MS Template Creator

Developer Documentation

Version 0.0.5

2018

# Version Control Table

|  |  |
| --- | --- |
| **Title** | MS Template Creator Developer Documentation |
| **Created By** | Jeremy John Selva |
| **Date Created** | 27 March 2018 |
| **Maintained By** | Jeremy John Selva |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version Number** | **Modified By** | **Date Modified** | **Modifications Made** | **Status** |
| 0.0.5 | Jeremy John | 14 Aug 2021 | Changed   * 7 b) i “Button” module * 10 Code Update guidelines | Draft |
| 0.0.4 | Jeremy John | 9 Mar 2021 | Added   * 7 a) vii Load\_Tidy\_Data module * 7 d) ii Load\_Tidy\_Data\_Test module * 7 f) vii Load\_Sample\_Annot\_Raw * 7 f) viii Load\_Sample\_Annot\_Tidy * 8 a) iv Categorising Unit Test | Draft |
| 0.0.3 | Jeremy John | 28 Jan 2021 | Added   * 5a) iv 64-bit compatibility * 10d) Dilution\_Annot changes * 10e) Additional help | Draft |
| 0.0.2 | Jeremy John | 21 Aug 2018 | Added   * 2. Code Repository * 5. Excel Add-Ins * 8. Unit and Integration Test * 9. Version Control   Changed   * 6. Code Structure Overview * 7. Code Structure Details | Draft |
| 0.0.1 | Jeremy John | 3 May 2018 | Initial commit and review | Draft |

# Content Page

[Version Control Table 1](#_Toc79836322)

[Content Page 2](#_Toc79836323)

[1. Introduction 4](#_Toc79836324)

[a) Purpose 4](#_Toc79836325)

[b) Scope 4](#_Toc79836326)

[c) Displaying the developer tab on the toolbar 4](#_Toc79836327)

[2. Code Repository 4](#_Toc79836328)

[3. Excel Column Settings 4](#_Toc79836329)

[4. Excel Buttons and Textboxes Settings 5](#_Toc79836330)

[a) Buttons Settings 5](#_Toc79836331)

[i. Code in the Buttons 5](#_Toc79836332)

[ii. Moving Buttons 5](#_Toc79836333)

[iii. Buttons format 5](#_Toc79836334)

[b) Textboxes Settings 6](#_Toc79836335)

[5. Excel Add-Ins 6](#_Toc79836336)

[a) Rubberduck 6](#_Toc79836337)

[b) Excel VBA Developer Tools 7](#_Toc79836338)

[i. Microsoft Visual Basic for Applications Extensibility 5.3 7](#_Toc79836339)

[ii. Trust access to Visual Basic Project 8](#_Toc79836340)

[iii. Enable the Excel VBA Developer Tools 9](#_Toc79836341)

[iv. 64-bit compatibility 10](#_Toc79836342)

[6. Code Structure Overview 12](#_Toc79836343)

[7. Code Structure Details 13](#_Toc79836344)

[a) Vba Dependencies 13](#_Toc79836345)

[b) Annotation Creating Modules 13](#_Toc79836346)

[i. “Button” module 13](#_Toc79836347)

[ii. Utilities module 13](#_Toc79836348)

[iii. Transition\_Name\_Annot module 13](#_Toc79836349)

[iv. ISTD\_Annot module 13](#_Toc79836350)

[v. Sample\_Annot module 13](#_Toc79836351)

[vi. Load\_Raw\_Data module 14](#_Toc79836352)

[vii. Load\_Tidy\_Data module 14](#_Toc79836353)

[viii. Sample\_Type\_Identifier module 14](#_Toc79836354)

[c) Sheet Tracking Modules 14](#_Toc79836355)

[i. ColourTracker 14](#_Toc79836356)

[d) Unit Test Modules 14](#_Toc79836357)

[i. Load\_Raw\_Data\_Test module 14](#_Toc79836358)

[ii. Load\_Tidy\_Data\_Test module 14](#_Toc79836359)

[iii. Sample\_Type\_Identifier\_Test module 14](#_Toc79836360)

[iv. Utilities Test module 14](#_Toc79836361)

[e) Integration Test Modules 14](#_Toc79836362)

[i. Integration\_Test module 14](#_Toc79836363)

[f) Forms 14](#_Toc79836364)

[i. Clear\_Dilution\_Annot 14](#_Toc79836365)

[ii. Clear\_ISTD\_Table 14](#_Toc79836366)

[iii. Clear\_Sample\_Annot 14](#_Toc79836367)

[iv. Clear\_Transition\_Name\_Annot 14](#_Toc79836368)

[v. Concentration\_Unit\_MsgBox 14](#_Toc79836369)

[vi. Invalid\_ISTD \_MsgBox 15](#_Toc79836370)

[vii. Load\_Sample\_Annot\_Raw 15](#_Toc79836371)

[viii. Load\_Sample\_Annot\_Tidy 15](#_Toc79836372)

[ix. Overwrite 15](#_Toc79836373)

[8. Unit and IntegrationTest 15](#_Toc79836374)

[a) Rubberduck for unit testing 15](#_Toc79836375)

[i. Initialising the Rubberduck 15](#_Toc79836376)

[ii. Opening the Unit Test 16](#_Toc79836377)

[iii. Adding a new unit test module 17](#_Toc79836378)

[iv. Categorising Unit Test 17](#_Toc79836379)

[v. Running the Unit Test 18](#_Toc79836380)

[b) Integration Test 19](#_Toc79836381)

[9. Version Control 20](#_Toc79836382)

[a) Exporting modules using Excel VBA Developer Tools 20](#_Toc79836383)

[b) Importing modules using Excel VBA Developer Tools 21](#_Toc79836384)

[10. Code Update guidelines 22](#_Toc79836385)

[a) Transition\_Name\_Annot changes 22](#_Toc79836386)

[i. Changes in the name of the column where the transition name is placed 22](#_Toc79836387)

[ii. Changes in the name of the column where the transition name ISTD is placed 23](#_Toc79836388)

[iii. Changes in the sheet name 25](#_Toc79836389)

[b) ISTD\_Annot changes 26](#_Toc79836390)

[i. Changes in the name of the column where the transition name ISTD is placed 26](#_Toc79836391)

[ii. Changes in the column name used in the calculation of ISTD concentration from ng/ML to nM 27](#_Toc79836392)

[iii. Changes in the sheet name 29](#_Toc79836393)

[c) Sample\_Annot changes 29](#_Toc79836394)

[i. Changes in the column names for MRM transition name data input 29](#_Toc79836395)

[ii. Changes in the column names for sample annotation input 31](#_Toc79836396)

[iii. Changes in the sheet name 32](#_Toc79836397)

[iv. Changes in the listbox in Sample\_Type and Sample\_Amount\_Unit 33](#_Toc79836398)

[d) Dilution\_Annot changes 33](#_Toc79836399)

[i. Changes in the name of the column where raw data file name and sample name is placed. 33](#_Toc79836400)

[ii. Changes in the sheet name 34](#_Toc79836401)

[e) Additional help 35](#_Toc79836402)

# Introduction

## Purpose

This document highlights the Excel VBA code structure for MS Template Creator.

The excel macro is created to provide users a more friendly interface to take in raw data exported directly from the mass spectrometry machines and create several annotation templates suited for automated data processing and statistical analysis.

## Scope

This guide will assist developers to understand the organisation of the code structure. We are still working on making the codes more modular so that changes in the feature of the MS Template Creator will require minimal efforts

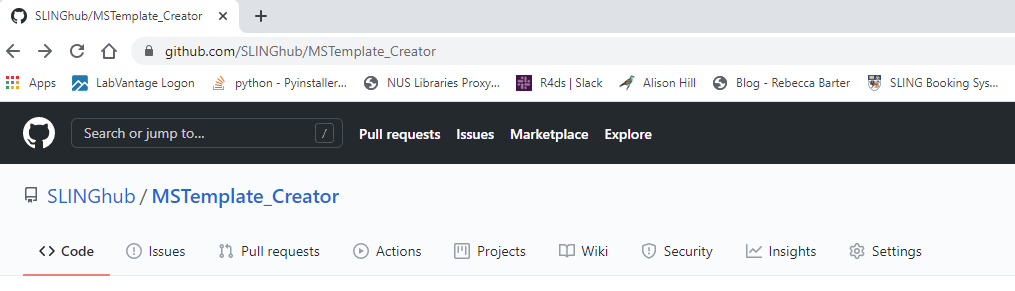
## Displaying the developer tab on the toolbar

As the excel sheet contains macros, to modify the codes, the developer tab must be present in the toolbar.

Refer to <https://www.excel-easy.com/examples/developer-tab.html> to display the developer tab

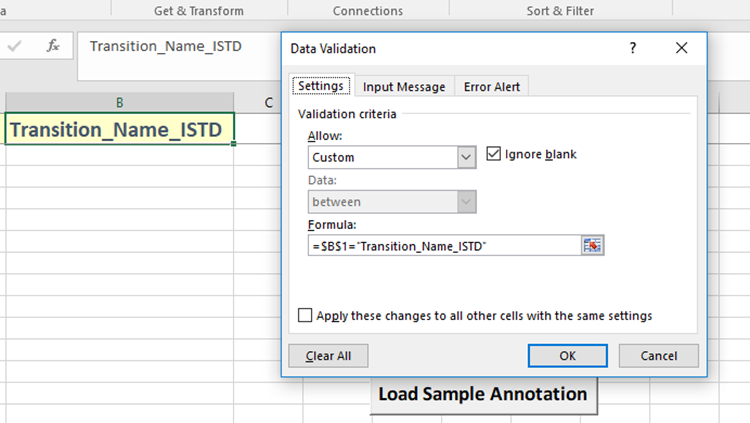
# Code Repository

The codes can be found in <https://github.com/SLINGhub/MSTemplate_Creator>



# Excel Column Settings

All headers in the excel column are fixed using Data validation



To change the column names in the excel sheet, you must disable the validation criteria first.

Please note that by changing the column names, the code in the macro file will be needed to be updating in order to be in working condition.

The updating steps can be found on Section “Code Updating Guidelines”

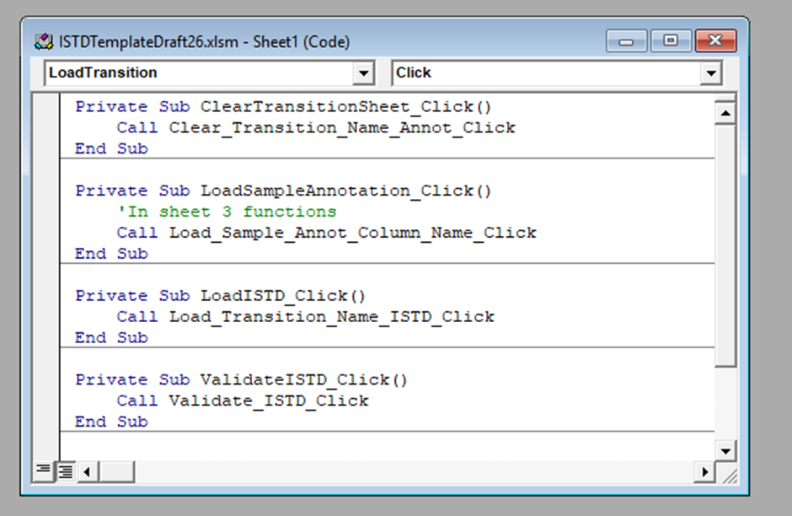
# Excel Buttons and Textboxes Settings

## Buttons Settings

### Code in the Buttons

Buttons in excel are Active X control to ensure that they can be moved around freely. However, their codes will be sheet specific

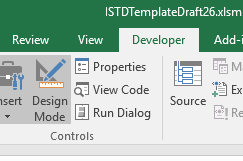
Deleting the sheet will cause these sheet specific function to be deleted. Remember to back up the code before deleting the script



All functions in the sheet leads to the Button Module.

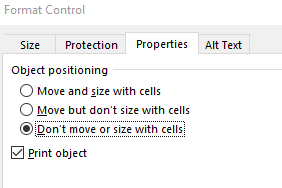
### Moving Buttons

To move the button, go to the Developer tab and click on “Design Mode”



### Buttons format

All buttons are set to this format property

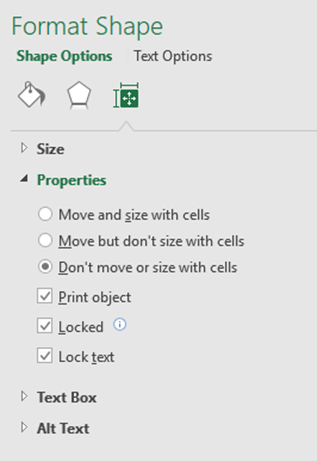


We take into consideration that users will change the number of rows more often than to change the number of columns.

If the number of columns need to be changed of resized, the buttons will not move with the columns. To move the button manually, refer to Section “Moving Buttons”.

## Textboxes Settings

All textboxes are set to this format shape property



We take into consideration that users will change the number of rows more often than to change the number of columns.

If the number of columns need to be changed of resized, the textboxes will not move with the columns. There will be a need to move the textbox manually.

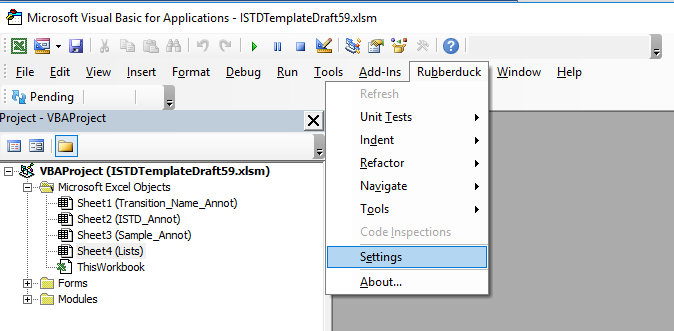
# Excel Add-Ins

## Rubberduck

The RubberDuck is used to create a fully integrated test environment for the VBA codes in MS Template Creator.

It can be installed in <http://rubberduckvba.com/>

A successful installation will give an additional “Rubberduck” tab in Microsoft Visual Basic for Applications

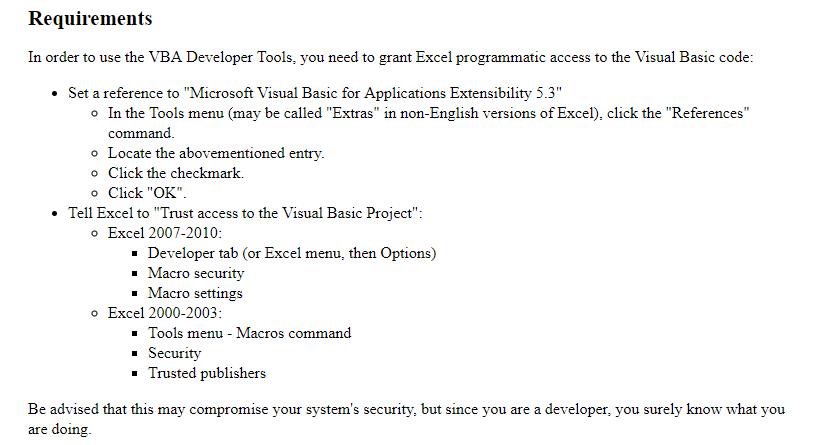


## Excel VBA Developer Tools

The Excel VBA Developer Tools is used to convert the VBA codes in the Excel macro file to text files. These text files can then be monitored by Git and send to Bitbucket for Version Control of the VBA scripts.

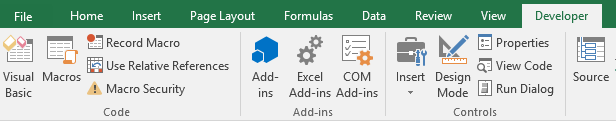
It can be installed in http://vbatools.sourceforge.net/

Please following the instructions given in the webpage carefully.

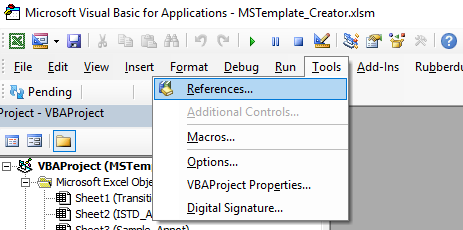


### Microsoft Visual Basic for Applications Extensibility 5.3

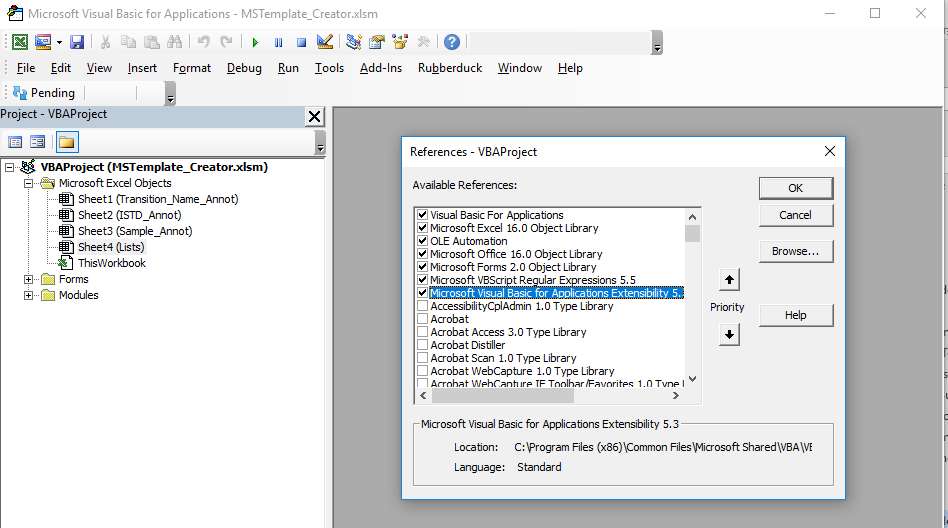
To do so open the Visual Basic in Excel in the Developer Tab.



Click on Tools -> Reference

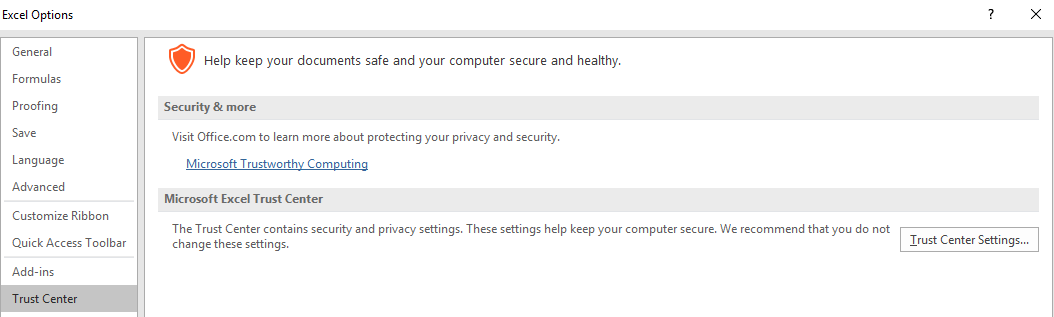


Look for Microsoft Visual Basic for Applications Extensibility 5.3 and click on the check box.

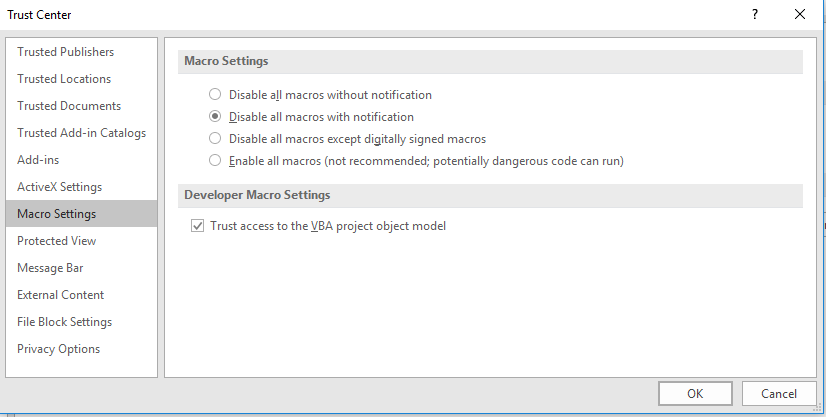


### Trust access to Visual Basic Project

To do so go to File tab. Click on Options. Click on Trust Center



Click on Trust Center Settings and then Macro Settings. Check the box Trust access to the VBA project object model. Click Ok



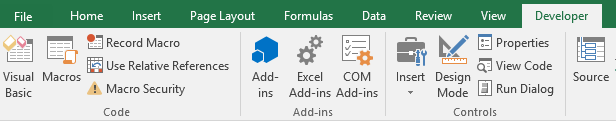
### Enable the Excel VBA Developer Tools

When the Excel VBA Developer Tools is downloaded, it will appear as an Excel Add-In file.

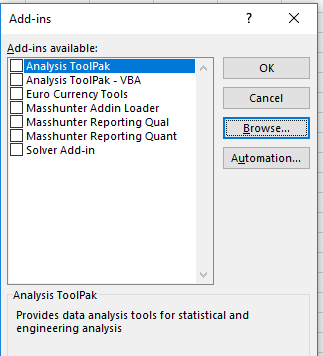


You must manually add this Excel Add In file when you need to use it.

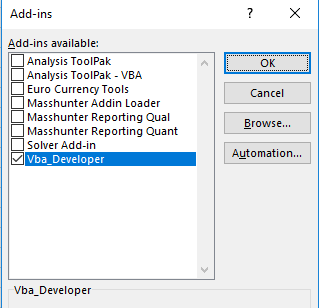
To do so, on the Developer Tab, click on “Excel Add-Ins”



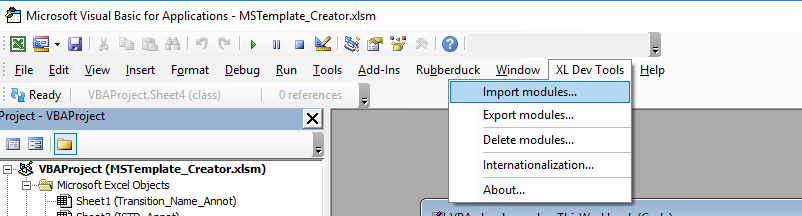
Browse for that Excel Add-In file



When it is added, it will look like this. Click Ok



To check that it is working, open the Visual Basic in the Developer Tab. You will see an additional tab called “XL Dev Tools”



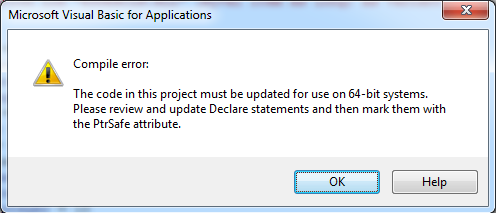
Note:

Each time that you close Microsoft Excel, this Add-In will be disabled automatically even though there is a tick on the check box.

To call it again, just uncheck the box, click Ok. Reopen the Excel Add-In box, rechecked the VBA Developer box and click Ok.

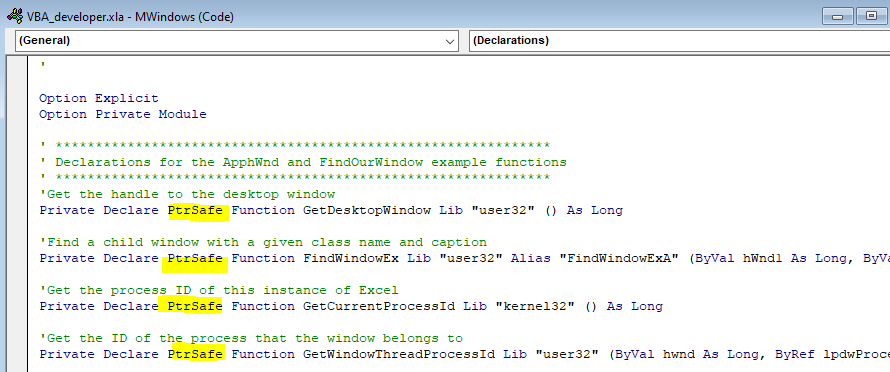
### 64-bit compatibility

Unfortunately, Excel VBA Developer Tools is meant to be used with 32-bit application. When you load the Add in for the first time, the following error may pop up



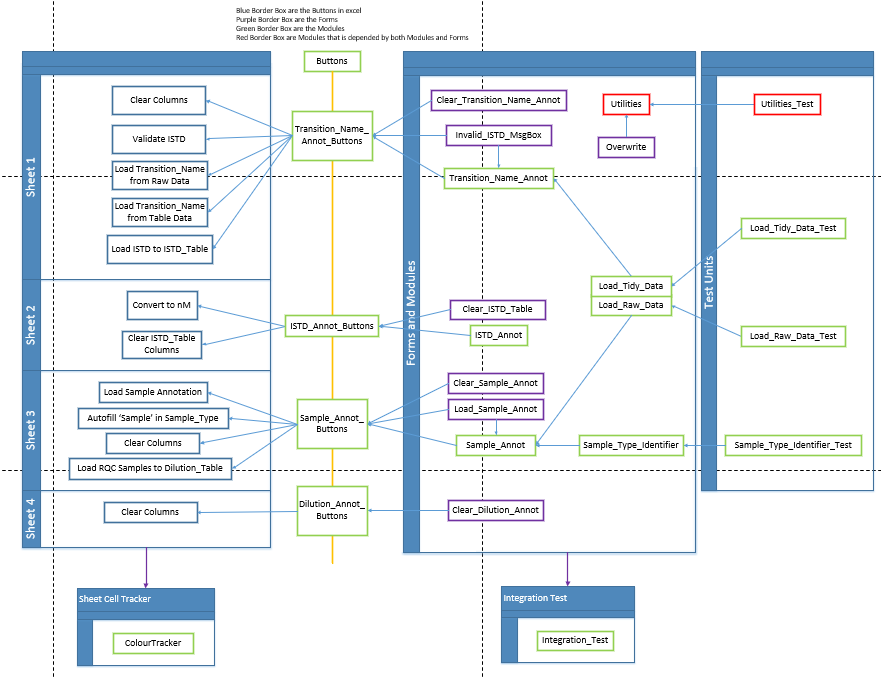
From <https://docs.microsoft.com/en-us/office/vba/language/concepts/getting-started/64-bit-visual-basic-for-applications-overview> ,

The key to resolve this error is to add the “PtrSafe” keyword beside each Declare in the source code of Excel VBA Developer Tools



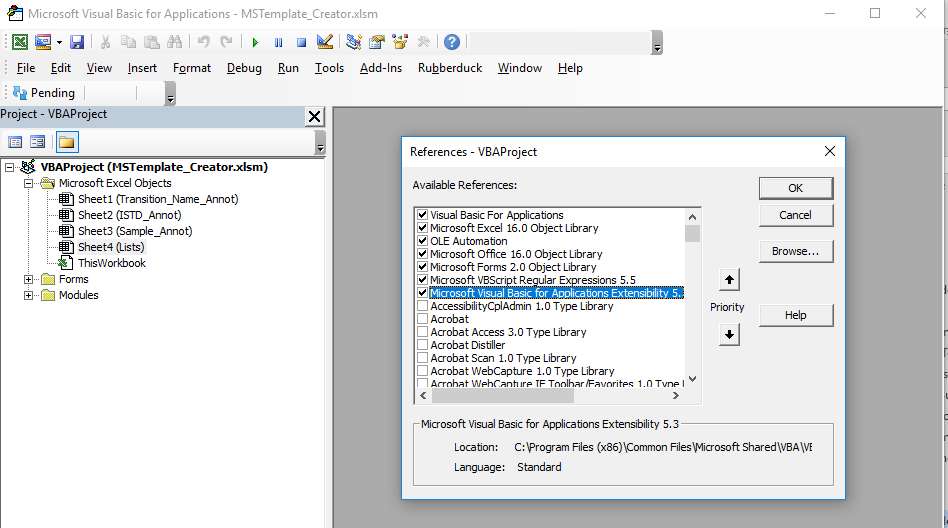
More info in <https://docs.microsoft.com/en-us/office/vba/language/reference/user-interface-help/ptrsafe-keyword>

# Code Structure Overview



# Code Structure Details

## Vba Dependencies



## Annotation Creating Modules

### “Button” module

The Button module serves as an interface/wrapper between the Active X buttons in the excel sheets to the modules are user forms in e\Excel VBA.

Please ensure that new Active X buttons are directed to this module first before linking it to other modules.

A total of four button modules are created for each excel sheet.

* Transition\_Name\_Annot\_Buttons
* ISTD\_Annot\_Buttons
* Sample\_Annot\_Buttons
* Dilution\_Annot\_Buttons

### Utilities module

The Utilities module are a collection of functions that is mostly used throughout the excel macro file. All forms and modules depend on this Utilities module. Please change this file with caution.

### Transition\_Name\_Annot module

Functions that are applied only the sheet “Transition\_Name\_Annot” should be written in this module.

### ISTD\_Annot module

Functions that are applied only the sheet “ISTD\_Annot” should be written in this module.

### Sample\_Annot module

Functions that are applied only the sheet “Sample\_Annot” and to create arrays directly from Sample Annotation files should be written in this module.

### Load\_Raw\_Data module

Functions that are designed to create arrays directly from Mass spec raw data files are to be written there.

### Load\_Tidy\_Data module

Functions that are designed to create arrays from csv files that in a in table form.

### Sample\_Type\_Identifier module

Functions to help identify the Sample Type from the Sample Name using regular expression are to be written there.

## Sheet Tracking Modules

### ColourTracker

Functions to add colours to cell when it has been modified unnecessarily. It only runs on Sheet 1 (Transition\_Name\_Annot) and Sheet 2 (ISTD\_Annot)

## Unit Test Modules

### Load\_Raw\_Data\_Test module

Functions to test the Load\_Raw\_Data module for logical errors.

### Load\_Tidy\_Data\_Test module

Functions to test the Load\_Tidy\_Data module for logical errors.

### Sample\_Type\_Identifier\_Test module

Functions to test the Sample\_Type\_Identifier module for logical errors

### Utilities Test module

Functions to test the Utilities module for logical errors.

## Integration Test Modules

### Integration\_Test module

Functions to run an integration test for MS Template Creator.

## Forms

### Clear\_Dilution\_Annot

UserForm and Functions to clear columns in sheet “Dilution\_Annot”

### Clear\_ISTD\_Table

UserForm and Functions to clear columns in sheet “ISTD\_Annot”

### Clear\_Sample\_Annot

UserForm and Functions to clear columns in sheet “Sample\_Annot”

### Clear\_Transition\_Name\_Annot

UserForm and Functions to clear columns in sheet “Transition\_Name\_Annot”

### Concentration\_Unit\_MsgBox

UserForm and Funtions to indicate what concentration units are generated automatically in sheet “Sample\_Annot”

### Invalid\_ISTD \_MsgBox

UserForm and Functions to highlight to users of potential invalid ISTD

### Load\_Sample\_Annot\_Raw

UserForm and Functions to help users load Sample Annotation from raw data file

### Load\_Sample\_Annot\_Tidy

UserForm and Functions to help users load Sample Annotation from csv files in table form.

### Overwrite

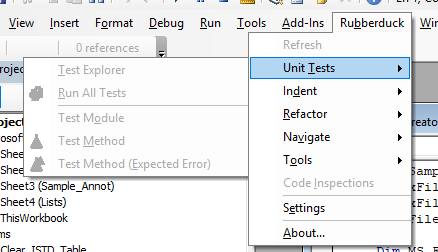
UserForm and Functions to ask users if a particular column should be overwritten or not.

# Unit and IntegrationTest

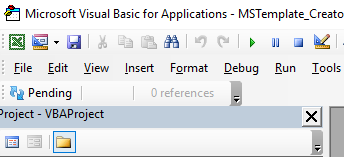
## Rubberduck for unit testing

### Initialising the Rubberduck

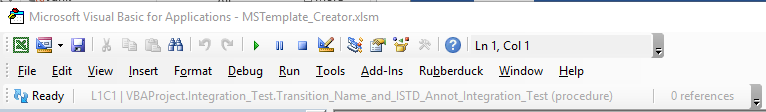
When you open the Visual Basic from the Developer Tab, Rubberduck is not activated yet.



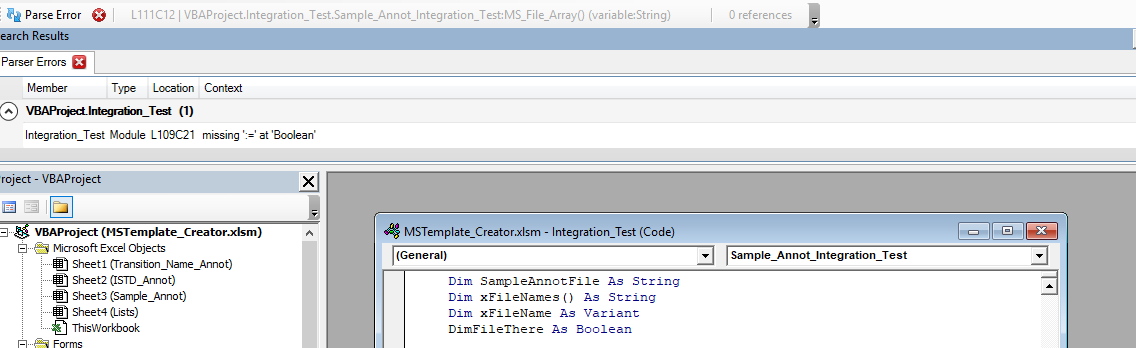
To start the Rubberduck, click on “Pending”



When it appears as “Ready”, you can start the unit testing



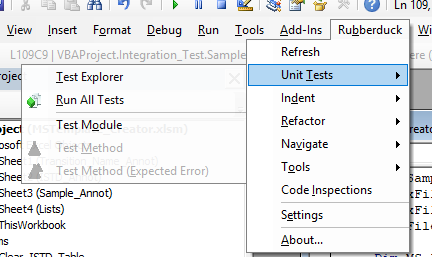
If it appears like this



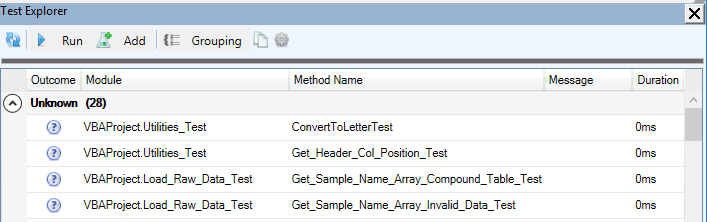
It means you have a syntax error (Should be “Dim FileThere” instead of “DimFileThere”) and Rubberduck clearly cannot proceed with unit testing on codes with bugs.

### Opening the Unit Test

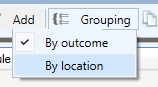
To start the unit testing, go to Rubberduck, Unit tests, Test Explorer



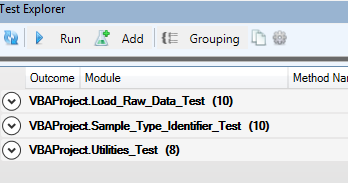
You will see this



You will need to Group the unit test according to their respective modules. Click on Grouping and then “By Location”

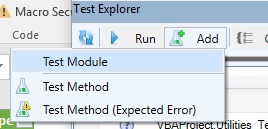


You will see the unit test are grouped based on the unit test modules

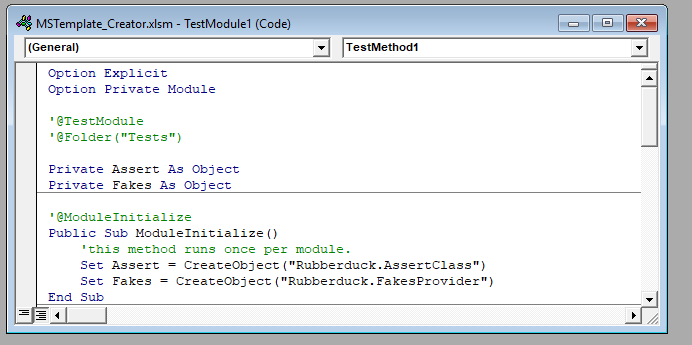


### Adding a new unit test module

Go to Add, Click on Test Module



A new test module will be automatically created



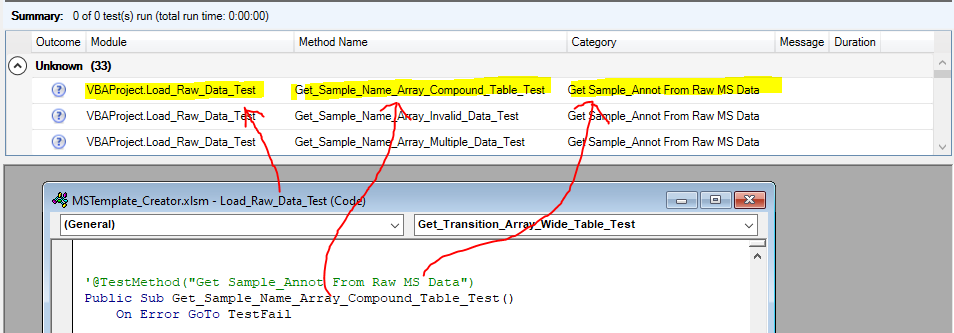
Refer to the documentation to see how to write a unit test properly

<https://github.com/rubberduck-vba/Rubberduck/wiki/Unit-Testing>

### Categorising Unit Test

To group unit test into the same category, put the group name beside

‘@TestMethod({Your group name})

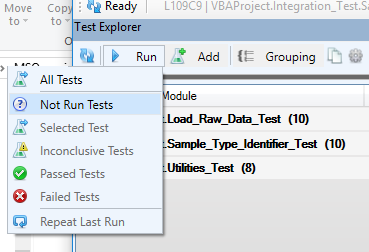


### Running the Unit Test

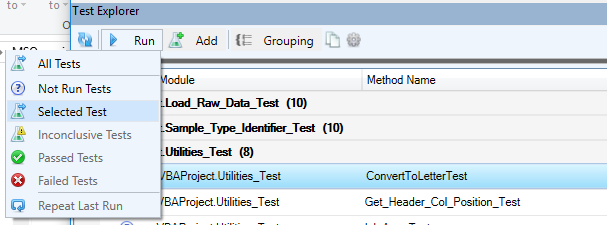
Click on the refresh button to update the Test Explorer to find any newly created unit test



To run all the unit test, Click on Run, then all test



To run a specific test, highlight a specific test, click on run and then “Selected test”



A passed test will look like this



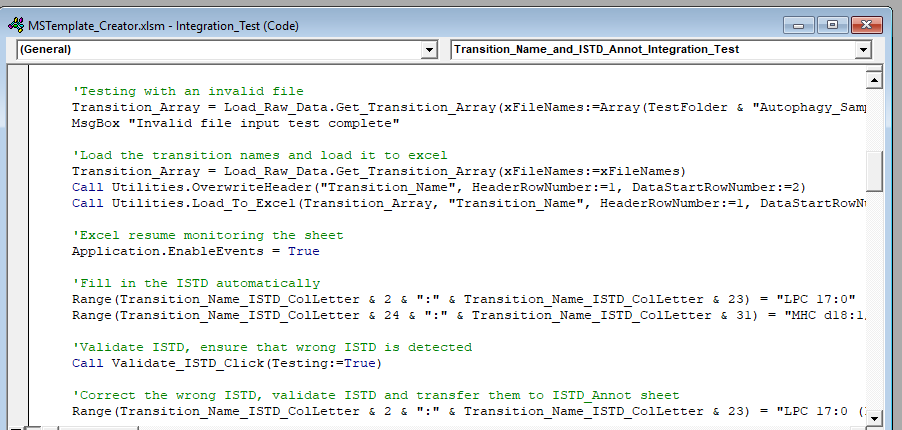
A failed test will look like this



## Integration Test

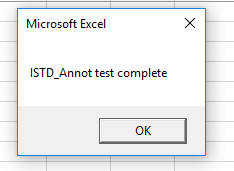
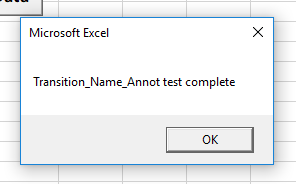
Open the Intergration\_Test module and run all the functions in this module.

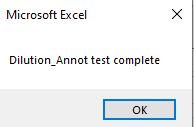
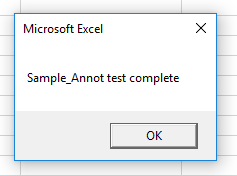
What each test is doing can be found in the comments of the code



Click Ok for every message box that appears.

Make sure all sheets are tested successfully

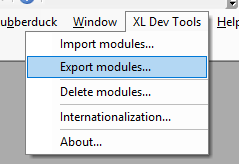




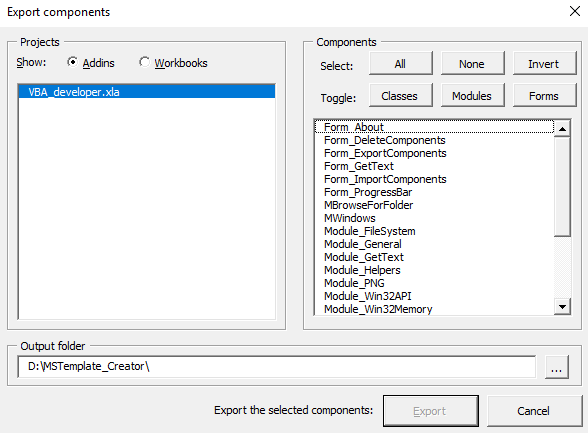
# Version Control

## Exporting modules using Excel VBA Developer Tools

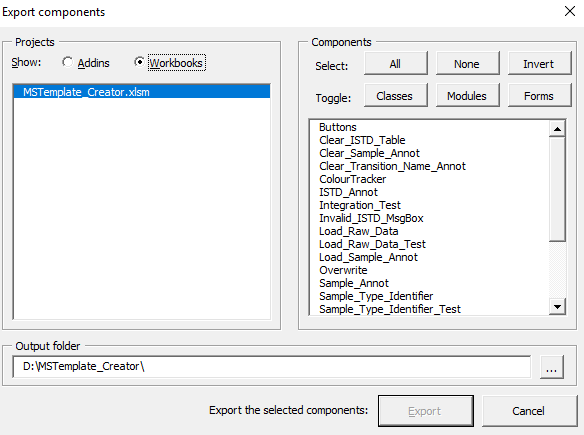
When you open the Visual Basic from the Developer Tab, click on XL Dev Tools, Export modules



The following box will pop up



Click on Workbooks, make sure the MSTemplate\_Creator macro file is selected and you can see all its modules.

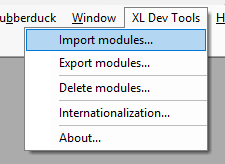


Select any modules you wish to export and click on export.

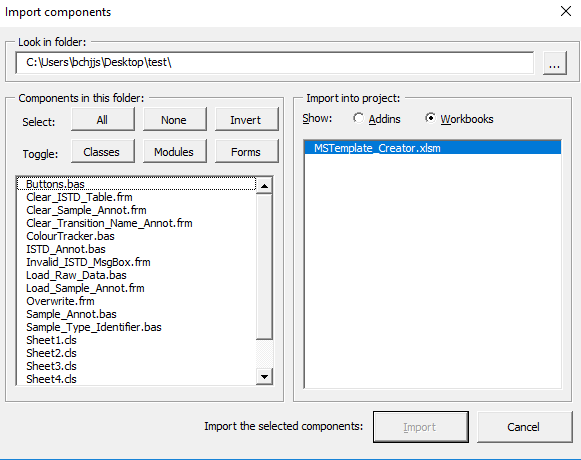
Modules will be exported as text files which Git can handle to do version control and send to a remote repository.

## Importing modules using Excel VBA Developer Tools

You can also import modules when you open the Visual Basic from the Developer Tab, click on XL Dev Tools, Import modules



Locate the folder with the module text files and import them to the correct workbook.



Note:

The software however is unable to copy cls type files, so copy and paste is required for this part.

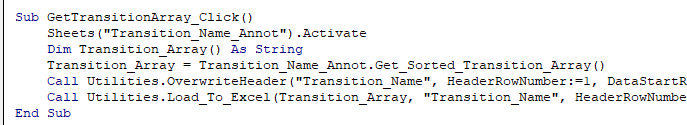
# Code Update guidelines

Below are some guiding steps to change in the code should there be any small minor changes needed.

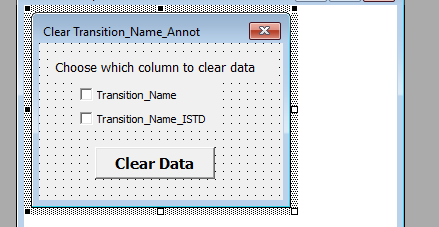
## Transition\_Name\_Annot changes

### Changes in the name of the column where the transition name is placed

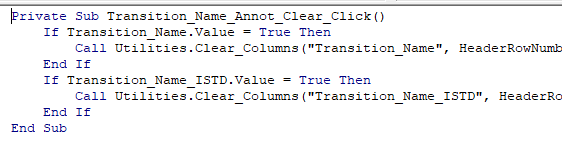
In the Transition\_Name\_Annot\_Buttons Modules, go to GetTranstionArray\_Click function and update the function Utilities.OverwriteHeader and Utilities.Load\_To\_Excel



In the Userform Clear\_Transition\_Name\_Annot, change the column name in



Change the relevant code for the userform



In the Colour Tracker module, , go to ChangeToBlankWhenChanged function and update the function Utilities.Get\_Header\_Col\_Position and Utilities.ConvertToLetter

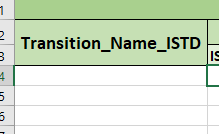
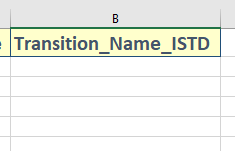


In the Integration\_Test module, go to Transition\_Name\_and\_ISTD\_Annot\_Integration\_Test and update the functions such as Utilities.Get\_Header\_Col\_Position, Utilities.ConvertToLetter, etc

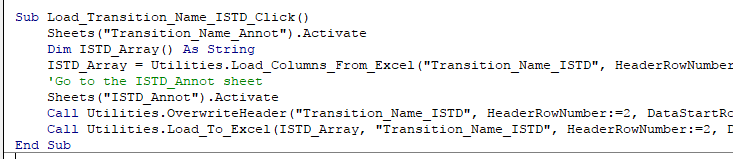


### Changes in the name of the column where the transition name ISTD is placed

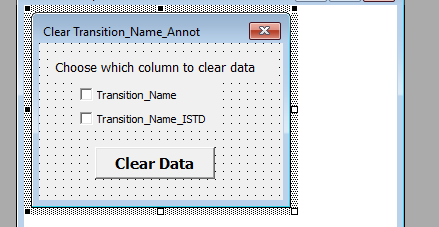
Firstly, you must change the column in both the sheet Transition\_Name\_Annot and ISTD\_Annot to the one you have changed



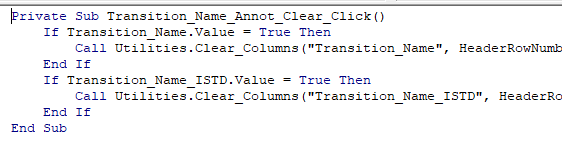
In the Transition\_Name\_Annot\_Button Modules, go to Load\_Transition\_Name\_ISTD\_Click function and change the function, Utilities.Load\_Columns\_From\_Excel, Utilities.OverwriteHeader and Utilities.Load\_To\_Excel



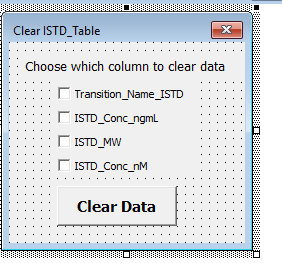
In the Userform Clear\_Transition\_Name\_Annot, change the column name in



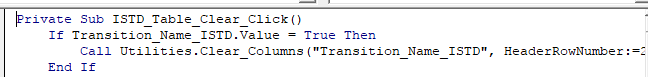
Change the relevant code for the userform



In the userform Clear ISTD\_Table, change the column name in



Change the relevant code for the userform



In the Colour Tracker module, , go to ChangeToBlankWhenChanged function and update the function Utilities.Get\_Header\_Col\_Position and Utilities.ConvertToLetter



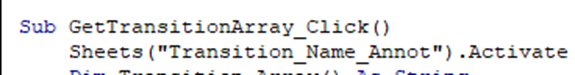
In the Integration\_Test module, go to Transition\_Name\_and\_ISTD\_Annot\_Integration\_Test and update the functions such as Utilities.Get\_Header\_Col\_Position, Utilities.ConvertToLetter, etc



### Changes in the sheet name

In the Button Modules, go to Load\_Transition\_Name\_ISTD\_Click function and GetTranstionArray\_Click function change the sheet name in the command





In the Colour Tracker module, go to ChangeToBlankWhenChanged function and update the sheet name to activate



In the Integration\_Test module, go to Transition\_Name\_and\_ISTD\_Annot\_Integration\_Test and update the sheet name.



## ISTD\_Annot changes

### Changes in the name of the column where the transition name ISTD is placed

Following the steps in 10a) part ii

In the Colour Tracker module , go to ISTDCalculationChecker function and update the function Utilities.Get\_Header\_Col\_Position and Utilities.ConvertToLetter

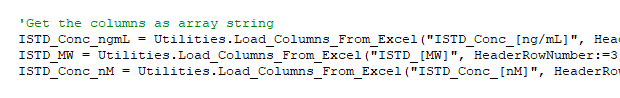


In the Integration\_Test module, go to Transition\_Name\_and\_ISTD\_Annot\_Integration\_Test and update the functions such as Utilities.Get\_Header\_Col\_Position, Utilities.ConvertToLetter, etc



### Changes in the column name used in the calculation of ISTD concentration from ng/ML to nM

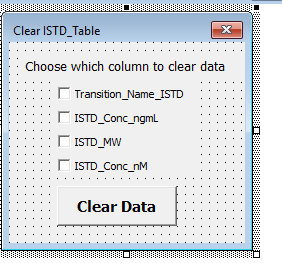
In the ISTD\_Annot module, go to Get\_ISTD\_Conc\_nM\_Array. Update the function Utilities.Load\_Columns\_From\_Excel



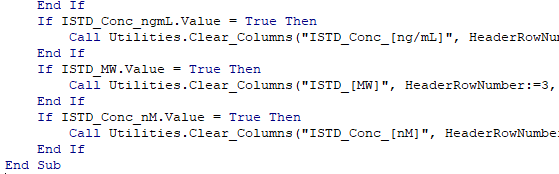
In the ISTD\_Annot\_Buttons module, go to nM\_Calculation\_Click. Update Utilities.Load\_To\_Excel “”ISTD\_Conc\_[nM]” if this column name is changed



In the userform Clear ISTD\_Table, change the column name in



Change the relevant code for the userform



In the Colour Tracker module , go to ISTDCalculationChecker function and update the function Utilities.Get\_Header\_Col\_Position and Utilities.ConvertToLetter

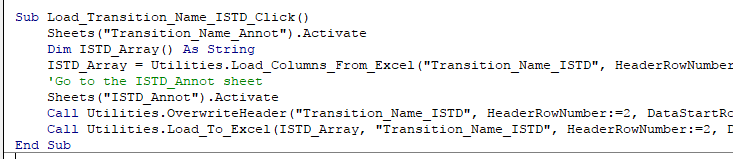


In the Integration\_Test module, go to Transition\_Name\_and\_ISTD\_Annot\_Integration\_Test and update the functions such as Utilities.Get\_Header\_Col\_Position, Utilities.ConvertToLetter, etc



### Changes in the sheet name

In the Button Modules, go to Load\_Transition\_Name\_ISTD\_Click function function change the sheet name in the command



In the Colour Tracker module , go to ISTDCalculationChecker function and update the sheet name to activate.



In the Integration\_Test module, go to Transition\_Name\_and\_ISTD\_Annot\_Integration\_Test and update the sheet name



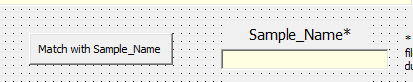
## Sample\_Annot changes

### Changes in the column names for MRM transition name data input

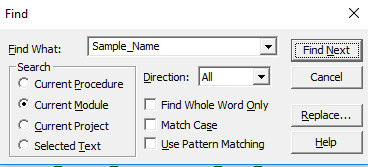
The column names affected are



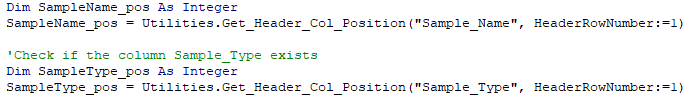
If Sample\_Name has been change, update the userfrom Load\_Sample\_Annot



And the corresponding codes with Sample\_Name involved if necessary. You may wish to use



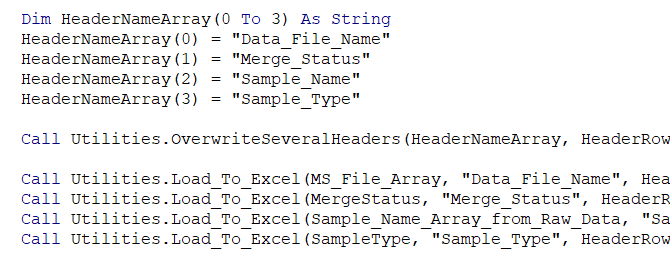
If Sample\_Name and Sample\_Type column has been changed, go to the Buttons module and update the function Autofill\_Sample\_Type\_Click()



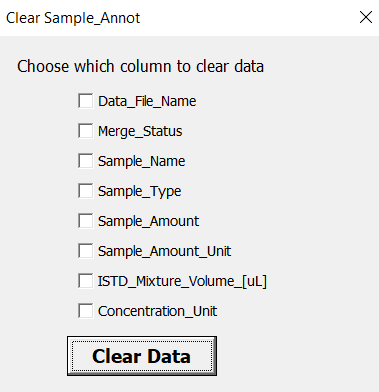


In the Sample\_Annot module, update the function

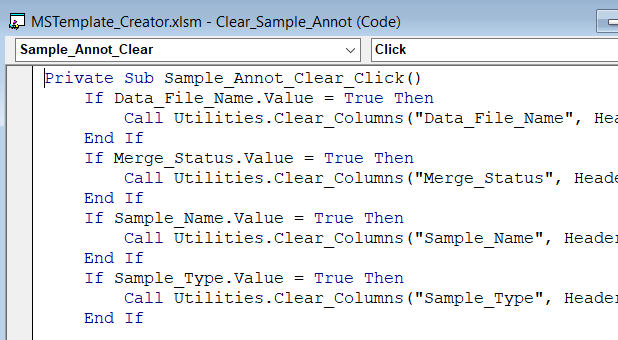
Create\_New\_Sample\_Annot\_Raw, Create\_New\_Sample\_Annot\_Tidy and Merge\_With\_Sample\_Annot



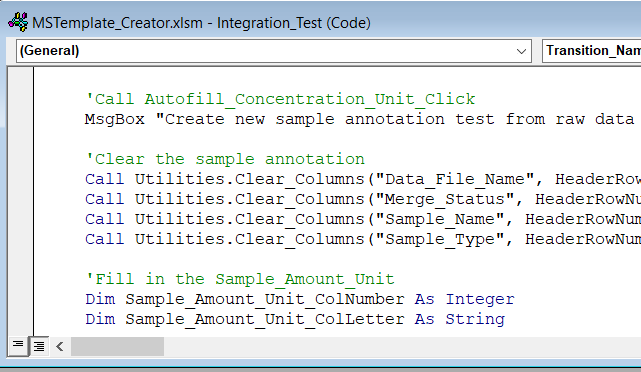
In the userform Clear\_Sample\_Annot, change the column name in



Change the relevant code for the userform

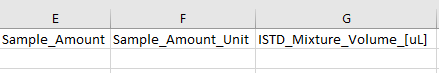


In the Integration\_Test module, go to Sample\_Annot\_Integration\_Test and Sample\_Annot\_and\_Dilution\_Annot\_Integration\_Test to update functions such as Utilities.Clear\_Columns

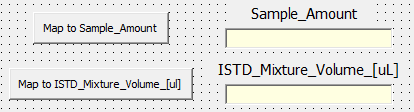


### Changes in the column names for sample annotation input

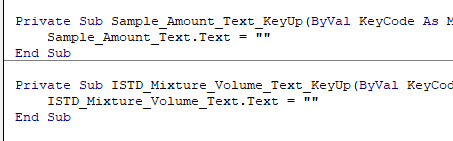
The column names affected are



If Sample\_Name has been change, update the userfrom Load\_Sample\_Annot

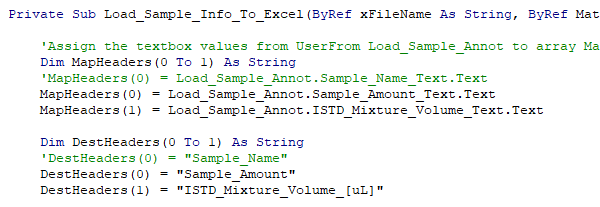


And the corresponding codes with Sample\_Amount and ISTD\_Mixture\_Volume\_[uL] involved if necessary.

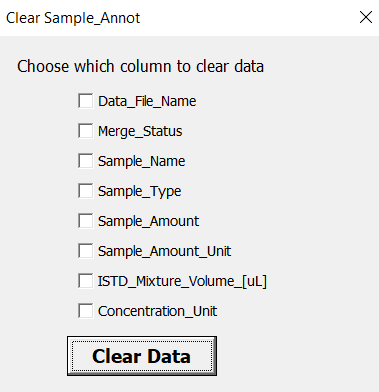


There will be a lot of changes. Use Ctrl-F to help you.

In the Sample\_Annot module, update the function Load\_Sample\_Info\_To\_Excel



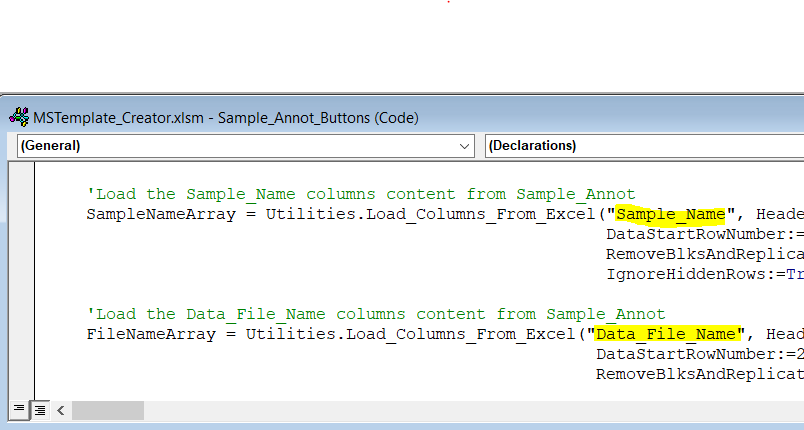
In the userform Clear\_Sample\_Annot, change the column name in



Change the relevant code for the userform

In the Sample\_Annot\_Buttons module, update the function Load\_Sample\_Name\_To\_Dilution\_Annot\_Click and make the changes where Utilities.Get\_Header\_Col\_Position and Utilities.Load\_Columns\_From\_Excel is called





### Changes in the sheet name

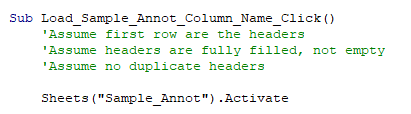
In the Sample\_Annot Modules, go to Create\_new\_Sample\_Annot and Merge\_With\_Sample\_Annot functions to change the sheet name





In the Button Modules, go to Autofill\_Sample\_Type\_Click, Load\_Sample\_Annot\_Column\_Name\_Click and Load\_Sample\_Name\_To\_Dilution\_Annot\_Click to change the sheet name

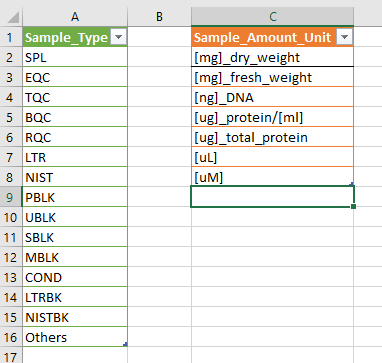




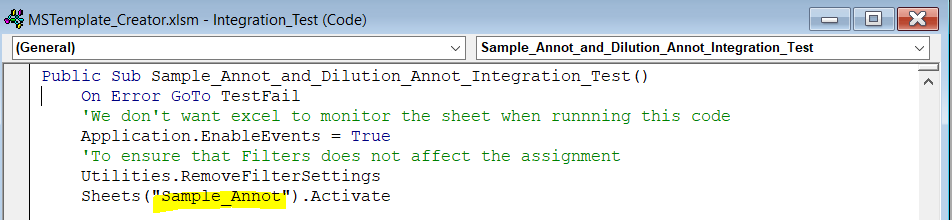


### Changes in the listbox in Sample\_Type and Sample\_Amount\_Unit

Go to the sheet “Lists” and update the table accordingly



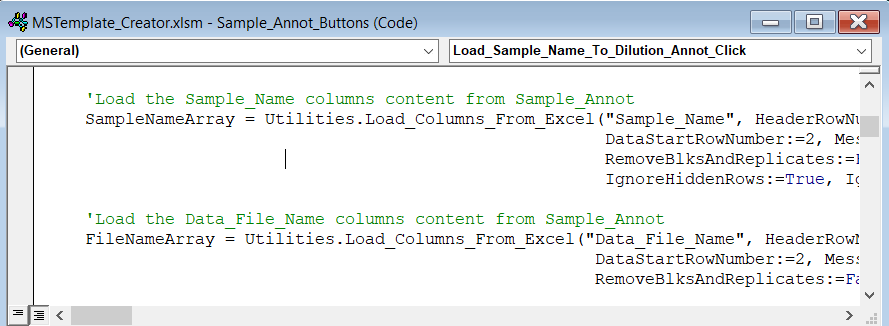
In the Integration\_Test module, go to Sample\_Annot\_Integration\_Test and Sample\_Annot\_and\_Dilution\_Annot\_Integration\_Test to update the sheet name



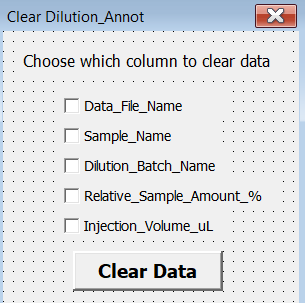
## Dilution\_Annot changes

### Changes in the name of the column where raw data file name and sample name is placed.

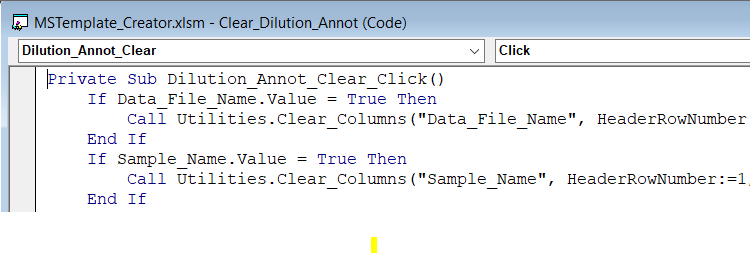
In the Sample\_Annot\_Buttons Modules, go to Load\_Sample\_Name\_To\_Dilution\_Annot\_Click function and update the function Utilities.OverwriteHeader and Utilities.Load\_To\_Excel



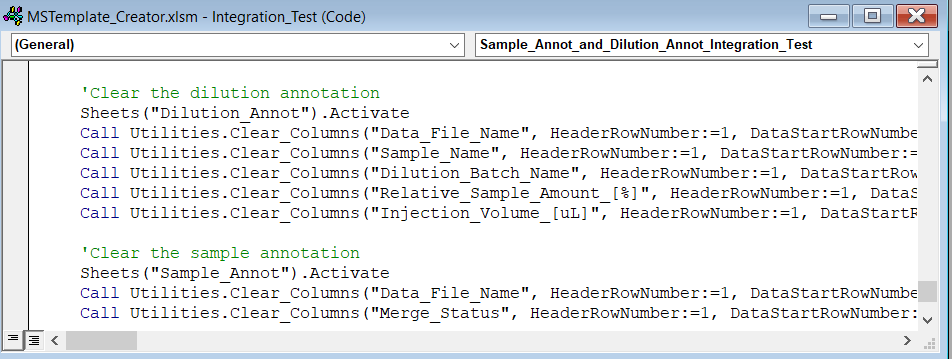
In the Userform Clear\_Dilution\_Annot, change the column name in



Change the relevant code for the userform

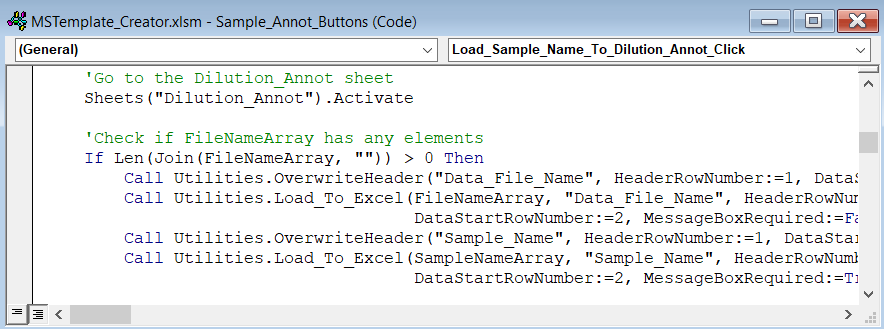


In the Integration\_Test module, go to Sample\_Annot\_and\_Dilution\_Annot\_Integration\_Test to update functions such as Utilities.Clear\_Columns

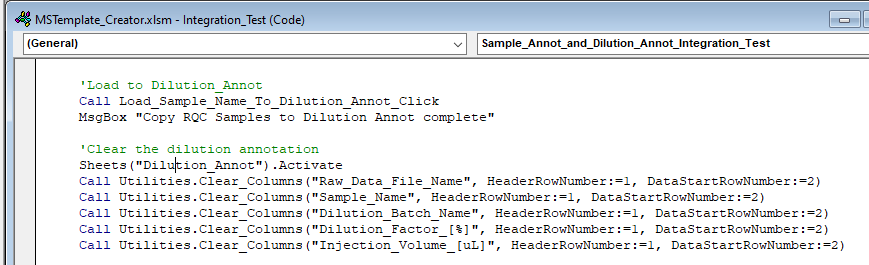


### Changes in the sheet name

In the Sample\_Annot\_Buttons Modules, go to Load\_Sample\_Name\_To\_Dilution\_Annot\_Click function and change the sheet name in the command



In the Integration\_Test module, go to Sample\_Annot\_and\_Dilution\_Annot\_Integration\_Test to update the sheet name to activate.



## Additional help

If you have troubles looking for a particular phrase, you may use the find button on Visual Basic or the Code Explorer in Rubberduck.

