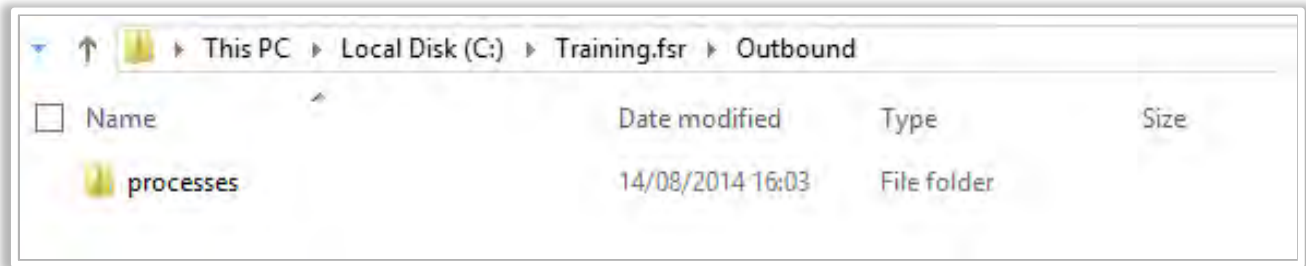
Add another Package click  on the toolbar and enter the package name. Click **OK**.

The Local Repository now contains two Packages as illustrated below



Create Subfolders

The Add Package process creates the folders in the Windows file system and by default creates a subfolder called processes in each package as illustrated below.



Step 1 – Create subfolders

From Windows File Explorer, browse to the Training.fsr repository directory, open to reveal the packages GlobalResources and Outbound sub directories.

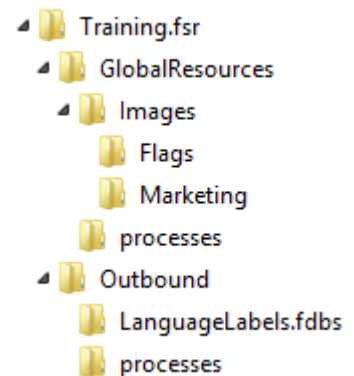
Create the following subfolders

GlobalResources

- GlobalResources\Images\Flags
- GlobalResources\Images\Marketing


Outbound

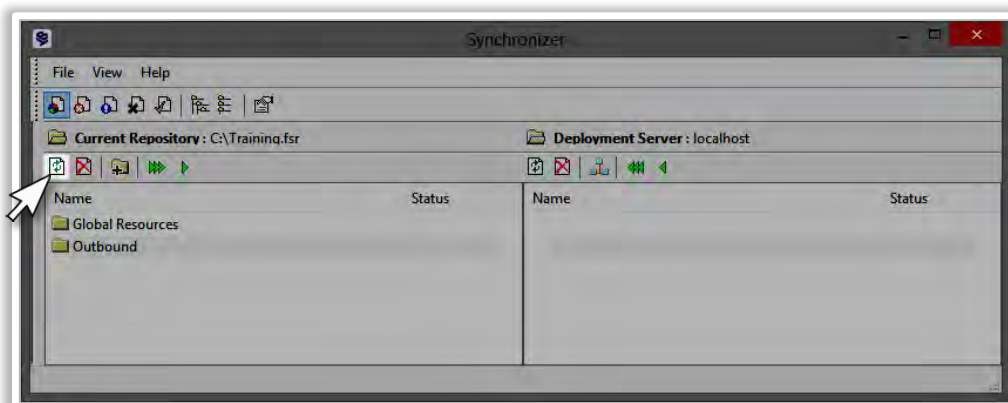
- Outbound\LanguageLabels.fdb




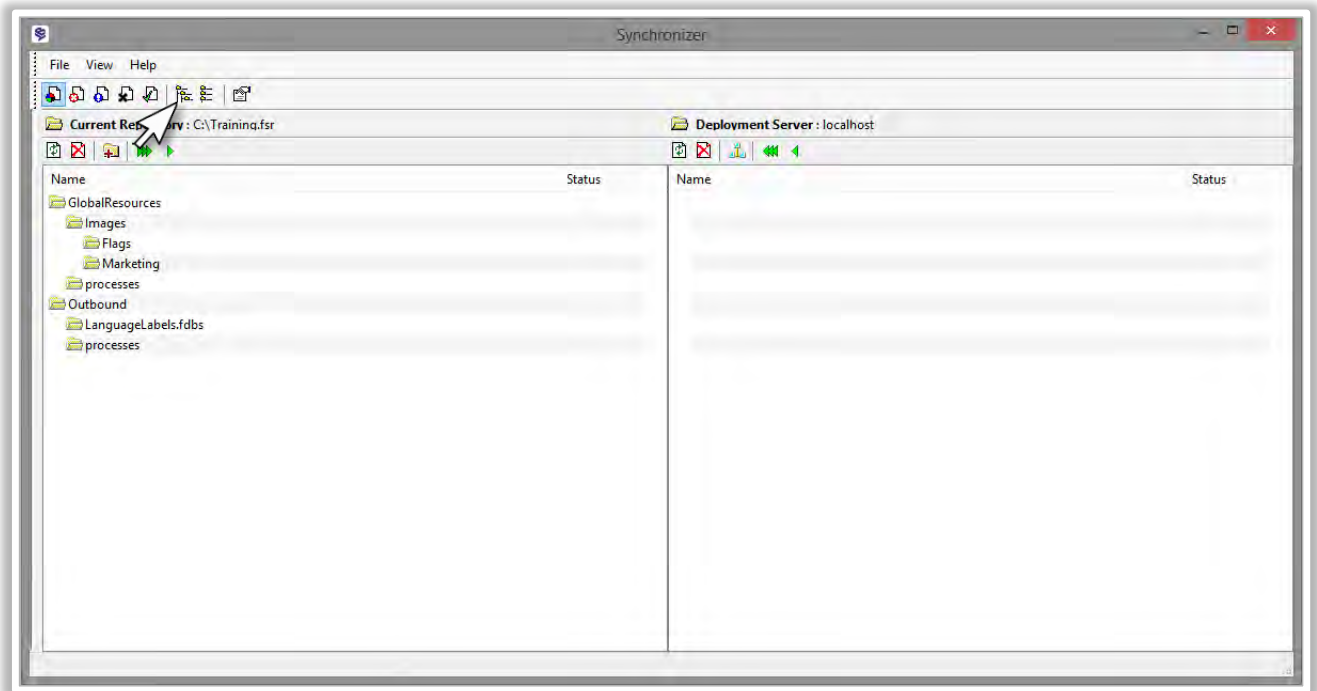
The Transform process recognises dynamic branch shortcuts only if they exist in a subfolder with the naming convention ***.fdb**, and the subfolder is located in the same package folder as the process calling branch.

Step 2 – Refresh Synchronizer view

Now use the Synchronizer tool to review the Repository structure. If the Synchronizer is already open then you need to click  Refresh source on the toolbar to update the view, as the application does not auto refresh.



To show the contents of the packages you can double click a package to open or click  expand all on the toolbar which expands all folders in the Repository.



Project Checklist

1. ~~Repository Design~~

2. ~~Create Local Repository~~

3. **Language Labels**

Draw Process

Open Simple Action Template

Define Parameters

Add Memory Objects

Empty Container

Copy Container

Add Unicode Text Objects

Add Log Event

Save in Repository Package

Create Language Branches GB.fsp, FR.fsp, DE.fsp

4. ~~Global Resources~~

5. ~~Plain Text Template Wizard~~

6. ~~Plain Text to Container~~

7. ~~Modify Branch Structure~~

8. ~~Dynamic Branch Shortcut~~

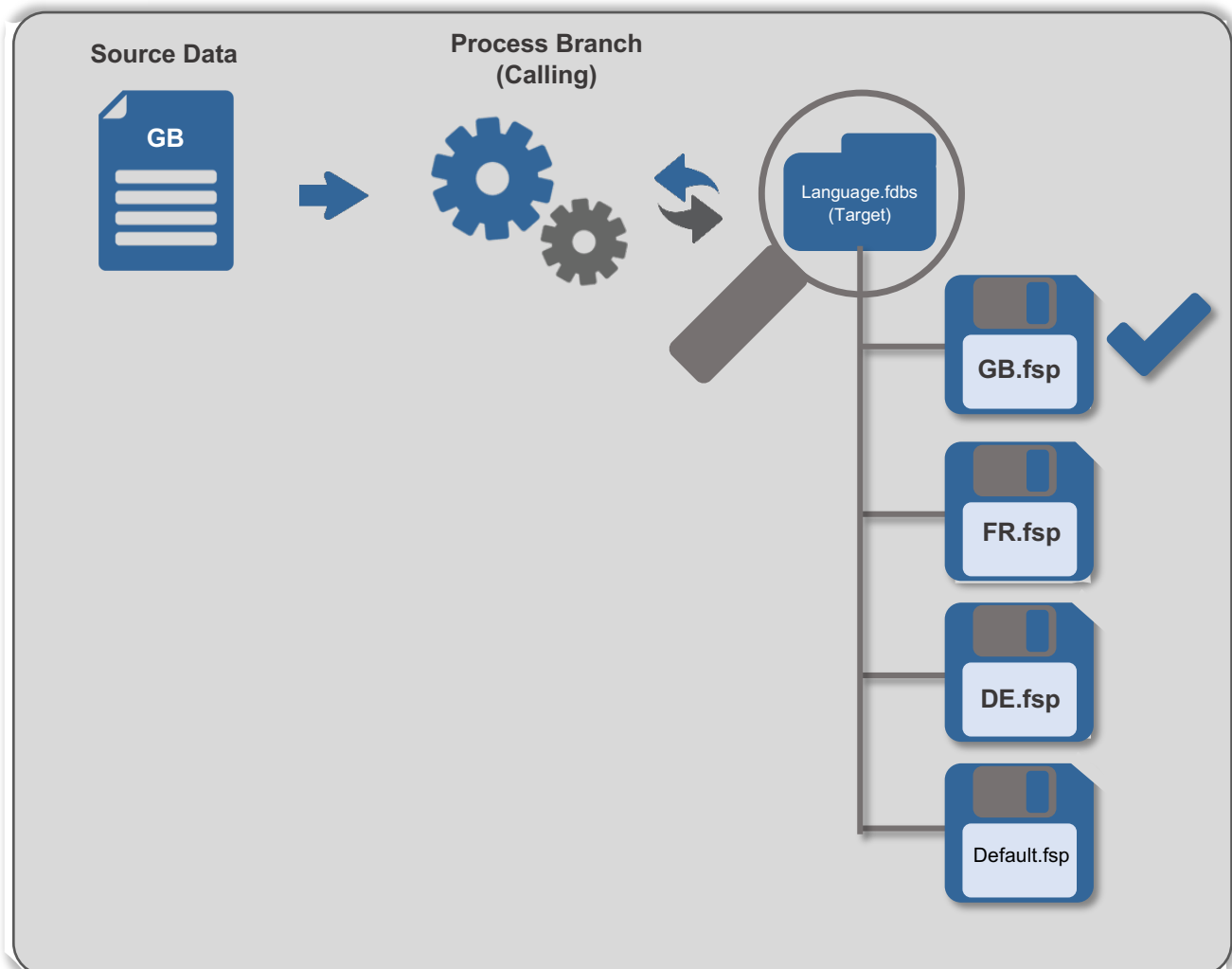
9. ~~Document Organizer~~

10. ~~Synchronize and Deploy~~



Create Language Label Branches

The technique we are going to use to produce the language labels is dynamic branch shortcuts. The following diagram depicts how this process will work.



In the example shown the source data file contains a language code 'GB'. The process branch, known as the calling branch, will extract the code and then execute a **Dynamic Branch Shortcut** object, passing the language code as the Dynamic Branch Name. The Dynamic Branch Shortcut, then inspects the **Language.fdb's** subfolder to verify there is a matching **GB.fsp** branch file, known as the target branch. If there is no matching branch in the **Language.fdb's** folder then the Dynamic Branch Shortcut will use the **Default.fsp** branch file as the target.

When called the target Dynamic Branch will write the label values into a Container Bag, which will be returned to the calling branch in the form of a parameter. A parameter represents a value passed to a branch, Passing parameters to a branch makes it more flexible and useful as a target.

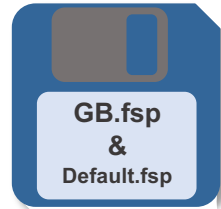
We are going to use three parameters in our target branches.

- **Bag** object container to store the language labels and pass them back to the calling branch
- **Text** object to pass the Language code.
- **System Information** object to pass the original calling branch for logging purposes.

The process of using Dynamic Branch Shortcuts is comparable to using subroutines in computer programming terms.

Label Values

The following table illustrates the labels and values that need to be created in each dynamic branch.



Label	English	French	German
Title	Invoice	Facture	Rechnung
InvoiceTo	Invoice To	Facture Pour	Rechnung an
Invoice No	Invoice No	No de Facture	Rechnungsnummer
InvoiceDate	Invoice Date	Date de la Facture	Rechnungsdatum
AccountNo	Account No	No de compte	Kontonummer
OurOrderRef	Our Order Ref	Notre Ordre ref	Unsere Auftrags
TransactionDetails	Transaction Details	Détails de la Transction	Details der Transaktion
Page	Page	Page	Seite
Of	of	sur	von
ItemCode	Item Code	Code de l'article	Artikel-Code
ItemDescription	Item Description	Description de l'objet	Arikel Beschreibung
Quantity	Quantity	Quantité	Menge
UnitPrice	Unit Price	Prix Unitaire	Einheitspreis
Tax	VAT	TVA	Steuer
Total Value	Total Value	Valeur Totale	Gesamtwert
Tax Rate	VAT Rate	Taux de TVA	Steuersatz
Continued	Continued	Continuer	Fortsetzen
SubTotal	Subtotal	Sous-Total	Zwischensumme
PaymentTerms	Payment Terms	Conditions de paiement	Zahlungsbedingungen
Days	Days	Jours	Tage
PleasePayBy	Please pay by	S'il vous plaît payer par	Bitte achten Sie durch
NetTotal	Net Total	Total net	Gesamt keine MwSt
Copy	Copy	Copie	Kopie

Default.fsp

When employing the Dynamic Branch Shortcut object a pre-requisite is the existence of a Default branch in the Dynamic Branch subfolder. It is good practice to create the default branch first as the parameters defined within will be used in all of the other target branches.

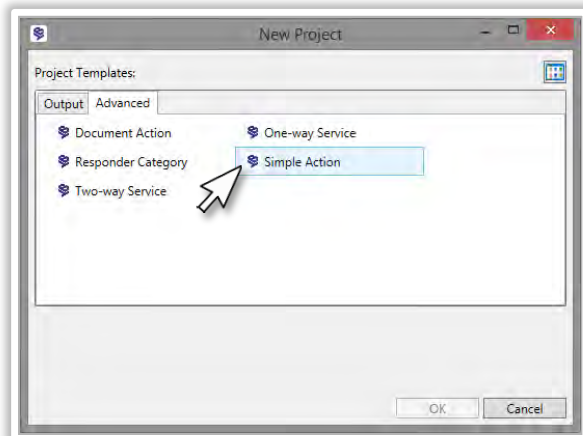
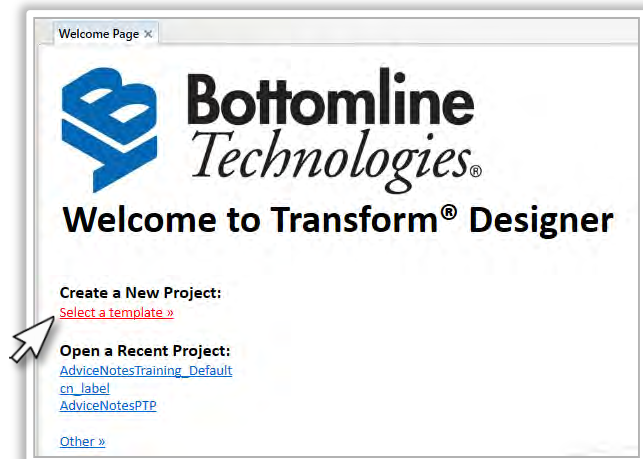
If the language code passed in the source data does not have a corresponding branch name in the dynamic branch shortcut folder then the default language should be English,

In addition to the label values, the default branch will log the language code received from the calling branch to identify to the project development team that that a language code passed does not have a corresponding branch.

The following provides the steps required to create the default branch.

Step 1 – Select Template

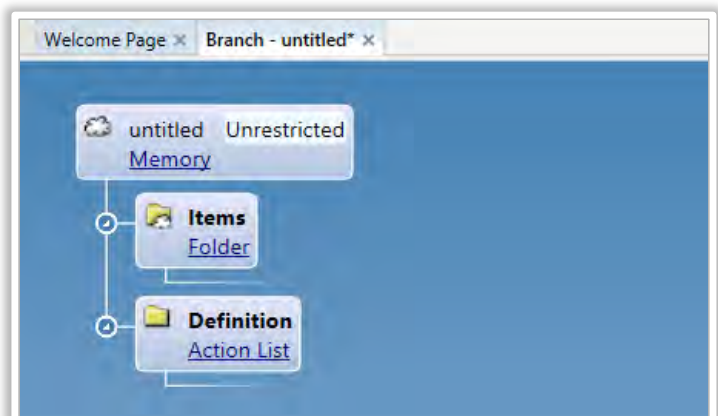
Open the Transform Designer and select **File > New** from the menu bar. Alternatively click **Select a template** from the Welcome page




Step 2 – Simple Action

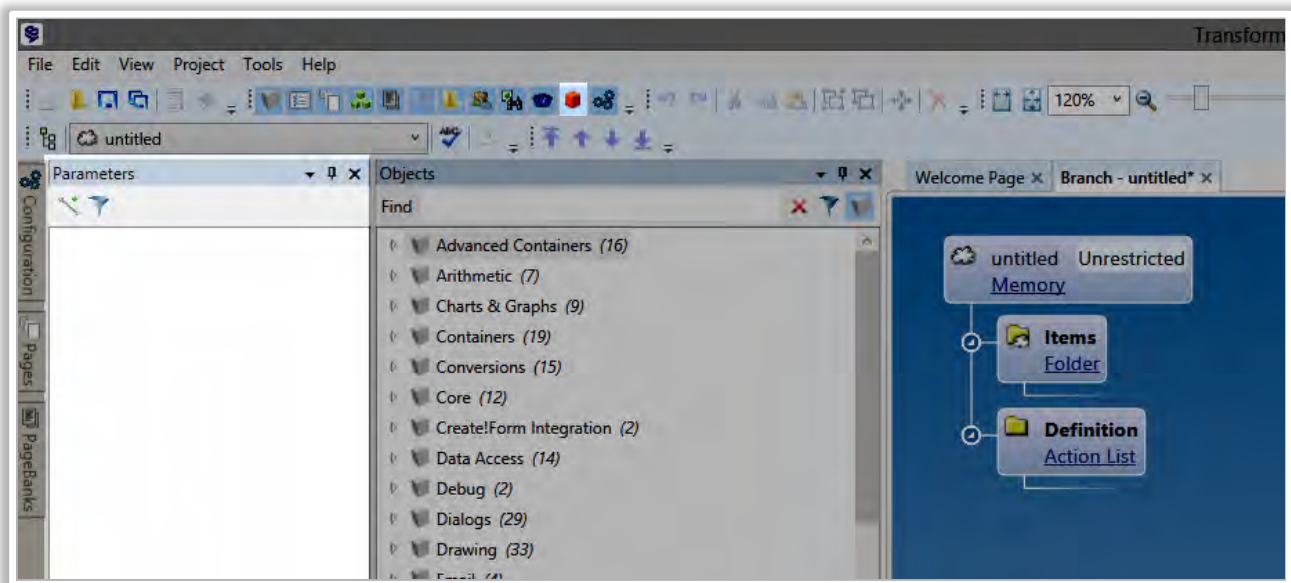
From the Advanced Tab select Simple Action and click **OK**.

This opens the branch editor view




Step 3 – Open Parameters window

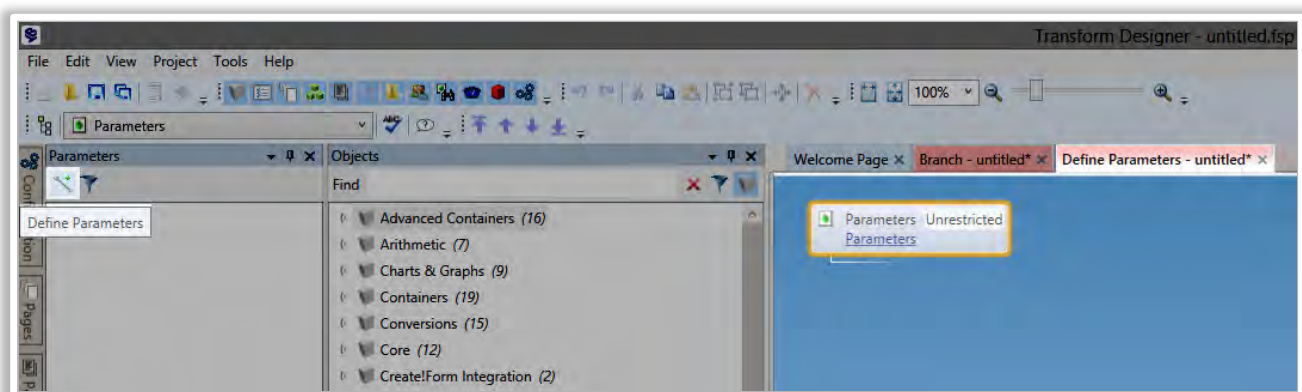
To create the parameters for the branch open the parameters window. From the menu bar select **View > Parameters**, alternatively on the view toolbar click 



By default the Parameters window opens on the left pane.

Step 4 – Define Parameters view

Click the Define Parameters  icon from the Parameters window toolbar. This opens the Define Parameters view.



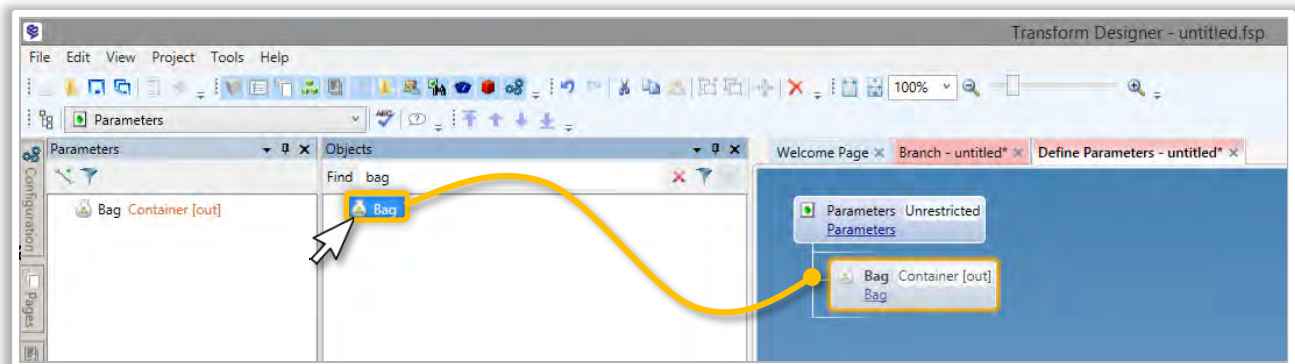
Step 5 – Define Parameters

From the Objects Palette Find field type bag. A **Bag** object is used as a temporary storage area for structured data, where we can write our language labels.



Step 6 – Add Bag Object

Drag the **Bag** object from the Objects palette onto the add point within the define parameters view.

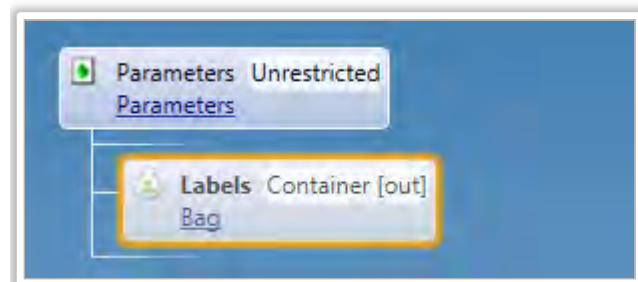


Step 7 – Rename Bag Object

As you add objects to a project give them meaningful names to help you keep track of them within the branch.

Rename the **Bag** object to **Labels**. To rename do one of the following

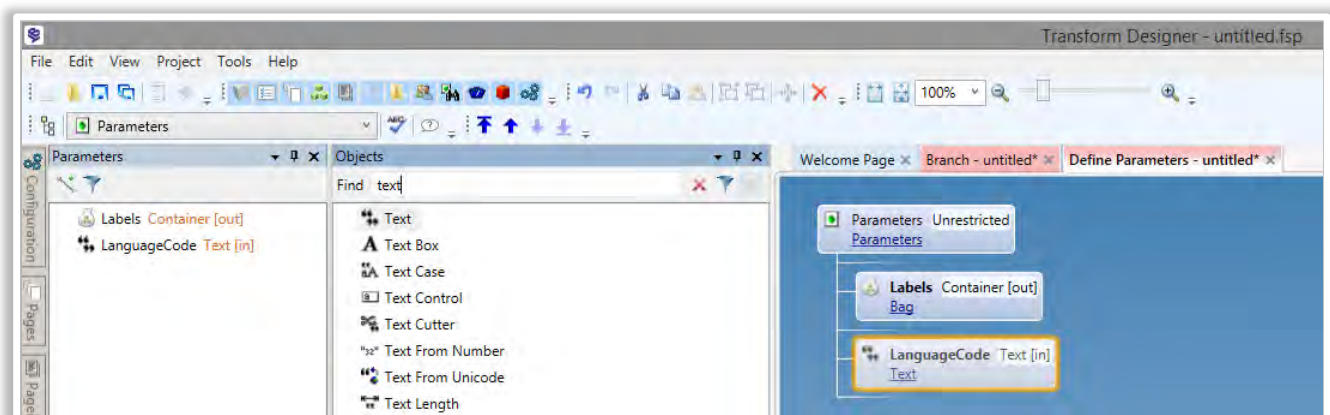
- Press F2
- Right click and select Rename from the menu
- Double click the existing label



Step 8 – Add Text Object

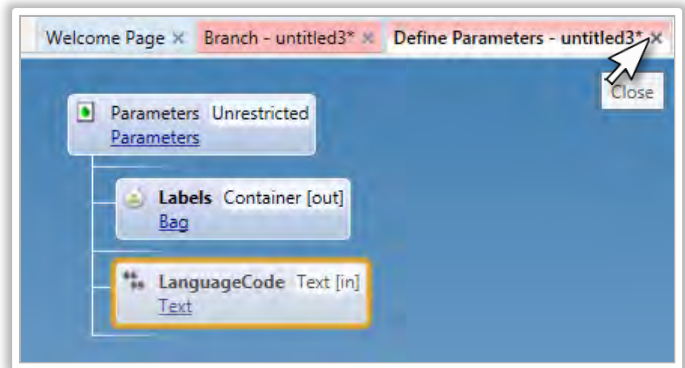
Add a **text** object to the parameter list, which will be used to read the language code from the client branch. This will be used to record the language code if the default branch is called, to identify to the development team that the target branch was not found.

From the Objects palette find field type text. Then drag the **Text** object under the **Bag** object in the parameter list, then rename to **LanguageCode**.



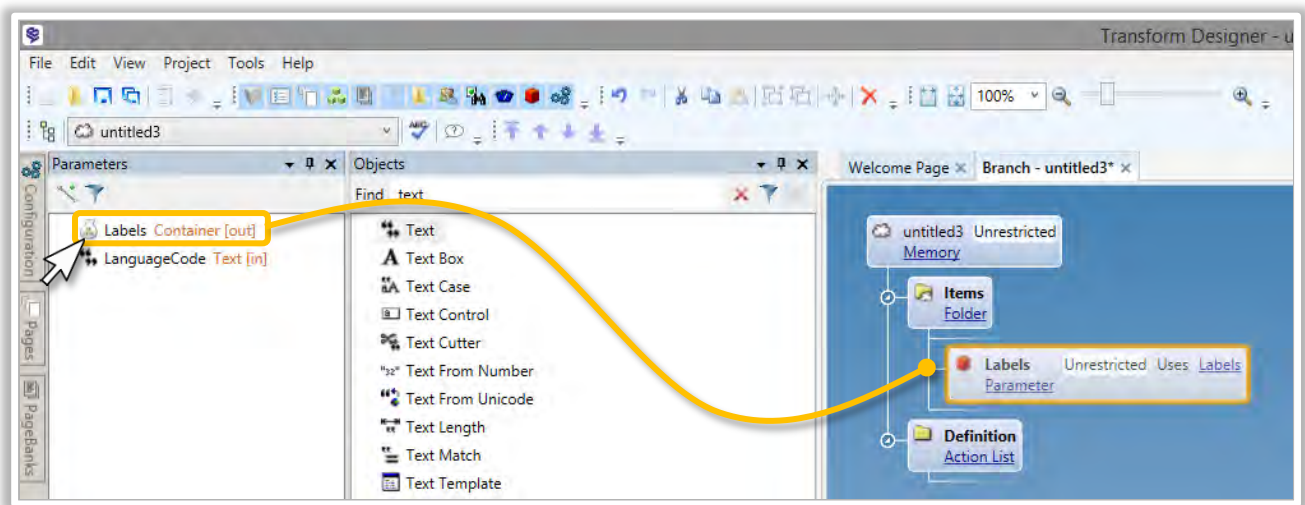
Step 9 – Close Define Parameters

Close the Define Parameters view, click the X on the Tab.



Step 10 – Create Parameter container

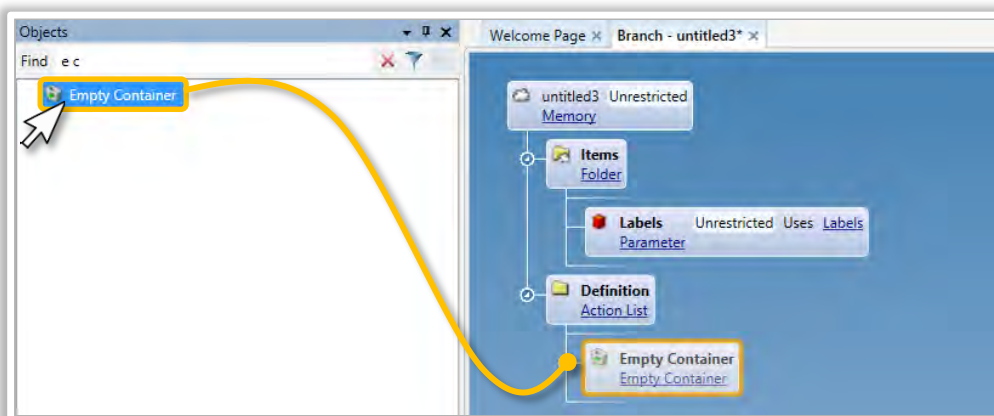
So that you can make it visible in the data palette as you create the labels. Drag the **Labels** Container parameter under the **Items** Folder on the branch.




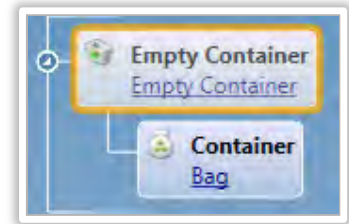
Step 11 – Add Empty Container Object

When the branch is called it will run an action to write the labels defined into the container, it is important to refresh the container every time the branch is called. The **Empty Container** object is used to perform this function.

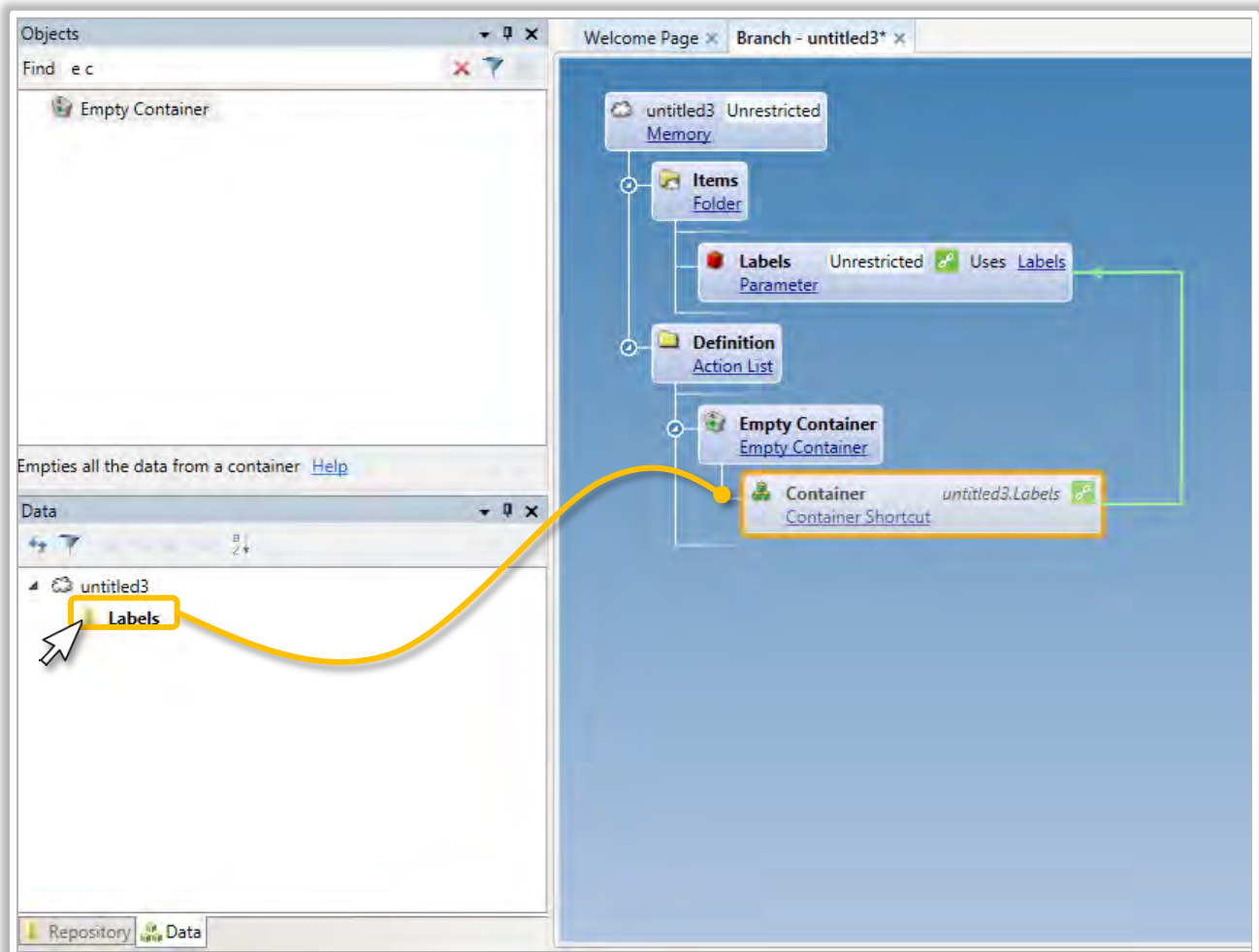
In the Objects palette Find field type **ec** and drag the **Empty Container** object under the Definition Action List




Expand the **Empty Container** object, click  expand control to reveal the child object. The default child object is a Container Bag as illustrated.



To empty the parameter container drag the Labels item from the Data palette over the Container Bag.

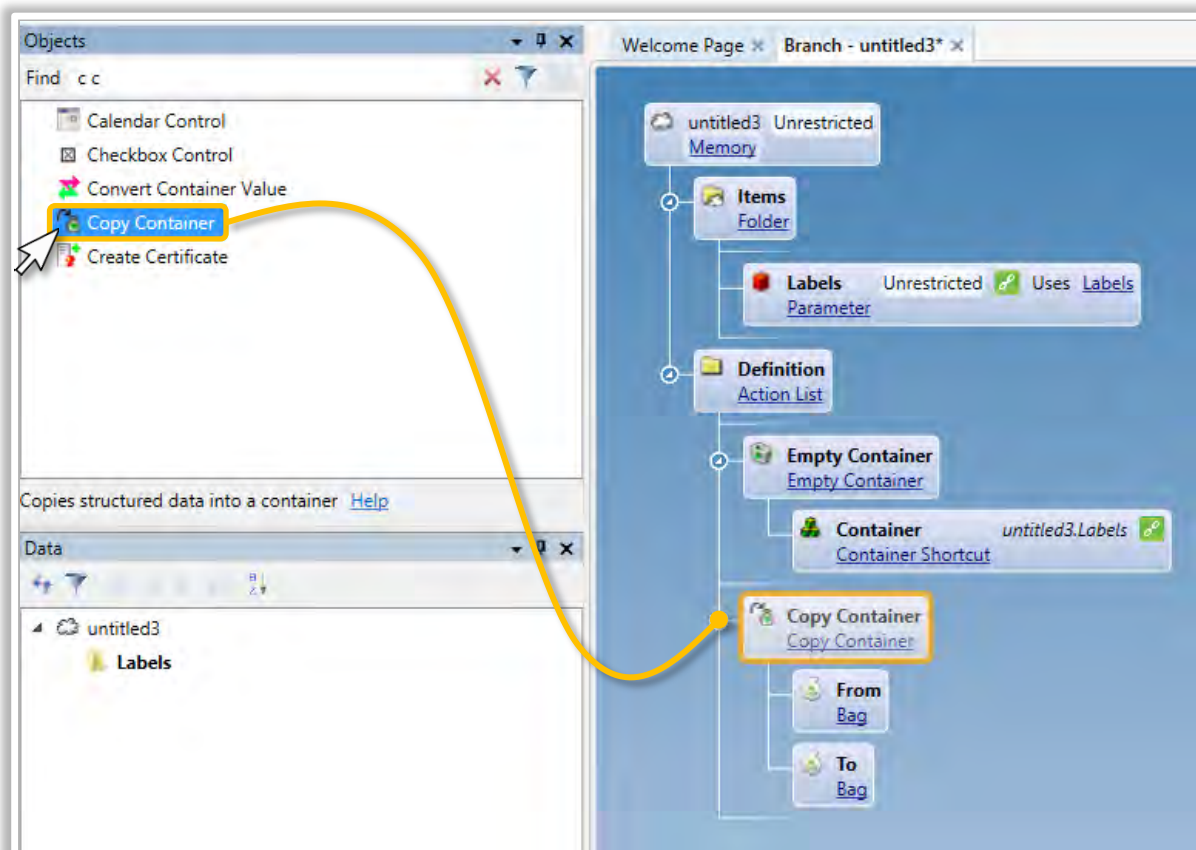


This creates a **Container Shortcut** which is presented on the branch by the link icon  and the green line showing the connection. A **Container Shortcut** stores the full path to an element inside a container, and then acts like the element it finds. It allows you to use such elements as if they were objects in the tree.

Step 12 – Add Copy Container Object

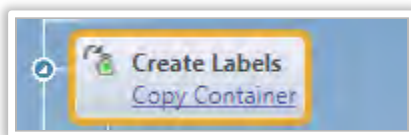
To write the labels and values into the container the **Copy Container** object will be used. The **Copy Container** object copies structured data from one container to another. The copied data is appended to any data that exists in the destination container, and does not overwrite the original data, hence we need to refresh the Container first.

From the Objects pallet Find field type **c c**. Select the **Copy Container** object from the list and drag it onto the branch beneath the Empty Container object so that it becomes the second action on the tree.



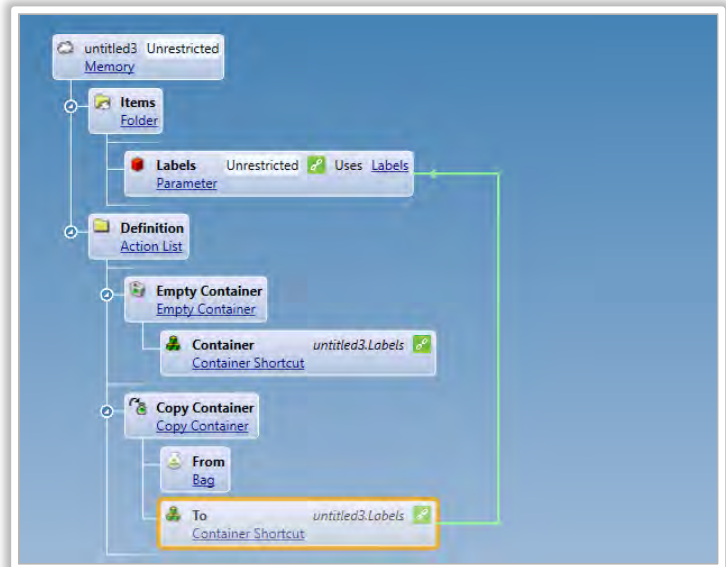
By default the **Copy Container** object has two child objects. **From** and **To** Bag, this allows you to define the source and target containers.

Rename the Copy Container object to Create Labels.



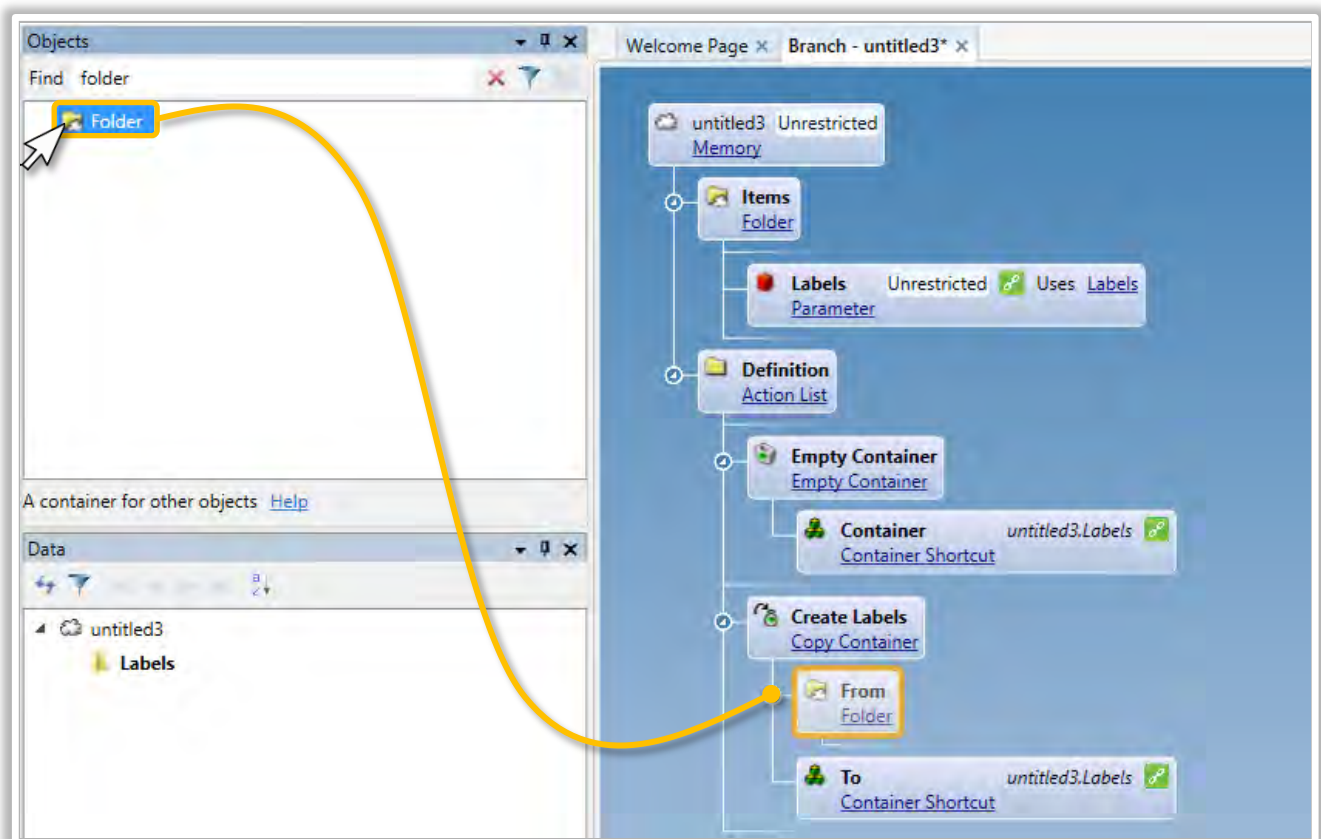
In our process the **From** will be the labels and values, which we will define in the next step. Whilst the **To** will be the parameter located in the memory folder list.

Create a Container Shortcut to the **Labels** item as described in **Step 11**.



Step 13 – Add Folder Object

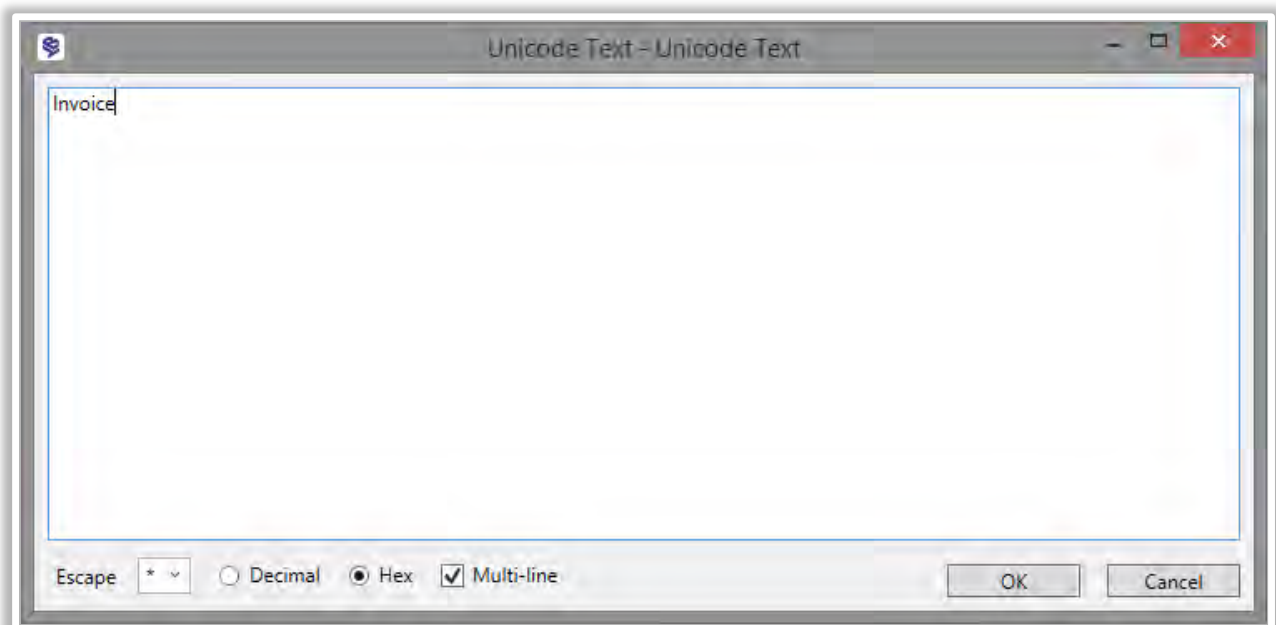
To create the labels and values change the From Bag to a Folder object. Type Folder into the Find field on the Objects Palette and drag it over the From Bag. A Folder is a general purpose container for other objects. Each child object becomes an element in the container



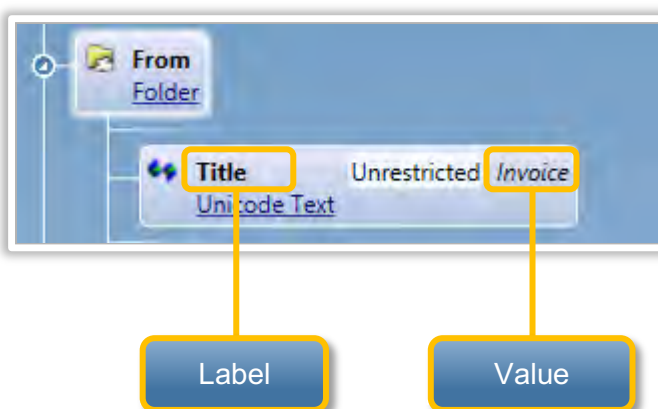
Add a **Unicode Text** object from the Objects palette under the Folder object as illustrated below.



Rename the object label from **Unicode Text** to **Title**, then click  icon to open the Unicode Text dialog box, then type Invoice as the value.



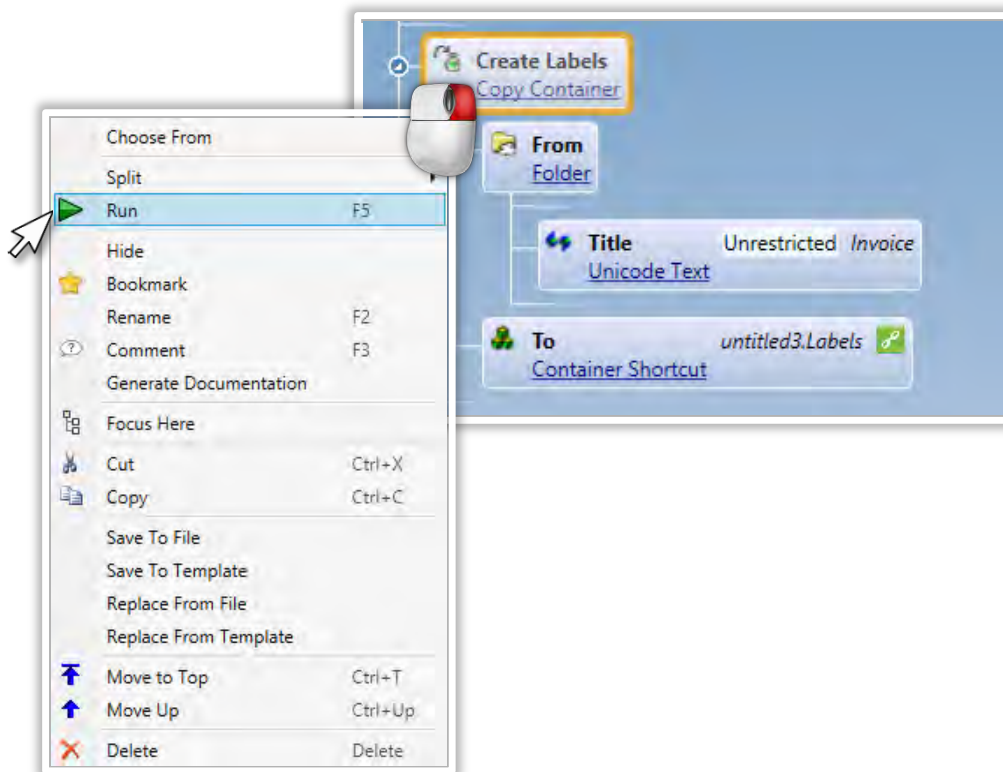
Click **OK** to close the dialog box.



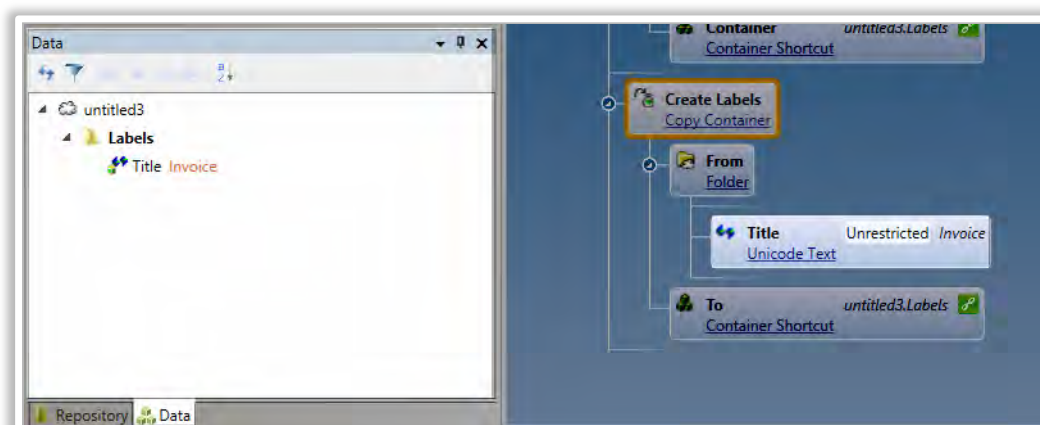
During the development stage it is important that you take time to check each section or function on the branch works as expected before you continue. To check to see if the new Title element writes into the container run the **Copy Container** object.

The Run option is available on the right-click menu when you right-click an **Action** object in the Branch view of a file, alternatively ensure the object is selected so it has an orange border and press F5.

The Events window shows the output when you run an action.



After you have Run the Copy Container object check the Data palette to see if the Container Labels has been updated with the new Title element, as illustrated on the below.

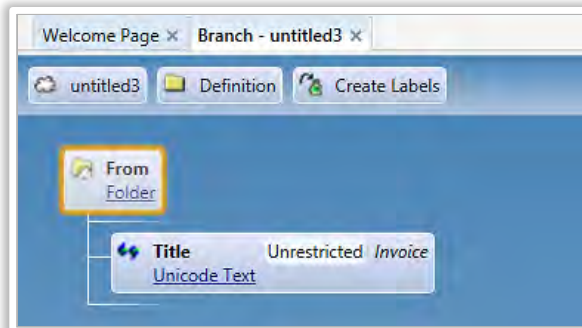



Continue to add the remaining label elements as required in the table on page 29.



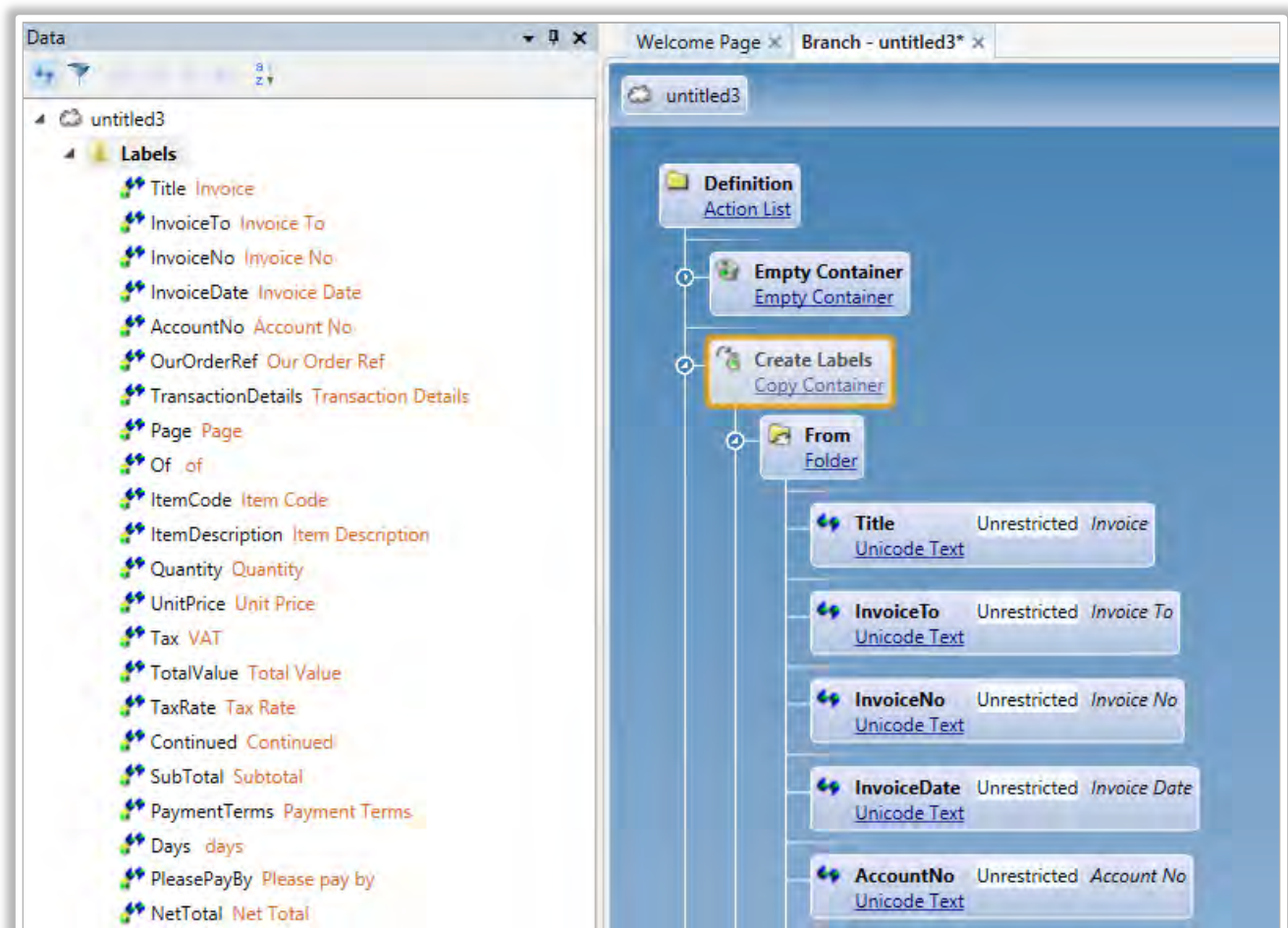
Tip

To clear the Branch editor view to display just the object and its child objects you are currently working on, right click the object you want to display and select set focus from the menu. The image below illustrates setting the focus on the From Folder object



To view a larger section of the branch again, click the Show Tree Root button  on the project toolbar or select a section of the branch from the list on the toolbar.

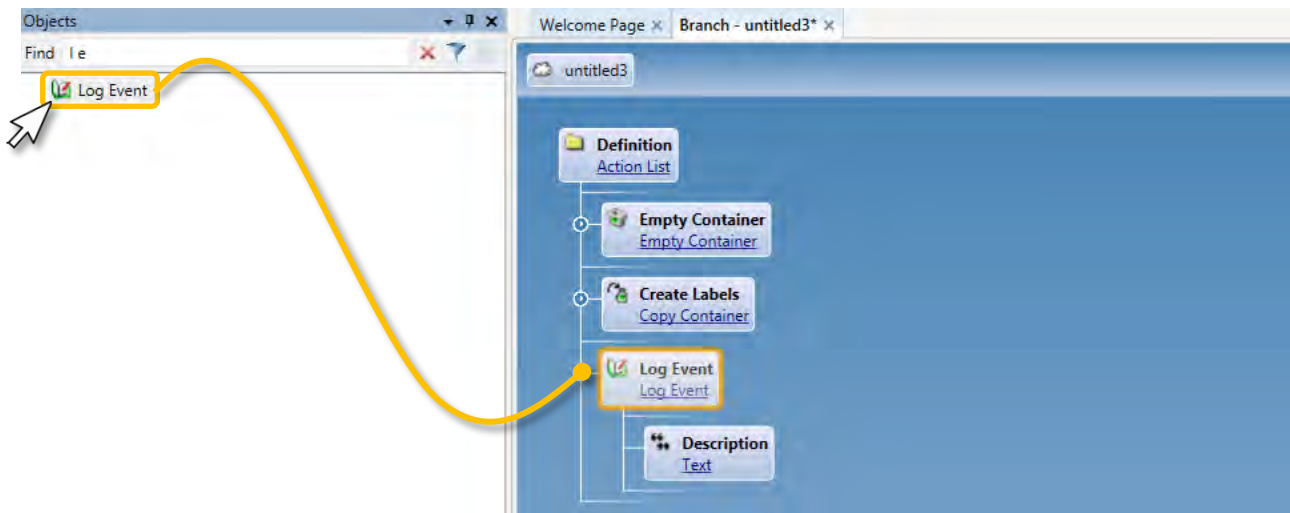
Once you have completed adding all the label elements run the Copy Container object to check they are written to the container as illustrated below.



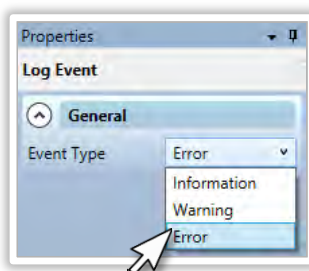
Step 14 – Add Log Event Object

To record the country code passed from the original source data file through the client branch to the language labels we need to create a Log Event. The Log Event object adds a message to the event log.

To add the Log Event object type **Log Event** in the Find field in the Objects Palette and drag the **Log Event** beneath the **Copy Container** object.



The Log Event has three Event Types available in the General section of the Properties window.



Select Error from the Event Type drop down list. As this branch is the default for the language labels, it will only be called if the language passed in the source data does not have a matching branch in the dynamic branch shortcut subfolder Language.fdb, so we would like to log this as an error.

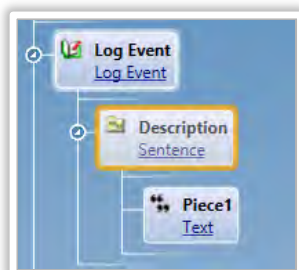
The following message is required to be written in the Log

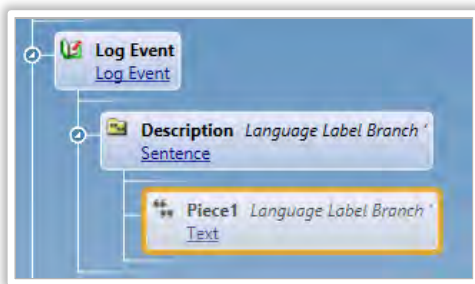
Language Label Branch '**XX.fsp**' not found

Where **XX** is a variable the language code passed in the source data.

To add this to the Log Event we need to create a sentence so we can build a string containing constants and variables.

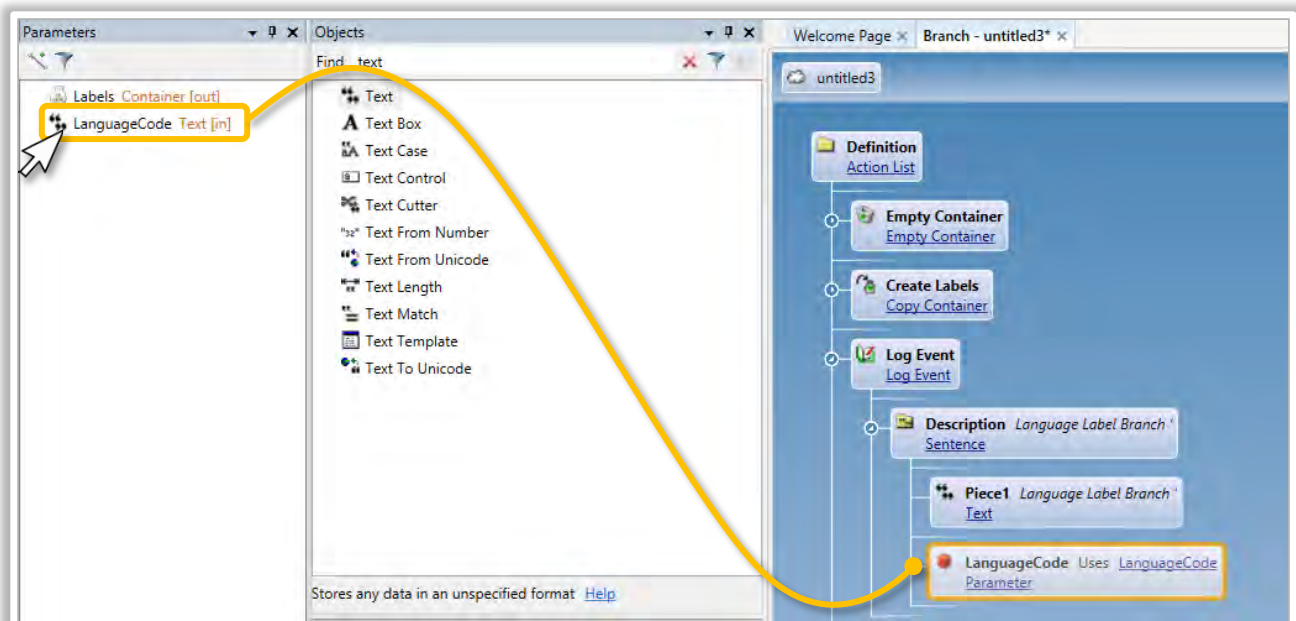
Select sentence from the objects palette, drag it over the Description **Text** object beneath the Log Event.



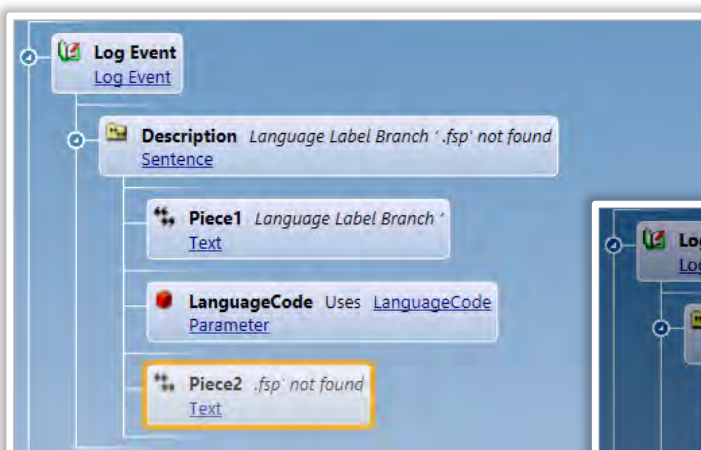


Click the Piece1 Text object and enter the following constant string
Language Label Branch

Drag the parameter **LanguageCode** from the Parameters window onto the branch beneath the Piece1 **Text** object.

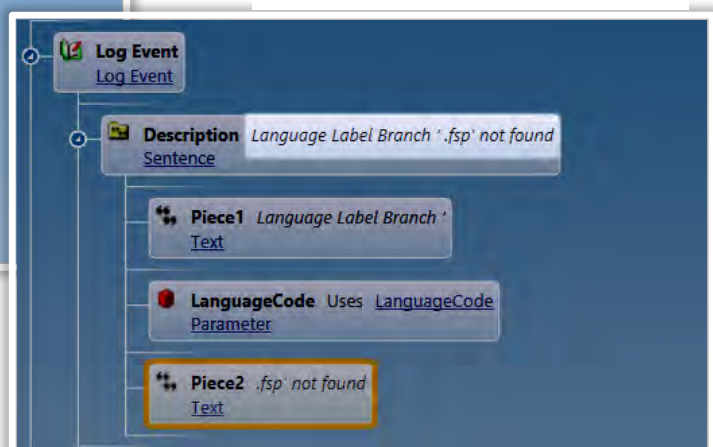



Add a Text box, beneath the Parameter. Click the icon and enter **.fsp' not found** to complete the sentence.



Tip

The sentence object displays the value of its child objects



To check the message created by the Log Event run the object and open the Events window. To open the Events window click  on the view toolbar, alternatively from the Menu bar select **View > Events**.

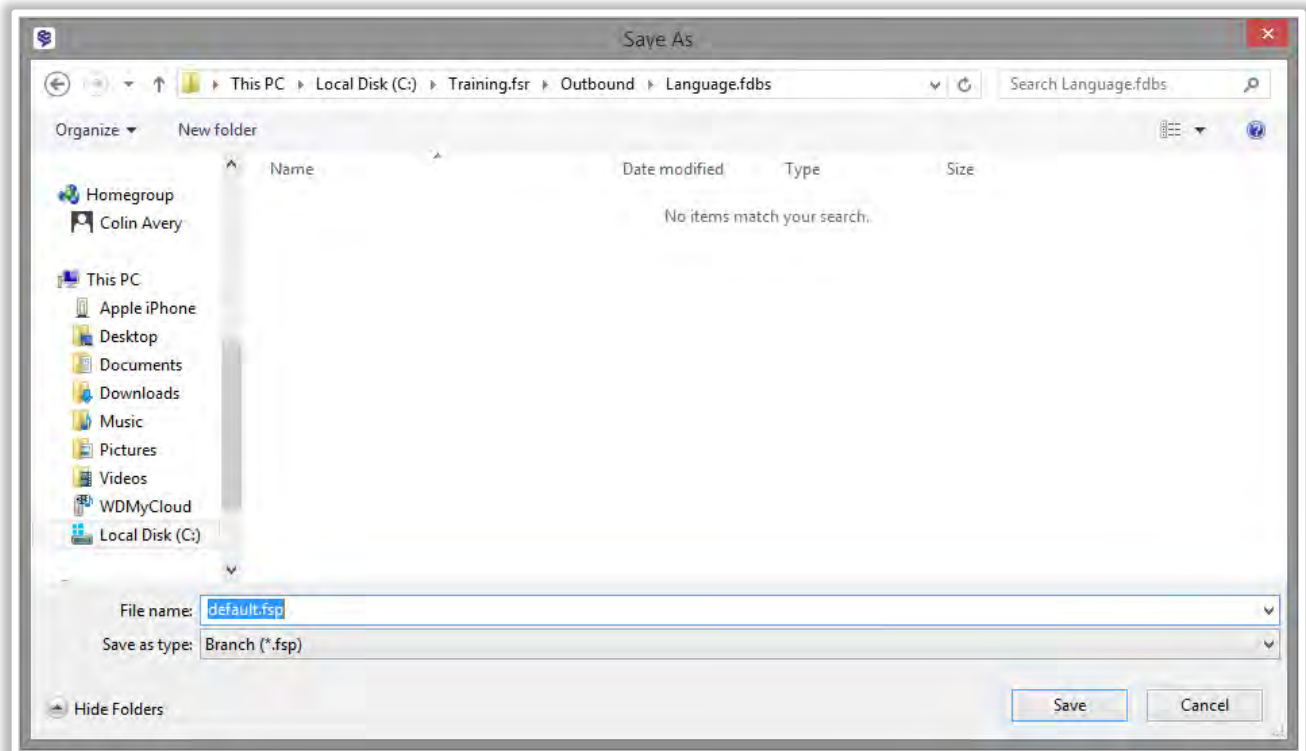


Errors are presented in Red within the Events window.

Step 15 – Save Branch

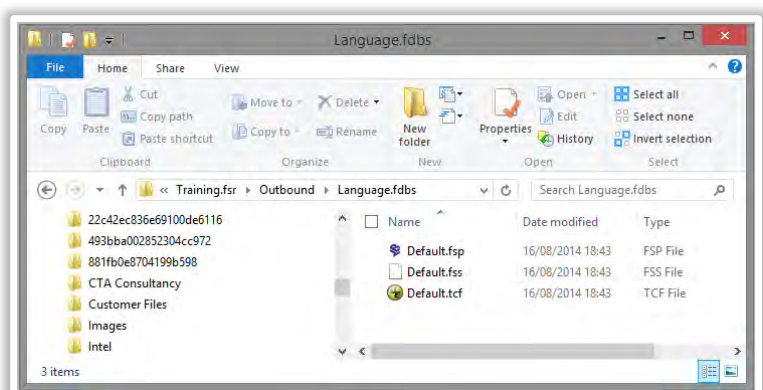
Save the branch as Default to the following path.

Training.fsr > Outbound > Languages.fdb



When a branch is saved it creates or overwrites three files

- **fsp** (Foundation Server Project)
- **fss** (Foundation Server State)
- **tcf** (Transform Configuration file)



Project Checklist

1. ~~Repository Design~~

2. ~~Create Local Repository~~

3. ~~Language Labels~~

4. **Global Resources**

Global Constants

Open Simple Action Template

Replace Root object with Items Folder

Create Input path

Create Output Path

Create Image Paths

Scheduled Process Branch

Open Simple Action Template

Create Array in Container

Shell Object to create Directory Structure

Review Alternate Technique

5. Plain Text Template Wizard

6. Plain Text to Container

7. Modify Branch Structure

8. Dynamic Branch Shortcut

9. Document Organizer

10. Synchronize and Deploy



Global Resources

Variables can be declared on a global basis using techniques similar to the declaration of the Language Labels. Use globally available objects to share exactly the same data amongst all branch files across the Repository.

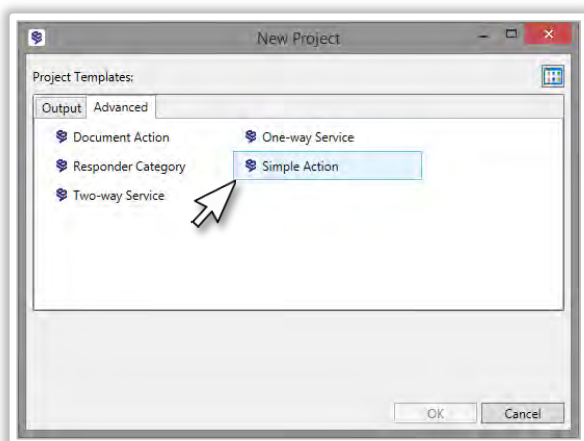
Global definitions reduce development time because many of the needed data definitions are simply imported rather than recreated.

Global Constants

Create a Global Constants branch to define the input and output directory paths used throughout the project.

Step 1 – Select Template

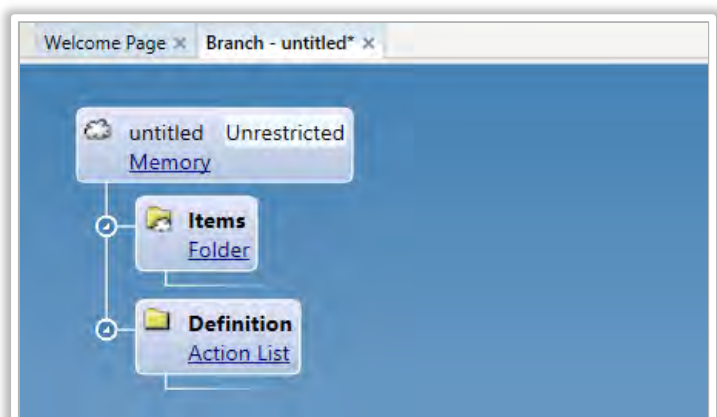
Open the Transform Designer and select **File** > **New** from the menu bar. Alternatively click **Select a template** from the Welcome page

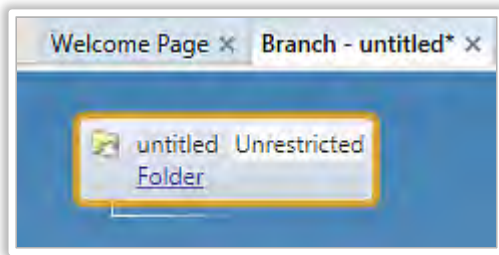


Step 2 – Simple Action

From the Advanced Tab select Simple Action and click **OK**.

This opens the branch editor view





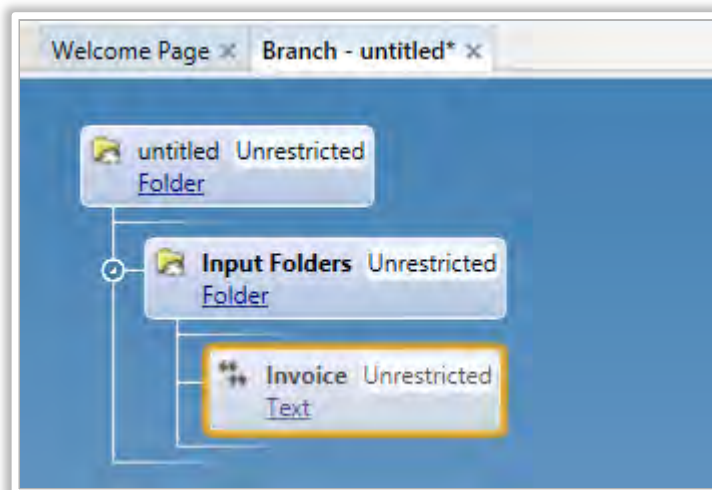
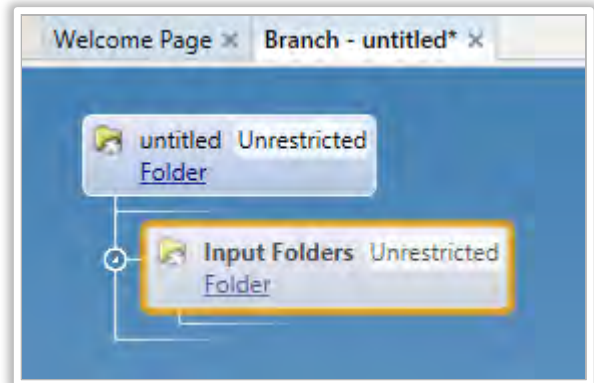
Step 3 – Replace Root object

Right click and hold the **Folder** object and drag it over the **Memory** object so that it becomes the root.

Step 4 – Add Folder

In the Objects palette Find field type **fol**, select **Folder** from the list and drag it onto the branch under the root **Folder** object.

Rename the object to **Input Folders**.



Step 5 – Add Input Path

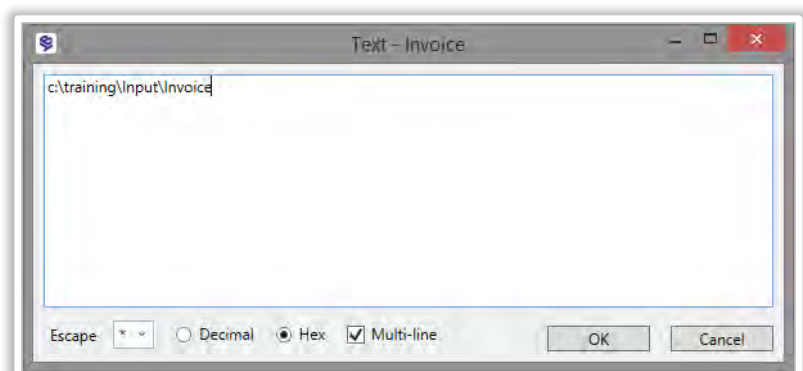
In the Objects palette Find field type **text**, select **Text** from the list and drag it onto the branch under the Input Folders **Folder** object.

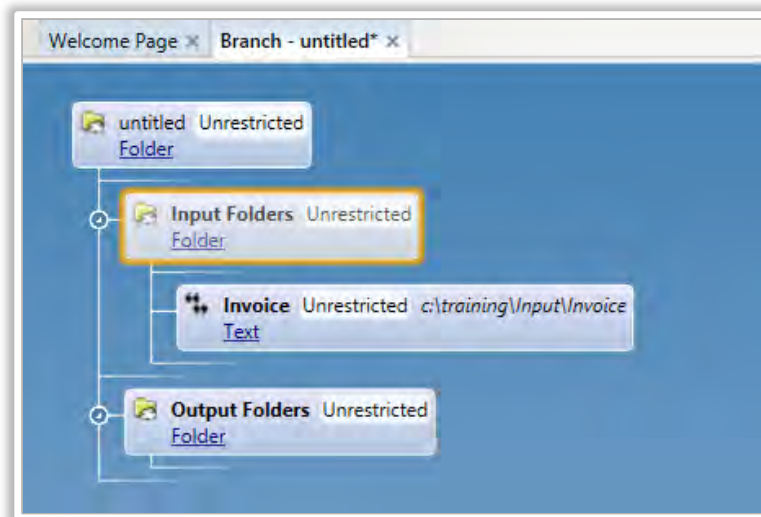
Rename the object to **Invoice**.

Step 6 – Type Path

Click **🔗** to open the Text – Invoice dialog box and enter the watch folder to be used to ingest the input plain text positional source data file.

This path will be used in the main queue process Invoice branch file.





Step 7 – Add Folder

Add another **Folder** object to the branch and rename it **Output Folders**.

Step 8 – Create Path

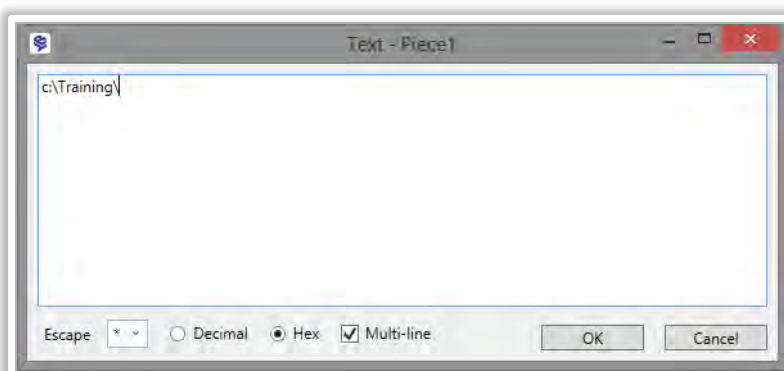
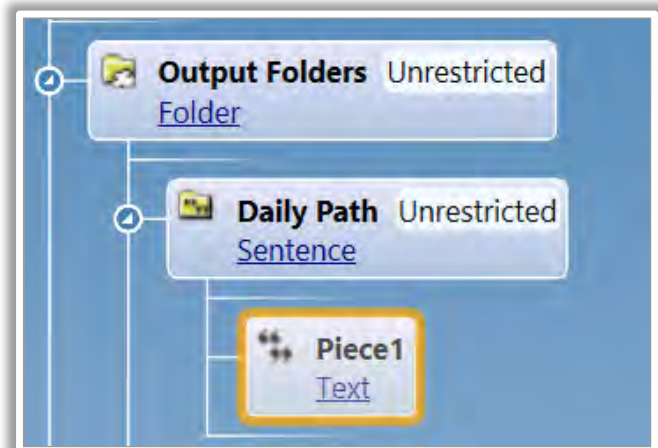
Use the **Sentence** object to construct the required output path.

C:\yyyy\MMMM\dd

Example c:\2014\August\27

From the Objects palette Find field type **sent** select the **Sentence** object and drag it onto the branch beneath the **Folder** object labelled **Output Folders**. Rename to **Daily Path**.

Expand the **Sentence** object to reveal its child **Text** object.



Step 7 – Root path

Click “” to open the Text – Piece1 dialog box and enter the root path to write the output files.

This path will be used in the main queue process Invoice branch file.



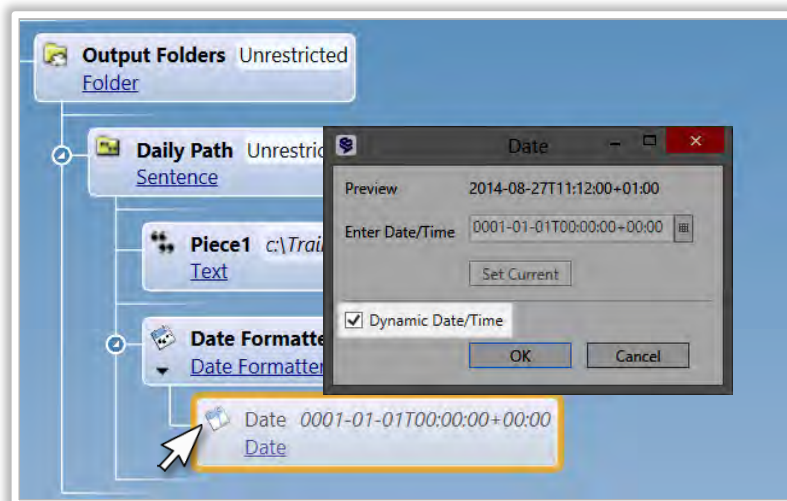
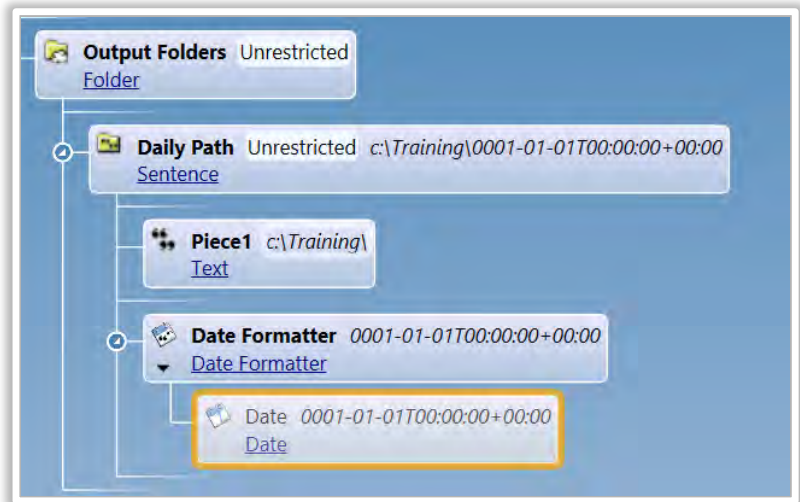
Note

When using a sentence to build a path remember to add the ‘\’ backslash path separator where required. In the example illustrated C:\Training\ is used as the next element of the path will be passed by the second object in the sentence.

Step 8 – Date Formatter

In the Objects palette Find field type **d f**, select **Date Formatter** and drag it onto the branch so it becomes the second object beneath the **Sentence** object.

Expand the **Date Formatter** object to reveal the **Date** object.



Step 9 – Date

Click **Date** object icon to open the Date dialog box.

Once open check the Dynamic Date/Time option.

This returns the current system date and time.

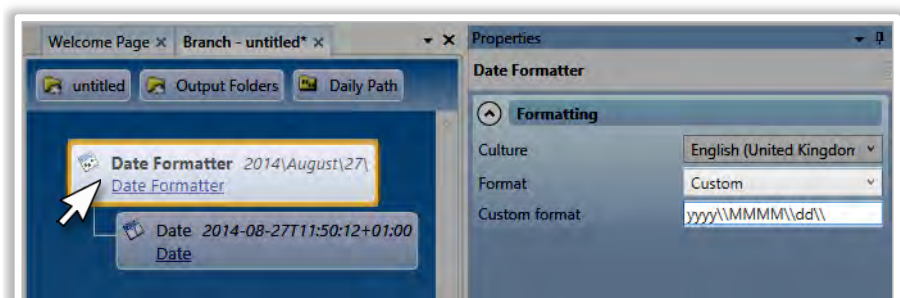
Click **OK** to close

Step 9 – Date Formatting

Select the Date Formatter object and then change the Formatting properties as follows:

Format **Custom**

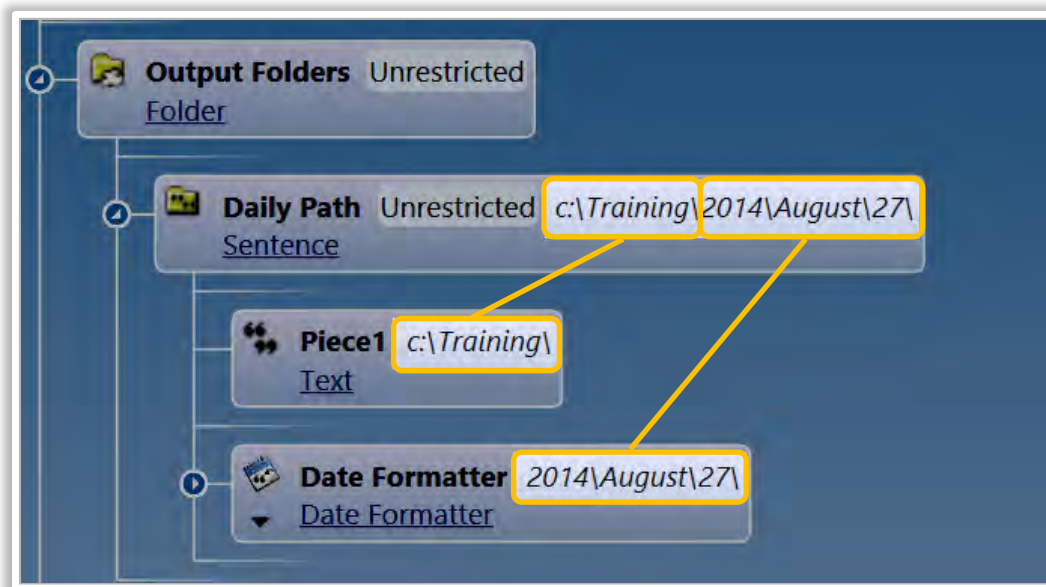
Custom format **yyyy\\MMMM\\dd**



Note

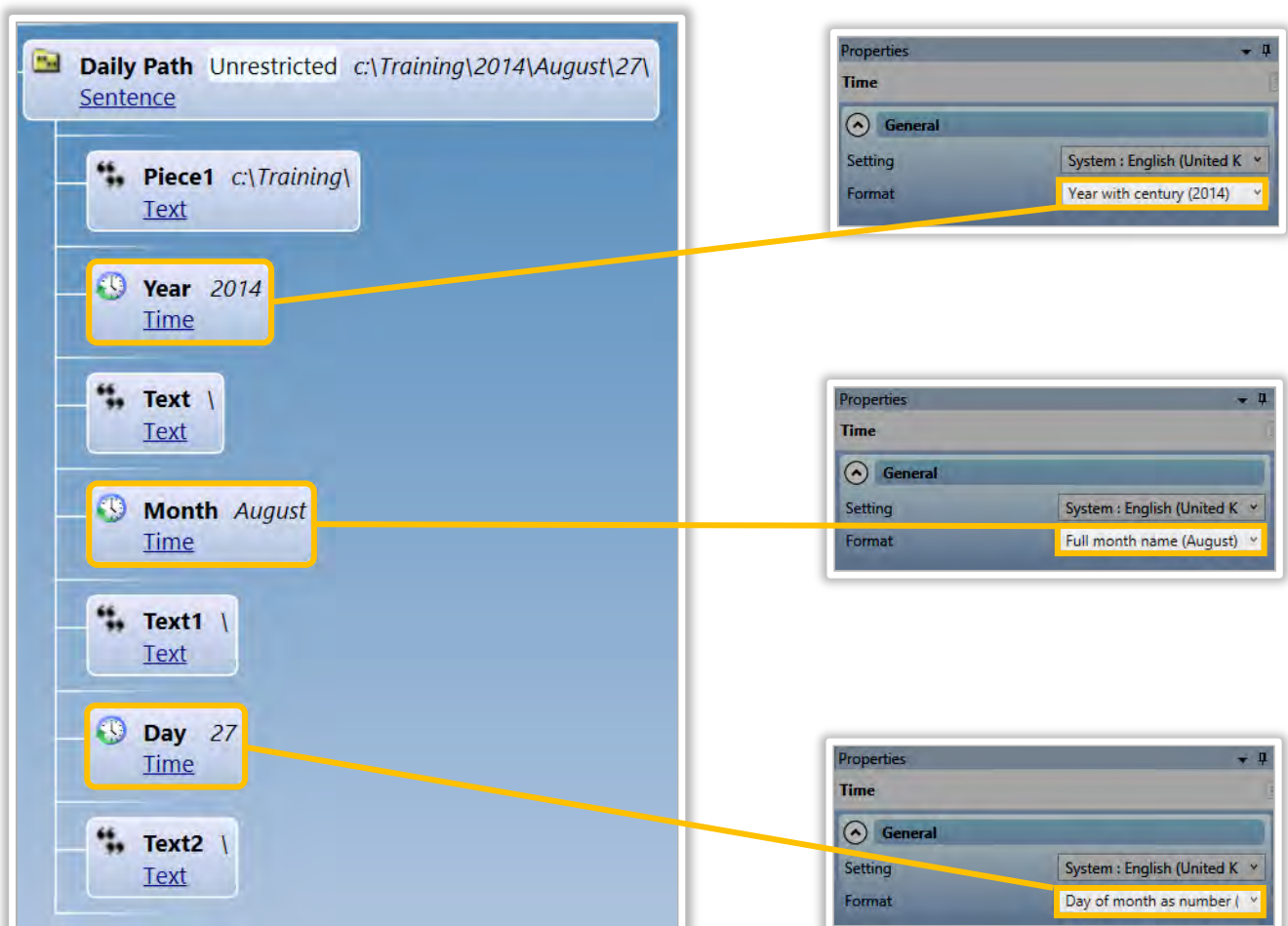
The double backslash \\ shown is not a typo as the date formatter uses the \ as an escape character. So if you want to return a backslash to the caller Date Formatter then you need to use \\ in the custom format.

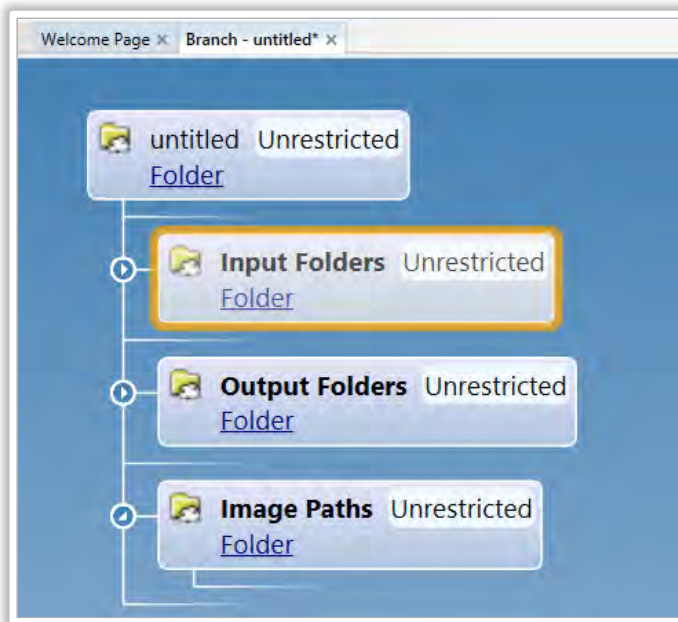
The image below illustrates how using a sentence objects builds the string required.



Alternative Technique

This method demonstrates an alternative method to achieve the same result. Rather than using a **Date Formatter** object within the **Sentence** object to build the path as previously described, use a number of **Time** and **Text** objects to construct the sentence. When using the **Time** object you can set the output Format required from the Time Properties window.





Step 10 – Create Image Paths

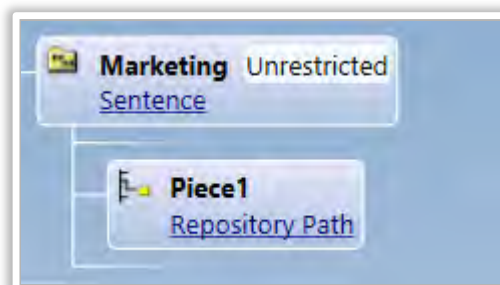
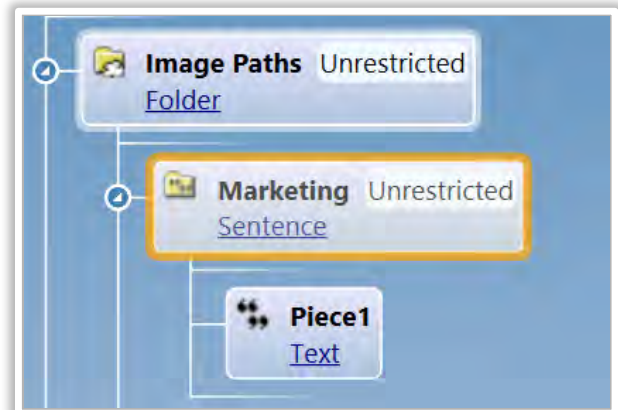
Add another Folder object and rename to Image Paths.

Expand the object to reveal the add point.

Step 11 – Add Sentence

Add a **Sentence** object to build the path to the Marketing image files, which will be used on each page of the invoices.

Rename to **Marketing**



Step 12 – Repository Path

In the Objects palette Find window type **r p**, select **Repository Path** and drag it onto the branch beneath the **Sentence** object.

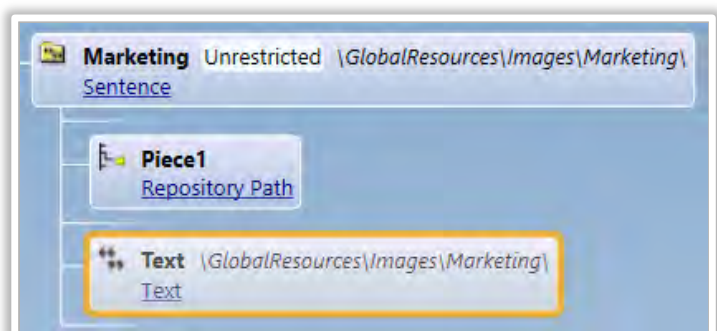
The **Repository Path** object identifies the path to the repository containing the current branch.

Step 13 – Subfolder Path

Add a **Text** object to build the path to the Marketing image files, which will be used on each page of the invoices.

Open the Text dialog box and add the following path:

\GlobalResources\Images\Marketing\





Step 14 – Create Flag path

Repeat Step 11 thru 13 to create the path to the Flag images.

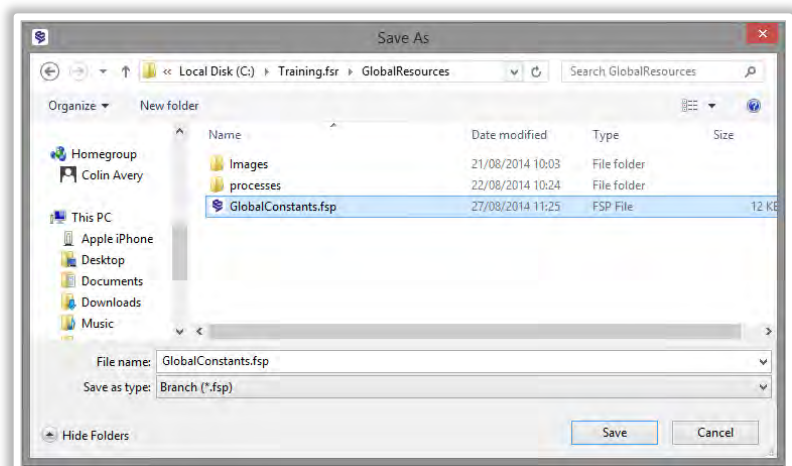
Open the Text dialog box and add the following path:

\\GlobalResources\\Images\\Flags\\

Step 15 – Save Branch

Save the branch as GlobalConstants in the following package.

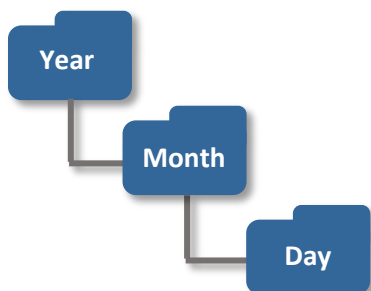
C:\\Training.fsr\\GlobalResources



Scheduled Process Branch

The project requires the output to be written to a specific windows folder structure which needs to be created on a daily basis. It is proposed that a Transform process will be used to fulfil the function.

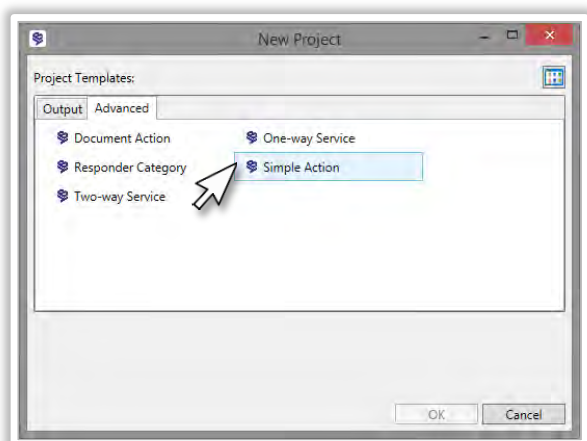
The required structure is shown below.



The following describe the steps required to build the Scheduled process branch.

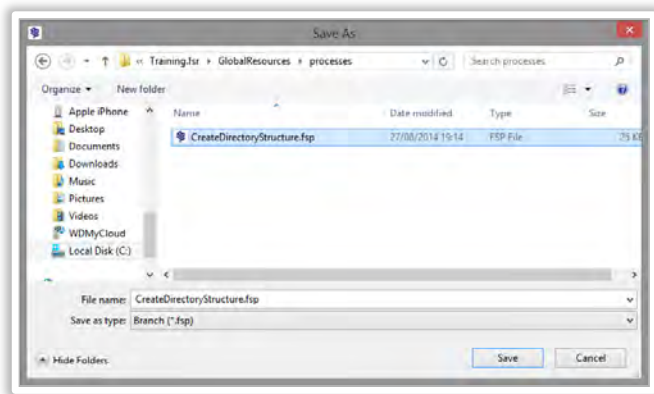
Step 1 – Select Template

Open the Transform Designer and select **File** > **New** from the menu bar. Alternatively click **Select a template** from the Welcome page



Step 2 – Simple Action

From the Advanced Tab select Simple Action and click **OK**.



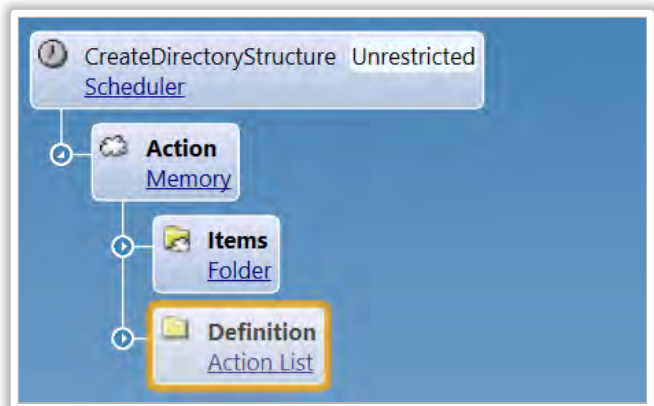
Step 3 – Save Branch

As this branch is a process branch you must save it in the processes folder within the package. Save the branch as DirectoryCreator within the following package.

C:\Training.fsr\GlobalResources\processes

Step 4 – Add Scheduler

In the Objects palette Find field type **sch** and select **Scheduler** and drag onto the branch so that it becomes the root object. This will adopt all the previous objects.

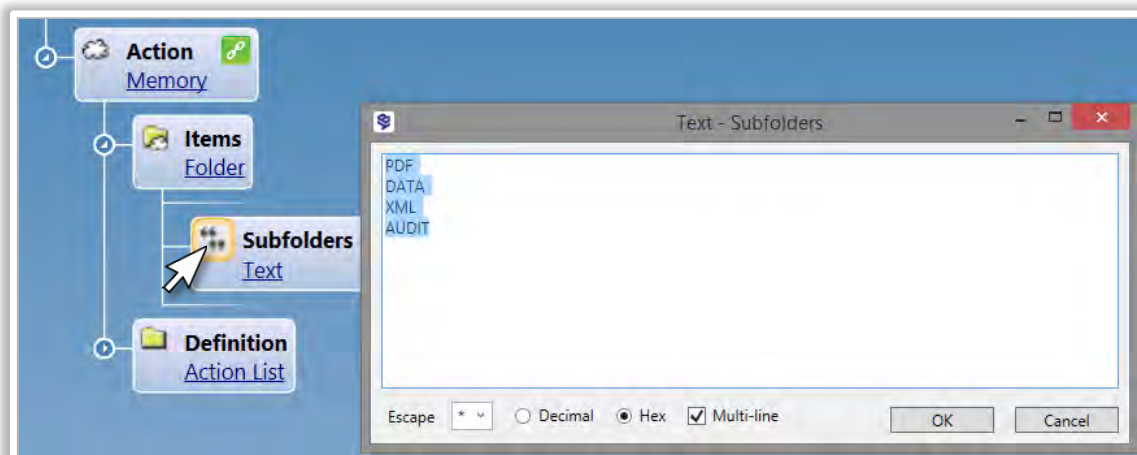


Step 5 – Subfolder Array

The requirement is to create the folder structure with Year\Month\Day containing four sub folders.

- Data
- Audit
- PDF
- XML

Create an array of the four subfolders by adding a **Text** object beneath the **Folder** object, click “to open the Text dialog box and create a list of the four subfolders required to build the array.

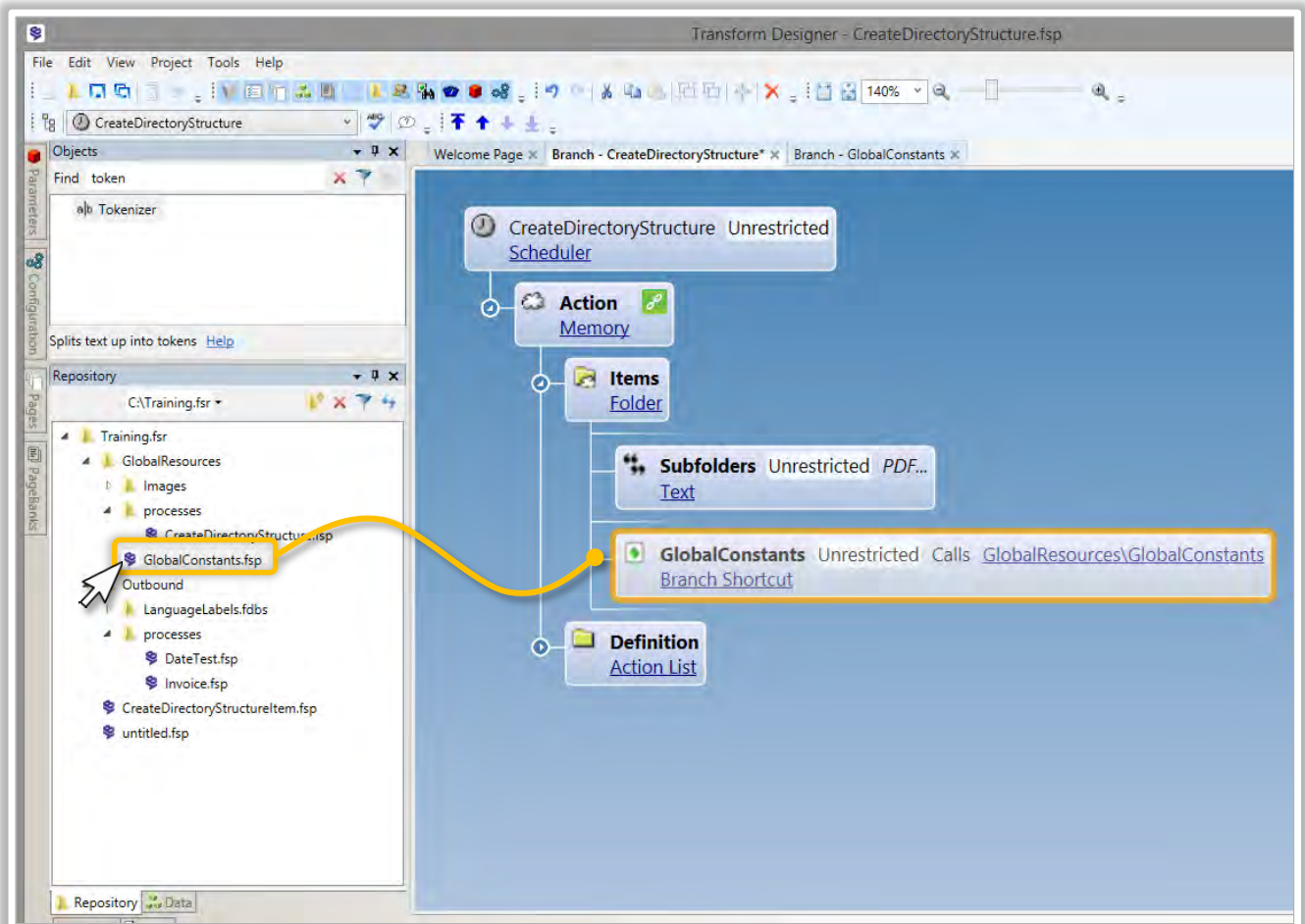


By creating an array we can loop through the values with a single process reducing the effort required to develop the branch.

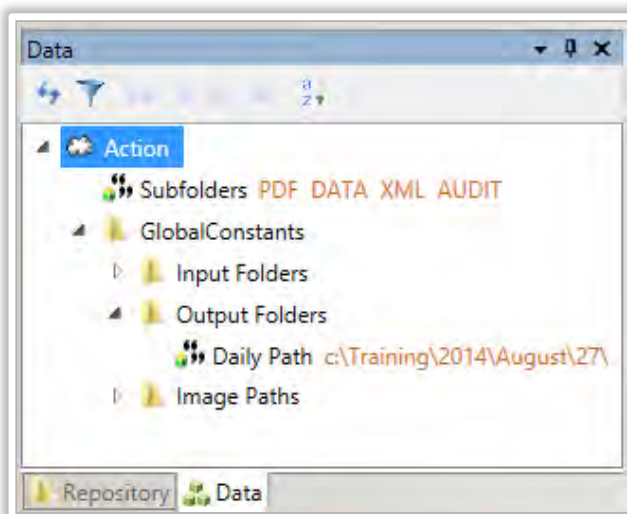
Step 6 - Branch Shortcut

As we have previously created and saved the **GlobalConstants** branch which contains the Output directory path we can use the variables declared within this branch by means of a Branch Shortcut.

From the Repository Window select the **GlobalConstants.fsp** and drag it onto the Branch beneath the **Folder** object as illustrated below.



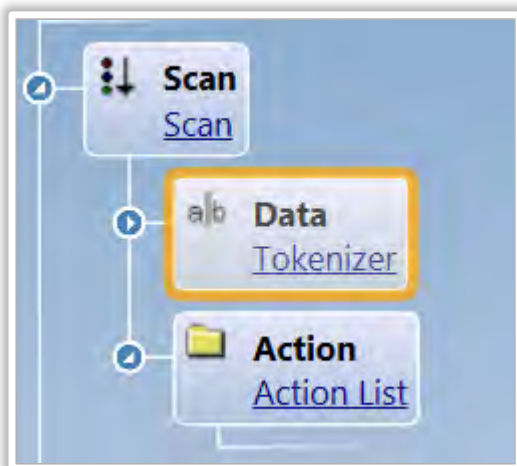
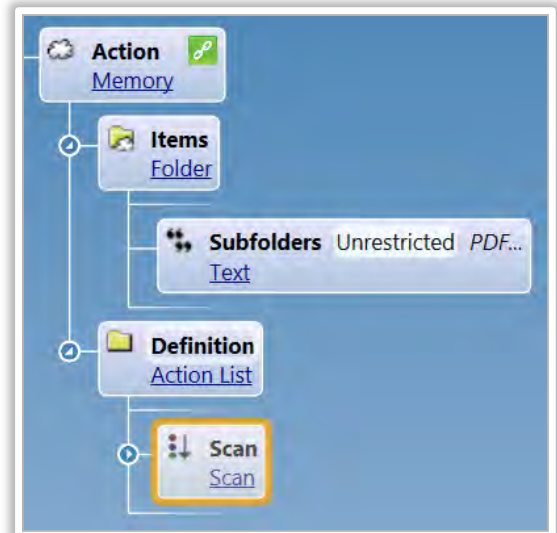
Branch Shortcut items become available as container objects visible in the Data palette, which can be used as container shortcuts throughout the branch.



Step 7 – Scan object

From the Objects palette select **Scan** object and drag onto the branch beneath the **Action List** labelled Definition.

A Scan object performs an action for each item in a data set.



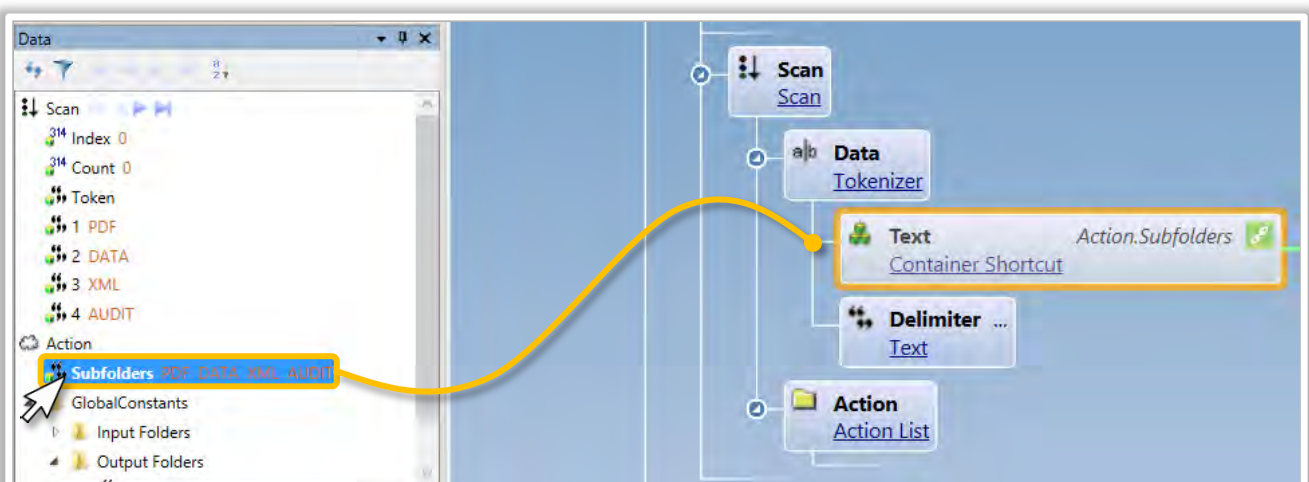
Step 8 – Tokenizer

Replace the current Walker object with a Tokenizer object from the Objects palette.

Using a Tokenizer under a Scan object breaks the subfolder list into pieces called tokens. As the Scan steps through the tokens it performs an action on each value as defined in the Action List.

Step 9 – Create Container Shortcut

Expand the **Tokenizer** object to reveal the child objects then from the Data Palette select Subfolders under the Action memory, drag and drop it to the **Text** object under the **Tokenizer**, so that it becomes the list to step through.

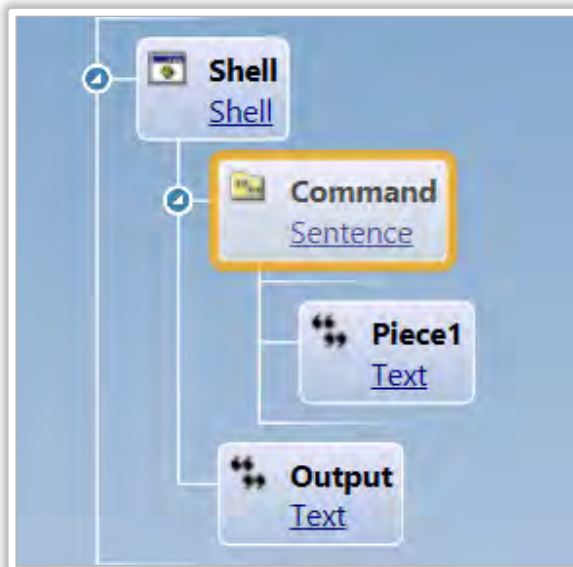
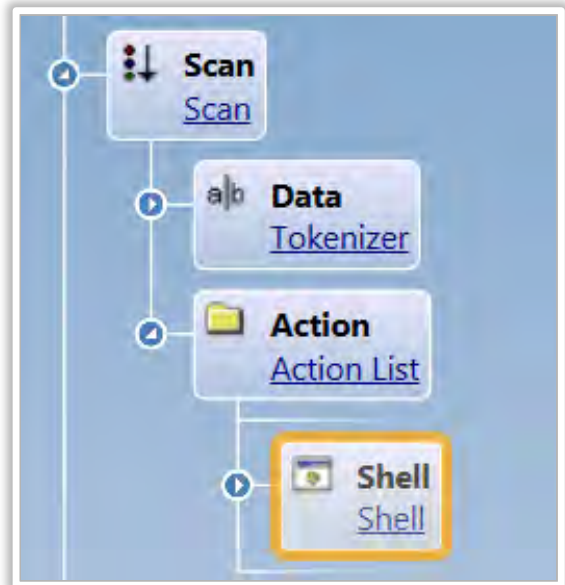


Step 10 – Shell Command

From the Objects palette select the **Shell** object and drag onto the branch beneath the **Action List**.

The Shell object runs the command specified in its Command child object.

We are going to use the **Shell** command to create the directory structure using the MSDOS md (make directory command).



Step 11 – Build Command

Expand the Shell object then from the Objects palette select the **Sentence** object and drag onto the branch beneath the Shell object.

Expand the Sentence object.

Step 12 – Make Directory Command

Click “” Piece1 **Text** object and enter md<space>

This will execute the MSDOS command md (make directory)



Step 13 – Global Resource Output Path

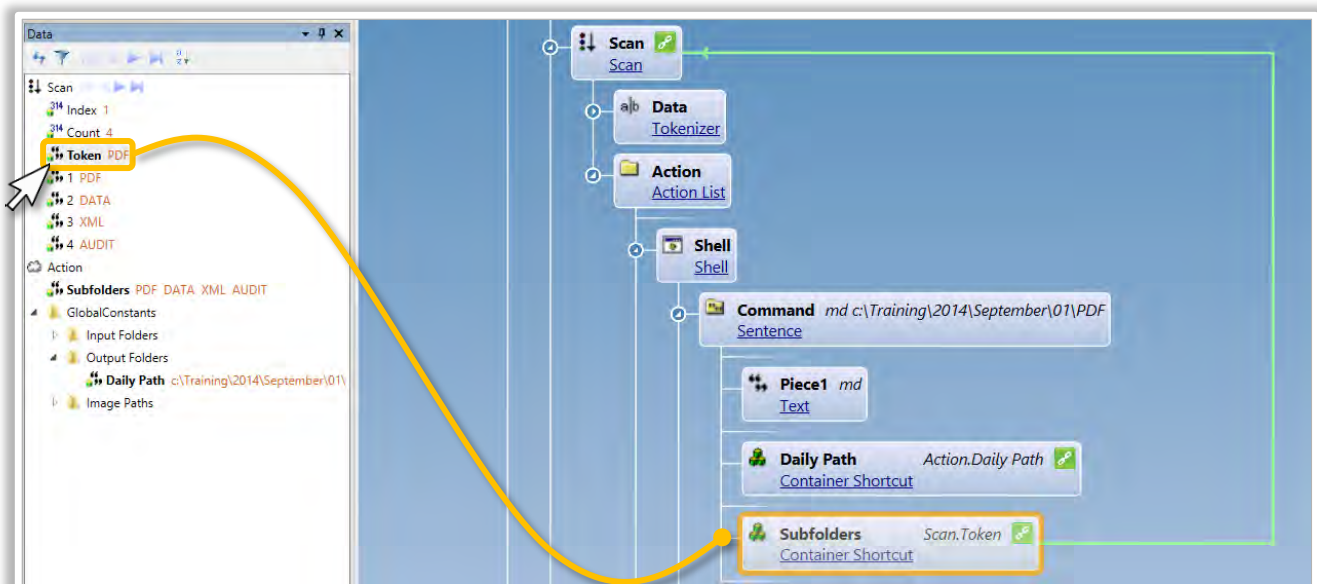
We previously created a GlobalConstants branch, which contains Output Folders Daily Path. Locate the GlobalConstants container within the Data palette and click the expand control to expose the Output Folders then click the expand control to expose the Daily Path text.

Drag and drop the Daily Path text from the Data palette to the branch beneath the Piece1 **Text** object so that it becomes the second piece of the sentence. You can see the format of the sentence by looking at the **Sentence** object value *md c:\Training\2014\September\01*



Step 14 – Subfolders

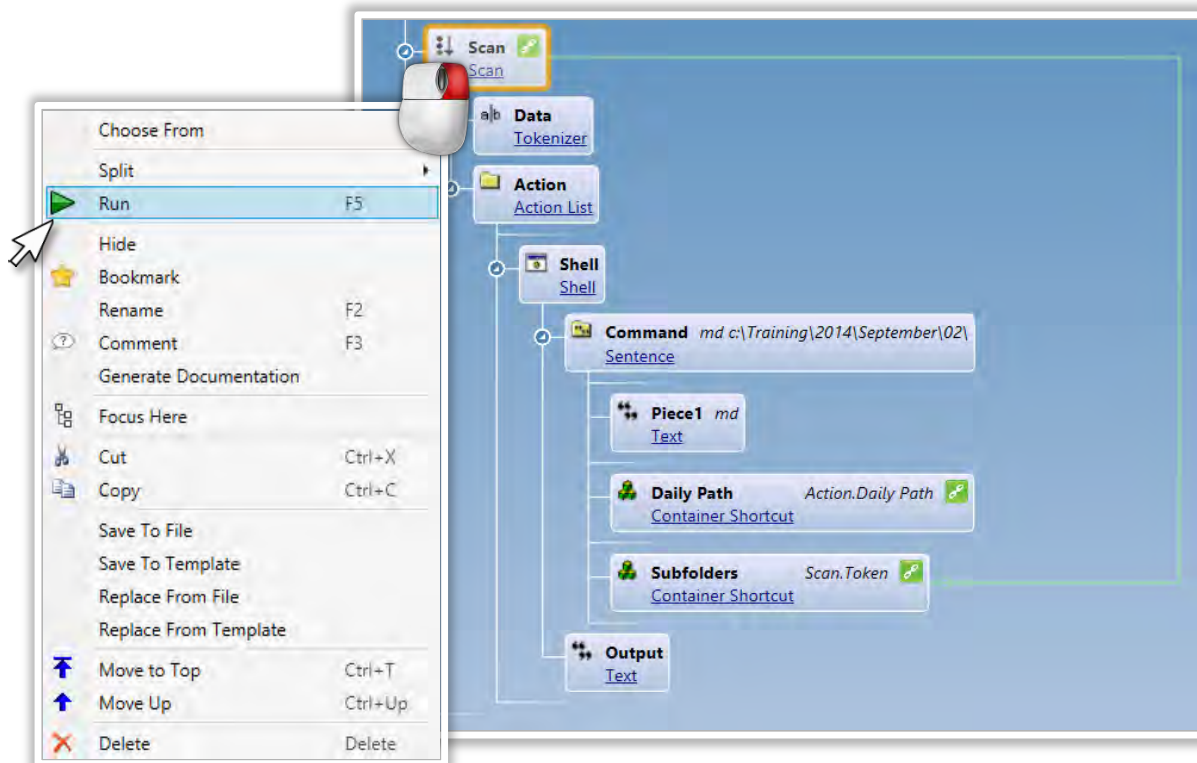
To use the array expand the Scan within the Data palette and drag the Token onto branch beneath the Daily Path Container Shortcut, this completes the sentence and the Shell command.



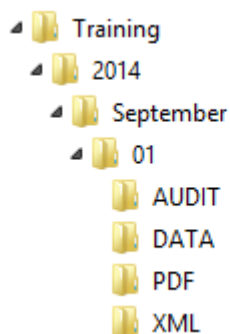
Step 15 - Test your Code

IT is always advisable to test your code as you progress through each step of your development. Transform Designer allows the developer to Run any Action object on the branch independently.

Right-click the **Scan** object and select Run ► from the right-click menu, alternatively if the Scan object is the selected press F5.



Open Windows file explorer and check to see if the directory structure has been created.



Step 16 - Save the Branch

Once you have checked the process works and it creates the directory structure as required then save the project. Make sure that the project is saved in the processes folder within the GlobalResources package as described in **Step 3**.

Project Checklist

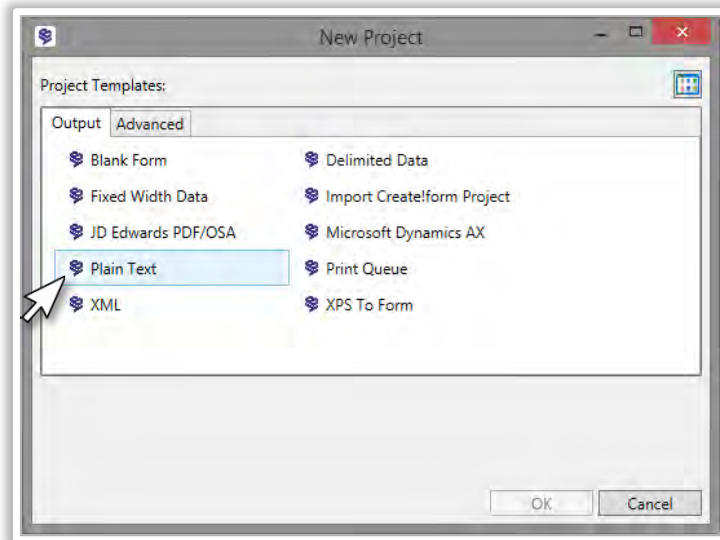
1. ~~Repository Design~~
2. ~~Create Local Repository~~
3. ~~Language Labels~~
4. ~~Global Resources~~
5. **Plain Text Template Wizard**
 - Open Plain Text Template**
 - Choose Project type**
 - Browse for Sample Data**
 - Select Output Type**
6. Plain Text to Container
7. Modify Branch Structure
8. Dynamic Branch Shortcut
9. Document Organizer
10. Synchronize and Deploy



Plain Text template

In the New Project window, select a template to use as a starting point for your project.

The following will walk you through the template wizard process.



Step 1 – Plain Text Template

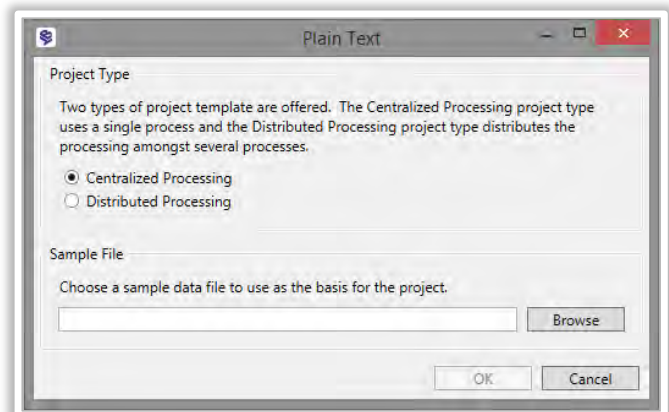
The input source data type is positional. From the New Project dialog box select Plain Text, click **OK** to launch the wizard.

Step 2 – Project Type

The Plain Text dialog box has two sections.

Project Type

- **Centralized Processing** Creates a Process module branch.
- **Distributed Processing** Creates an Action One way branch. Use when you want to call the branch using a branch shortcut from a client branch.



Select Centralized Processing.

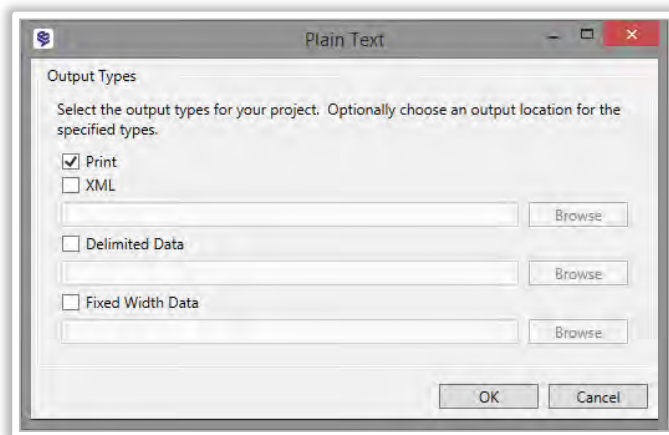
Sample File

Browse for a sample data file to use during the development stage. Click **OK**



Important

It is imperative to have a comprehensive input data sample during development, which contains multiple records and multiple pages containing all possible sections and fields.

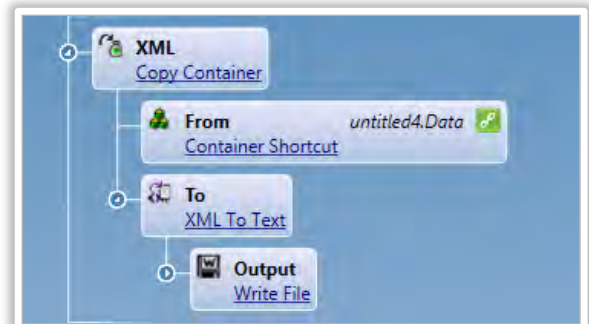


Step 3 – Select Output Types

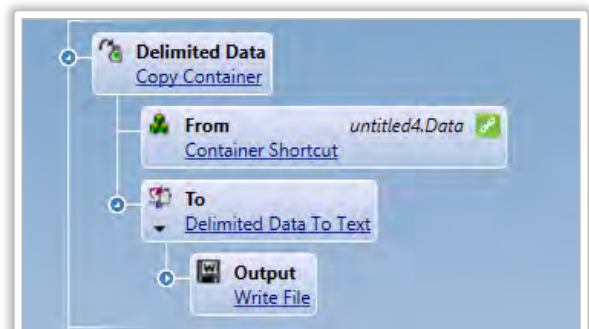
Select the output types for the project

- **Print Default uses Advanced printer** object to create a windows printer spool file.

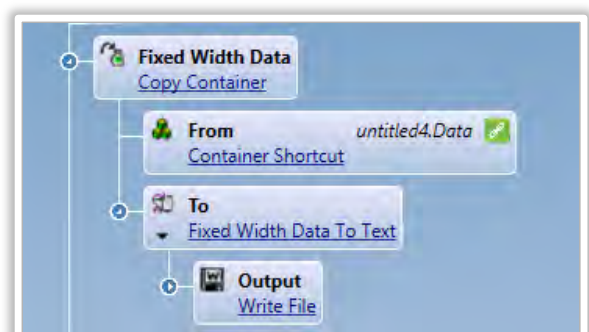
- **XML** Browse for output location. Creates **Copy Container** and **XML to Text** objects to write container to file.



- **Delimited Data** Browse for output location. Creates **Copy Container** and **Delimited Data to Text** objects to write the container to file.



- **Fixed Width Data** Browse for output location. Creates Copy Container and Fixed Width Data to Text objects



Select **Print** and **XML** for this exercise. Click **OK**.

Project Checklist

1. ~~Repository Design~~
2. ~~Create Local Repository~~
3. ~~Language Labels~~
4. ~~Global Resources~~
5. ~~Plain Text Template Wizard~~

6. **Plain Text to Container**

Define Plain Text to Container Properties

Document Grouping

Edit Standard Sections

Define Header Section

Create Header Blocks

Define Custom Sections

Footer Section

Build Test Expression

Change Section Type

Item Lines Section

Description Lines Section

Update Container

Save Branch

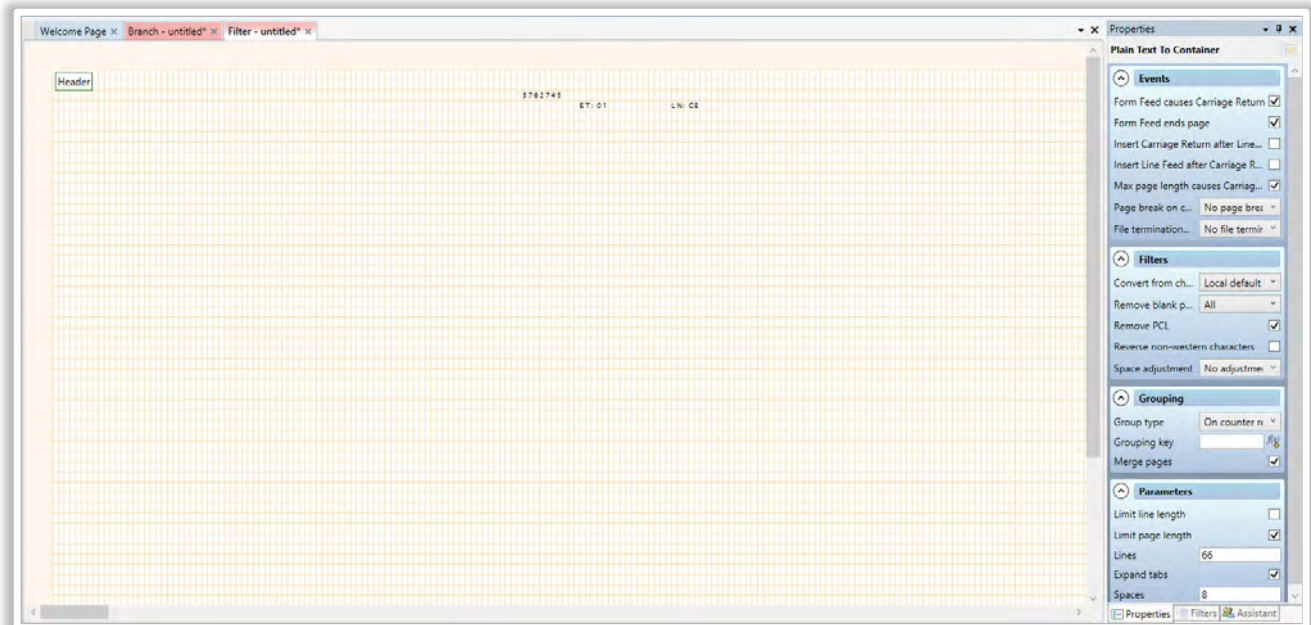
7. ~~Modify Branch Structure~~
8. ~~Dynamic Branch Shortcut~~
9. ~~Document Organizer~~
10. ~~Synchronize and Deploy~~



Plain Text To Container

Use The Section Filter editor to.

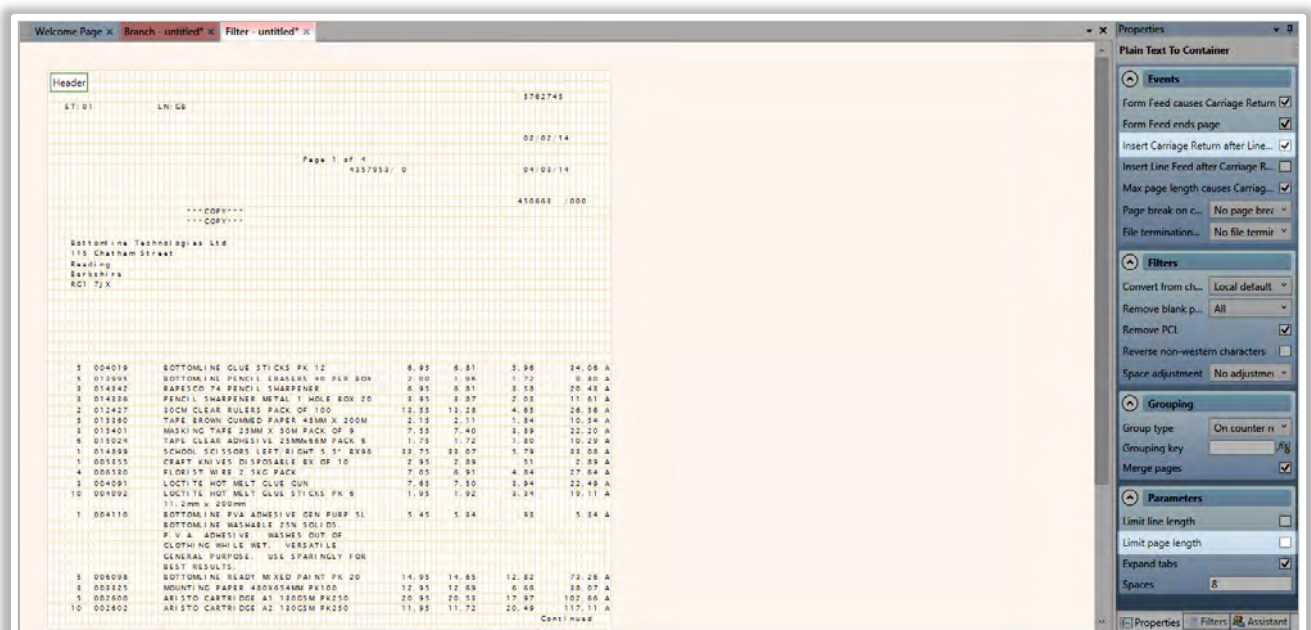
- Define sections of the page to identify the location of different types of data.
- Identify blocks of data within each section to create fields.
- Define a condition for each custom section you create, to specify how to identify that section.




Step 1 – Define Plain Text to Container Properties

The source data does not contain carriage returns, check **Insert Carriage Return after Line Feed** under Events in the Plain Text to Container Properties windows.

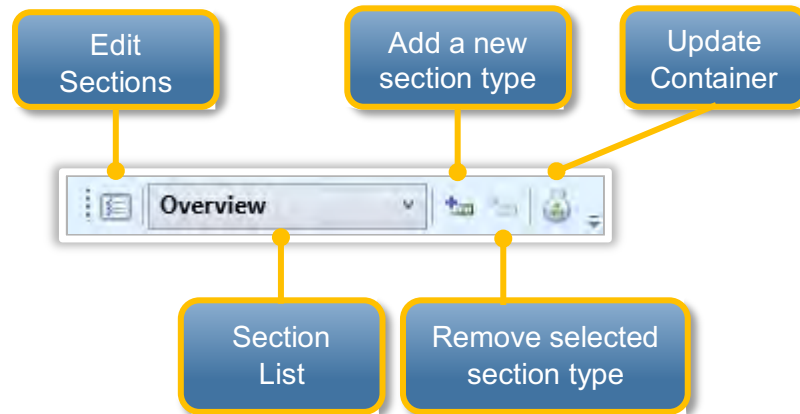
Also the uncheck **Limit Page length** under Parameters in the Plain Text to Container Properties window. The default setting is 66 lines.



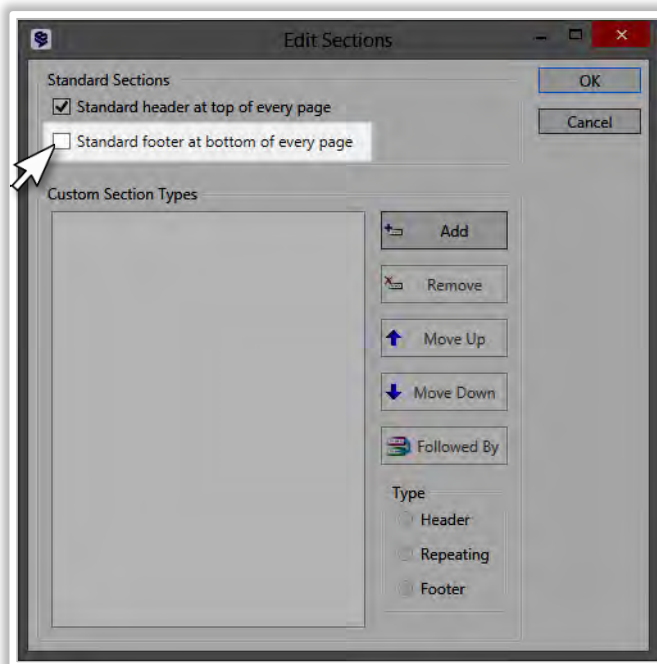
Step 2 – Edit Standard Sections

We have established in the project scope data definition that there is a fixed header on every page but all other sections are conditional. Click Edit Sections  on the section filter toolbar.

Section Filter toolbar



From Edit Sections dialog box uncheck **Standard footer at bottom of every page**.



Click **OK** to close the Edit sections.

Step 3 – Define Header Section

Define the size of the header section. Double-click in the header area in the filter view or select the Header section in the Section list. This will open the section bounding box presented as a silver box.

The screenshot shows a document header section enclosed in a silver bounding box. The header contains the following text:

ET: 01 LN: G8 5762745

02/02/14

Page 1 of 4 4357953/ 0 04/03/14

450663 /000

COPY

COPY

Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX

5	004019	BOTTOMLINE GLUE STICKS PK 12	6.95	6.81	5.96	34.06 A
5	012995	BOTTOMLINE PENCIL ERASERS 40 PER BOX	2.00	1.96	1.72	9.80 A
3	014342	RAPESCO 74 PENCIL SHARPENER	6.95	6.81	3.58	20.43 A
3	014336	PENCIL SHARPENER METAL 1 HOLE BOX 20	3.95	3.87	2.03	11.61 A

Resize the Header section by dragging the bottom of section area to extend the header to 28 lines.

The screenshot shows the same document header section as before, but with a silver bounding box that has been extended to 28 lines. A double-headed arrow at the bottom of the bounding box indicates the new height, labeled "28 lines". The text content remains the same as in the previous screenshot.

Step 4 – Create Header Blocks

Add the blocks within the header to define the header elements.

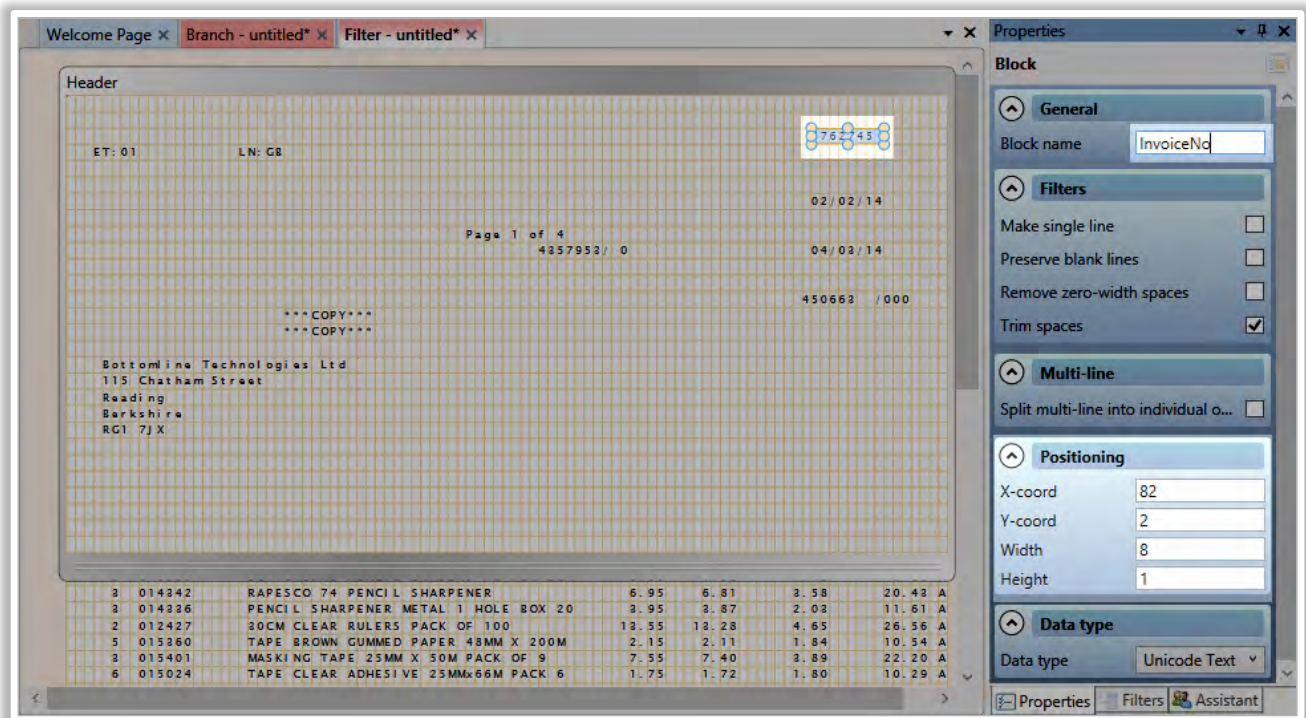
Drag a rectangle round the data you want to include.

Enter a name for the block in the label box that appears at the bottom left of the block or on the General section in the Properties window. This is the name of the element within the container.

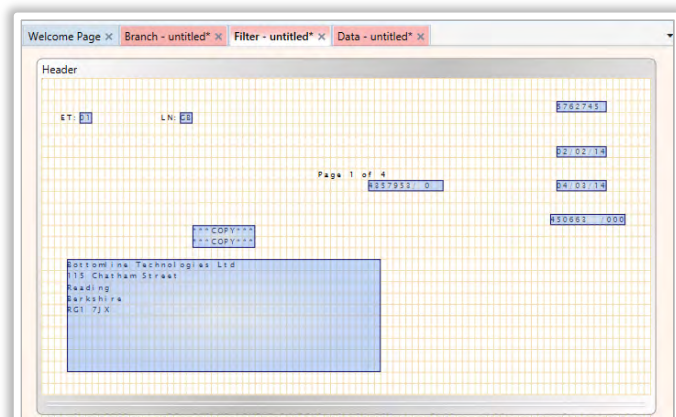
As we have selected XML output, elements must follow these naming rules.

- Names can contain letters, numbers, and other characters.
- Names cannot start with a number or punctuation character.
- Names cannot start with the letters xml (or XML, or Xml, etc).
- Names cannot contain spaces.

Set the position and size of the block. Either use the resize handles on the bounding box or the Positioning section in the Properties window.



Use the Header data definition on Page 53 to map the remaining elements.



Header Section Definition

Fixed on every page of the positional data for the first 28 lines.

The diagram illustrates a plain text header section with the following fields and their values:

- A**: 5762751
- B**: ET:01
- C**: LN:FR
- D**: 02/02/14
- E**: 03/05/14
- F**: 4357953/ 0
- G**: 450663 /000
- H**: ***COPY***
- I**: Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX

Page 1 of 1

Reference	Field Name	X position Column	Y position Line	Width	Height
A	InvoiceNo	82	2	8	1
B	EstType	6	3	2	1
C	Language	22	3	2	1
D	InvoiceDate	82	6	8	1
E	PaymentDate	82	9	8	1
F	CustRef	52	9	12	1
G	AccountNo	81	12	12	1
H	Copy	24	13	10	2
I	NameAddress	4	16	50	8


Step 5 – Document Grouping

The original source data contains multiple documents and as such the Transform project should treat each one as a separate record. To separate the documents we are going to group when the InvoiceNo from the Header section changes.

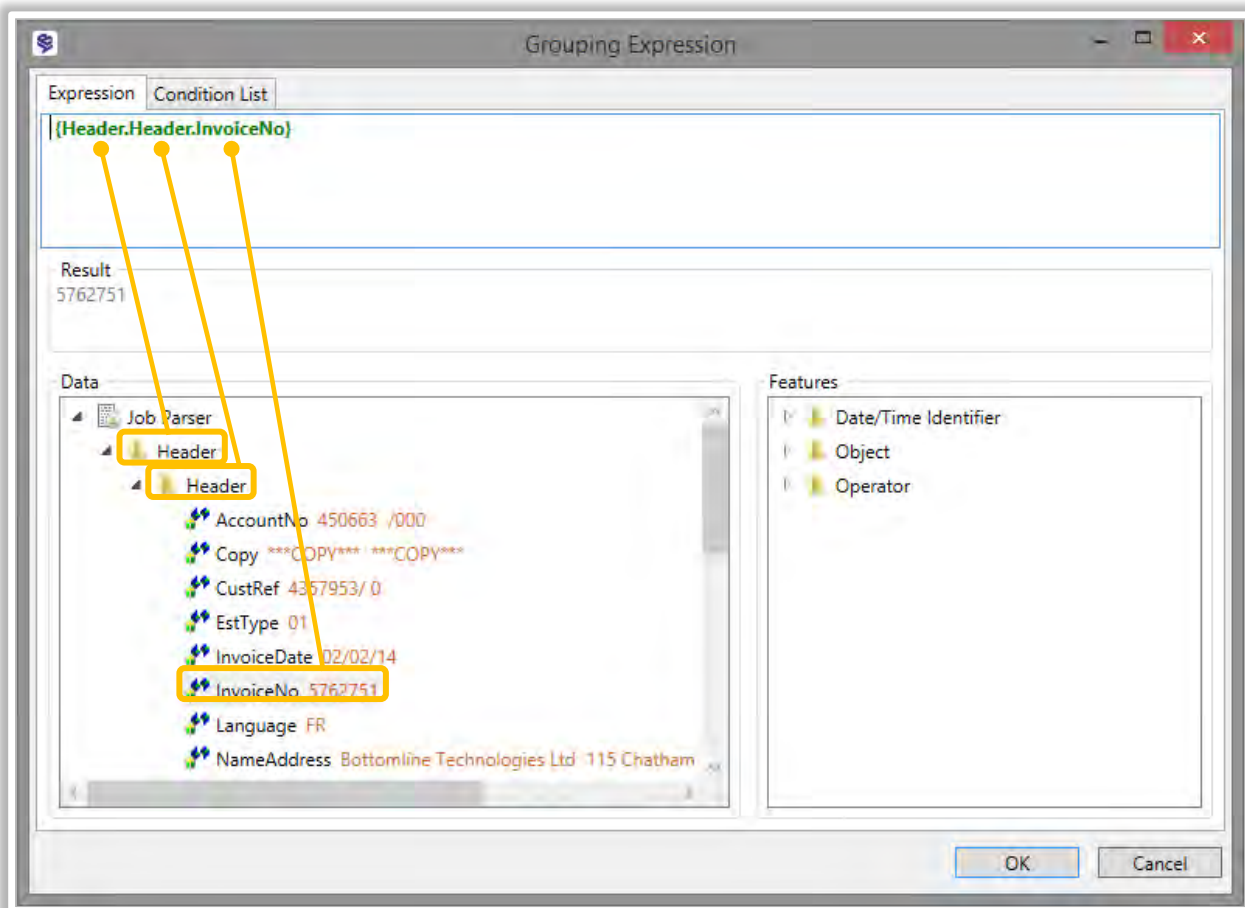
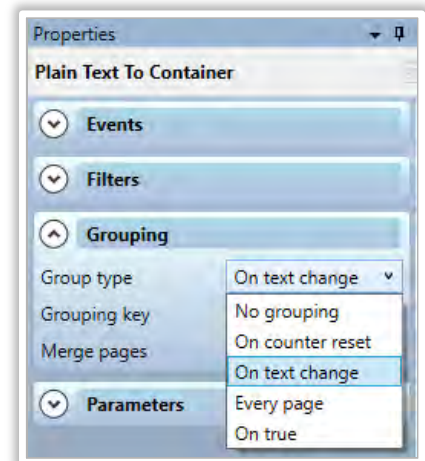
Group type

From Plain Text to Container properties window set the Group type to **On text change**.

Grouping key

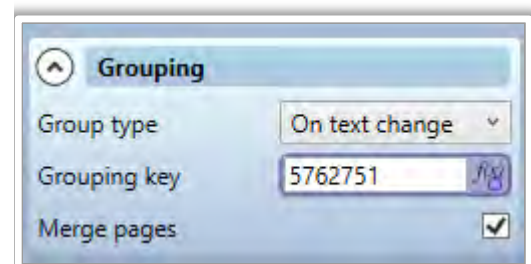
To set the Grouping key click  to open the Grouping Expression editor window.

Once open remove the double quotes from the Expression window and then select InvoiceNo as the grouping key. Double-click the element in Data window to add it to the Expression window. Data elements are referenced by their path within the data container as illustrated below.



Merge pages



Specifies whether or not to retain the page structure of the input data in the Output container. Select this to merge the section data from all pages of the input.

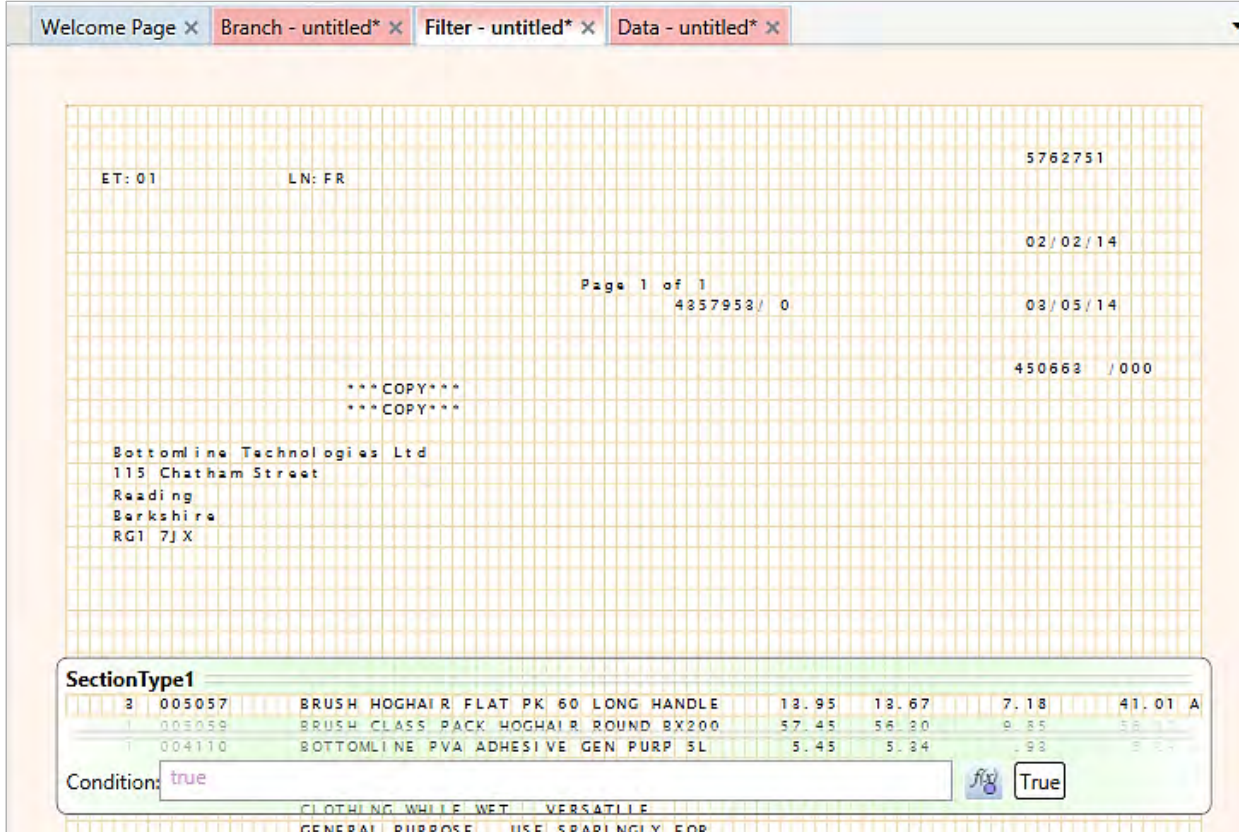


Next we need to create the following custom sections.

- Footer
- Item Lines
- Description Lines

Footer

Click Last Record  on the Navigate toolbar. Click Add new section type  the Section Filter toolbar.



ET: 01 LN: FR 5762751

02/02/14

Page 1 of 1
4357953/ 0


03/05/14

450663 /000

*** COPY ***
*** COPY ***

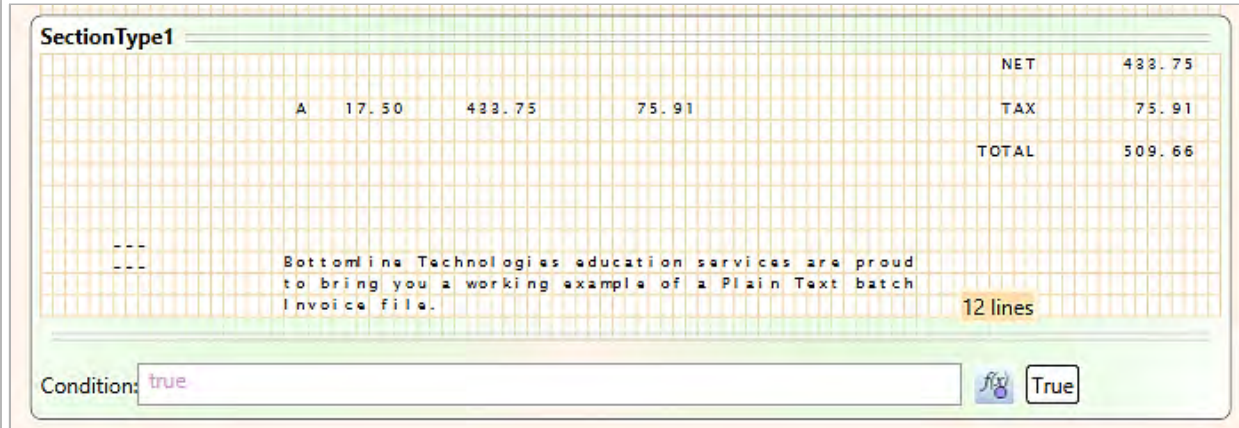
Bottomline Technologies Ltd
115 Chatham Street
Reading
Berkshire
RG1 7JX

SectionType1						
3	005057	BRUSH HOGHAIR FLAT PK 60 LONG HANDLE	13.95	13.67	7.18	41.01 A
1	003059	BRUSH CLASS PACK HOGHAIR ROUND BX200	57.45	56.20	9.35	56.17
1	004110	BOTTOMLINE PVA ADHESIVE GEN PURP 5L	5.45	5.34	1.93	5.34

Condition: true 

CLOTHING WHILE WET VERSATILE
GENERAL PURPOSE. USE SPARINGLY FOR

Drag the section down the page until the first line contains the NET data field. Then resize the section to 12 lines.




SectionType1

				NET	433.75
A	17.50	433.75	75.91	TAX	75.91
				TOTAL	509.66

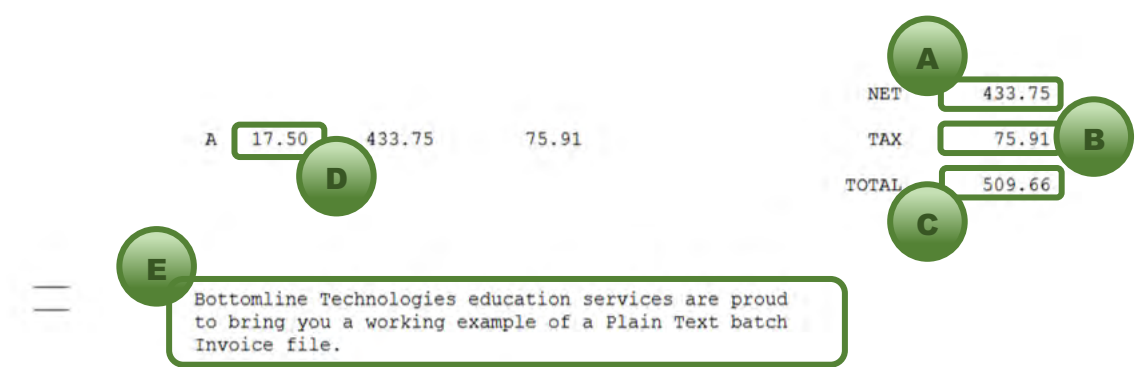
Bottomline Technologies education services are proud to bring you a working example of a Plain Text batch Invoice file.

12 lines

Condition: true 

Footer Section Definition

Custom Footer section only appears on final page of each Invoice (document).



Reference	Field Name	X position Column	Y position Line	Width	Height
A	Net	84	2	12	1
B	Tax	84	4	12	1
C	Total	84	6	12	1
D	TaxRate	24	6	6	1
E	StrapLine	20	11	70	3

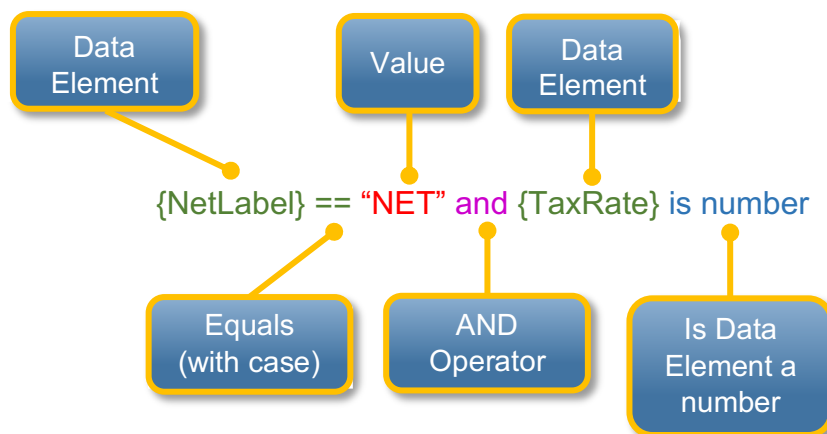
Rename the Section from **SectionType1** to **InvoiceFooter**. Then map the blocks as described on page 56.

A	NET	TAX	TOTAL
17.50	433.75	75.91	509.66

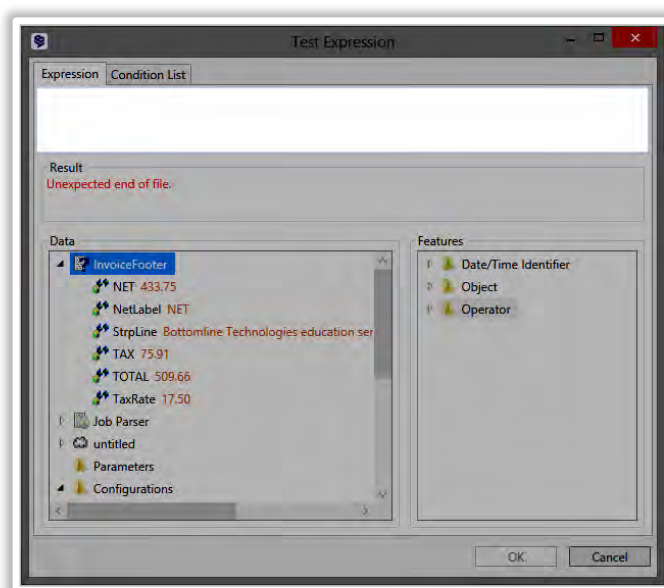
Bottomline Technologies education services are proud to bring you a working example of a Plain Text batch invoice file.

Condition: true

To make the InvoiceFooter custom section conditional click open Test Expression window. The condition used to identify the InvoiceFooter section is shown below.



To build the above expression complete the following steps.



Clear Expression

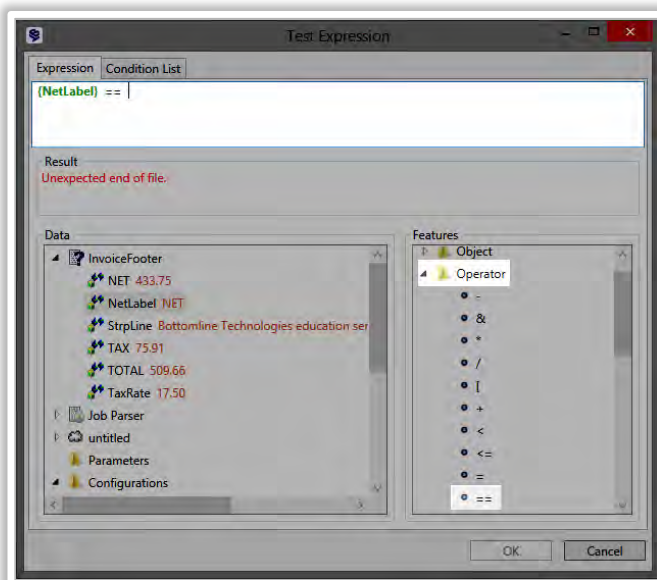
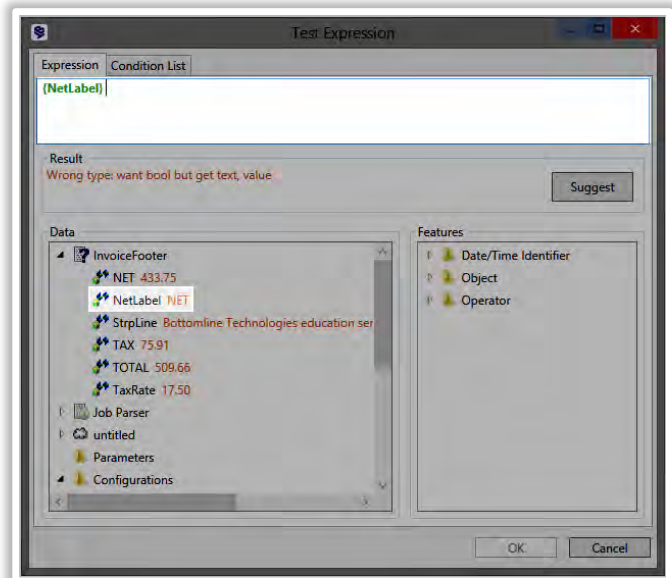
Remove **true** from the Expression window.

Comparison

Double-click **NetLabel** data element within the data pane. This will place the path of the element in the Expression window.

Move the cursor to the right of the element.

```
{NetLabel} |
```



Operator

Within the Features pane expand the Operator section. Then double-click the == (Equals case sensitive) to add it to the expression.

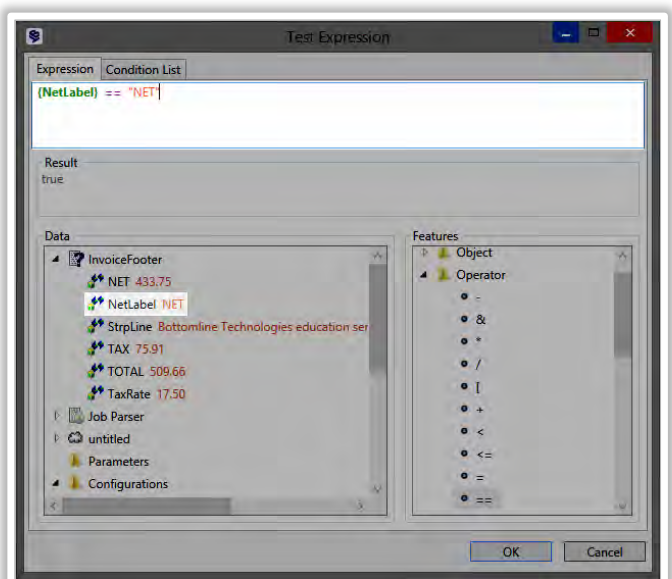
```
{NetLabel} == |
```

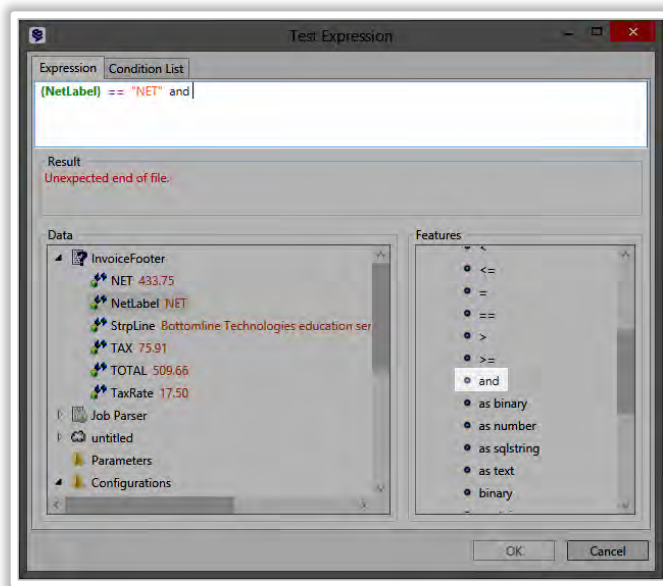
Value

Always enclose Text or Unicode Text values in double quotes.

Type "NET"

```
{NetLabel} == "NET" |
```





Operator

To make the expression stronger we want to add a second comparison. Double-click and from the Features Operator list to add it to the expression.

Move the cursor to the right of the expression.

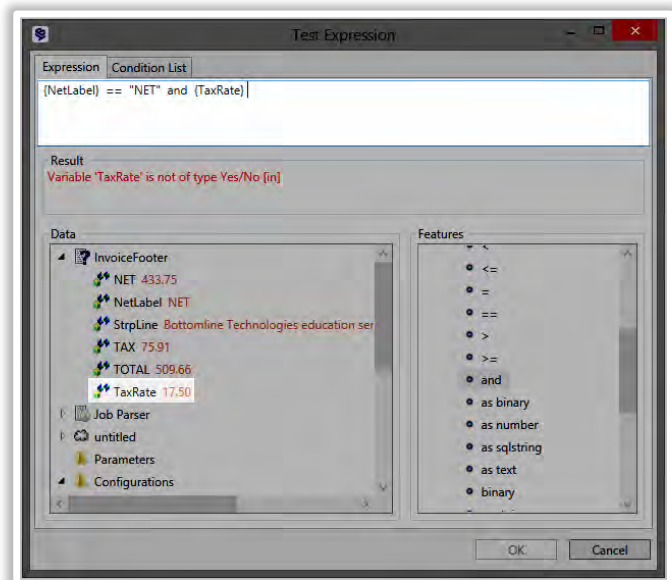
```
{NetLabel} == "NET" and |
```

Comparison

Double-click the TaxRate element from the data pane to add it to the expression

Move the cursor to the right of the expression.

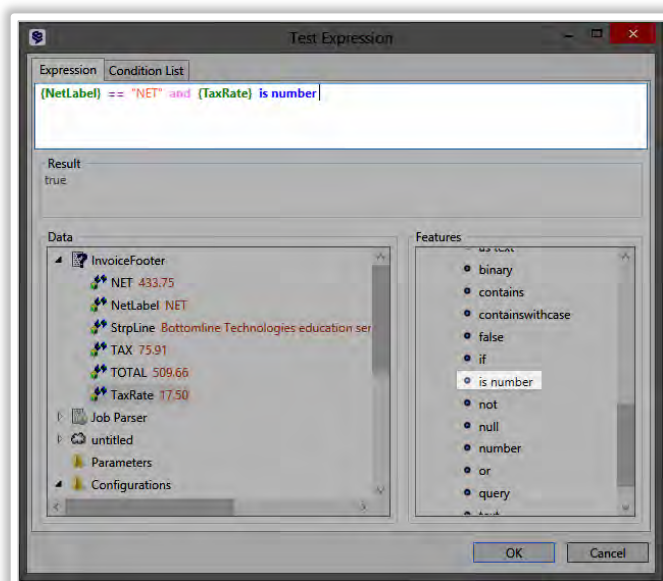
```
{NetLabel} == "NET" and {TaxRate} |
```



Operator

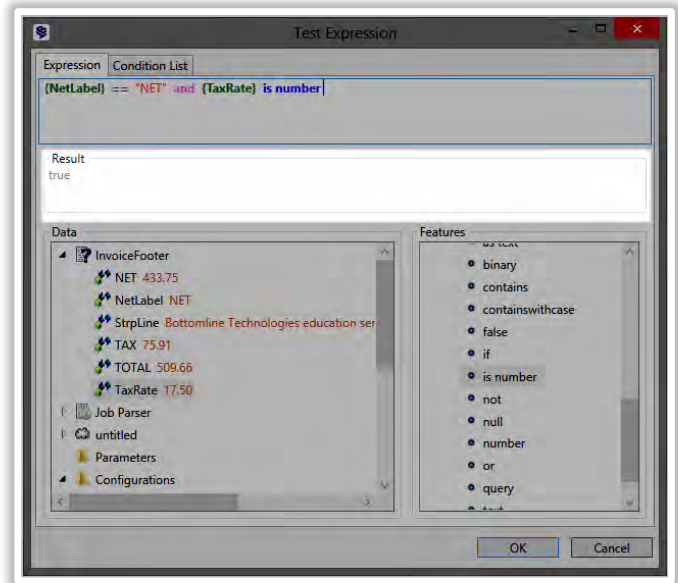
To test the TaxRate value is a number double-click is number from the Features Operator list to complete the expression

```
{NetLabel} == "NET" and {TaxRate} is number
```



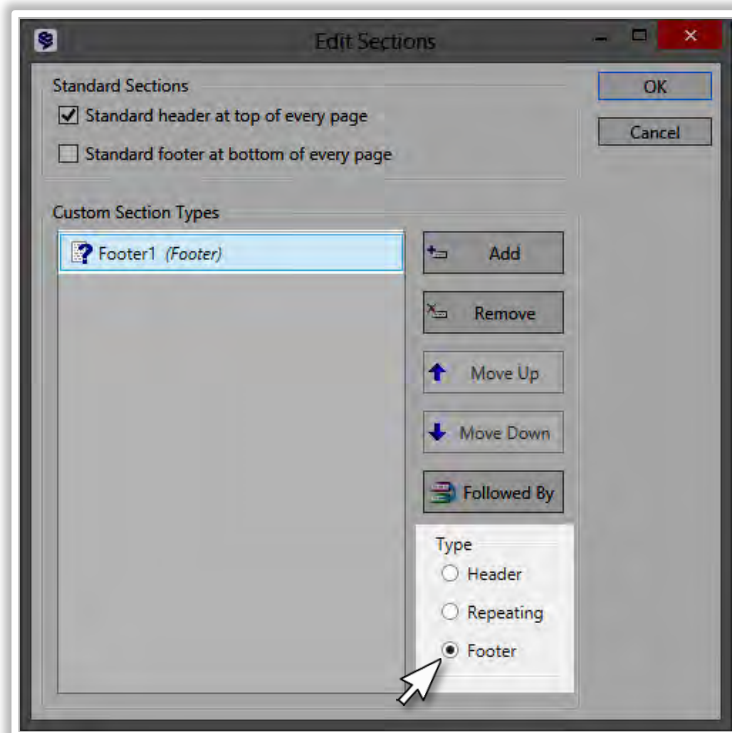
Result

The outcome of the Boolean expression is displayed in the Results pane. Click **OK** to close the Test Expression window.




Section Type

Each new custom section added defaults to a Repeating section type. Click  on the Section Filter toolbar, from the Edit Sections dialog box select **Footer1** from the Custom Section Types list and change the Type to Footer as shown.

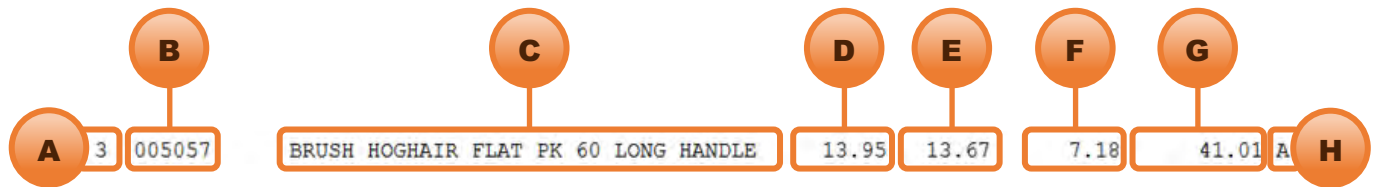


Item Lines

Click Add new section type  on the Section Filter toolbar. Rename the custom section to **ItemLine** and map the fields as described below

Section Definition

Custom repeated section as an Invoice can contain multiple item lines that span over pages.



The diagram illustrates the mapping of fields A through H to a specific layout. Field A is at position 3 with a width of 1. Field B is at position 8 with a width of 7. Field C is at position 20 with a width of 38. Field D is at position 59 with a width of 7. Field E is at position 67 with a width of 7. Field F is at position 75 with a width of 9. Field G is at position 85 with a width of 10. Field H is at position 96 with a width of 1. The text line shown is: 3 005057 BRUSH HOGHAIR FLAT PK 60 LONG HANDLE 13.95 13.67 7.18 41.01 A

Reference	Field Name	X position Column	Y position Line	Width	Height
A	QTY	3	0	4	1
B	ItemCode	8	0	7	1
C	Description	20	0	38	1
D	UnitPrice	59	0	7	1
E	DiscountPrice	67	0	7	1
F	Tax	75	0	9	1
G	LineTotal	85	0	10	1
H	TaxPoint	96	0	1	1

ItemLines Section Expression

ItemLine	5	004019	BOTTOMLINE GLUE STICKS PK 12	6.95	6.81	5.96	34.06	A
	013995		BOTTOMLINE PENCIL ERASERS 40 PER BOX	2.00	1.96	1.72	0.88	
Condition:	{QTY} is number	and not	{Description} = ""	and	{UnitPrice} is number			
	013995		PENCIL SHARPENER METAL 1 HOLE BOX 20	2.95	2.87	2.61	1.55	

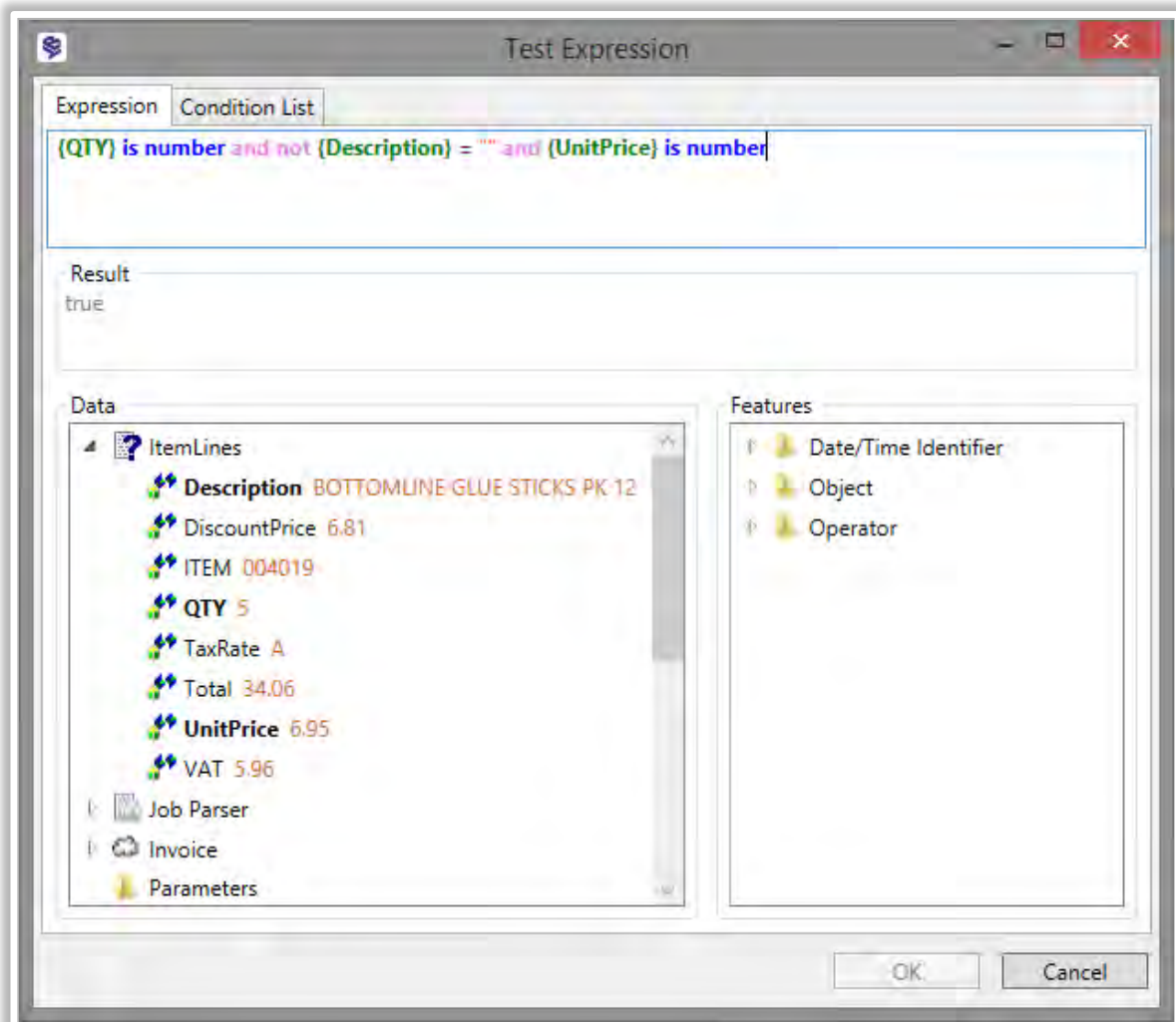
{QTY} is number and not {Description} = "" and {UnitPrice} is number

Tests whether the **QTY** value is a floating-point number

Tests whether the **Description** value is not empty

Tests whether the **UnitPrice** value is a floating-point number

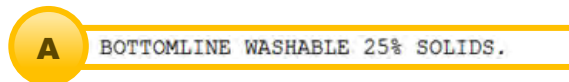
Create the above condition, using the Test Expression Editor.



Description Lines

Click Add new section type  on the Section Filter toolbar. Rename the custom section to **DescriptionLine** and map the fields as described below.

Section Definition



Reference	Field Name	X position Column	Y position Line	Width	Height
A	Description	20	0	38	1

Description Section Expression

DescriptionLine	004092	LOCTITE HOT MELT GLUE STICKS PK 6	1.95	1.92	3.34
		11.2mm x 200mm			
	004110	BOTTOMLINE PVA ADHESIVE GEN PURP 5L	5.45	5.34	.93
Condition:	[QTY] = "" and not [Description] = "" and [UnitPrice] = "" and [Total] = ""				

{QTY} = "" and not {Description} = "" and {UnitPrice} = "" and {Total} = ""

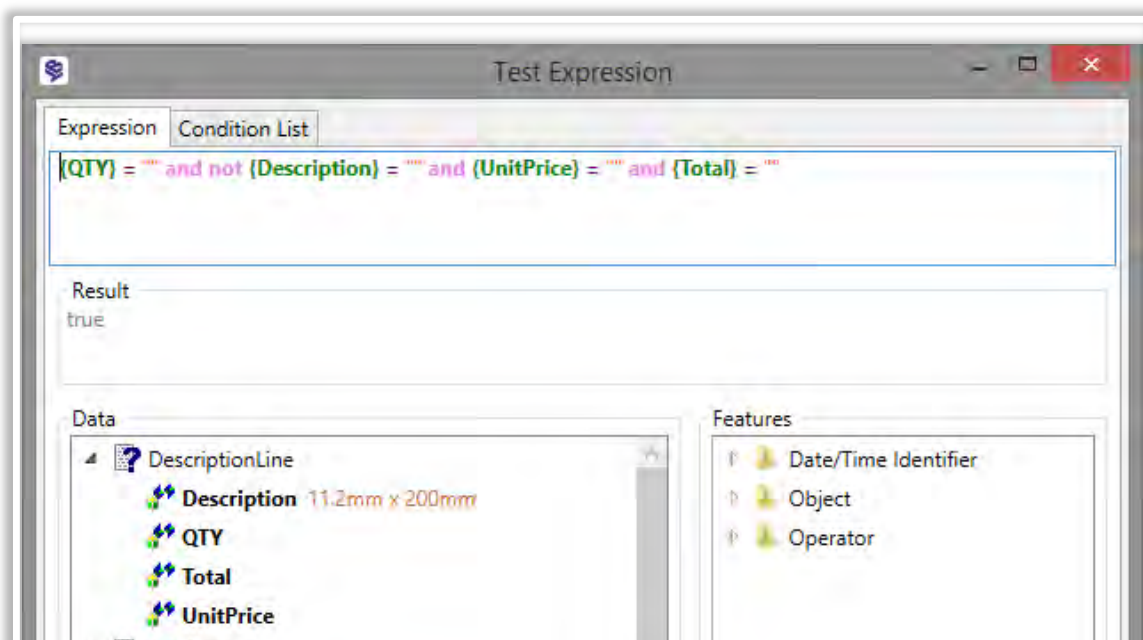
Tests whether the **QTY** value is empty

Tests whether the **Description** value is not empty

Tests whether the **UnitPrice** value is empty

Tests whether the **Total** value is empty

Create the above condition, using the Test Expression Editor.



Step 7 - Update Container

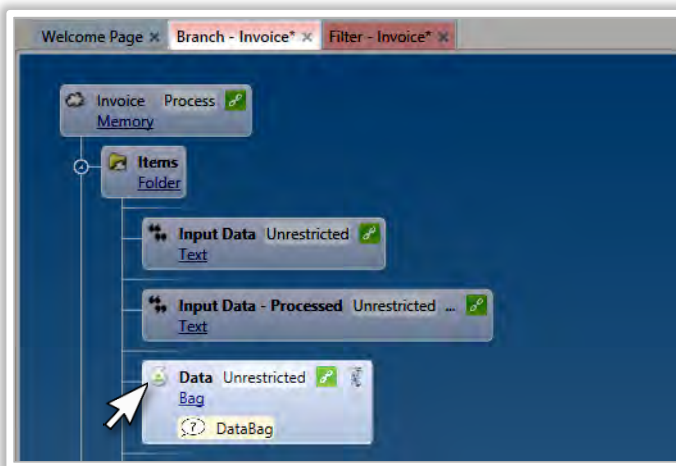
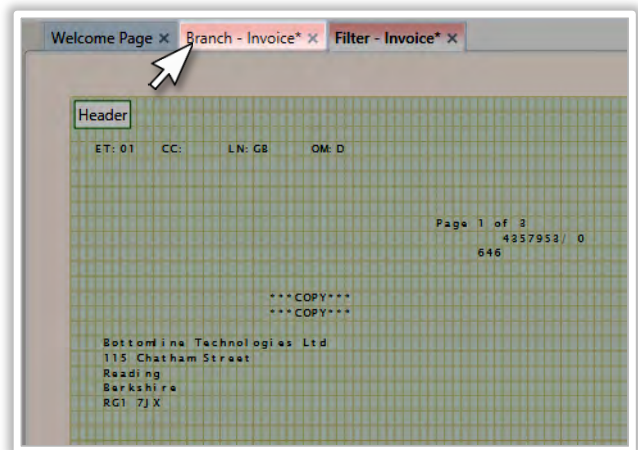
Following the creation of the sections and the fields it is recommended that you check the structure of the data by inspecting the container bag.


Follow the steps below to update the container and inspect the contents.

On the Section Filter toolbar click  update container.

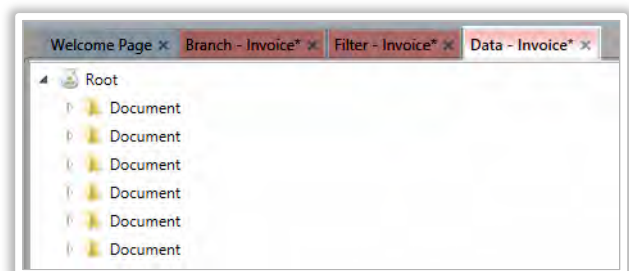


Select the Branch editor view tab



From the branch click  Bag object to open the Data view tab.

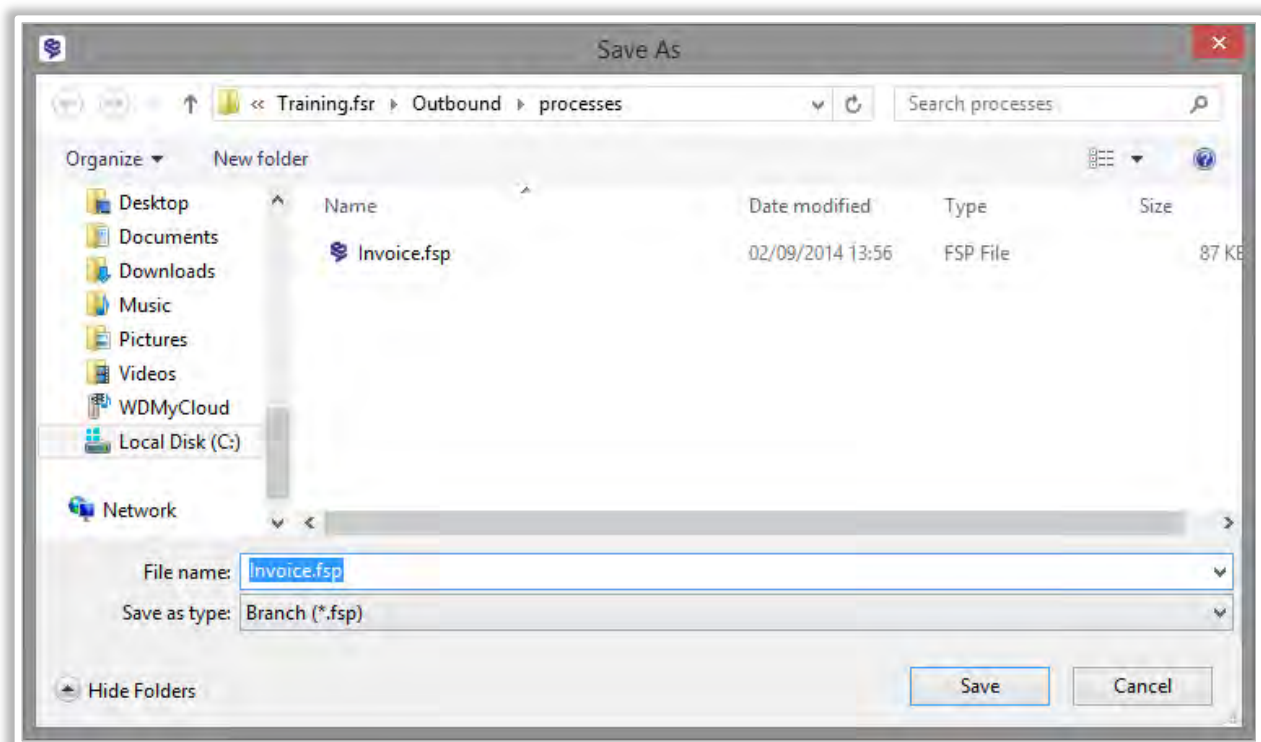
From the Data view use the expand controls to inspect the structure of the container.



Step 8 - Save the Branch

From the Menu bar select file > Save As and save the branch as follows:

Training.fsr\Outbound\processes\Invoice.fsp



Project Checklist

1. ~~Repository Design~~
2. ~~Create Local Repository~~
3. ~~Language Labels~~
4. ~~Global Resources~~
5. ~~Plain Text Template Wizard~~
6. ~~Plain Text to Container~~

7. **Modify Branch Structure**

GlobalConstants Branch Shortcut

Input Method

Scan Through Each Document

Delivery Method

8. ~~Dynamic Branch Shortcut~~
9. ~~Document Organizer~~
10. ~~Synchronize and Deploy~~



Modify Branch Structure

During the Plain Text template wizard a branch structure was created. This branch can be used as a starting point for the project, it contains required objects for a process branch.

Now we have a well formed branch we can modify the structure to meet the demands of the project.

We are going to modify the following items.

- Input Method
- Scan Through Each Document
- Output Method

Input Method

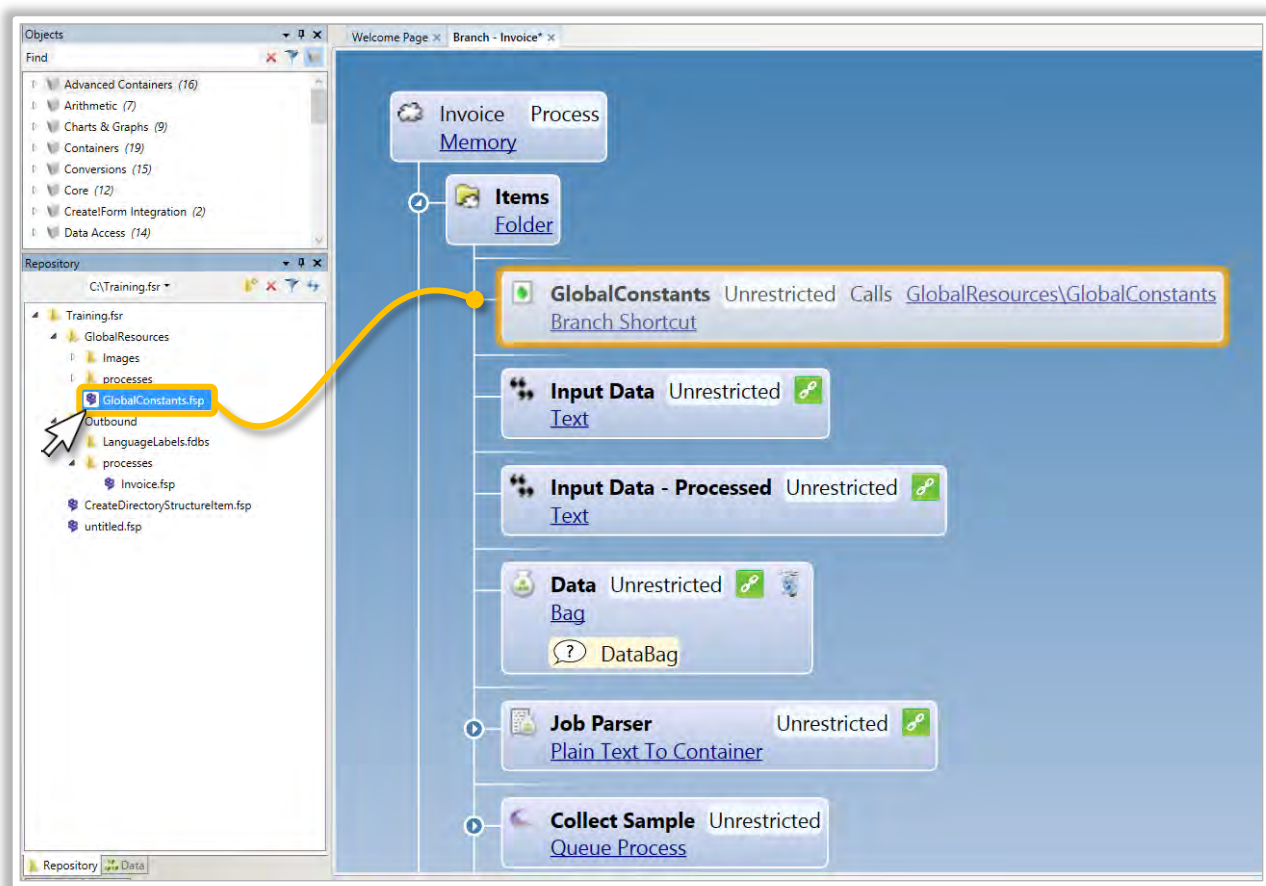
The default input method for the Plain Text template is Print Queue. The project scope dictates that the required input method is File Queue.

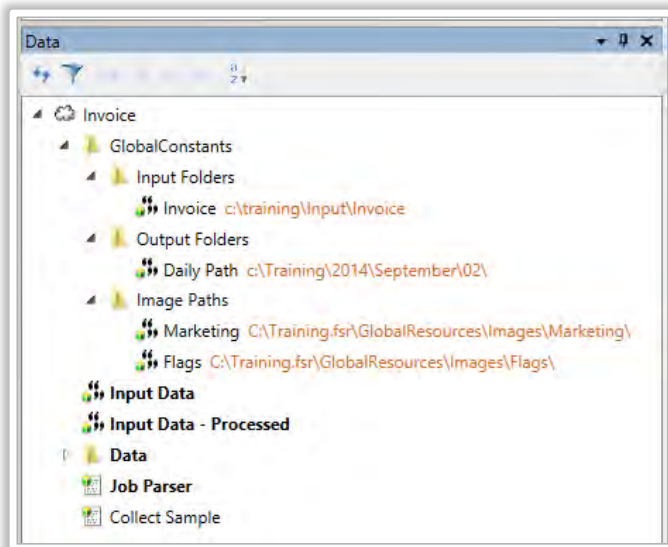
Complete the following steps to change the input method.

Step 1 - GlobalConstants Branch Shortcut

During the process of creating GlobalResources we built a branch called GlobalConstants which contains the input path to be used as the watch folder where the ERP invoice spool files will be written.

To make use of this branch create a branch shortcut within the memory items folder. From the Repository window expand GlobalResources select GlobalConstants and drag and drop it onto the branch beneath the **Items** Folder as shown below.



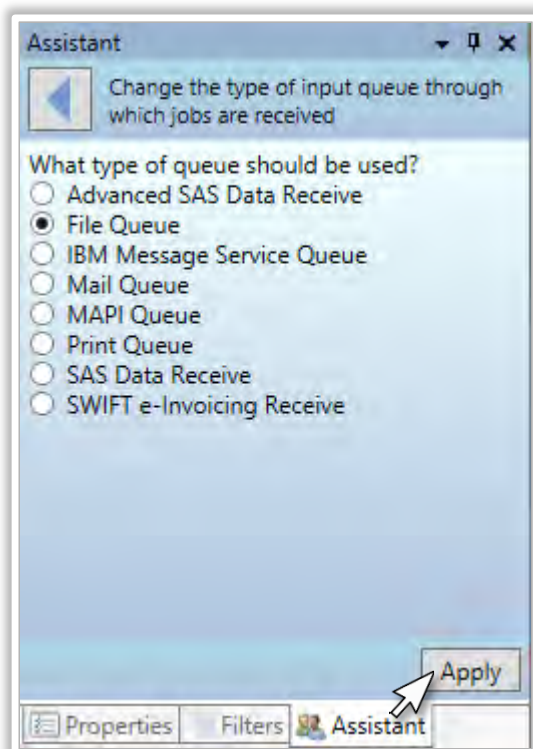
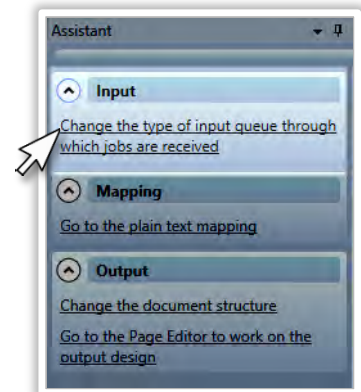


Adding the GlobalConstants to the memory items list makes the objects available to the branch via the Data palette.

Open the Data palette to reveal the GlobalConstants then click the expand control to view the values.

Step 2 – Assistant

From the Assistant Input section select **Change the type of input queue through which jobs are received.**

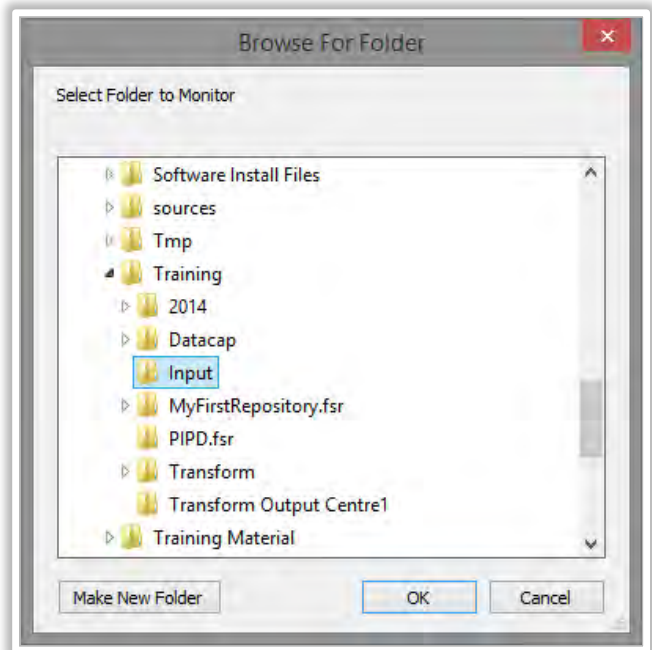


Step 3 – File Queue

From the list What type of queue should be used? select File Queue and click Apply at the bottom right of the window.

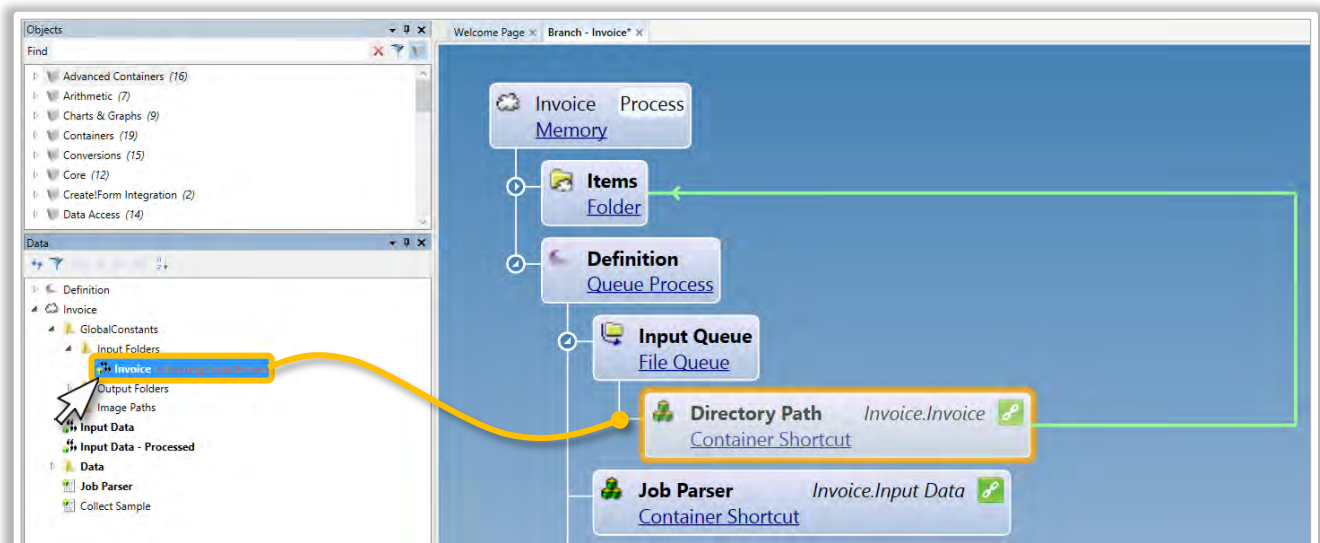
Step 4 – Select Folder to Monitor

From the Browse For Folder window select a folder to complete the Assistant steps.



Step 5 – Container Shortcut to Input Path

To use the Input Directory Path defined in the GlobalConstants expand the File Queue object on the branch, locate from the Data palette Input Folders Invoice and drag and drop it over the current Text object to create a Container Shortcut as shown below.



Scan Through Each Document

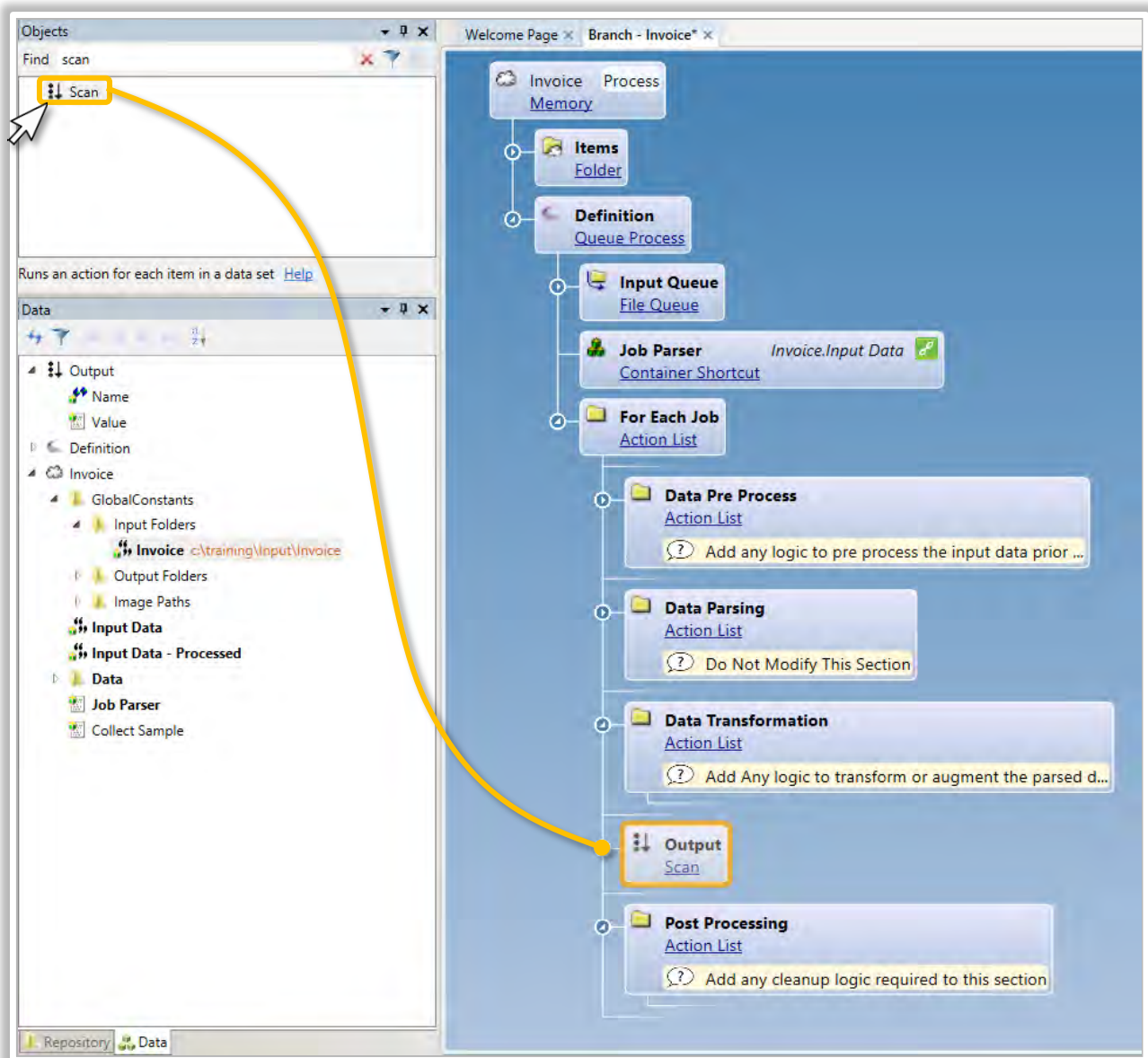
The Plain Text template branch creates an Action List to complete the Output processing. The branch structure is configured with a **Print Graphic** and **Reporter** object which loops through the documents in the data set and produces the output defined.

To perform multiple actions for each document within the data set we need to change the current structure so it can complete the required actions.

- **Dynamic Branch Shortcut** - Call language labels.
- **Append text** – Record document key fields to audit file.
- **Print Graphic** – Produce the output

Step 1 - Add Scan Object

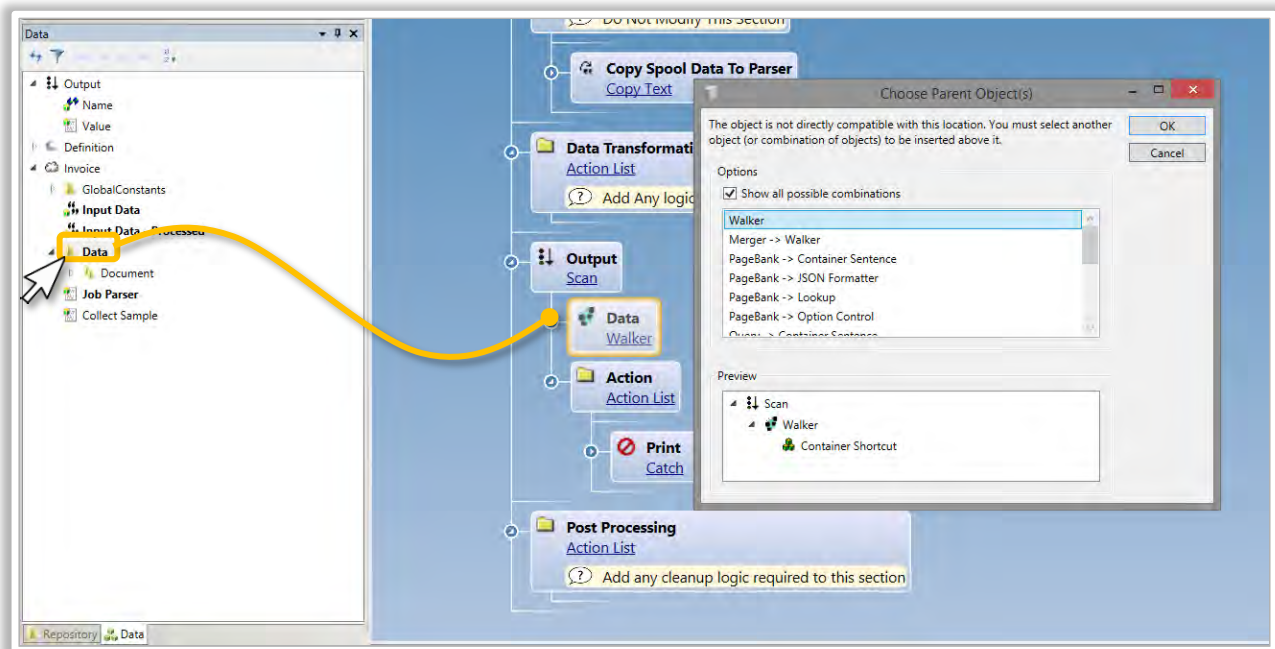
To expose each document so that we can perform the necessary actions use a Scan object. From the Objects palette Find field type **scan** then drag and drop the **Scan** object over the current **Action List** labelled Output.



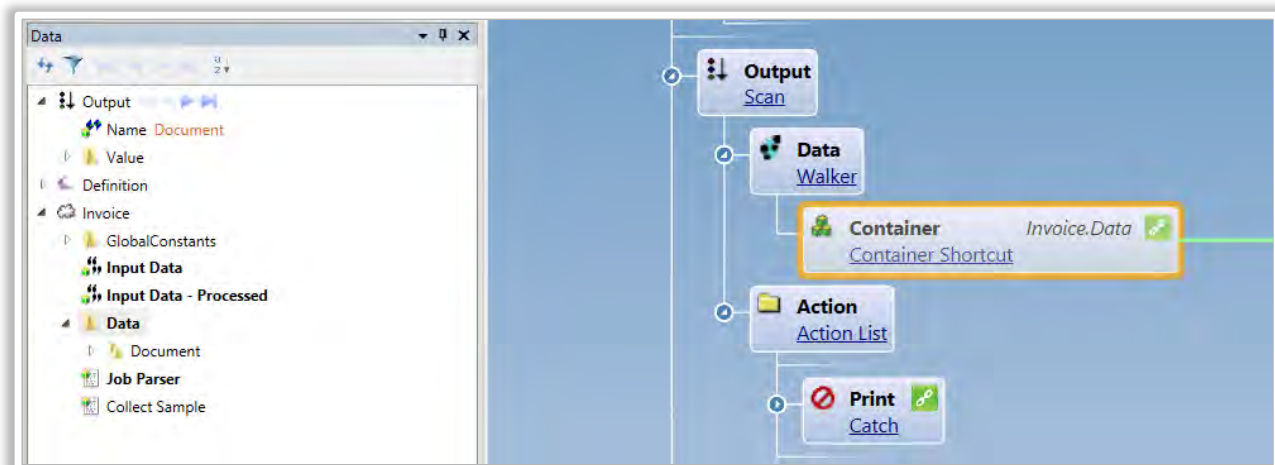
Step 2 – Define Record Set

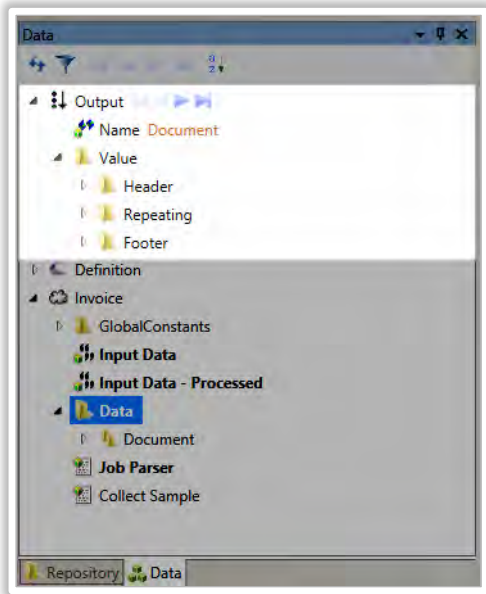
Add a **Walker** object beneath the parent **Scan** object to step through the elements in a container. The Scan then acts as the container for the current record, so you can drag an element from the Data palette and create a Container Shortcut that refers to an element within the current record.

Expand the **Scan** object and from the Data palette drag and drop Data container onto the object directly beneath the **Scan** object. Depending on the preferences this will be either a **Reporter** or **Walker** object. You will be presented with the Choose Parent Object(s) dialog box, select Walker and click OK to continue.



This will create the child Container Shortcut object linked to the *Invoice.Data* container.





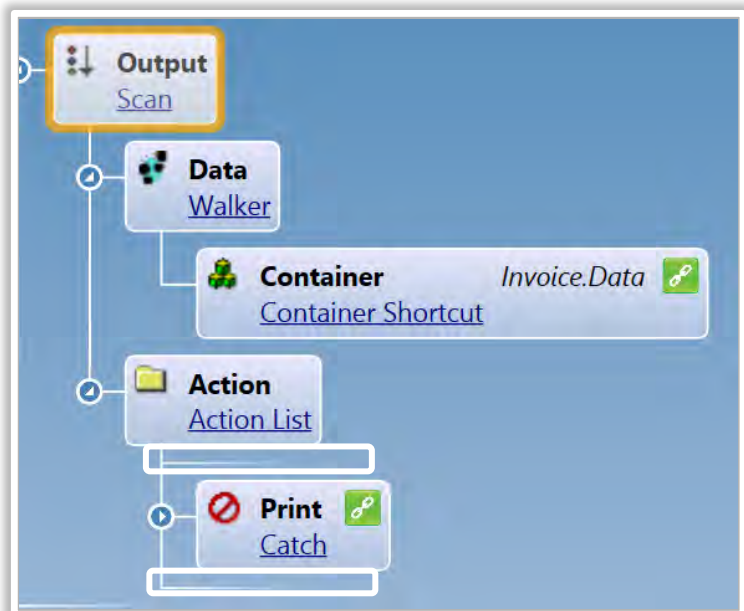
Important

Always check that you have selected to correct container to walk through, the Data palette displays the Scan with the name of the container and its content. The navigate buttons also become visible allowing you to iterate through the records to ensure the values within the container are correct.

Action List

The Action List is a child of the **Scan** object which allows you to add additional actions for each record in the data set as it steps through the list.

The add points are available within the Action List as shown on the image to the right.

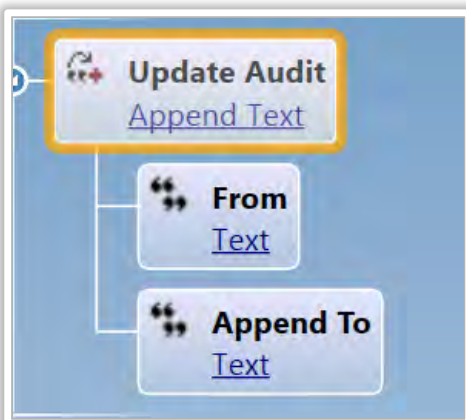
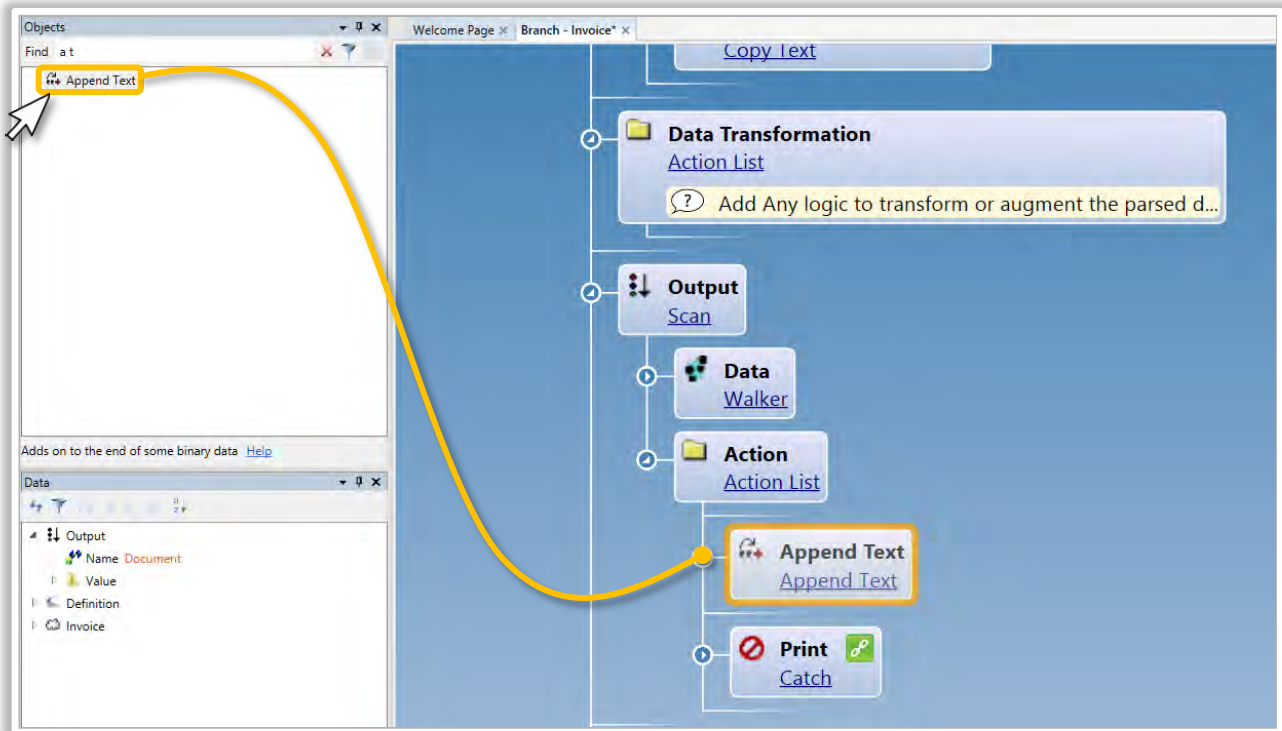


Append Text

To create an Audit each time the project runs we will add an Append Text object so we can record details of each invoice as we step through each document in the data set.

Step 1 – Add Append Text

From the Objects palette Find field type **a t** select **Append Text** drag and drop onto the branch so that it is the first object beneath the Action List.



Step 2 - Rename

Rename the object to **Update Audit**. Click the expand control to reveal the child objects, by default the **Append Text** object has From and Append To Text objects.

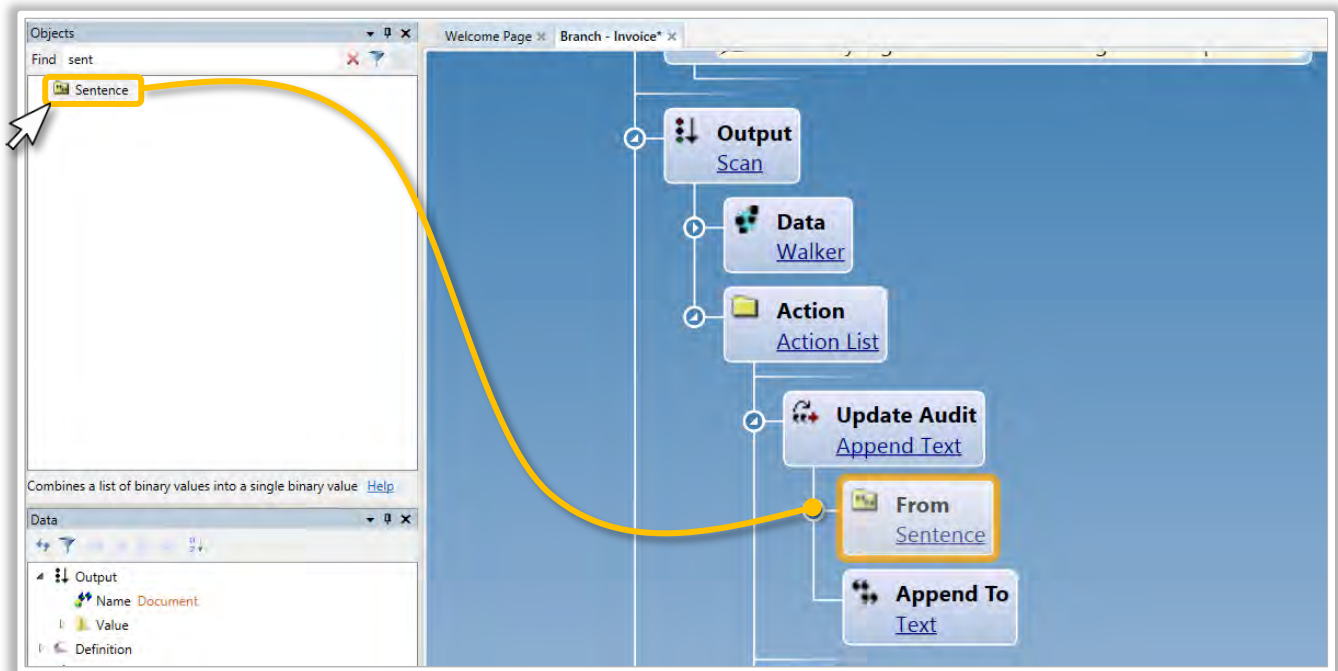
Step 3 – Build Audit Entry

The audit file need to be produced in comma separated values (CSV) format, each record containing the following values and terminated with a carriage return line feed.

Invoice Date,InvoiceNo,AccountNo,NetTotal,Tax,Total,EstType

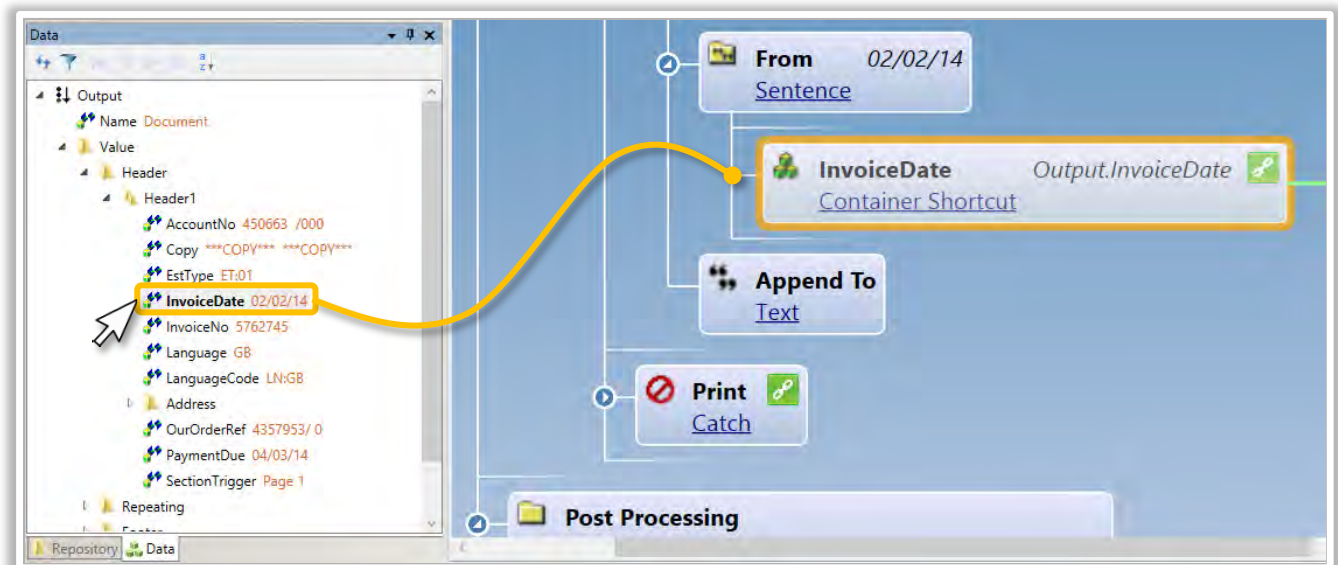
To be able to build the audit file we need to change the From **Text** object to a **Sentence** object so that we can combine the required values into a string.

From the Objects palette Find field type sent select **Sentence** and drag it onto the branch over the From **Text** object.



Step 4 – Container Shortcut

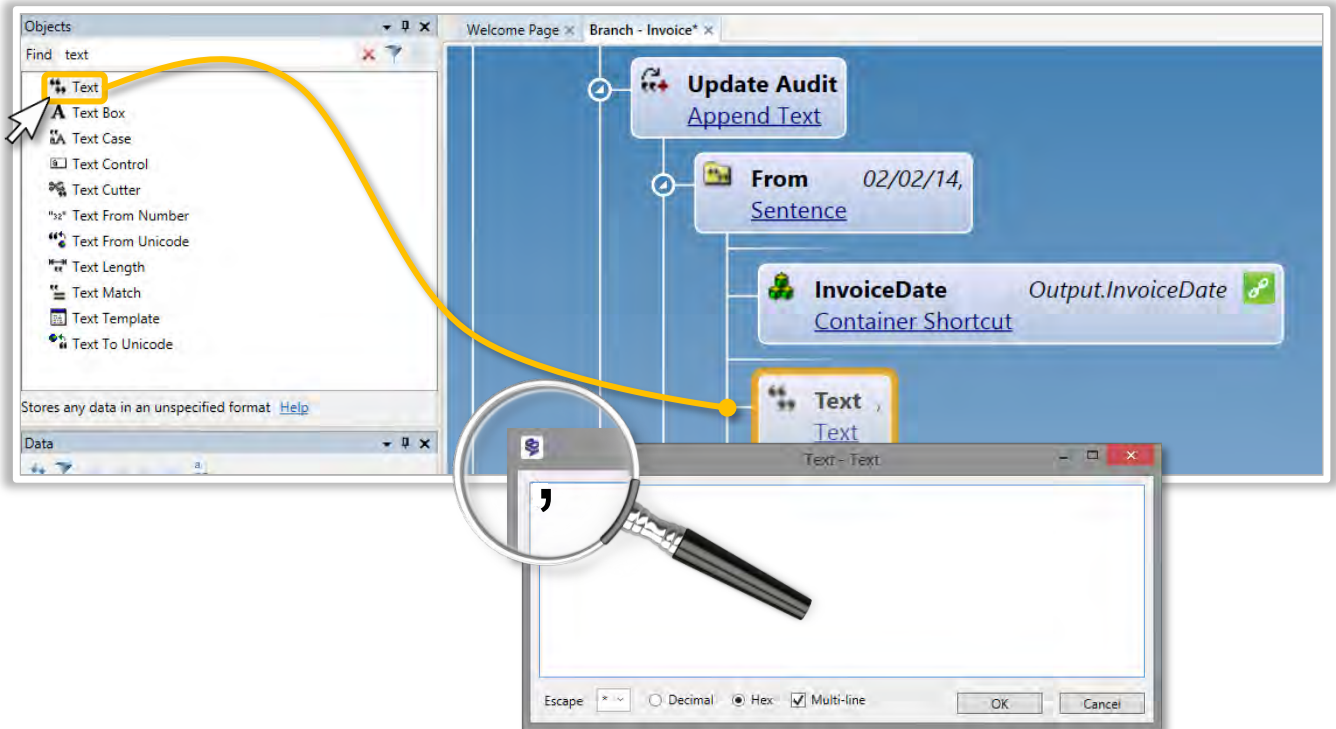
Expand the **Sentence** object, then from the Data palette expand Value > Header > Header1 and select **InvoiceDate** and drag it onto the branch so that it becomes the first piece of the Sentence.



Step 5 – Add Delimiter Character

As the format of the audit file is CSV we need to separate each field within a record with a delimiter character (,) comma.

From the Object palette Find field text select the Text and drag it onto the branch beneath the InvoiceDate Container Shortcut so that it becomes the second piece of the sentence. Click “con and enter a comma.



Step 6 – Add Fields

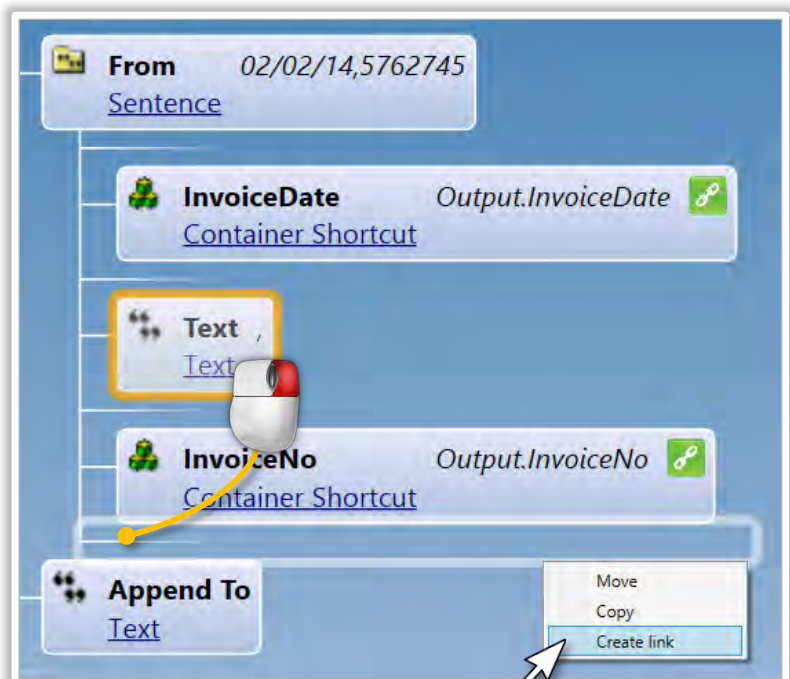
Repeat Steps 4 & 5 to add the remaining fields.



Tip

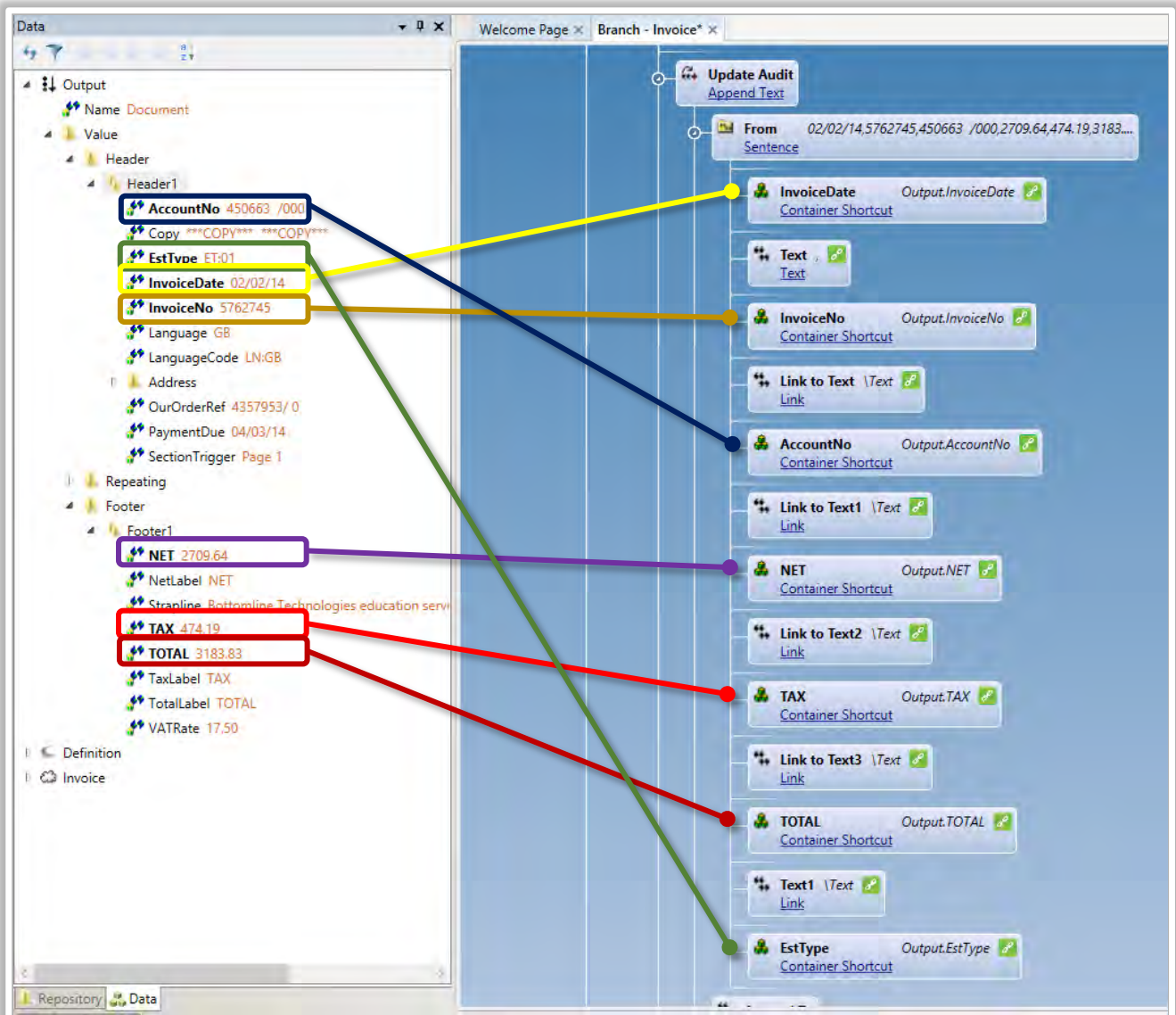
You can create a link to the first delimiter character. Right click the object and whilst holding down the right button drag it to the required target and release the button, this will display an options menu select **Create link**.

Alternatively you could create a Text memory item which becomes available in the Data palette.




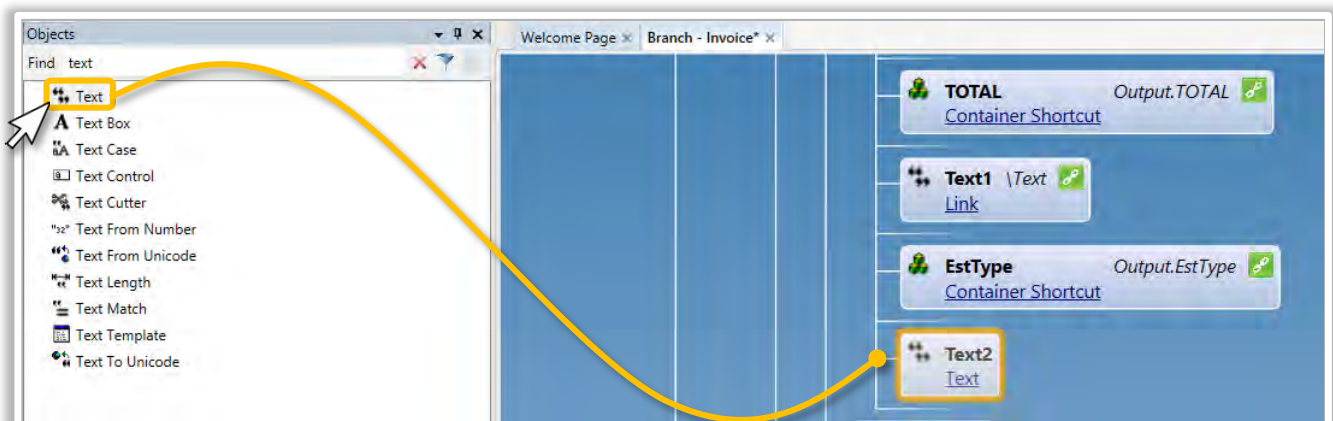
Sentence Format

The image below illustrates the structure of the Sentence required for the Audit record.



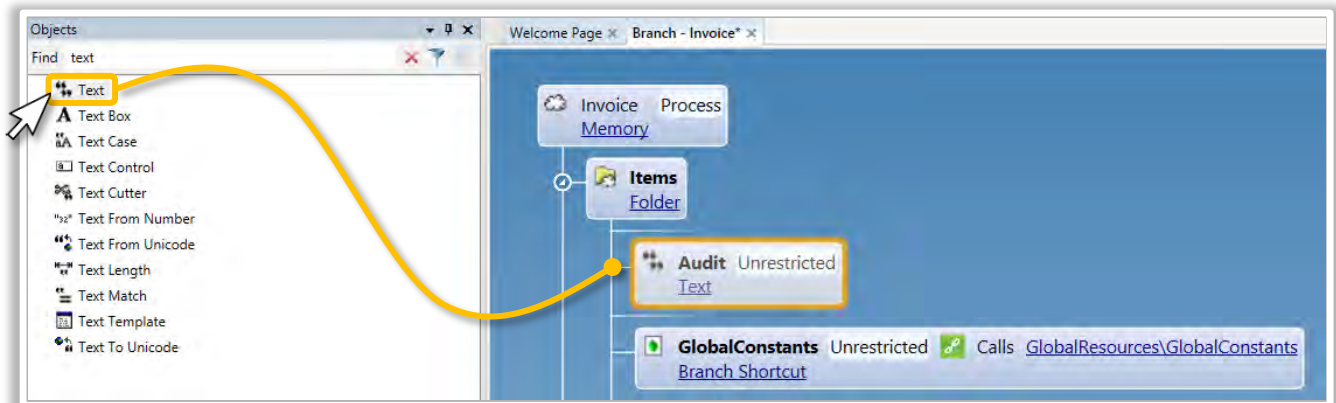
Step 7 – Record Terminator

Add a carriage return line feed as the record terminator. Use a Text object as the last item in the Sentence, click , press the Enter key.



Step 8 – Append To Memory Object

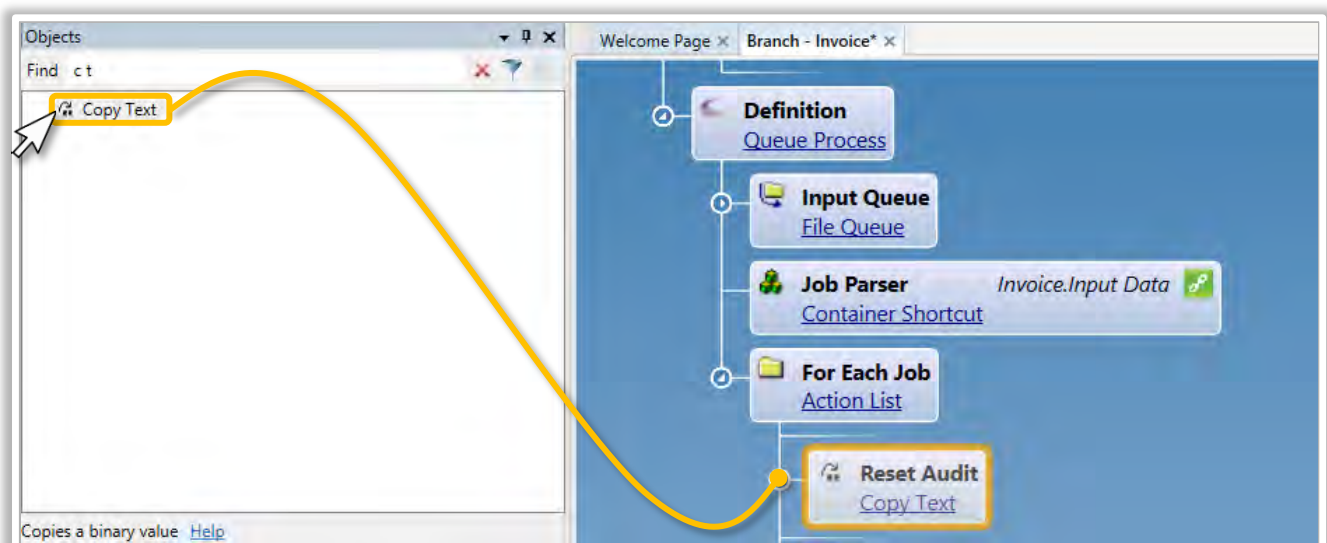
We are going to use a memory text object to append to. From the Objects palette select a Text object and drag onto the branch within the Items Folder item list and rename to Audit.



Step 9 – Initialize Memory Objects

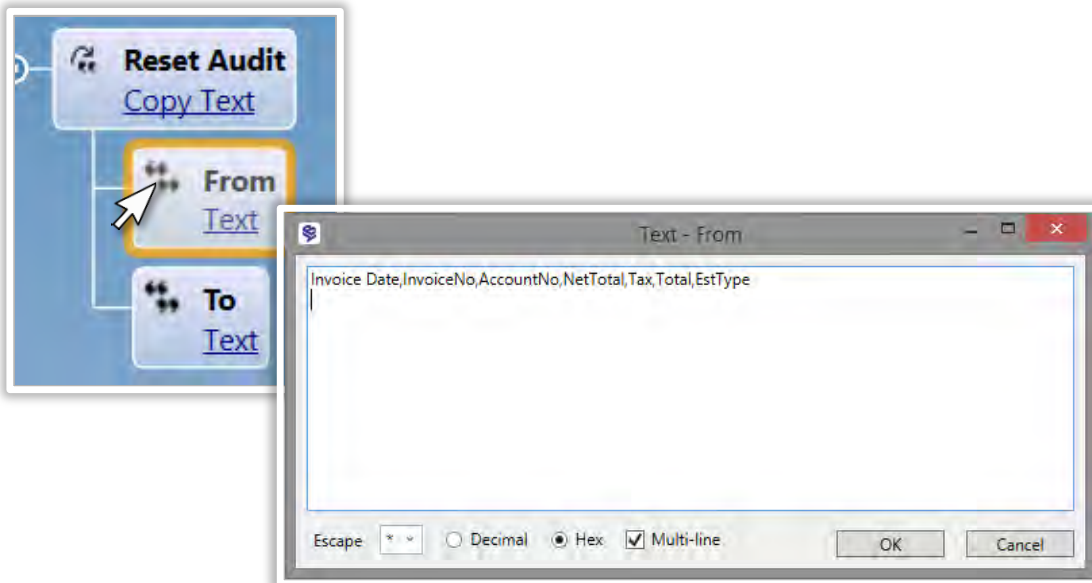
As we add items in memory it is good practice to initialize them when the project runs.

From the Objects palette Find field type **ct** select Copy Text and drag it onto the branch so that it becomes the first object under For Each Job Action List, rename the **Copy Text** object to Reset Audit.



Step 10 - Copy Text From

Expand the Copy Text object to reveal the child objects. By default the Copy Text object has two Text child objects a From and To. Click **From Text** object to open the Text-From dialog box and enter a field header row as illustrated below, remember to enter the carriage return line feed.



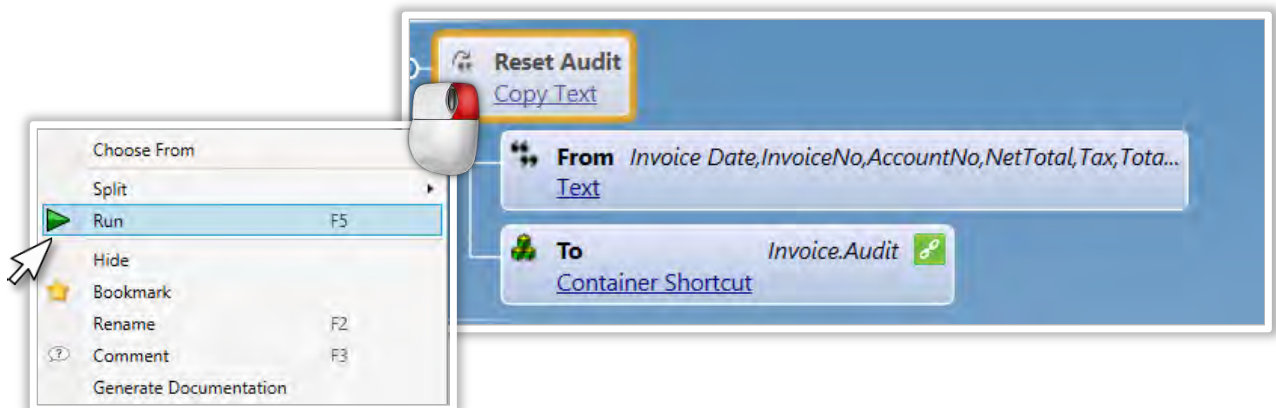
Step 11 - Copy Text To

From the Data palette expand the memory then select Audit and drag it on the To **Text** object.

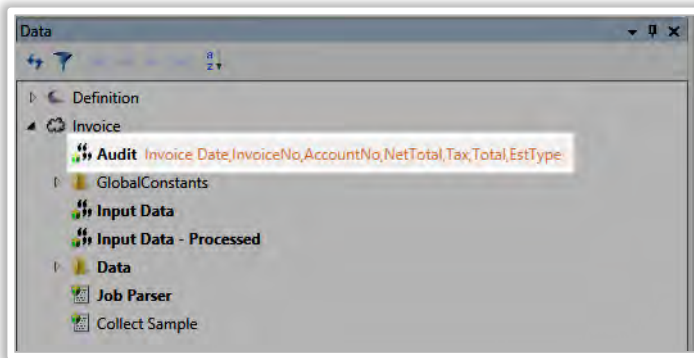


Step 12 - Test the Action

To test the Reset Audit **Copy Text** object right-click and select **Run** from the list.

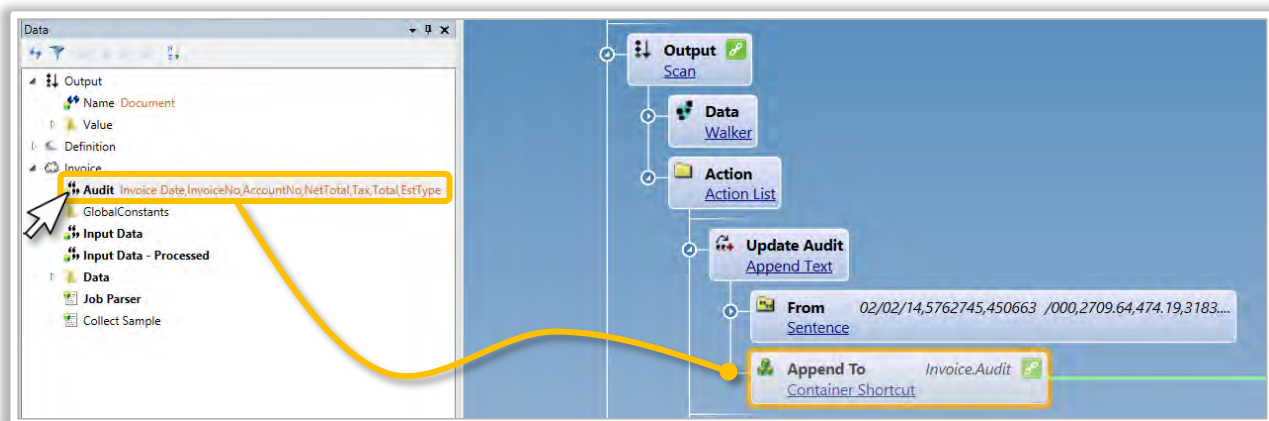


Within the Data palette the Audit item has been updated as illustrated below.



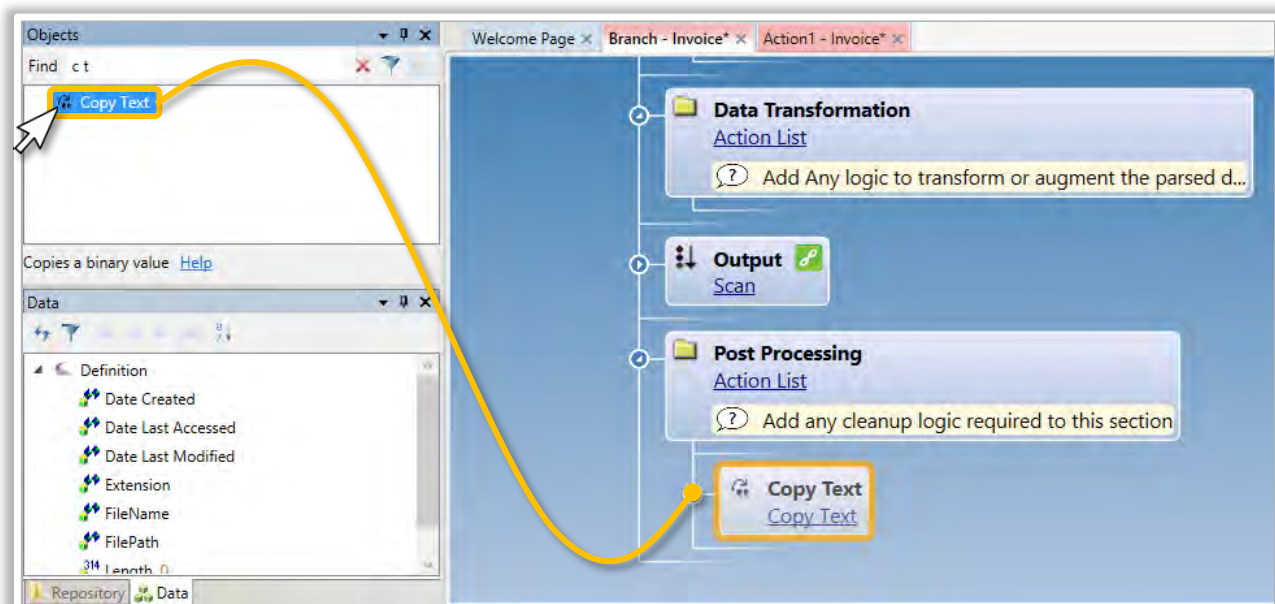
Step 13 – Append To Container Shortcut

From the Data palette select Audit and drag it onto the Append To **Text** object beneath the **Append Text** object to create the Container Shortcut.



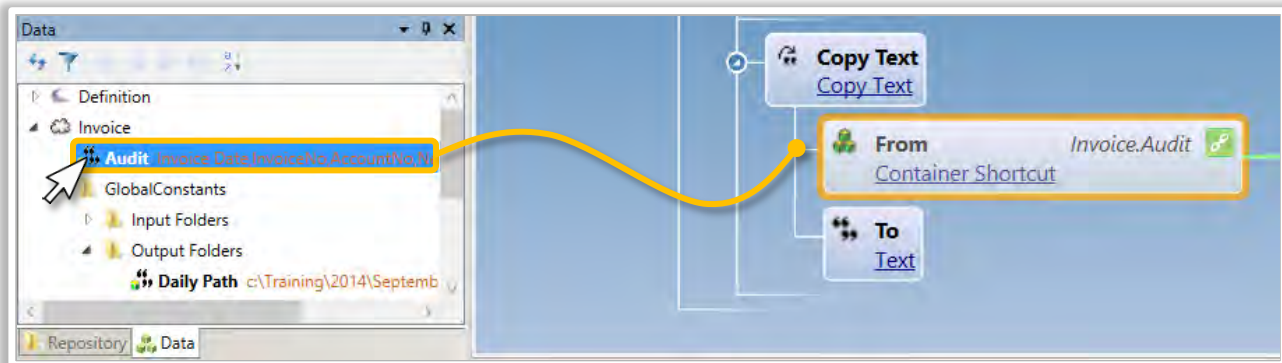
Step 14 – Write Audit to File

Before the project exits we need to take care to write the audit Text object to the file system. From the Objects palette Find field type **c t** and select Copy Text and drag it onto the branch so it becomes a child of the Action List at the bottom of the branch labelled Post Processing.



Step 15 – Copy Text From

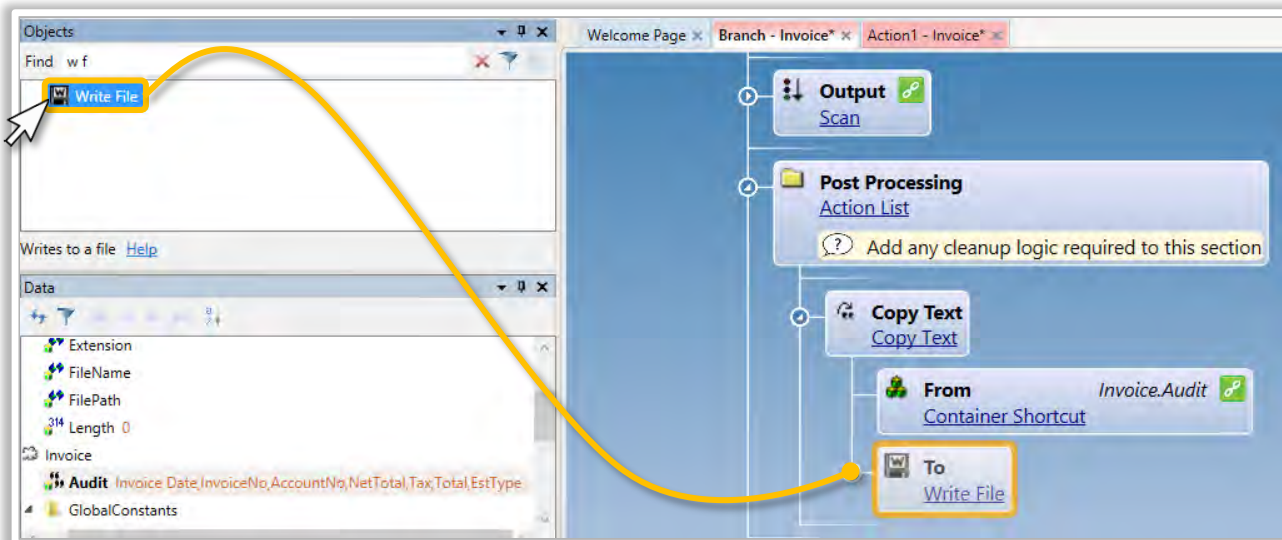
From the Data palette select the Audit item and drag it onto the branch over the From Text object.



Step 16 – Copy Text To

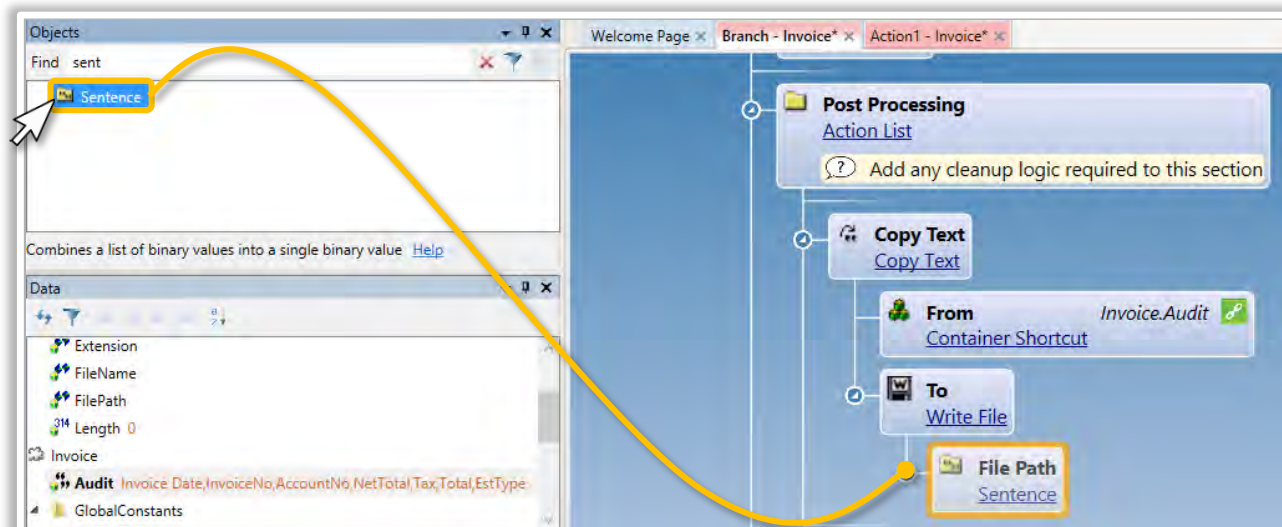
The audit file should be written the following directory **C:\Training\yyyy\MMMM\dd\Audit**

From the Objects palette Find field type **w f** select Write File drag it onto the branch over the To Text object



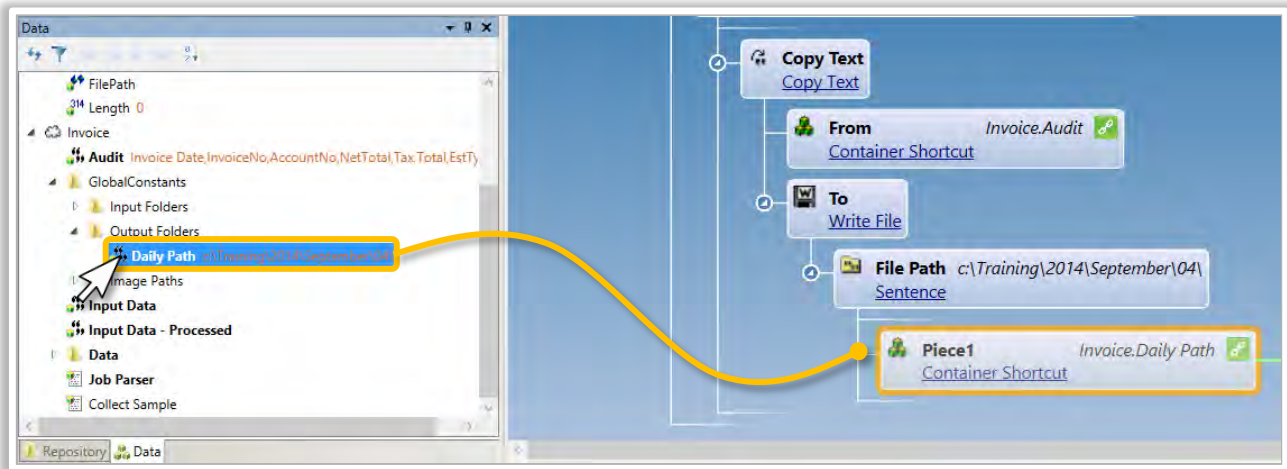
Step 16.1 - Sentence

Expand the **Write File** object and replace the **Text** object with a **Sentence** object



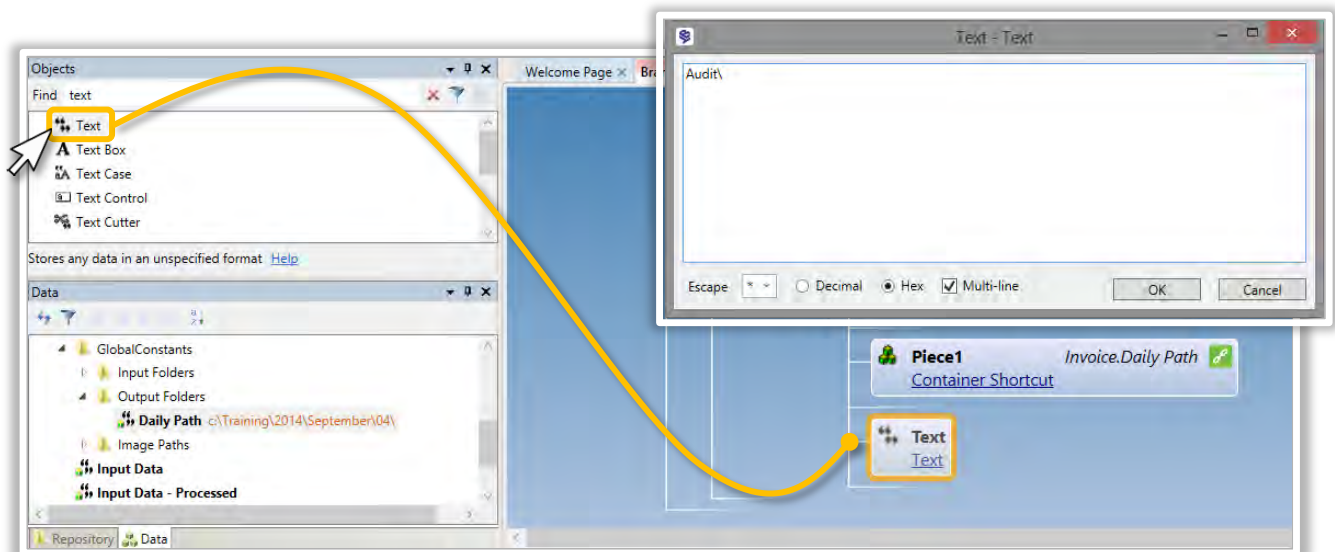
Step 16.2 – Daily Path Container Shortcut

Expand the Sentence object and replace Piece1 **Text** object with the Daily Path from the Data palette.



Step 16.3 – Complete Path

Add a Text object so it becomes the second piece of the sentence click  and enter Audit\



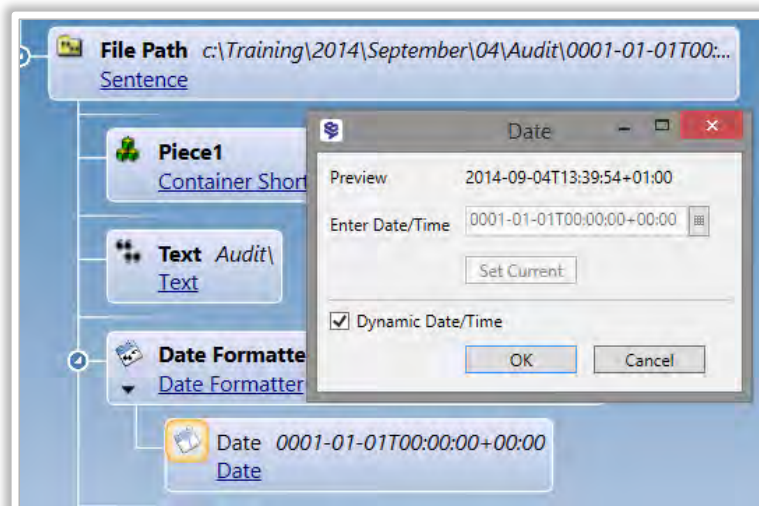
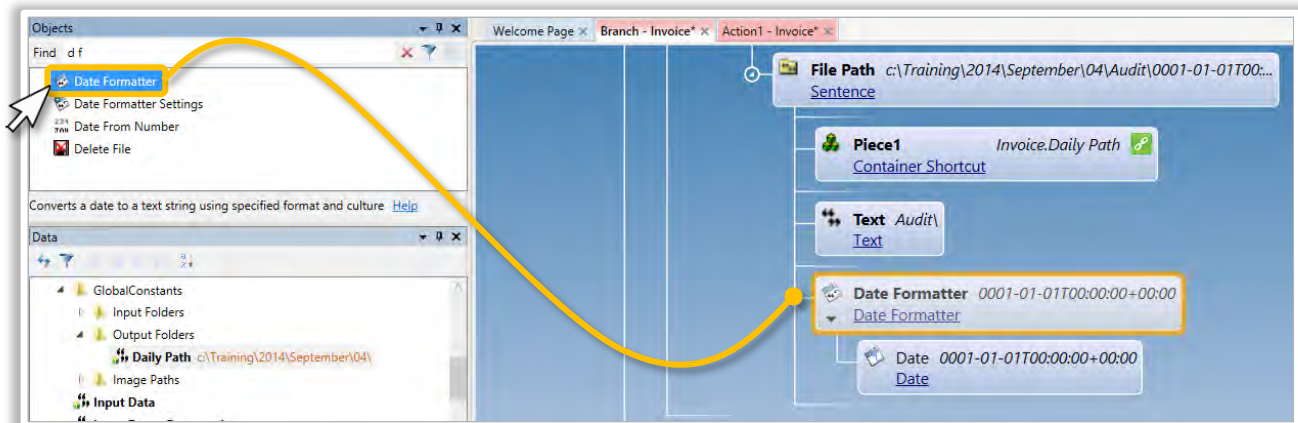
Step 16.4 – Filename Date Formatter

The Audit filename convention will be constructed as follows.

yyyyMMdd.csv

In the Objects palette Find field type **d f**, select **Date Formatter** and drag it onto the branch so it becomes the third piece of the **Sentence** object.

Expand the **Date Formatter** object to reveal the **Date** object



Step 16.5 – Date

Click **Date** object icon to open the Date dialog box.

Once open check the Dynamic Date/Time option.

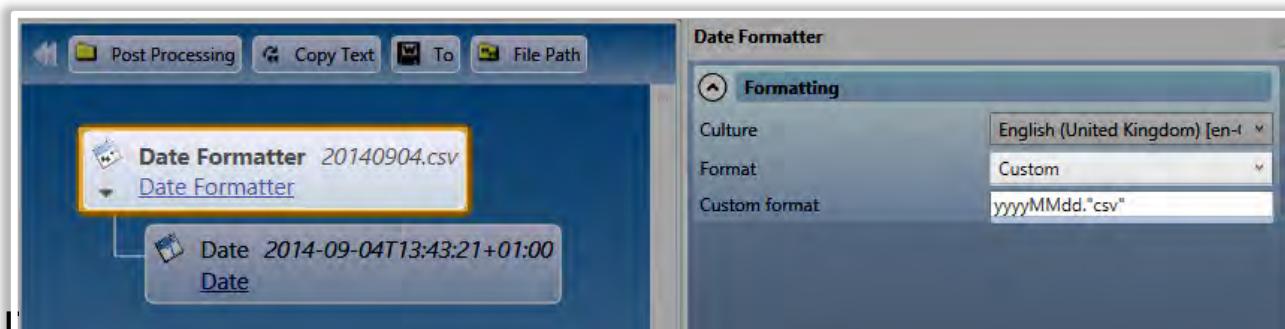
This returns the current system date and time.

Click **OK** to close

Step 16.6 – Date Formatting

Select the Date Formatter object and then change the Formatting properties as follows:

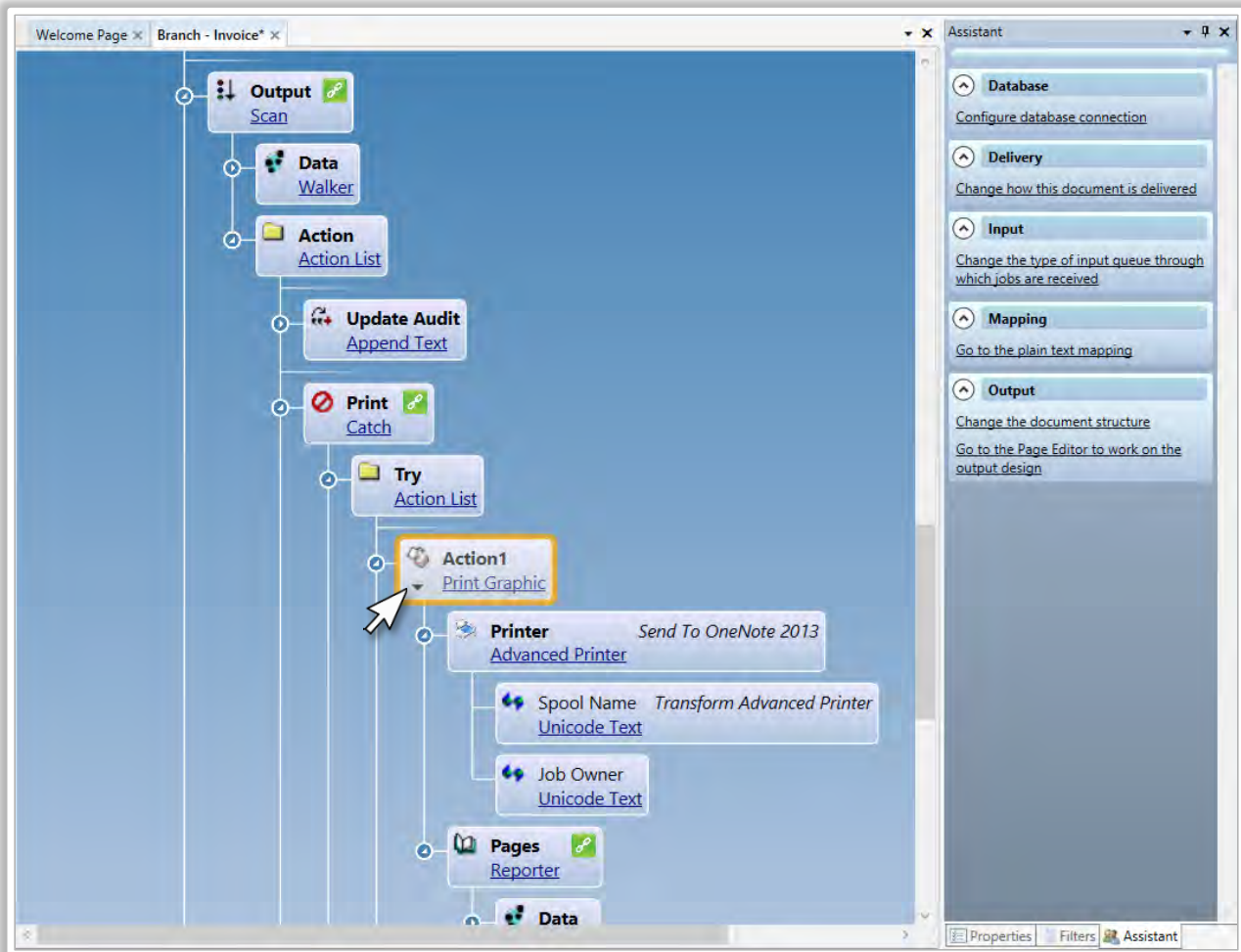
Format **Custom**
Custom format **yyyyMMdd."csv"**



The default output method for the Plain Text template is Print. To modify the output complete the following steps.

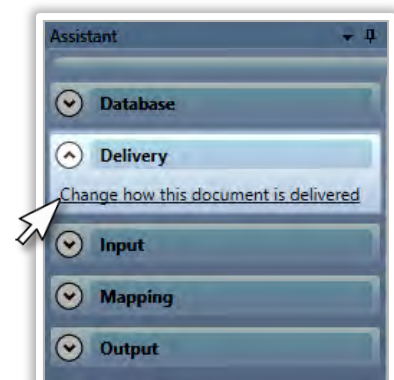
Step 1 – Assistant

From the branch select the Print Graphic object to change the content listed in the Assistant window.



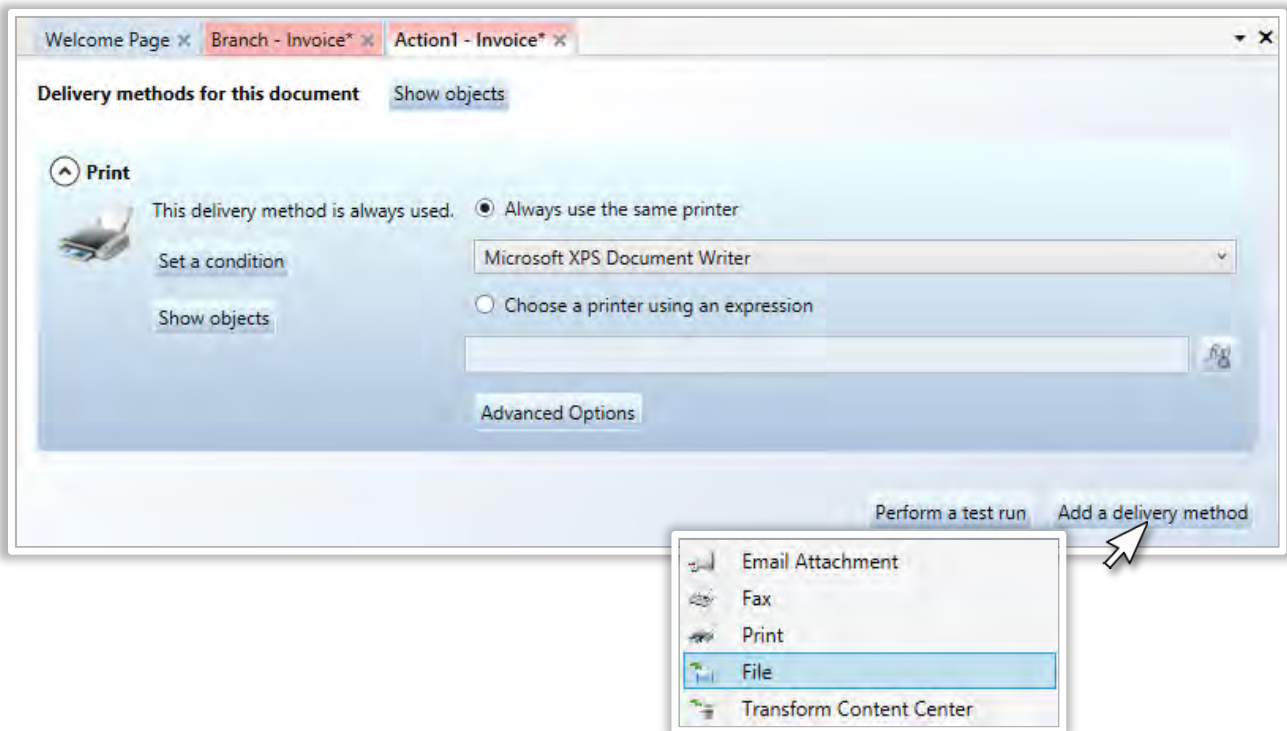
Step 2 – Assistant Delivery

From the Assistant window Delivery section click Change how this document is delivered



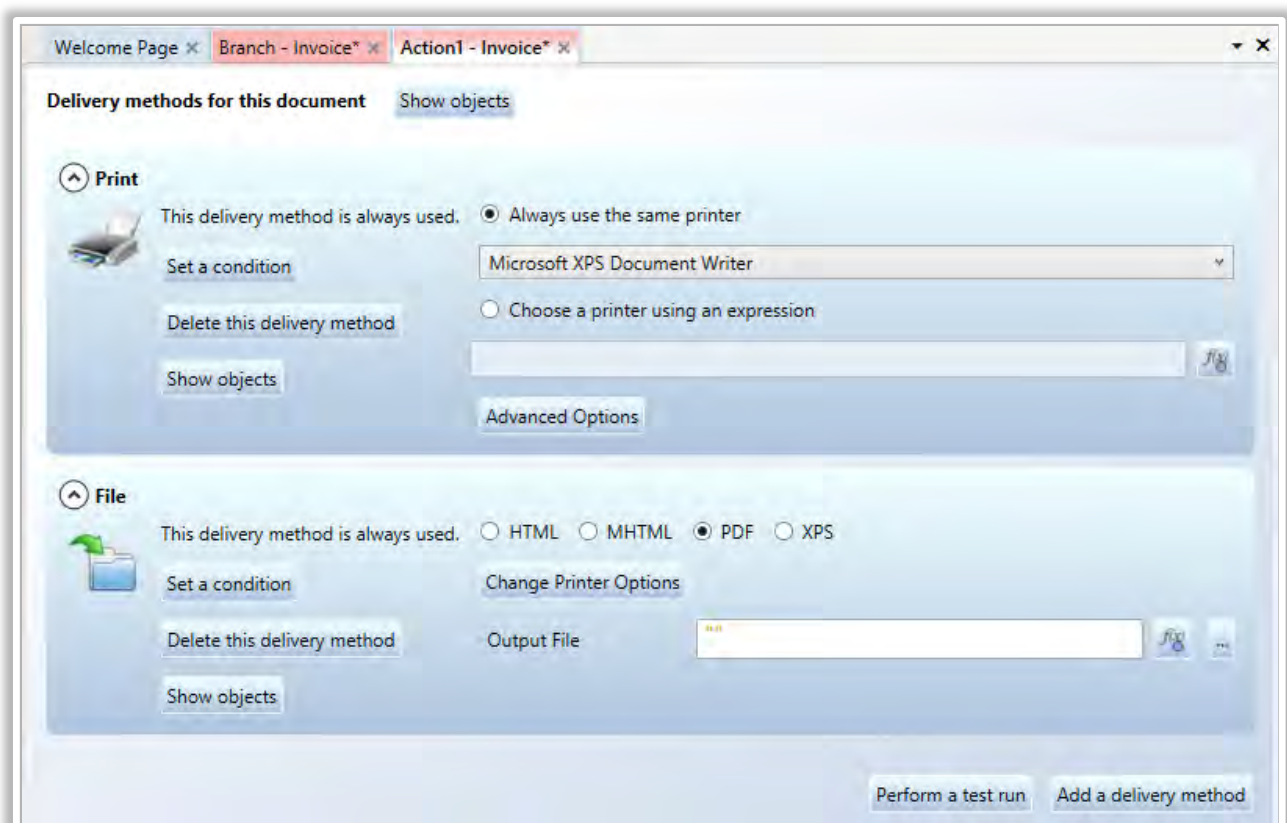
Step 3 – Add Delivery Method

The Delivery view open as a Tab in the file view window. Click Add Delivery Method and select File from the menu.




Step 4 – Set File Delivery Options

Select PDF as the file type.



Step 5 – Output File

Click  to open the File Path Expression editor to build the output filename.

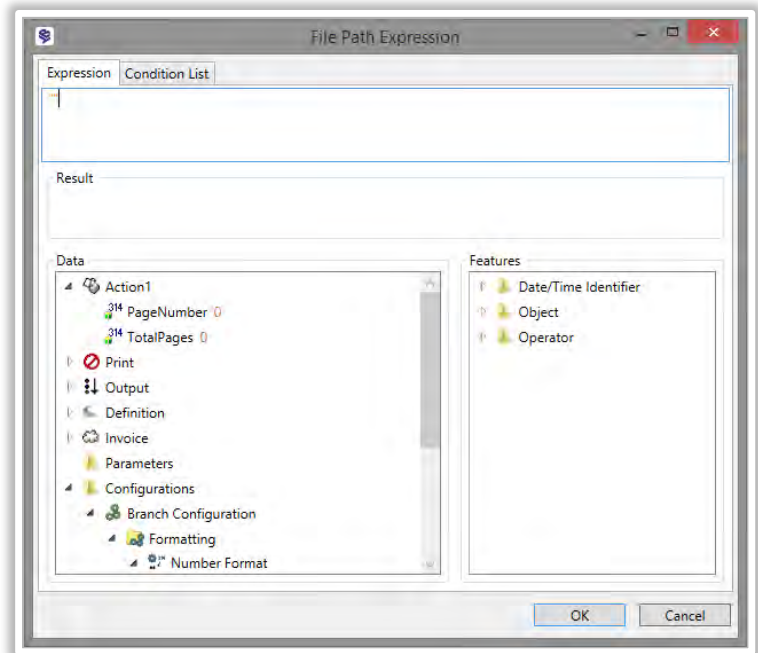
PDF File Path

C:\Training\YYYY\MMMM\dd\PDF

Filename

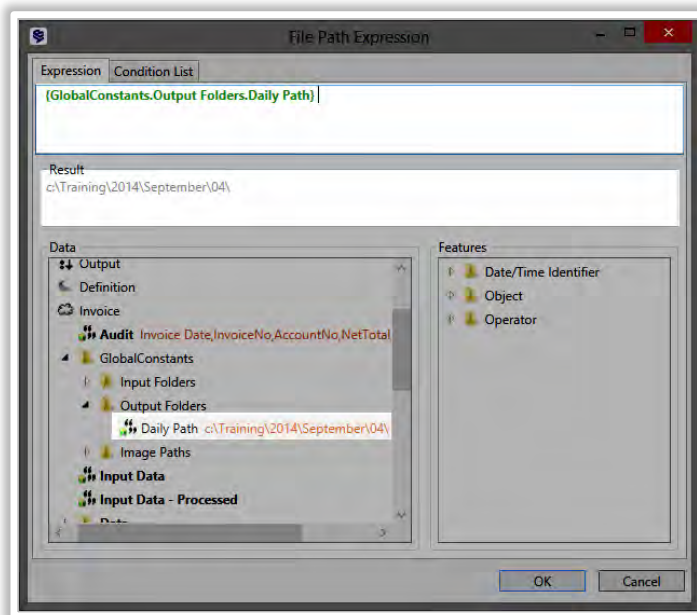
InvoiceNo.pdf

Remove the double quotes "" from the Expression window.




Complete the following steps to build the expression.

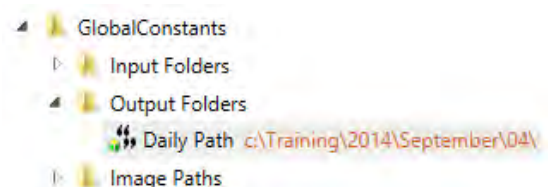
{GlobalConstants.Output Folders.Daily Path} & "PDF\" & {Value.Header.Header1.InvoiceNo} & ".pdf"



Step 5.1 – Output Path

Remove the double quotes "" from the Expression window.

From the Data window expand  > GlobalConstants > Output Folders

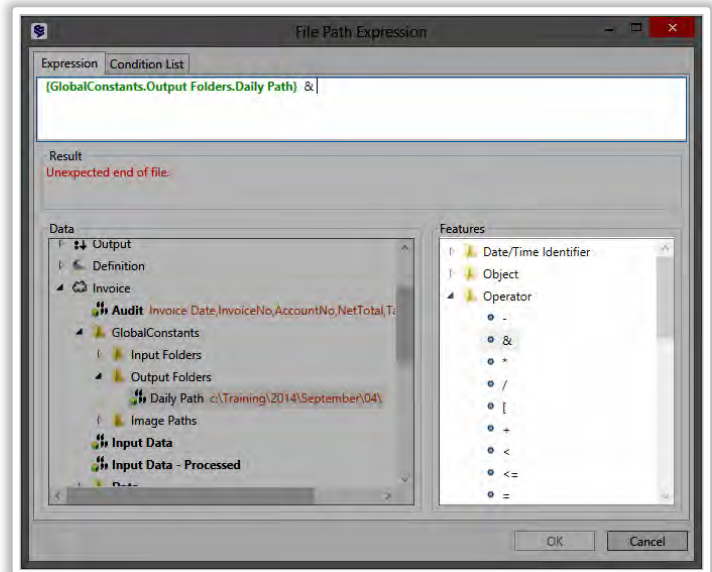


Move the cursor to the right of the expression.

{GlobalConstants.Output Folders.Daily Path} |

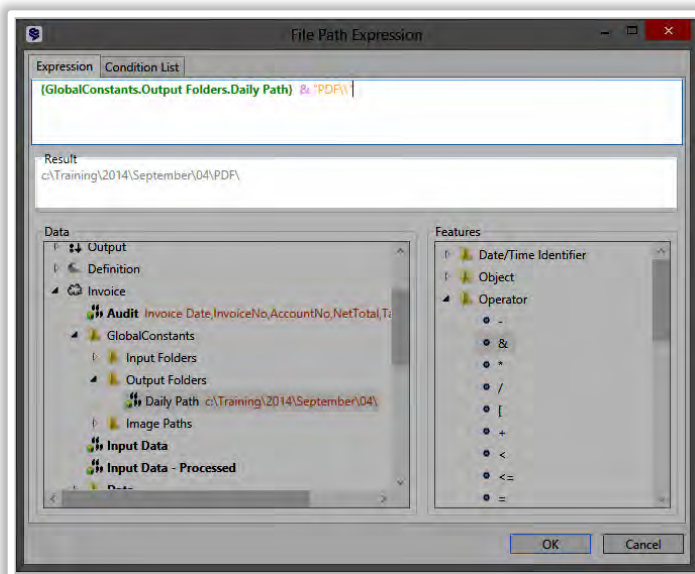
Step 5.2 – Operator

From Features Operator section select **&**



Move the cursor to the right of the expression.

{GlobalConstants.Output Folders.Daily Path} & |



Step 5.3 – Constant

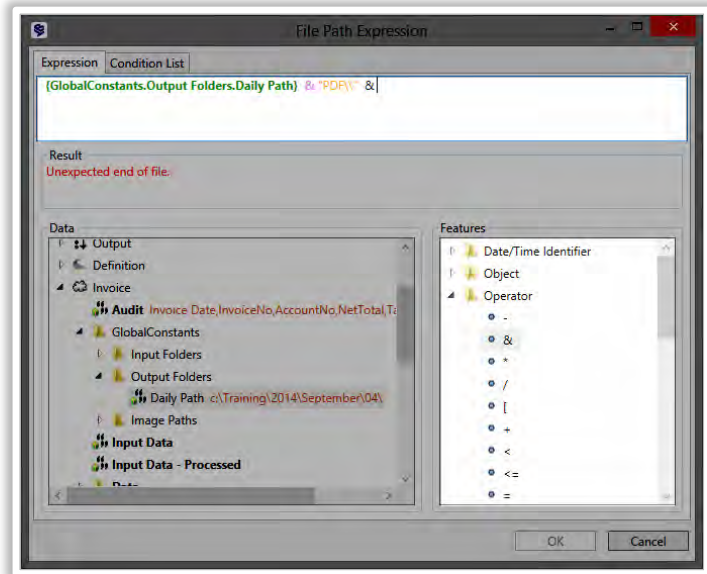
Type **"PDF\\"** to complete the output path.

Move the cursor to the right of the expression.

{GlobalConstants.Output Folders.Daily Path} & "PDF\\" |

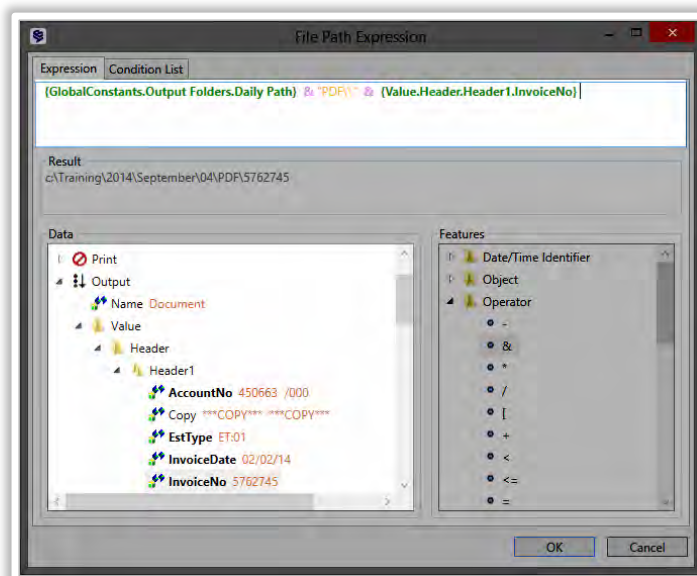
Step 5.4 – Operator

From Features Operator section select &



Move the cursor to the right of the expression.

{GlobalContants.Output Folders.Daily Path} & "PDF\\" & |

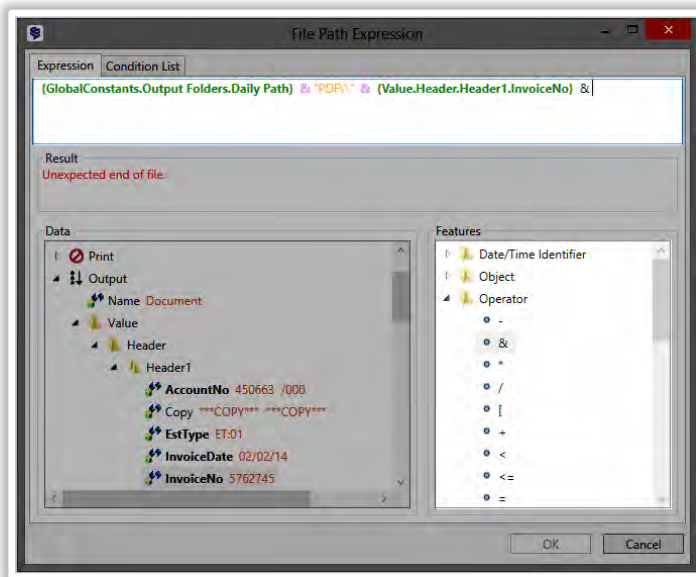


Step 5.5 – Filename

From Data expand Output > Value > Header > Header1 then select **InvoiceNo**

Move the cursor to the right of the expression.

{GlobalContants.Output Folders.Daily Path} & "PDF\\" & {Value.Header.Header1.InvoiceNo} |



Step 5.6 – Operator

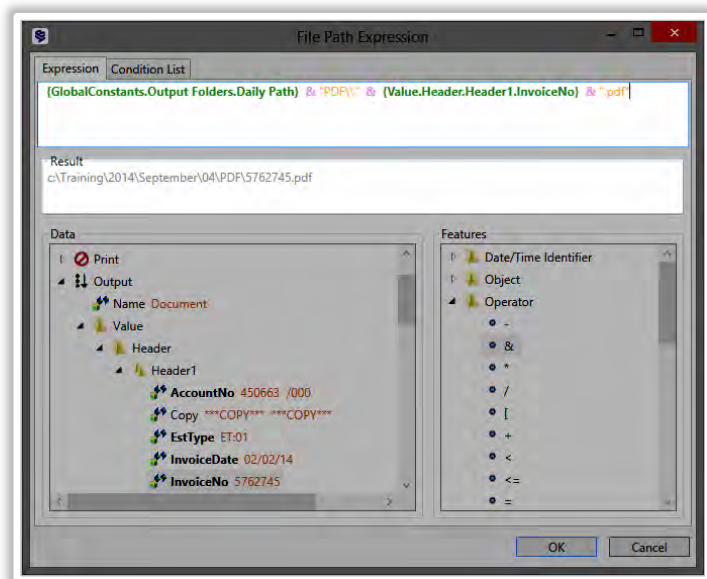
From Features Operator section select &

Move the cursor to the right of the expression.

{GlobalConstants.Output Folders.Daily Path} & "PDF\" & {Value.Header.Header1.InvoiceNo} & |

Step 5.7 – File Extension

Add the file extension as a constant type
".pdf"



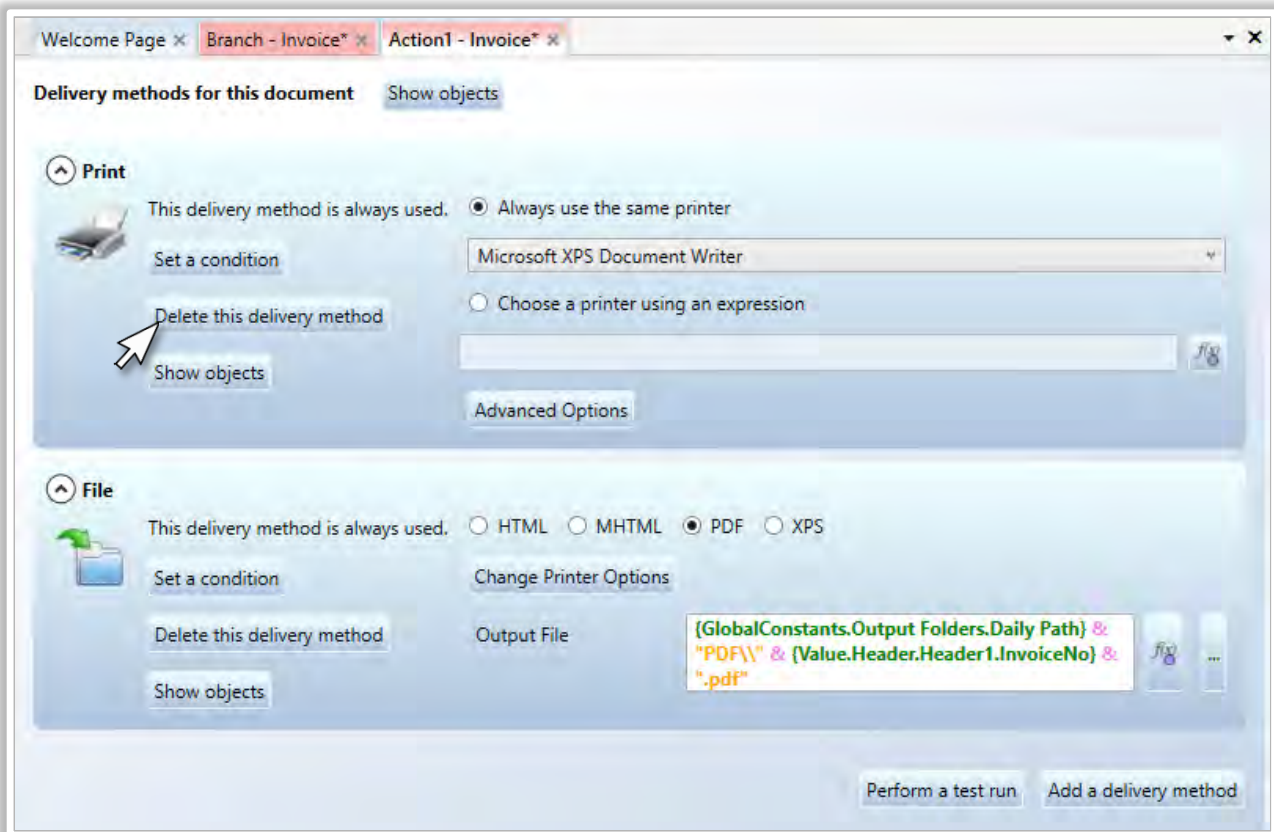
Move the cursor to the right of the expression.

{GlobalConstants.Output Folders.Daily Path} & "PDF\" & {Value.Header.Header1.InvoiceNo} & ".pdf"

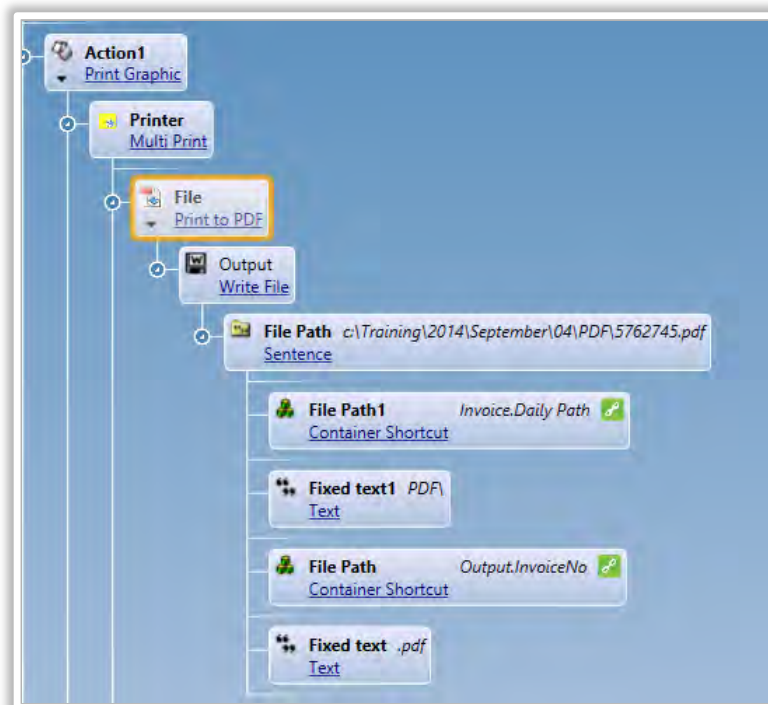
Click **OK** to close the File Path Expression editor.

Step 6 – Delete Print Delivery Method

To remove a delivery method from the File view click **Delete this delivery method**.



The Assistant has modified the branch, to view the changes click Show objects.



Alternative Technique

You can add the objects required to produce the required delivery method directly on the branch by adding the following objects.

Print to PDF

Write File

Sentence

Container Shortcut
Text

Project Checklist

1. ~~Repository Design~~
2. ~~Create Local Repository~~
3. ~~Language Labels~~
4. ~~Global Resources~~
5. ~~Plain Text Template Wizard~~
6. ~~Plain Text to Container~~
7. ~~Modify Branch Structure~~
8. **Dynamic Branch Shortcut**
 - Container Memory Item**
 - Create Dynamic Branch Shortcut**
 - Parameters**
 - Test Code**
9. Document Organizer
10. Synchronize and Deploy



Dynamic Branch Shortcut

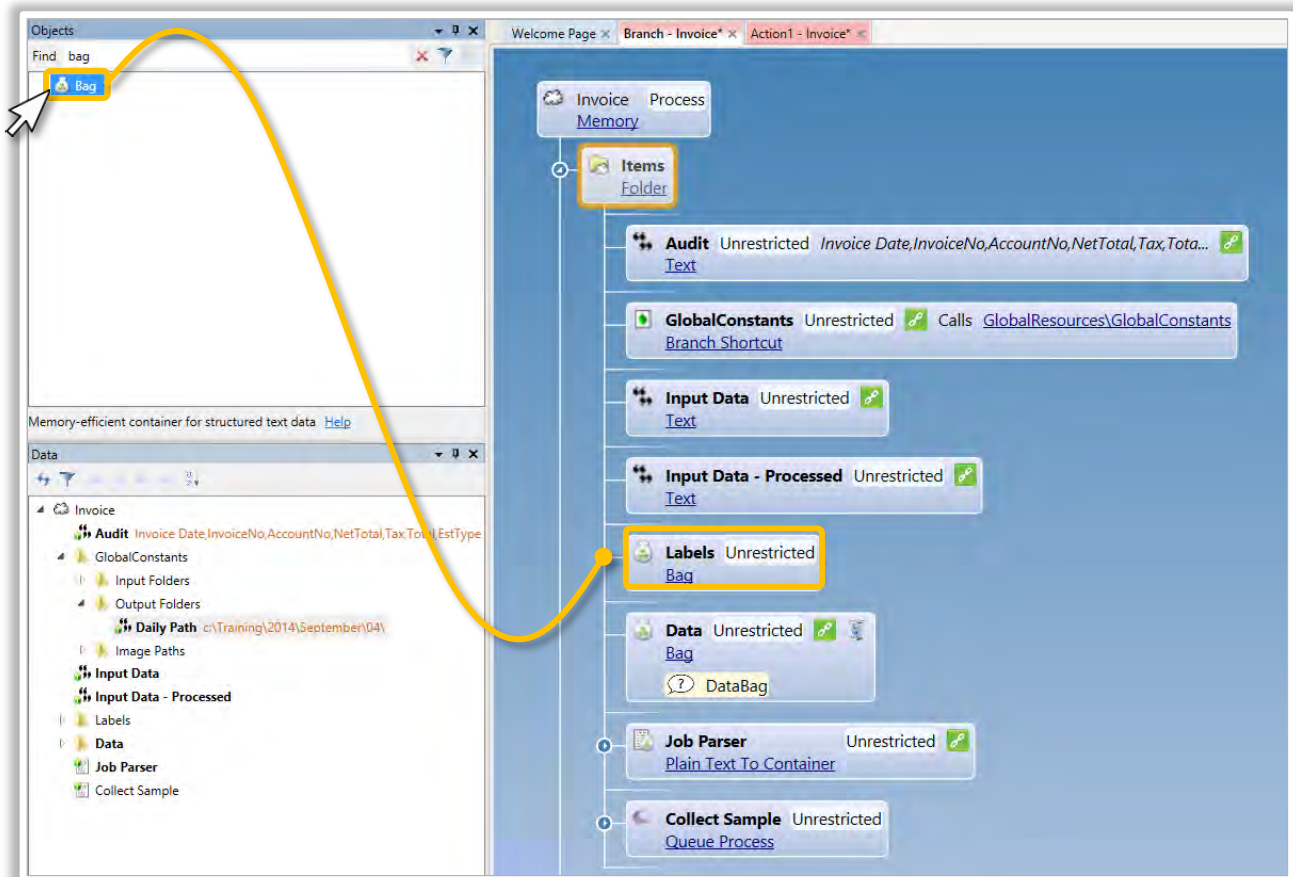
As the Scan object steps through each record we are going to use a dynamic branch shortcut to call the corresponding target branch to create the language label container.

Complete the following steps to integrate the dynamic branch shortcut.

Step 1 – Container Memory Item

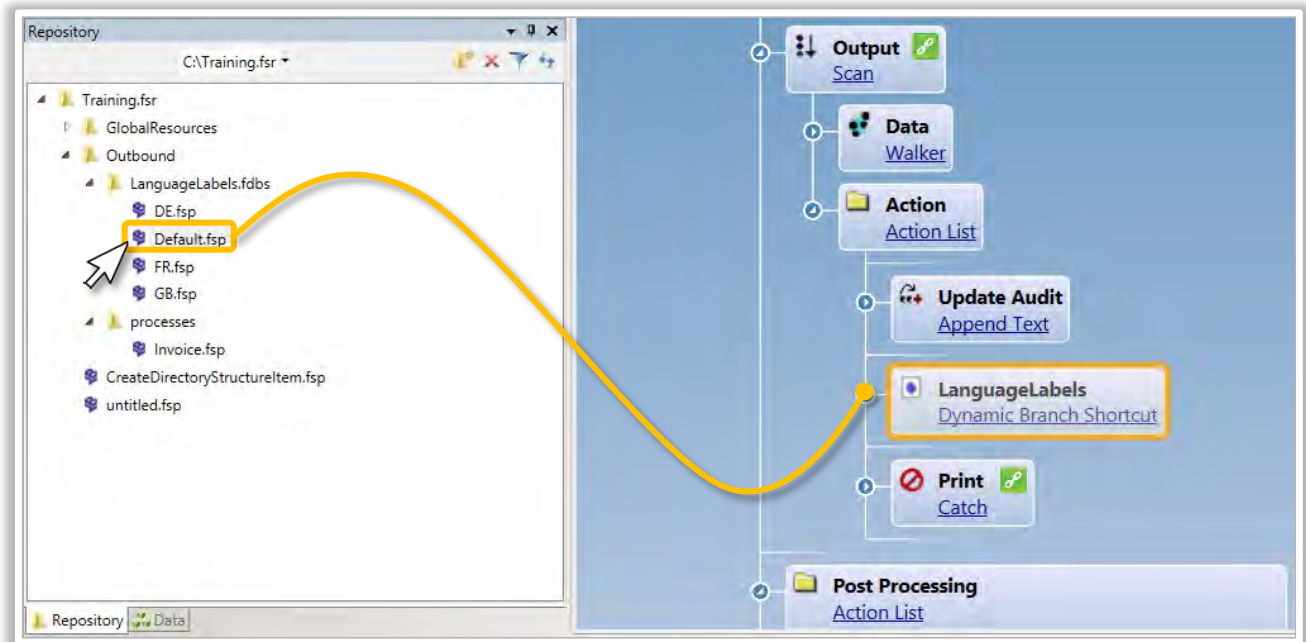
The language label branches make use of a parameter to pass the container. We need the container values to be visible within the Data palette so that they can be used on the output.

From the Objects palette select **Bag** and drag it onto the branch within the Items Folder. Rename the Bag to **Labels**.



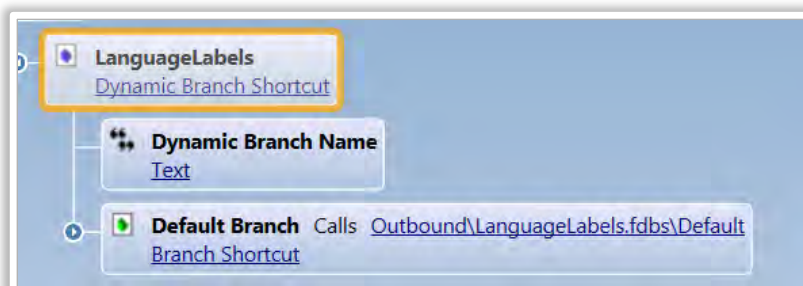
Step 2 – Create Dynamic Branch Shortcut

On the branch editor locate the Output **Scan** object and expand, from the Repository Browser expand Outbound > LanguageLabels.fdb's select Default.fsp and drag onto the branch beneath the Action List, ensuring it is positioned before the Print **Catch** object.



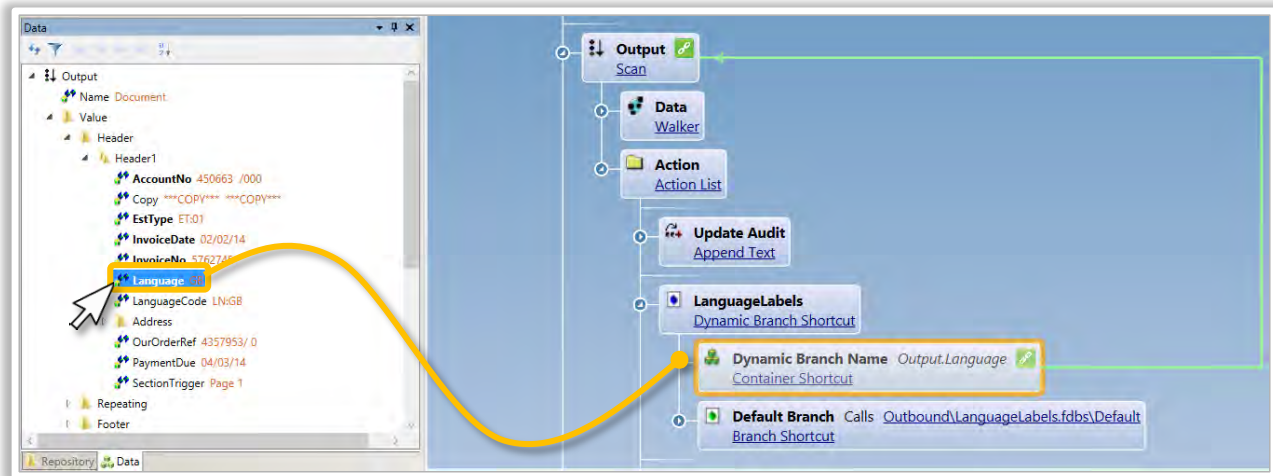
Expand the **Dynamic Branch Shortcut** object to reveal the two child objects.

- **Dynamic Branch Name** – Use to pass LanguageCode from the header of each record.
- **Default Branch** – The default branch to use if the Dynamic Branch Name is not found in the LanguageLabels.fdb's directory.



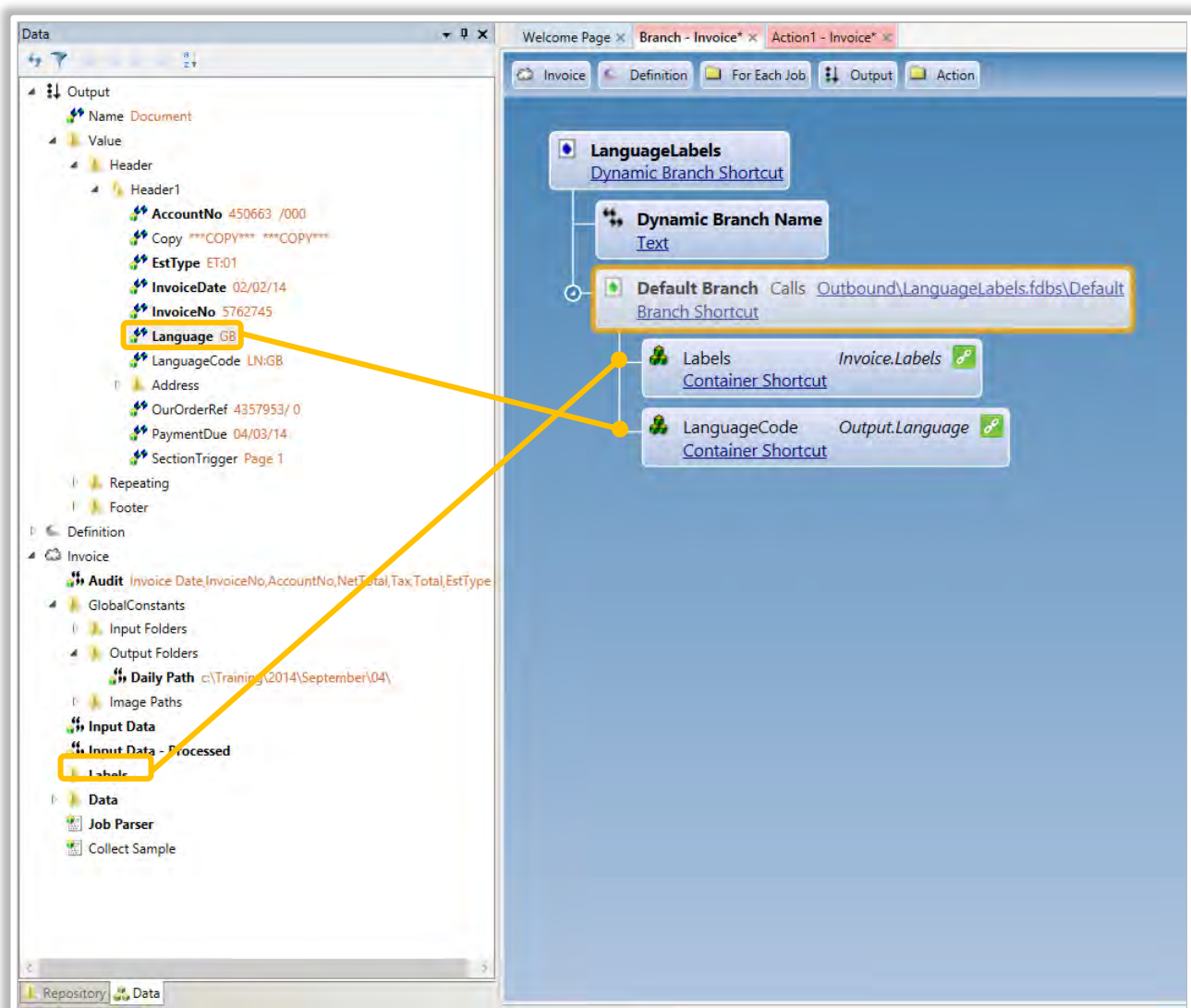
Step 3 – Dynamic Branch Name Container Shortcut

From the Data palette expand Output Scan > Value > Header > Header1 then select the Language element and replace the Dynamic Branch Name **Text** object to create Container Shortcut.




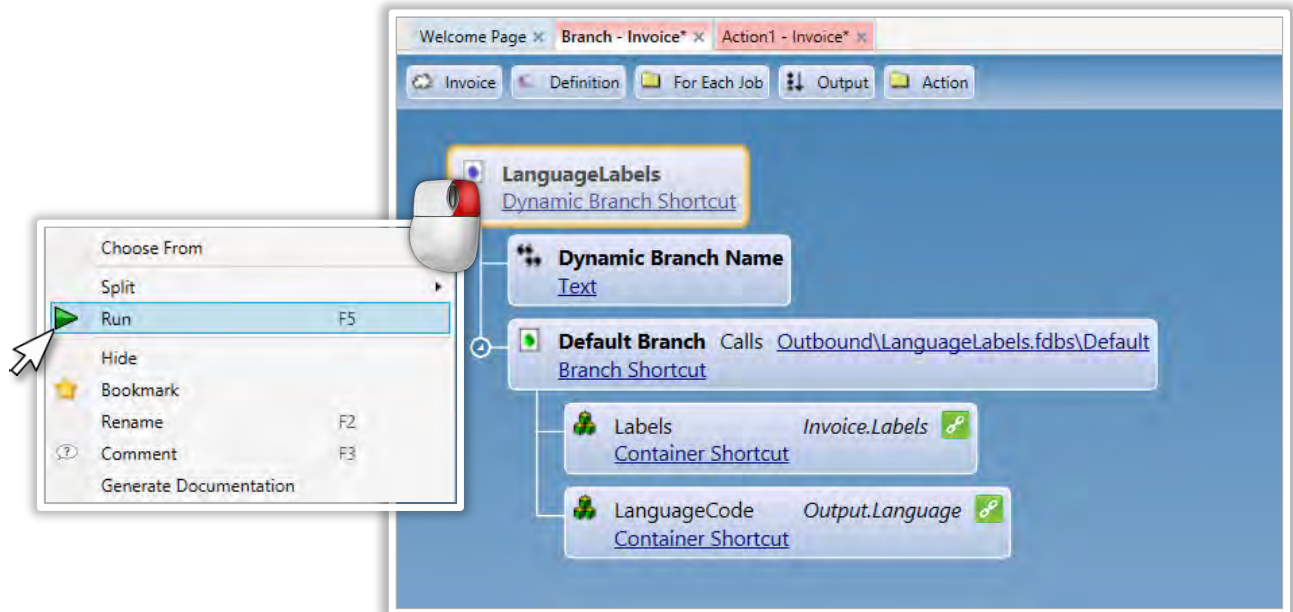
Step 4 – Parameters

Expand the Default Branch Shortcut to reveal the branch parameters. From the Data palette drag Language onto LanguageCode and Labels within the memory items onto Labels.




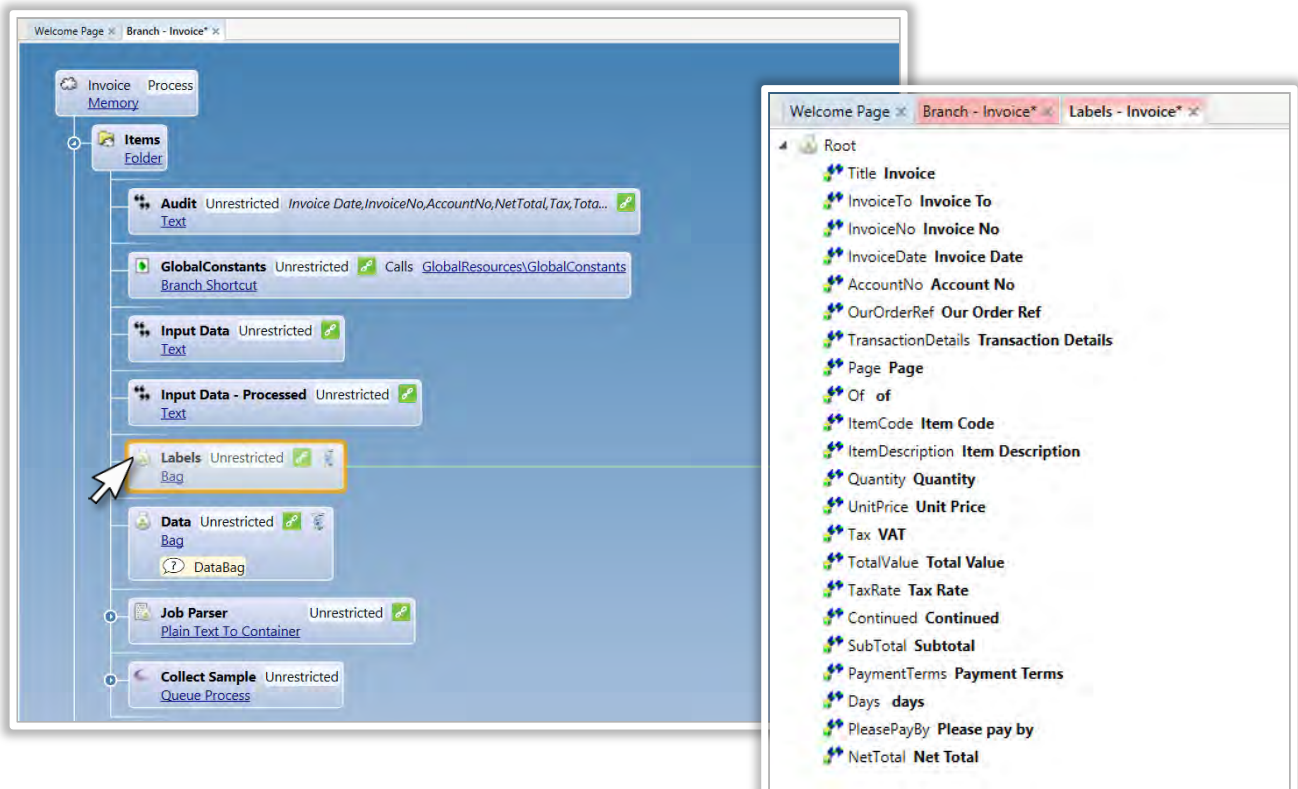
Step 5 – Test your Code

To see if the correct language labels are being returned run the Dynamic Branch Shortcut. Right-click the Dynamic Branch Shortcut object and click  Run from the menu. This will populate the Labels container in memory.



This will call the corresponding language label and populate the Label container bag in memory

You can inspect the Label value from the Data palette alternatively open the container on the branch, click  icon on the Label **Bag** object. This will open a tab on the file view window.





Tip

If an Empty Container icon is displayed this indicates the Bag contains data. To empty the container click the recycle bin.



Project Checklist

1. ~~Repository Design~~
2. ~~Create Local Repository~~
3. ~~Language Labels~~
4. ~~Global Resources~~
5. ~~Plain Text Template Wizard~~
6. ~~Plain Text to Container~~
7. ~~Modify Branch Structure~~
8. ~~Dynamic Branch Shortcut~~
9. **Document Organizer**
 - Move Document Organizer on Branch**
 - Repeated Section Container Shortcut**
 - Add Page Types**
 - Define Page Settings**
 - Repeating Detail Configuration**
 - Map Data onto Sortable Table Row**
 - Create Background Forms**
 - Gradient Fill**
 - Map Language Labels**
 - Build Text Expression**
 - Map Header Data**
 - Page N of M**
 - Dynamic Flag Image**
 - QR Barcodes**
 - Map Footer Data**
 - Date Difference**
 - Dynamic Marketing Image**

10. ~~Synchronize and Deploy~~

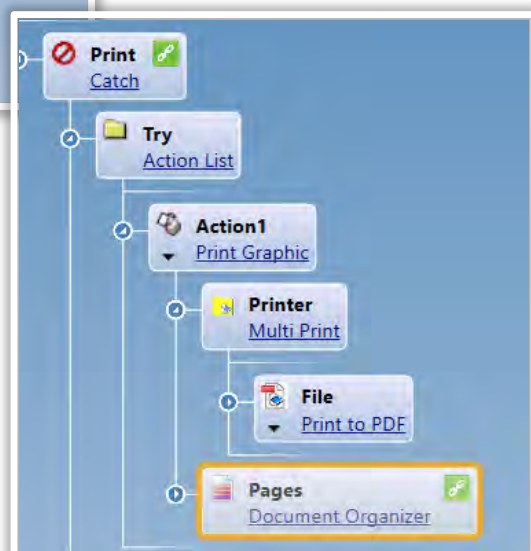


Step 1 – Move Document Organizer

By default the Plain Text template places the Document Organizer beneath a Reporter object, which will step through the records in the data set. As we have already changed the structure of the branch by adding a Scan object that also steps through the records we do not need the Reporter object.



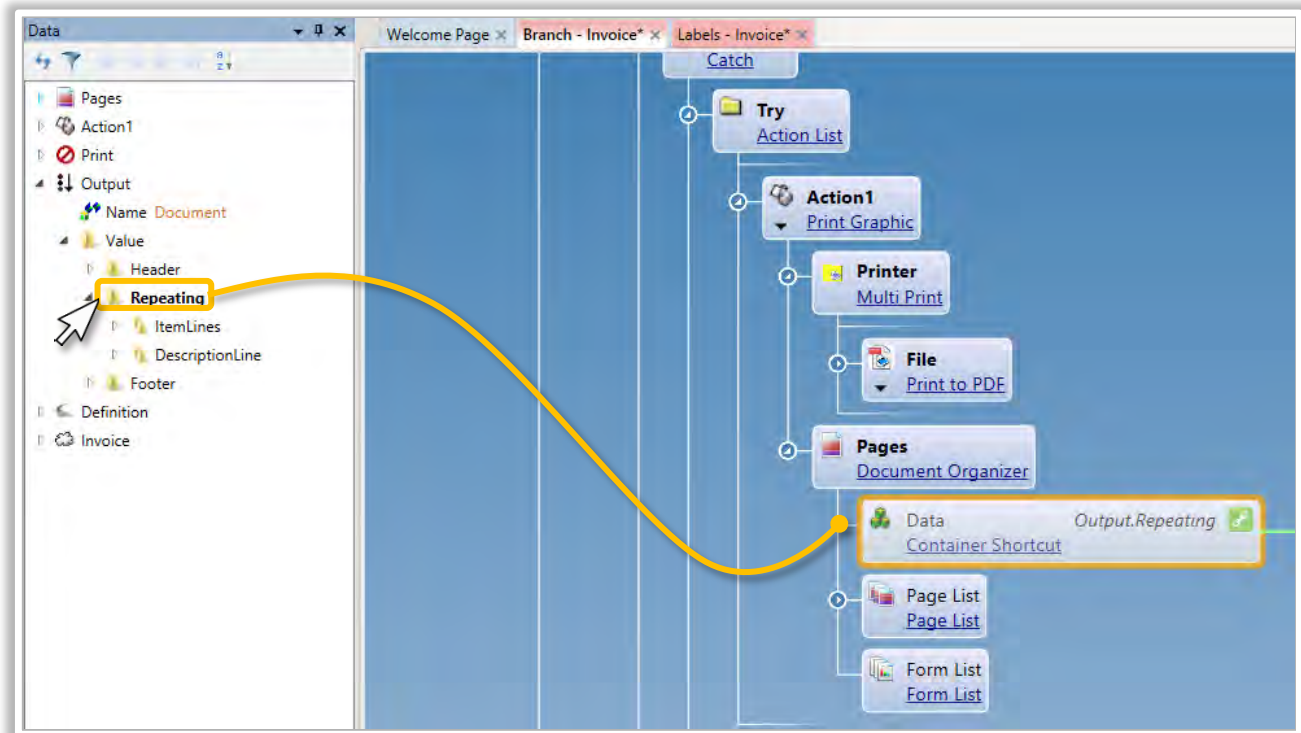
Right-click and drag the Document Organizer over the Reporter object.



Step 2 – Repeated Section Container Shortcut

The Document Organizer paginates the repeating elements. Since we moved the Document Organizer over the Reporter we have broken the container shortcut to the repeating elements.


Expand the Document Organizer then from the Data palette expand Output > Value and drag Repeating onto the branch over the current Container Shortcut.

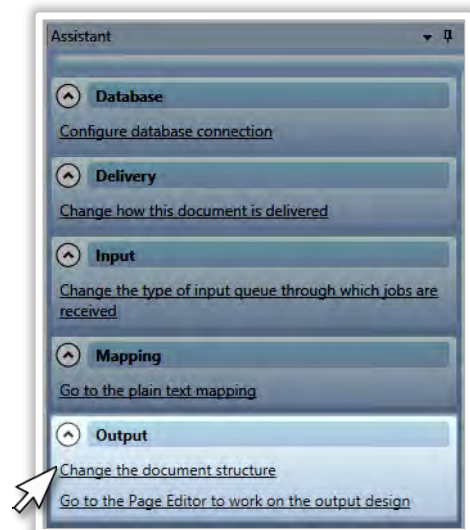
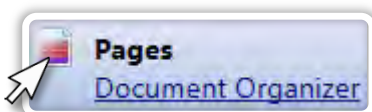


Step 3 – Change Document Structure

From the Assistant Output section select

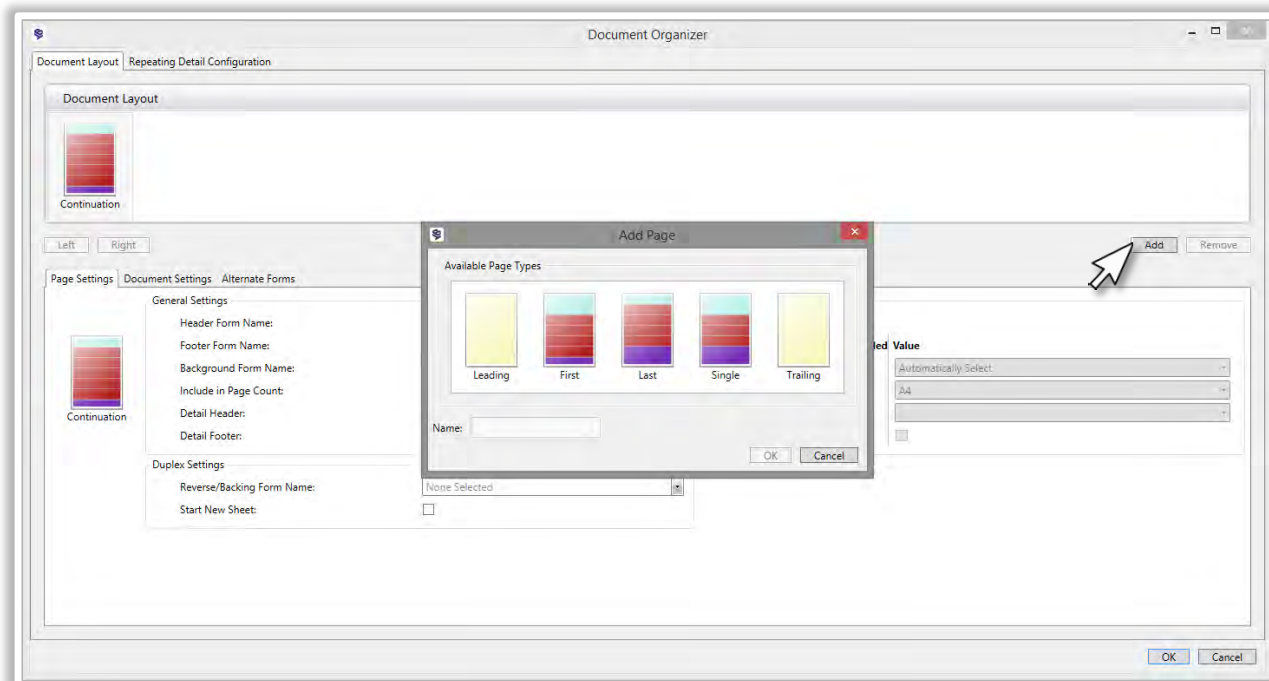
Change the document structure

Alternatively click the  icon.



Step 4 – Add Page Types

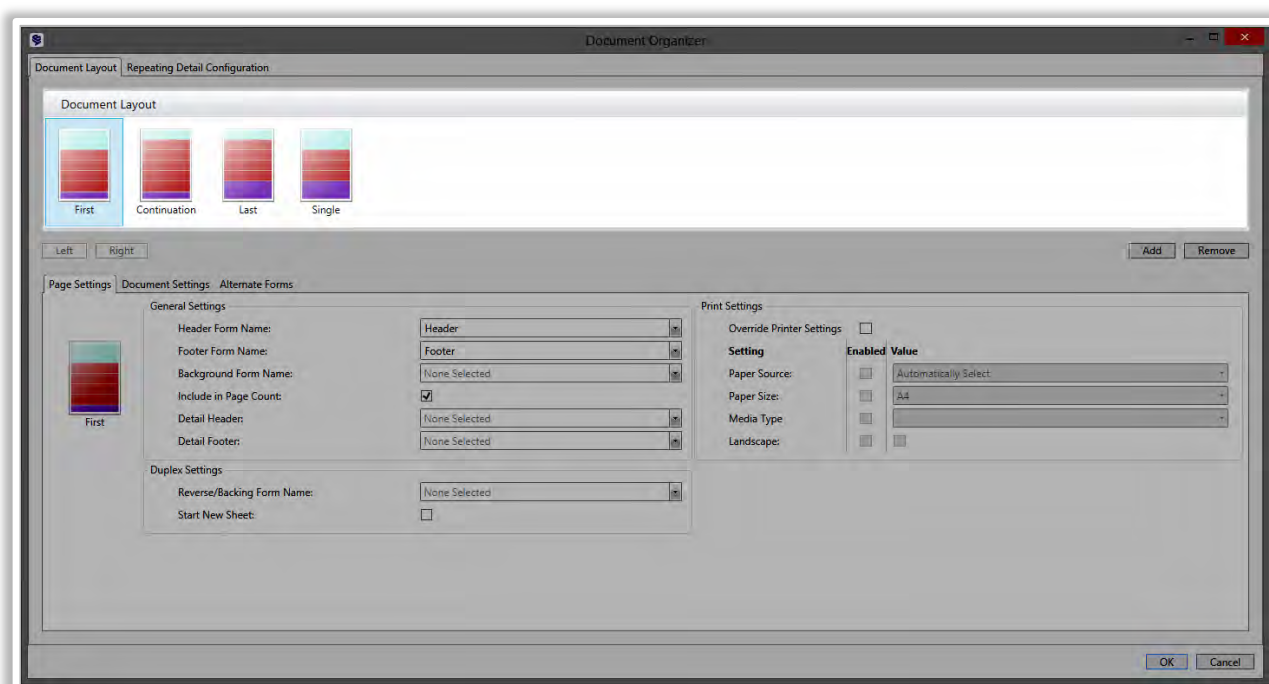
Within the Document Organizer you can change the structure of the document. Once open click Add that opens the Add Page dialog box



Select the following Pages

- First
- Continuation
- Last
- Single

As you add each Page Type the dialog box closes so you need to click the Add button to add the additional Pages.



Step 5 – Define Page Settings

Each Page Type can be defined with different page setting options.

For our project we need to change the following General Settings



Note

The drop down lists update dynamically, as such if a Form Name does not exist in the list type the name in the field to add it to the list.

First Page

Header Form Name:

Header

Footer Form Name:

SmallFooter (type this as it does not exist in the list)

Background Form Name:

FirstPage (type this as it does not exist in the list)

Detail Header:

TableHeader

Header

FirstPage

Item Code	Item Description	Quantity	Unit Price	Total Price	Tax
070001	CONDUCTIVITY GEL (2000000012)	5	8.00	40.00	A
070002	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070003	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070004	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070005	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070006	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070007	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070008	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070009	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070010	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070011	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070012	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070013	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070014	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070015	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070016	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070017	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070018	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070019	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070020	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070021	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070022	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070023	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070024	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070025	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070026	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070027	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070028	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070029	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070030	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070031	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070032	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070033	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070034	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070035	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070036	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070037	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070038	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070039	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070040	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070041	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070042	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070043	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070044	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070045	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070046	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070047	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070048	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070049	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070050	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070051	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070052	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070053	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070054	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070055	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070056	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070057	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070058	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070059	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070060	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070061	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070062	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070063	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070064	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070065	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070066	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070067	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070068	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070069	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070070	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070071	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070072	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070073	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070074	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070075	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070076	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070077	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070078	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070079	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070080	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070081	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070082	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070083	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070084	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070085	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070086	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070087	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070088	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070089	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070090	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070091	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070092	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070093	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070094	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070095	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070096	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070097	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070098	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070099	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A
070100	CONDUCTIVITY PENCIL ERASERS (4000000012)	5	0.00	0.00	A

TableHeader

SmallFooter

Page Settings | Document Settings | Alternate Forms

General Settings

Header Form Name:

Footer Form Name:

Background Form Name:

Include in Page Count: ☒

Detail Header:

Detail Footer:

Duplex Settings

Reverse/Backing Form Name:

Start New Sheet: ☐

Continuation Page

Header Form Name: **SmallHeader** (type this as it does not exist in the list)

Footer Form Name: **SmallFooter**

Background Form Name: **ContinuationPage** (type this as it does not exist in the list)

Detail Header: **TableHeader**

The diagram illustrates the layout of a Continuation Page. It features a central image of a document page with four callout boxes pointing to specific sections:

- SmallHeader**: Points to the top header section of the document.
- ContinuationPage**: Points to the main body of the document, which contains a table of transaction details.
- TableHeader**: Points to the header row of the table within the main body.
- SmallFooter**: Points to the bottom footer section of the document.

Page Settings | Document Settings | Alternate Forms

Continuation

General Settings

Header Form Name:

Footer Form Name:

Background Form Name:

Include in Page Count: ☒

Detail Header:

Detail Footer:

Duplex Settings

Reverse/Backing Form Name:

Start New Sheet: ☐

Last Page

Header Form Name:

SmallHeader

Footer Form Name:

Footer

Background Form Name:

LastPage (type this as it does not exist in the list)

Detail Header:

TableHeader

The screenshot shows a sample invoice from Bottomline Technologies. Callouts point to specific sections of the invoice:

- SmallHeader**: Points to the top header section containing the company logo and contact information.
- TableHeader**: Points to the header row of the main table, which lists items, quantities, and prices.
- LastPage**: Points to the bottom section of the invoice, including the payment terms and total amount.
- Footer**: Points to the very bottom of the page, which includes the company logo and a slogan.

Page Settings | Document Settings | Alternate Forms

General Settings

Header Form Name:

Footer Form Name:

Background Form Name:

Include in Page Count: ☒

Detail Header:

Detail Footer:

Duplex Settings

Reverse/Backing Form Name:

Start New Sheet: ☐

Single Page

Header Form Name:

Header

Footer Form Name:

Footer

Background Form Name:

SinglePage (type this as it does not exist in the list)

Detail Header:

TableHeader

The diagram illustrates the mapping of form names to sections of an invoice. The invoice is divided into four sections: Header, TableHeader, SinglePage, and Footer. Arrows point from these sections to their respective form names in a list on the right.

Section	Form Name
Header	Header
TableHeader	TableHeader
SinglePage	SinglePage
Footer	Footer

Page Settings | Document Settings | Alternate Forms

General Settings

Header Form Name:

Footer Form Name:

Background Form Name:

Include in Page Count: ☒

Detail Header:

Detail Footer:

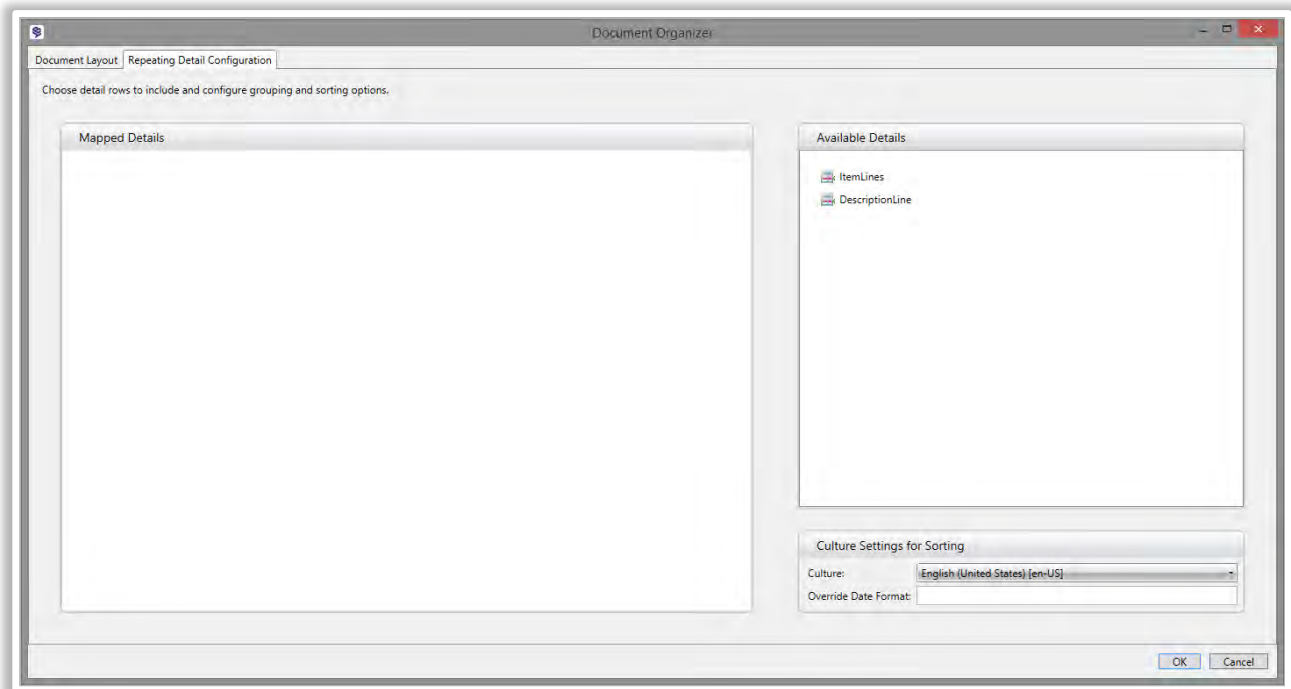
Duplex Settings

Reverse/Backing Form Name:

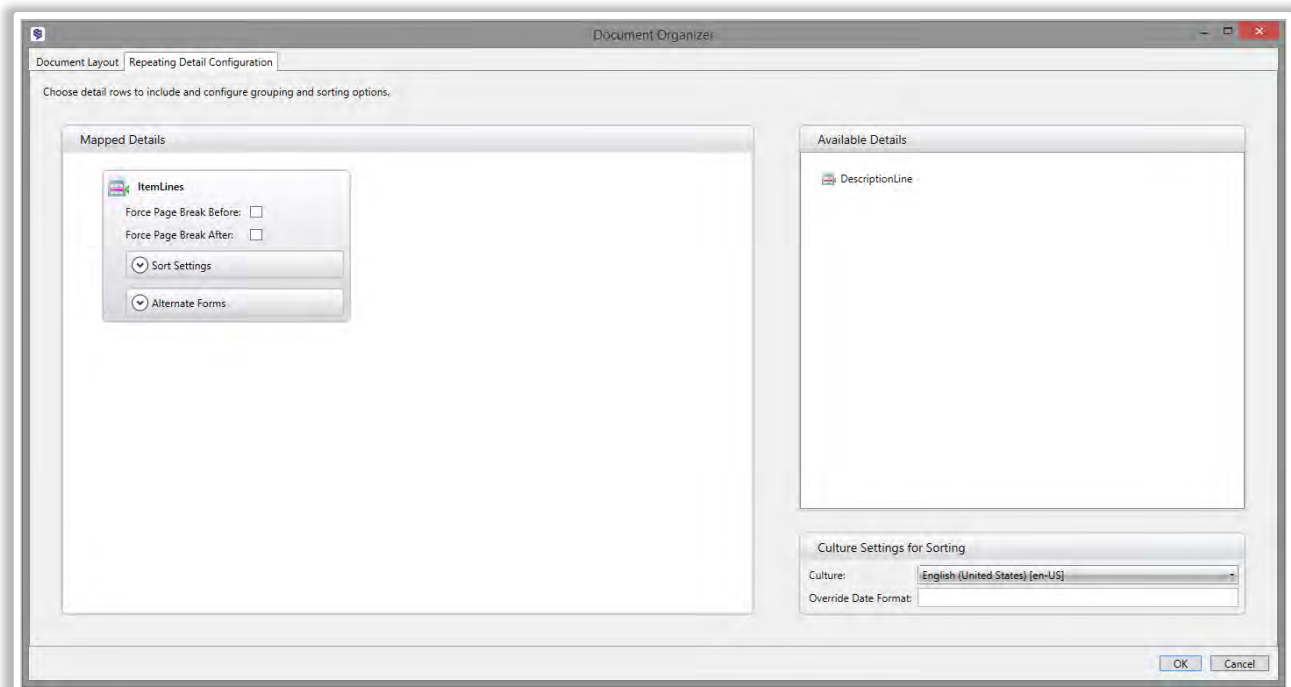
Start New Sheet: ☐

Step 6 – Repeating Detail Configuration

The Repeating Detail Configuration tab allows you to define the repeated elements in the container. When selected it presents the Mapped Details (Sortable Table Rows) on the left pane and the Available Details on the right pane.



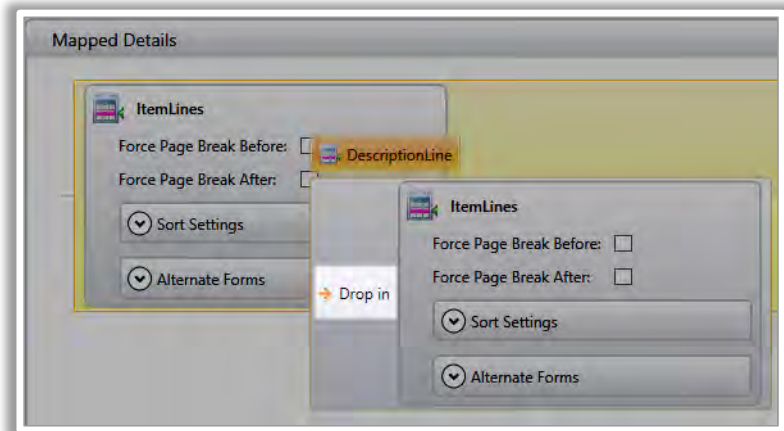
Drag the **ItemLines** from the Available Details pane into the Mapped Details



Create Group

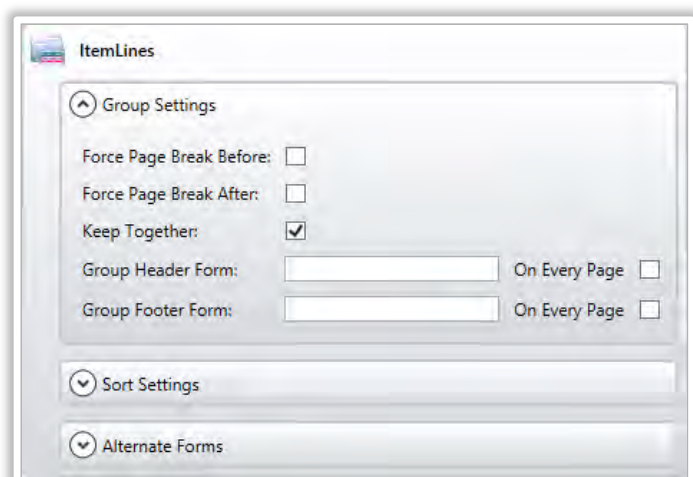
As the Document Organizer will repaginate the repeated elements we need to take care to group the **ItemLines** with the **DescriptionLines** and select keep together so that they never split across a page.

Drag the **DescriptionLines** from the Available Details pane onto the **ItemLines** within the Mapped Details pane taking care the position indicator displays **DropIn**



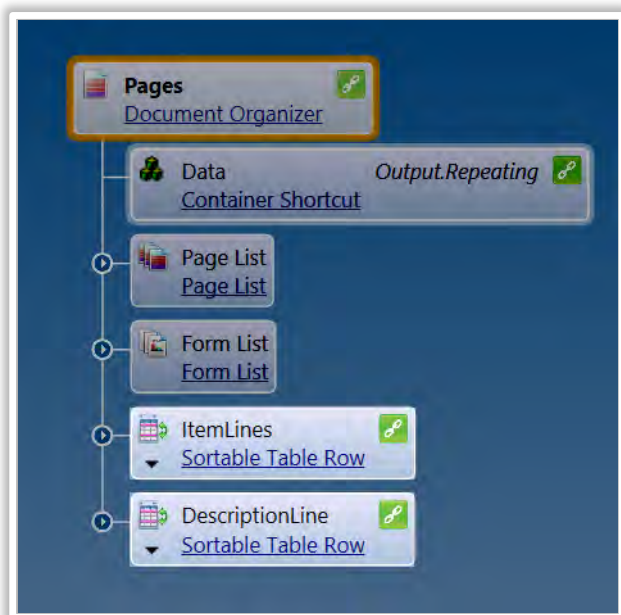
Once grouped the parent has an expand control to reveal the child details, also Group Settings become available.

Make sure Keep Together is checked. If not then the group will split across a page. Click **OK** to commit changes.



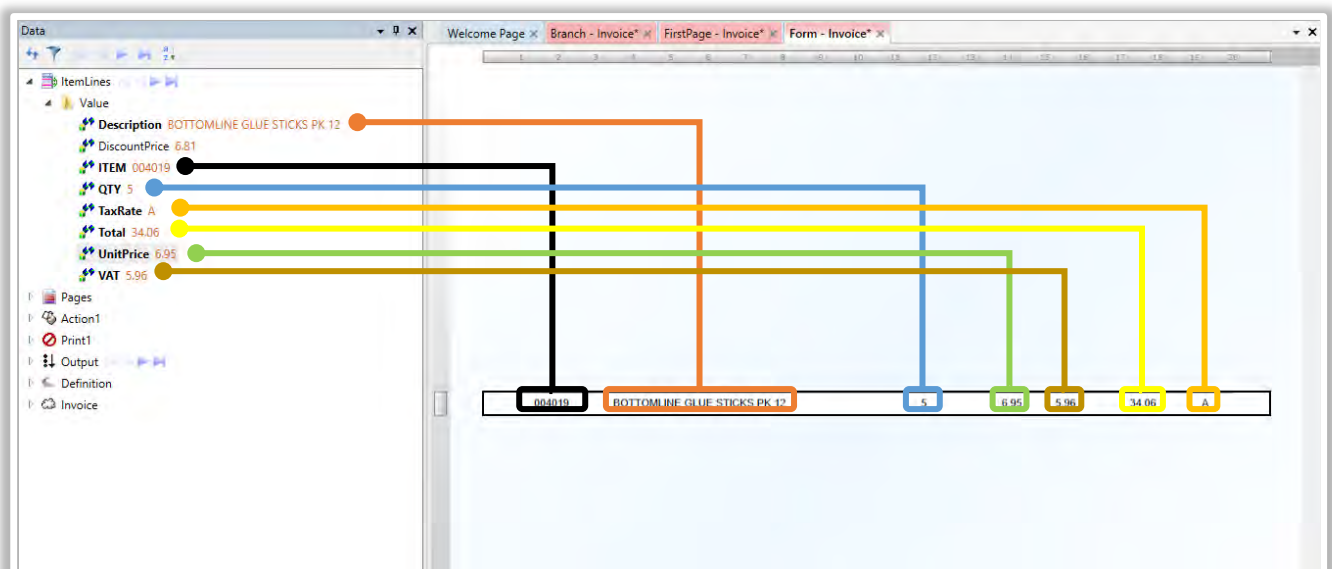
Step 7 - Map Data onto Sortable Table Rows

Once you have completed the Repeated Details Configuration the corresponding Sortable Table Row objects are created on the branch. From the Branch view explore the branch to locate the Document Organizer object click the expand control to view the Sortable Table Rows.



ItemLines

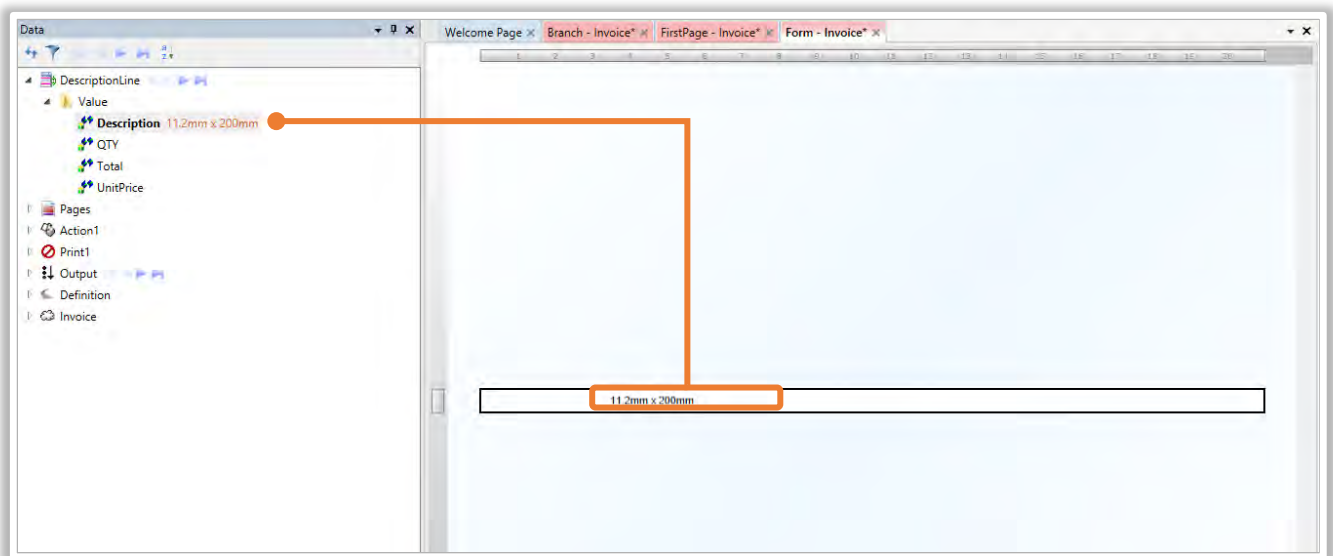
Expand the ItemLines **Sortable Table Row** click on the expand control to reveal the **Form** object. Click on the Form to open the single row subform then from the top of the Data palette expand ItemLines drag and drop the values onto the form to create Text boxes.



Do not worry about positioning the text boxes just ensure the bounding boxes are large enough to accommodate maximum field values. Set Horizontal justification to right on the **UnitPrice**, **VAT** and **Total** text boxes.

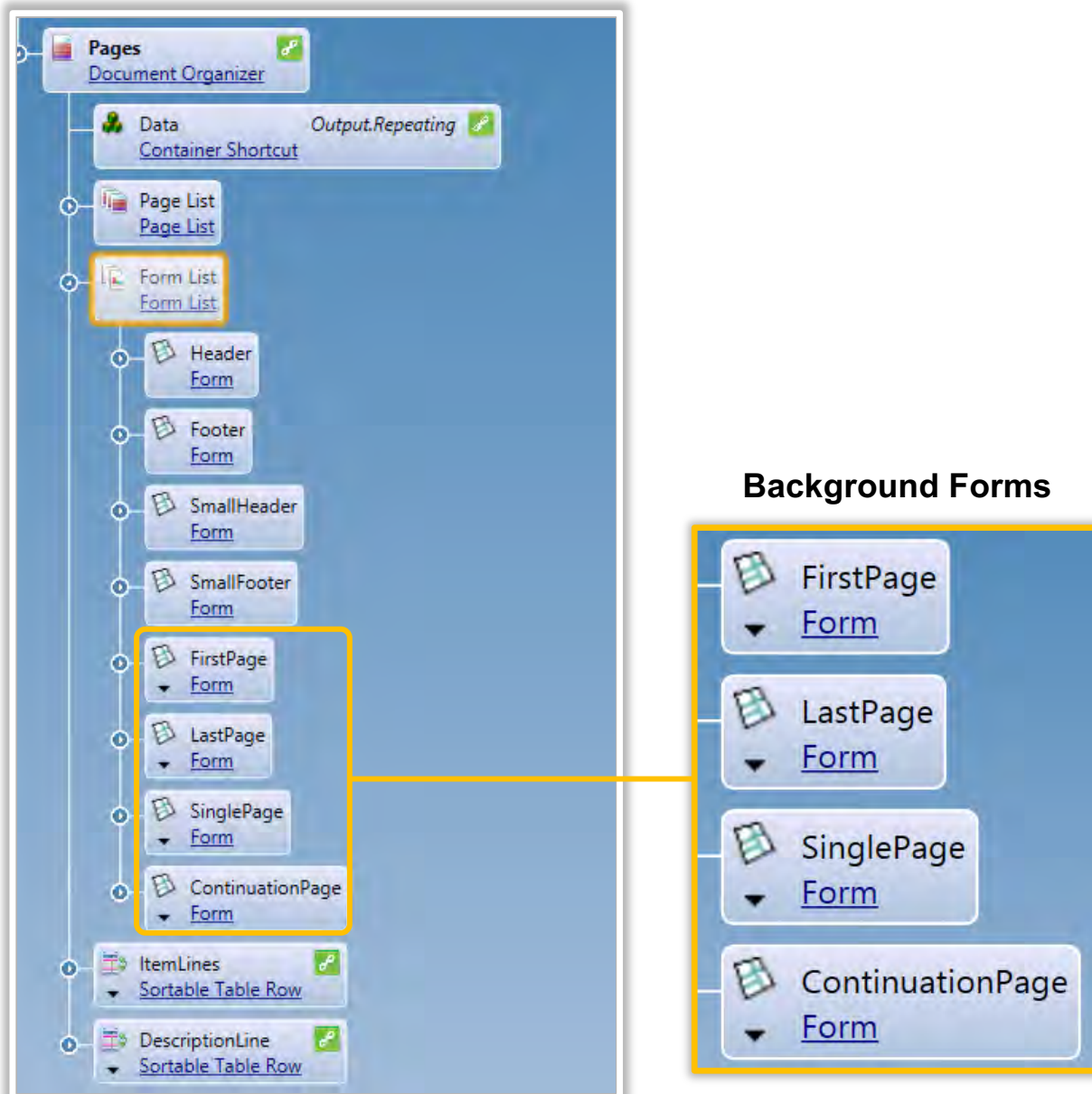
DescriptionLine

Expand the DescriptionLine **Sortable Table Row** click on the expand control to reveal the **Form** object. Click on the Form to open the single row subform then from the top of the Data palette expand DescriptionLine drag and drop the values onto the form to create Text boxes.



Step 8 – Create Background Forms

From the branch locate the **Document Organizer** object and click the expand control then expand **Form List** to expose the forms you defined in the previous step.



Step 8.2 Continuation Page

The background for the continuation page is shown below, use the following objects available on the drawing toolbar.

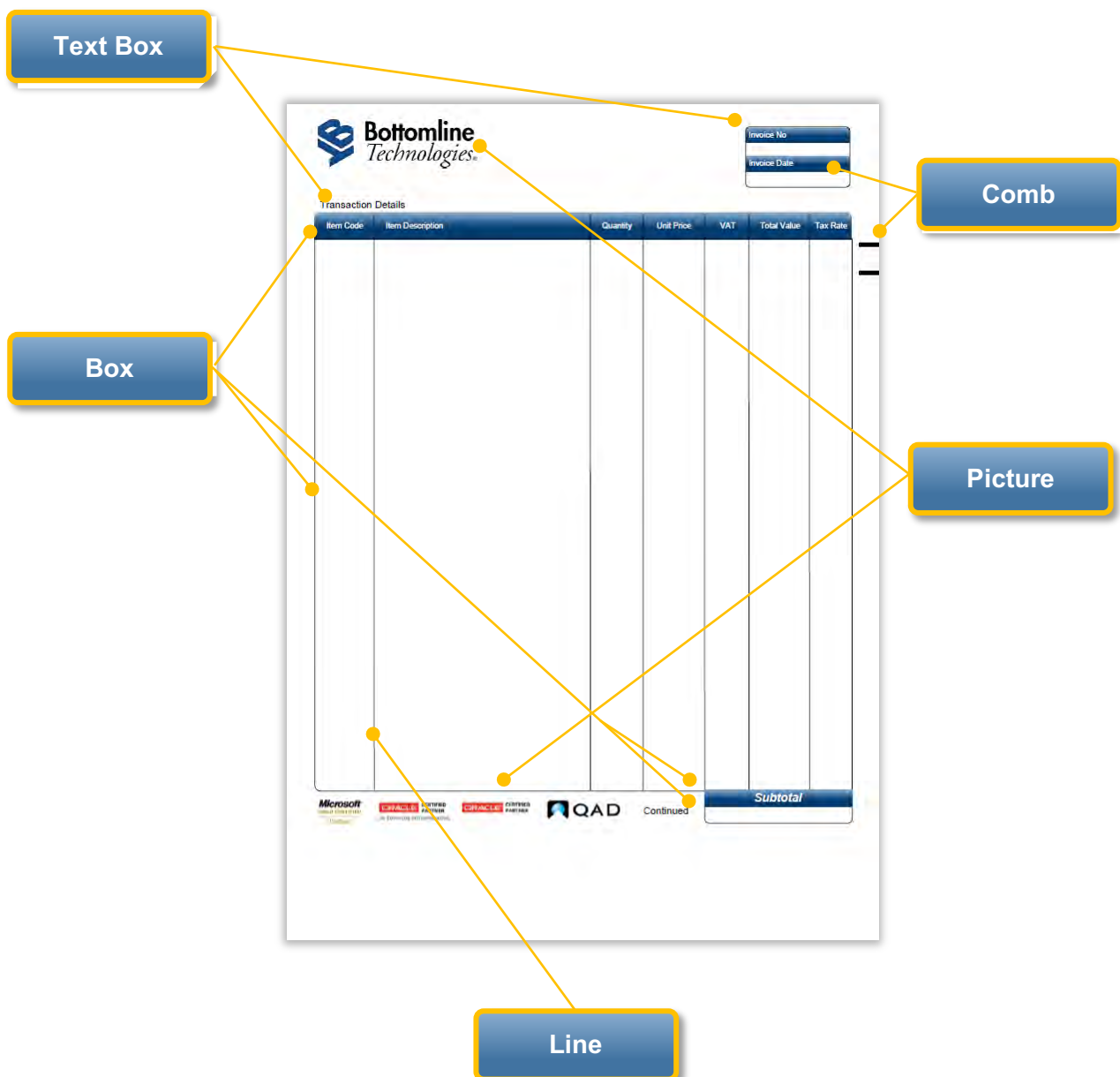
a Text Box

☐ Box

≡ Comb

 Picture

— Line



Step 8.3 Last Page

The background for the last page is shown below, use the following objects available on the drawing toolbar.

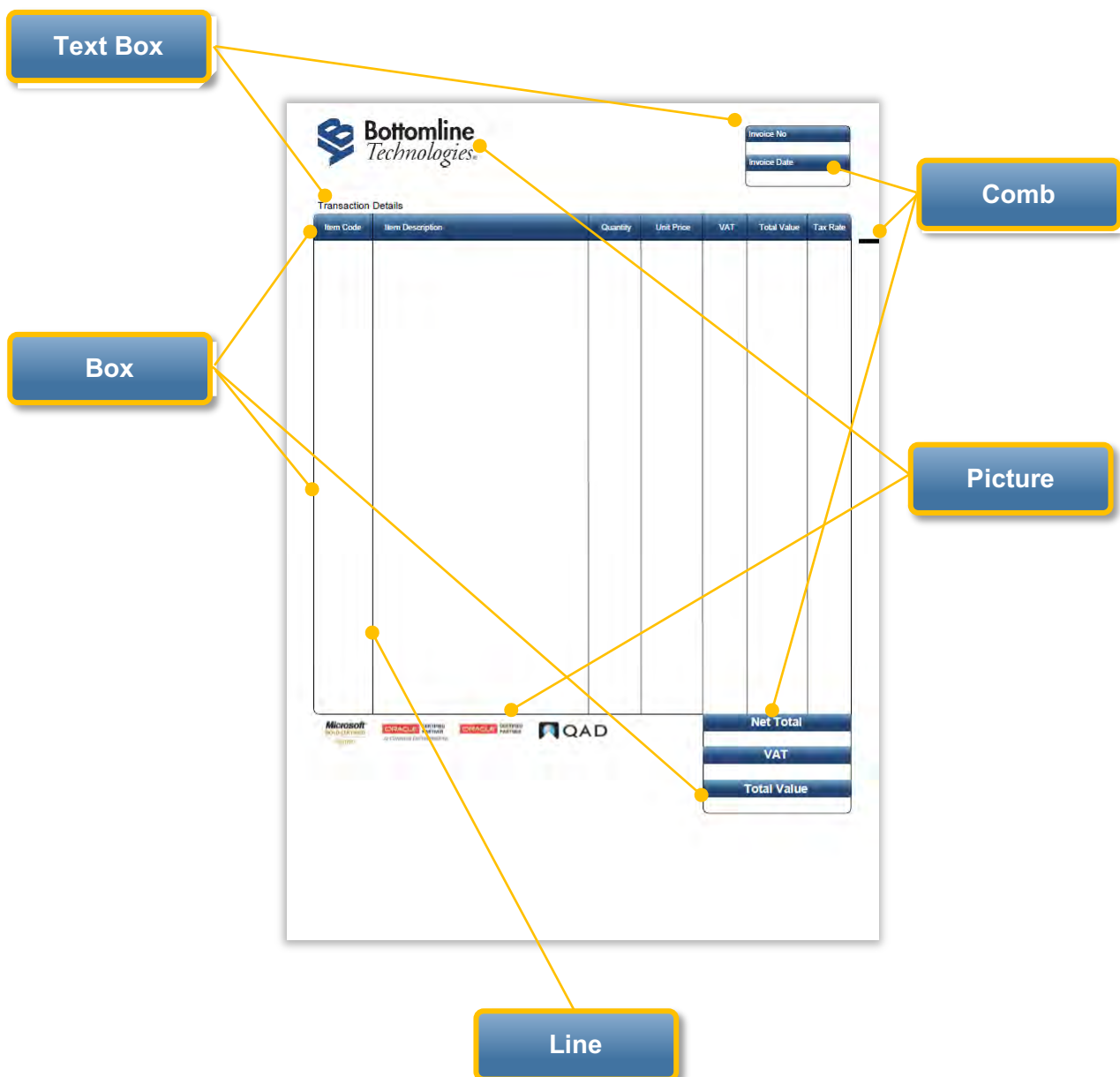
 **Text Box**

 **Box**

 **Comb**

 **Picture**

 **Line**



Step 8.4 Single Page

The background for the single page is shown below, use the following objects available on the drawing toolbar.

a Text Box

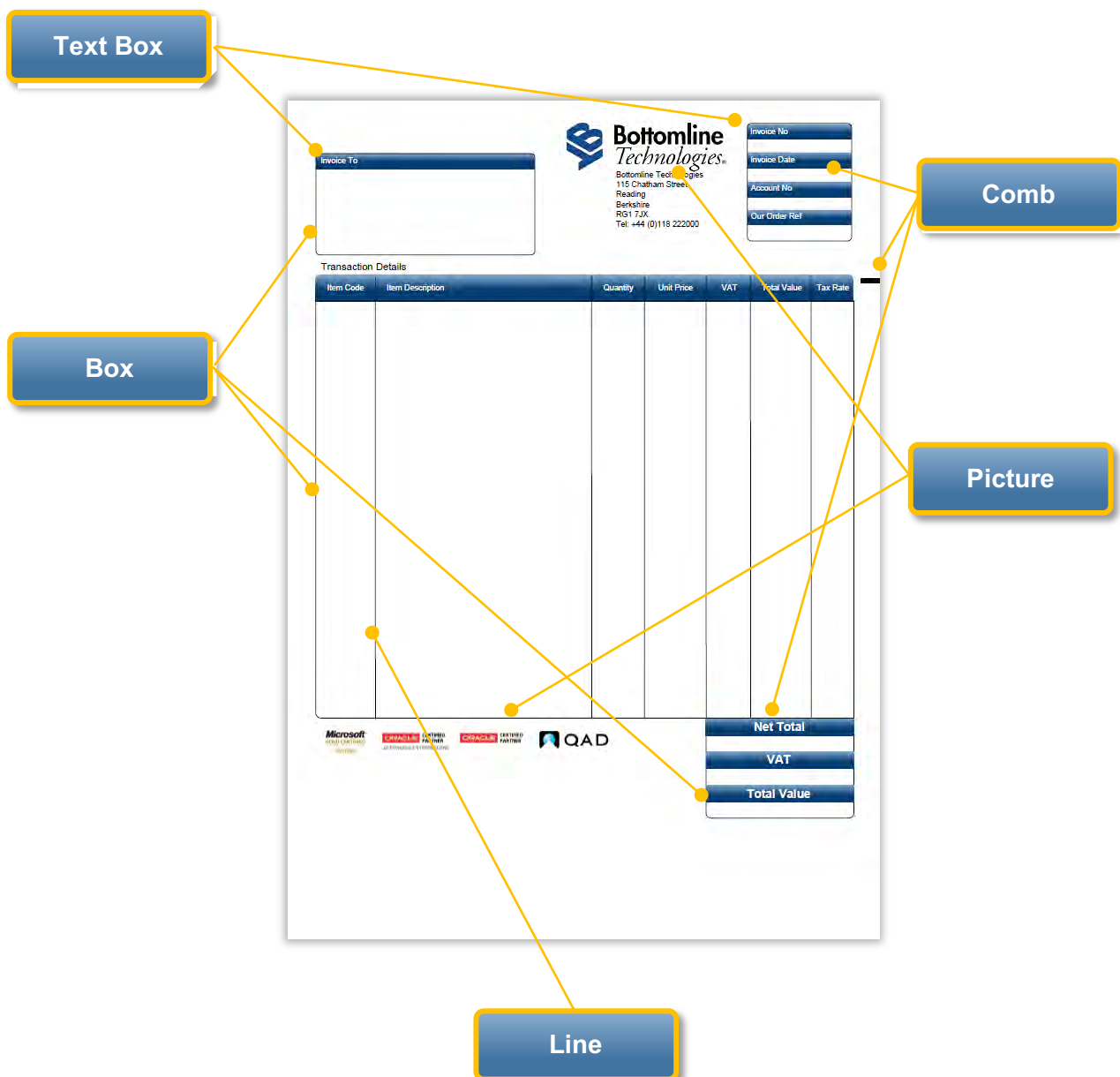
☐ Box

≡ Comb




Picture

— Line

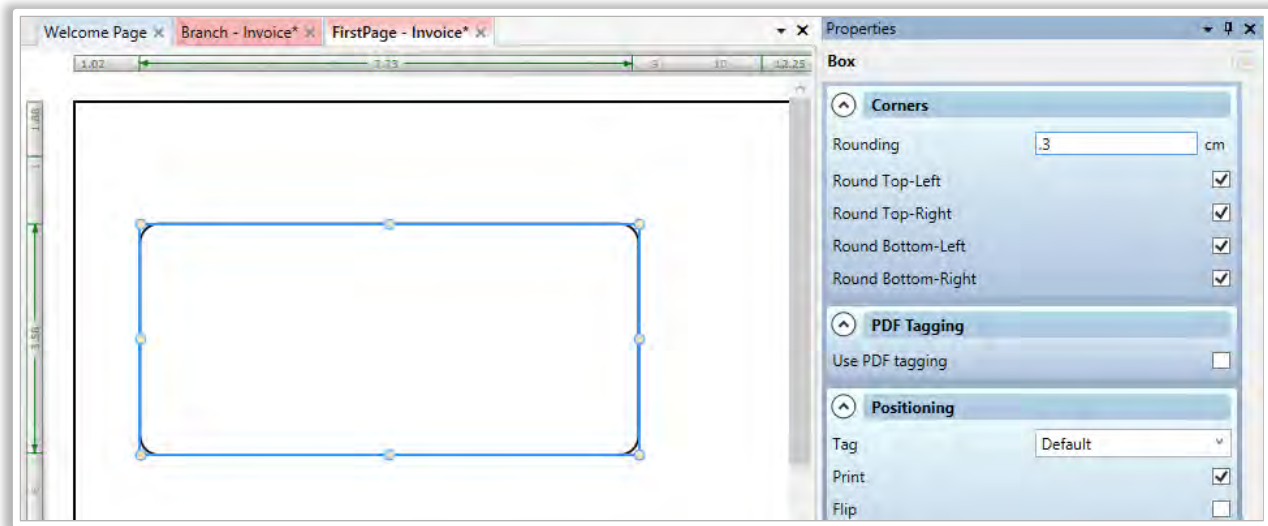


Step 8.5 Gradient Fill

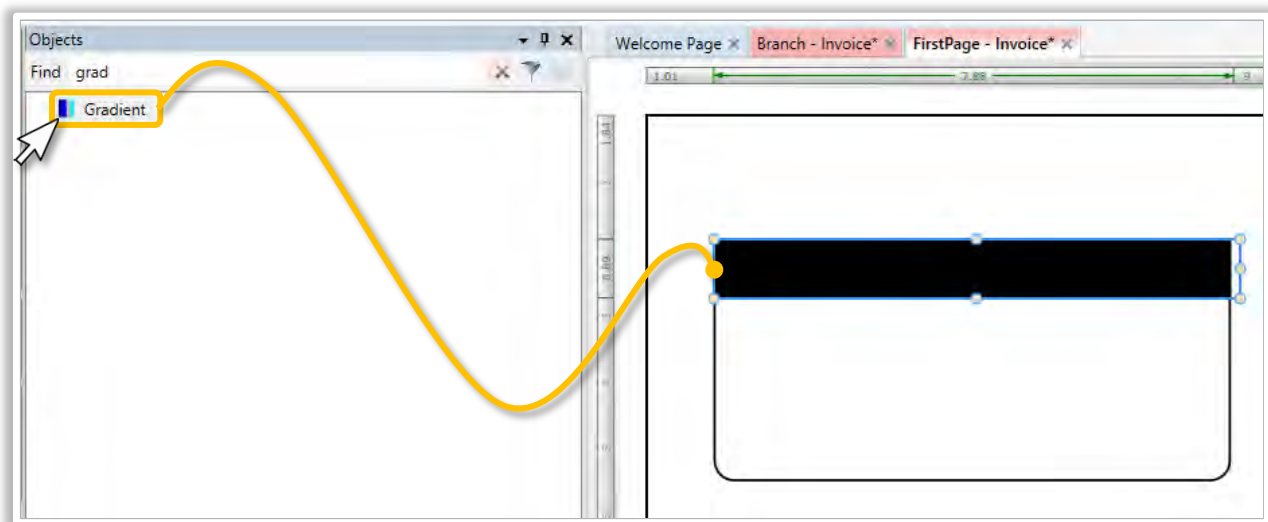
To create the gradient filled box headers with rounded corners complete the following steps.

From the drawing toolbar select Draw Box  then draw the area on the form where you want the box to appear.

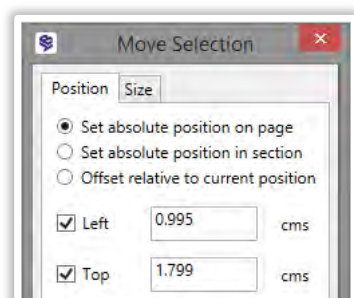
From the Box Properties window set the Rounding to .3 cm and check all rounded corners.




In the Object palette Find field type **grad** then select Gradient then drag and drop it onto the form then using the resize handles make it fit within the current box.

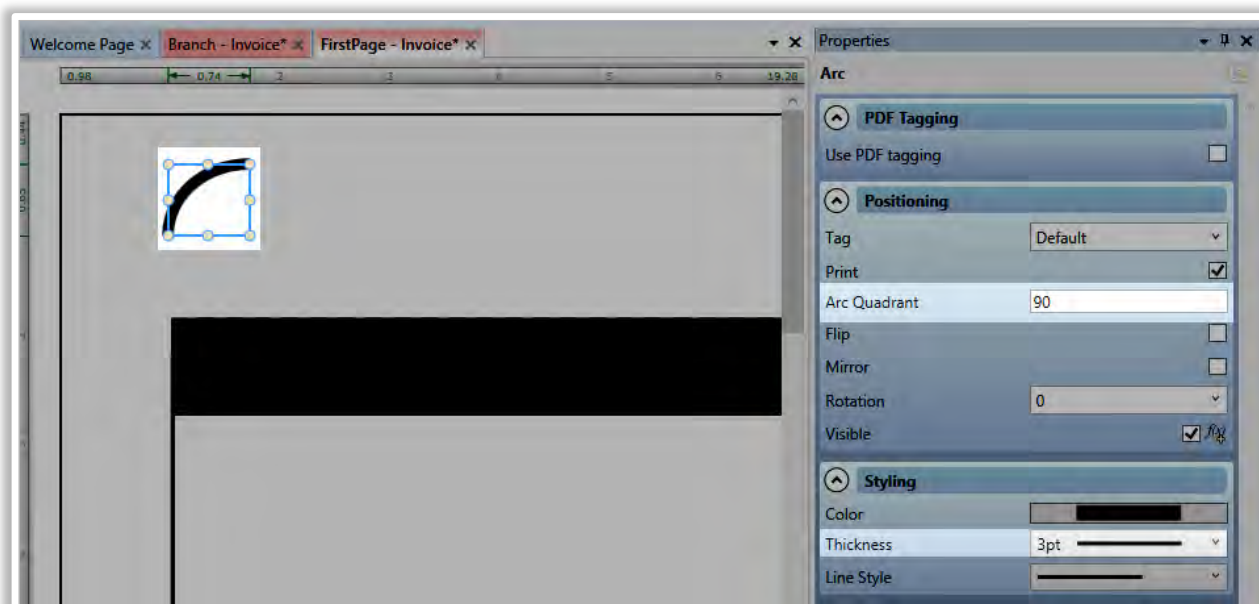


Check the position of the gradient by opening the Move Selection dialog, click the icon on the edit toolbar or CTRL + M.



From the Drawing toolbar select Arc  then draw the arc on the form. From the Arc Properties window set the following options.

- **Arc Quadrant** 90
- **Thickness** 3pt



Change the position and size of the Arc, click Move Selection on the edit toolbar or CTRL + M. On the Position tab change the following parameters.

Where 1 point = 0.0352777778 cms

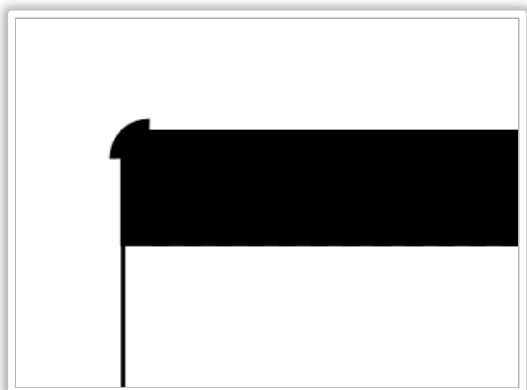
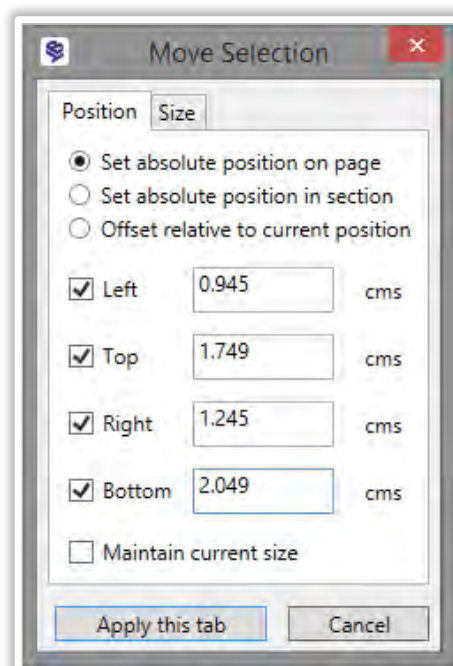
3 points = 0.10

Formula Size of Rounded corners / 2 – 3 points

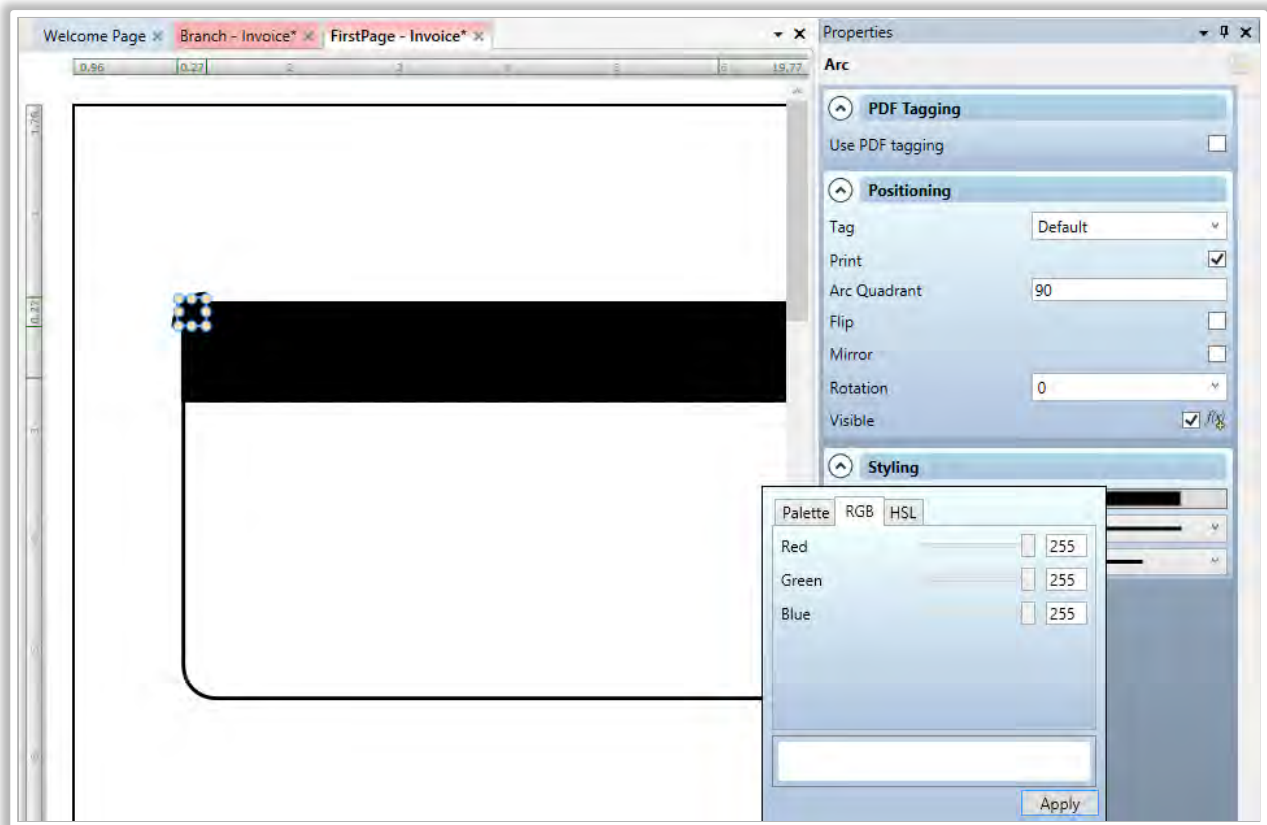
$0.3\text{cm} / 2 - 0.10 = 0.05$

- **Left** 0.945 (Gradient Left position 0.995 - 0.05)
- **Top** 1.749 (Gradient Left position 1.799 - 0.05)
- **Right** 1.245 (Left + 0.3)
- **Bottom** 2.049 (Top + 0.3)

Click Apply this tab to move and resize the Arc.



From the Arc Properties window change the color to white RGB 255.

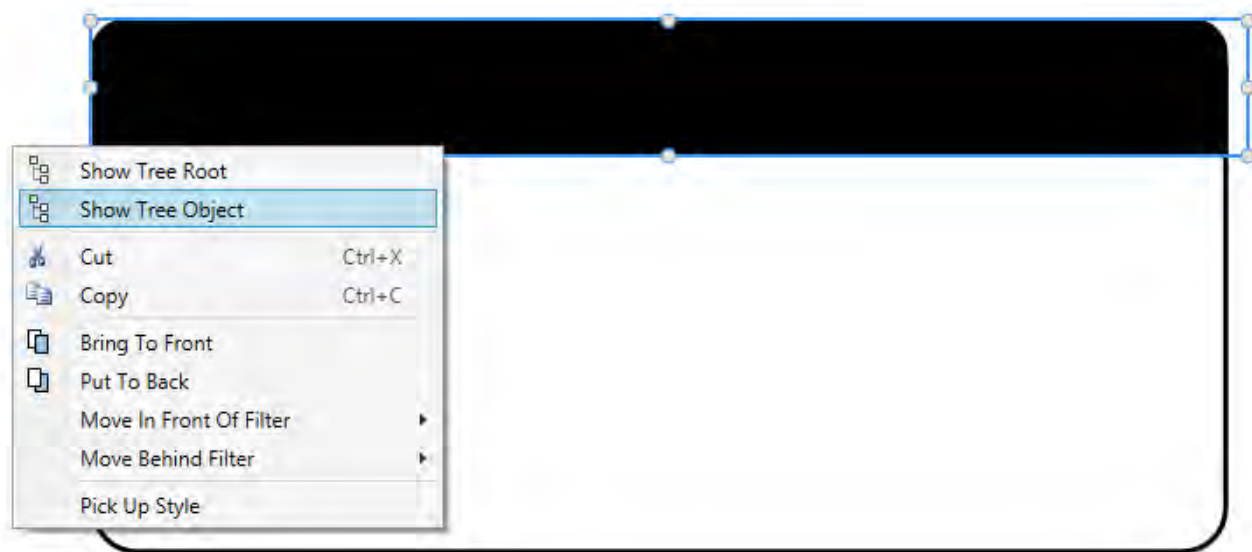


Now complete the process to create another Arc to round the top right corner. Copy the current Arc and paste it changing the Arc Quadrant to 180 and from the move selection set the left position using the formula on the previous page (remember to tick Maintain current size).

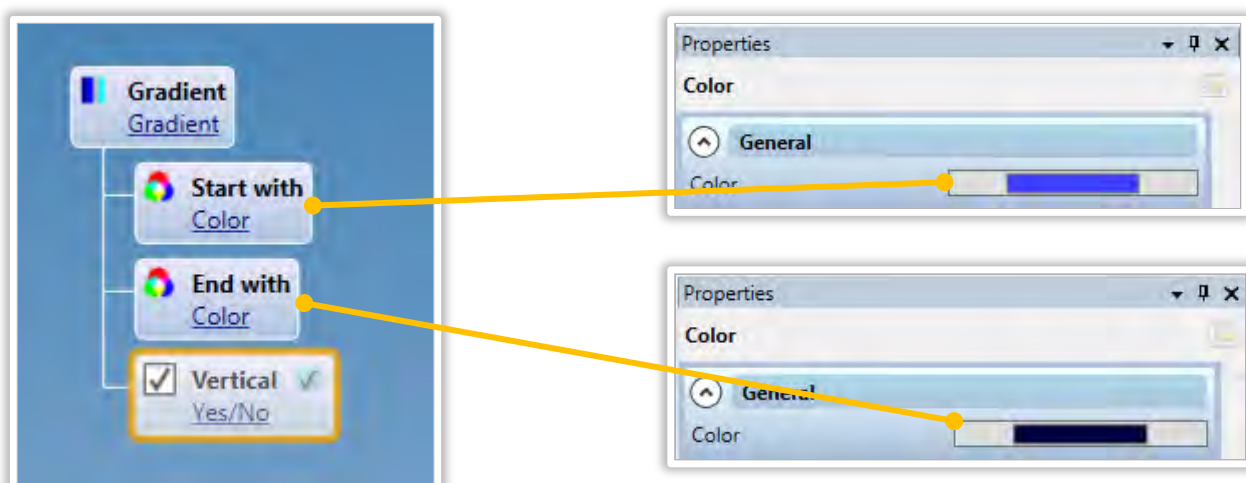


Change the Color of Gradient

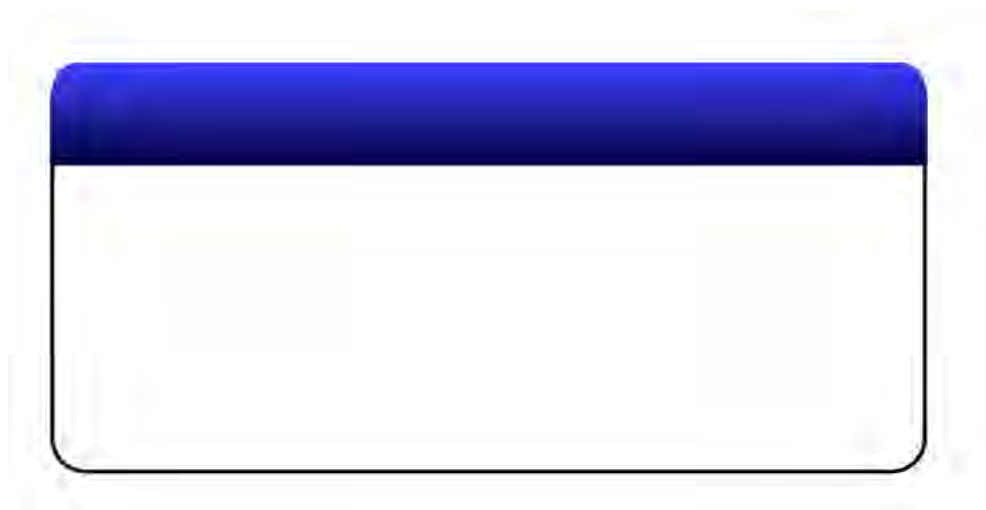
Select the Gradient object right-click and select Show Tree Object from the menu.



From the branch editor select **Start with** and **End with** Color from the Color Properties window.



To preview the output click on Page view tab.

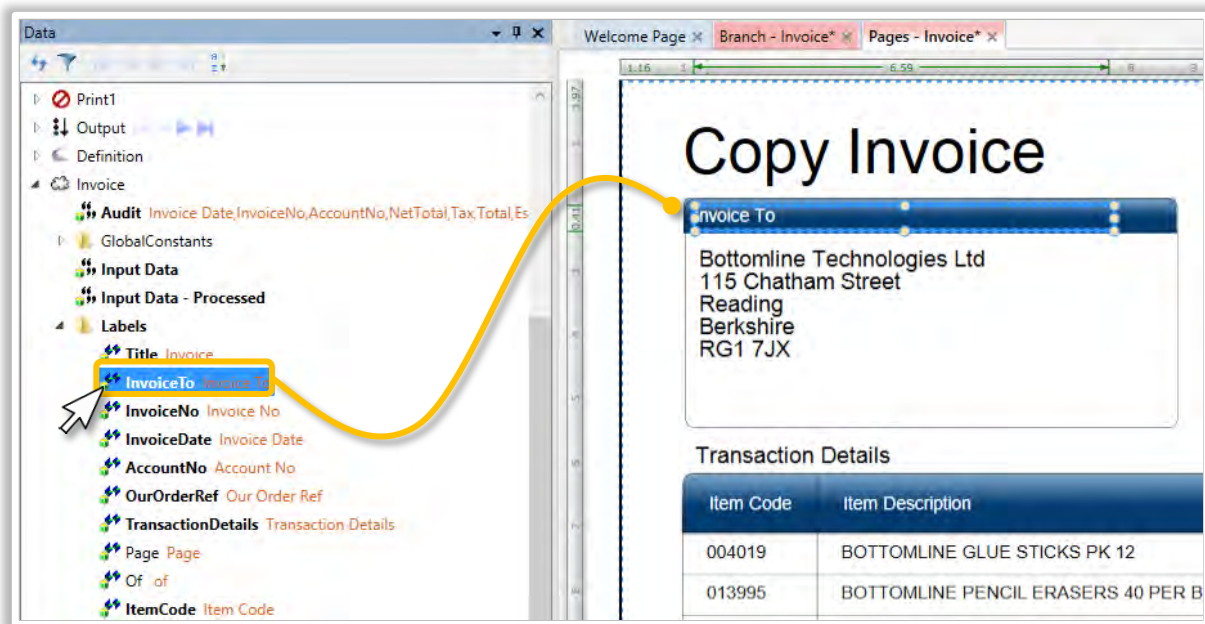


Step 8.6 Mapping Data

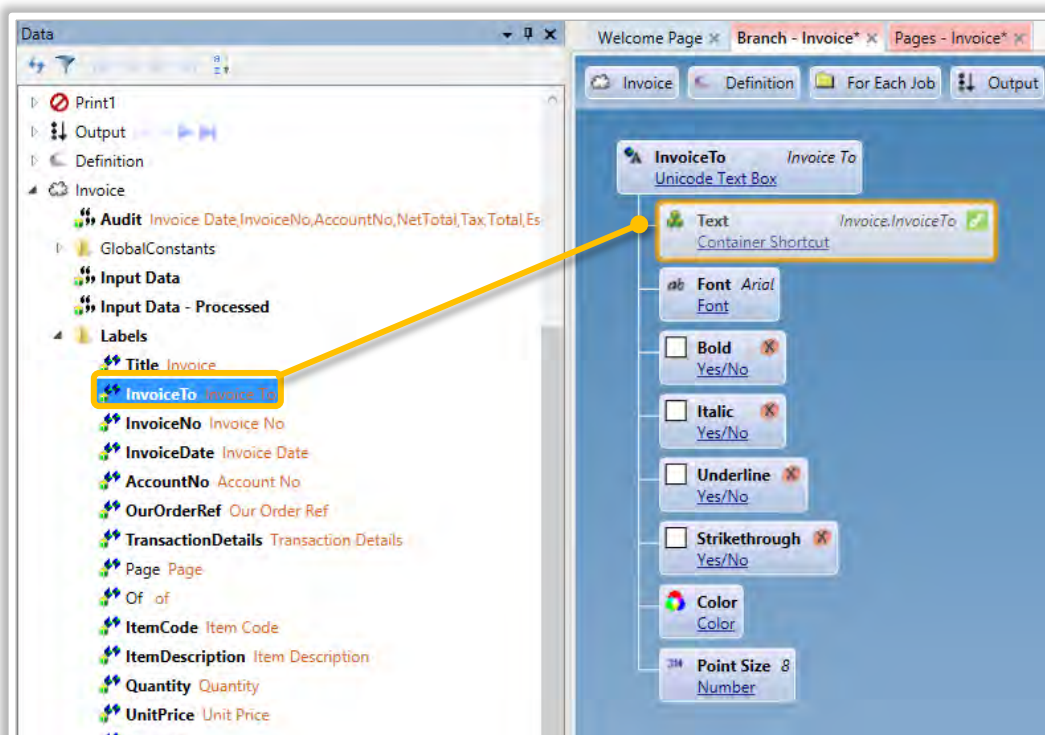
Once you have drawn all the boxes on your background forms you need to map the language labels onto the appropriate text box. There are two method of mapping the data into the text box.

From the Data Palette


Expand Memory Invoice then expand the Labels container select the label then drag and drop it onto the target position on the form. In the example below we have selected **InvoiceTo** and dragged it onto the form which creates the text box with a Container Shortcut.

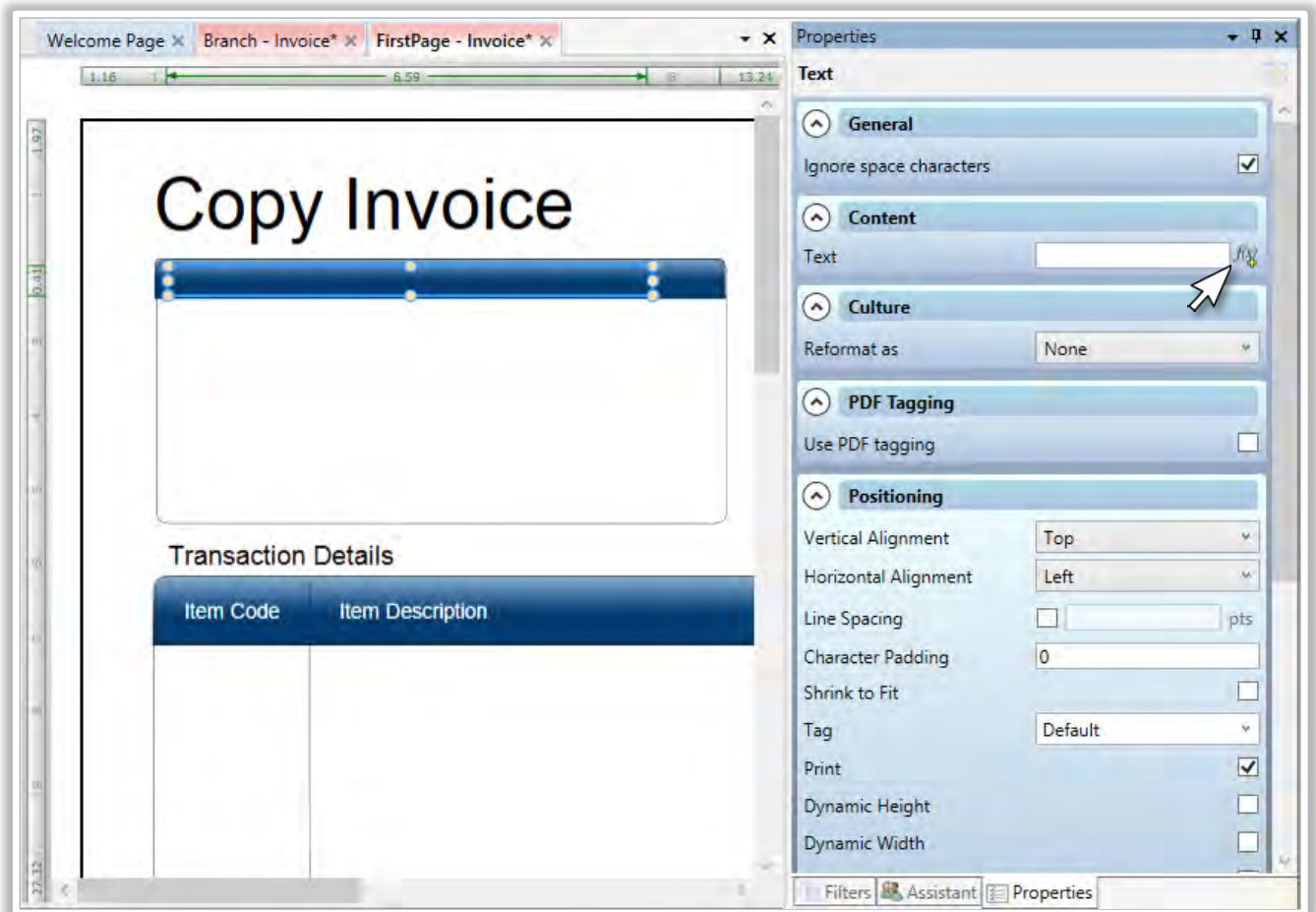


To see how this looks on the branch, right-click the Text box and from the menu select Show Tree Object.

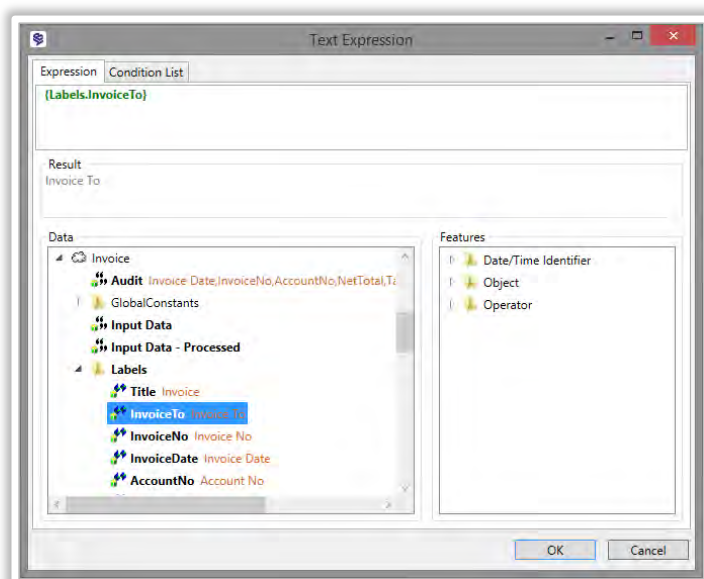


From the Text Properties

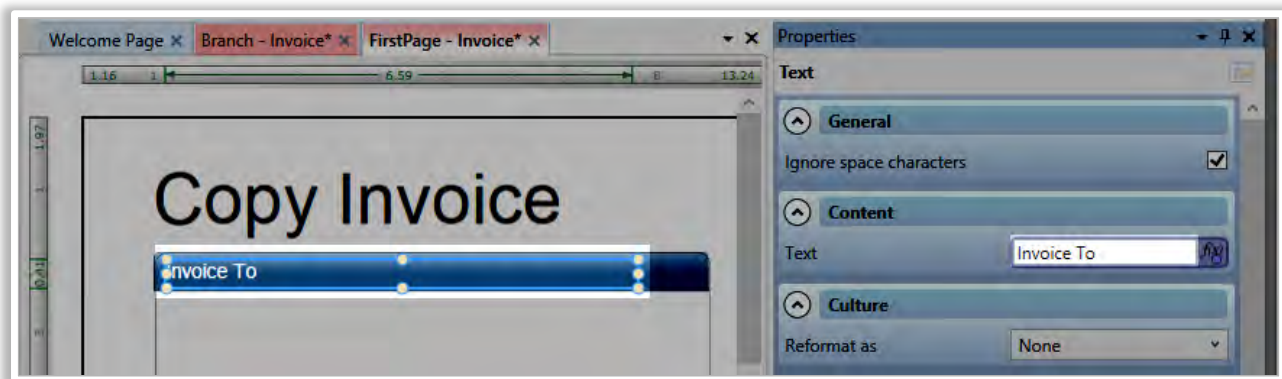
On the drawing toolbar select text box  draw a bounding box where you want the text to appear, then from the Text properties window under the Content section click the expression icon to the right of the Text field.



This opens the Text Expression editor, remove the double quotes from the expression pane then within the Data pane expand Invoice and Labels containers then double click InvoiceTo to commit it to the expression, click OK to close the Text Expression editor.




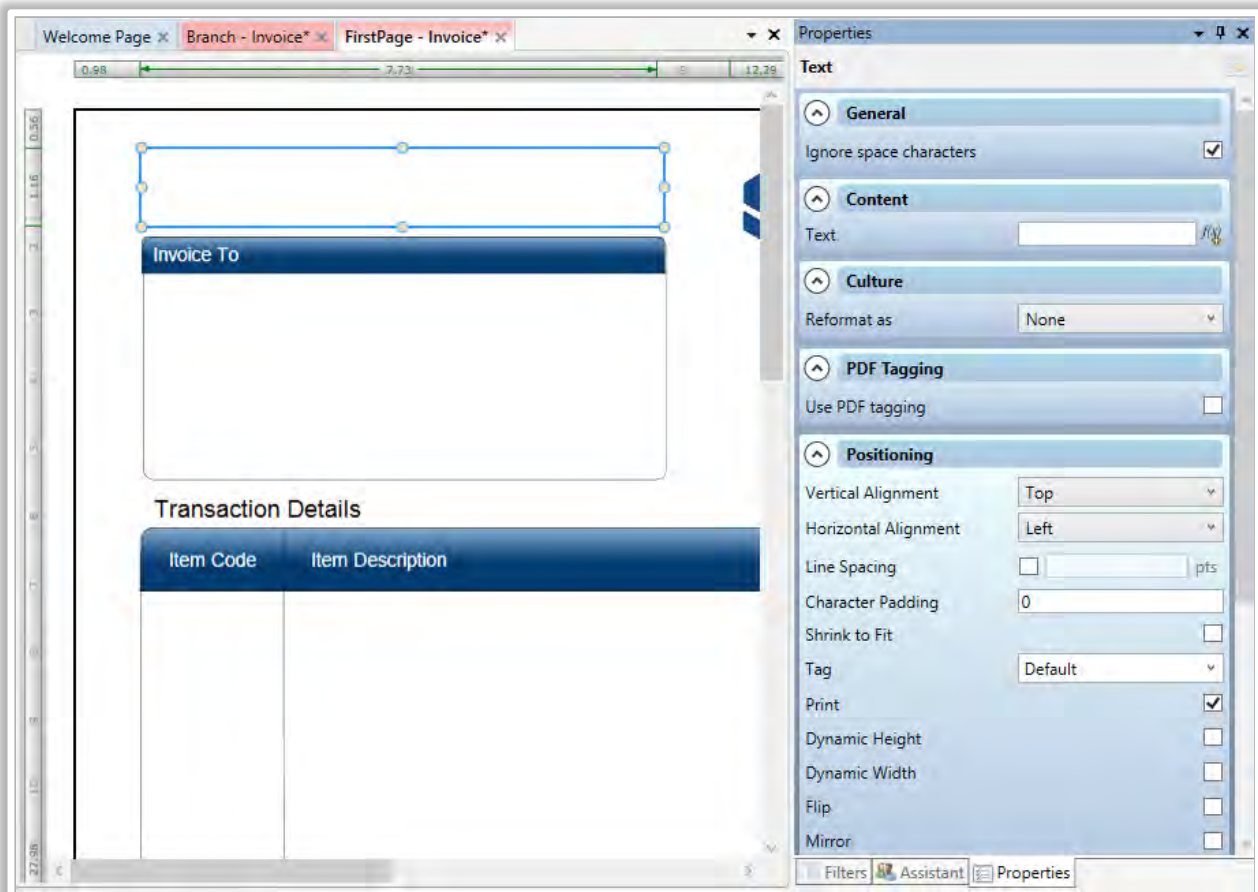
Once closed the InvoiceTo value is displayed on the Form and in the Properties Content Text field.



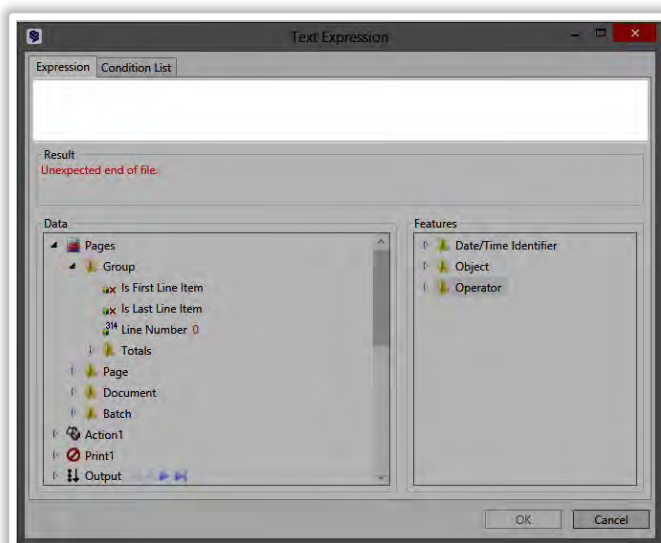
Step 8.7 Build Text Expressions

The title of the Document is Invoice however to identify if it is a copy invoice we need to build a text expression that tests the content of the Copy field in the original data stream and if it contains the string 'COPY' then prefix the title with the language label Copy.

From the drawing toolbar select text box  draw a bounding box where you want the text to appear, then from the Text properties window under the Content section click the expression icon to the right of the Text field.



This opens the Text Expression editor, complete the following steps to build the required expression.



Clear Expression

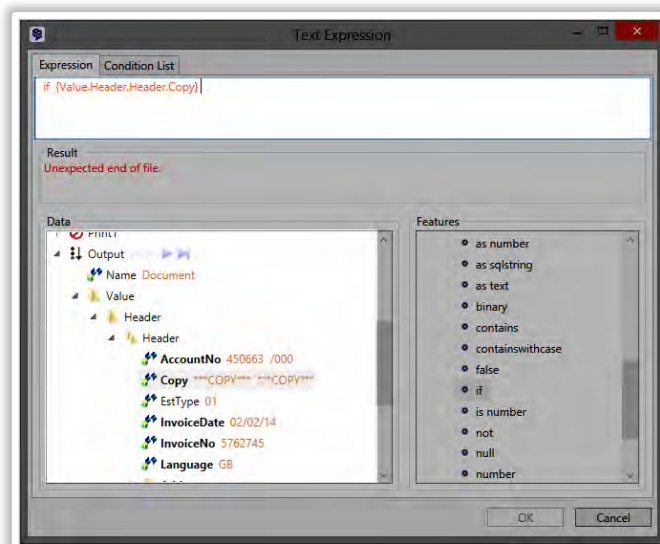
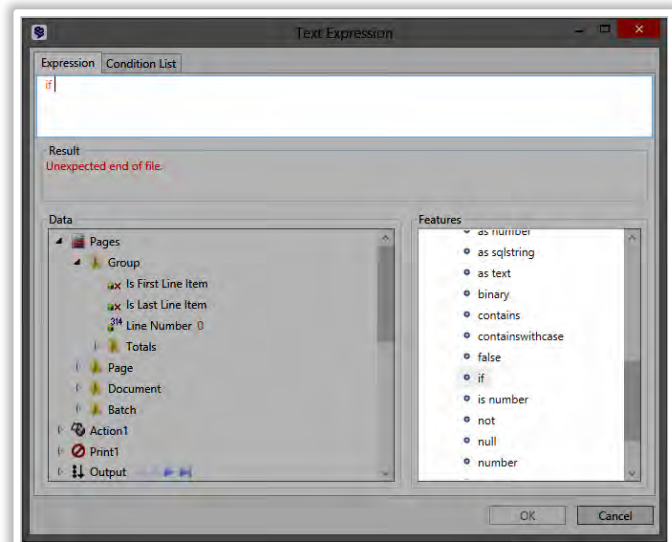
Remove **""** from the Expression window.

Operator

From the Features pane expand Operator and double click **if** from the list to commit the operator to the expression.

Move the cursor to the right of the operator

if |



Data Value

From the Data pane expand Scan Output > Value > Header > Header then double click Copy element to commit to the expression.

Move the cursor to the right of the value

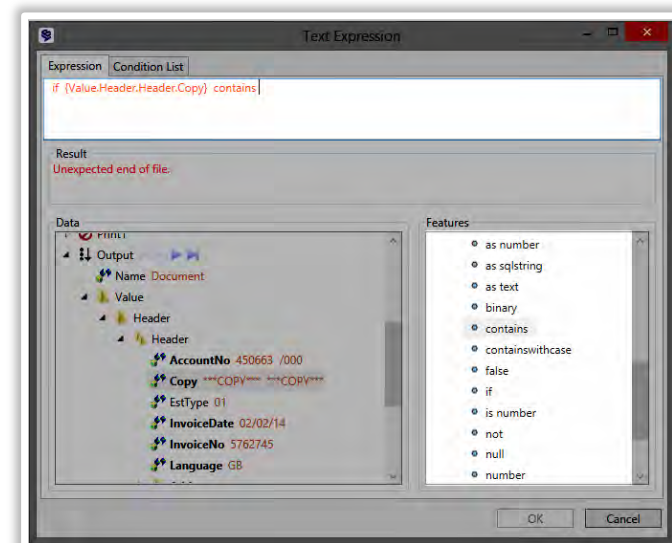
if {Value.Header.Header.Copy} |

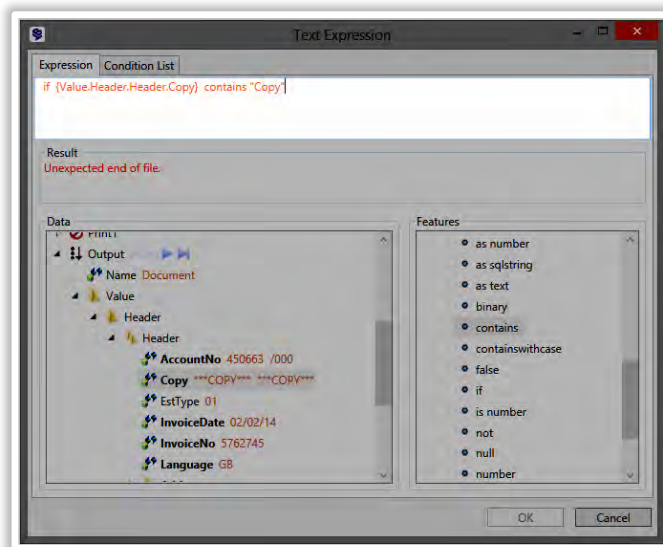
Operator

From the Features pane expand Operator and double click **contains** from the list to commit the operator to the expression.

Move the cursor to the right of the operator

if {Value.Header.Header.Copy} **contains** |





Comparison

Type in the string to match within the selected data value **"Copy"**. If you want to match the case then change the operator to **containswithcase**

Move the cursor to the right of the operator

if {Value.Header.Header.Copy} contains
"Copy" |

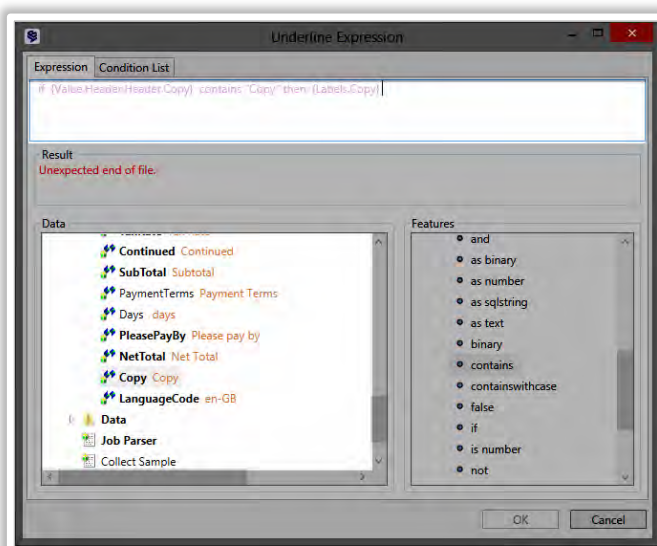
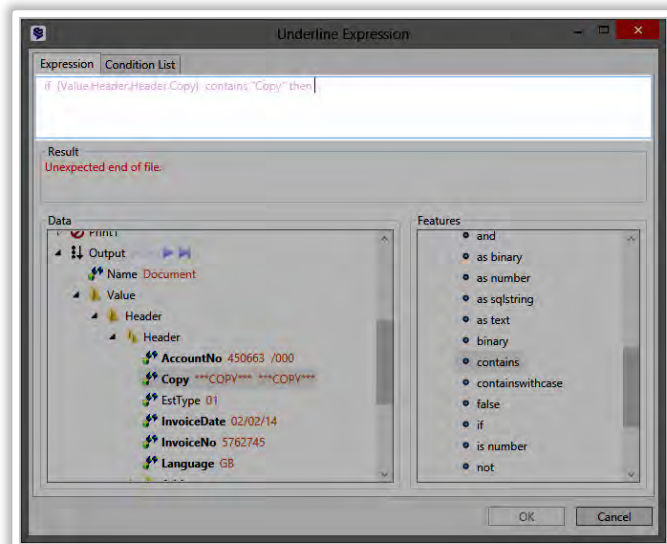
Condition True (then)

As we are using the if statement within the expression editor the results of the comparison are presented by **then** and **else**.

Type **then**

Move the cursor to the right of the operator

if {Value.Header.Header.Copy} contains
"Copy" then |



Label Value

From the Data pane expand Invoice > Labels and double click **Copy** to commit the value to the expression.

Move the cursor to the right of the operator

if {Value.Header.Header.Copy} contains
"Copy" then {Labels.Copy} |

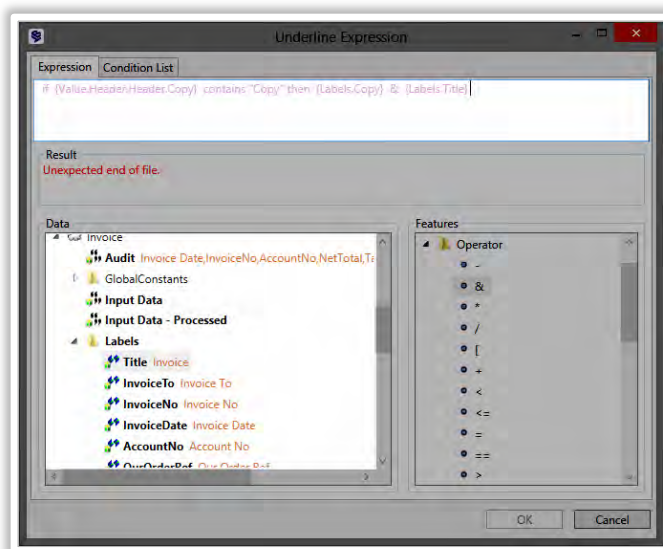
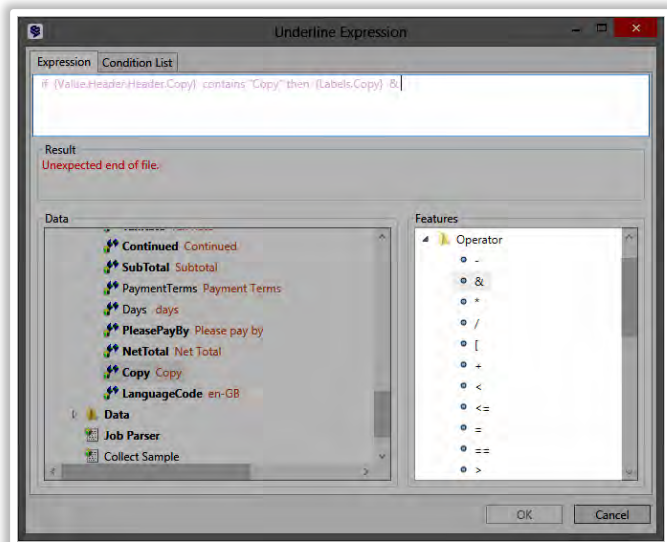
Operator

As we need to construct a sting made up of multiple variable values then we need to use **&** Operator.

From the Features pane expand Operator and double click **&** to commit to the expression.

Move the cursor to the right of the operator

if {Value.Header.Header.Copy} contains
"Copy" then {Labels.Copy} & |



Label Value

From the Data pane expand Invoice > Labels and double click **Title** to commit the value to the expression.

Move the cursor to the right of the operator

if {Value.Header.Header.Copy} contains
"Copy" then {Labels.Copy} & {Labels.Title} |

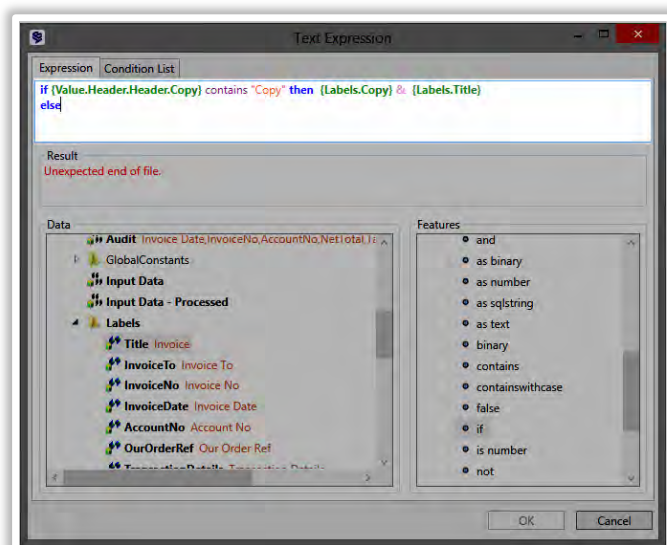
Condition false (else)

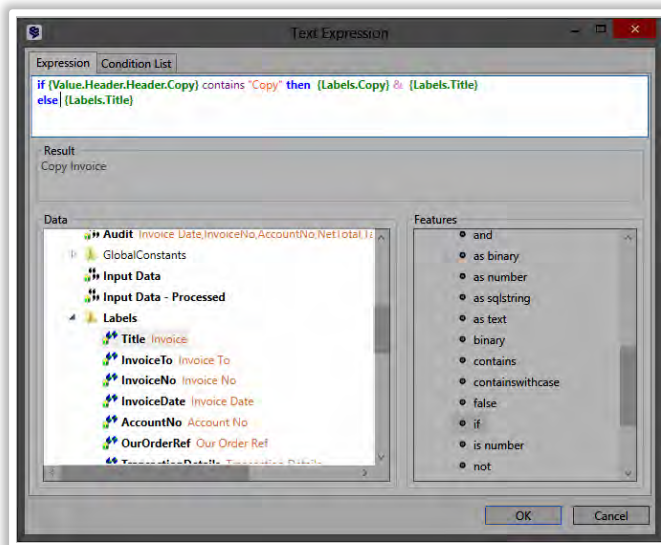
As we are using the if statement within the expression editor the results of the boolean condition are presented by **then** and **else**.

Press enter to start a new line and type **else**

Move the cursor to the right of the operator

if {Value.Header.Header.Copy} contains
"Copy" then {Labels.Copy} & {Labels.Title}
else |





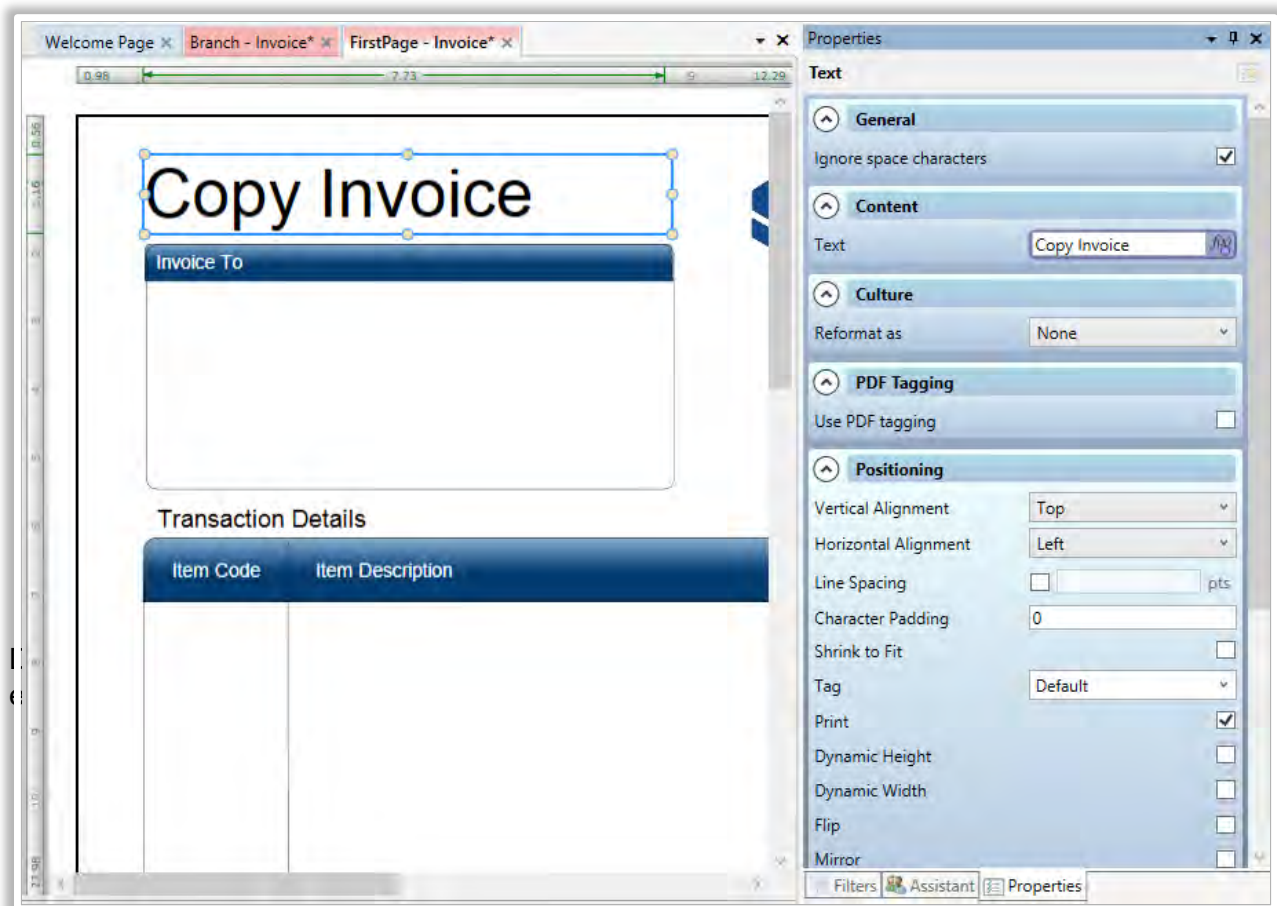
Label Value

From the Data pane expand Invoice > Labels and double click **Title** to commit the value to the expression.

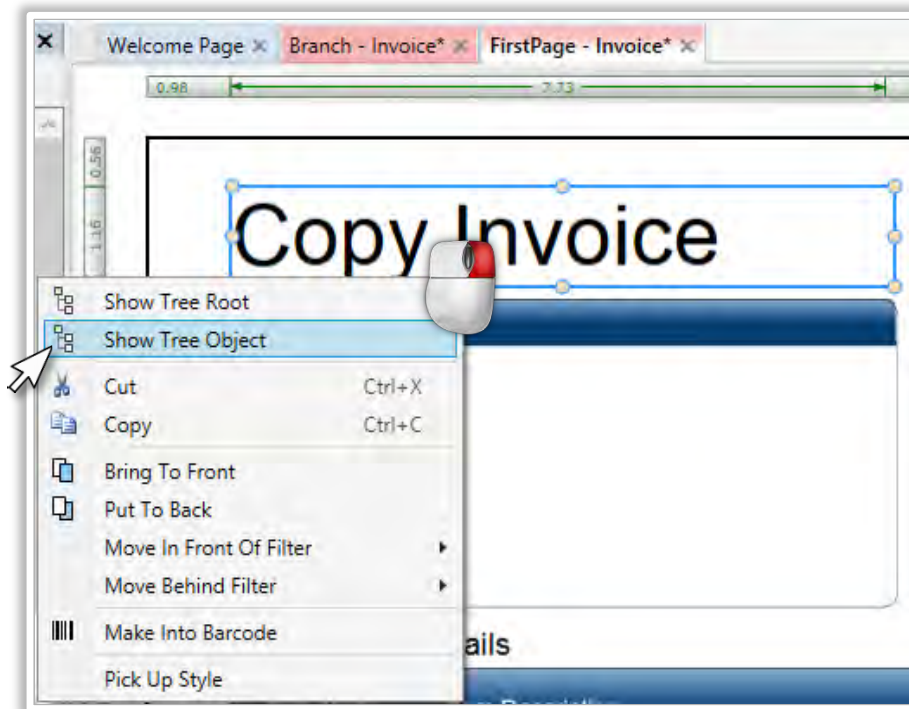
if {Value.Header.Header.Copy} contains
"Copy" then {Labels.Copy} & {Labels.Title}
else {Labels.Title}

Click OK.

You will now see that the text appears in the bounding box on the form and also in the Properties Content Text field.

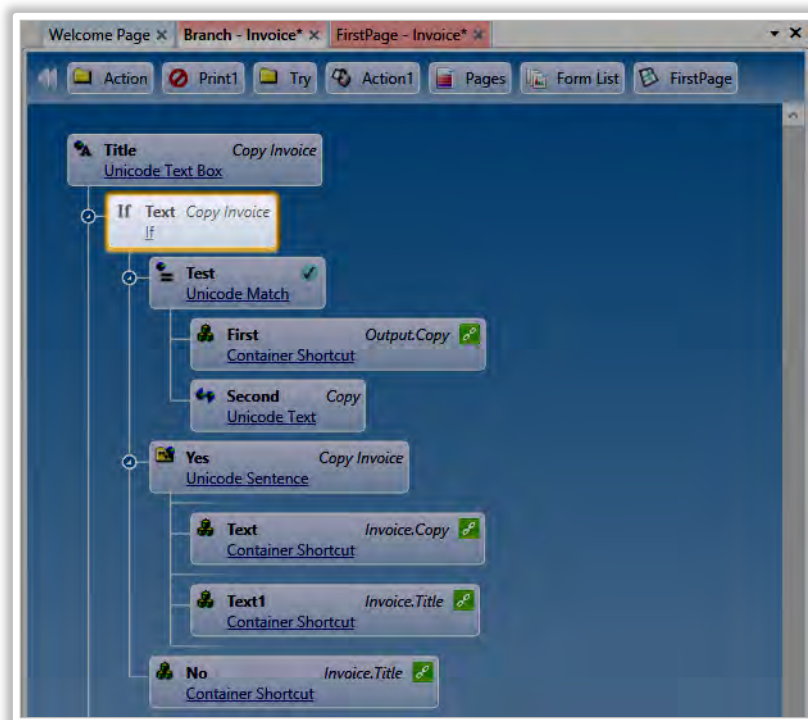


To see how this expression appears on the branch right-click the bounding box and from the menu select Show Tree Object.

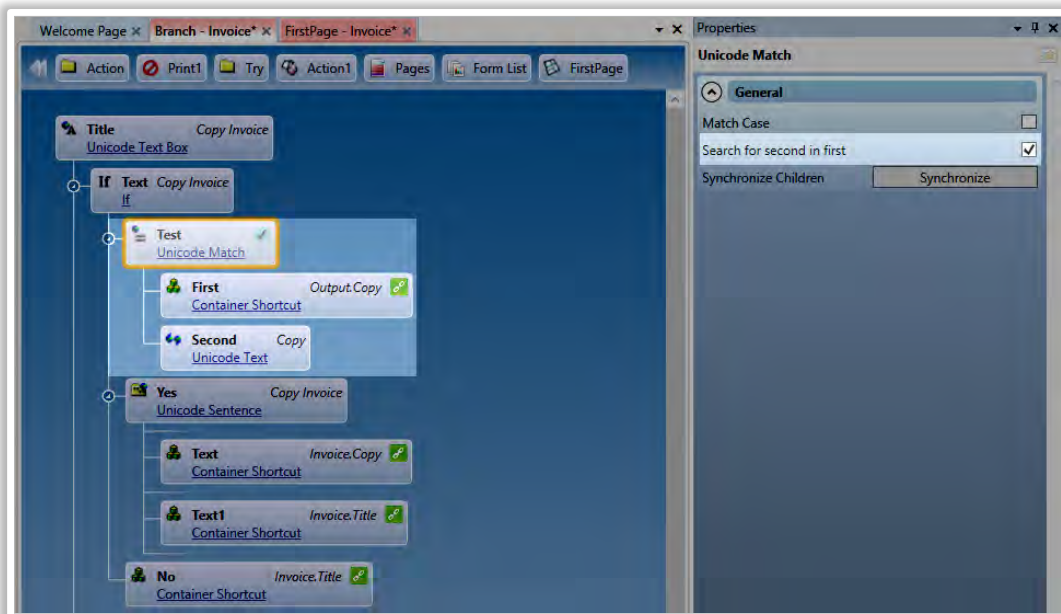


Expand the **If** object to show its children. Let us take some time to explore how this structure relates to the expression created.

if {Value.Header.Header.Copy} contains "Copy" then {Labels.Copy} & {Labels.Title}
else {Labels.Title}



if {Value.Header.Header.Copy} contains “Copy” then {Labels.Copy} & {Labels.Title}
else {Labels.Title}

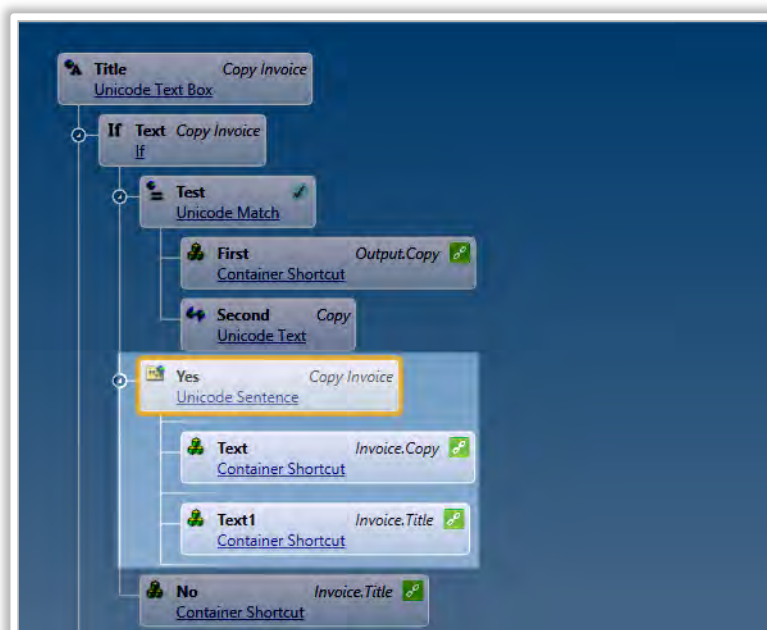


The **Unicode Match** object is used as the Boolean comparison. Under the Unicode Match object is a First and Second which is used to build the test.

- **First** is from the data {Value.Header.Header.Copy}
- **Second** is the constant we want to search for in the data value “Copy”

As the operator **contains** was used then **Search for second in first** option is checked in the Unicode Match properties General settings.

if {Value.Header.Header.Copy} contains “Copy” then {Labels.Copy} & {Labels.Title}
else {Labels.Title}



The **Yes** relates to **then**, the **Unicode Sentence** object is used because the Operator **&** is used in the expression to construct a string containing multiple variable values.

The Container Shortcuts objects under the Unicode Sentence relate to the Label value

{Labels.Copy}

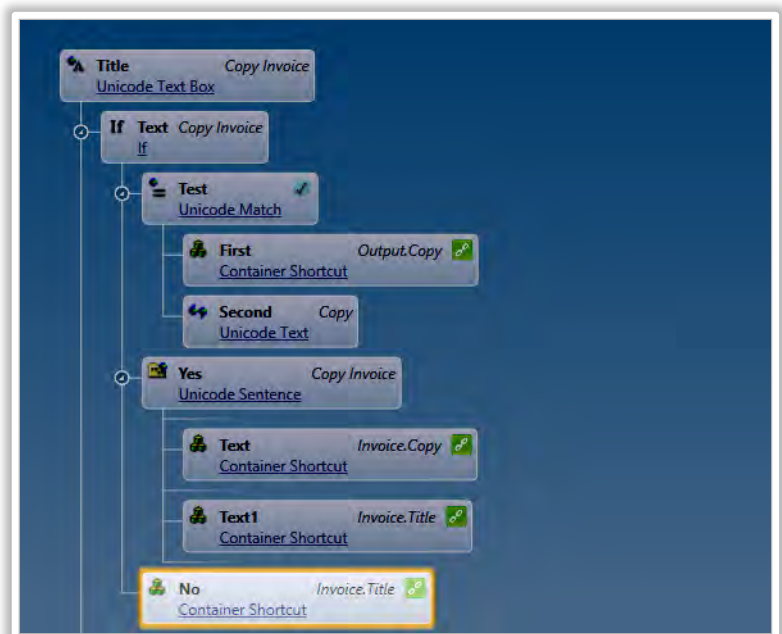
{Labels.Title}

if {Value.Header.Header.Copy} contains “Copy” then {Labels.Copy} & {Labels.Title}

else {Labels.Title}

The **No** relates to **else**, the Container Shortcut is a link to the Label value

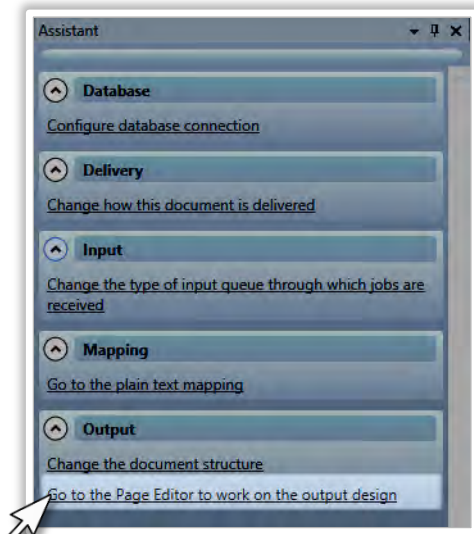
{Labels.Title}



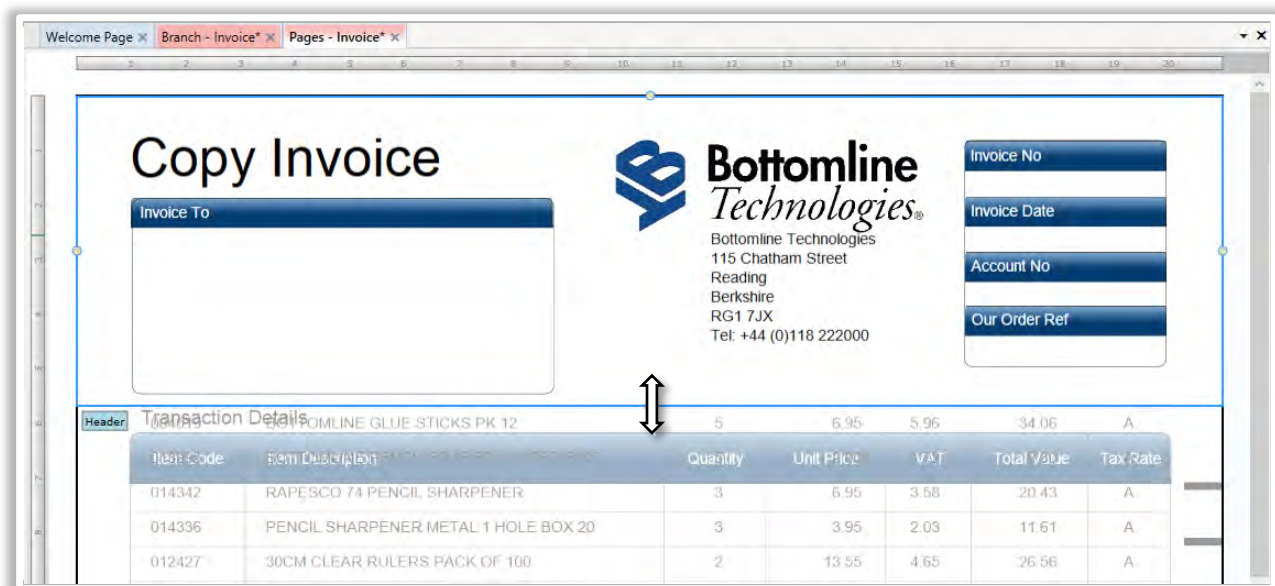
Step 9 Map Header Data

As we have completed the background forms and sortable table rows we can now adjust the size of the header and map the variable data.

From the Assistant window select **Go to the Page Editor to work on the output design**

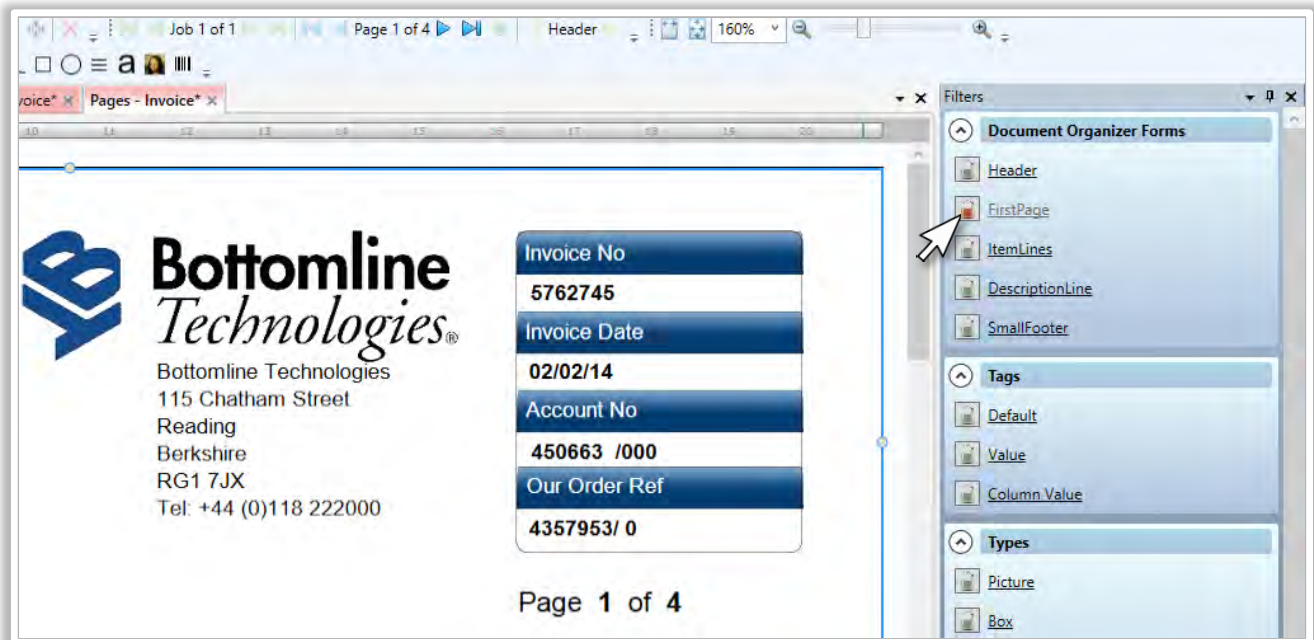


Adjust the size of the header so that the item lines fit the background form, click in the header section so that the header bounding box becomes active then using the resize handles make the necessary adjustment

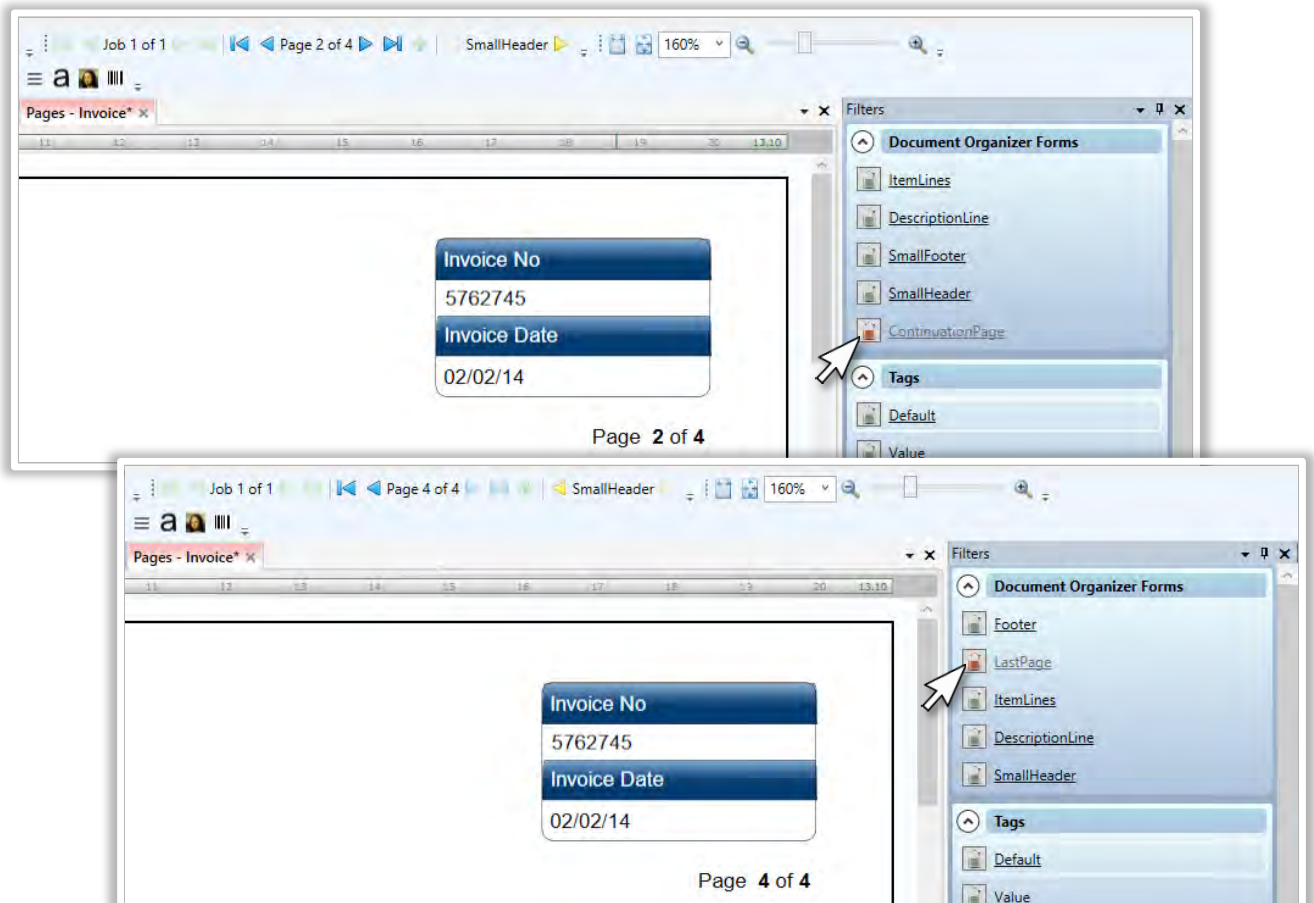


Step 9.1 Lock Background Forms

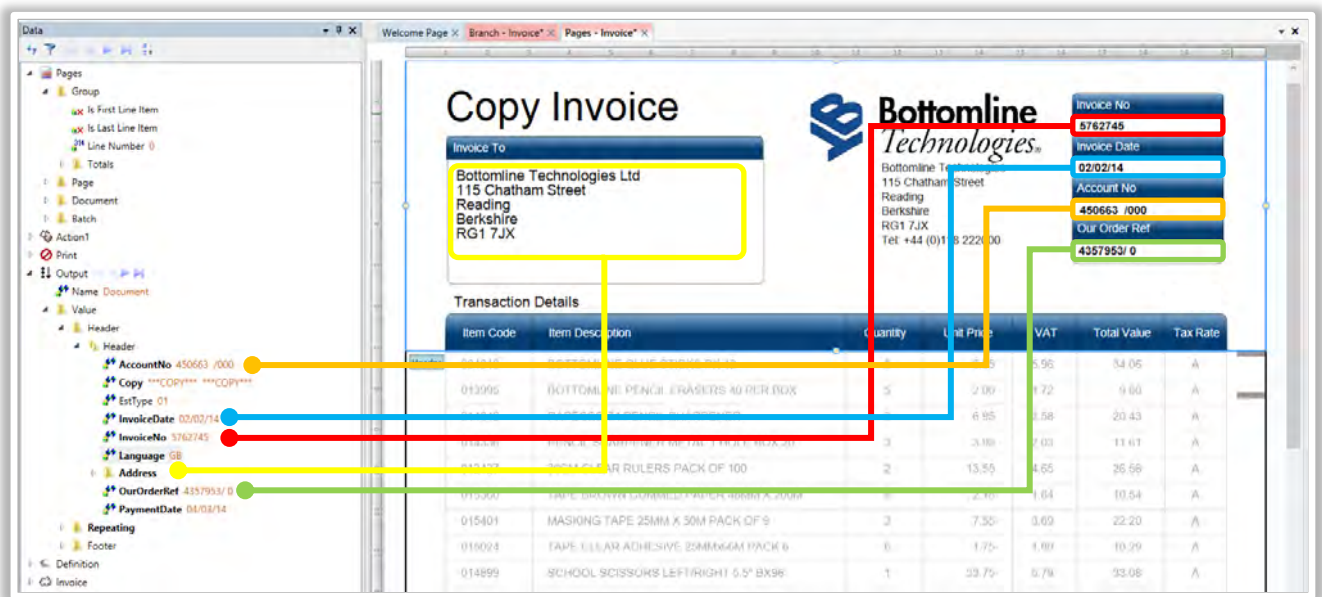
To avoid accidentally changing the background forms during the data mapping you can lock them. From the Filters window Document Organizer Forms click the lock icon next to FirstPage.



As the application is context based the Filter only shows the active background forms, to view the other forms use the navigate button to step through the pages to show the background forms in the Filter window.



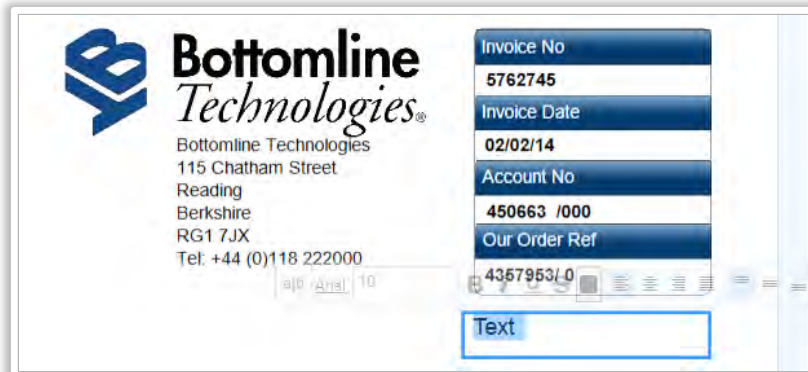
Now map the header data onto the form. From the data palette expand Output > Value > Header > Header.



Step 9.2 Page N of M

The Document Organizer automatically counts the pages produced and as such the page number **N** and page count **M** are available within the data palette, which can be mapped onto the form.

Complete the following steps to add Page N of M to your output.

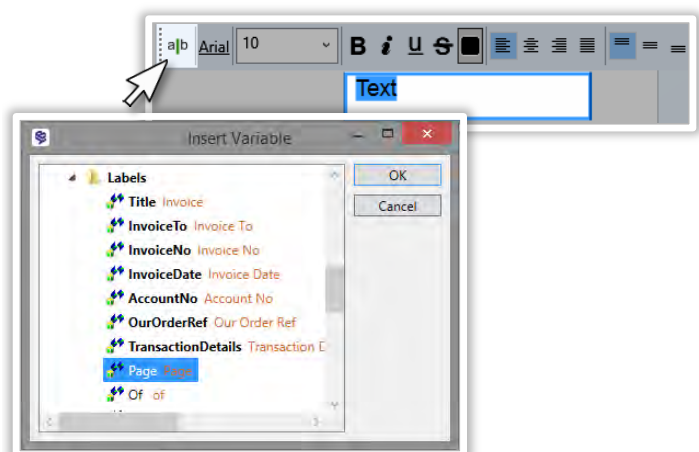


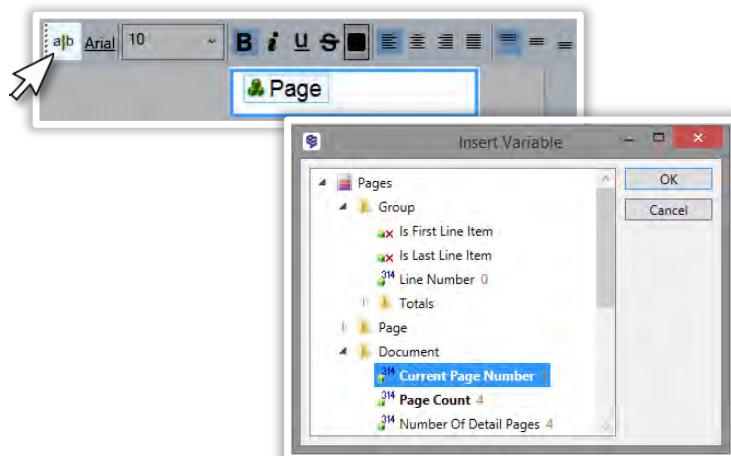
Text Box

From the drawing toolbar select text box **a** and draw the bounding box where you want the Page number to appear.

Page Label

From the floating toolbar click Insert Variable then Expand Invoice > Labels and select Page, click OK.



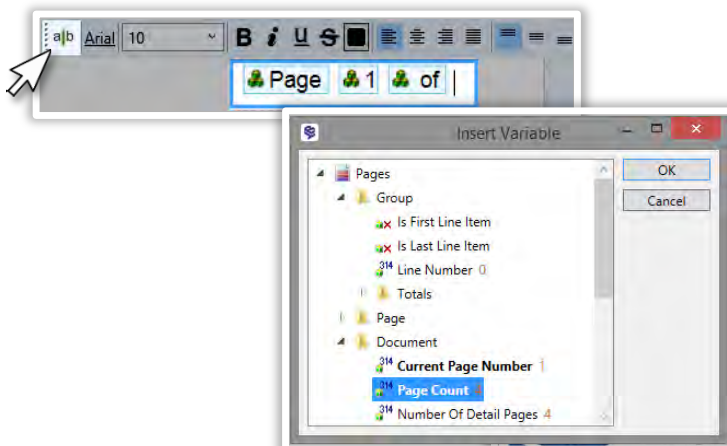
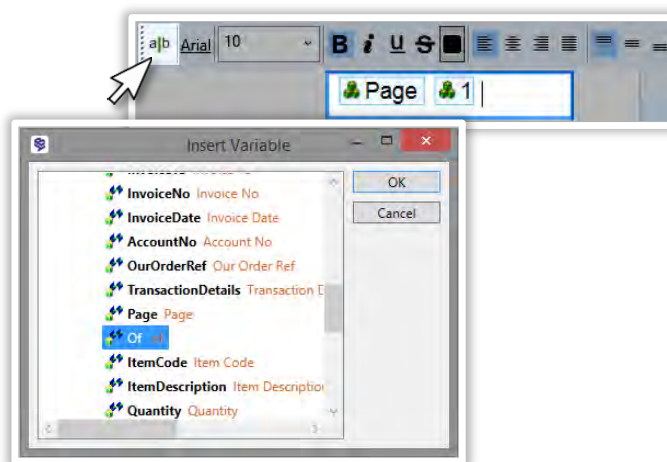


Current Page Number

Add a space after Page then click **B** for Bold style. Click on Insert Variable expand Pages > Document and select Current Page Number, click OK.

Of Label

Add a space after Current Page Number then click **B** to turn off Bold style. Click on Insert Variable expand Invoice > Labels and select Of, click OK.



Page Count

Add a space after Of then click **B** for Bold style. Click on Insert Variable expand Pages > Document and select Page Count, click OK.




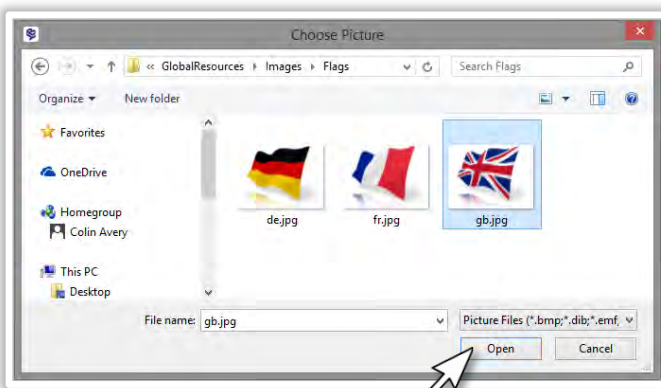
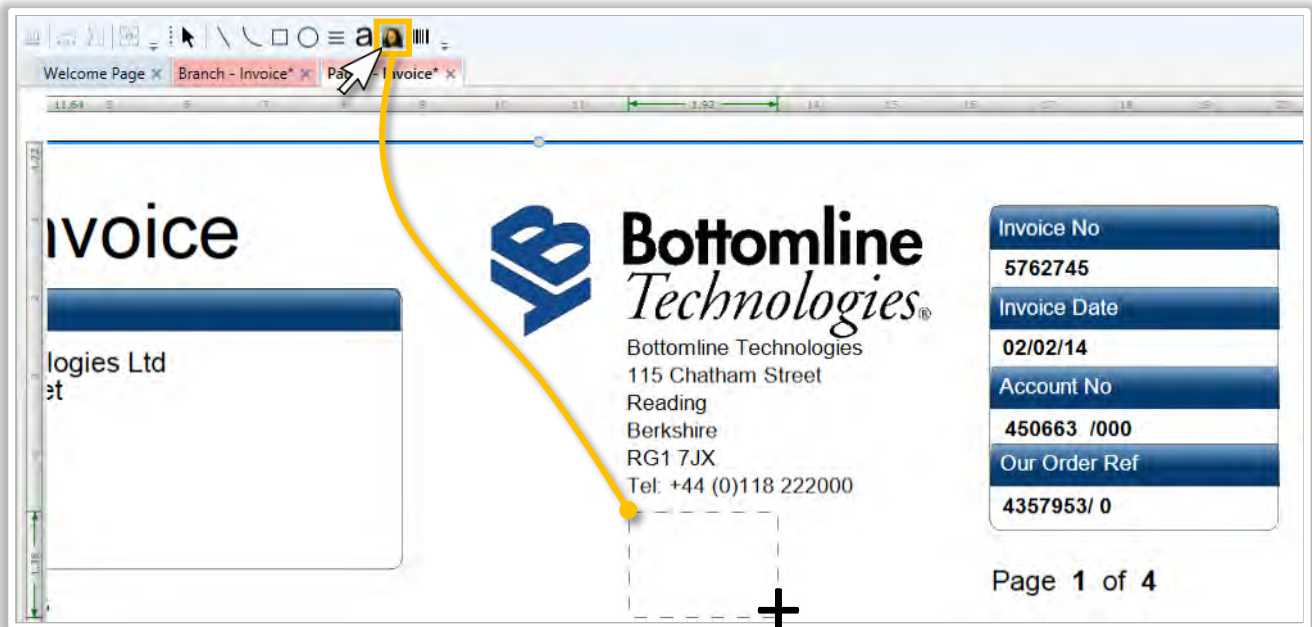
Step 9.3 Dynamic Flag Image

Within the header a national flag should appear based on the language code contained in the incoming data. We will introduce a Dynamic Picture object to meet this requirement.

Complete the following steps to create the dynamic image.

Insert Picture

From the drawing toolbar select Picture  then draw an area on the header where you want the flag image to appear.

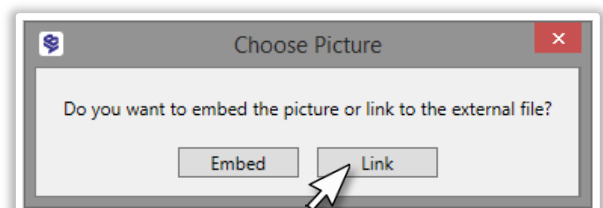


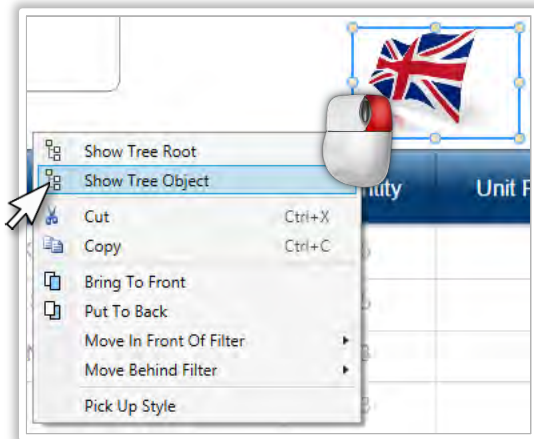
Choose Picture

Browse the file system for an image, this is for reference only as we will make the image path and name dynamic.

Link

To reference an external file for the image based on a variable click Link.



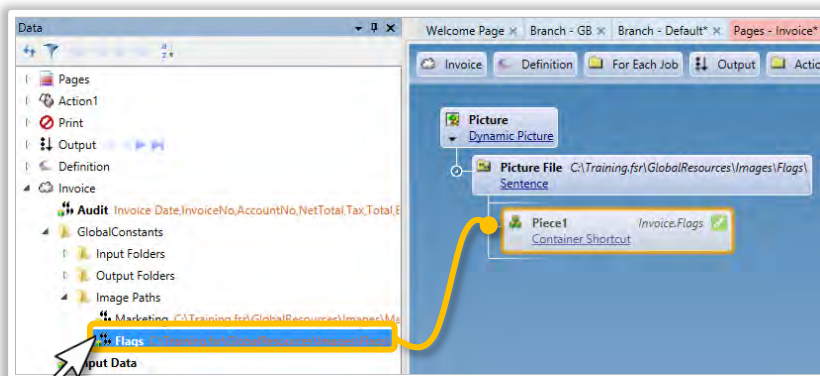
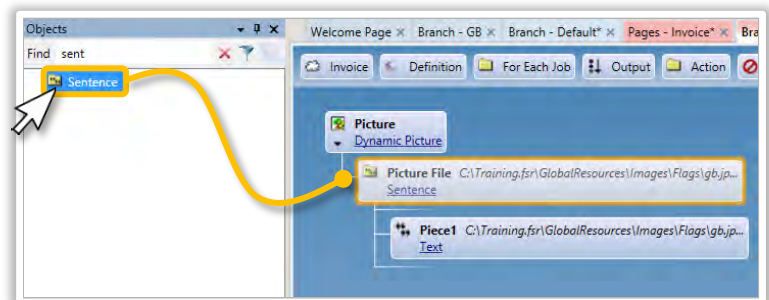


Show Tree Object

Right-click the picture and from the menu click Show Tree Object.

Sentence

In the Objects palette Find field type **sent** select **Sentence** and drag it onto the branch beneath the **Dynamic Picture** object.

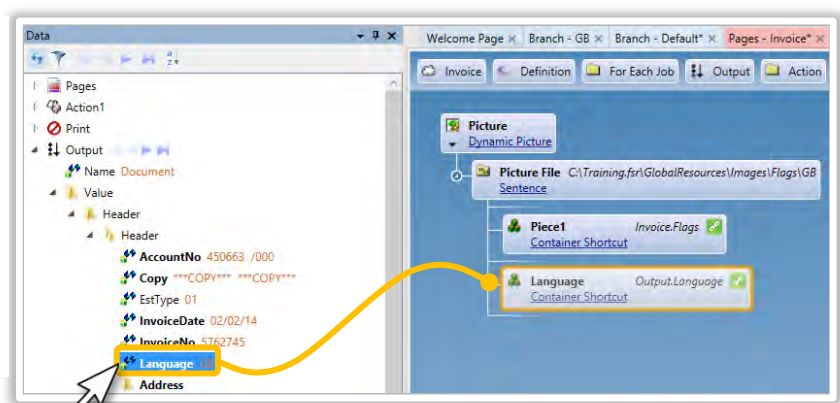


Path

Use the Image Paths from the GlobalConstants branch. From the Data palette expand Invoice > GlobalConstants > Image Paths and select **Flags** and drag it onto the branch beneath the **Sentence** object

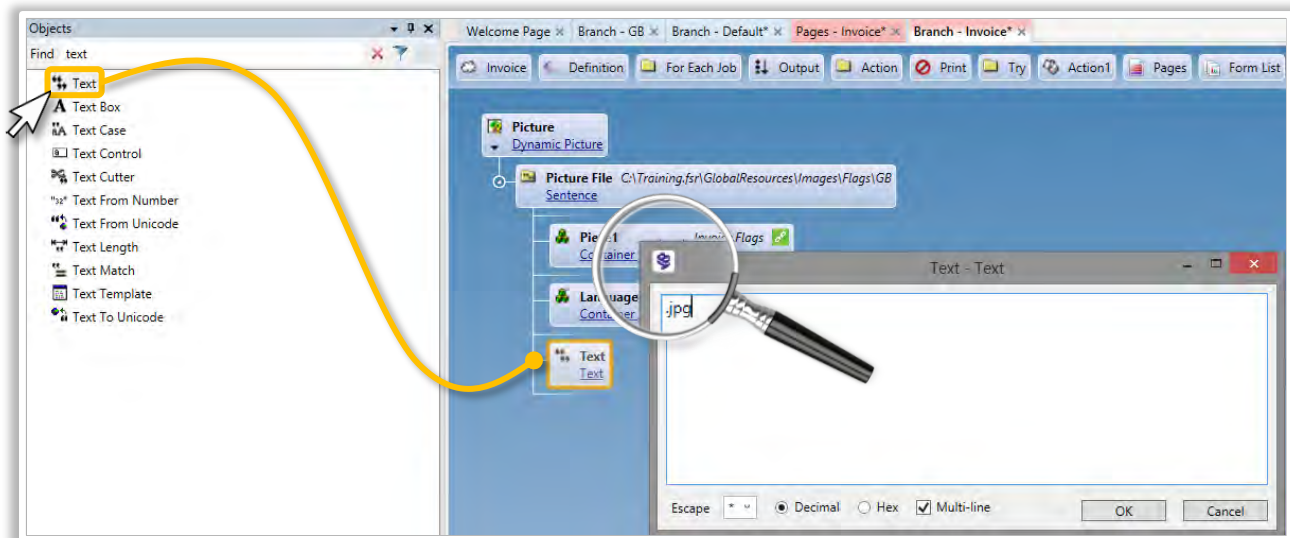
Filename

From the Data palette expand Output > Value > Header > Header and select **Language** and drag it onto the branch beneath the **Container Shortcut** object



File Extension

To complete the file path add the image file extension. In the Objects palette Find field type **text** select Text and drag it onto the branch beneath the Container Shortcut. Click **o** to open the Text dialog box and type **.jpg**. Click OK



Step 9.4 QR Barcode

The QR Barcode in the header is controlled by the marketing department where they have the freedom to change the content of the barcode based on the Establishment Type by means of a lookup file.

The format of the lookup file is illustrated below.

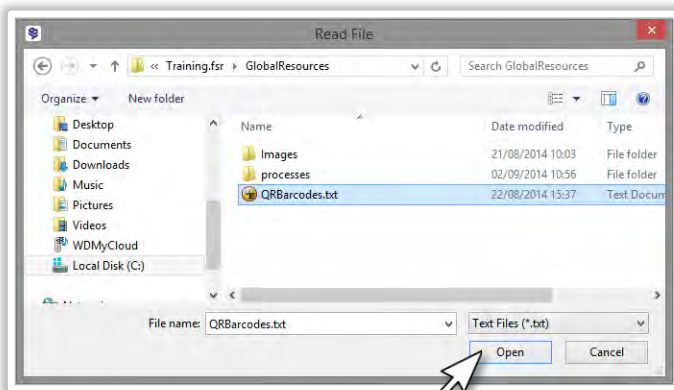
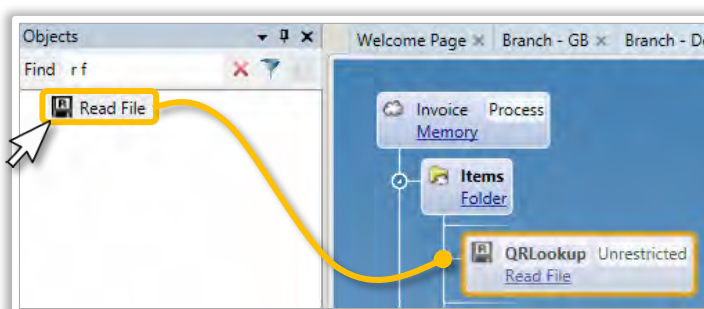
```
01=http://www.bottomline.co.uk/document_management/sales-order-management.html
02=http://www.bottomline.co.uk/document_management/transform.html
03=http://www.bottomline.co.uk/transform-logistics/index.html
04=http://www.bottomline.co.uk/document_management/transform-dynamics-ax.html
05=http://www.bottomline.co.uk/document_management/transform-filer.html
```

To create the barcode lookup complete the following steps.


Read Lookup File

In the Objects palette Find field type **r f** select **Read File** and drag it onto the branch beneath the **Items Folder**.

Press F2 and rename the object from Read File to QRLookup




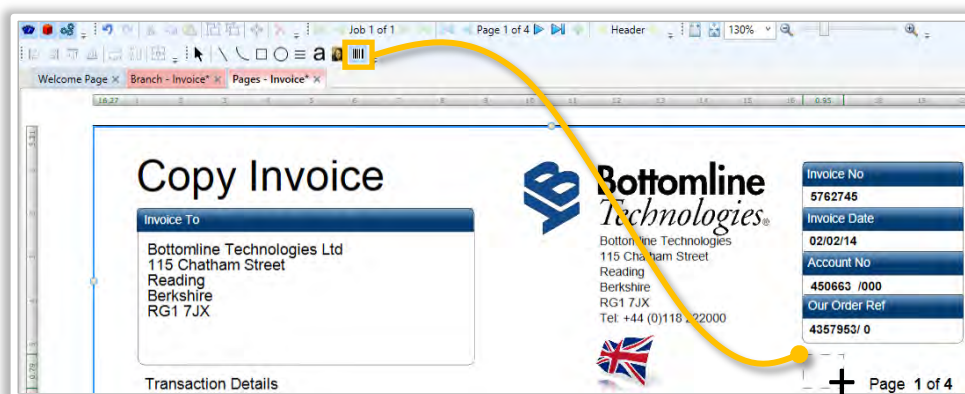
Browse

Click  Read File icon to open the Read File browse window.

Browse the file system to locate the QRBarcodes.txt and click Open.

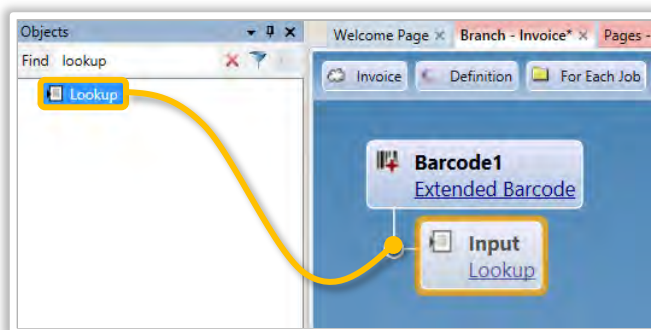
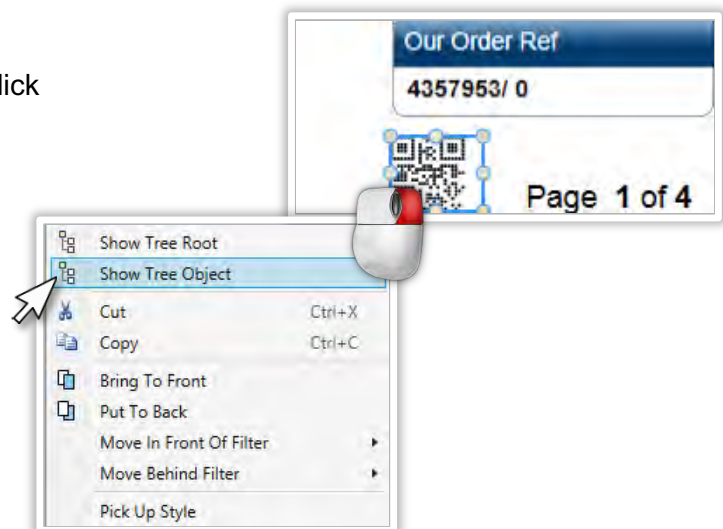
Add Barcode

From the drawing toolbar click  Barcode and then draw an area on the header where you want the barcode to appear.



Show Tree Object

Right-click the Barcode and from the menu click Show Tree Object.

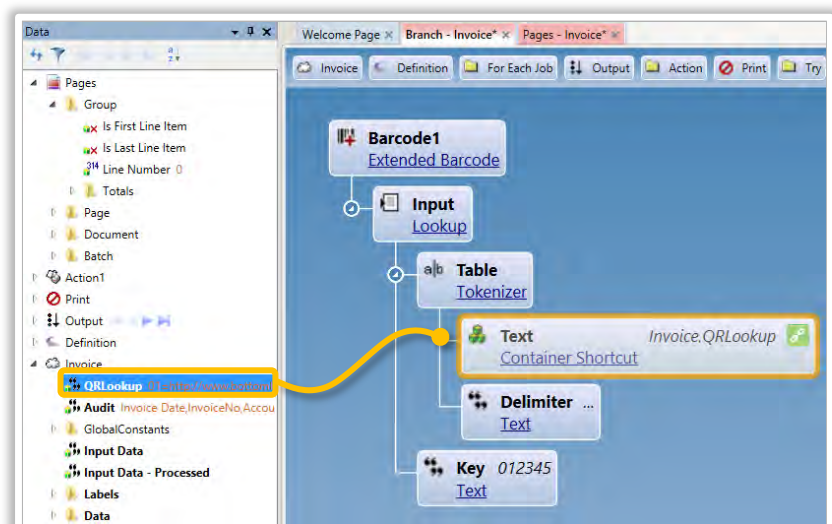


Lookup

In the Objects palette Find field type **lookup** and drag the lookup onto the branch beneath the Extended Barcode.

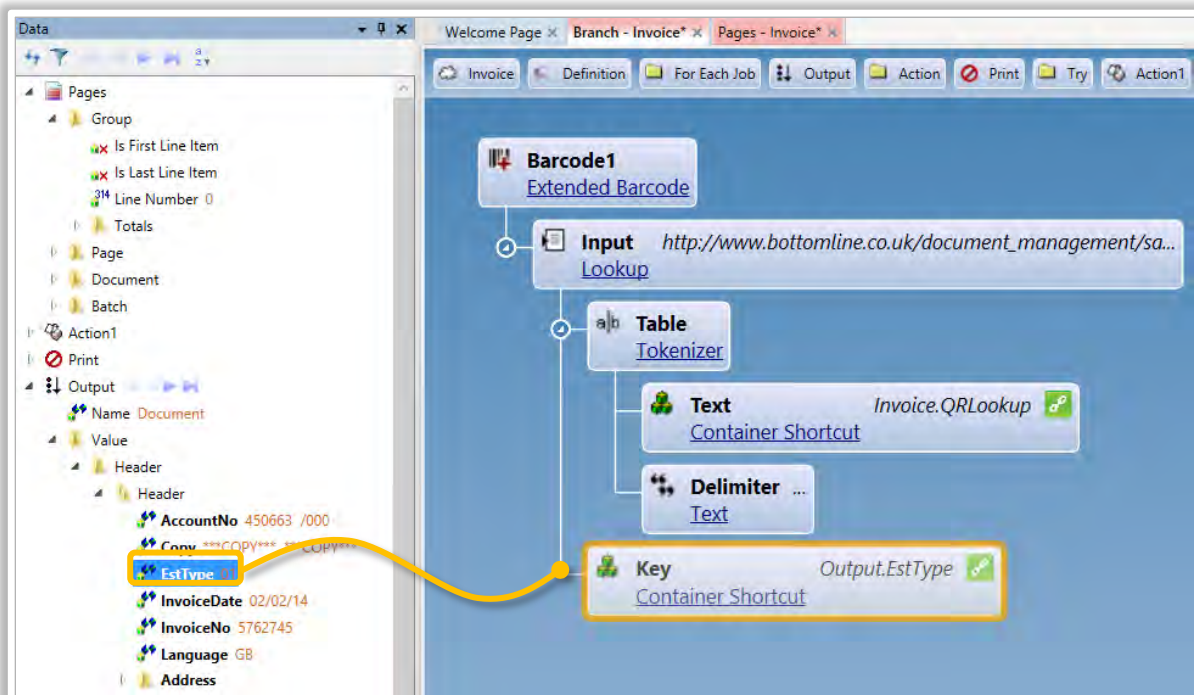
Table

On the branch Expand Lookup > Tokenizer, from the Data palette expand Invoice and drag QRLookup onto the branch over the Text object. This reads the external text file consisting of tokens separated by the delimiter sequence.



Key

To complete the Lookup add the EstType as the key. From the Data palette expand Output > Value > Header > Header select EstType and drag it onto the branch over the Key **Text** object.



The result of the lookup is presented on the Lookup object. The example above illustrates what is returned when the EstType is 01.

01 = http://www.bottomline.co.uk/document_management/sales-order-management.html

Small Header

Now that you have learned the techniques to create the header form used on the First and Single pages you need to replicate these to create the Small Header used on the Continuation and Last pages.



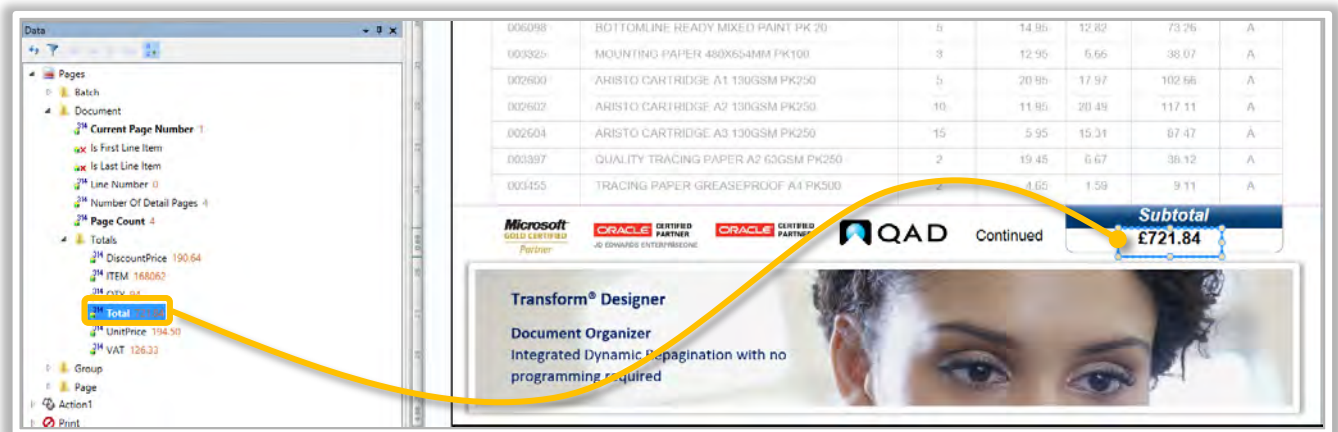
Step 10 Map Footer Data

This is the final step in the creation of the Invoice layout, we need to add subtotals on the first and continuation footers and the totals on the last footer.

Subtotal

The Document Organizer can create running totals and subtotals of any repeating numeric data item in a document, which are available in the Data Palette allow you to map them to your form. The Document Organizer adds up the values for each of the sortable table rows as it draws them on the form.

Expand the Document Organizer within the Data Palette and Document then within Totals select Total and drag it onto the SmallFooter form, as this is used on First and Continuation Page types.

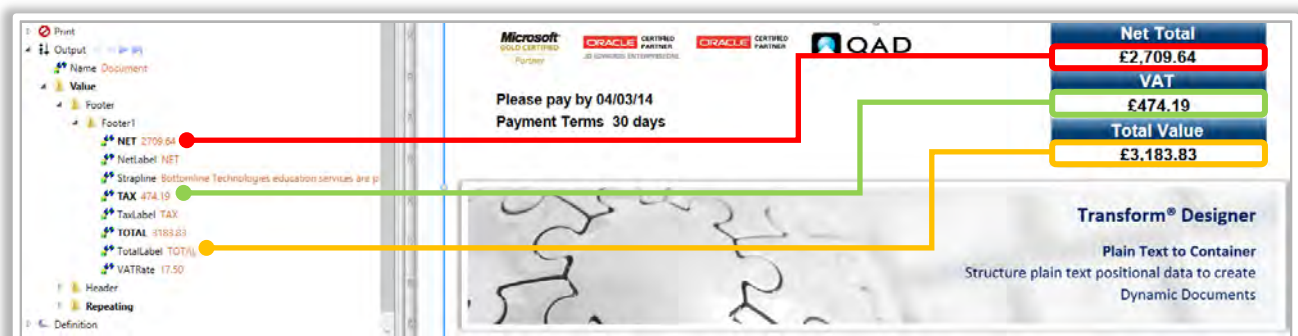


You could also add the same value to the SmallHeader form used on the Continuation and Last page layouts if you want carried forward.

Spend some time exploring the values available within the Document Organizer as using the data generated can save extensive coding.

Map Totals on Last Page

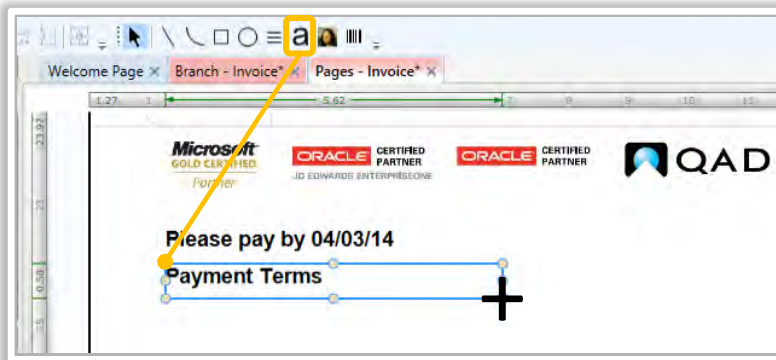
Use the navigate buttons to navigate to the last page and then map the required footer elements onto the Footer form. From the data palette expand Output > Value > Footer > Footer1 and map the fields as illustrated below



Step 11 Payment Terms (Date Difference)

Part of the requirements is to display the payment terms on the last page. The invoice date and payment date are in the original source data but we need to add the payment terms but using the date difference object.

Complete the following steps to add the Payment terms.

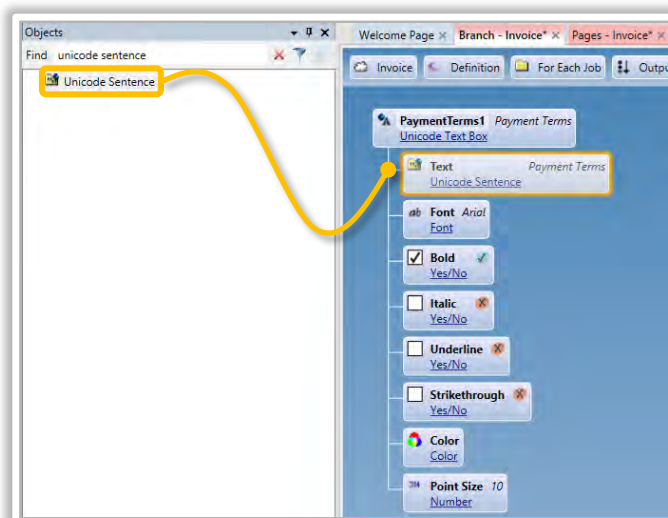
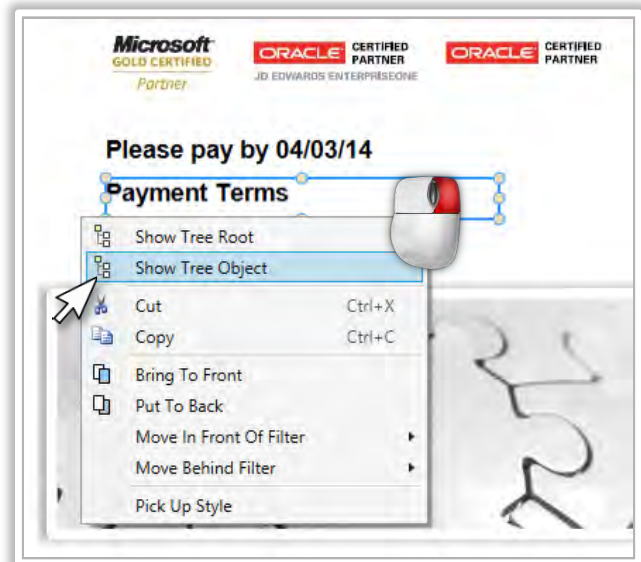


Text Box

From the drawing toolbar select text box **a** and draw the bounding box where you want the Payment Terms to appear. Then from the language labels select Payment terms and add it to the text box.

Show Tree Object

Right-click the Bounding box and from the menu click Show Tree Object.

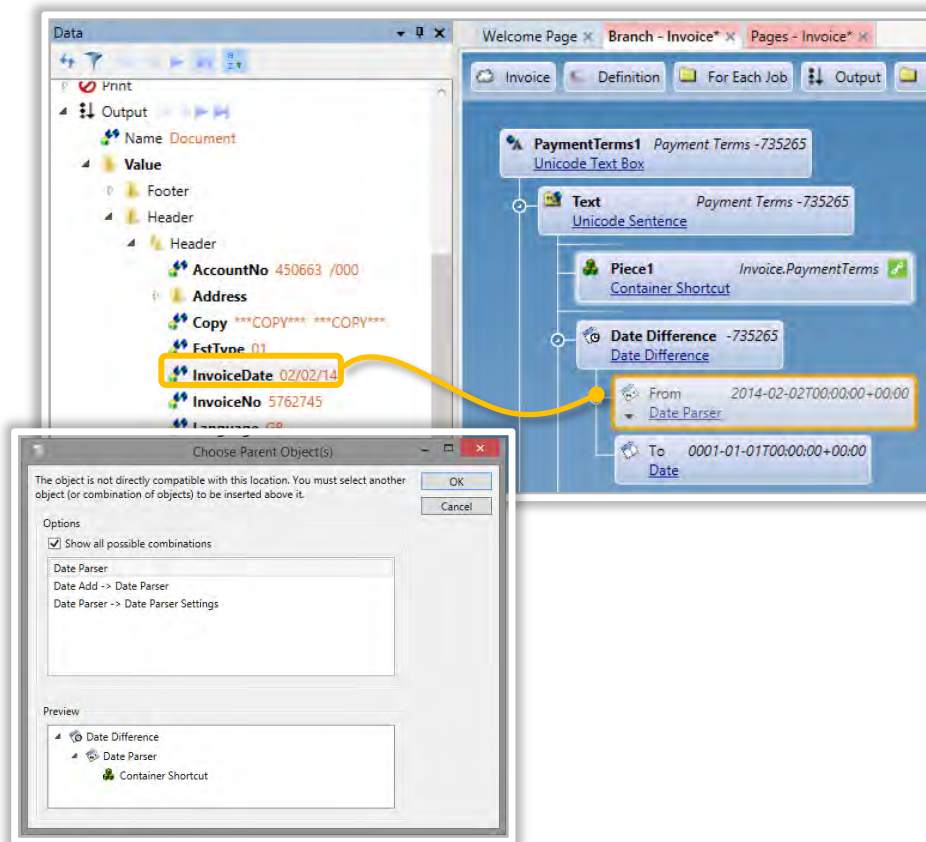
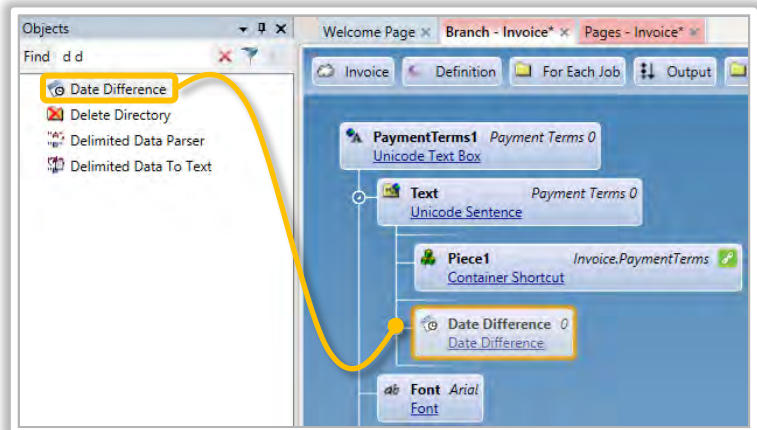


Unicode Sentence

In the Objects palette Find field type **u s** select **Unicode Sentence** and drag it onto the branch over the **Unicode Text** object

Date Difference

Expand the Unicode Sentence object to reveal the child objects. From the Objects palette Find field type **dd** select **Date Difference** and drag it onto the branch beneath the Container shortcut.

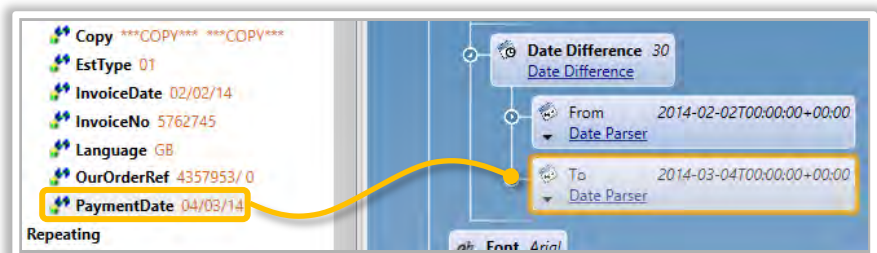


From Date

From the Data palette select Output > Header > Header and drag **InvoiceDate** onto the **From Date** then from the Choose Parent Objects window select **Date Parser**

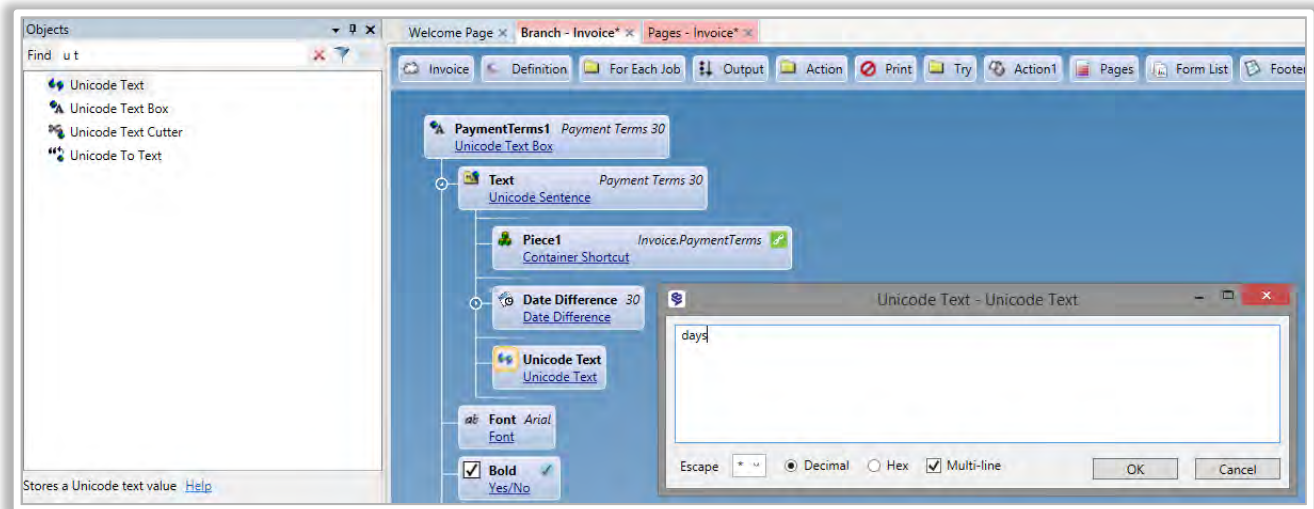
To Date

From the Data palette select Output > Header > Header and drag **PaymentDate** onto the **To Date** then from the Choose Parent Objects window select **Date Parser**



Unicode Text

From the Objects palette select **Unicode Text** and drag it onto the branch beneath the **Date Difference** object in the **Unicode Sentence**. Click the icon and type <space>days.

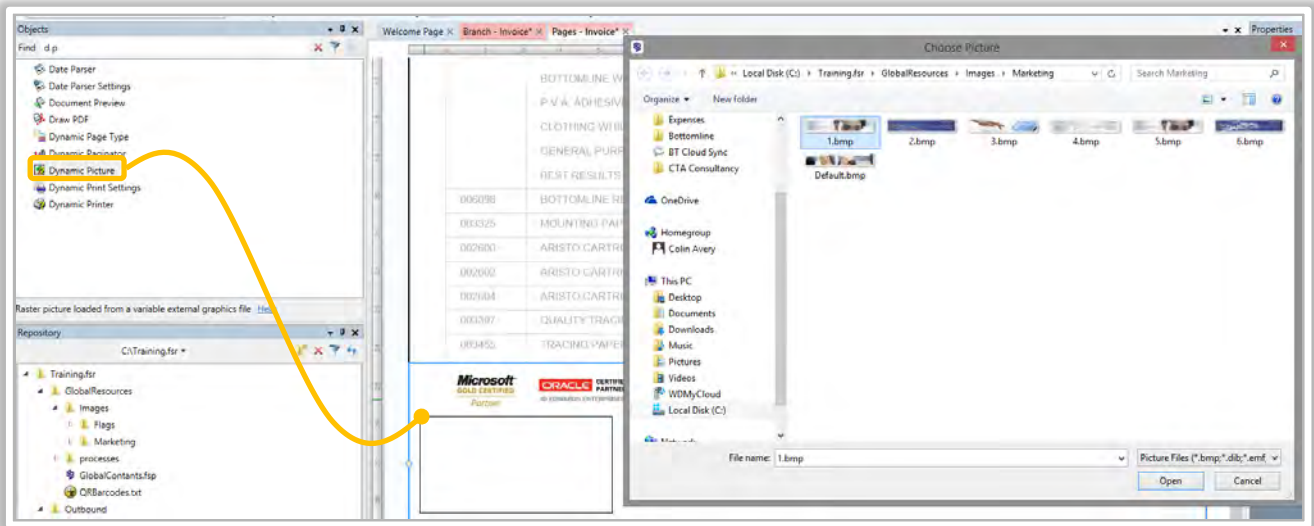


Check the result by opening the Pages view.

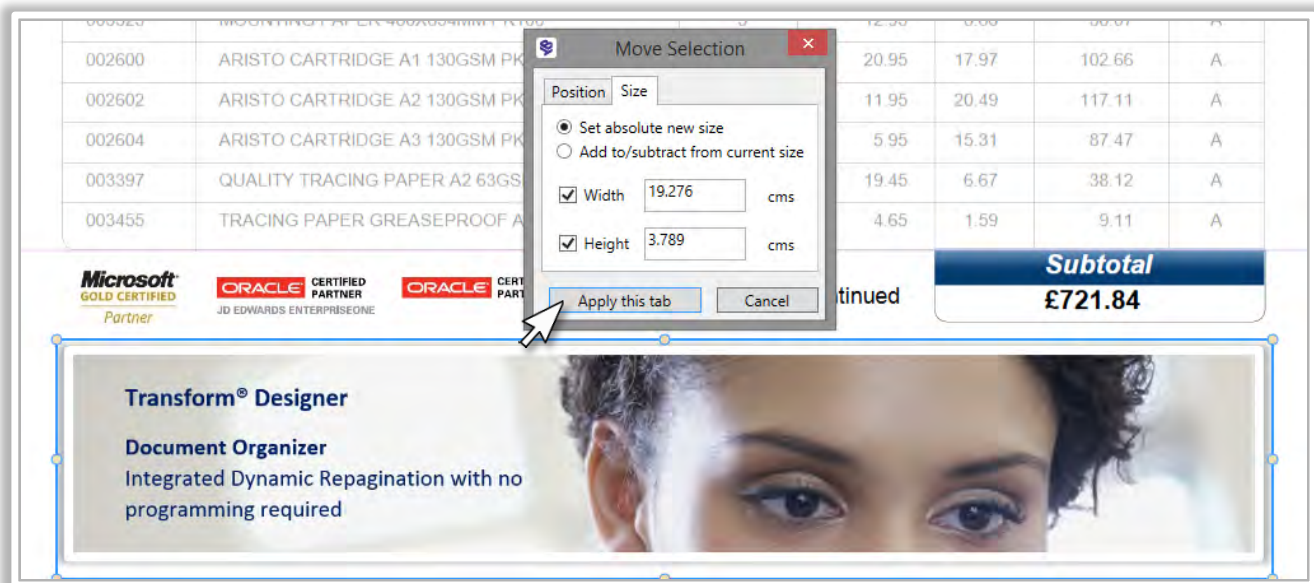


Step 12 Footer Banner Dynamic Picture

To allow the marketing department to add messages to each page of the invoice the project needs to use the dynamic Picture object to place a different image based on the page number. From the Objects palette Find field type **d p** select **Dynamic Picture** and draw an area on the footer where the banner image will be placed, this will open the Choose Picture dialog window browse to the Marketing images folder and select **1.bmp** and click Open.



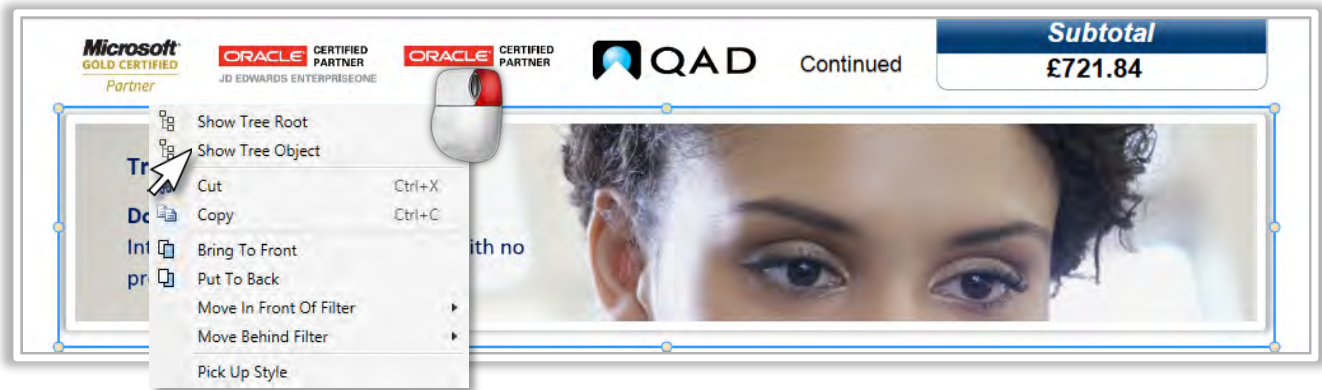
Resize the picture bounding box, click CTRL+M to open the Move Selection window, click the Size tab and change the width and height as illustrated below.



Please remember to click the Apply this tab button to commit the change.

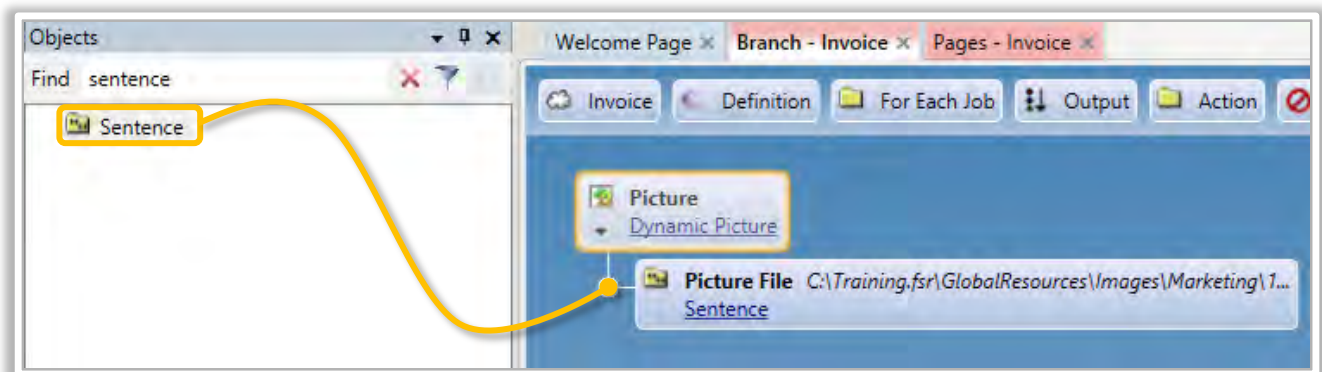
Show Tree Object

Right click the picture and from the menu select Show tree object.



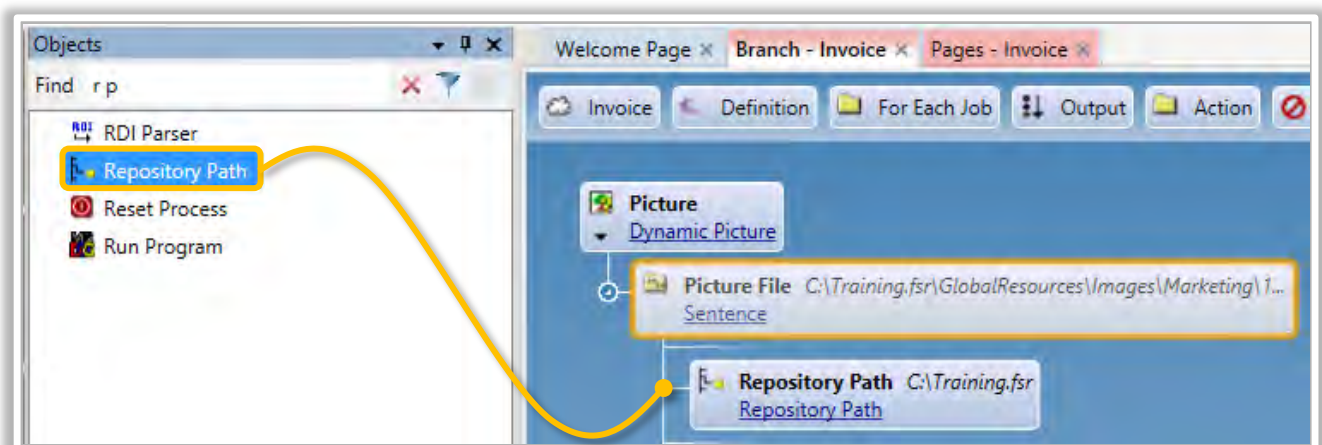
Sentence

From the Object palette select Sentence and drag it over the Text object beneath the Dynamic Picture.



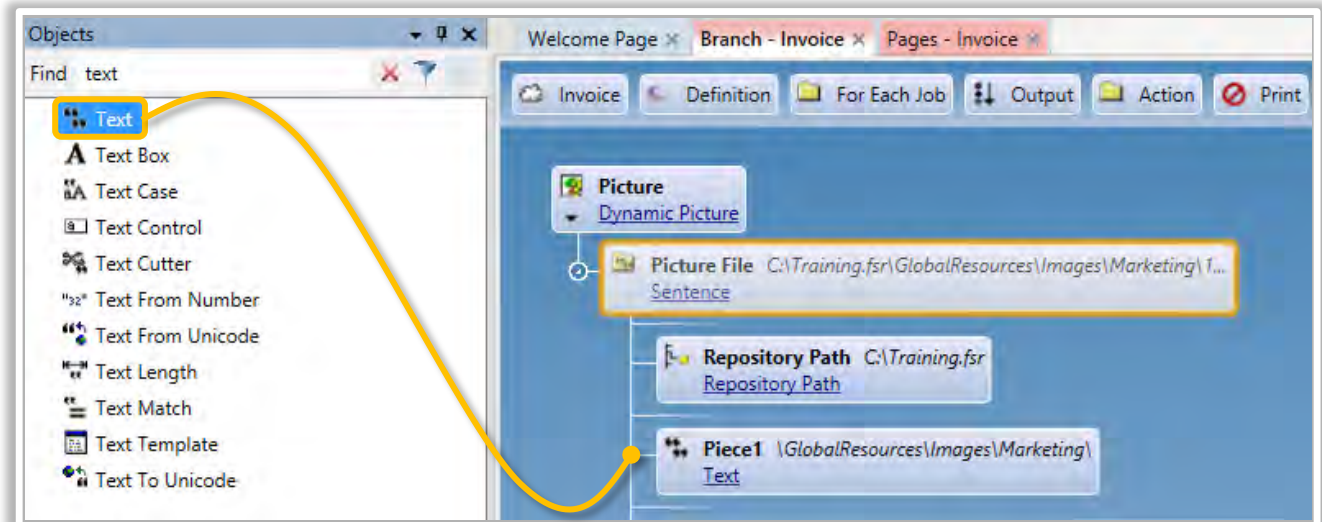
Repository Path

From the Objects palette Find field type **r p** select **Repository Path** and drag it onto the branch beneath the **Sentence** object.



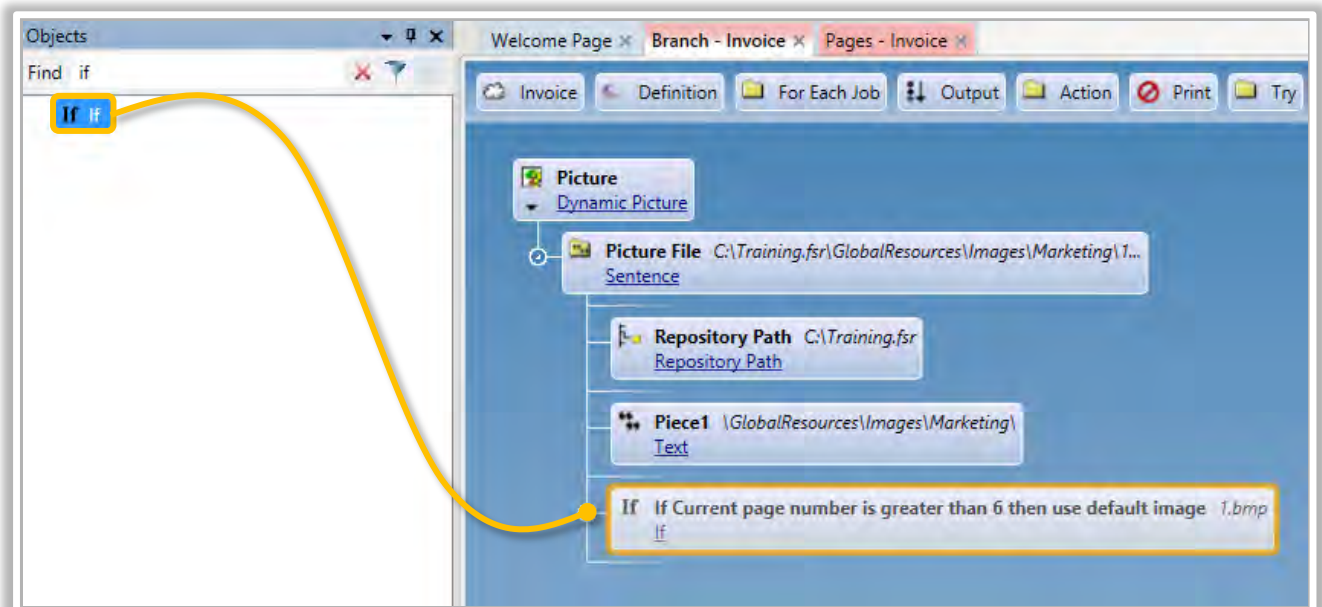
Text Object

To complete the path to the image folder add a **Text** object, type text in the Find field within the Objects palette select and drag onto the branch beneath the Repository Path object. Click the text object and complete the path remember to add the backslash \.



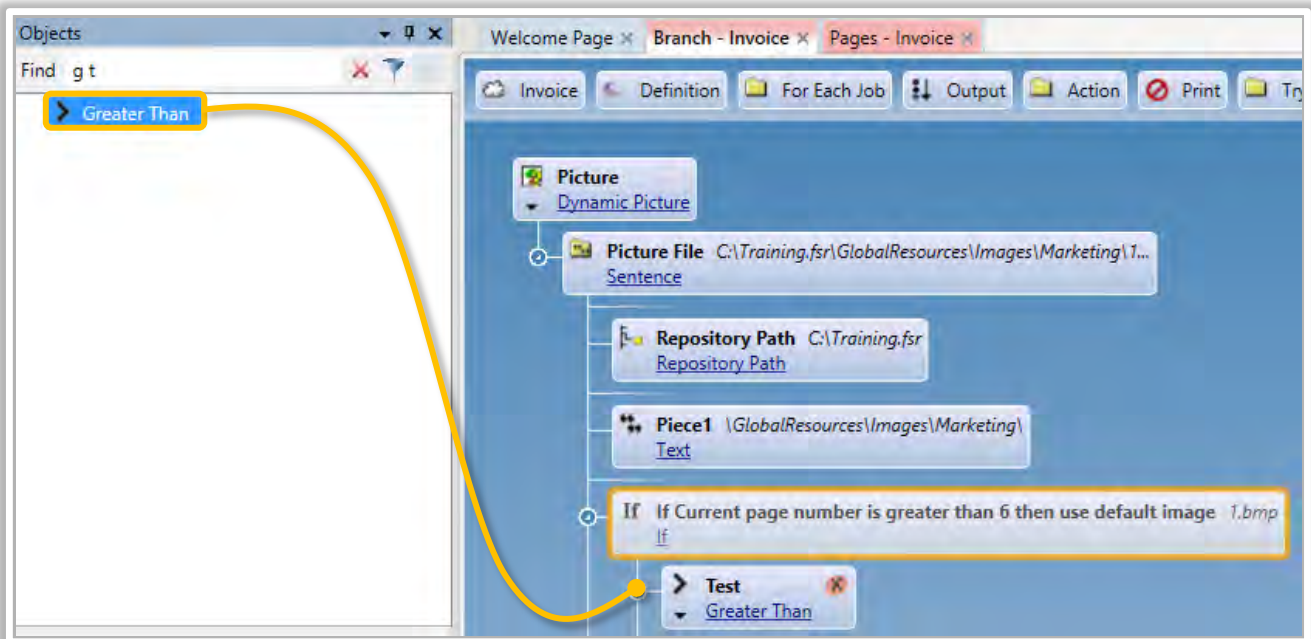
If

From the Objects palette select **If** and drag it onto the branch beneath the Text object and rename it to something meaningful.



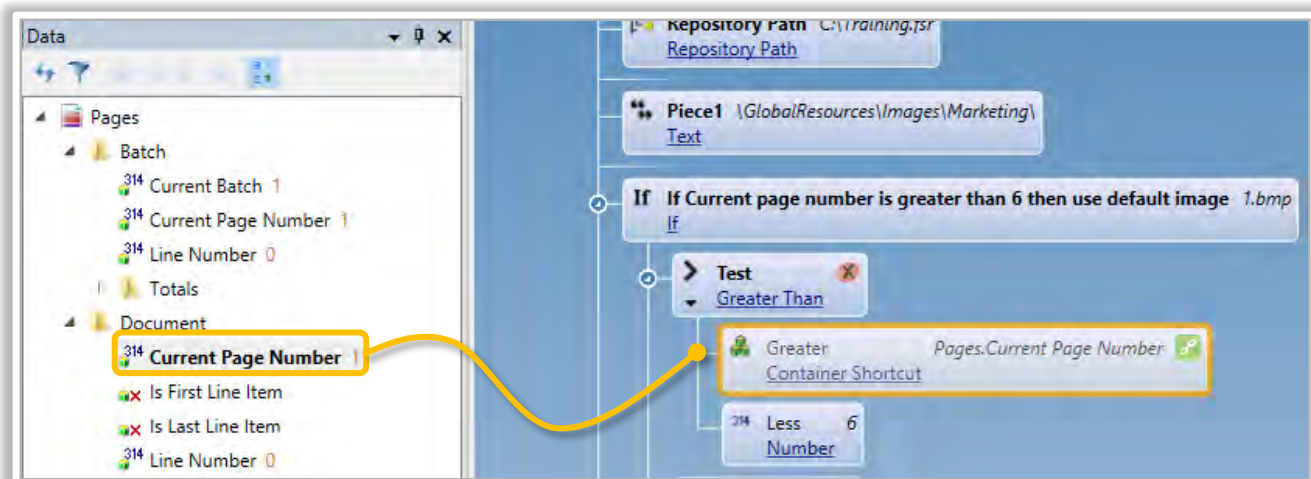
Greater Than

From the Objects palette Find field type **g p** select **Greater Than** and drag it onto the branch over the Test beneath the If object.



Current Page Number

As we only have six marketing message images any Invoices that span across more than six pages should display a default image. Expand the **Greater Than** object then from the Data palette Pages > Document drag the **Current Page Number** element onto the Greater number.

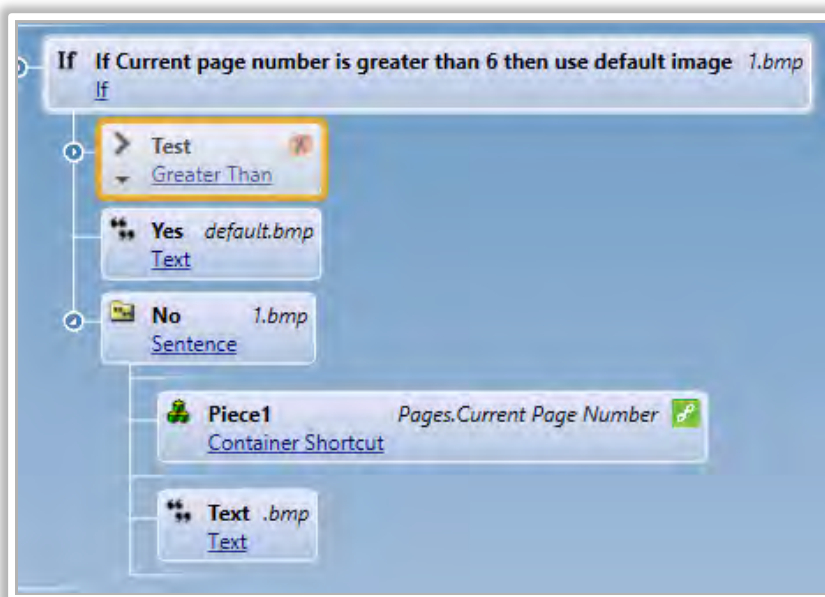
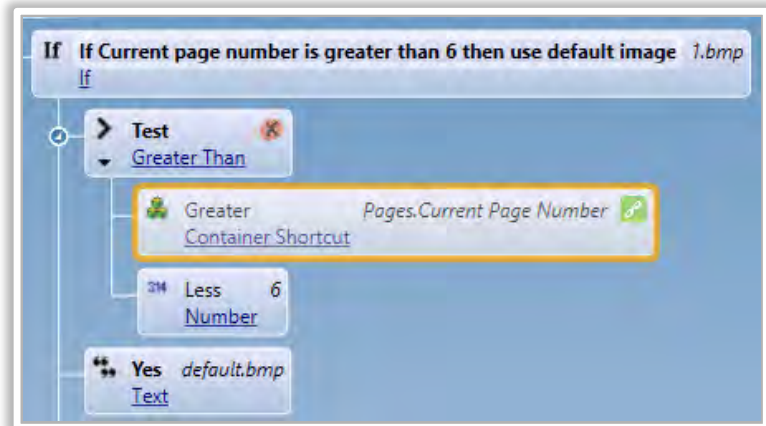


Click the Less **Number** object and change the value to **6**.

What this means is that the Boolean will return true if the Current Page Number is greater than 6. To perform a less than test you place the variable on the Less Number and the constant on the Greater Number.

Yes

If the Boolean returns true then the current page number is greater than 6 so we need to use the default image. Click on the Yes **Text** object and type **default.bmp**.



No

From the Objects palette select a Sentence object and drag it onto the No text object. Expand the Sentence and from the Data palette select Pages > Document and drag Current Page Number so that it becomes Piece1 of the sentence. Then add a Text object and enter the file extension **.bmp**

Project Checklist

1. ~~Repository Design~~
2. ~~Create Local Repository~~
3. ~~Language Labels~~
4. ~~Global Resources~~
5. ~~Plain Text Template Wizard~~
6. ~~Plain Text to Container~~
7. ~~Modify Branch Structure~~
8. ~~Dynamic Branch Shortcut~~
9. ~~Document Organizer~~

10. Synchronize and Deploy

Synchronizer Application

Synchronize with Deployment Server

Deploy to Runtime Server

Package Watch List



Synchronize and Deploy

The project is now complete and ready for deployment. Save the project in the following directory
Training.fsr\Outbound\processes.

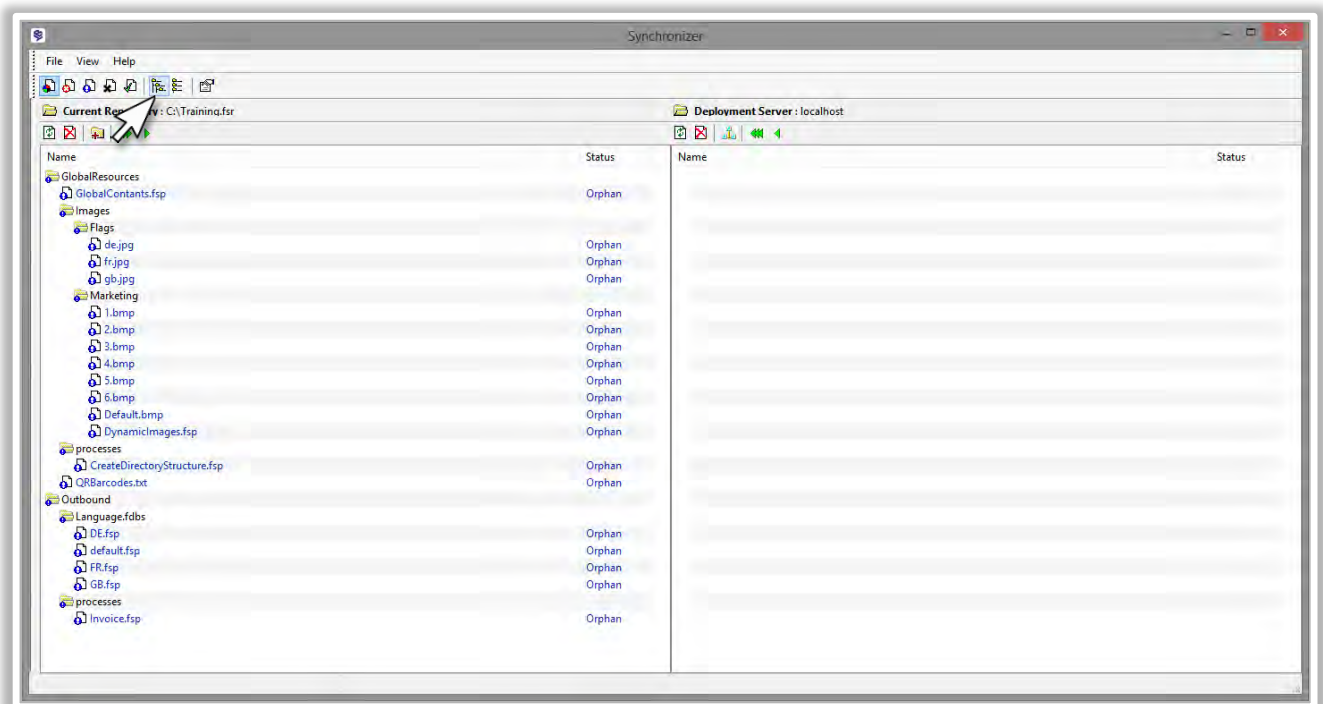
Step1 – Open Synchronizer Application

The Transform Synchronizer Application is a standalone desktop application, which can be opened through programs menu and also launched through the Transform Designer menu bar **Tools > Synchronizer**.



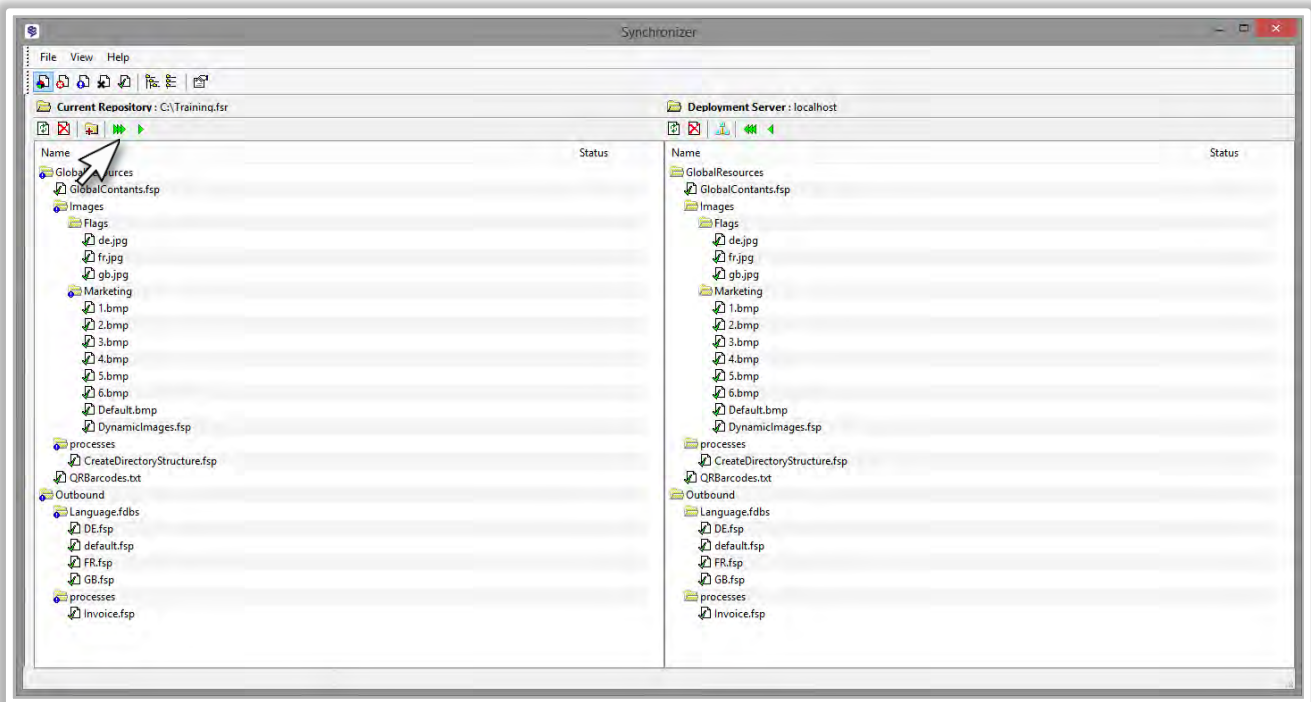
Step 2 – Expand All

Once open click expand all button on the toolbar to open all the Packages and folders.



Step 3 – Copy Visible

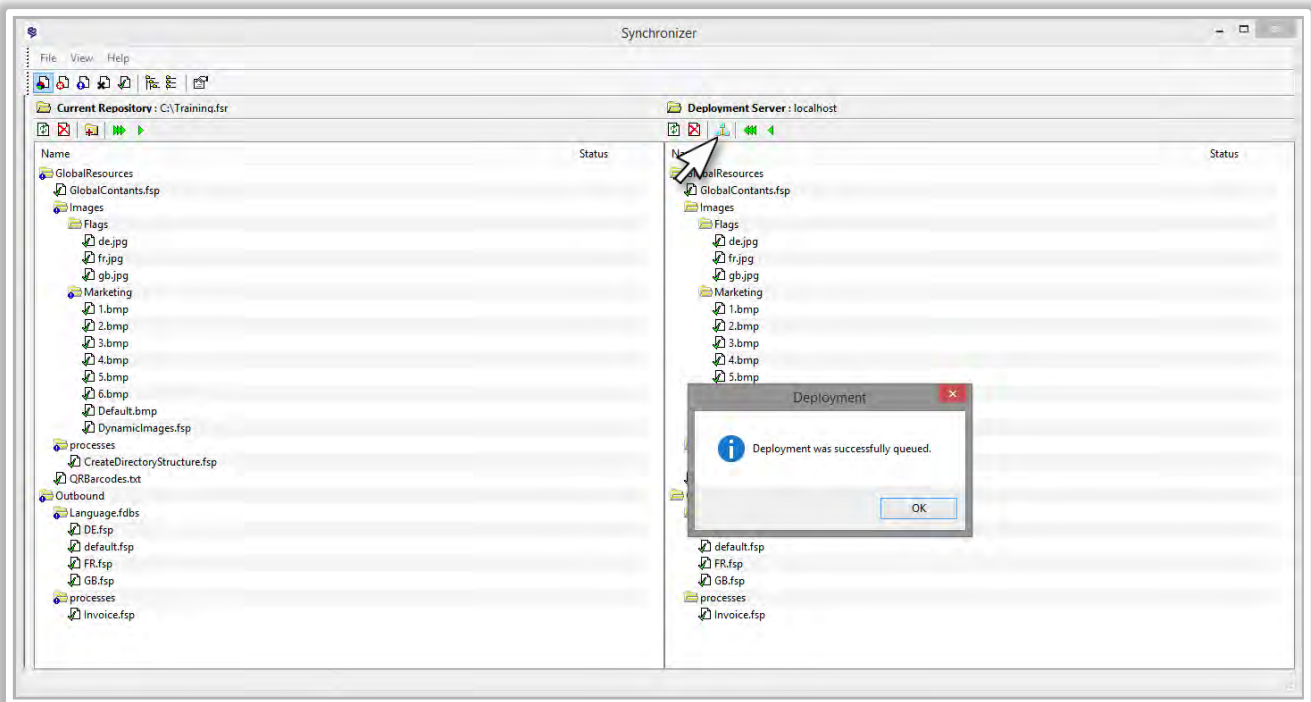
Click the Copy Visible button on the Content Source Window to Synchronize the local repository on the left pane with the Deployment Server on the localhost.



Once synchronized the content is presented with green ticks to demonstrate that files match.

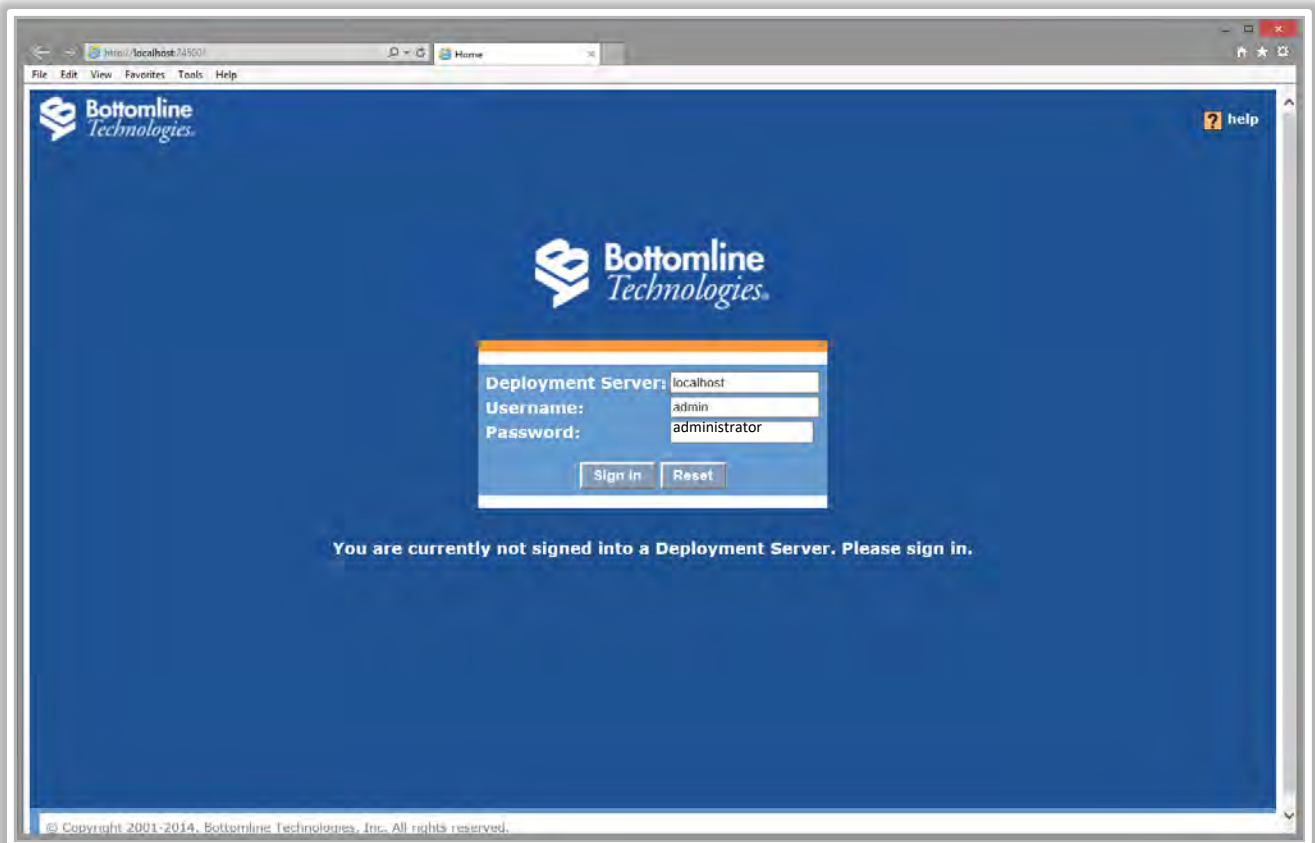
Step 4 – Deploy

To promote projects to the Runtime server attached to the deployment server then click the deploy button on the Deployment Server toolbar. You are then presented with a message box indicating that the Deployment was successfully queued.

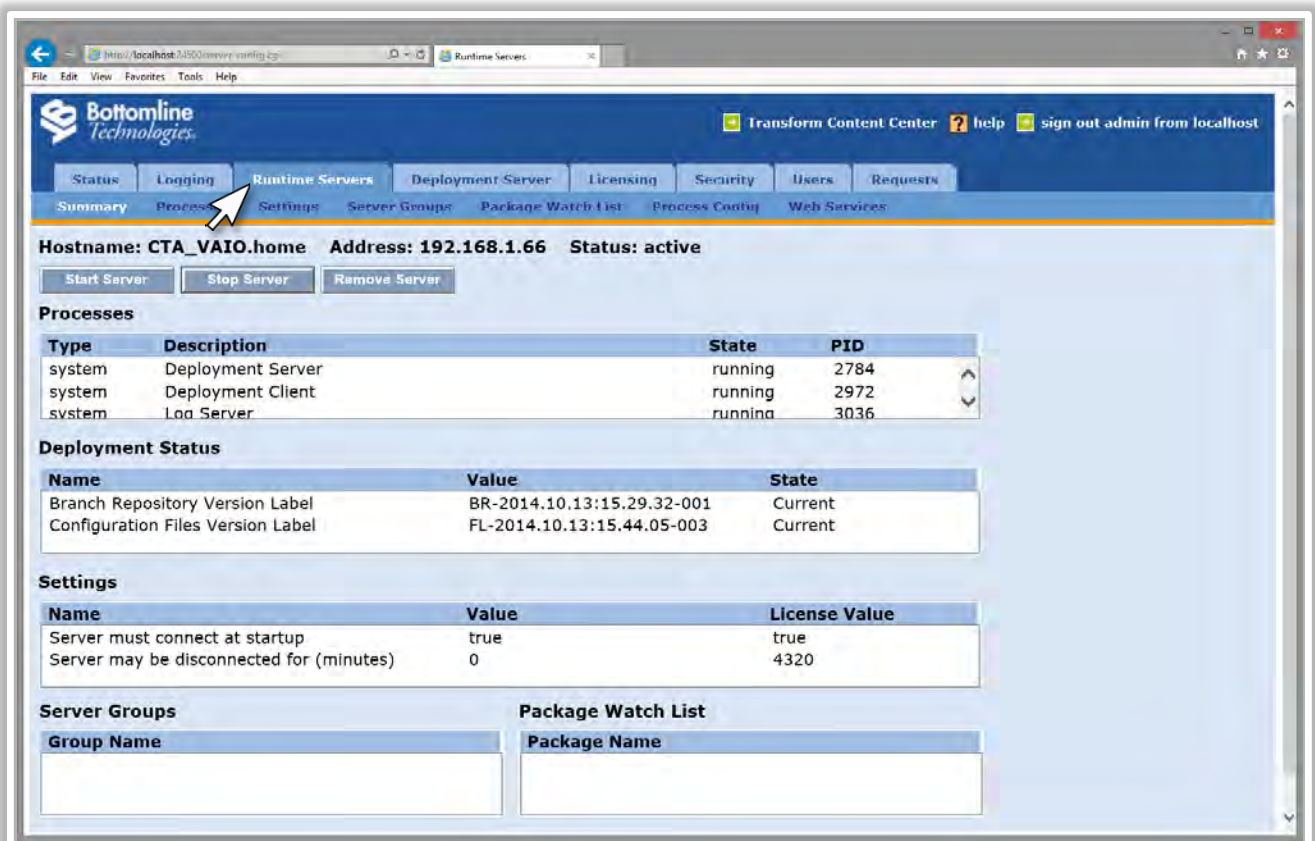


Step 4 – Package Watch List

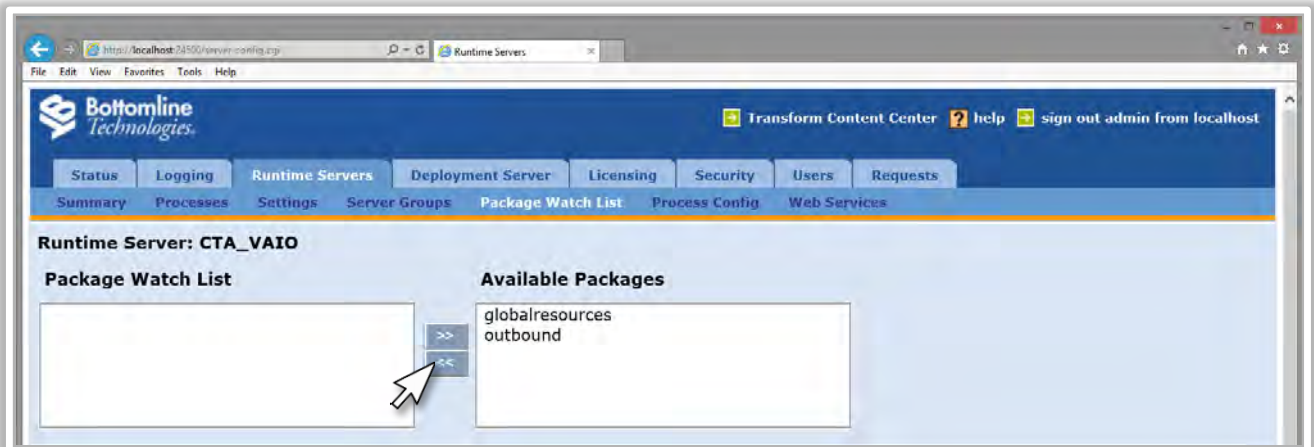
Open the Transform Administrator web application and login to the Deployment server



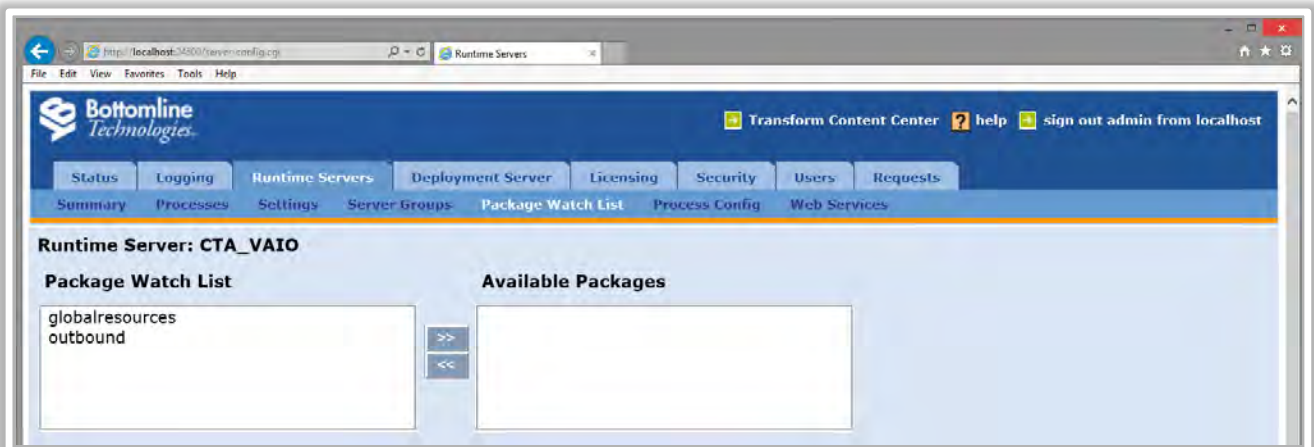
Select the Runtime Server Tab.



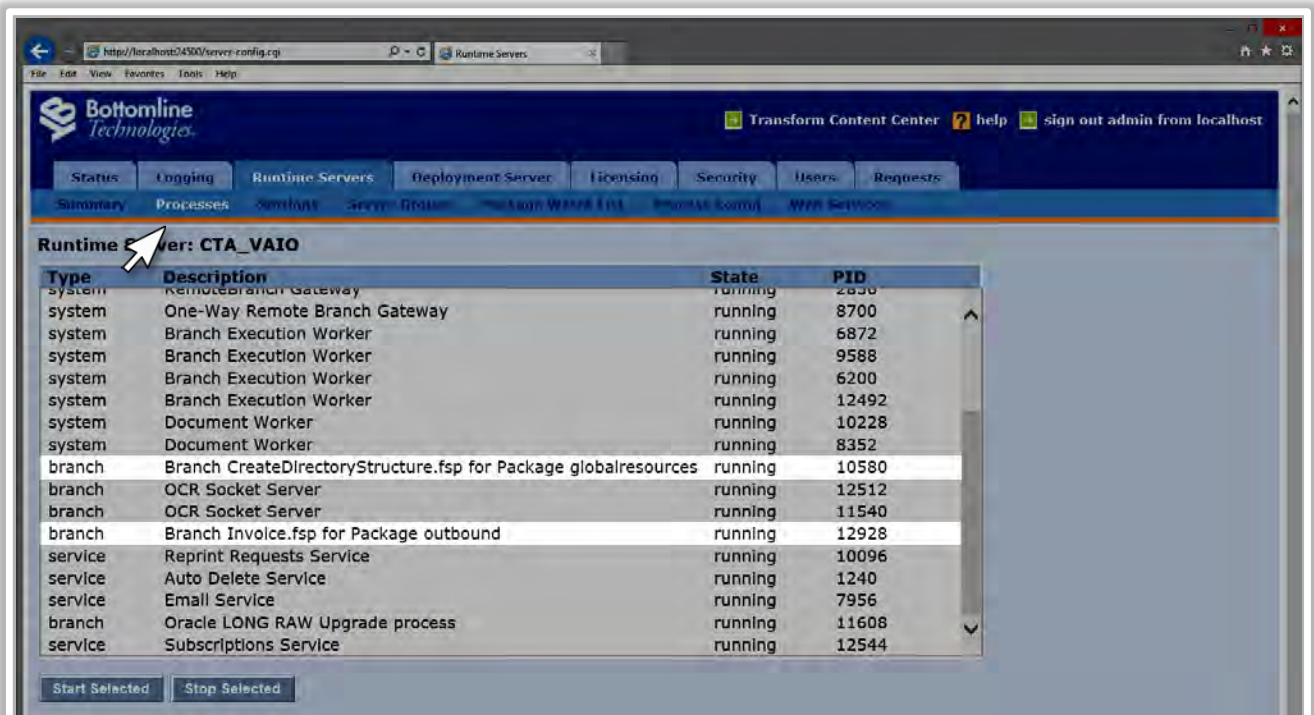
Select Package Watch List from the sub menu.



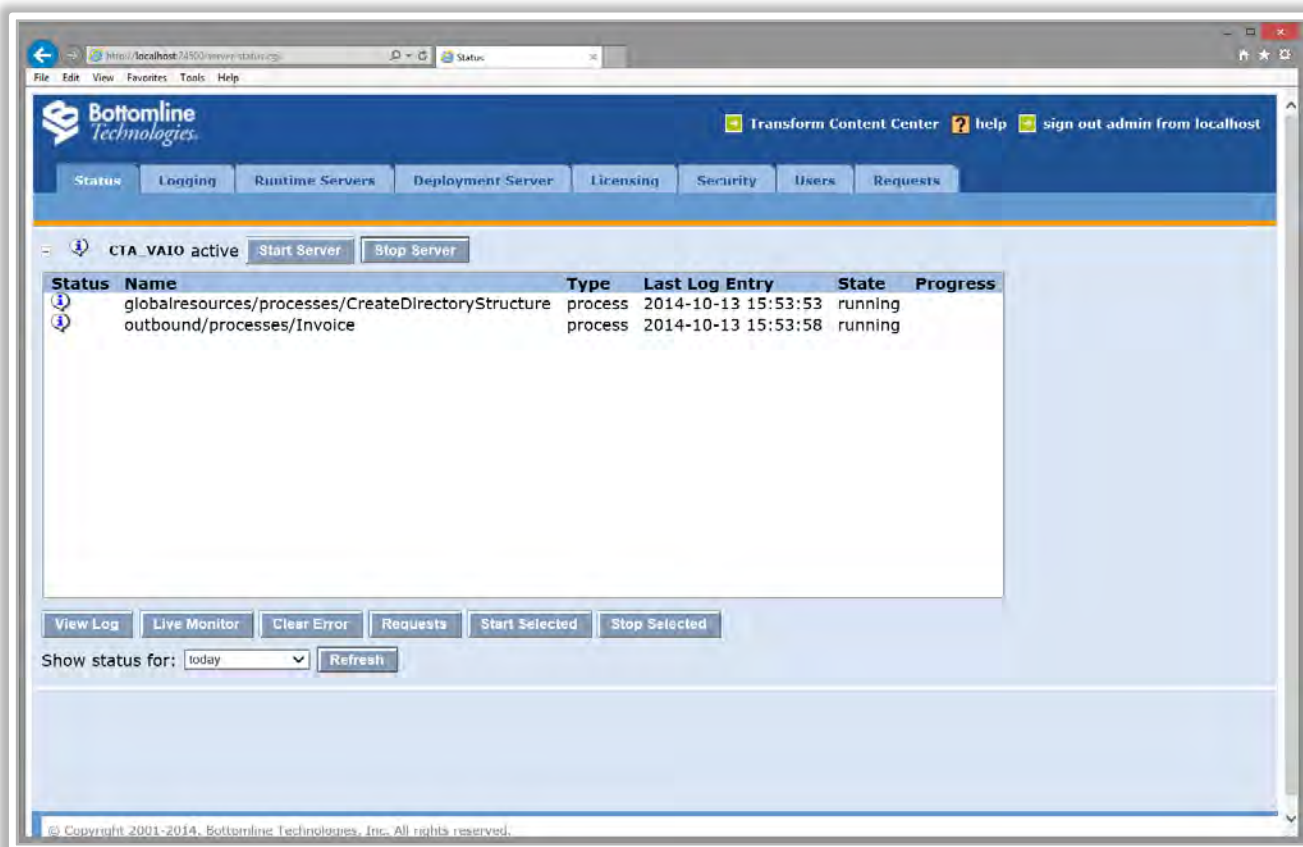
Select each of the Available Packages and use the << to move them into the Package Watch List.



Click on the Processes to show the active processes on the Runtime Server.



Click on the Status tab to view the status of the active projects.



Test your Masterpiece

Now the project is deployed and active you can copy the original source file into the watch folder and witness the fruits of your labour.

PDF's created

Audit File created