



Version 1.0

# Transform<sup>®</sup>

*Training*  
*HTML Email Integration*





# Legal Statement

CTA Consultancy Ltd

Version: Version X.X

© Copyright 2025 CTA Consultancy Ltd All rights reserved.

Information contained in this document is confidential and is supplied for internal purposes between CTA Consultancy Ltd and ClientName. 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 CTA Consultancy Ltd shall not be liable for any consequences arising from the use of outdated or superseded versions of this document.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.

# Contents

Introduction .....	3
What will you learn .....	3
What you will NOT learn .....	3
Intended Audience .....	3
Conventions Used in This Manual .....	3
HTML template library .....	4
Email Development Best Practices .....	4
Email Design .....	4
Use Single Column Design .....	4
Set 600px As the Default Width .....	4
Keep Mobile Users in Mind .....	4
Every Email Client Is Different .....	4
Plan for Missing or Blocked Images .....	5
Use Email-Safe Fonts .....	5
Avoid Image-Only Emails .....	5
Don't Forget to Add an Unsubscribe Link! .....	5
Email Development .....	6
HTML .....	6
Know Your Framework .....	6
Make Sure You Include Comments in Your Code .....	6
Encode Special Characters .....	6
Keep Email File Size Under 100kb .....	7
Code for High DPI Displays .....	7
Include Preheader Text .....	7
Avoid Javascript, Flash, Forms and other Complex CSS/HTML .....	7
Use Cellpadding for Spacing .....	7
Test Your Email! .....	7
Images .....	7
Use Absolute Addresses for Images .....	8
Get Rid of Strange Spacing Around an Image .....	8
Don't Use Image Maps .....	8
Take Some Extra Time with Background Images .....	8
CSS .....	8
Use Inline Styles .....	8
Avoid Shorthand CSS When Possible .....	8

Get Used to Important .....	8
Get Comfortable with Media Queries .....	8
The Most Important Email Practice: Testing .....	9
Static HTML Template .....	10
Transform Static HTML Integration .....	11
Template.html .....	11
Read File .....	11
Memory .....	11
Replace Fields .....	11
Send Mail .....	11
Objects Required .....	12
Create Email Body .....	13
Create Email Body - Decision .....	14
Memory objects .....	15
CurrencySymbol – Lookup object .....	16
Attempt to load HTML template - Catch Try .....	19
On Error - Action List .....	20
Log error – Log Event .....	21
Update EmailBody using plain text – Copy Unicode Text object .....	21
From Text Template .....	22
Subject – Unicode Sentence .....	24
Body – Text Template .....	25
Update Field_EmailSubject – Copy Unicode Text .....	28
Update Placeholders in HTMLTemplate – Action List .....	29
Placeholder mappings .....	30
AMPM – Copy Unicode Text .....	31
OrderNo – Copy Unicode Text .....	32
AccountNo – Copy Unicode Text object .....	32
OrderDate – Copy Unicode Text object .....	32
Currency – Copy Unicode Text object .....	33
Total – Copy Unicode Text object .....	33
Commit EmailBody – Copy Unicode Text object .....	34
Sample Static Email Output .....	35
Dynamic HTML Template .....	36
Challenge .....	36
Resources .....	37

## Introduction

The purpose of this document is to support the training session which demonstrates how to create and deliver HTML Emails using Transform.

### What will you learn

During the session you will learn the following techniques, all of which are developed on the Transform branch editor.

- Read a file from HTML template library
- Catch Try error trapping
- Replace function
- Text template
- Send mail settings

We make use of a Dynamics AX SalesConfirm\_Report project to apply the HTML template integration.

### What you will NOT learn

In the session we do not cover how to create HTML email template, it assumes that you have the necessary knowledge or access to another development team who can build the HTML templates for you.

We will supply you with some sample templates, which should **ONLY** be used for the sole purpose of the training to understand how to integrate them within the Transform process.

### Intended Audience

This manual is intended for Transform developers who maintain outbound documents, which require HTML email delivery. It assumes that you have experience with the Transform Designer tool and are comfortable with modifying projects on the branch editor.

### Conventions Used in This Manual

This manual uses design elements and conventions to help you prioritise and reference the information it contains.



**Important**



**Note**



**Tip**

# HTML template library

The HTML email layout is NOT developed in Transform, rather read from a library at runtime. This allows the HTML templates to be developed outside by web or Email developers.

The following information is an extract from Email on Acid

<https://www.emailonacid.com/blog/article/email-development/email-development-best-practices-2/>

## Email Development Best Practices

Email development can be a daunting task. Many new email developers, or web developers who are new to email, find that the complex and multi-layered email client ecosystem causes endless headaches; a fix for one client breaks their email in three others, or a simple float functions in only half of the email clients available.

The best practices that we outline below cover both email design and email development. Keeping these best practices in mind can save you hours of hair pulling down the road.

Even with these best practices, don't forget to test your email. Finicky email clients are just one of the many reasons why you should test your email code every time. With Email on Acid, you can see what your design will look like in more than 70 clients and devices.

## Email Design

### Use Single Column Design

Keep the email design simple to make life easy! A single column design is sufficient for most emails and will help the design look good on mobile devices. It's also easier for your readers to scan a single column of material.

### Set 600px As the Default Width

We recommend that you keep your email's maximum width close to 600px. This width should give you plenty of space for content and will fit nicely on most web and desktop clients. You can scale it down to fit better on mobile screens using media queries or fluid design (see below).

### Keep Mobile Users in Mind

With the rise in popularity of mobile devices, some email designers have embraced "mobile first" design. This means that they design the email with mobile clients primarily in mind, and then make sure it also looks good on desktop. By putting mobile users first, designers hope to increase engagement and click-throughs on mobile devices. We recommend this approach especially for simpler emails like password resets, transactional emails and account updates.

### Every Email Client Is Different

When designing an email, keep in mind that it's going to be very difficult to achieve "pixel perfection" on every single client. Instead, try to achieve an email that maintains your branding while being easy to read (and click) on all email clients.

The only way to know how your email will look across the multiple clients is to test it. With a platform like Email on Acid, you can view your email in more than 70 email clients and devices, so you know how your email will look before you hit "send."

## Plan for Missing or Blocked Images

Some clients will block images by default, and some users will change their settings to block images so that they can use less data. If you rely on images to communicate your message, your subscribers may miss out if images aren't downloaded. This is why it's important to include descriptive alt text for your images and style the alt text to improve its appearance.

## Use Email-Safe Fonts

If you use Google Fonts, you may find that many clients don't support them. For this reason, it's important to have a good fallback font. Your fallback font ensures your design still looks good without custom web fonts.

You'll also want to make sure that you use a font-stack compatible with Outlook. One of the many quirks of Outlook is that an unrecognized font in the stack will cause it to fall back to Times New Roman. You can address this annoying bug using one of these four fixes.

Some of the fonts that are supported universally include:

- Arial
- Arial Black
- Comic Sans MS
- Courier New
- Georgia
- Impact
- Times New Roman
- Trebuchet MS
- Verdana
- Σψμβολ2 (Symbol)
- Webdings

MailChimp also has an excellent list of email-safe fonts.

## Avoid Image-Only Emails

If a subscriber has turned on image blocking, your whole message may be lost. If you must use text in an image, use styled alt text to make sure your message gets across if the image doesn't load.

Image-based emails are also very hard on your recipients' data plans and can be very difficult for the visually impaired. Use HTML text where possible, instead.

## Don't Forget to Add an Unsubscribe Link!

...And don't try to hide it. You don't want to email people who aren't interested in reading your emails. It's also illegal to leave an unsubscribe link out of a commercial email.

The unsubscribe usually appears in or below the footer. If you want to go for extra credit, set up a preference center, which allows subscribers to select what type of emails they receive, or how often they receive them. This can help reduce the number of unsubscribes you see.

## Email Development

### HTML

#### Use Tables When Possible

Forget divs and floats. While it may seem like you're coding in the dark ages of the internet, tables are the most reliable way to achieve a consistent layout. They also allow you to replicate something that many email clients otherwise don't allow: floats (okay, not really CSS floats). With tables, you can take advantage of the align attribute, which was the predecessor of modern CSS floats.

When using align="left", tables will stack on top of each other on smaller screens. This technique is the basis of responsive and fluid design. It works like this: You have two tables that are each 300px wide with align="left" inside the same container. If the screen is 600 or more pixels wide (as it would be for most desktop clients) then the tables will appear side by side. If the screen is only 400px wide, then the two tables will stack on top of each other. Nested tables are totally safe, so feel free to nest away.

You can also use colspan and rowspan, as long as you count your columns and rows carefully.

However, watch out for empty TDs, as some email clients don't handle these as you'd expect. Usually this issue can be fixed by adding " " or non-breaking space character. You can control the size of this character using CSS, so it doesn't mess with your layout.

#### Know Your Framework

There are two popular approaches to coding email. The most popular framework is called "responsive." The basis of responsive emails is to start with a 100%-width table (to which you can apply styles that will affect the whole email) and then floating a fixed-width table in the center of this wrapper (using align="center"). If you use media queries, you can adapt the width to various screen sizes. Our free responsive template is a great example of this coding technique.

The other popular framework is called hybrid fluid or "spongy" design. With the hybrid technique, you set container tables to width="100%" and constrain the container tables with a max-width style. When a hybrid design displays on screens wider than the max-width, the table will reach its max. On a smaller screen, the table will naturally fill the available space.

The "hybrid" part of this technique is that you must surround each table with a conditional table visible only to Outlook. The hybrid table has a fixed width, which solves the main problem with fluid design: Outlook ignores max-width statements.

The main advantage of the hybrid or fluid technique is that it works pretty much everywhere, regardless of whether the client supports embedded styles or media queries. For more on hybrid fluid design, check out our primer. We also have a few free hybrid fluid templates.

#### Make Sure You Include Comments in Your Code

Including comments in your code will make editing templates easier. Because email development is full of hacks and fixes for client quirks, it can be helpful to note why you added a particular style or element.

#### Encode Special Characters

If your email service provider (ESP) uses a different kind of encoding from the kind you selected for your email, it may cause your special characters (like ©) to appear incorrectly, often as a black square or a diamond. This can affect quotation marks and apostrophes, as well. To avoid this problem, use a character encoder, or take advantage of the one that is included with the Email on Acid Email Editor.

## Keep Email File Size Under 100kb

There are a couple good reasons to keep your email under 100kb. First, it will pass through more spam filters by staying light. Keeping your email under 102kb will also prevent Gmail from “clipping” your email.

To keep your email under the limit, consider removing redundant or unused styles, moving some of the content of the email to a landing page, or removing any unnecessary characters from your code. Just make sure to test any changes before the final send!

## Code for High DPI Displays

High DPI displays can often cause issues when scaling email designs. This is because it will scale certain parts of the email (height, width, font-size etc. that are coded in px), but not other parts. To make sure your whole email scales properly, just follow the steps in our coding for DPI scaling in Outlook blog post.

## Include Preheader Text

In most inboxes, preheader text displays after the email subject line. This text is easy to code and can make a huge difference in open rates. Just make sure you don’t hard-code “default” preheader text into your template.

If you include hard coded preheader text (usually in a field that can be modified), you may forget to customize it and you’ll send out an email with preheader text like, “PREHEADER TEXT HERE.” What a faux pas!

Instead, just include the “default” preheader text as an HTML comment. This way, other marketers and developers you work with will know why that code exists, but recipients won’t see it if you forget to customize.

## Avoid Javascript, Flash, Forms and other Complex CSS/HTML

Javascript and Flash are completely unsupported in email clients, so don’t use them at all. Newer code, such as HTML5 and CSS3 have limited support, but are sometimes possible (and fun!) to use. These enhancements should be used with caution. As always, test thoroughly when using any advanced code.

## Use Cellpadding for Spacing

Cellpadding provides reliable spacing across all email clients. If you need spacing only on one side of an image or container, you may want to use another spacing technique. Check out our blog post on spacing techniques in email for more info.

## Test Your Email!

Email coding is hard! Every email client has different quirks when it comes to rendering code. Outlook for desktop (2007, 2010, 2013 and 2016) can be especially challenging. The only way to know your email will look great everywhere is to test it. Email on Acid can help you test by generating screenshots of your email in more than 70 email devices and clients – all in less than 30 seconds.

## Images

### Make Email Images Retina Ready

Many devices now include “retina” displays. This means that the devices have more physical pixels than their CSS dimensions would otherwise indicate. For example, a 10px-wide image might use 20 or more physical pixels to display. By using extra-large images, you can make sure the images appear extra crisp on these displays. For more on this technique, read our article about retina images in email and fluid retina images for email.

## Use Absolute Addresses for Images

You may be using local image references for your testing, but when you do your final send absolute image references are a must!

## Get Rid of Strange Spacing Around an Image

This is a doctype issue. Use `display:block` and it will usually remove this extra spacing.

## Don't Use Image Maps

If you need to connect one image to multiple locations, you're going to have to slice it. Put each slice in its own table cell, and then link the images. This can cause all kinds of havoc trying to get the slices to line up perfectly, so only do this as a last resort.

## Take Some Extra Time with Background Images

Outlook can't handle the background attribute or backgrounds set through CSS, so you'll have to use VML to get backgrounds working in Outlook. Even with this workaround, Outlook can still be finicky. If you're having a hard time, try using Stig's button and background generators.

# CSS

## Use Inline Styles

Some Gmail clients, like Gmail Android App for Non-Gmail Accounts (GANGA), still don't support embedded styles. In addition to this, there are a few smaller email clients like Yandex and Telstra that still require inline styles.

To do this, you can code with classes and IDs and then make use of a CSS inliner. Email on Acid has an inliner that you can use from any email test, or from within the Email Editor.

## Avoid Shorthand CSS When Possible

If you see problems with a client interpreting your CSS, check to make sure you're not using a shorthand declaration. For example, "`margin-top: 5px`" may work where "`margin: 5px 0 0 0;`" does not. It's important to also avoid three-digit hex codes. Some clients will not recognize these, so you'll want to make sure you always use the full six-digit hex code.

## Get Used to !important

If you are a web developer, you may have been trained to avoid !important at all costs. When coding email, though, you'll find this declaration can be invaluable. You can use it to override styles that the email client adds or modifies (especially web clients). You'll also get a lot of use out of !important when writing media queries, where this declaration will let you override a default style with a mobile-specific one.

## Get Comfortable with Media Queries

Media queries are commonly used to create custom styles for different clients or screen sizes. The basic format of a media query for email is:

```
@media only screen and (max-device-width: 640px){ styles here }
```

This will cause the styles contained in the query to trigger only on screens of 640px or smaller. "Min-device-width" would do the opposite, triggering on screens of 640px or larger.

Media queries are most often used to control font sizes, image sizes, and to make some tables become 100% width so that they will fill a mobile screen. You can also use media queries to hide content that isn't necessary for mobile users. Just make sure that you use !important on styles within the media query, so that they will overwrite existing styles. Check out this article for more information on using media queries in HTML email.

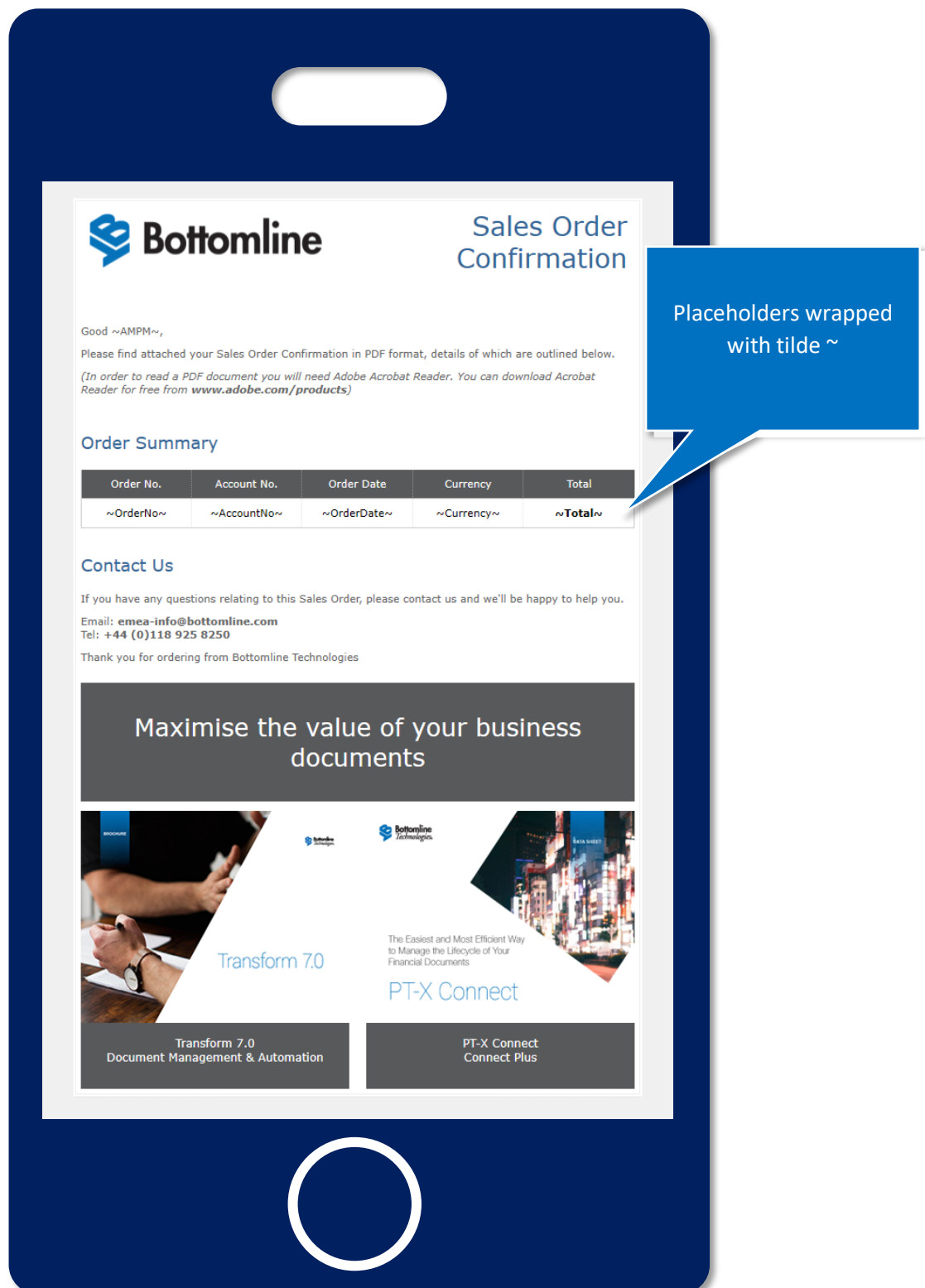
## **The Most Important Email Practice: Testing**

Even if you're using these email best practices, the most important part of any email process is testing. There's no point spending the time making sure your emails look great on different devices if you don't test them. When you use Email on Acid you can see how your email looks in more than 70 devices, giving you the confidence to hit "send."

## Static HTML Template

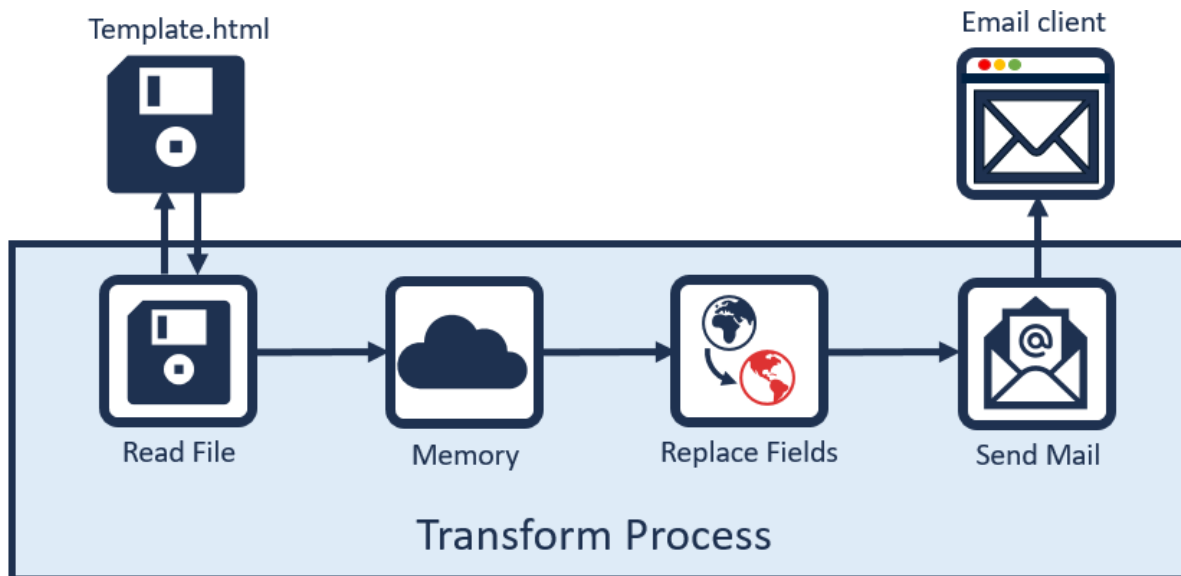
The following is an example transactional static HTML email template with no repeating item lines. The template shown is available within the Resources section of this Manual.

Take note of the placeholders wrapped in the tilde (~) character. These will be replaced within the Transform process. You need to agree with the Email developer the field names used within the HTML template.



## Transform Static HTML Integration

The diagram below depicts the Transform process required to integrate the Static HTML template.



### Template.html

External HTML template containing placeholders where you want to replace with values from the AX report data fields.

### Read File

Transform reads the external HTML template and loads it into Transform memory object.

### Memory

Holds the original external HTML template in a Text variable and a Unicode Text object to update with the replace field values.

### Replace Fields

Replace each placeholder within the HTML template with values from the AX report data.

### Send Mail

Ensure we update the Email Body field with the integrated HTML and enable the HTML content flag.

## Objects Required

To integrate the HTML Email templates we will use the following objects:-

-  Action List
-  Any Of
-  Catch
-  Container Shortcut
-  Copy Text
-  Copy Unicode Text
-  Decision
-  Dynamic Reformat
-  Folder
-  If
-  Lookup
-  Memory
-  Not
-  Number
-  Read File
-  Replace
-  Sentence
-  System Information
-  Text
-  Text Match
-  Text Template
-  Time
-  Tokenizer
-  Unicode From Text
-  Unicode Sentence
-  Unicode Text
-  Yes/No

## Create Email Body

In this session we are using a Dynamics AX report project, SalesConfirm\_Report. It is important that we add the routine to create the HTML body before the Presentation when integrating into a standard Dynamics AX project. This is because the output of the HTML integration will update the Section\_BT\_DPA > Field\_EmailBody field, which is used later in another Transform process responsible for Email delivery.

From the Branch editor add a [Decision](#) object at the add point between **Data Transformation** and **Presentation Action Lists**.



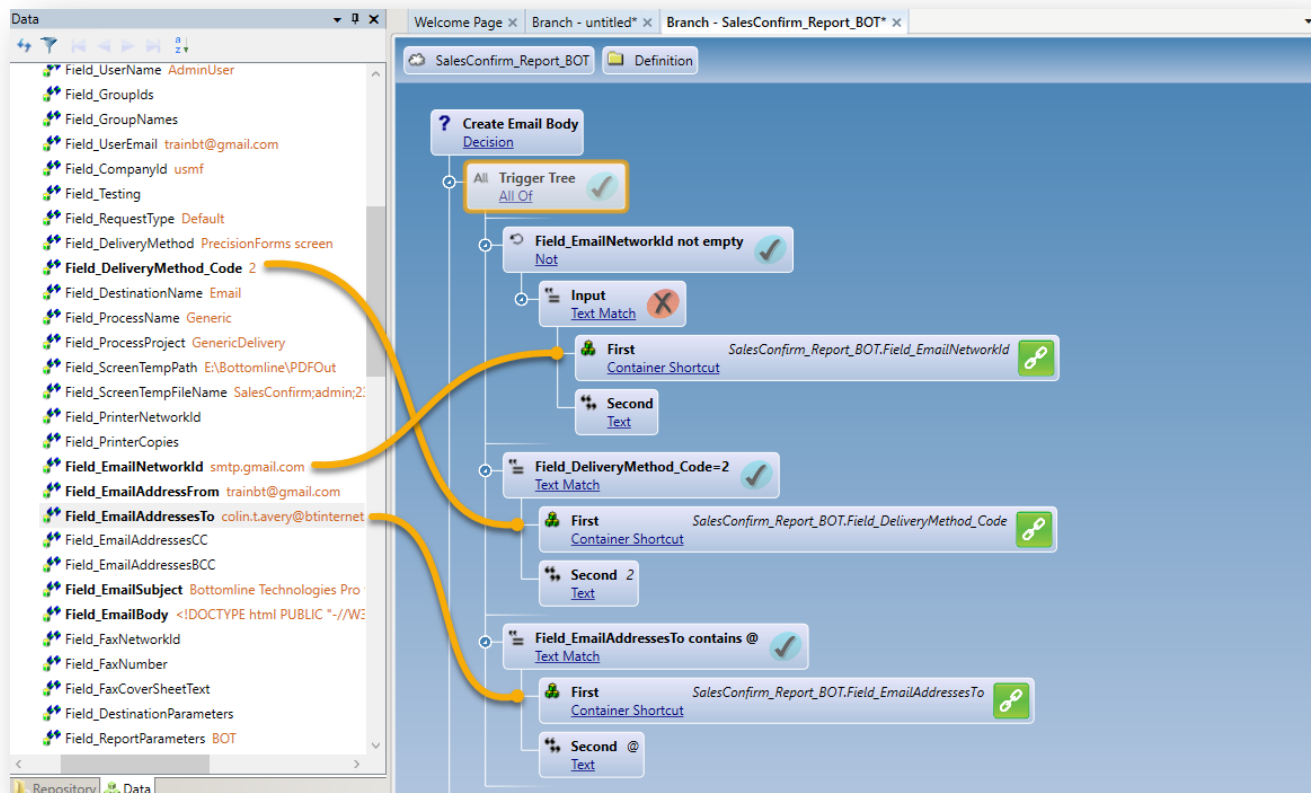
## Create Email Body - Decision

So that we do not perform unnecessary processing we only perform the HTML integration routine when the delivery destination is defined as Email.

In this example we have an **All of** (AND) expression which returns true when:

- **Field\_EmailNetworkId** is NOT empty
- **Field\_DeliveryMethod\_Code** is equal to 2
- **Field\_EmailAddressesTo** contains @

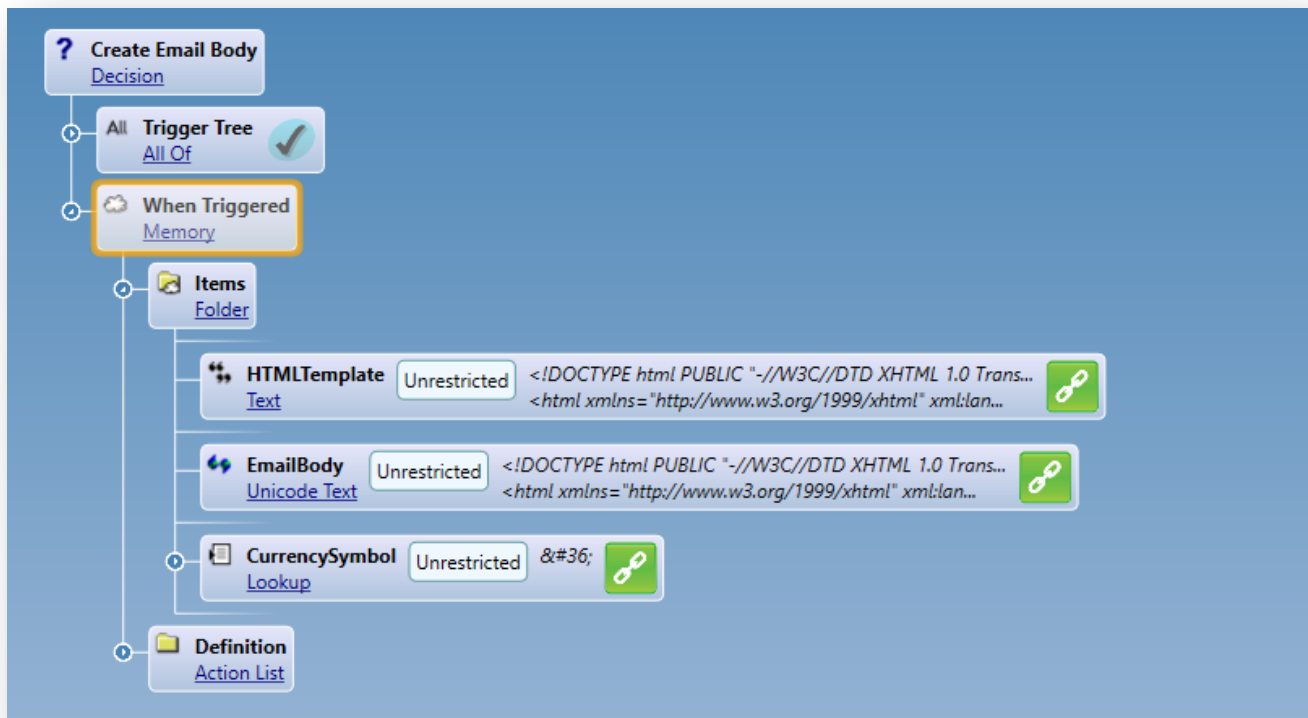
Under the **Create Email Body Decision** expand the **Trigger Tree All of** object and create the expressions required to trigger the email routine.



## Memory objects

Replace the **When Triggered** [Action List](#) with a [Memory](#) object. Then expand and add the following child objects within the **Items** [Folder](#).

- **HTMLTemplate** – [Text](#) object used to hold the content of the external HTML template once read into the project.
- **EmailBody** – [Unicode Text](#) object the email body with the replaced fields used as the Body in the Send Mail.
- **CurrencySymbol** – [Lookup](#) object used to return the decimal code for the currency symbols.



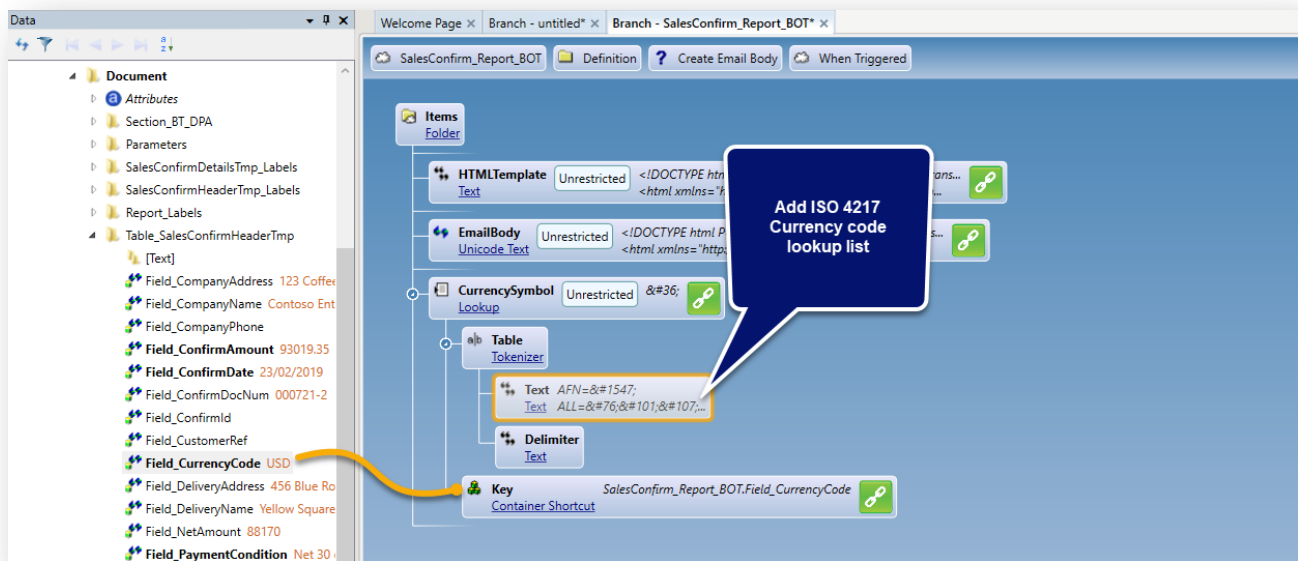
## CurrencySymbol – Lookup object

In order to add the currency symbol to the Total within the email body (\$ 12,546.00) we need to return the decimal value for that symbol.

The Lookup table has the ISO 4217 currency code as the key to the left of the equal sign (=) and returns the decimal values to the right of the equal sign. This will then be used to present the currency symbol within the HTML body.

Add the **Lookup** object and expand the **Table Tokenizer** and copy the ISO 4217 lookup below into the child **Text** object.

Link the child **Key** to the *Table\_SalesConfirmHeaderTmp > Field\_CurrencyCode*.



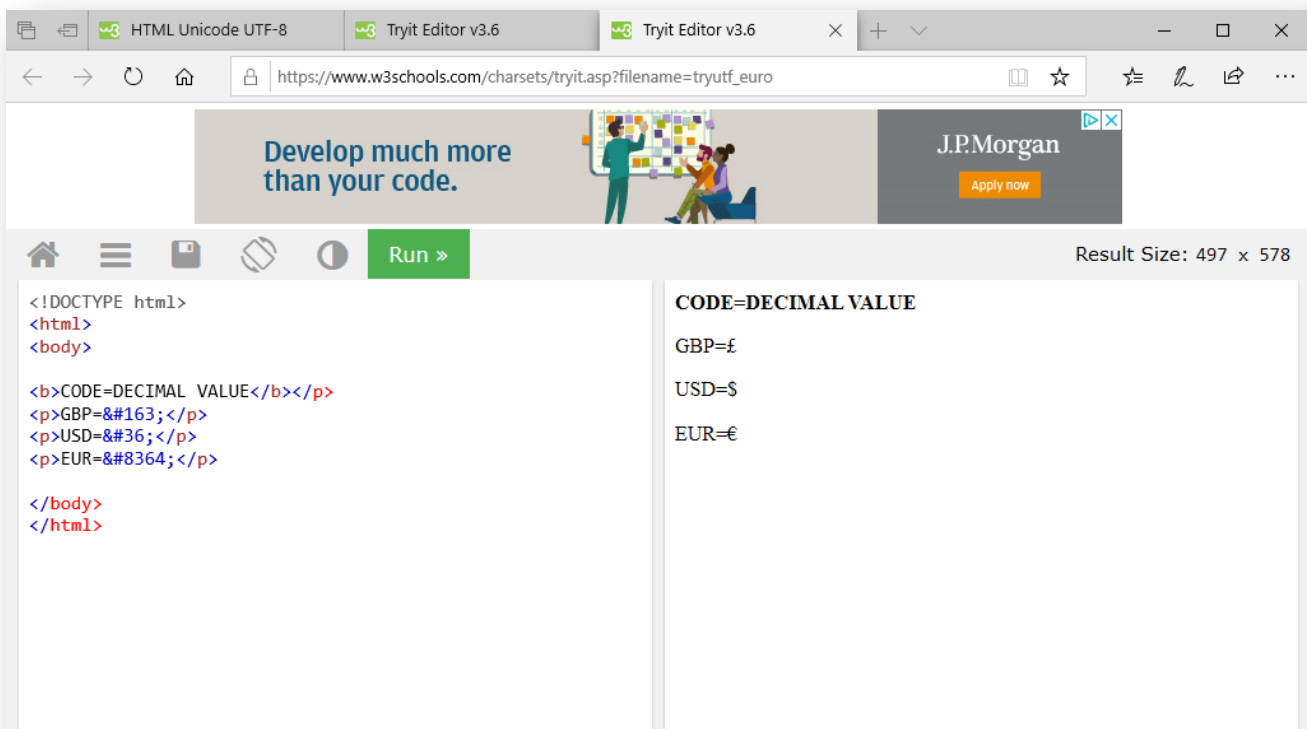
Lookup table of ISO 4217 Currency symbols and the decimal values. Copy into **Tokenizer Text** child.

```
CODE=DECIMAL VALUE
AFN=&#1547;
ALL=&#76;&#101;&#107;
ANG=&#402;
ARS=&#36;
AUD=&#36;
AWG=&#402;
AZN=&#1084;&#1072;&#1085;
BAM=&#75;&#77;
BBD=&#36;
BGN=&#1083;&#1074;
BMD=&#36;
BND=&#36;
BOB=&#36;&#98;
BRL=&#82;&#36;
BSD=&#36;
BWP=&#80;
BYN=&#66;&#114;
BZD=&#66;&#90;&#36;
CAD=&#36;
CHF=&#67;&#72;&#70;
CLP=&#36;
CNY=&#165;
COP=&#36;
CRC=&#8353;
CUP=&#8369;
CZK=&#75;&#269;
```

DKK=&#107;&#114;  
 DOP=&#82;&#68;&#36;  
 EGP=&#163;  
 EUR=&#8364;  
 FJD=&#36;  
 FKP=&#163;  
 GBP=&#163;  
 GGP=&#163;  
 GHS=&#162;  
 GIP=&#163;  
 GTQ=&#81;  
 GYD=&#36;  
 HKD=&#36;  
 HNL=&#76;  
 HRK=&#107;&#110;  
 HUF=&#70;&#116;  
 IDR=&#82;&#112;  
 ILS=&#8362;  
 IMP=&#163;  
 INR=&#8377;  
 IRR=&#65020;  
 ISK=&#107;&#114;  
 JEP=&#163;  
 JMD=&#74;&#36;  
 JPY=&#165;  
 KGS=&#1083;&#1074;  
 KHR=&#6107;  
 KPW=&#8361;  
 KPW=&#8361;  
 KRW=&#8361;  
 KRW=&#8361;  
 KYD=&#36;  
 KZT=&#1083;&#1074;  
 LAK=&#8365;  
 LBP=&#163;  
 LKR=&#8360;  
 LRD=&#36;  
 MKD=&#1076;&#1077;&#1085;  
 MNT=&#8366;  
 MUR=&#8360;  
 MXN=&#36;  
 MYR=&#82;&#77;  
 MZN=&#77;&#84;  
 NAD=&#36;  
 NGN=&#8358;  
 NIO=&#67;&#36;  
 NOK=&#107;&#114;  
 NPR=&#8360;  
 NZD=&#36;  
 OMR=&#65020;  
 PAB=&#66;&#47;&#46;  
 PEN=&#83;&#47;&#46;  
 PHP=&#8369;  
 PKR=&#8360;  
 PLN=&#122;&#322;  
 PYG=&#71;&#115;  
 QAR=&#65020;  
 RON=&#108;&#101;&#105;  
 RSD=&#1044;&#1080;&#1085;&#46;  
 RUB=&#1088;&#1091;&#1073;  
 SAR=&#65020;  
 SBD=&#36;  
 SCR=&#8360;  
 SEK=&#107;&#114;  
 SGD=&#36;  
 SHP=&#163;

SOS=&#83;  
 SRD=&#36;  
 SVC=&#36;  
 SYP=&#163;  
 THB=&#3647;  
 TRY=&#;  
 TTD=&#84;&#84;&#36;  
 TVD=&#36;  
 TWD=&#78;&#84;&#36;  
 UAH=&#8372;  
 USD=&#36;  
 UYU=&#36;&#85;  
 UZS=&#1083;&#1074;  
 VEF=&#66;&#115;  
 VND=&#8363;  
 XCD=&#36;  
 YER=&#65020;  
 ZAR=&#82;  
 ZWD=&#90;&#36;

You can test any characters online at [https://www.w3schools.com/charsets/ref\\_utf\\_currency.asp](https://www.w3schools.com/charsets/ref_utf_currency.asp)



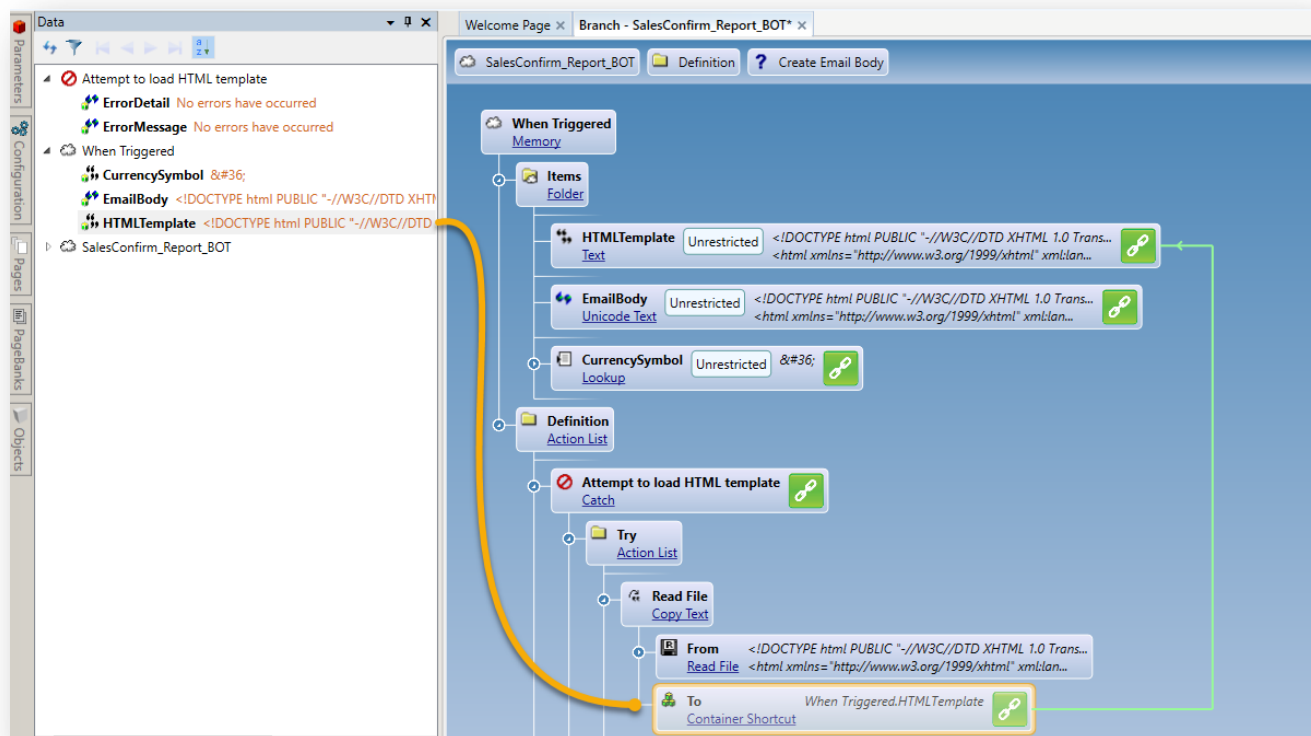
## Attempt to load HTML template - Catch Try

As we are reading the HTML template from the filesystem, we need to take care that we include some error trapping within the Transform process. This way if the external template is not available, we can construct the email body as plain text, to ensure we do not delay the delivery of the email.

Under the **Definition Action List** add a **Catch** object, expand the **Try Action List** and add a **Copy Text** object. The **Copy Text** has two child objects.

**From** - Replace the From **Text** object with a **Read File** object. The Read File contains a child **Path Text** object, this is the path to the HTML template. Left click the **Read File** icon and browse for the HTML template provided.

**To** – **Container Shortcut** to the **HTMLTemplate Text** memory object.



### Note

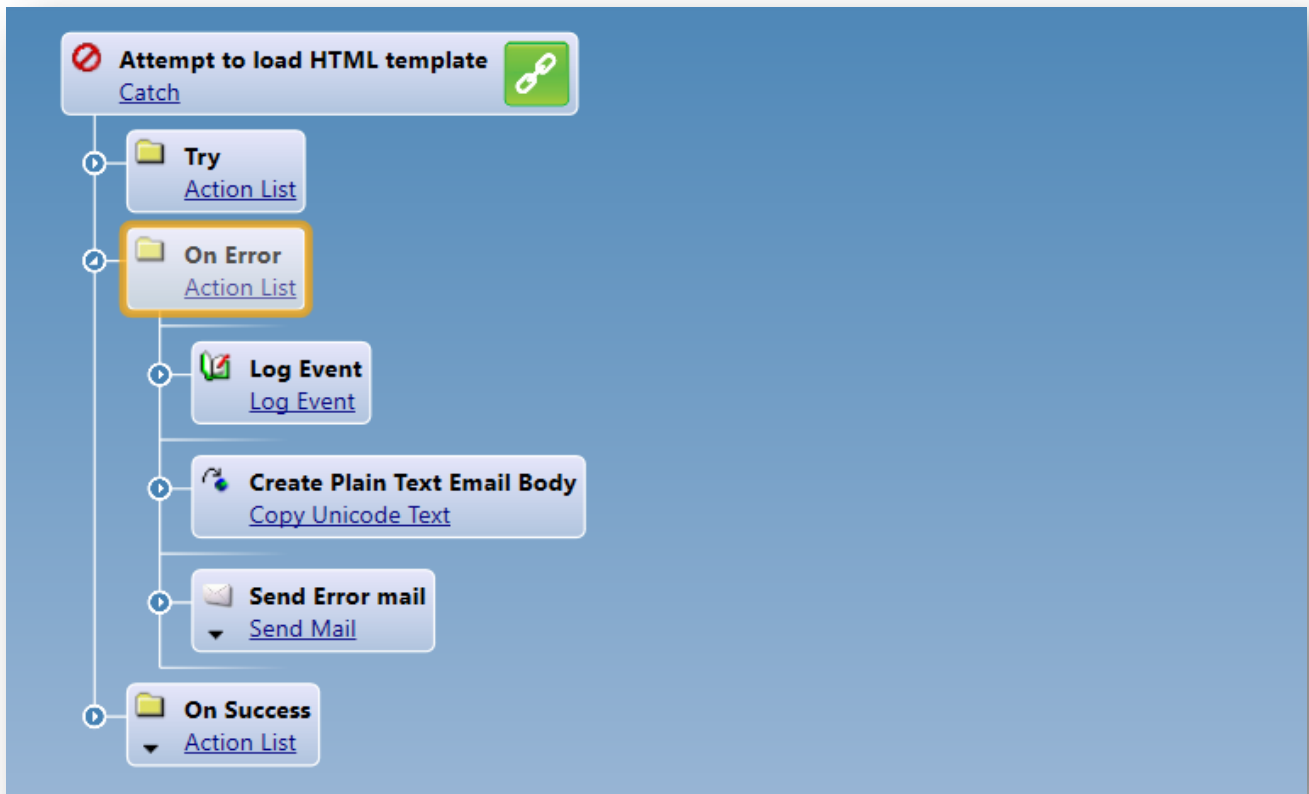
When using Read or Write file objects it is best practice to make the Path a global resource, so they are not buried deep inside your Transform branch. You can do this several ways either through a global resource branch or through the Transform Configuration File. Ask your trainer for examples of using global resources.

## On Error - Action List

If the HTML template fails to load, then **On Error** is executed. We want this to perform three actions.

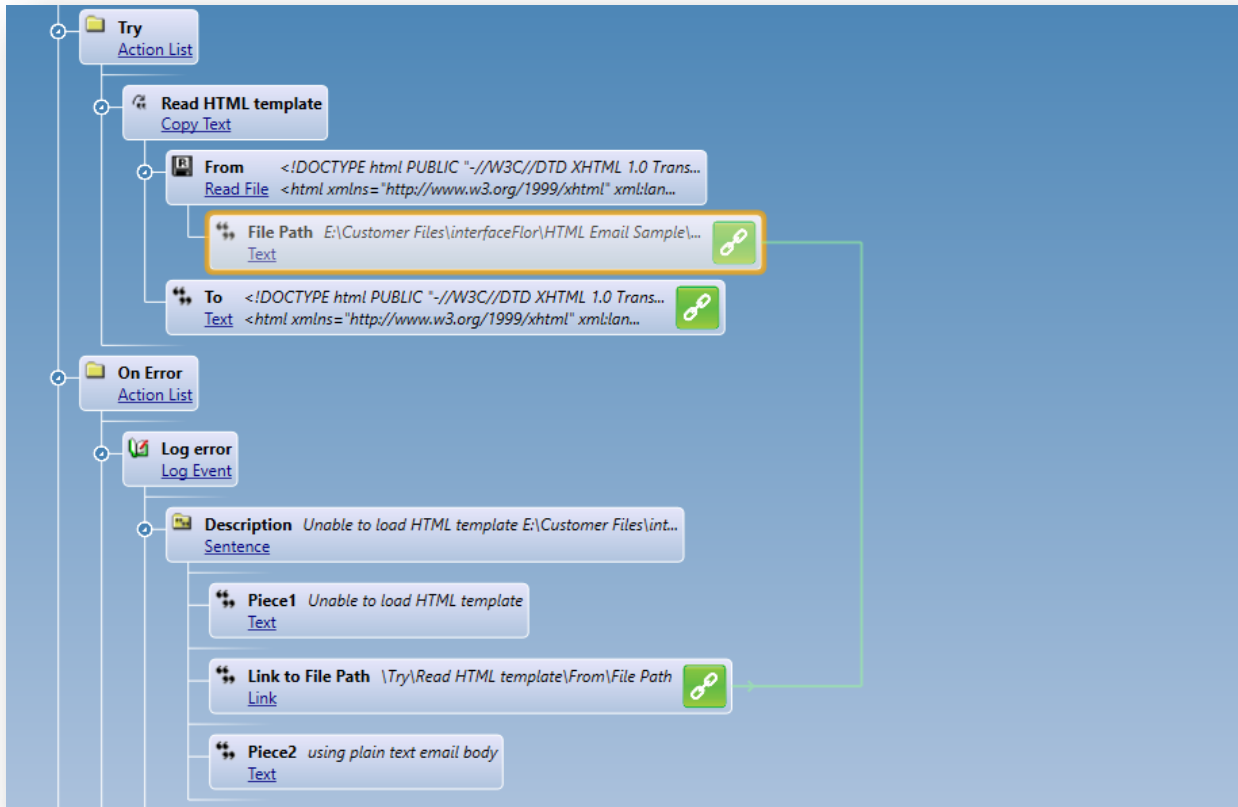
- **Log Error** – [Log Event](#) object
- **Create Plain Text Email Body** – [Copy Unicode Text](#) object
- **Send Error mail** – [Send Mail](#) object

Replace the **OnError** [Log Event](#) with an [Action List](#). This will adopt the [Log Event](#) as a child of the [Action List](#).



## Log error – Log Event

So that we can manage errors, we create a Log entry with the **Log Event** object. To make the description clear replace the child **Description Text** object with a **Sentence**, add some text on **Piece1** then create a link to the HTML File Path used within the **Read File** object. To create the link right click the source and drag onto the target then select **Create link**, complete the log description by adding another **Text** object.

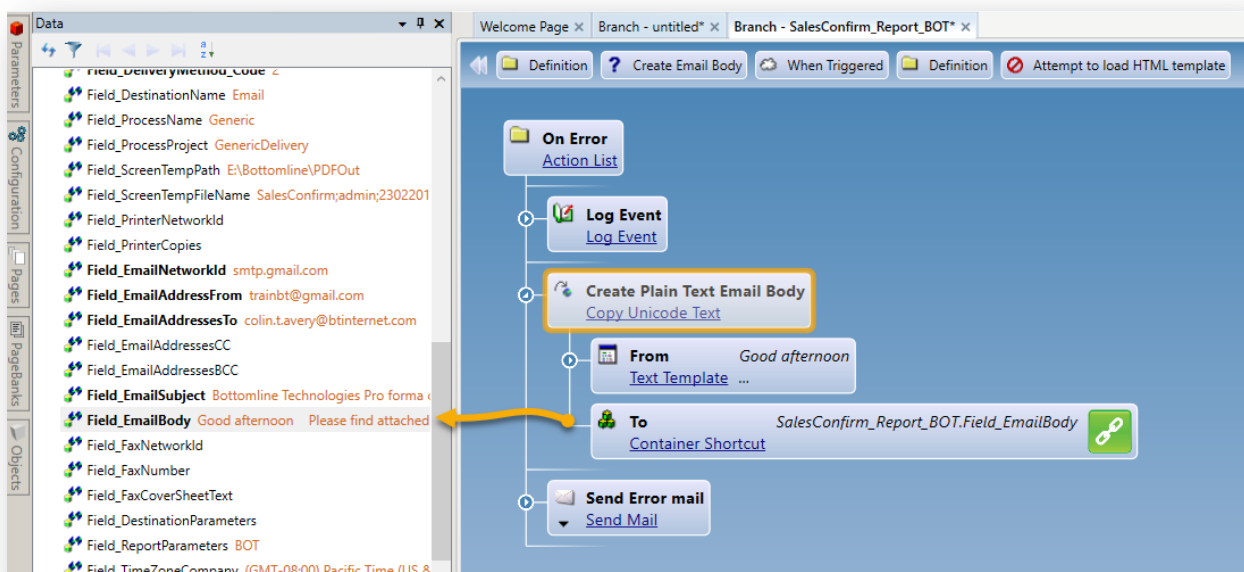


## Update EmailBody using plain text – Copy Unicode Text object

To create the plain text email body add a **Copy Unicode Text** object, containing two child objects.

**From** – Replace the From **Text** with a **Text Template** object used to construct the plain text email body.

**To** – **Container Shortcut** to **Section\_BT\_DPA > Field\_EmailBody**



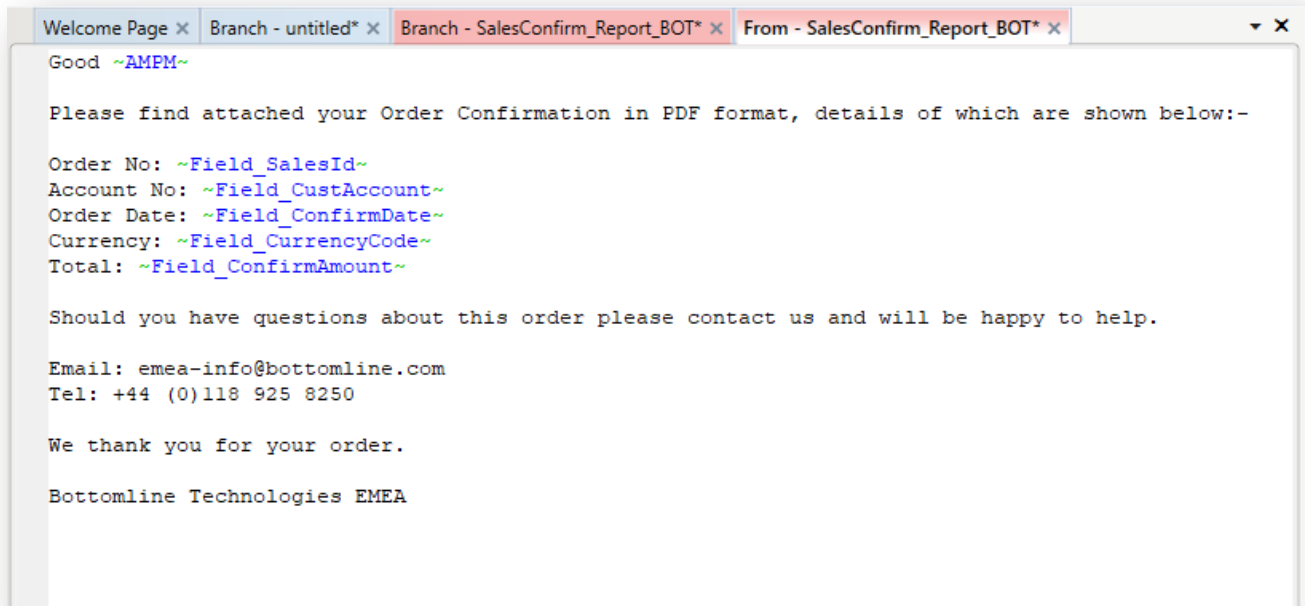
## From Text Template

Stores a stream of Unicode text data and combines it with variables to create a single stream of text.

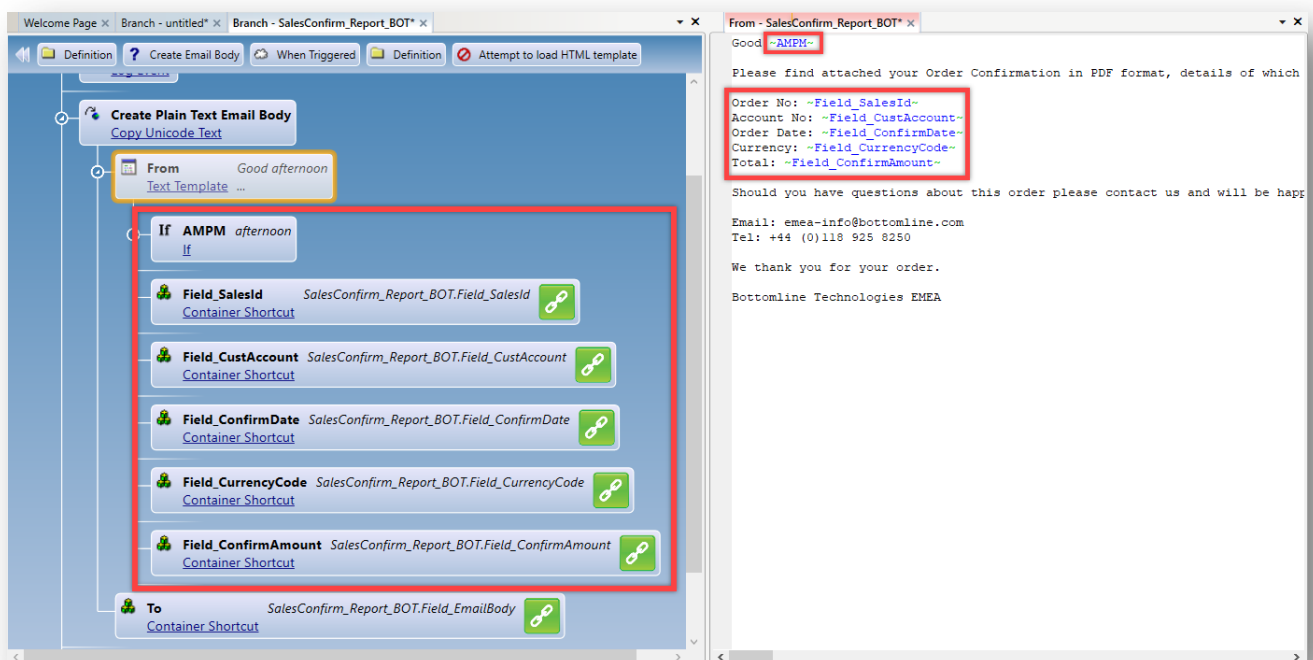
The **Text Template** object has its own editor, where you enter the static text and mark the position of the variable values to be embedded in the text stream. To open the editor, click the **Text Template** icon.

To include the value of a variable in the text stream, in the TextTemplate editor, enter the label of the child object, enclosed by a pair of tilde (~) characters. If you enter the name of a variable that does not match the **label** of any of the child objects, or that is invalid in some other way (for example, by containing a space or other non-alphanumeric character), the name appears in red. Valid variable labels are displayed in blue.

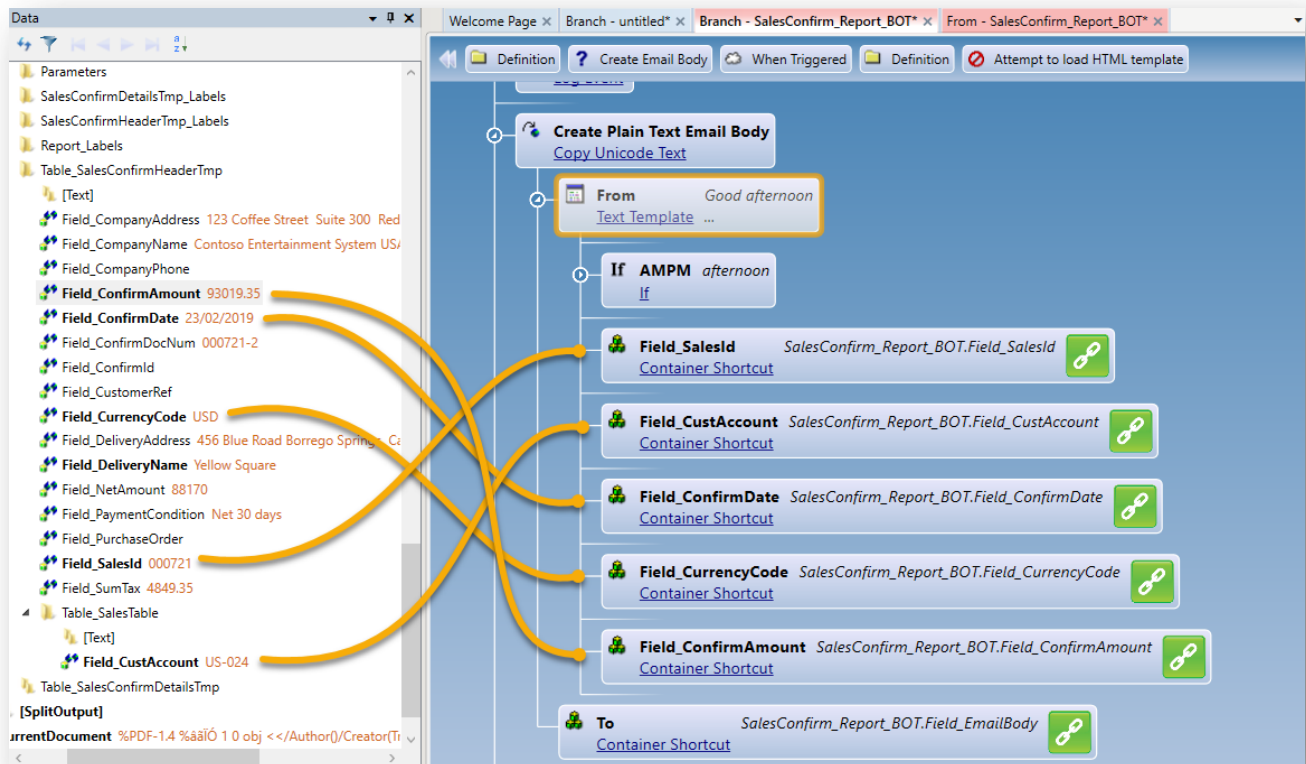
This allows us to create the following plain text email body.



The **Text Template** child objects are mapped from the data and the labels match the placeholders defined between the tilde (~) character within the Text Template editor.



The **Text Template** child objects are mapped from values within the Input AX file.

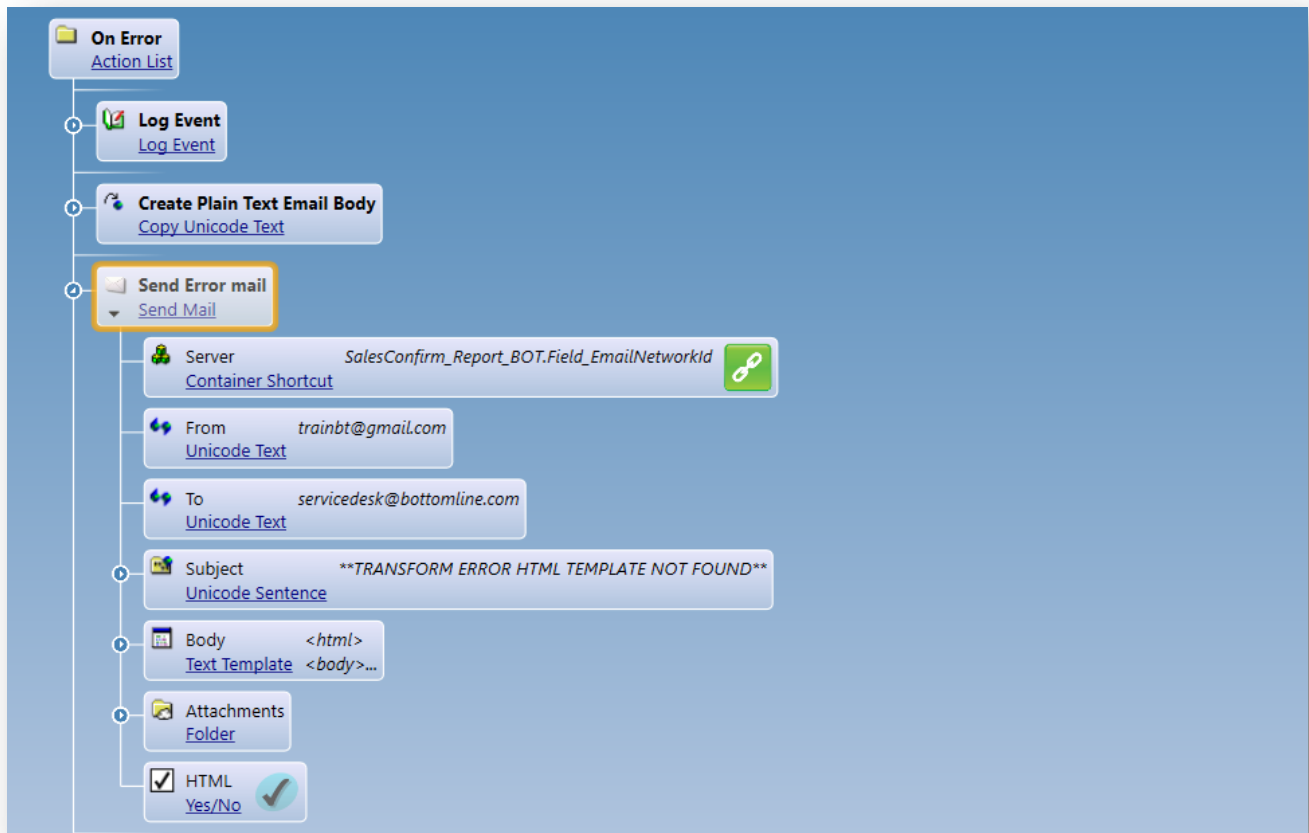


### Note

Text Template identifies each child object by its label. The label must begin with an alphabetic character, the remaining characters must contain alphanumeric characters only and must not contain spaces.

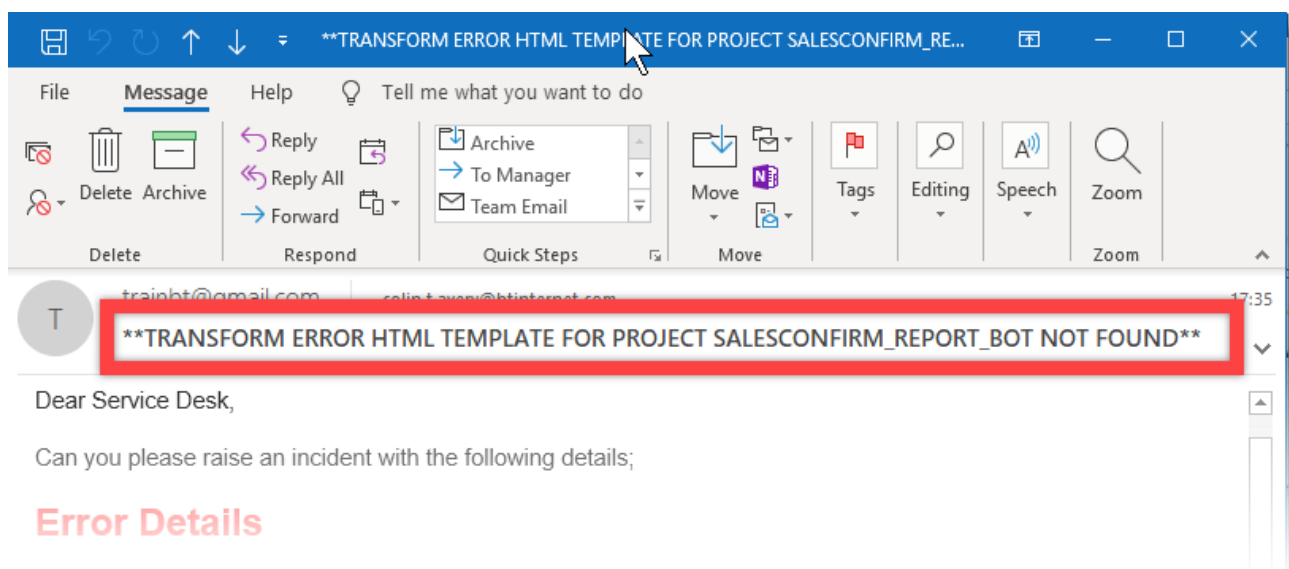
## Send Error mail – Send Mail object

So that the error does not go un-noticed we create a mail to send to the Transform support team. Add a **Send Mail** object, making use of a **Unicode sentence** for the **Subject** and a **Text Template** for the **Body**.

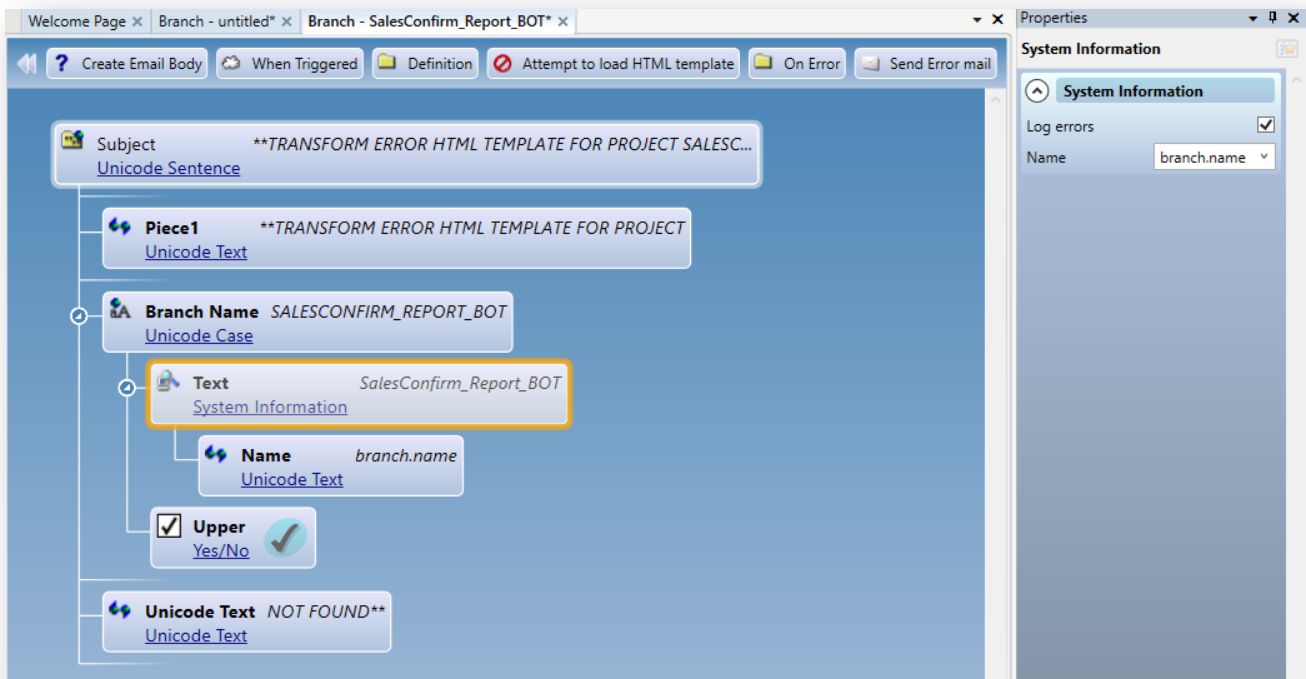


### Subject – Unicode Sentence

The subject as it appears in the email client.

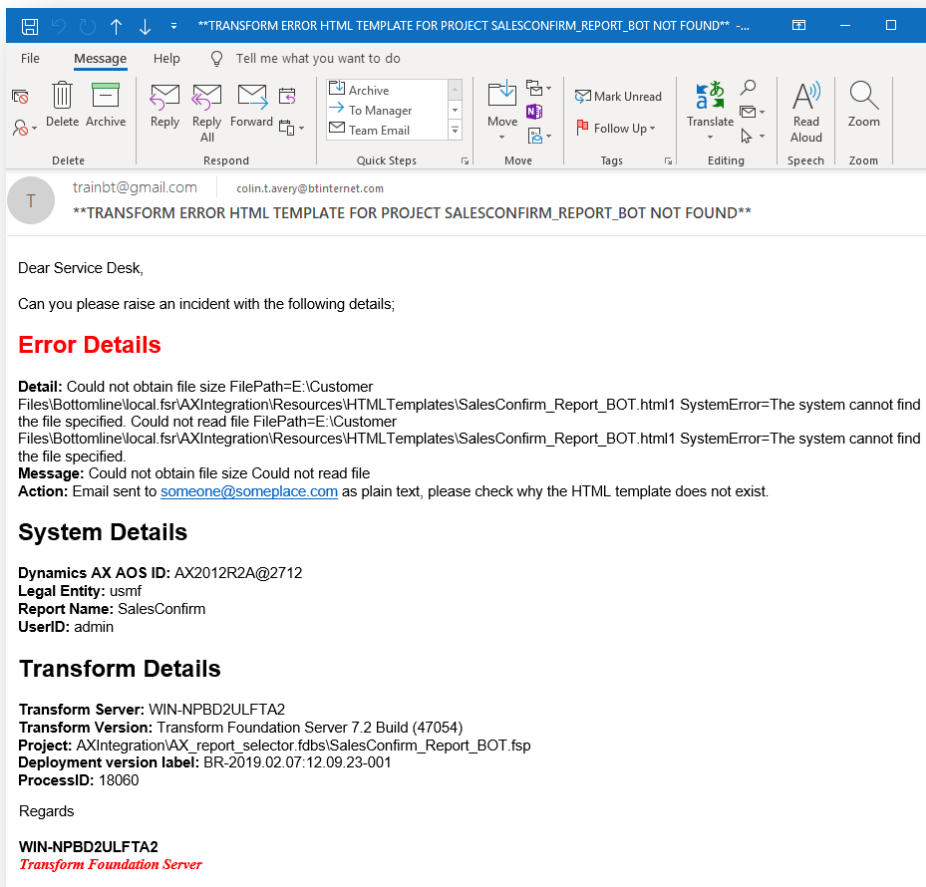


Replace the **Subject Unicode Text** with a **Unicode Sentence**, and add the child objects illustrated.



## Body – Text Template

The body of the error email as displayed in the email client.



By making use of the [Text Template](#) object we can add all the required information that will help the support desk understand and resolve the problem.

Replace the **Body Unicode Text** with a [Text Template](#) object. In this example we make use of simple HTML for the body of the text template with placeholders for the variable fields, copy the code below into your Text Template editor.

```
<html>
<body>
<font face="arial">
Dear Service Desk,
<p>
Can you please raise an incident with the following details;
</p>

<p>
<font color="red">
<h2>Error Details</h2>
<font color="black">
<b>Detail:</b> ~ErrorDetail~<br>
<b>Message:</b> ~ErrorMessage~<br>
<b>Action:</b> ~Action~<br>
</p>

<p>
<h2>System Details</h2>
<b>Dynamics AX AOS ID:</b> ~Field_AX_AOSId~<br>
<b>Legal Entity:</b> ~Field_CompanyId~<br>
<b>Report Name:</b> ~Field_Object~<br>
<b>UserID:</b> ~Field_UserId~
</p>

<p>
<h2>Transform Details</h2>
<b>Transform Server:</b> ~Envcomputername~<br>
<b>Transform Version:</b> ~version~<br>
<b>Project:</b> ~Branchpath~<br>
<b>Deployment version label:</b> ~deploymentbranchversionlabel~<br>
<b>ProcessID:</b> ~Processid~</p>

<p>
Regards
</p>
<b>~Envcomputername~</b><br>
<font face="times new roman" color="red">
<i><b>Transform Foundation Server</b></i>

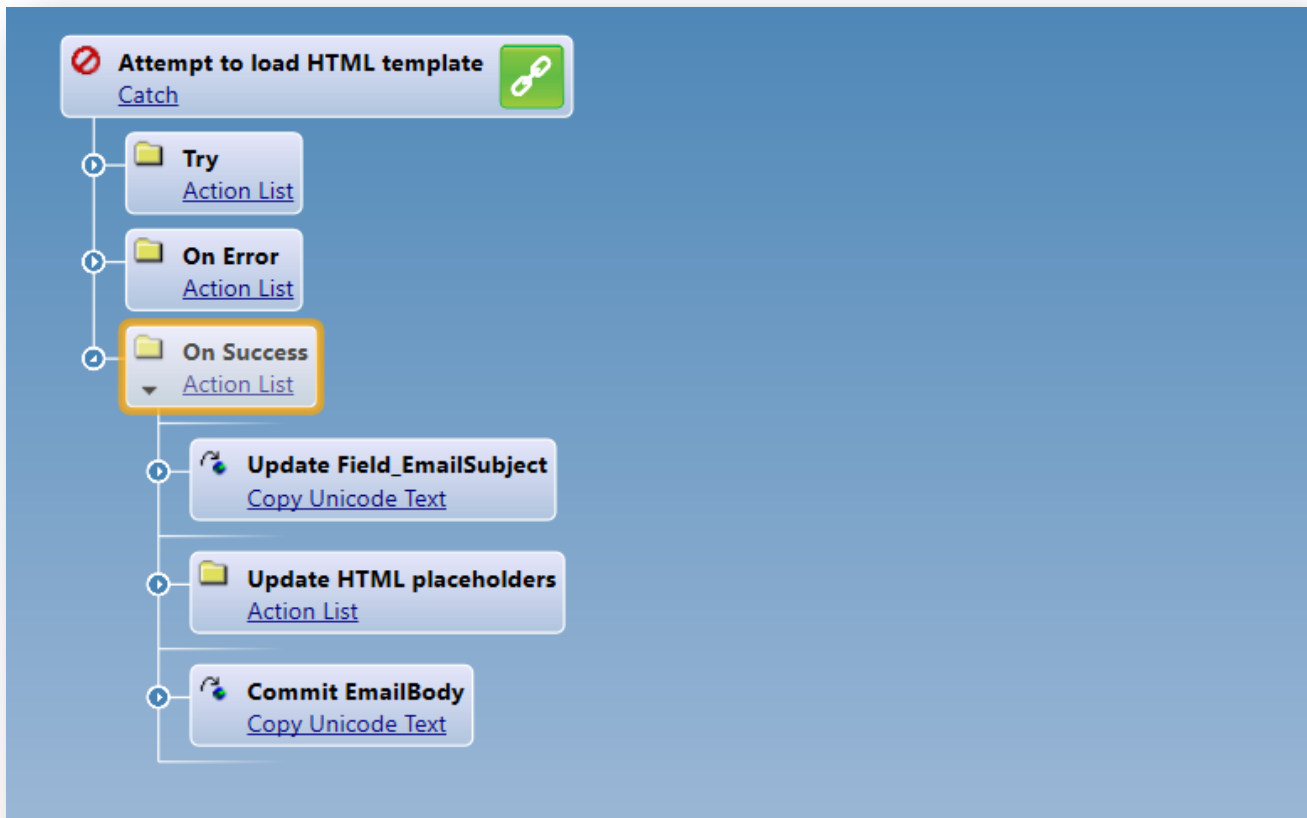
</body>
</html>
```

## On Success - Action List

When successful we need to perform the following actions:

- **Update Field\_EmailSubject** - [Copy Unicode Text](#) to construct the mail subject.
- **Update Placeholders in HTMLTemplate** – [Action List](#) object that contains multiple [Copy Unicode Text](#) objects to perform replace function on template placeholders with Dynamics AX report data.
- **Commit EmailBody** – [Copy Unicode Text](#) object to update the *Section\_BT\_DPA > Field\_EmailBody*.

Replace the **On Success Log Event** with an [Action List](#) object.

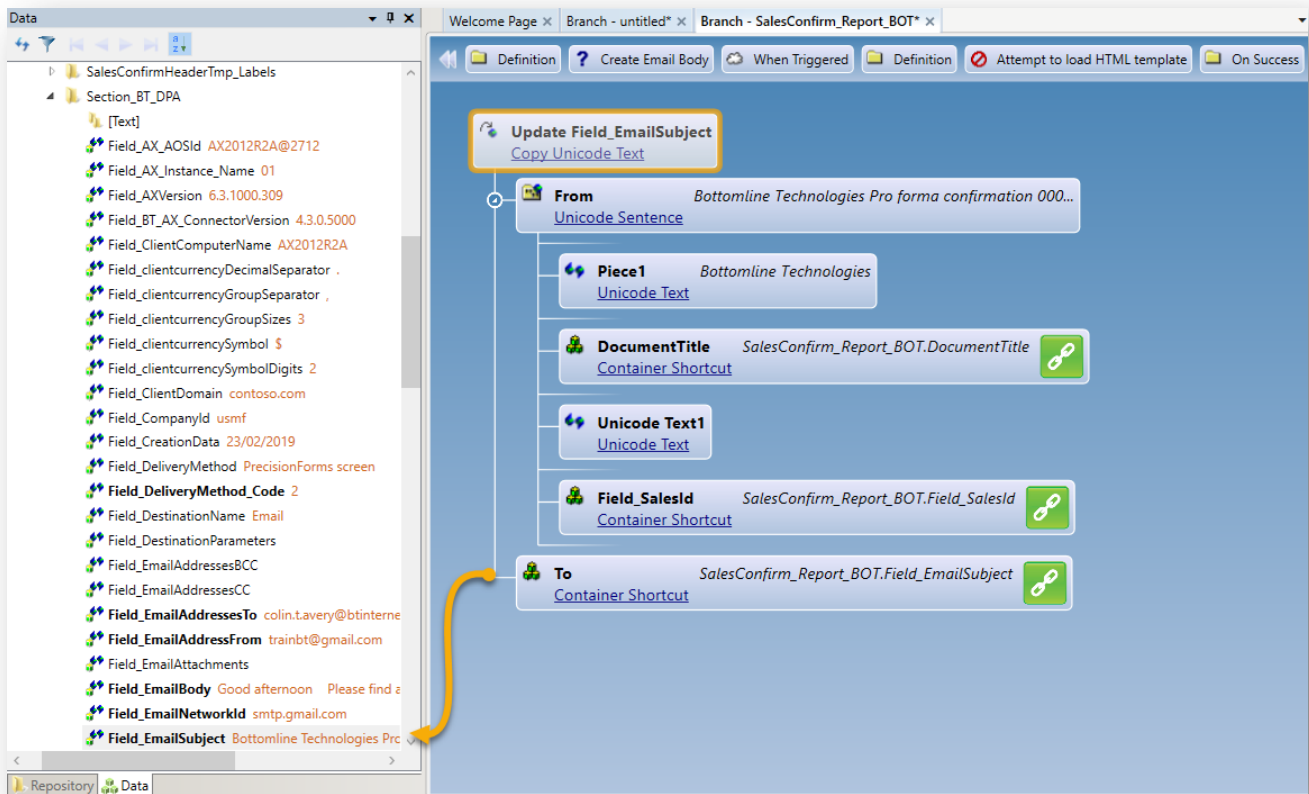


## Update Field\_EmailSubject – Copy Unicode Text

Add a [Copy Unicode Text](#) object as a child of the **On Success Action List**. The [Copy Unicode Text](#) has two child objects.

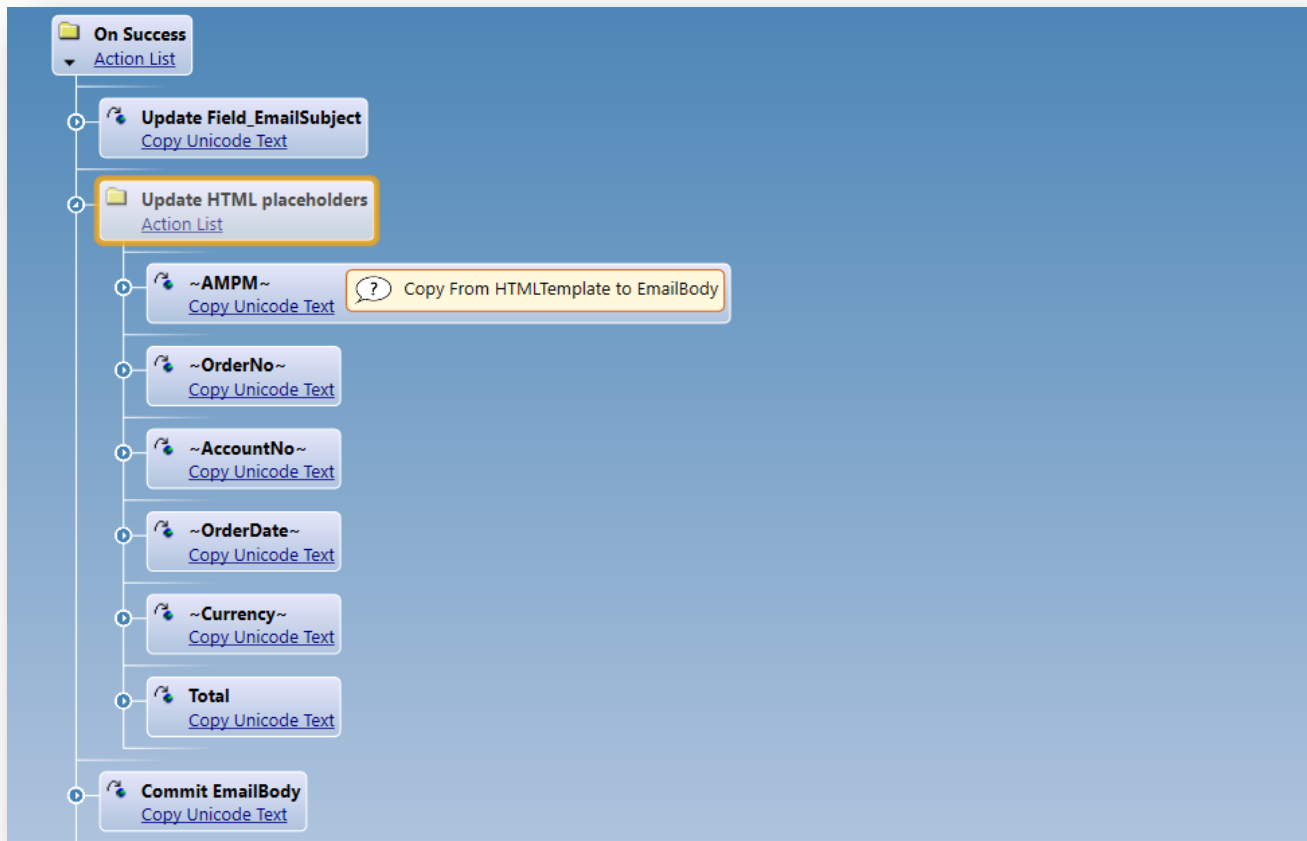
**From** – Replace the [Unicode Text](#) with a [Unicode Sentence](#) object to construct the email subject.

**To** - [Container Shortcut](#) to *Section\_BT\_DPA > Field\_EmailSubject*.

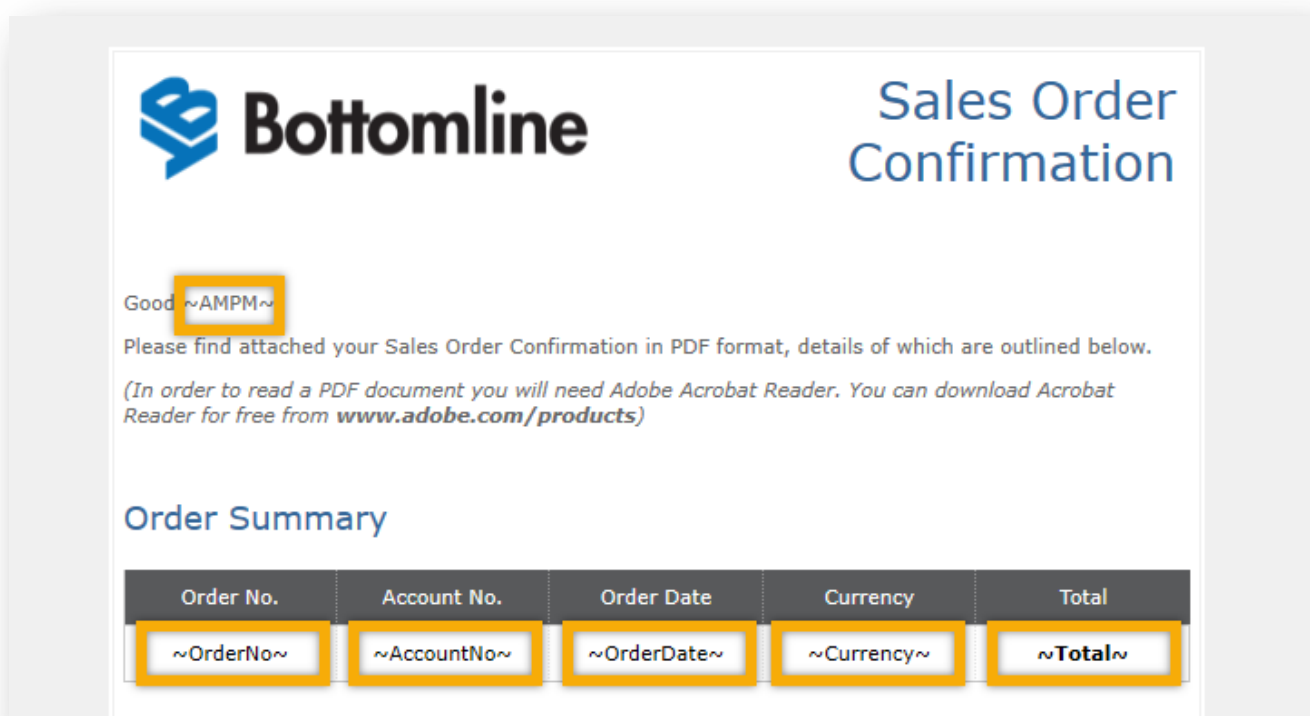


## Update Placeholders in HTMLTemplate – Action List

Add an **Action List**, this will contain multiple **Copy Unicode Text** objects used to update the placeholders within the HTML template with fields from the Dynamics AX report data.



The placeholders in the template are outlined in orange below:-



## Placeholder mappings

The following table depicts the HTML placeholder names and what value we should replace it within the Transform process.

### Order Summary

The Order Summary Dynamics AX report data fields are within **Table\_SalesConfirmHeaderTmp**

HTML Placeholder	Replace with AX Field	Output Sample
~AMPM~	Condition to return morning or afternoon	morning
~OrderNo~	Field_SalesId	000721
~AccountNo~	Field_CustAccount	US-024
~OrderDate~	Field_ConfirmDate	14/12/10
~Currency~	Field_CurrencyCode	USD
~Total~	Field_ConfirmAmount	\$ 5,898.12



---

### Important

Make sure you create a mapping table to ensure you know the placeholder names within the Text Template and how they relate to the variable fields within the Dynamics AX report data.

---

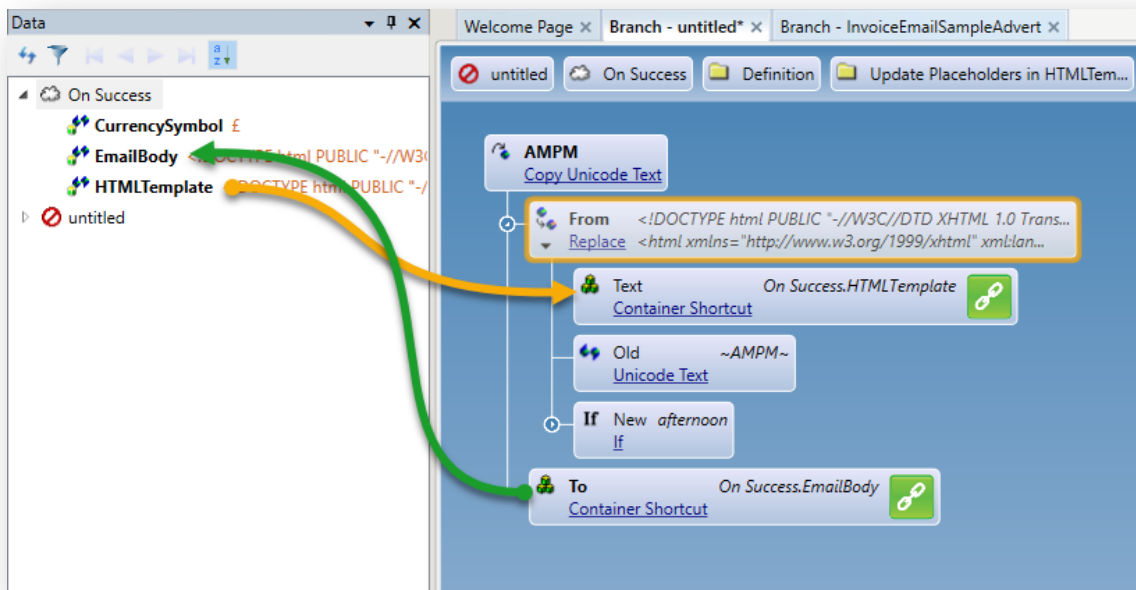
## AMPM – Copy Unicode Text

Add a [Copy Unicode Text](#), expand to reveal its two child objects.

**From** – Replace the From [Unicode Text](#) with a [Replace](#) object which has three children

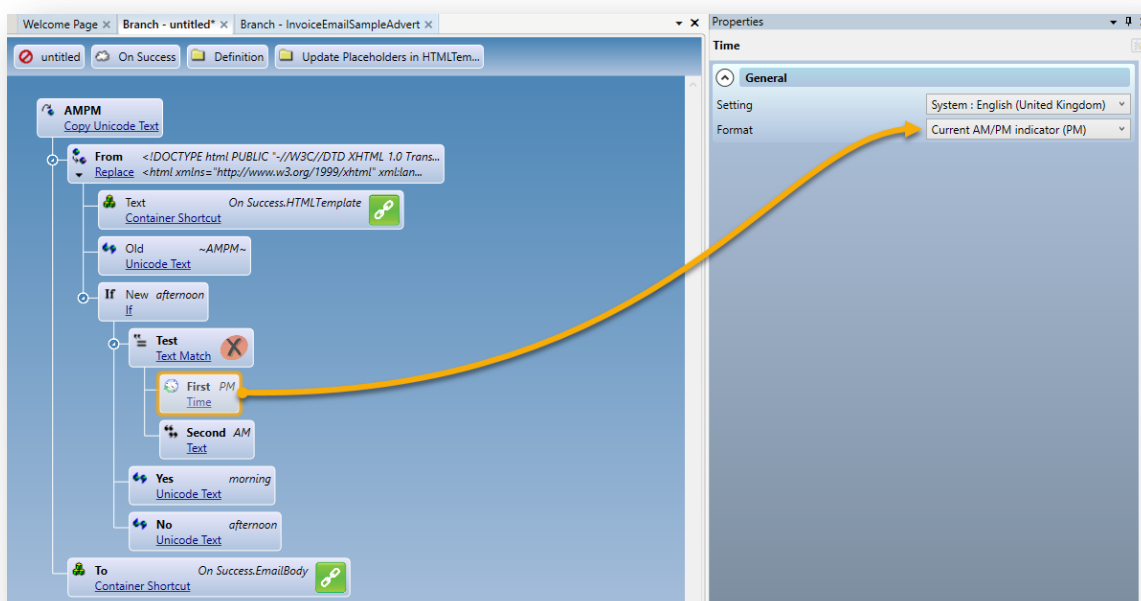
- **Text** – [Container Shortcut](#) to HTMLTemplate important that ONLY the first [Copy Unicode Text](#) object is replacing from HTMLTemplate.
- **Old** – [Unicode Text](#) string to replace ~AMPM~
- **New** – Replace the [Unicode Text](#) with an [If](#) object

**To** – [Container Shortcut](#) to [EmailBody Unicode Text](#) memory object.



The [If](#) has three child objects:

- **Test** - Replace the **Test Yes/No** with a [Text Match](#) object. Replace the **First Text** with a [Time](#) object. Set the Format properties of the [Time](#) object to Current AM/PM indicator. On the **Second Text** object add the value AM.
- **Yes** – When the test returns True AM then change the [Unicode Text](#) value to **morning**.
- **No** – When the test returns false PM then change the [Unicode Text](#) value to **afternoon**.



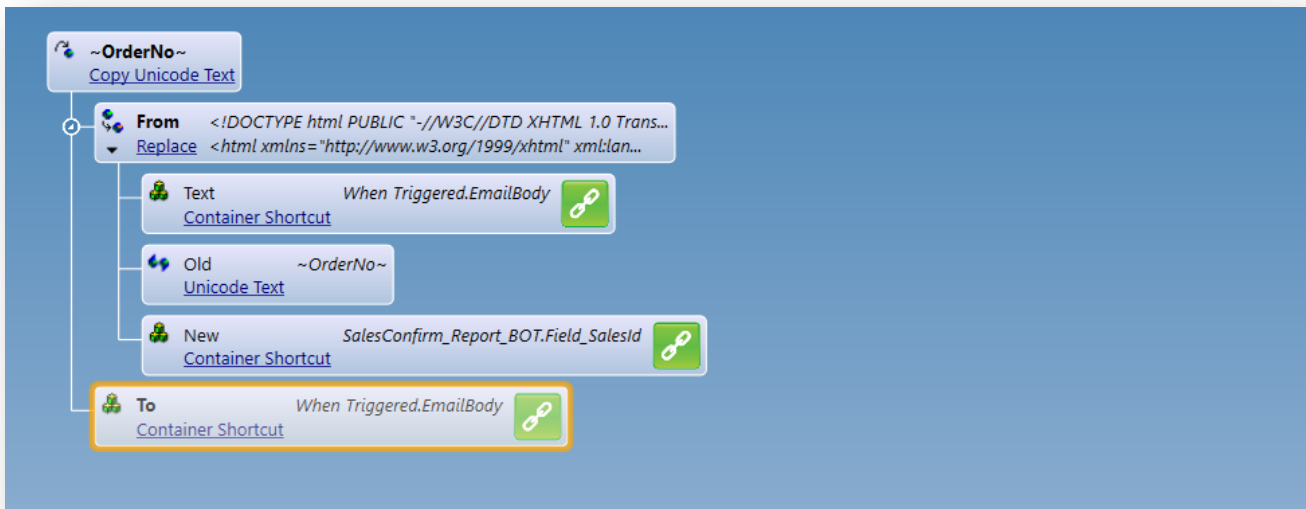
## OrderNo – Copy Unicode Text

Add a [Copy Unicode Text](#) object expand and change the **From** and **To** child objects as shown.

**From** – Change to a [Replace](#) object with the three children set:

- **Text** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object
- **Old** – String to replace ~OrderNo~
- **New** – [Container Shortcut](#) to *Table\_SalesConfirmHeaderTmp > Field\_SalesId*

**To** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object.



## AccountNo – Copy Unicode Text object

Add a [Copy Unicode Text](#) object expand and change the **From** and **To** child objects as shown.

**From** – Change to a [Replace](#) object with the three children set:

- **Text** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object
- **Old** – String to replace ~AccountNo~
- **New** – [Container Shortcut](#) to *Table\_SalesConfirmHeaderTmp > Table\_SalesTable > Field\_CustAccount*

**To** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object.

## OrderDate – Copy Unicode Text object

Add a [Copy Unicode Text](#) object expand and change the **From** and **To** child objects as shown.

**From** – Change to a [Replace](#) object with the three children set:

- **Text** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object
- **Old** – String to replace ~OrderDate~
- **New** – [Container Shortcut](#) to *Table\_SalesConfirmHeaderTmp > Field\_ConfirmDate*

**To** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object.

## Currency – Copy Unicode Text object

Add a [Copy Unicode Text](#) object expand and change the **From** and **To** child objects as shown.

**From** – Change to a [Replace](#) object with the three children set:

- **Text** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object
- **Old** – String to replace ~Currency~
- **New** – [Container Shortcut](#) to *Table\_SalesConfirmHeaderTmp > Field\_CurrencyCode*

**To** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object.

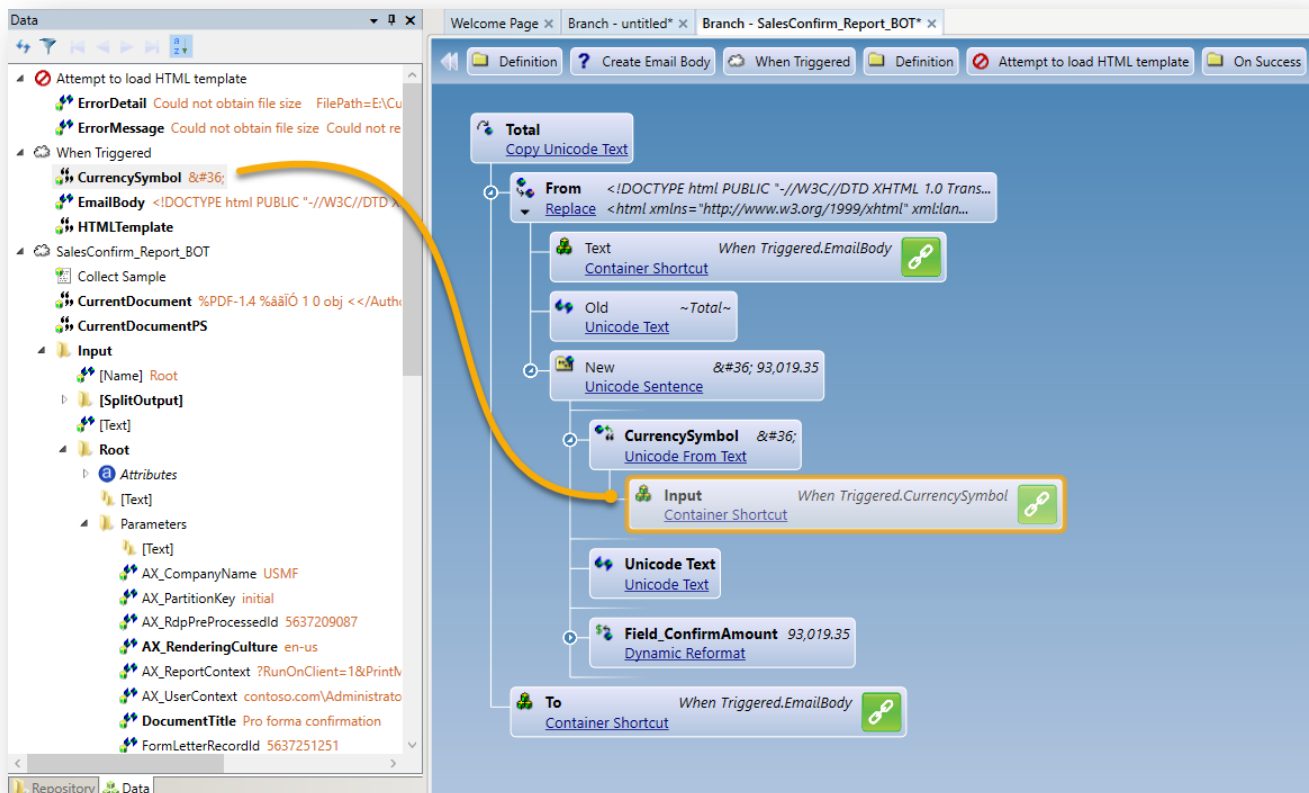
## Total – Copy Unicode Text object

Add a [Copy Unicode Text](#) object expand and change the **From** and **To** child objects as shown.

**From** – Change to a [Replace](#) object with the three children set:

- **Text** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object
- **Old** – String to replace ~Currency~
- **New** – [Unicode Sentence](#) to include the currency symbol decimal value returned by the memory [Lookup](#) and [Container Shortcut](#) to *Table\_SalesConfirmHeaderTmp > Field\_ConfirmAmount*

**To** – [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object.

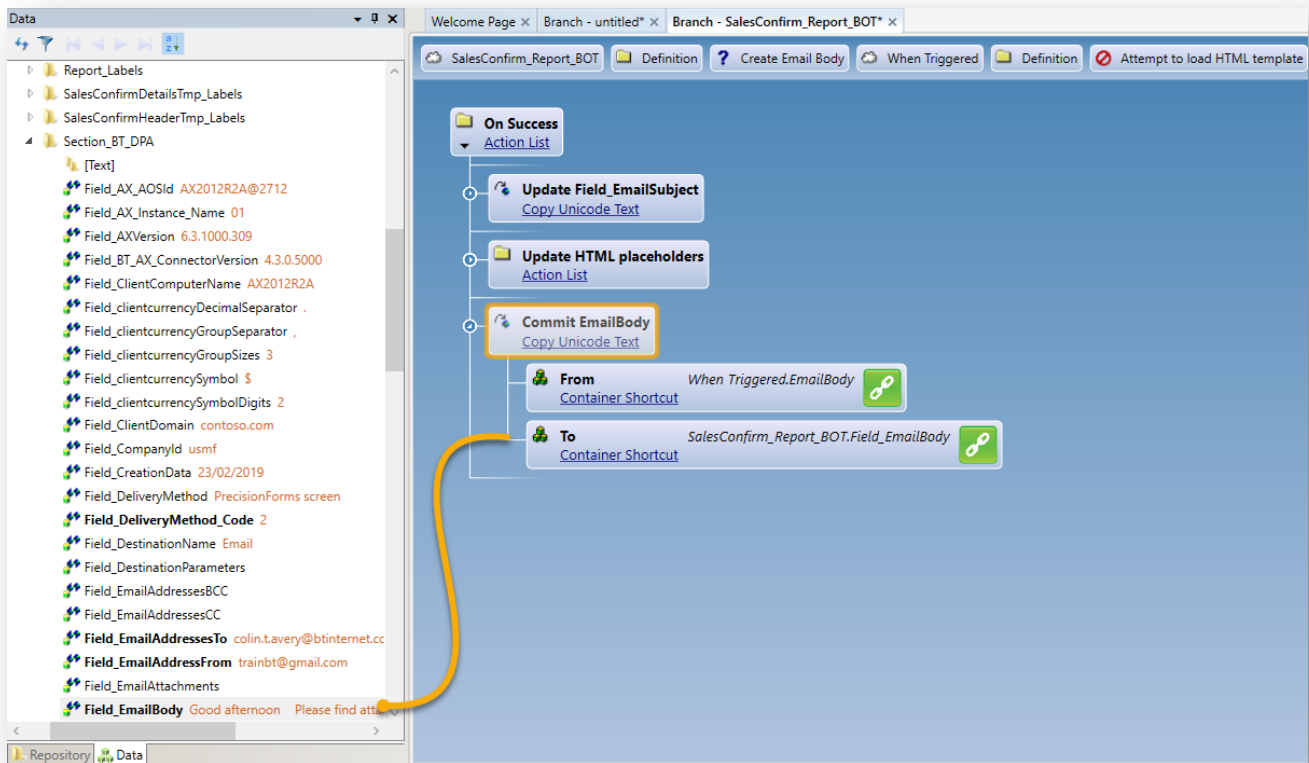


## Commit EmailBody – Copy Unicode Text object

Finally we need to commit the EmailBody, add a [Copy Unicode Text](#) object to copy:


**From** - [Container Shortcut](#) to **EmailBody** memory [Unicode Text](#) object.

**To** - [Container Shortcut](#) to *Section\_BT\_DPA > Field\_EmailBody*.



## Sample Static Email Output

Once sent the static HTML email body is presented in the email client as illustrated, which includes Dynamics AX report data values.

 **Bottomline**

Sales Order Confirmation

Good afternoon,

Please find attached your Sales Order Confirmation in PDF format, details of which are outlined below.

*(In order to read a PDF document you will need Adobe Acrobat Reader. You can download Acrobat Reader for free from [www.adobe.com/products](http://www.adobe.com/products))*

### Order Summary

Order No.	Account No.	Order Date	Currency	Total
000721	US-024	23/02/2019	USD	\$ 93,019.35





### Contact Us

If you have any questions relating to this Sales Order, please contact us and we'll be happy to help you.

Email: [emea-info@bottomline.com](mailto:emea-info@bottomline.com)  
Tel: +44 (0)118 925 8250

Thank you for ordering from Bottomline Technologies

Maximise the value of your business documents



Transform 7.0

The Easiest and Most Efficient Way to Manage the Lifecycle of Your Financial Documents

PT-X Connect


Transform 7.0  
Document Management & Automation

PT-X Connect  
Connect Plus

Bottomline Technologies EMEA.

## Dynamic HTML Template

It is possible to create a dynamic HTML body which contains a table for the detail lines. You can build on the what you learnt in the previous Static HTML template to add the item lines.


**Bottomline**

Sales Order  
Confirmation

Good afternoon,

Please find attached your Sales Order Confirmation in PDF format, details of which are outlined below.

*(In order to read a PDF document you will need Adobe Acrobat Reader. You can download Acrobat Reader for free from [www.adobe.com/products](http://www.adobe.com/products))*

### Order Summary

Order No.	Account No.	Order Date	Currency	Total
000721	US-024	23/02/2019	USD	\$ 93,019.35

### Order Details

Item	Description	Qty	Amount
D0001	Mid-Range Speaker	5	2,400.00
L0001	Mid-Range Speaker 2	5	2,500.00
D0003	Standard Speaker	6	1,320.00
T0001	Speaker cable 10	15	7,500.00
T0004	Television M120 37" Silver	7	2,450.00
T0002	Projector Television	18	67,500.00
T0003	Surround Sound Receiver	10	4,500.00
T0003	Surround Sound Receiver	10	4,500.00
T0003	Surround Sound Receiver	10	4,500.00

### Contact Us

If you have any questions relating to this Sales Order, please contact us and we'll be happy to help you.

Email: [emea-info@bottomline.com](mailto:emea-info@bottomline.com)  
Tel: +44 (0)118 925 8250

Thank you for ordering from Bottomline Technologies

## Challenge

If you are up to the challenge discuss with the group how you would approach adding the item lines to the HTML Email.

## Resources

To complete the Email layout shown within this manual you can use the following HTML templates and Dynamics AX report data file. Copy the resources to your filesystem so you can access them from the Transform process.

### Static Template



SalesConfirm\_Repo  
rt\_BOT.html

### Dynamic Template



SalesConfirm\_Repo  
rt\_BOT\_Lines.html

### Dynamics AX Report Data



SalesConfirm;admin  
;23022019\_082748\_1