**Project 3613 – H. J. Enterprises – Foreign Parts**

**BUG**

**FIXED – Needs testing**

**TESTED**

**A. Component: Foreign Parts Screen**

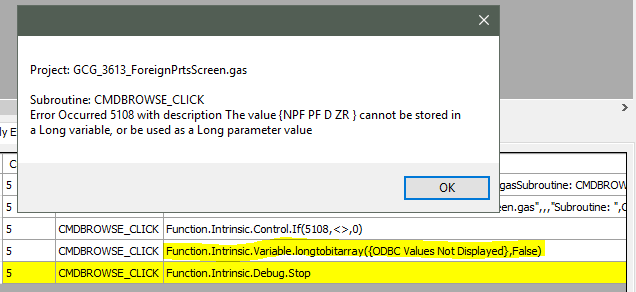
**Date: Dec 13rd, 14th, 15th, 16th /2016**

1. Add button should be inactive till the Part Information has been filled in.DRH => OK
2. There is no button to Remove rows in the grid DRH => OK
3. Zone\_Status column in the grid should display characters instead of integer. For example, if user chooses PF and D, the integer for managing the combination of the 4 checkboxes will be 6, but Zone\_Status should display “PD,D”. DRH => OK
4. Textboxes: HTSUS Rate, HTSUS Weight, Conver. Factor, Add/CVD Rate should have been configured with Max Length, and/or number convertible validation. Otherwise, when we save DataTable to database, it will crash.DRH => OK

**Date: Dec 28th, 29th, 30th/ 2016:**

5) After adding a row to the grid, the ADD button should return to be inactive. (should do this at the end of subroutine **cmdADD\_Click**) DRH

6) When we retrieve a part that is already existed in the grid and has a Zone\_Status value (combination of NPF/PF/D/ZR), it causes error (subroutine **cmdBrowse\_Click**):



That error comes from this line of code:

Function.Intrinsic.Variable.longtobitarray(v.ODBC.conx!rstPart.FieldVal!Zone\_Status,V.Local.bRet)] which is used to retrieve data and set the checkboxes. As the Zone\_Status datatype is String (from CHAR in database table), we should use other way, for example, splitting the Zone\_Status string. DRH

7) When we add a part that is existed in the grid, user should be **notified.** DRH

8) Remove button should be at the end of each row. DRH- Functionality is there-choose a row

**B. Component: Taxes Paid Parts Screen**

**Date: Dec 28th, 29th, 30th/ 2016:**

1) If user takes a blank QTY cell and enters a numeric value, then he changes his mind and wants it to be blank, then the grid will not accept the blank string. The grid only accepts numeric input for QTY. You might think of inputting a 0 value, but it will cause the grid to remove that Zero QTY row if that part no longer exists in the ForeignPart table (as a business rule).

We should add an event handler CellValueChanged to the grid to handle this validation.

~This grid function is ran before any event is called, therefore we are unable to add a check to prevent this error from happening unless we reformatted the grid data type for the column and then we would need to add other checks to make sure we input numbers in the grid in the cellvaluechanged event. I think it will just be easier and cleaner to just let them type in a zero if they don’t want any data on that cell. FEQ~

**C. Component: Estimated Ship Qty**

**rewrote DRH**

**DRH** did rewrite the whole script. The bugs below are in the old script stated here just for documentation purpose.

1) The two requirements (Only the Parts that are also in the Foreign Parts table will be added to the grid; and the largest value of usage will be selected then multiplied by 1.3 and fill the Quantity field in the grid) are implemented in the subroutine txtPart\_LostFocus. This means they are implemented only when the Part textbox is clicked, then entered manually without the browser, then lost focus. This will not be implemented when user select a part from the browser button. So we should call this subroutine (txtPart\_LostFocus) at the end of the subroutine cmdBrowse\_Click.

Furthermore, in the txtPart\_LostFocus, we should add:

v.Local.sPartsHold.Declare(string)

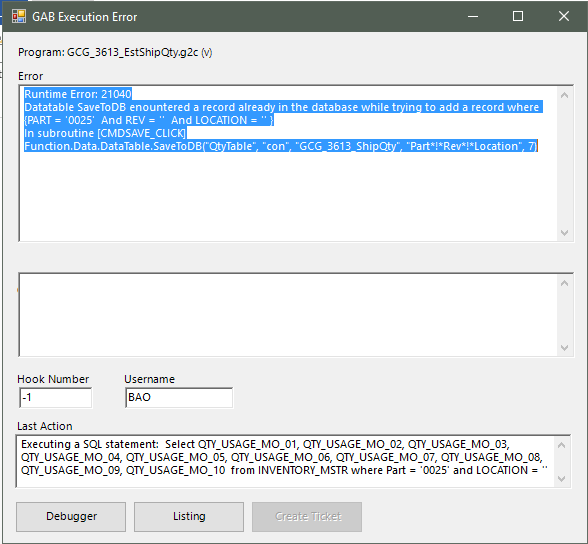
Function.Intrinsic.String.RPad(V.Screen.F\_Ship\_Qty!txtPart.Text, " ", 22, v.Local.sPartsHold)

Before the line:

Function.Data.Dictionary.ReturnKeyFromValue("Part", v.Local.sPartsHold, Variable.Local.sPartRev)

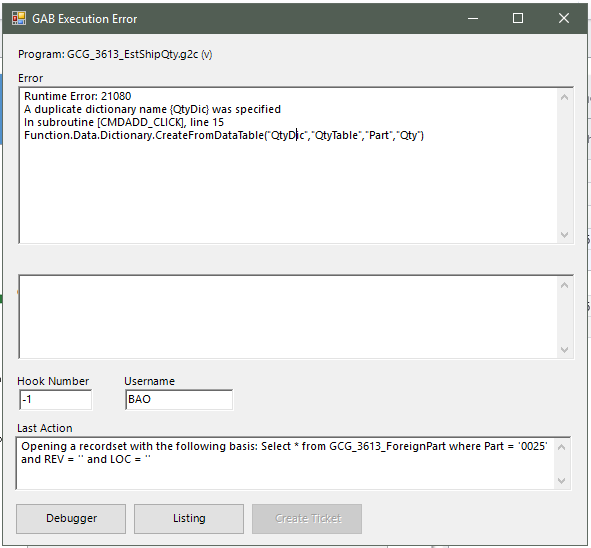
And these codes should be wrapped in an exception handler, just in case the string value of the Part textbox manually entered is not in the Part Dictionary.

2) If we add a Part to the grid, then we hit Save button twice, the program spits error:



To prevent this from happening, we should make the Save button disabled after the saving, and it will be enabled again after we hit the Add button.

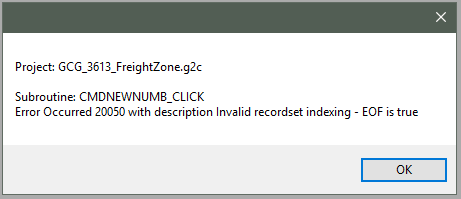
3) If we Add one particular Part twice, error occurs:



**D. Component: Freight Zone Screen**

**Date: Dec 13rd, 14th, 15th, 16th /2016**

1) Subroutine cmdNewNumb\_Click: after OpenLocalRecordset from table GCG\_3613\_Freightzone, should have exception handler in case the recordset has no data. Otherwise the MoveLast would cause error like this: FEQ



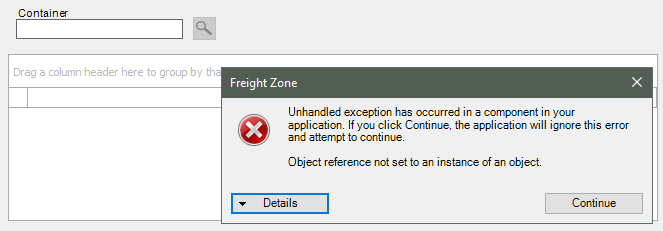
2) If user clicks New to create a Zone, a number will be generated and added to the Dropdownlist. Then if user forgets to click Save, next time when user select that zone number from the dropdownlist, system will crash, because data for many columns except the Zone\_Admission in table GCG\_3613\_Freightzone has not been initialized. FEQ

3) Air Freight datatype in GCG\_3613\_Freightzone should be Bit. FEQ

4) Max Length and number validation for the textboxes that are allowed to enter.

5) I need example data to test on Container, PO, Parts grids

6) Still has bug when cursor is inside the grid below Container textbox



**E. Component: Update Qty in Inventory**

**Date: Dec 13rd, 14th, 15th, 16th /2016**

**rewrote DRH**

**DRH** did rewrite the whole script. The bugs below are in the old script stated here just for documentation purpose.

1) Requirement “If there are Parts that exist in the Foreign Parts table, then the program will check for the Container and Purchase Order in the Zone Container table” has not been checked in the subroutine Checkzone.

2) Typo in the closing of the subroutine UpdateQtyInInv

Program.Sub.UpdateQtyInInv.Start

Program.Sub.UpdateUpdateQtyInInv.End

3) Subroutine UpdateQtyInInv was created but has not been called after all the conditions have been met. It should be called in the Main after the line F.Intrinsic.Control.CallSub(checkzone)

**Dec 28, 2016**

**From the script rewrote by DRH**

*“If none of the Parts are in the Foreign Parts table, the receiving process will continue as normal”,* that means in the subroutine CheckPart, we should implement this test after the line F.Data.Dictionary.Close("dict") (of QTY dictionary) before going further. If every parts are not in the Foreign Parts table, the program should exit the subroutine CheckPart without calling UpdateQtyInInv to update the ZonePart table.

-Added a dataview that filtered upon whether the parts were in the foreign parts and if empty, the program will end. FEQ

**F. Component: Update Foreign Part Quantities in Taxes Paid and Containers**

**Date: Dec 13rd, 14th, 15th, 16th /2016**

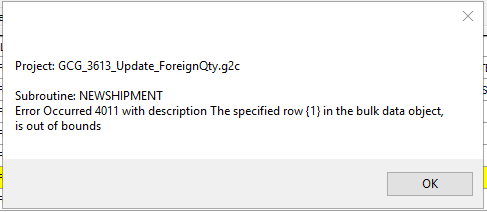
1) Line 27 in Main: Function.Intrinsic.Control.CallSub(deleteshp) => should be deleted => OK

2) Line 165 in subroutine ForeignPartCheck: V.Local.sRet.Set("TEST") => should be deleted => OK

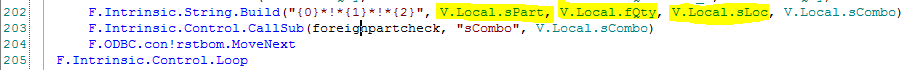
**Date: Dec 28th, 29th, 30th/ 2016:**

3) In subroutine **NewShipment**:

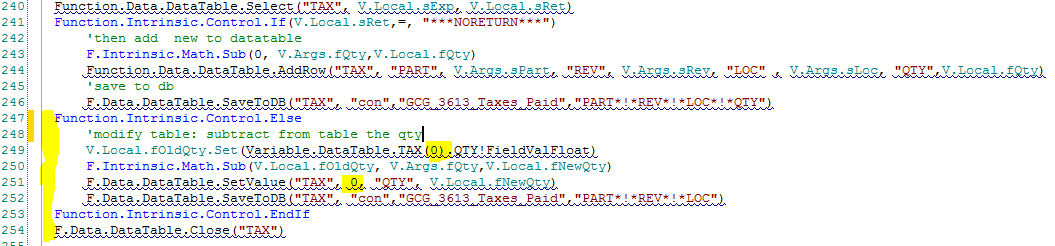
The upper bound of the For loop should be V.Local.iLim – 1. Otherwise, there will be error of “bulk data object…out of bound”. FEQ added extention ‘.—‘



4) In the subroutine **BOMCheck:**



when building the sCombo string, the order should be sPart, sLoc, fQty ( instead of sPart, fQty, sLoc), so that the right order string can be received and split to feed the arguments in the **ForeignPartCheck** subroutine. FEQ

5) In the subroutine **Taxes\_Paid** :

We should select the **right** row of the target part from data table TAX which is pulled from the GCG\_3613\_Taxes\_Paid database table. Even though we apply the function F.Data.DataTable.Select before the **If** section, in the **Else** section, the TAX data table is still the whole table of GCG\_3613\_Taxes\_Paid. So then, if we set value to row 0, it will not be the **right** row.

We should use DataView with filter to select the **right** row and apply some codes in the **Else** clause like this:

Function.Data.DataView.Create("TAX", "TaxDataView")

Function.Data.DataView.SetFilter("TAX", "TaxDataView", v.Local.sExp)

V.Local.fOldQty.Set(Variable.DataView.TAX!TaxDataView(0).QTY!FieldValFloat)

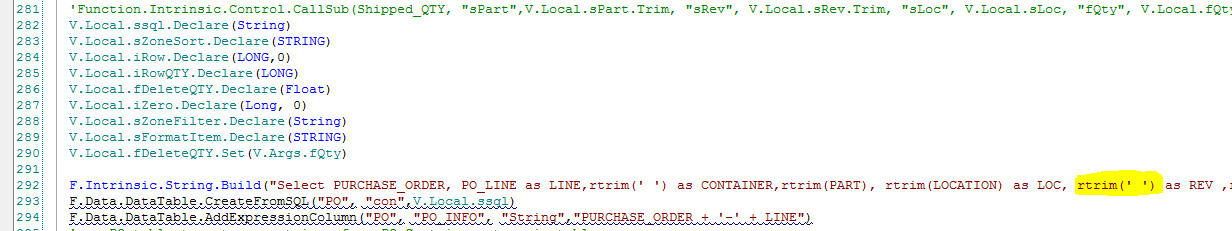
F.Intrinsic.Math.Sub(V.Local.fOldQty, V.Args.fQty,V.Local.fNewQty)

Function.Data.DataView.SetValue("TAX", "TaxDataView", 0, "QTY", V.Local.fNewQty)

F.Data.DataTable.SaveToDB("TAX", "con","GCG\_3613\_Taxes\_Paid","PART\*!\*REV\*!\*LOC")

FEQ

6) Subroutine Shipped\_QTY:



The blank REV column is created in the Select statement, then is saved back to database table GCG\_3613\_ZONEPARTS:



That will cause every value of the REV column in the GCG\_3613\_ZONEPARTS to be blank. This is not expected.

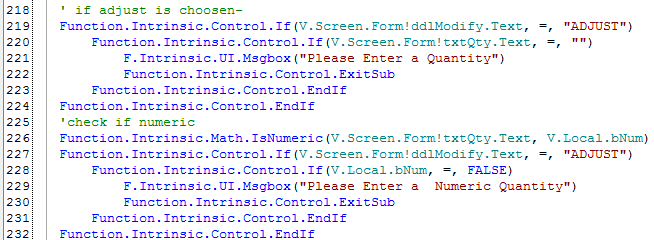
The REV column should be deleted, or be filled in between the 2 actions.

Part was being passed as just part without rev, but PO table uses part/rev so I added portion to combine into gss part for the query, and then filled the DT with Part rev before saving into custom table ~FEQ

**G. Component: Scrap and Adjustments**

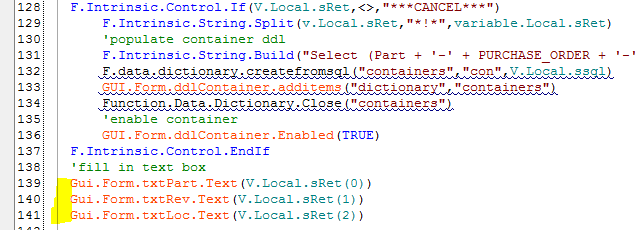
**Dec 29, 2016:**

1) In subroutine **cmdSave\_Click**

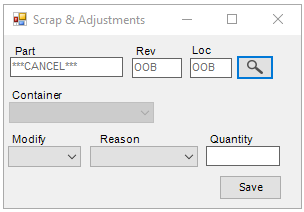


The section **“check if numeric”** should be inside the section **“if adjust is chosen” If…EndIf clause**, because we only check the value if the Adjust is selected. Otherwise, if Scrap is selected, the subroutine will unexpectedly continue to check numeric, and in that case, with an empty Quantity, the program will exit the subroutine at that point. FEQ

2) In subroutine **cmdPartBrowser\_Click**



The yellow parts above should be inside the If ..EndIf clause, meaning we only set the TextBoxes when there is value retrieved. Otherwise, in case there is no value retrieved (the GCG\_3613\_ZONEPARTS is empty), the TextBoxes will be set like following, which is not expected.



We should also create an Else clause for the case where that table is empty, to notify users. FEQ moved the set lines in the loop and added an else statement to make them blank if they cancel

3) Max Length for Quantity Texboxes should be set -FEQ