Open Video : https://youtu.be/A1H5zJFcx9o

Download Below Code and Paste on the Code Window :

Before paste Insert List boxes, Change Name and insert labels to display sales figure. Watch Video First. If Line start with ‘ meant it’s Comments not code.

```
Dim ws As Worksheet
Dim wsData As Worksheet
Dim wsCust As Worksheet
Dim wsSR As Worksheet
Dim ProductID As String
Dim CustID As String
Dim SrId As String
```

```
Private Sub CmdExt_Click()
    Unload Me
End Sub


Private Sub CustListBox_Click()
    On Error Resume Next
    Dim selectedCustRow As Long
    Set wsSRcust = ThisWorkbook.Sheets("Customer_Master")

    selectedCustRow = CustListBox.ListIndex + 1

    CustID = wsCust.Cells(selectedCustRow + 1, 1).value
    CustName = wsCust.Cells(selectedCustRow + 1, 2).value
    'Price = wssr.Cells(selectedRow + 1, 3).value
    'DiscRate = wssr.Cells(selectedRow + 1, 4).value

    LblCustID.Caption = CustID
    LblCustName.Caption = CustName

    PopulateCustSales

    If Err.Number <> 0 Then
        MsgBox "Error: " & Err.Description
```

```
    End If
    On Error GoTo 0

End Sub

Private Sub ProdListBox_Click()
    On Error Resume Next
    Dim selectedRow As Long
    Set ws = ThisWorkbook.Sheets("Product_Master")
    selectedRow = ProdListBox.ListIndex + 1

    ProductID = ws.Cells(selectedRow + 1, 1).value
    ProductName = ws.Cells(selectedRow + 1, 2).value
    Price = ws.Cells(selectedRow + 1, 3).value
    DiscRate = ws.Cells(selectedRow + 1, 4).value

    Label7.Caption = ProductID
    Label8.Caption = ProductName
    Label9.Caption = "Rs." & Format(Price, "#0.00)")
    Label10.Caption = DiscRate & "%"
    Label11.Caption = TotalSales
    Label27.Caption = Label8.Caption & " Sales by :"
    Label28.Caption = Label8.Caption & " Sales by :"
    PopulateProdSales
```

```
    If Err.Number <> 0 Then
        MsgBox "Error: " & Err.Description
    End If
    On Error GoTo 0

End Sub

Private Sub PopulateProdSales()
    Dim i As Long
    Set wsData = ThisWorkbook.Sheets("Data")
    TotalSales = 0
    Totalpaid = 0
    Totaldues = 0

    ' Loop through the "Data" sheet and calculate total sales for the selected
Product_ID
    For i = 2 To wsData.Cells(wsData.Rows.Count, "D").End(xlUp).row '
Assuming data starts from row 2
        If wsData.Cells(i, "F").value = ProductID Then
            If wsData.Cells(i, "H").value = "Y" Then
                Totalpaid = Totalpaid + wsData.Cells(i, "D").value ' Amount (Col
D)
        Else
                Totaldues = Totaldues + wsData.Cells(i, "D").value
            End If
        End If
```

```
    Next i
    TotalSales = Totalpaid + Totaldues

    PaidPercent = 0
    DuePercent = 0
    PaidPercent = (Totalpaid / TotalSales) * 100
    DuePercent = (Totaldues / TotalSales) * 100

    ' Display total sales
    Label11.Caption = "Rs." & Format(TotalSales, "#0.00")
    Label12.Caption = "Rs." & Format(Totalpaid, "#0.00")
    Label13.Caption = "Rs." & Format(Totaldues, "#0.00")
    Label16.Caption = Format(PaidPercent, "(#0") & "%)"
    Label17.Caption = Format(DuePercent, "(#0") & "%)"

End Sub

Private Sub PopulateSRSales()
    Dim i As Long

    Set wsData = ThisWorkbook.Sheets("Data")

    TotalsrSales = 0
    Totalpaid = 0
    Totaldues = 0
```

```
    ' Loop through the "Data" sheet and calculate total sales for the selected
Product_ID
    For i = 2 To wsData.Cells(wsData.Rows.Count, "D").End(xlUp).row
        If wsData.Cells(i, "I").value = SrId Then
            If wsData.Cells(i, "F").value = ProductID Then
                TotalsrSales = TotalsrSales + wsData.Cells(i, "D").value
            End If
        End If
    Next i

    LblSalesValue.Caption = "Rs." & Format(TotalsrSales, "#0.00")

End Sub

Private Sub PopulateCustSales()
    Dim i As Long
    Set wsData = ThisWorkbook.Sheets("Data")

    TotalcustSales = 0
    Totalpaid = 0
    Totaldues = 0

    ' Loop through the "Data" sheet and calculate total sales for
    ' the selected Sales Cust_ID and selected Product_ID
    For i = 2 To wsData.Cells(wsData.Rows.Count, "D").End(xlUp).row
```

```
        If wsData.Cells(i, "A").value = CustID Then
            If wsData.Cells(i, "F").value = ProductID Then
                TotalcustSales = TotalcustSales + wsData.Cells(i, "D").value
            End If
        End If
    Next i

    LblCustSales.Caption = "Rs." & Format(TotalcustSales, "#0.00")

End Sub

Private Sub SRListBox_Click()
    On Error Resume Next
    Dim selectedSRRow As Long
    Set wsSR = ThisWorkbook.Sheets("SR_Master")
    selectedSRRow = SRListBox.ListIndex + 1

    SrId = wsSR.Cells(selectedSRRow + 1, 1).value
    srName = wsSR.Cells(selectedSRRow + 1, 2).value

    LblSRID.Caption = SrId
    LblSRName.Caption = srName

    PopulateSRSales
```

```
    If Err.Number <> 0 Then
        MsgBox "Error: " & Err.Description
    End If
    On Error GoTo 0
End Sub

Private Sub UserForm_Initialize()
    ListpopulateProd
    ListPopulateSR
    ListPopulateCust
End Sub

Private Sub ListpopulateProd()

    Dim lastRow As Long
    Dim i As Long

    ' Set a reference to the "Product_Master" worksheet
    Set ws = ThisWorkbook.Sheets("Product_Master")

    ' Find the last used row in the "Product_Master" sheet
    lastRow = ws.Cells(ws.Rows.Count, "B").End(xlUp).row ' Assuming data starts from column B

    ' Loop through the "Product_Master" sheet and populate the ProductList ListBox
```

```
    For i = 2 To lastRow ' Assuming the data starts from row 2
        ProdListBox.AddItem ws.Cells(i, 2).value ' Product_Name (second column)
    Next i

    ' Initialize variables to store select'ed values
    ProductID = ""
    ProductName = ""
    Price = 0
    DiscRate = 0

End Sub

Private Sub ListPopulateSR()

    Dim lastSRRow As Long
    Dim i As Long

    ' Set a reference to the "SR_Master" worksheet
    Set wsSR = ThisWorkbook.Sheets("SR_Master")

    ' Find the last used row in the "SR_Master" sheet
    lastSRRow = wsSR.Cells(wsSR.Rows.Count, "A").End(xlUp).row ' Assuming data starts from column B
```

```
    ' Loop through the "SR_Master" sheet and populate the ProductList ListBox
    For i = 2 To lastSRRow ' Assuming the data starts from row 2
        SRListBox.AddItem wsSR.Cells(i, 2).value ' SR_Name (second column)
    Next i

    ' Initialize variables to store select'ed values
    SrId = ""
    srName = ""

End Sub

Private Sub ListPopulateCust()

    Dim lastCustRow As Long
    Dim j As Long

    ' Set a reference to the "Customer_Master" worksheet
    Set wsCust = ThisWorkbook.Sheets("Customer_Master")

    ' Find the last used row in the "Customer_Master" sheet
    lastCustRow = wsCust.Cells(wsCust.Rows.Count, "A").End(xlUp).row ' Assuming data starts from column A

    ' Loop through the "Customer_Master" sheet and populate the Customer ListBox
```

```
    For j = 2 To lastCustRow ' Assuming the data starts from row 2
        CustListBox.AddItem wsCust.Cells(j, 2).value ' Customer_Name (second column)
    Next j

    ' Initialize variables to store select'ed values
    CustID = ""
    CustName = ""

End Sub
```

Here are some definition : (Below code not for Copy and paste)

```
Definition the below command :
Private Sub UserForm_Initialize()
ListpopulateProd
ListPopulateSR
ListPopulateCust
End Sub
```

The code you provided is part of a UserForm in Excel VBA and specifically relates to the UserForm_Initialize event handler. Here's the definition and explanation:

Private Sub UserForm_Initialize()

This line marks the beginning of a subroutine (procedure) in VBA code that is associated with the Initialize event of a UserForm. The Private keyword

indicates that this subroutine is accessible only within the module in which it is defined, and it is not accessible from other modules.

ListpopulateProd

This is a VBA function or subroutine call. It's calling a subroutine or function named ListpopulateProd. The purpose of this call is to execute the code within the ListpopulateProd subroutine or function. The absence of parentheses indicates that it's not passing any arguments to the subroutine.

ListPopulateSR

Similar to the previous line, this is another subroutine or function call, and it's calling a subroutine named ListPopulateSR. It's used to execute the code within the ListPopulateSR subroutine or function.

ListPopulateCust

Again, this is a subroutine or function call, and it's calling a subroutine named ListPopulateCust. Like the previous calls, it's used to execute the code within the ListPopulateCust subroutine or function.

Overall, the UserForm_Initialize event handler is being used to initialize or set up the UserForm when it is loaded or opened. It accomplishes this by calling three separate subroutines or functions (ListpopulateProd, ListPopulateSR, and ListPopulateCust) to populate or configure elements within the UserForm. These subroutines likely contain code to load data, set properties, or perform other tasks necessary to initialize the UserForm.

```
Private Sub ProdListBox_Click()
    On Error Resume Next
    Dim selectedRow As Long
    Set ws = ThisWorkbook.Sheets("Product_Master")
    selectedRow = ProdListBox.ListIndex + 1

    ProductID = ws.Cells(selectedRow + 1, 1).value
    ProductName = ws.Cells(selectedRow + 1, 2).value
```

```
    Price = ws.Cells(selectedRow + 1, 3).value
    DiscRate = ws.Cells(selectedRow + 1, 4).value

    Label7.Caption = ProductID
    Label8.Caption = ProductName
    Label9.Caption = "Rs." & Format(Price, "#0.00)")
    Label10.Caption = DiscRate & "%"
    Label11.Caption = TotalSales
    Label27.Caption = Label8.Caption & " Sales by :"
    Label28.Caption = Label8.Caption & " Sales by :"
    PopulateProdSales

    If Err.Number <> 0 Then
        MsgBox "Error: " & Err.Description
    End If
    On Error GoTo 0

End Sub
```

The above code is a VBA subroutine associated with the Click event of a ListBox control named ProdListBox in an Excel UserForm. It appears to handle the interaction when a user clicks on an item in the ProdListBox. Here's a description of what this code does:

When a user clicks on an item in the ProdListBox, this event handler is triggered.

It retrieves the selected row index from the ListBox (selectedRow) and then retrieves data from the "Product_Master" worksheet based on that index. The data includes Product ID, Product Name, Price, and Discount Rate.

The retrieved data is then displayed in various labels within the UserForm, such as Label7, Label8, Label9, Label10, Label11, Label27, and Label28.

It calls a subroutine named PopulateProdSales, which appears to be responsible for populating some information related to product sales.

Error handling is used to capture and display any errors that might occur during the process.

Overall, this code is designed to update the UserForm with information about a selected product from the "Product_Master" worksheet when an item in the ProdListBox is clicked.

Description Same as above for SRListBox_Click and CustListBox_Click

```
Private Sub PopulateProdSales()
    Dim i As Long
    Set wsData = ThisWorkbook.Sheets("Data")
    TotalSales = 0
    Totalpaid = 0
    Totaldues = 0

    ' Loop through the "Data" sheet and calculate total sales for the selected Product_ID
    For i = 2 To wsData.Cells(wsData.Rows.Count, "D").End(xlUp).row ' Assuming data starts from row 2
        If wsData.Cells(i, "F").value = ProductID Then
            If wsData.Cells(i, "H").value = "Y" Then
```

```
            Totalpaid = Totalpaid + wsData.Cells(i, "D").value ' Amount (Col
D)
        Else
            Totaldues = Totaldues + wsData.Cells(i, "D").value
          End If
        End If
    Next i
    TotalSales = Totalpaid + Totaldues

    PaidPercent = 0
    DuePercent = 0
    PaidPercent = (Totalpaid / TotalSales) * 100
    DuePercent = (Totaldues / TotalSales) * 100

    ' Display total sales
    Label11.Caption = "Rs." & Format(TotalSales, "#0.00")
    Label12.Caption = "Rs." & Format(Totalpaid, "#0.00")
    Label13.Caption = "Rs." & Format(Totaldues, "#0.00")
    Label16.Caption = Format(PaidPercent, "(#0") & "%)"
    Label17.Caption = Format(DuePercent, "(#0") & "%)"

End Sub
```

The provided code is a VBA subroutine named PopulateProdSales. This subroutine appears to calculate and display various sales-related information based on data from the "Data" worksheet, with a focus on a selected ProductID. Here's an explanation of what this code does:

This subroutine calculates various sales-related metrics for a selected ProductID based on data from the "Data" worksheet. It calculates the total sales, total amount paid (Totalpaid), and total dues (Totaldues) for the selected product.

It also calculates the percentage of total sales that is paid (PaidPercent) and the percentage that is due (DuePercent).

These calculated values are then displayed in various labels within the UserForm (Label11, Label12, Label13, Label16, and Label17).

The data for the calculations is obtained by looping through the "Data" sheet and checking for rows where the ProductID matches the selected product. Depending on whether the payment status is marked as "Y," the amount is added to either Totalpaid or Totaldues.

Overall, this code populates the UserForm with sales-related information based on the selected product from the "Data" sheet and calculates and displays totals and percentages related to sales and payments.

```
Dim ws As Worksheet
Dim wsData As Worksheet
Dim wsCust As Worksheet
Dim wsSR As Worksheet
Dim ProductID As String
Dim CustID As String
Dim SrId As String

Private Sub CmdExt_Click()
    Unload Me
End Sub
```

The provided code snippet declares several variables and defines a subroutine associated with a UserForm in Excel VBA. Here's an explanation of the code:

Variable Declarations:

ws, wsData, wsCust, and wsSR are variables that will represent Excel worksheets. These variables are declared at the module level, making them accessible to all procedures within the module.

ProductID, CustID, and SrId are variables that will store string values.

Subroutine:

Private Sub CmdExt_Click(): This line marks the beginning of a subroutine (procedure) named CmdExt_Click. This subroutine is associated with a control named CmdExt and is executed when the control is clicked.

Subroutine Body:

Unload Me: This line of code unloads (closes) the UserForm associated with the current instance, which is achieved by calling Unload Me. This action effectively closes the UserForm when the "CmdExt" control (e.g., a button) is clicked.

Overall, the code you provided declares variables for worksheets and string values and defines a subroutine that closes (unloads) the UserForm when a specific control (likely a button) is clicked. The worksheets (ws, wsData, wsCust, wsSR) and string variables (ProductID, CustID, SrId) can be used within other procedures to store and manipulate data related to the UserForm.

Gautam Banerjee

“Helping beginners learn something new is a great way to share your knowledge and make a positive impact”.

Email: gincom1@yahoo.com

If you have any queries, please visit our "Contact Us" page.

Thank You. See you again!

PLEASE DONATE

Gautam Banerjee

Age: 63

Pay by UPI

9748327614