IPI International

Working with C++ DLLs

Name	: multiplyBy
Type	: C++ DLL
Designer	: Paul I Ighofose
Date Created	: 15/01/2020
Date Updated	: 30/01/2020
Function	: Multiplies 2 variables passed in by each other and returns the result

Introduction

When writing Excel VBA code at work I have often wondered if some missing function could be written in another language and compiled to a DLL that VBA code can then make use of. Again, currently learning R Programming the idea keeps coming up.

Starting simple, a multiplication function would allow me to develop and test code in other programming languages loading the compiled DLL. At first I only developed console connect functions in R and Python; a VBA inputbox and msgbox in Outlook VBA; and VBA Functions that could be used to calculate row data as formulae in Excel and Ms Access.

After developing a PowerShell Windows Form I then re-addressed Outlook, Python and then R by amending their code to create GUIs similar to the PowerShell form. But the actual order of the forms development was:

- PowerShell Windows Form
- Outlook User Form using a Listbox
- Java JFrame and JTable Classes
- Python Tkinter GUI
- R Programming TCLTK GUI

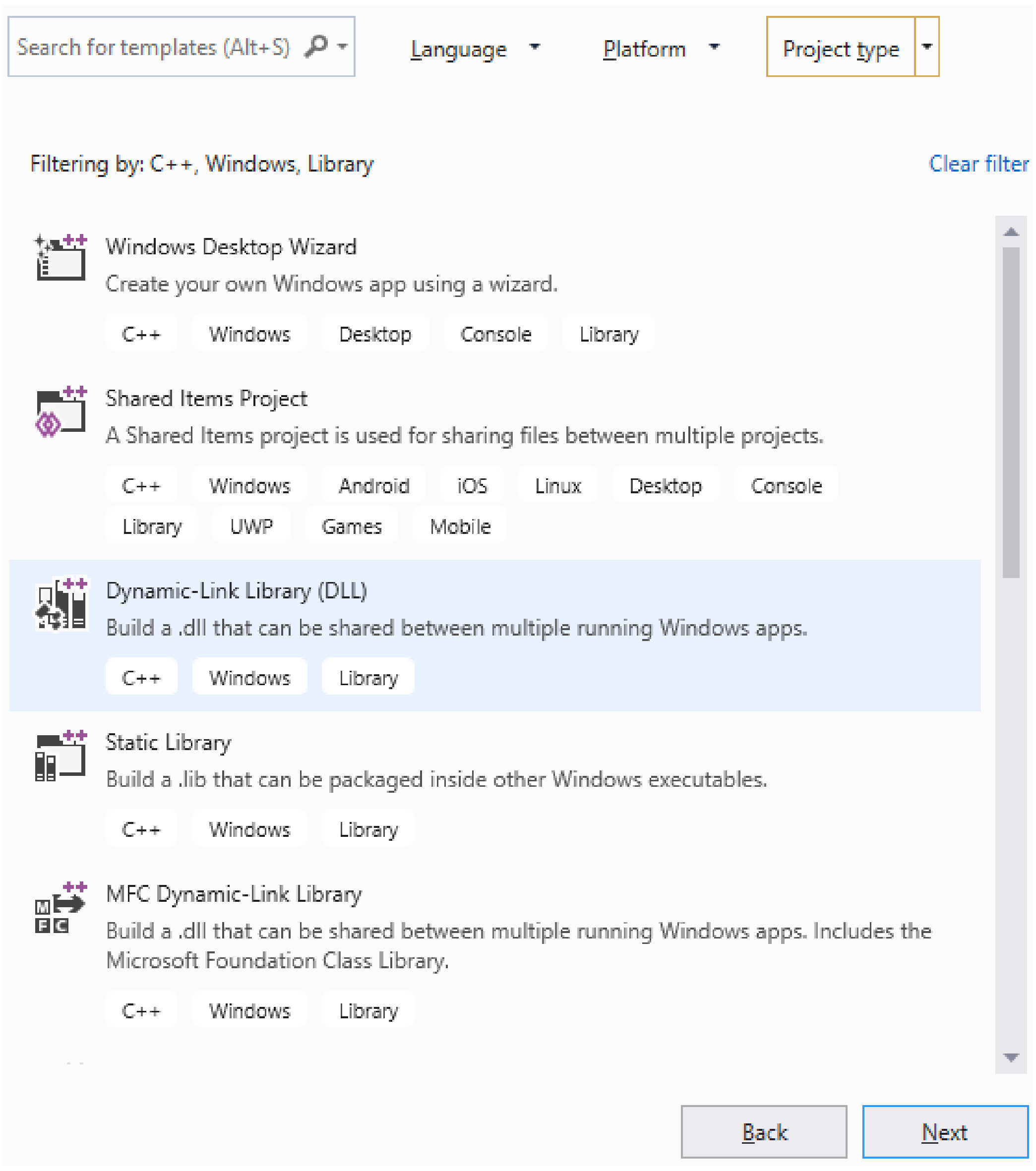
Package Structure

The C++ Project files have been copied to the cpp subfolder
multiplyBy C++ Project has been compiled to both 32 and 64 bit DLLs and the files copied to the dll subfolder
Each programming language script of file that access multiplyBy have been copied to subfolders named after their extensions

multiplyBy C++ Project

Create New Project

- Using Visual Studio 2019 you can create a new C++ Dynamic-Link Library project



Filtering by: C++, Windows, Library

Windows Desktop Wizard
Create your own Windows app using a wizard.
C++ Windows Desktop Console Library

Shared Items Project
A Shared Items project is used for sharing files between multiple projects.
C++ Windows Android iOS Linux Desktop Console Library UWP Games Mobile

Dynamic-Link Library (DLL)
Build a .dll that can be shared between multiple running Windows apps.
C++ Windows Library

Static Library
Build a .lib that can be packaged inside other Windows executables.
C++ Windows Library

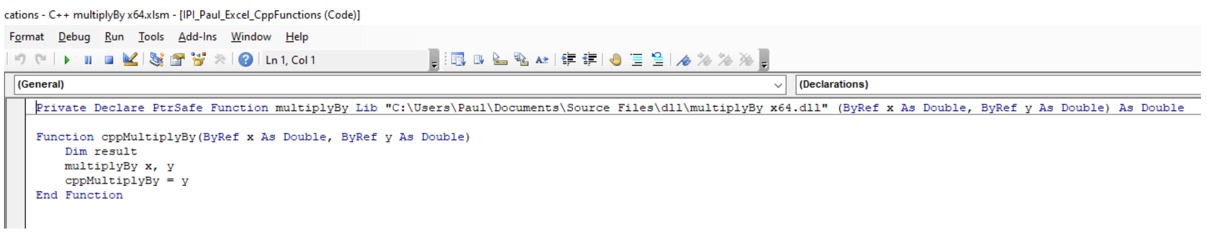
MFC Dynamic-Link Library
Build a .dll that can be shared between multiple running Windows apps. Includes the Microsoft Foundation Class Library.
C++ Windows Library

- Once created you can copy the multiplyBy.cpp file to the Source Files directory in your project. You will also need to replace the framework.h file with the one provided. These are the only two files altered by me when following the same process
- Depending on your system or if you like, you can compile the project for 32 bit, 64 bit or both
- I choose to rename the DLL files appending x86 to the 32 bit file and x64 for the 64 bit
- You can also just use the 2 DLL files I compiled from the subfolder dll

Excel Files

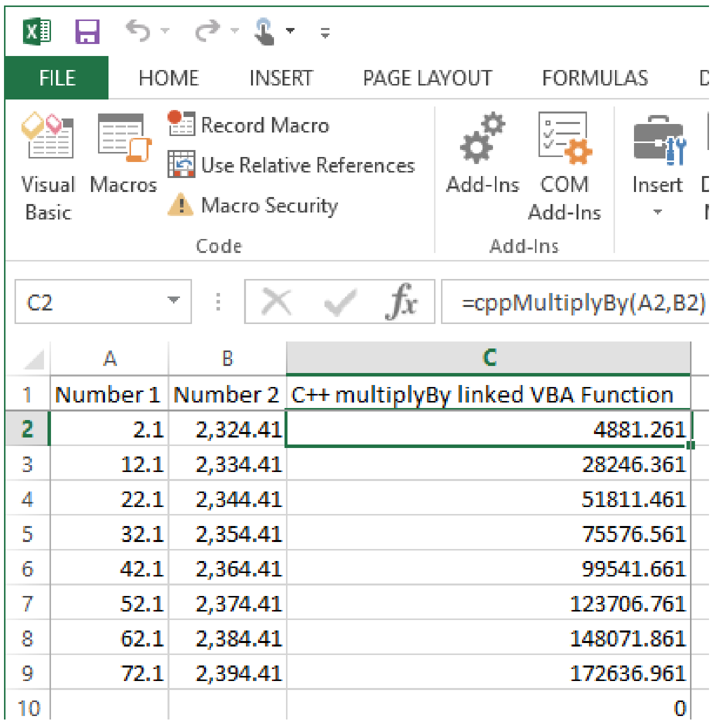
C++ multiplyBy x64.xlsm

- You will have to modify the VBA code declaration path to the location where you store copies of your DLL file


cations - C++ multiplyBy x64.xlsm - [IPI_Paul_Excel_CppFunctions (Code)]
Format Debug Run Tools Add-Ins Window Help
Ln 1, Col 1
(General)
(Declarations)

```
Private Declare PtrSafe Function multiplyBy Lib "C:\Users\Paul\Documents\Source Files\dll\multiplyBy x64.dll" (ByRef x As Double, ByRef y As Double) As Double

Function cppMultiplyBy(ByRef x As Double, ByRef y As Double)
    Dim result
    multiplyBy x, y
    cppMultiplyBy = y
End Function
```

- The cppMultiplyBy VBA function can either be used in other VBA code or worksheet formulae
- The **multiplyBy Example** worksheet tab uses the VBA function to as a formula in the end column to calculate the two preceding columns



=cppMultiplyBy(A2,B2)

	A	B	C
1	Number 1	Number 2	C++ multiplyBy linked VBA Function
2	2.1	2,324.41	4881.261
3	12.1	2,334.41	28246.361
4	22.1	2,344.41	51811.461
5	32.1	2,354.41	75576.561
6	42.1	2,364.41	99541.661
7	52.1	2,374.41	123706.761
8	62.1	2,384.41	148071.861
9	72.1	2,394.41	172636.961
10			0

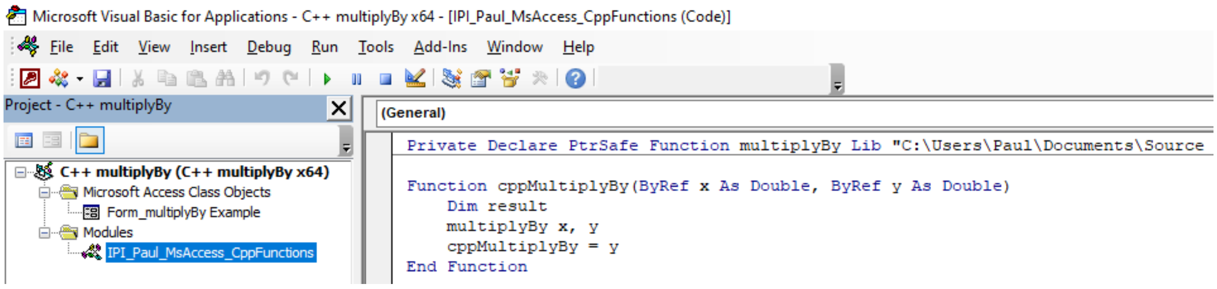
IPI_Paul_Excel_CppFunctions_x64.bas or x86.bas

- The above VBA module was exported as these file names and the x86 file was amended to load the x86 DLL file
- You can import these modules in to any VBA Project
- Again, you will need to amend the file location of the DLL

Ms Access Files

C++ multiplyBy x64.accdb

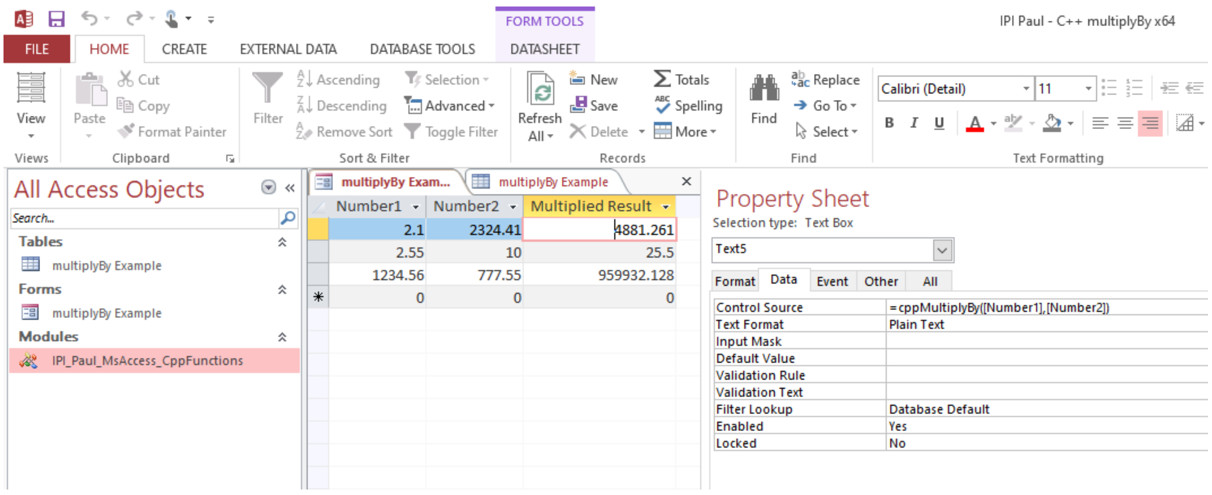
- Like Excel, you will have to modify the VBA code declaration path



Microsoft Visual Basic for Applications - C++ multiplyBy x64 - [IPI_Paul_MsAccess_CppFunctions (Code)]
File Edit View Insert Debug Run Tools Add-Ins Window Help
Project - C++ multiplyBy
C++ multiplyBy (C++ multiplyBy x64)
Microsoft Access Class Objects
Form_multiplyBy Example
Modules
IPI_Paul_MsAccess_CppFunctions
(General)

```
Private Declare PtrSafe Function multiplyBy Lib "C:\Users\Paul\Documents\Source

Function cppMultiplyBy(ByRef x As Double, ByRef y As Double)
    Dim result
    multiplyBy x, y
    cppMultiplyBy = y
End Function
```

- Like Excel too, you can use the cppMultiplyBy function in both VBA code and field formulae
- A simple Table and Form combination provides an example



FORM TOOLS
IPI Paul - C++ multiplyBy x64
FILE HOME CREATE EXTERNAL DATA DATABASE TOOLS DATASHEET
View Paste Cut Copy Format Painter Filter Ascending Descending Remove Sort Selection Advanced Toggle Filter Refresh All New Save Delete Totals Spelling More Find Replace Go To Select
Views Clipboard Sort & Filter Records Find Text Formatting
Calibri (Detail) 11
All Access Objects
Search...
Tables
multiplyBy Example
Forms
multiplyBy Example
Modules
IPI_Paul_MsAccess_CppFunctions
multiplyBy Exam...
multiplyBy Example

Number1	Number2	Multiplied Result
2.1	2324.41	4881.261
2.55	10	25.5
1234.56	777.55	959932.128
0	0	0

Property Sheet

Selection type: Text Box

Text5

Format | Data | Event | Other | All

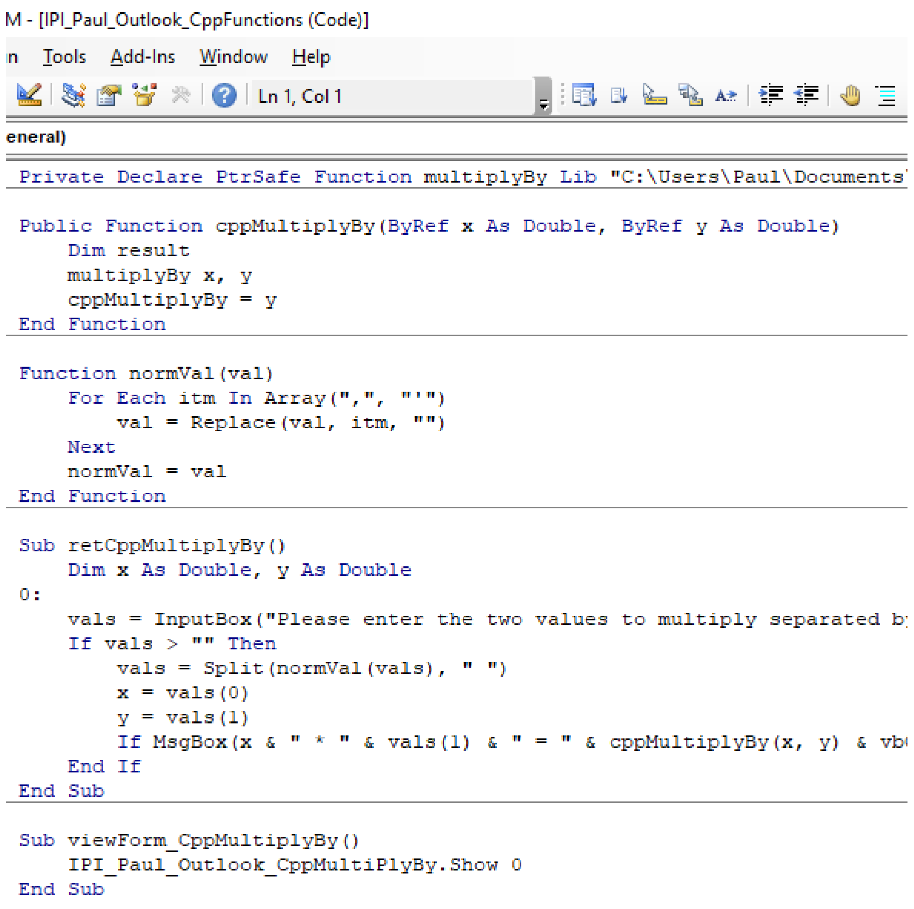
Control Source	=cppMultiplyBy([Number1],[Number2])
Text Format	Plain Text
Input Mask	
Default Value	
Validation Rule	
Validation Text	
Filter Lookup	Database Default
Enabled	Yes
Locked	No

IPI_Paul_MsAccess_CppFunctions_x64.bas or x86.bas

- The above VBA module was exported as these file names and the x86 file was amended to load the x86 DLL file
- You can import these modules in to any VBA Project
- Again, you will need to amend the file location of the DLL

retCPPMultiplyBy

- This was the initial test, offering an inputbox where the user enters the 2 numbers separated by a space. A message box then displays the answer and prompts the user to choose between entering another combination or quitting

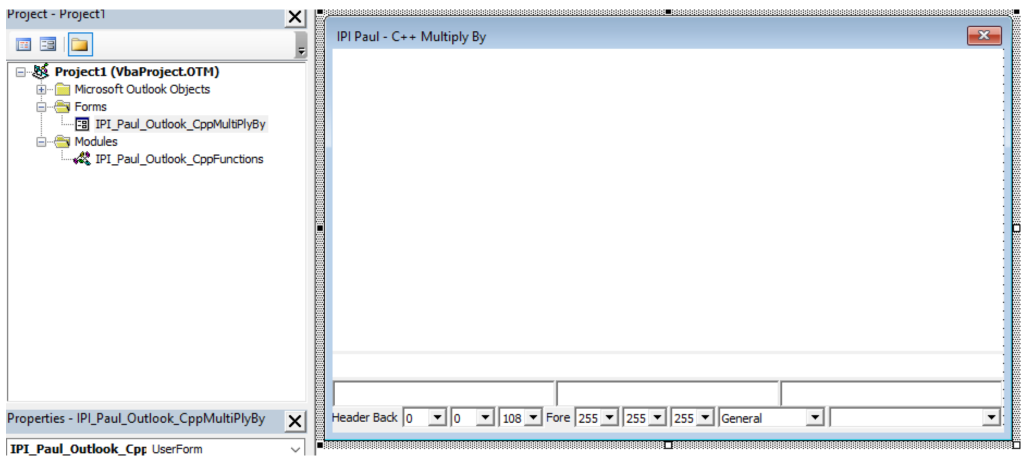


M - [IPI_Paul_Outlook_CppFunctions (Code)]
n Tools Add-Ins Window Help
Ln 1, Col 1
eneral)

```
Private Declare PtrSafe Function multiplyBy Lib "C:\Users\Paul\Documents'

Public Function cppMultiplyBy(ByRef x As Double, ByRef y As Double)
    Dim result
    multiplyBy x, y
    cppMultiplyBy = y
End Function

Function normVal(val)
    For Each itm In Array(",", "'")
        val = Replace(val, itm, "")
    Next
    normVal = val
End Function

Sub retCppMultiplyBy()
    Dim x As Double, y As Double
0:
    vals = InputBox("Please enter the two values to multiply separated b
    If vals > "" Then
        vals = Split(normVal(vals), " ")
        x = vals(0)
        y = vals(1)
        If MsgBox(x & " * " & vals(1) & " = " & cppMultiplyBy(x, y) & vb
    End If
End Sub

Sub viewForm_CppMultiplyBy()
    IPI_Paul_Outlook_CppMultiPlyBy.Show 0
End Sub
```

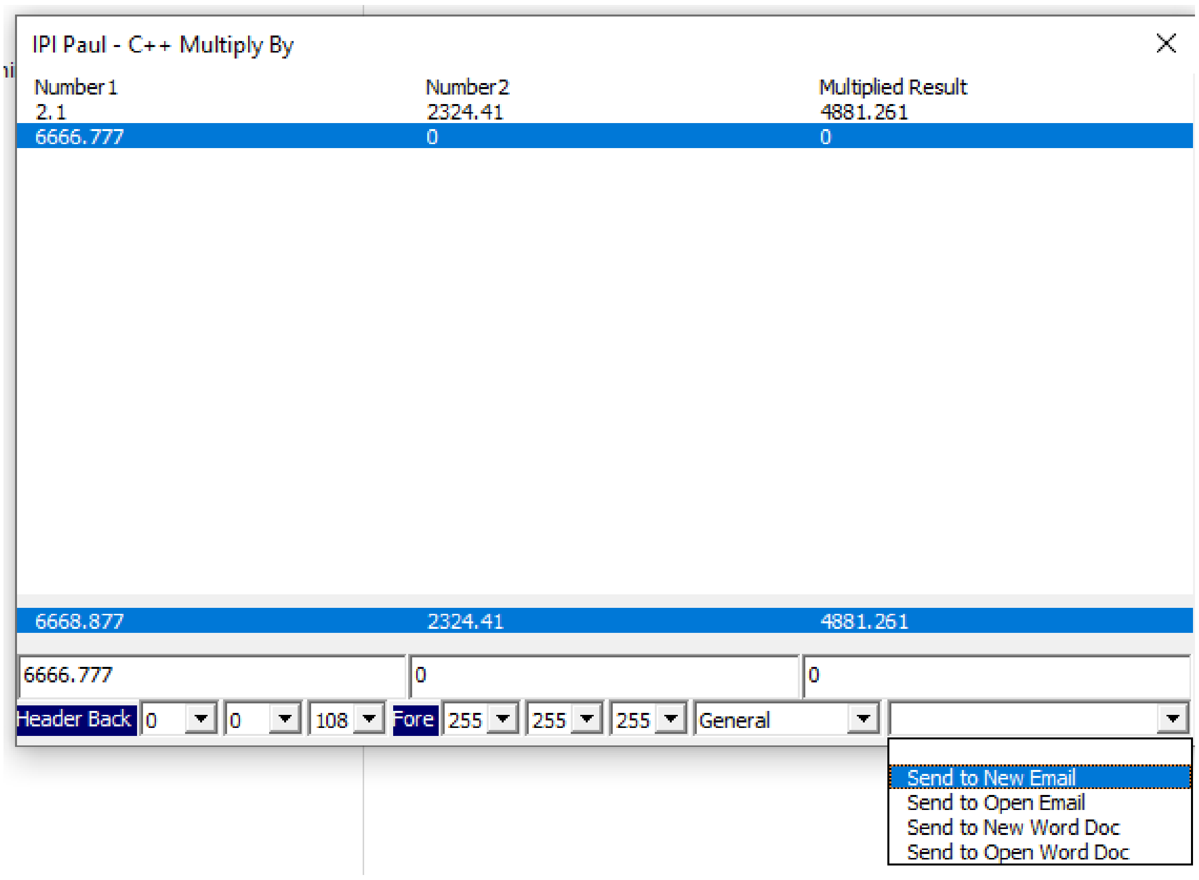
IPI_Paul_Outlook_CppMultiPlyBy

- This user form comprises of two list boxes, the main one displaying the row data and the bottom one showing the totals



Project - Project1
Project1 (VbaProject.OTM)
Microsoft Outlook Objects
Forms
IPI_Paul_Outlook_CppMultiPlyBy
Modules
IPI_Paul_Outlook_CppFunctions
Properties - IPI_Paul_Outlook_CppMultiPlyBy
IPI_Paul_Outlook_Cpp UserForm
IPI Paul - C++ Multiply By
Header Back 0
0
108
Fore 255
255
255
General

- Clicking on the main list box will set each of the three text boxes below the totals list box to the values of that selected row. Changing the entries in the text boxes will update the main list box entries
- Pressing Ctrl+C will put the Main and Totals list in to the clipboard as Tab Delimited Values which can then be pasted in to Excel or a TSV Text file
- Double Clicking the first text box will add a new row in the main list box


IPI Paul - C++ Multiply By

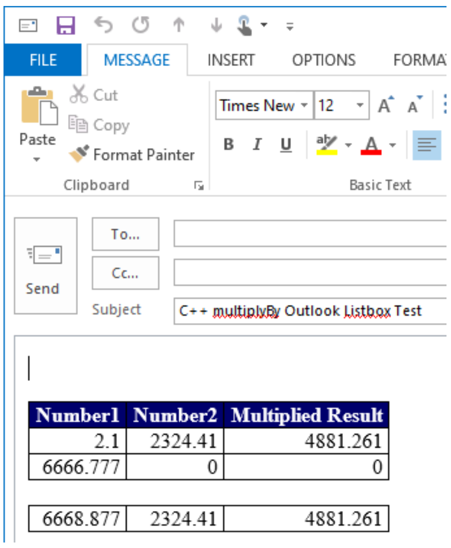
Number1	Number2	Multiplied Result
2.1	2324.41	4881.261
6666.777	0	0

6668.877	2324.41	4881.261

6666.777	0	0

Header Back 0 0 108 Fore 255 255 255 General

- There are 3 combos each for the Header background and fore colours, representing Red; Green; and Blue. Altering these will update the label colours and any exported list header
- A Drop down menu offers the chance to export the list to an Email or a Word Document. When Open … is selected, the table is inserted at the current cursor position of an open message/document

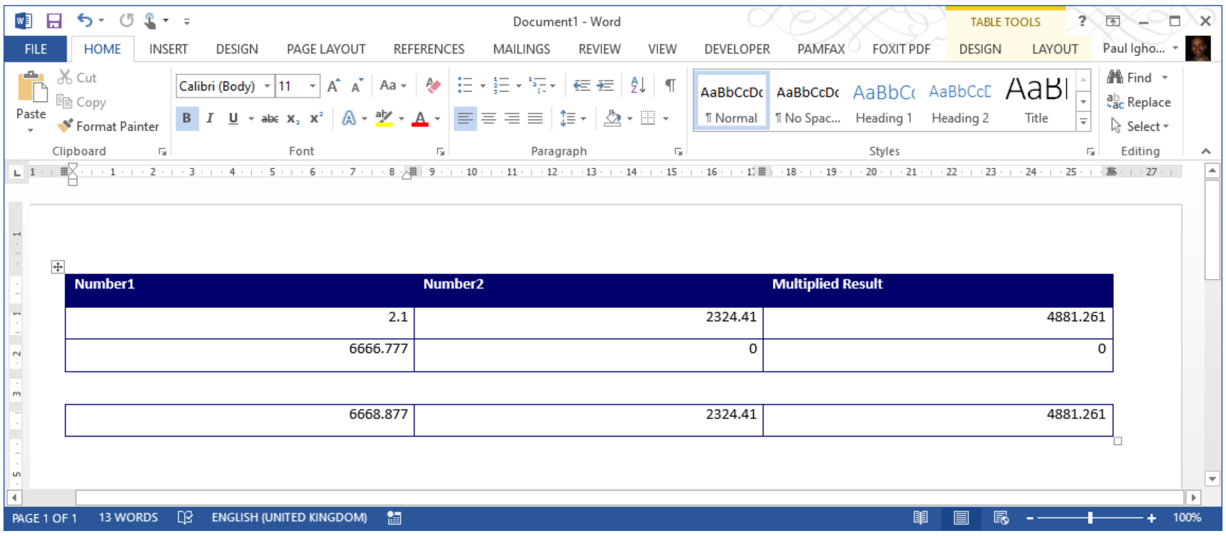


Subject: C++ multiplyBy Outlook Listbox Test

Number1	Number2	Multiplied Result
2.1	2324.41	4881.261
6666.777	0	0

6668.877	2324.41	4881.261

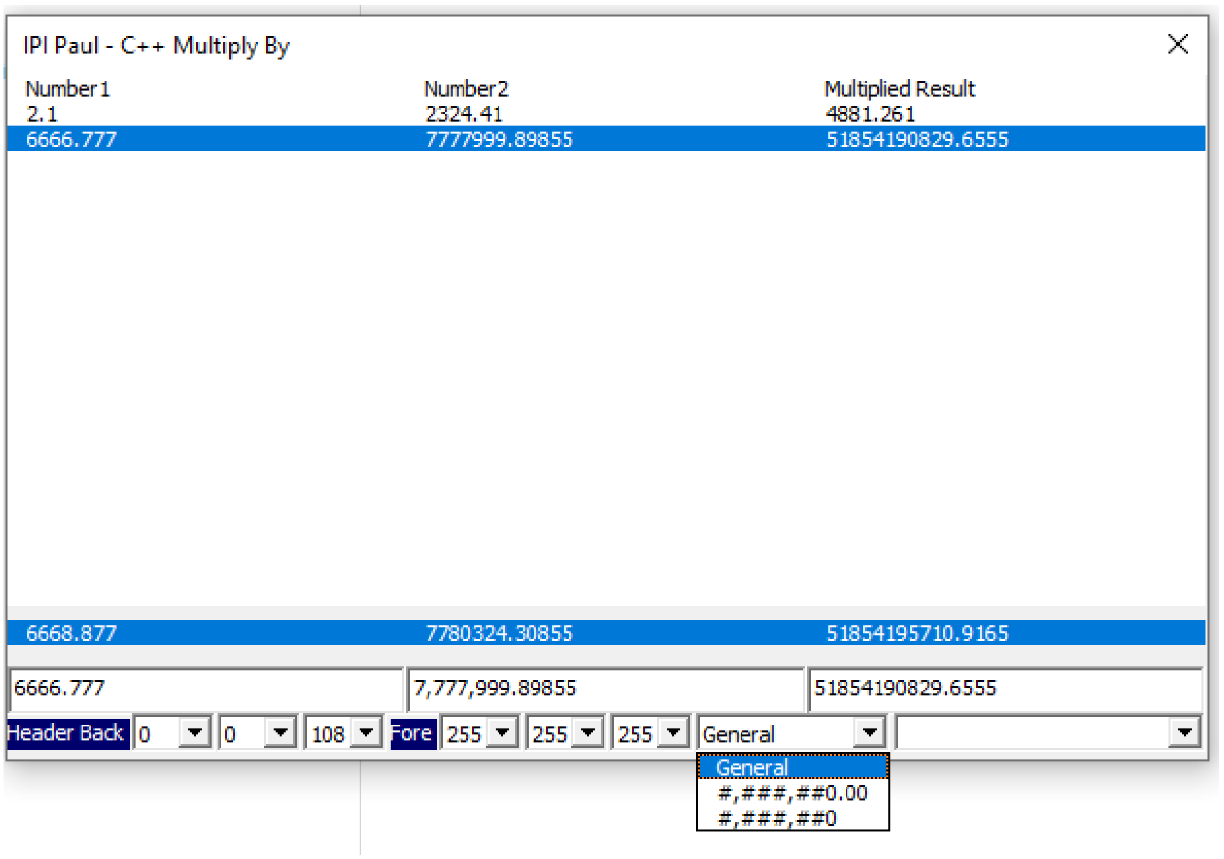
- Because html is used to populate the email, the table layout automatically sizes to the content widths
- Sending to a New Word document will set up the Page to be landscape, with 1.75 cm margins
- Further coding could be done to the Word export function to size the resulting table, but, as this is meant to be a simple example the generated table sizing is left as is



Number1	Number2	Multiplied Result
2.1	2324.41	4881.261
6666.777	0	0

6668.877	2324.41	4881.261

- Changing the Format combo will set the email number formats

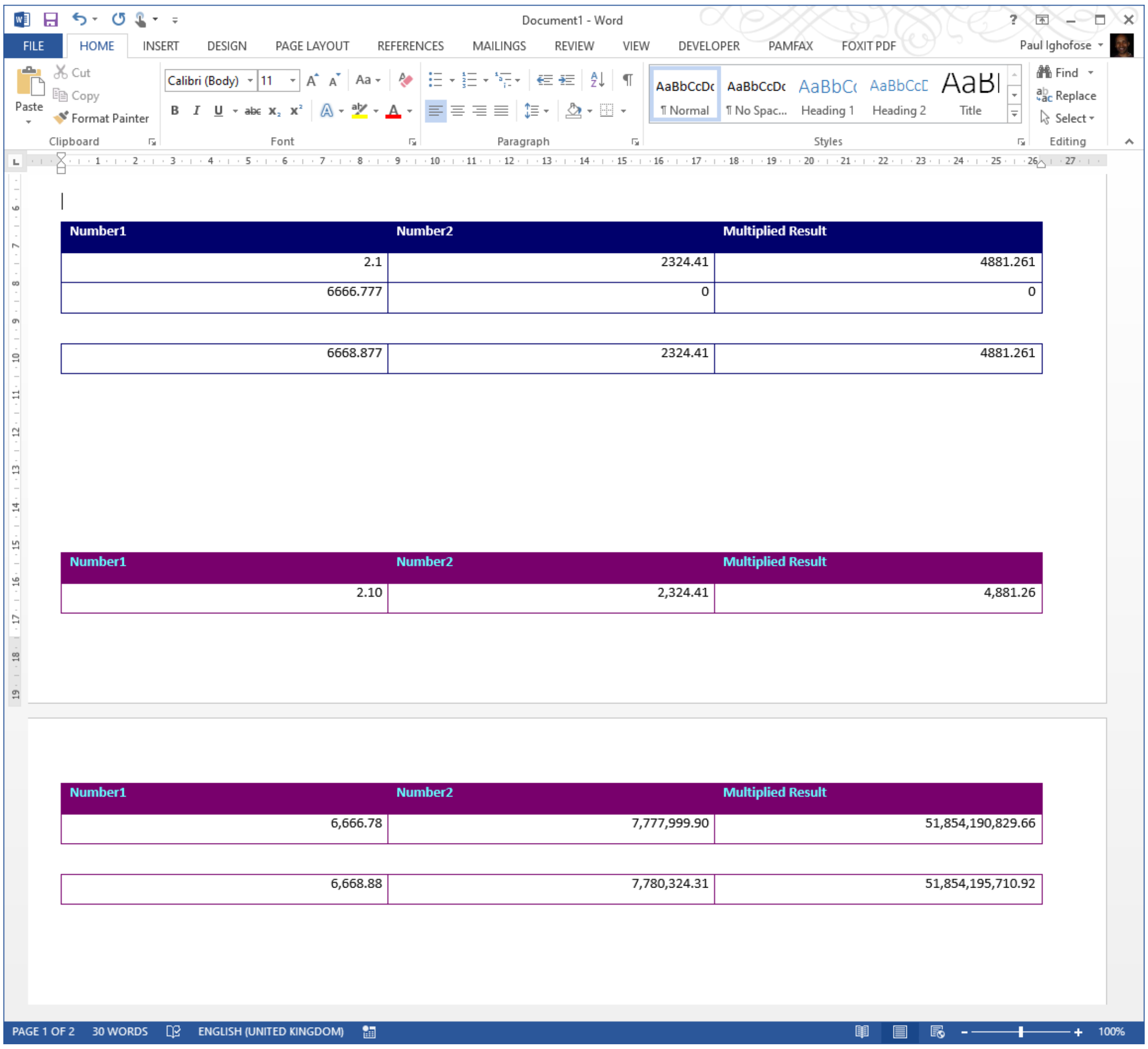

IPI Paul - C++ Multiply By

Number1	Number2	Multiplied Result
2.1	2324.41	4881.261
6666.777	7777999.89855	51854190829.6555
6668.877	7780324.30855	51854195710.9165

6666.777	7,777,999.89855	51854190829.6555

Header Back 0 0 108 Fore 255 255 255 General

- When Exporting to Word, a new row is added if the table overlaps pages and each new page is given a header



Number1	Number2	Multiplied Result
2.1	2324.41	4881.261
6666.777	0	0

6668.877	2324.41	4881.261

Number1	Number2	Multiplied Result
2.10	2,324.41	4,881.26

Number1	Number2	Multiplied Result
6,666.78	7,777,999.90	51,854,190,829.66

6,668.88	7,780,324.31	51,854,195,710.92

IPI_Paul_Outlook_CppFunctions_x64.bas or 86.bas

- Like Excel and Access, these are the Exported Modules containing the Declare statement to load the C++ DLL
- These can be imported in to your Outlook Application or any of Excel, Word or Ms Access VBA Projects

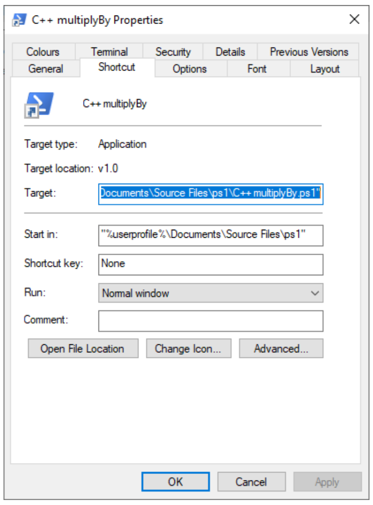
IPI_Paul_Outlook_CppMultiPlyBy.frm and .frx

- These files are the Exported user form
- And again, can be imported in to your Outlook Application or any of Excel, Word or Ms Access VBA Projects

C++ multiplyBy.ps1

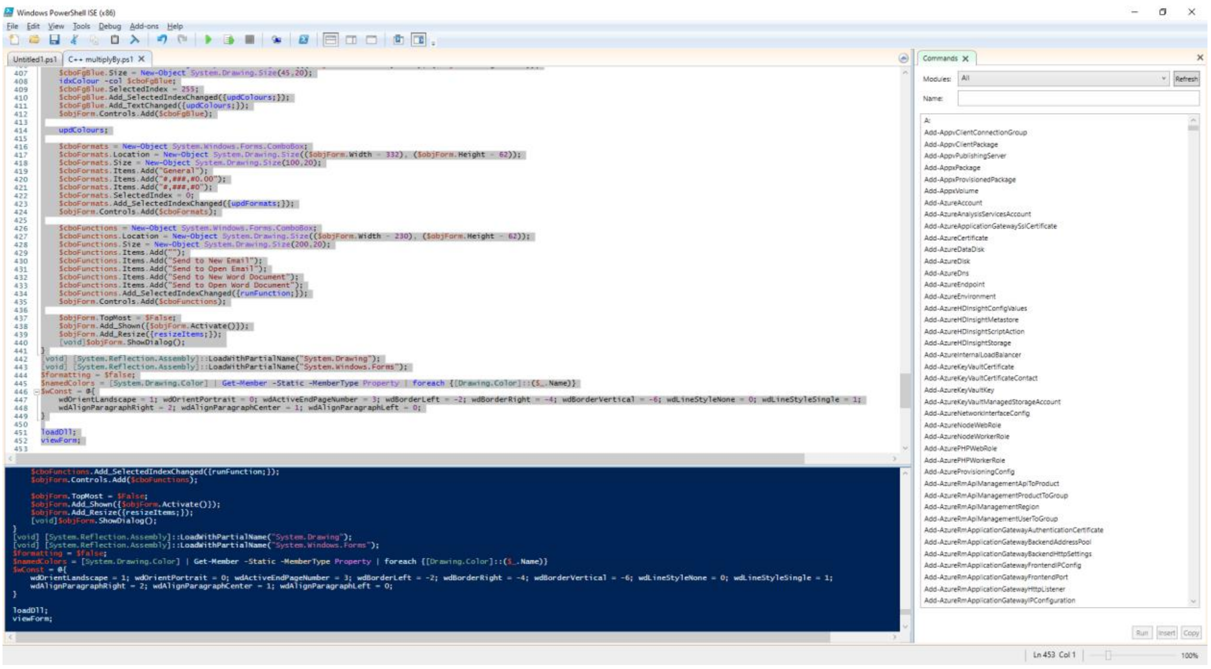
- There are two ways to run the Script
- You can either create a Windows Shortcut like

  C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -WindowStyle Hidden -ExecutionPolicy Unrestricted -File "%userprofile%\Documents\Source Files\ps1\C++ multiplyBy.ps1"



Target type: Application
Target location: v1.0
Target: Documents\Source Files\ps1\C++ multiplyBy.ps1"
Start in: "%userprofile%\Documents\Source Files\ps1"
Shortcut key: None
Run: Normal window
Comment:

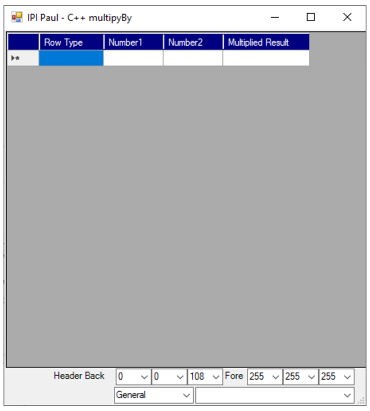
- Do not use Advanced, Run As Administrator as Outlook will need to be opened using Elevated Privileges for the two to communicate
- Or you can open the script in PowerShell and copy and paste it in to the console, then run it by pressing enter



```
    $cboFgBlue.Size = New-Object System.Drawing.Size(45,20);
    idxColour -col $cboFgBlue;
    $cboFgBlue.SelectedIndex = 255;
    $cboFgBlue.Add_SelectedIndexChanged({updColours;});
    $cboFgBlue.Add_TextChanged({updColours;});
    $objForm.Controls.Add($cboFgBlue);

    updColours;

    $cboFormats = New-Object System.Windows.Forms.ComboBox;
    $cboFormats.Location = New-Object System.Drawing.Size(($objForm.Width - 332), ($objForm.Height - 62));
    $cboFormats.Size = New-Object System.Drawing.Size(100,20);
    $cboFormats.Items.Add("General");
    $cboFormats.Items.Add("#,###,#0.00");
    $cboFormats.Items.Add("#,###,#0");
    $cboFormats.SelectedIndex = 0;
    $cboFormats.Add_SelectedIndexChanged({updFormats;});
    $objForm.Controls.Add($cboFormats);

    $cboFunctions = New-Object System.Windows.Forms.ComboBox;
    $cboFunctions.Location = New-Object System.Drawing.Size(($objForm.Width - 230), ($objForm.Height - 62));
    $cboFunctions.Size = New-Object System.Drawing.Size(200,20);
    $cboFunctions.Items.Add("");
    $cboFunctions.Items.Add("Send to New Email");
    $cboFunctions.Items.Add("Send to Open Email");
    $cboFunctions.Items.Add("Send to New Word Document");
    $cboFunctions.Items.Add("Send to Open Word Document");
    $cboFunctions.Add_SelectedIndexChanged({runFunction;});
    $objForm.Controls.Add($cboFunctions);

    $objForm.TopMost = $False;
    $objForm.Add_Shown({$objForm.Activate()});
    $objForm.Add_Resize({resizeItems;});
    [void]$objForm.ShowDialog();
}
[void] [System.Reflection.Assembly]::LoadWithPartialName("System.Drawing");
[void] [System.Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms");
$formatting = $false;
$namedColors = [System.Drawing.Color] | Get-Member -Static -MemberType Property | foreach {[Drawing.Color]::($_.Name)}
$wConst = @{
    wdOrientLandscape = 1; wdOrientPortrait = 0; wdActiveEndPageNumber = 3; wdBorderLeft = -2; wdBorderRight = -4; wdBorderVertical = -6; wdLineStyleNone = 0; wdLineStyleSingle = 1;
    wdAlignParagraphRight = 2; wdAlignParagraphCenter = 1; wdAlignParagraphLeft = 0;
}

loadDll;
viewForm;
```

```
    $cboFunctions.Add_SelectedIndexChanged({runFunction;});
    $objForm.Controls.Add($cboFunctions);

    $objForm.TopMost = $False;
    $objForm.Add_Shown({$objForm.Activate()});
    $objForm.Add_Resize({resizeItems;});
    [void]$objForm.ShowDialog();
}
[void] [System.Reflection.Assembly]::LoadWithPartialName("System.Drawing");
[void] [System.Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms");
$formatting = $false;
$namedColors = [System.Drawing.Color] | Get-Member -Static -MemberType Property | foreach {[Drawing.Color]::($_.Name)}
$wConst = @{
    wdOrientLandscape = 1; wdOrientPortrait = 0; wdActiveEndPageNumber = 3; wdBorderLeft = -2; wdBorderRight = -4; wdBorderVertical = -6; wdLineStyleNone = 0; wdLineStyleSingle = 1;
    wdAlignParagraphRight = 2; wdAlignParagraphCenter = 1; wdAlignParagraphLeft = 0;
}

loadDll;
viewForm;
```

- Once loaded, start by inserting a value in the Number1 cell of the new record row

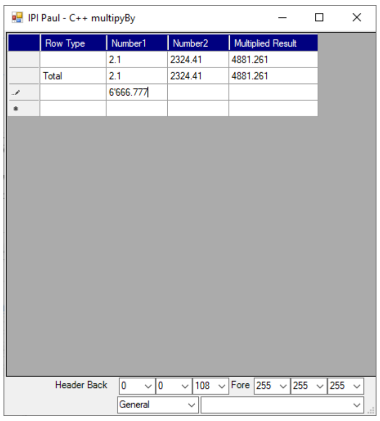

IPI Paul - C++ multipyBy

Row Type	Number1	Number2	Multiplied Result

Header Back 0 0 108 Fore 255 255 255

General

- The first column is disabled and is used by the script to indicate to itself which is the Total row

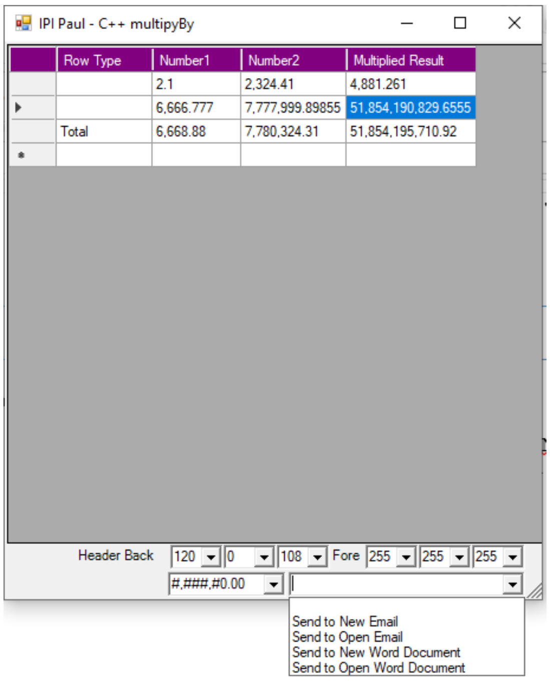

IPI Paul - C++ multipyBy

Row Type	Number1	Number2	Multiplied Result
	2.1	2324.41	4881.261
Total	2.1	2324.41	4881.261
	6'666.777		

Header Back 0 0 108 Fore 255 255 255

General

- When adding new rows or after updating cell values, the Totals row is deleted and then re-inserted at the bottom of the DataGridView with the calculated column totals
- Although sorting may change the row levels, amending the values in a cell will cause the Totals row to be removed and re-inserted at the bottom
- Changing the values in the RGB comobo boxes will update the DataGridView Header colour
- The DataGridView will automatically resize with resizing of the Forms window


IPI Paul - C++ multipyBy

Row Type	Number1	Number2	Multiplied Result
	2.1	2,324.41	4,881.261
	6,666.777	7,777,999.89855	51,854,190,829.6555
Total	6,668.88	7,780,324.31	51,854,195,710.92

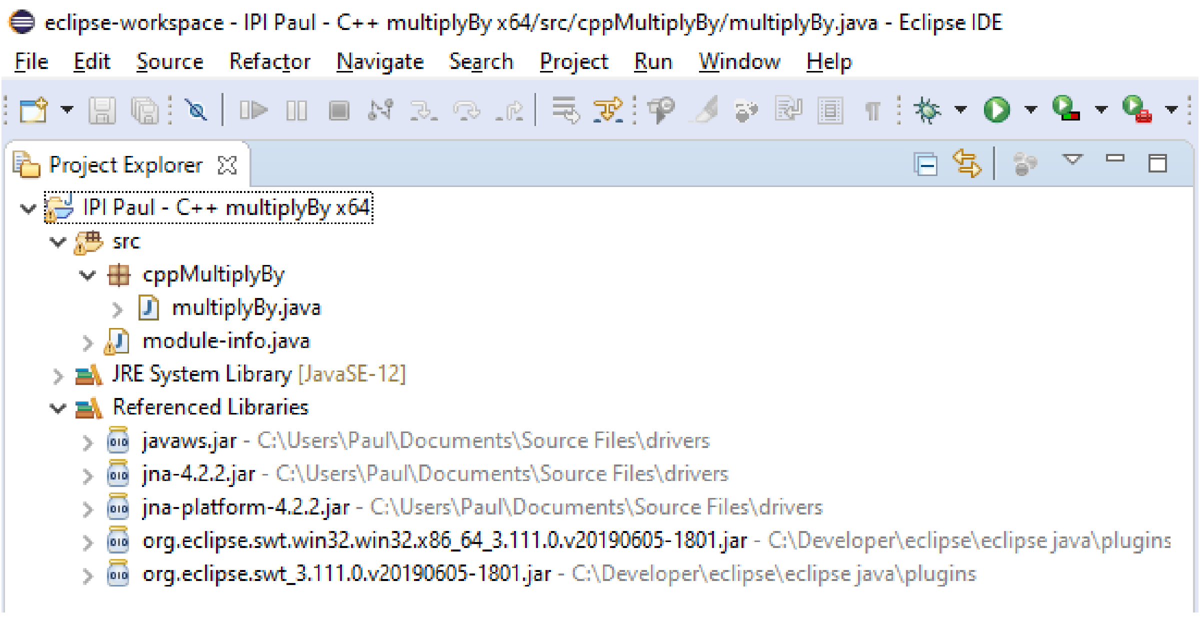
Header Back 120 0 108 Fore 255 255 255

#,###,#0.00

Java

multiplyBy.java

- I used eclipse IDE when developing this GUI and also tested it in Netbeans
- You will need to download the following dependency jar files and store them in a folder of you own choosing

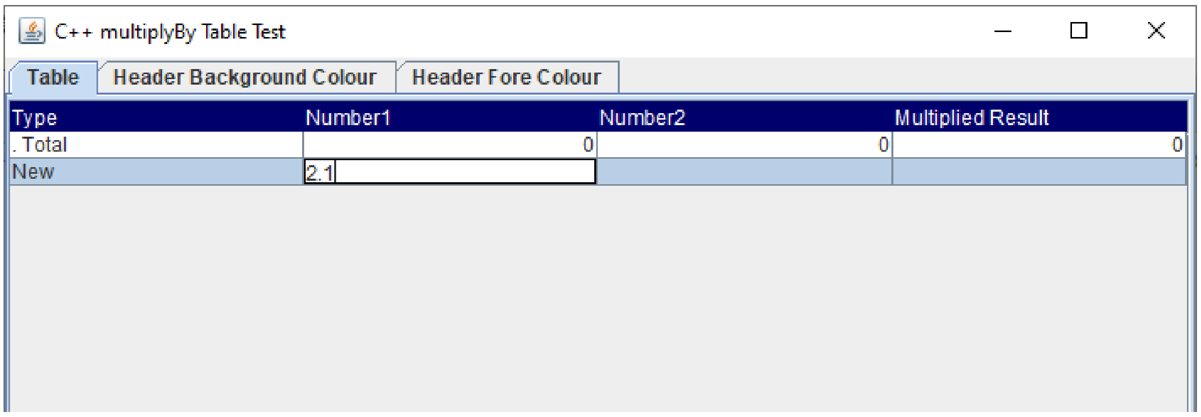


eclipse-workspace - IPI Paul - C++ multiplyBy x64/src/cppMultiplyBy/multiplyBy.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer

```
IPI Paul - C++ multiplyBy x64
  src
    cppMultiplyBy
      multiplyBy.java
    module-info.java
  JRE System Library [JavaSE-12]
  Referenced Libraries
    javaws.jar - C:\Users\Paul\Documents\Source Files\drivers
    jna-4.2.2.jar - C:\Users\Paul\Documents\Source Files\drivers
    jna-platform-4.2.2.jar - C:\Users\Paul\Documents\Source Files\drivers
    org.eclipse.swt.win32.win32.x86_64_3.111.0.v20190605-1801.jar - C:\Developer\eclipse\eclipse java\plugins
    org.eclipse.swt_3.111.0.v20190605-1801.jar - C:\Developer\eclipse\eclipse java\plugins
```

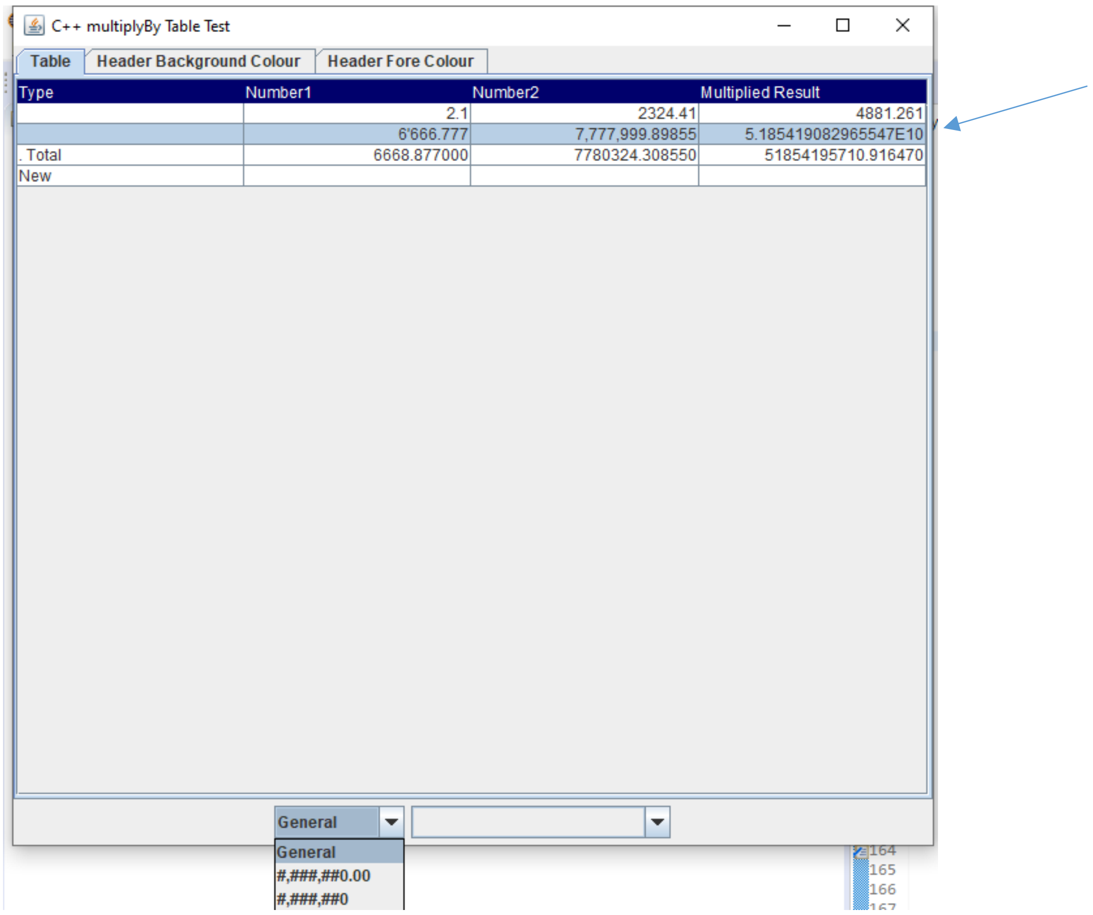
- You can then create a new project and copy the multiplyBy.java file I have provided in to the src folder
- The OLE driver was a bit challenging to understand
- DefaultTableCellRenderer occasionally fails to capture update events. Meaning that the C++ multiplyBy function is not called and the result not calculated. This is remedied by moving back and forth from one row to another and pressing the Return or Tab keys for the row requiring calculation
- Again, like the PowerShell Form, the first and last columns are not editable. The word New is used to indicate where rows can be added. Editing the second cell of the row indicated as new will move that row directly above the Totals row (or when New and Totals is sorted to the top, move it to the las row)



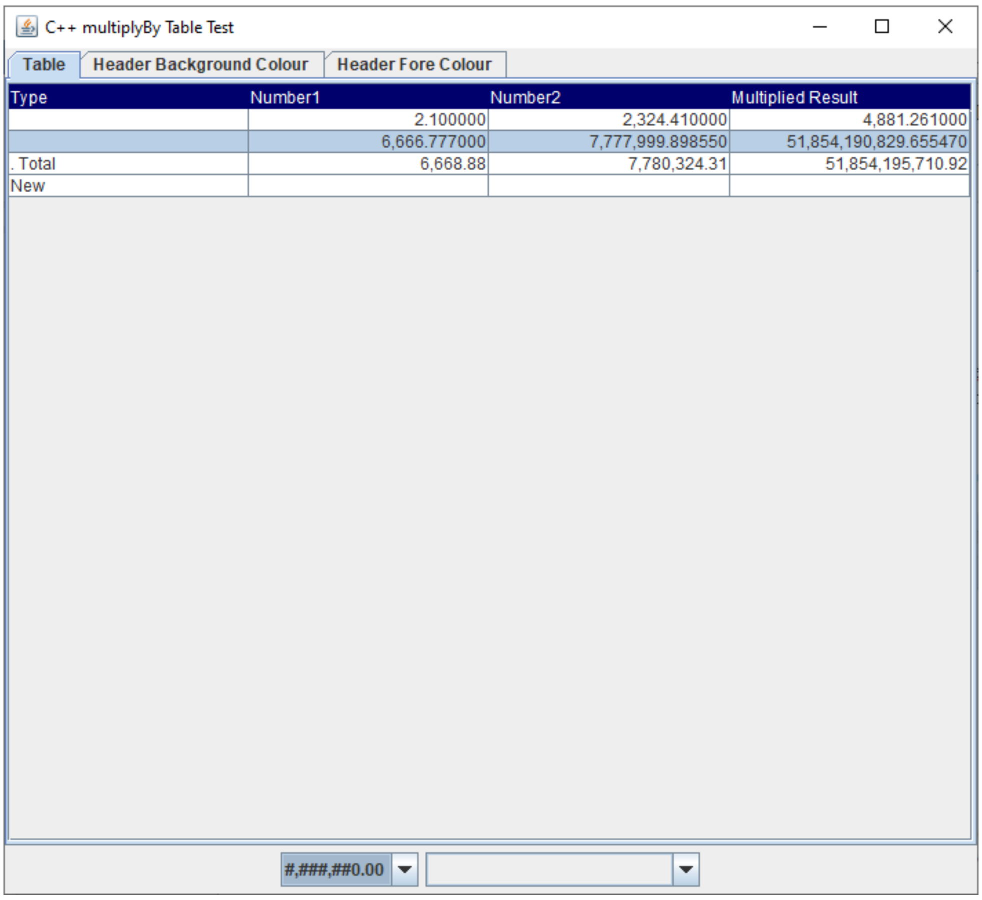
C++ multiplyBy Table Test

Type	Number1	Number2	Multiplied Result
. Total	0	0	0
New	2.1		

- Like the Outlook List box and the PowerShell Form, changing the Formats combo selection will reformat the display of the Totals values which stops any values being displayed as an Exponential

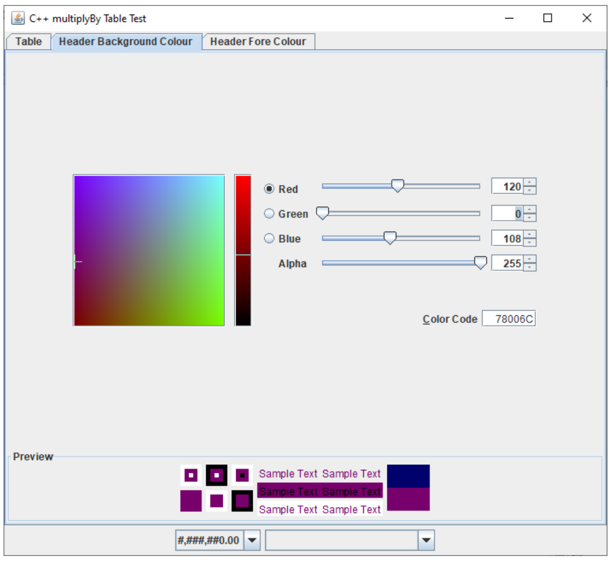


Type	Number1	Number2	Multiplied Result
	2.1	2324.41	4881.261
	6'666.777	7,777,999.89855	5.185419082965547E10
. Total	6668.877000	7780324.308550	51854195710.916470
New			

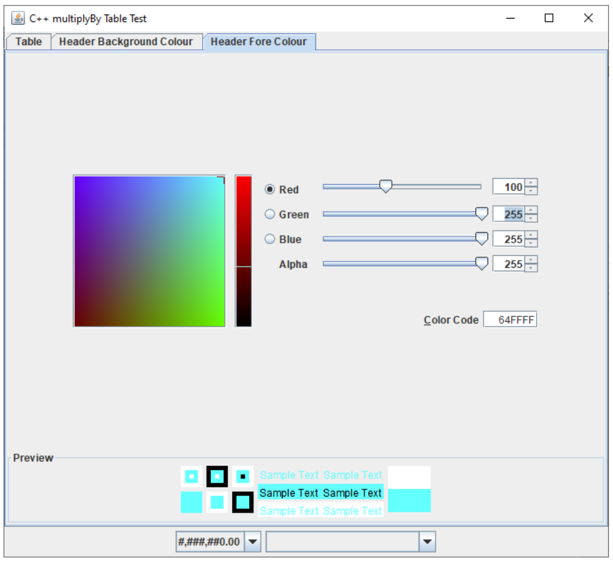


Type	Number1	Number2	Multiplied Result
	2.100000	2,324.410000	4,881.261000
	6,666.777000	7,777,999.898550	51,854,190,829.655470
. Total	6,668.88	7,780,324.31	51,854,195,710.92
New			

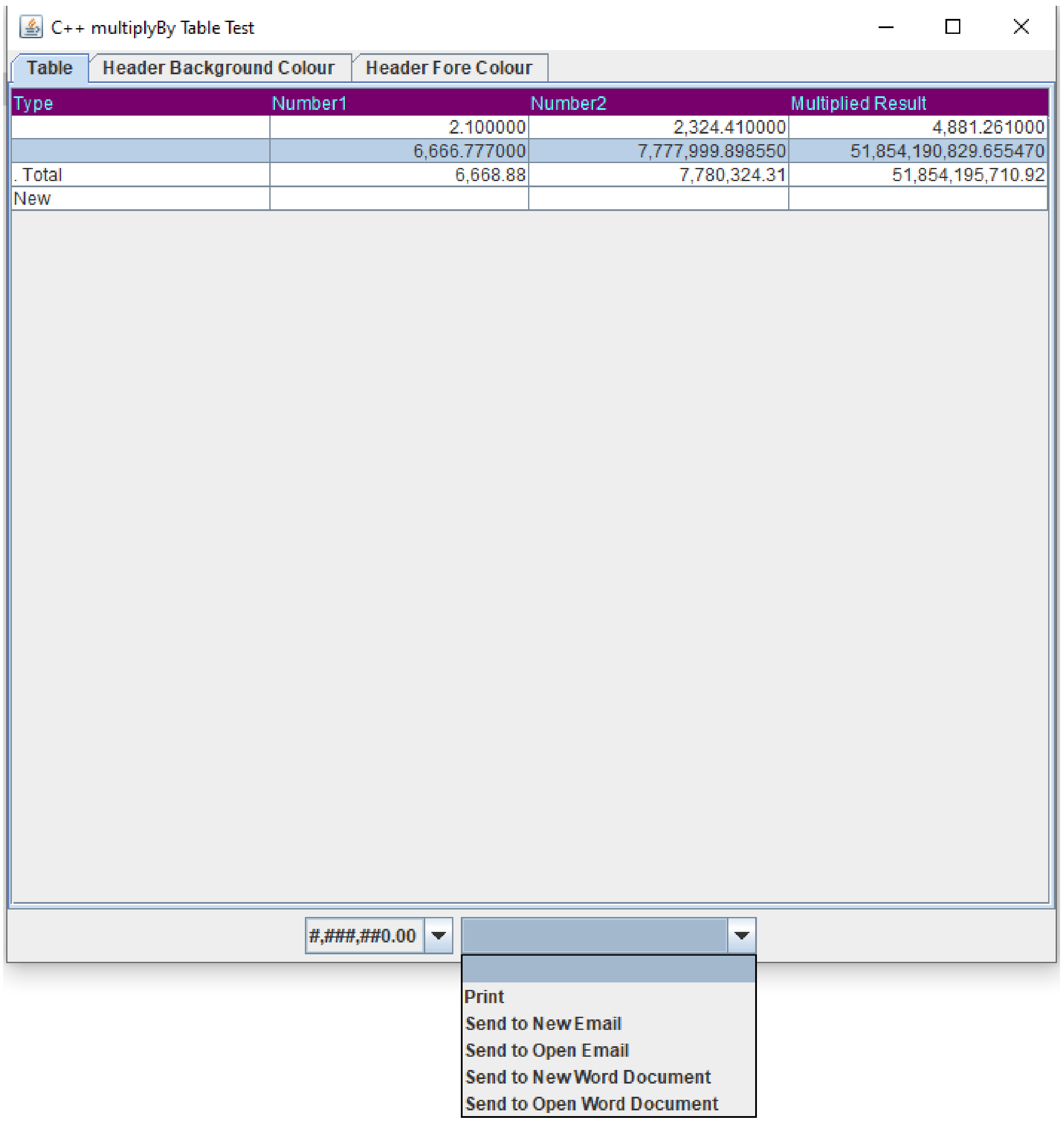
- I have also made good use of Java’s JColorChooser and amending either of these will effect both the Table Tab’s Headers and any exported table



Red 120
Green 0
Blue 108
Alpha 255
Color Code 78006C



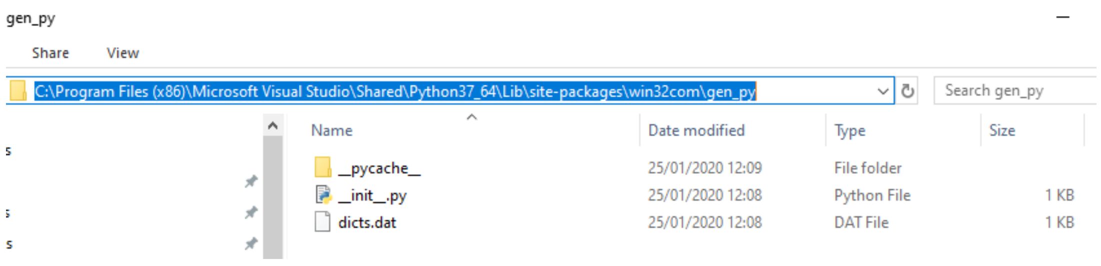
Red 100
Green 255
Blue 255
Alpha 255
Color Code 64FFFF



Type	Number1	Number2	Multiplied Result
	2.100000	2,324.410000	4,881.261000
	6,666.777000	7,777,999.898550	51,854,190,829.655470
. Total	6,668.88	7,780,324.31	51,854,195,710.92
New			

C++ multiplyBy.py

- You will need to download the correct binaries from Mark Hammond's github web folders to get the win32 COM drivers for exporting to Outlook and Word
- If like me, you have installed Visual Studio 2019 and Python 3.7.3 with it, then you will need to give everyone permissions to the 'C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\Lib\site-packages\win32com\gen_py' folder. Without that the module will fail to run as it cannot populate it when run in Python

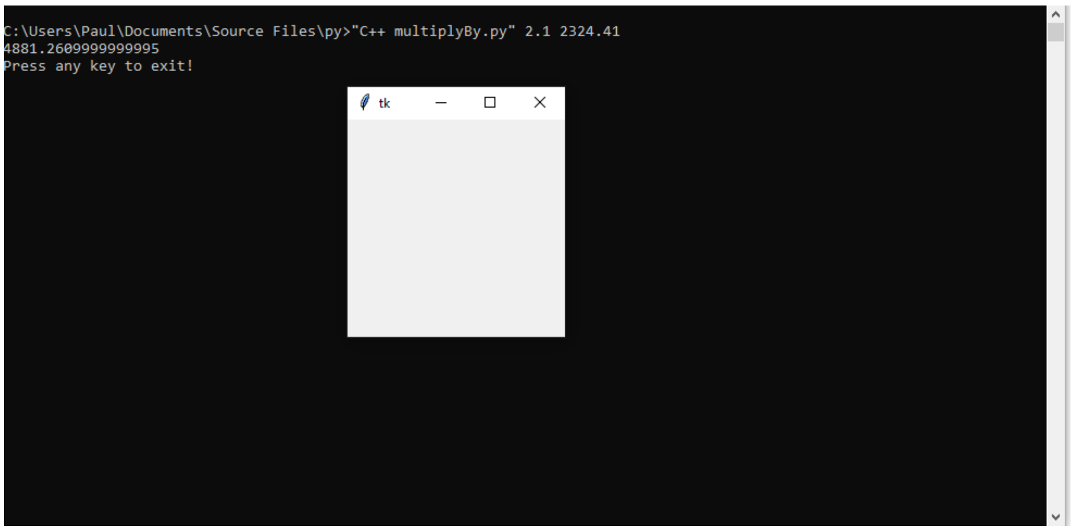


gen_py

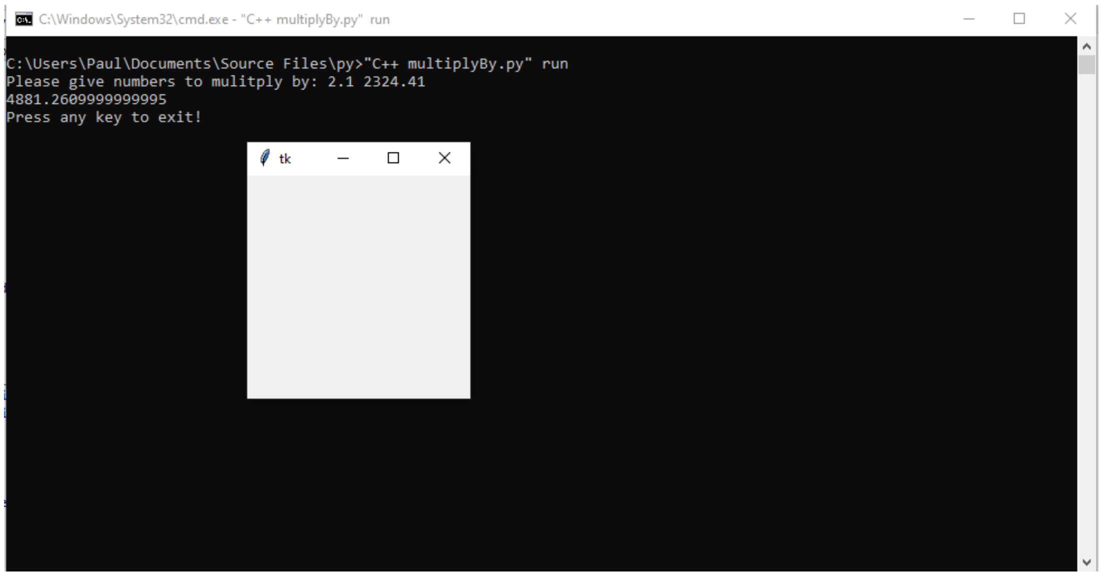
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\Lib\site-packages\win32com\gen_py

Name	Date modified	Type	Size
__pycache__	25/01/2020 12:09	File folder	
__init__.py	25/01/2020 12:08	Python File	1 KB
dicts.dat	25/01/2020 12:08	DAT File	1 KB

- The original DLL load tests are still available for when using console entries with an arg



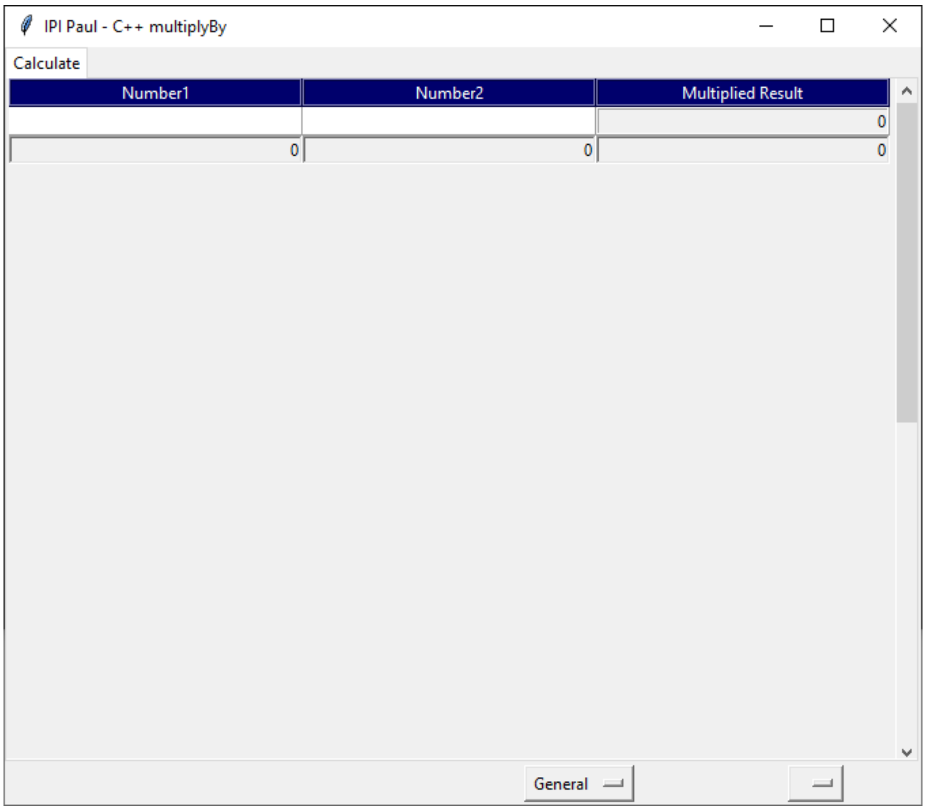
```
C:\Users\Paul\Documents\Source Files\py>"C++ multiplyBy.py" 2.1 2324.41
4881.2609999999995
Press any key to exit!
```



C:\Windows\System32\cmd.exe - "C++ multiplyBy.py" run

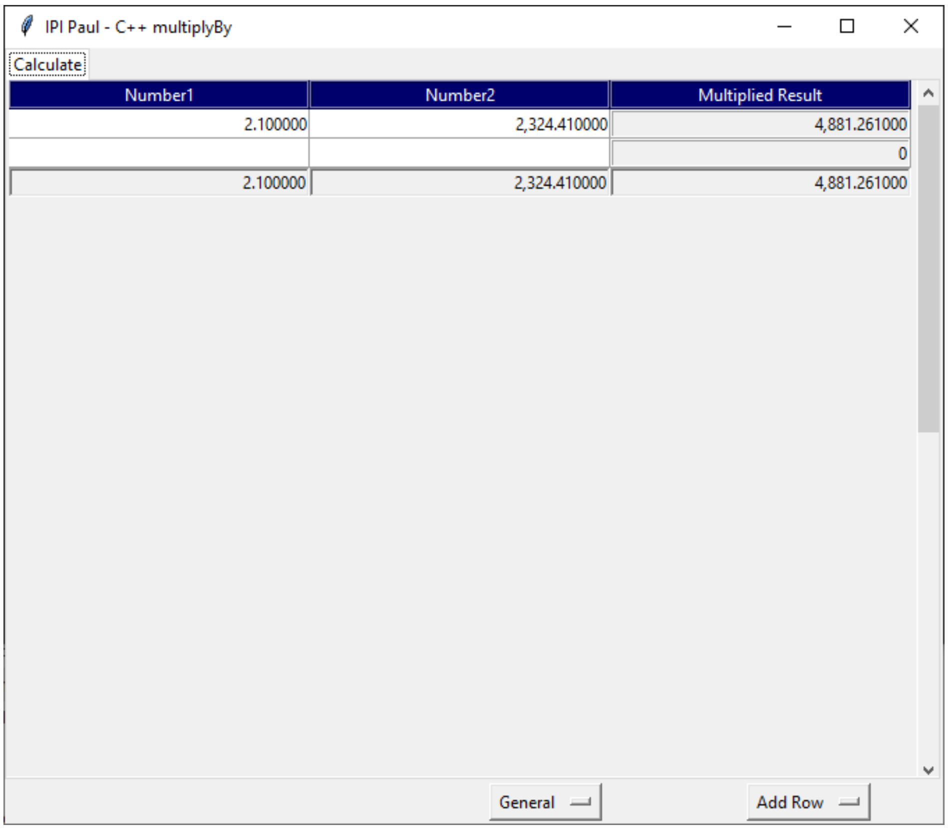
```
C:\Users\Paul\Documents\Source Files\py>"C++ multiplyBy.py" run
Please give numbers to mulitply by: 2.1 2324.41
4881.2609999999995
Press any key to exit!
```

- Double clicking the file or entering the file path without args will display the main window



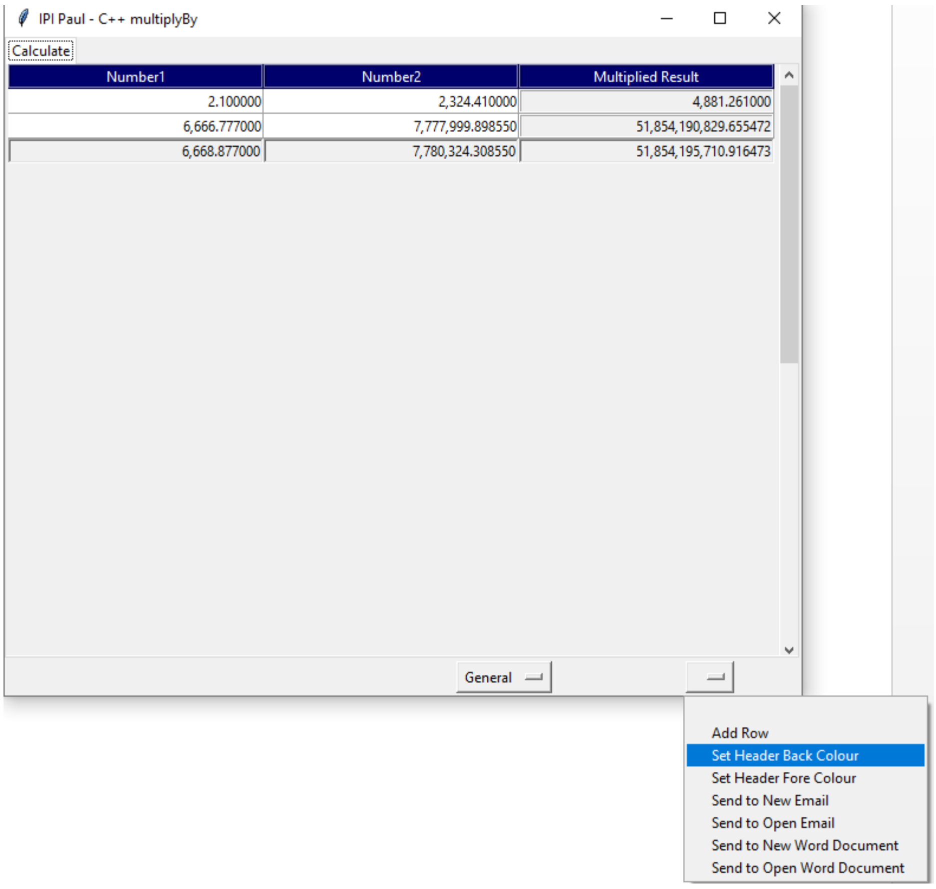
Number1	Number2	Multiplied Result
		0
0	0	0

- The last column and Totals row are not editable. The DLL multiplyBy function is called after an editable cell has been updated
- Rows are added using the Functions combo box

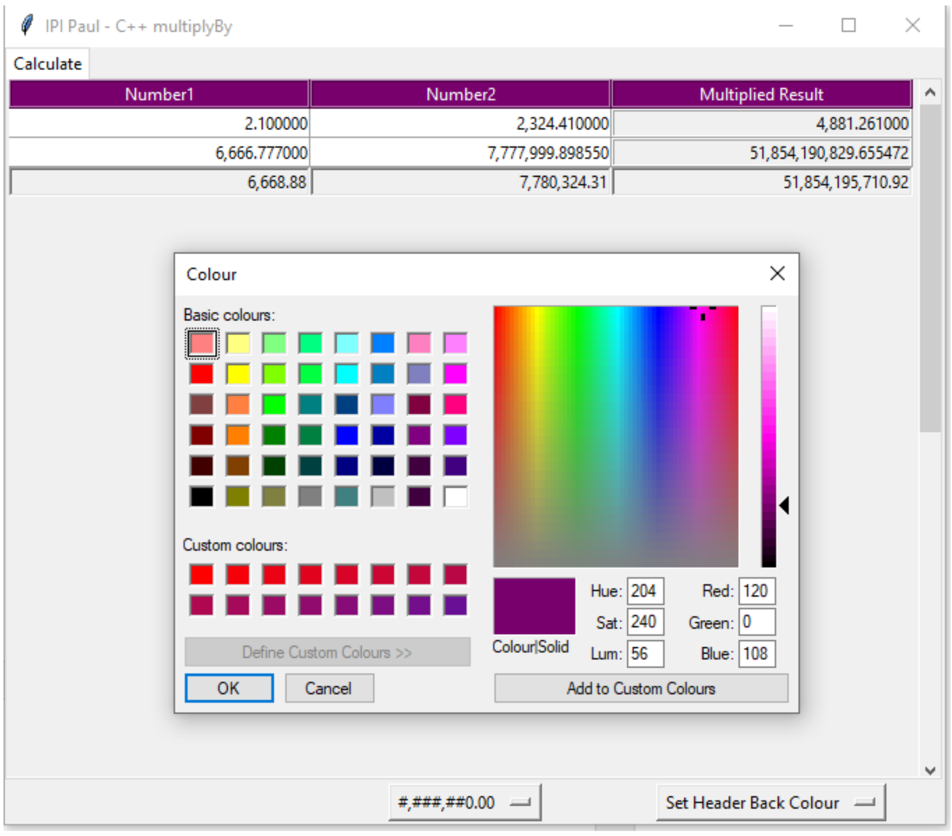


Number1	Number2	Multiplied Result
2.100000	2,324.410000	4,881.261000
		0
2.100000	2,324.410000	4,881.261000

- Formatting is applied to the Totals row on changing the Format combo. The row data does not fully reflect this formatting until exported to Outlook or Word
- I have made use of tkinter.colorchooser askcolor. So when the selecting Set Header Colour items from the Functions combo, the table header is updated and any Header colours in exported tables reflect this change



Number1	Number2	Multiplied Result
2.100000	2,324.410000	4,881.261000
6,666.777000	7,777,999.898550	51,854,190,829.655472
6,668.877000	7,780,324.308550	51,854,195,710.916473



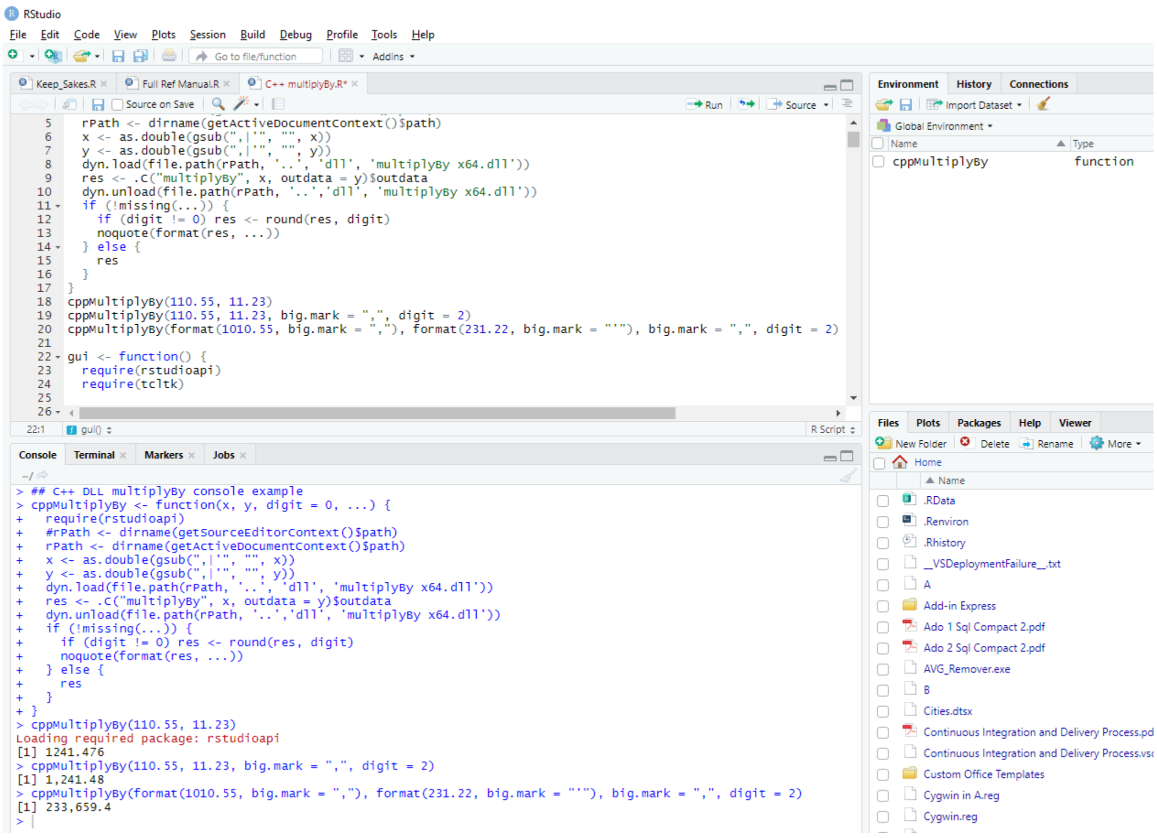
Number1	Number2	Multiplied Result
2.100000	2,324.410000	4,881.261000
6,666.777000	7,777,999.898550	51,854,190,829.655472
6,668.88	7,780,324.31	51,854,195,710.92

Hue: 204 Sat: 240 Lum: 56
Red: 120 Green: 0 Blue: 108

#,###,##0.00 Set Header Back Colour

C++ multiplyBy.R

- Again like Python and Outlook, I have left the original test C++ DLL load function in.
- Using R Studio, you can load and call the C++ multiplyBy functions from the console



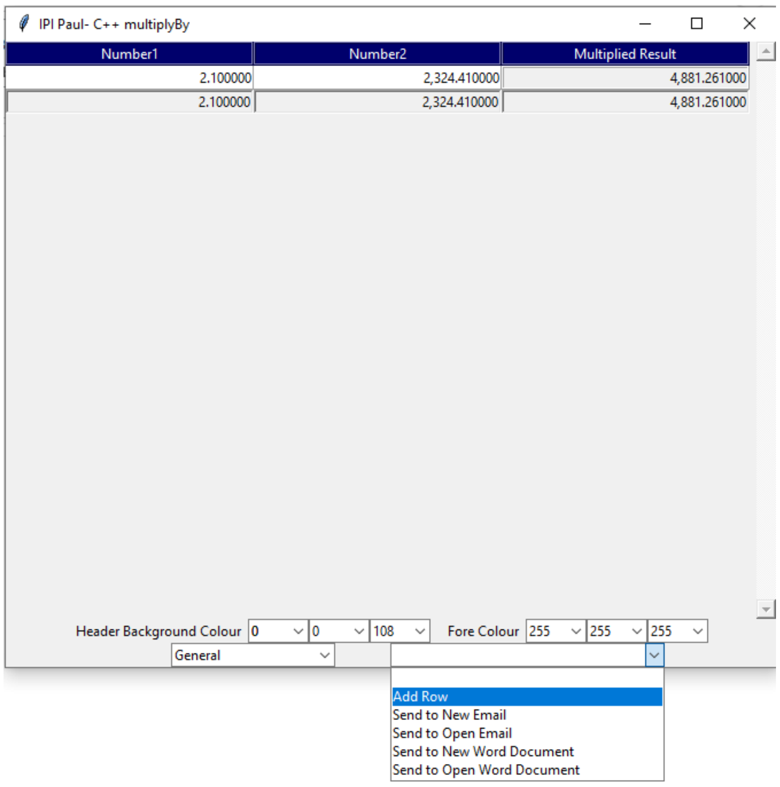
```r
  rPath <- dirname(getActiveDocumentContext()$path)
  x <- as.double(gsub(",|'", "", x))
  y <- as.double(gsub(",|'", "", y))
  dyn.load(file.path(rPath, '..', 'dll', 'multiplyBy x64.dll'))
  res <- .C("multiplyBy", x, outdata = y)$outdata
  dyn.unload(file.path(rPath, '..','dll', 'multiplyBy x64.dll'))
  if (!missing(...)) {
    if (digit != 0) res <- round(res, digit)
    noquote(format(res, ...))
  } else {
    res
  }
}
cppMultiplyBy(110.55, 11.23)
cppMultiplyBy(110.55, 11.23, big.mark = ",", digit = 2)
cppMultiplyBy(format(1010.55, big.mark = ","), format(231.22, big.mark = "'"), big.mark = ",", digit = 2)

gui <- function() {
  require(rstudioapi)
  require(tcltk)
```

```
> ## C++ DLL multiplyBy console example
> cppMultiplyBy <- function(x, y, digit = 0, ...) {
+   require(rstudioapi)
+   #rPath <- dirname(getSourceEditorContext()$path)
+   rPath <- dirname(getActiveDocumentContext()$path)
+   x <- as.double(gsub(",|'", "", x))
+   y <- as.double(gsub(",|'", "", y))
+   dyn.load(file.path(rPath, '..', 'dll', 'multiplyBy x64.dll'))
+   res <- .C("multiplyBy", x, outdata = y)$outdata
+   dyn.unload(file.path(rPath, '..','dll', 'multiplyBy x64.dll'))
+   if (!missing(...)) {
+     if (digit != 0) res <- round(res, digit)
+     noquote(format(res, ...))
+   } else {
+     res
+   }
+ }
> cppMultiplyBy(110.55, 11.23)
Loading required package: rstudioapi
[1] 1241.476
> cppMultiplyBy(110.55, 11.23, big.mark = ",", digit = 2)
[1] 1,241.48
> cppMultiplyBy(format(1010.55, big.mark = ","), format(231.22, big.mark = "'"), big.mark = ",", digit = 2)
[1] 233,659.4
> 
```

Name	Type
cppMultiplyBy	function

- For the tcltk GUI, you will have to run devtools::install_github("dkyleward/RDCOMCLIENT") for the export to Outlook and Word functions to work
- Again, for the GUI you the following are needed to help with file location in loading the C++ DLL.
  - require(rstudioapi)
  - require(tcltk)
  - require(RDCOMClient)
  - rPath <- dirname(getActiveDocumentContext()$path)
  - dyn.load(file.path(rPath, '..', 'dll', 'multiplyBy x64.dll'))
  - tkwait.window(self) This is so that the DLL is not unloaded before the GUI is exited
  - dyn.unload(file.path(rPath, '..','dll', 'multiplyBy x64.dll'))
- Like Python and Java, I have ensured to unload the DLL as gracefully as possible
- I used data.frames for a lot of the elements as it gives me the ability to refer to them using names rather than indices. Unfortunately tcltk would not allow the convention tclvalue(tkget(row[1, (-2, -4, -6)][[i]])) to access individual pointers
- I was not able to figure out the Colour Pallete availability in R to have the same Chooser window or frame as offered in Java and Python

- Like Python, you will have to select **Add New Row** from the Functions combo after the initial row has been filled
- Although Java, Python and R display the same floating point precision errors when using the console for calculations, formatted values populated to cells in the forms are more accurate
- Doing the Ctrl+C on Java, Python and R then pasting the results in Excel and doing a formula to compare the results show them to be accurate

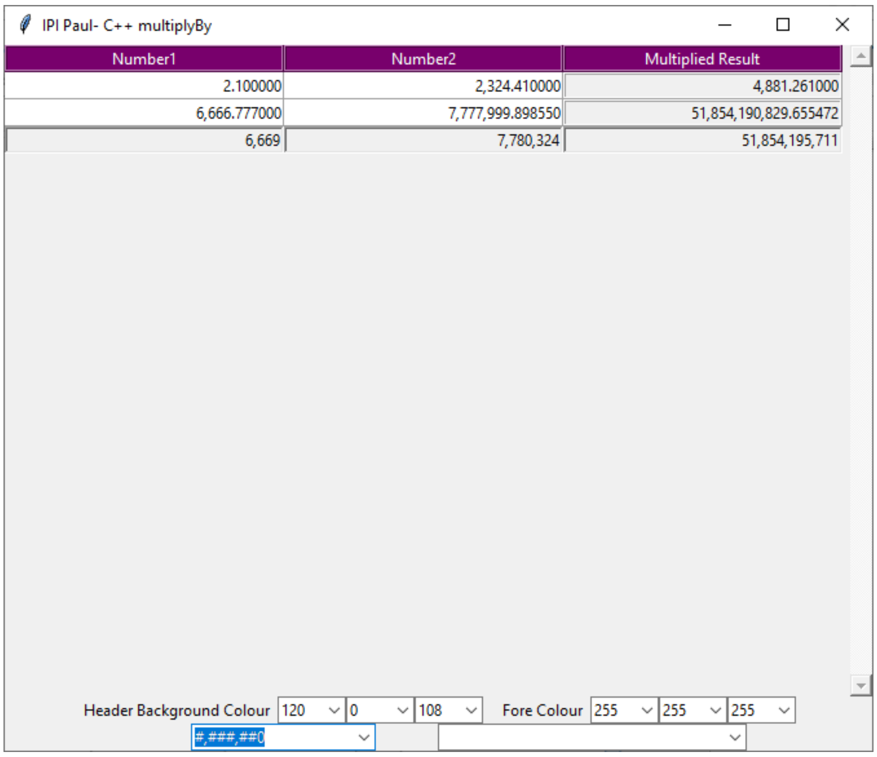

IPI Paul- C++ multiplyBy

Number1	Number2	Multiplied Result
2.100000	2,324.410000	4,881.261000
2.100000	2,324.410000	4,881.261000

Header Background Colour 0 0 108 Fore Colour 255 255 255

General

- Again, changing the RGB combos will affect both the Table headers and those of any Export


IPI Paul- C++ multiplyBy

Number1	Number2	Multiplied Result
2.100000	2,324.410000	4,881.261000
6,666.777000	7,777,999.898550	51,854,190,829.655472
6,669	7,780,324	51,854,195,711

Header Background Colour 120 0 108 Fore Colour 255 255 255

#,###,##0