Video Link : https://youtu.be/j4voq5KbPSs

Last two Videos Link :

1. https://youtu.be/xJ0_vbWUnEA

2. https://youtu.be/YDWwF6FAIGg

Presented by Gautam Banerjee, E-mail: gincom1@yahoo.com

How you can create a search and filter UserForm in Excel VBA to filter a list of images based on keywords:

Set Up Your Data: First, you'll need a worksheet with a list of images and associated data. For this example, let's assume you have a worksheet named "ImageData" with the following columns: "Image Name," "Location," and "Keywords."

Create a UserForm: Insert a UserForm in your Excel workbook. You can do this by going to the VBA editor (ALT + F11), right-clicking on your VBAProject (your workbook's name), and selecting Insert > UserForm.

Design Your UserForm: On your UserForm, add the following controls:

Label and TextBox for entering search keywords.

ComboBox for selecting a location.

ListBox for displaying the filtered images.

CommandButton for executing the search/filter.

You can add more controls as per your requirement.

You can use the toolbox in the VBA editor to insert these controls onto your UserForm.

Overall this VBA code appears to be well-structured and organized. It's designed to create a user-friendly interface for browsing and searching images based on location, keywords, and categories. Here's a brief overview of what each subroutine does:

ComboLocation_Change: This subroutine is triggered when the user selects a location from the combo box. It clears the list box and populates it with image names that match the selected location from the worksheet.

CommandButton1_Click: This subroutine is called when the "Refresh All" button is clicked. It clears and refreshes the combo box, list box, and search text box by calling the RefreshAll subroutine.

CommandButton2_Click: This subroutine unloads the current user form and shows another user form named MainMenuForm.

Image2_Click: This subroutine unloads the current user form when the user clicks on Image2.

ListBoxImages_Click: This subroutine is triggered when the user clicks on an item in the list box. It displays the selected image in an image control and updates labels with category and introduction information based on the selected image's data in the worksheet.

UserForm_Initialize: This subroutine initializes the user form. It populates the combo box with unique location values from the worksheet using a collection to ensure uniqueness. It also calls the byDefaultPicture subroutine to display a default image and set default labels.

byDefaultPicture: This subroutine sets and displays a default image in the image control. It also sets a label with a description.

RefreshAll: This subroutine clears and refreshes the combo box, list box, and search text box by calling the UserForm_Initialize subroutine.

CommandButtonSearch_Click: This subroutine is triggered when the user clicks the "Search" button. It searches for images based on the selected location and keyword, populating the list box with matching image names.

The code is well-commented, making it easy to understand. It provides a user-friendly interface for selecting and viewing images based on various criteria. If you have any specific questions or need further assistance with any part of the code, please feel free to ask in Comment Section.

आप कीवर्ड के आधार पर छवियों की सूची को फ़िल्टर करने के लिए Excel VBA में एक खोज और फ़िल्टर UserForm कैसे बना सकते हैं:

अपना डेटा सेट करें: सबसे पहले, आपको छवियों और संबंधित डेटा की सूची के साथ एक वर्कशीट की आवश्यकता होगी। इस उदाहरण के लिए, मान लें कि आपके पास "इमेजडेटा" नाम की एक वर्कशीट है जिसमें निम्नलिखित कॉलम हैं: "छवि का नाम," "स्थान," और "कीवर्ड।"

एक यूजरफॉर्म बनाएं: अपनी एक्सेल वर्कबुक में एक यूजरफॉर्म डालें। आप इसे VBA संपादक (ALT + F11) पर जाकर, अपने VBAProject (आपकी कार्यपुस्तिका का नाम) पर राइट-क्लिक करके और Insert > UserForm का चयन करके कर सकते हैं।

अपना यूजरफॉर्म डिज़ाइन करें: अपने यूजरफॉर्म पर, निम्नलिखित नियंत्रण जोड़ें:

खोज कीवर्ड दर्ज करने के लिए लेबल और टेक्स्टबॉक्स।

किसी स्थान का चयन करने के लिए कॉम्बोबॉक्स।

फ़िल्टर की गई छवियों को प्रदर्शित करने के लिए लिस्टबॉक्स।

खोज/फ़िल्टर निष्पादित करने के लिए कमांडबटन।

आप अपनी आवश्यकता के अनुसार अधिक नियंत्रण जोड़ सकते हैं।

आप इन नियंत्रणों को अपने UserForm पर सम्मिलित करने के लिए VBA संपादक में टूलबॉक्स का उपयोग कर सकते हैं।

कुल मिलाकर यह वीबीए कोड अच्छी तरह से संरचित और व्यवस्थित प्रतीत होता है। इसे स्थान, कीवर्ड और श्रेणियों के आधार पर छवियों को ब्राउज़ करने और खोजने के लिए एक उपयोगकर्ता-अनुकूल इंटरफ़ेस बनाने के लिए डिज़ाइन किया गया है। प्रत्येक सबरूटीन क्या करता है इसका संक्षिप्त विवरण यहां दिया गया है:

कॉम्बोलोकेशन_चेंज: जब उपयोगकर्ता कॉम्बो बॉक्स से कोई स्थान चुनता है तो यह सबरूटीन ट्रिगर हो जाता है। यह सूची बॉक्स को साफ़ करता है और इसे उन छवि नामों से भर देता है जो वर्कशीट से चयनित स्थान से मेल खाते हैं।

CommandButton1_Click: "रिफ्रेश ऑल" बटन पर क्लिक करने पर यह सबरूटीन कॉल किया जाता है। यह RefreshAll सबरूटीन को कॉल करके कॉम्बो बॉक्स, सूची बॉक्स और खोज टेक्स्ट बॉक्स को साफ़ और ताज़ा करता है।

CommandButton2_Click: यह सबरूटीन वर्तमान उपयोगकर्ता फॉर्म को अनलोड करता है और MainMenuForm नामक एक अन्य उपयोगकर्ता फॉर्म दिखाता है।

Image2_Click: जब उपयोगकर्ता Image2 पर क्लिक करता है तो यह सबरूटीन वर्तमान उपयोगकर्ता फॉर्म को अनलोड कर देता है।

ListBoxImages_Click: जब उपयोगकर्ता सूची बॉक्स में किसी आइटम पर क्लिक करता है तो यह सबरूटीन ट्रिगर हो जाता है। यह चयनित छवि को छवि नियंत्रण में प्रदर्शित करता है और वर्कशीट में चयनित छवि के डेटा के आधार पर श्रेणी और परिचय जानकारी के साथ लेबल को अपडेट करता है।

UserForm_Initialize: यह सबरूटीन उपयोगकर्ता फॉर्म को आरंभ करता है। यह विशिष्टता सुनिश्चित करने के लिए एक संग्रह का उपयोग करके वर्कशीट से अद्वितीय स्थान मानों के साथ कॉम्बो बॉक्स को पॉप्युलेट करता है। यह डिफ़ॉल्ट छवि प्रदर्शित करने और डिफ़ॉल्ट लेबल सेट करने के लिए बायडिफॉल्टपिक्चर सबरूटीन को भी कॉल करता है।

बायडिफॉल्टपिक्चर: यह सबरूटीन छवि नियंत्रण में एक डिफ़ॉल्ट छवि सेट और प्रदर्शित करता है। यह विवरण के साथ एक लेबल भी सेट करता है।

RefreshAll: यह सबरूटीन UserForm_Initialize सबरूटीन को कॉल करके कॉम्बो बॉक्स, सूची बॉक्स और खोज टेक्स्ट बॉक्स को साफ़ और ताज़ा करता है।

CommandButtonSearch_Click: जब उपयोगकर्ता "खोज" बटन पर क्लिक करता है तो यह सबरूटीन चालू हो जाता है। यह चयनित स्थान और कीवर्ड के आधार पर छवियों की खोज करता है, मिलान वाले छवि नामों के साथ सूची बॉक्स को पॉप्युलेट करता है।

कोड पर अच्छी तरह से टिप्पणी की गई है, जिससे इसे समझना आसान हो गया है। यह विभिन्न मानदंडों के आधार पर छवियों को चुनने और देखने के लिए एक उपयोगकर्ता-अनुकूल इंटरफ़ेस प्रदान करता है। यदि आपके पास कोई विशिष्ट प्रश्न है या कोड के किसी भाग के बारे में और सहायता की आवश्यकता है, तो कृपया बेझिझक टिप्पणी अनुभाग में पूछें।

Download the Full Code : (Change the control names)

```
Private Sub ComboLocation_Change()
    Dim selectedValue As String
    Dim ws As Worksheet
    Dim i As Long
```

```
    ' Clear the list box
    ListBoxImages.Clear

    ' Get the selected item from the combo box
    selectedValue = ComboLocation.value

    ' Set your worksheet (change the sheet name if needed)
    Set ws = ThisWorkbook.Sheets("ImageData")

    ' Loop through the data in the worksheet
    For i = 2 To ws.Cells(ws.Rows.Count, "A").End(xlUp).row
        If ws.Cells(i, 2).value = selectedValue Then
            ' Add matching items to the list box
            ListBoxImages.AddItem ws.Cells(i, 1).value
        End If
    Next i
End Sub

Private Sub CommandButton1_Click()
    RefreshAll
End Sub

Private Sub CommandButton2_Click()
    Unload Me
    MainMenuForm.Show
```

```
End Sub

Private Sub Image2_Click()
    Unload Me
End Sub

Private Sub ListBoxImages_Click()
     ' Get the selected item from the listbox
    Dim selectedItem As String
    selectedItem = ListBoxImages.value

    ' Construct the image file path based on the selected item
    Dim imagePath As String
    imagePath = "E:\MyPicture\" & selectedItem & ".jpg"

    ' Load and display the image in an Image control (assuming you have an Image control named Image1)
    If Len(Dir(imagePath)) > 0 Then
        Image1.Picture = LoadPicture(imagePath)
    Else
        ' Clear the Image control if the image file does not exist
        Image1.Picture = LoadPicture("") ' This will clear the image control
    End If

    Dim ws As Worksheet
    Dim i As Long
```

```
    ' Set your worksheet (change the sheet name if needed)
    Set ws = ThisWorkbook.Sheets("ImageData")

    ' Loop through the data in the worksheet
    For i = 2 To ws.Cells(ws.Rows.Count, "A").End(xlUp).row
        If ws.Cells(i, 1).value = selectedItem Then
            ' Add matching items to the list box
            Label5.Caption = "Category : " & ws.Cells(i, 3).value
            Label6.Caption = "Introduction : " & ws.Cells(i, 4).value
        End If
    Next i
End Sub

Private Sub UserForm_Initialize()

' Populate the location ListBox with unique location values from the worksheet
 Dim ws As Worksheet
    Set ws = ThisWorkbook.Sheets("ImageData")

    Dim cell As Range
    Dim uniqueLocations As Collection
    Set uniqueLocations = New Collection

    On Error Resume Next
```

```
    For Each cell In ws.Range("B2:B" & ws.Cells(ws.Rows.Count, "B").End(xlUp).row)
        uniqueLocations.Add cell.value, CStr(cell.value)
    Next cell
    On Error GoTo 0

    Dim Loc As Variant ' Declare Loc as a variant

    For Each Loc In uniqueLocations
        ComboLocation.AddItem Loc
    Next Loc

    byDefaultPicture
End Sub

Private Sub byDefaultPicture()

    Dim defaultImagePath As String
 ' Set the default image file path
    defaultImagePath = "E:\MyPicture\Default1.jpg" ' Change to the path of your default image

    ' Load and display the default image in the Image control
    If Len(Dir(defaultImagePath)) > 0 Then ' Check if the default image file exists
        Image1.Picture = LoadPicture(defaultImagePath)
    Else
```

```
        ' Clear the Image control if the default image file does not exist
        Image1.Picture = LoadPicture("") ' This will clear the image control
    End If

    Label6.Caption = "Search and Filter UserForm: Implement a search and filter feature that allows users to search for images based on keywords, locations, or other criteria. This UserForm can include search fields and filter options."
End Sub

Private Sub RefreshAll()

 ' Clear the ComboBox
    ComboLocation.Clear
    ' Clear the ListBox
    ListBoxImages.Clear
    ' Clear the TextBox
    TextBoxKeyword.Text = ""

    ' Reinitialize the ComboBox with data (assuming you have a UserForm_Initialize event)
    UserForm_Initialize
End Sub

Private Sub CommandButtonSearch_Click()
    Dim ws As Worksheet
    Set ws = ThisWorkbook.Sheets("ImageData")
```

```
    Dim keyword As String
    Dim location As String

    keyword = TextBoxKeyword.Text
    location = ComboLocation.value

    ' Clear the ListBox to show new search results
    ListBoxImages.Clear

    ' Loop through the data and add matching images to the ListBox
    Dim i As Long
    For i = 2 To ws.Cells(ws.Rows.Count, "A").End(xlUp).row
        If (location = "" Or ws.Cells(i, "B").value = location) And _
           (InStr(1, ws.Cells(i, "C").value, keyword, vbTextCompare) > 0) Then
            ListBoxImages.AddItem ws.Cells(i, "A").value
        End If
    Next i
End Sub
```

Line by line definitions as under :- (Don’t copy the below Code)

UserForm_Initialize Procedure : The UserForm_Initialize procedure is a special subroutine in Visual Basic for Applications (VBA) associated with a UserForm. This procedure is automatically executed when the UserForm is initialized or loaded. In other words, it runs when the UserForm is displayed or shown to the user.

UserForm_Initialize प्रक्रिया: UserForm_Initialize प्रक्रिया, UserForm से संबद्ध विज़ुअल बेसिक फॉर एप्लिकेशन (VBA) में एक विशेष सबरूटीन है। UserForm आरंभ या लोड होने पर यह प्रक्रिया स्वचालित रूप से निष्पादित होती है। दूसरे शब्दों में, यह तब चलता है जब UserForm उपयोगकर्ता को प्रदर्शित या दिखाया जाता है।

```
Private Sub UserForm_Initialize()

' Populate the location ListBox with unique location values from the worksheet
 Dim ws As Worksheet
   Set ws = ThisWorkbook.Sheets("ImageData")

   Dim cell As Range
   Dim uniqueLocations As Collection
   Set uniqueLocations = New Collection

   On Error Resume Next
   For Each cell In ws.Range("B2:B" & ws.Cells(ws.Rows.Count, "B").End(xlUp).row)
      uniqueLocations.Add cell.value, CStr(cell.value)
   Next cell
   On Error GoTo 0

   Dim Loc As Variant ' Declare Loc as a variant

   For Each Loc In uniqueLocations
      ComboLocation.AddItem Loc
   Next Loc
   byDefaultPicture
```

End Sub

The above code is a UserForm_Initialize procedure in Visual Basic for Applications (VBA). This procedure is associated with a UserForm, and it's executed automatically when the UserForm is initialized or loaded. Its purpose is to set up the initial state of the UserForm and its controls. Let's break down what this specific code is doing:

उपरोक्त कोड विज़ुअल बेसिक फॉर एप्लिकेशन (VBA) में एक UserForm_Initialize प्रक्रिया है। यह प्रक्रिया UserForm से संबद्ध है, और UserForm प्रारंभ या लोड होने पर यह स्वचालित रूप से निष्पादित होती है। इसका उद्देश्य यूजरफॉर्म की प्रारंभिक स्थिति और उसके नियंत्रण स्थापित करना है। आइए देखें कि यह विशिष्ट कोड क्या कर रहा है

Worksheet Setup:

Dim ws As Worksheet: This declares a variable named ws to represent a worksheet.

Set ws = ThisWorkbook.Sheets("ImageData"): It assigns the "ImageData" worksheet of the workbook containing the VBA code to the ws variable. This is the sheet from which data will be read.

Unique Locations Collection:

Dim uniqueLocations As Collection: This declares a variable named uniqueLocations as a Collection. Collections are used to store a unique set of items.

Set uniqueLocations = New Collection: It initializes the uniqueLocations collection.

Loop Through Worksheet Data:

On Error Resume Next: This statement tells VBA to ignore runtime errors and continue executing the code. It's commonly used when trying to add items to a collection that might contain duplicates to avoid raising an error.

For Each cell In ws.Range("B2:B" & ws.Cells(ws.Rows.Count, "B").End(xlUp).Row): This loop goes through each cell in the specified range, which is typically a column containing location values. The loop starts from cell B2 and goes up to the last used cell in column B (assuming it contains the locations).

uniqueLocations.Add cell.Value, CStr(cell.Value): It adds each unique location value to the uniqueLocations collection. The CStr function is used to convert the cell value to a string to ensure uniqueness.

On Error GoTo 0: This statement resets error handling to its default state, allowing VBA to raise errors again.

Populate ComboBox:

Dim Loc As Variant: This declares a variable Loc as a Variant, which will be used to iterate through the unique locations.

For Each Loc In uniqueLocations: This loop goes through each unique location stored in the uniqueLocations collection.

ComboLocation.AddItem Loc: It adds each unique location to a ComboBox control named ComboLocation on the UserForm.

Call Another Procedure:

byDefaultPicture: This line calls another procedure named byDefaultPicture. It's likely used to set default values or display default images and text on the UserForm.

In summary, this UserForm_Initialize procedure prepares the UserForm by populating a ComboBox named ComboLocation with unique location values obtained from the "ImageData" worksheet and then calls another procedure byDefaultPicture to set up default visuals. This is common in UserForm initialization to provide users with predefined options and an initial interface state.

वर्कशीट सेटअप:

वर्कशीट के रूप में डिम डब्ल्यूएस: यह वर्कशीट का प्रतिनिधित्व करने के लिए डब्ल्यूएस नामक एक वेरिएबल घोषित करता है।

सेट ws = ThisWorkbook.Sheets("ImageData"): यह कार्यपुस्तिका की "ImageData" वर्कशीट को ws वेरिएबल में VBA कोड युक्त असाइन करता है। यह वह शीट है जिससे डेटा पढ़ा जाएगा।

अद्वितीय स्थान संग्रह:

संग्रह के रूप में अद्वितीय स्थान को मंद करें: यह संग्रह के रूप में अद्वितीय स्थान नामक एक चर घोषित करता है। संग्रह का उपयोग वस्तुओं के एक अनूठे सेट को संग्रहीत करने के लिए किया जाता है।

अद्वितीय स्थान सेट करें = नया संग्रह: यह अद्वितीय स्थान संग्रह प्रारंभ करता है।

वर्कशीट डेटा के माध्यम से लूप करें:

त्रुटि पर अगला पुनरारंभ करें: यह कथन VBA को रनटाइम त्रुटियों को अनदेखा करने और कोड निष्पादित करना जारी रखने के लिए कहता है। इसका उपयोग आमतौर पर किसी संग्रह में आइटम जोड़ने का प्रयास करते समय किया जाता है जिसमें त्रुटि उत्पन्न होने से बचने के लिए डुप्लिकेट हो सकते हैं।

प्रत्येक सेल के लिए ws.Range("B2:B" & ws.Cells(ws.Rows.Count, "B").End(xlUp).Row में: यह लूप निर्दिष्ट रेंज में प्रत्येक सेल से होकर गुजरता है, जो है आम तौर पर स्थान मान वाला एक कॉलम। लूप सेल B2 से शुरू होता है और कॉलम B में अंतिम उपयोग किए गए सेल तक जाता है (यह मानते हुए कि इसमें स्थान शामिल हैं)।

यूनिकलोकेशन्स.सेल.वैल्यू, सीएसटीआर(सेल.वैल्यू) जोड़ें: यह प्रत्येक अद्वितीय स्थान मान को यूनिकलोकेशन्स संग्रह में जोड़ता है। विशिष्टता सुनिश्चित करने के लिए सेल मान को स्ट्रिंग में बदलने के लिए CStr फ़ंक्शन का उपयोग किया जाता है।

त्रुटि GoTo 0 पर: यह कथन त्रुटि प्रबंधन को उसकी डिफ़ॉल्ट स्थिति में रीसेट करता है, जिससे VBA को फिर से त्रुटियाँ बढ़ाने की अनुमति मिलती है।

कॉम्बोबॉक्स पॉप्युलेट करें:

वेरिएंट के रूप में डिम लोक: यह एक वेरिएबल लोक को वेरिएंट के रूप में घोषित करता है, जिसका उपयोग अद्वितीय स्थानों के माध्यम से पुनरावृत्त करने के लिए किया जाएगा।

अद्वितीय स्थानों में प्रत्येक स्थान के लिए: यह लूप अद्वितीय स्थान संग्रह में संग्रहीत प्रत्येक अद्वितीय स्थान से होकर गुजरता है।

ComboLocation.AddItem Loc: यह प्रत्येक अद्वितीय स्थान को UserForm पर ComboLocation नामक ComboBox नियंत्रण में जोड़ता है।

किसी अन्य प्रक्रिया को कॉल करें:

बायडिफॉल्टपिक्चर: यह लाइन बायडिफॉल्टपिक्चर नाम की एक अन्य प्रक्रिया को कॉल करती है। इसका उपयोग संभवतः डिफ़ॉल्ट मान सेट करने या UserForm पर डिफ़ॉल्ट छवियां और टेक्स्ट प्रदर्शित करने के लिए किया जाता है।

संक्षेप में, यह UserForm_Initialize प्रक्रिया "ImageData" वर्कशीट से प्राप्त अद्वितीय स्थान मानों के साथ ComboLocation नामक कॉम्बोबॉक्स को पॉप्युलेट करके UserForm तैयार करती है और फिर डिफ़ॉल्ट विज़ुअल सेट करने के लिए byDefaultPicture द्वारा एक अन्य प्रक्रिया को कॉल करती है। उपयोगकर्ताओं को पूर्वनिर्धारित विकल्प और प्रारंभिक इंटरफ़ेस स्थिति प्रदान करने के लिए UserForm आरंभीकरण में यह आम है।

```
Private Sub byDefaultPicture()

    Dim defaultImagePath As String

' Set the default image file path

    defaultImagePath = "E:\MyPicture\Default1.jpg" ' Change to the path of your default image
```

```
' Load and display the default image in the Image control
If Len(Dir(defaultImagePath)) > 0 Then ' Check if the default image file exists
    Image1.Picture = LoadPicture(defaultImagePath)
Else
    ' Clear the Image control if the default image file does not exist
    Image1.Picture = LoadPicture("") ' This will clear the image control
End If

Label6.Caption = "Search and Filter UserForm: Implement a search and filter feature that allows users to search for images based on keywords, locations, or other criteria. This UserForm can include search fields and filter options."
End Sub
```

This procedure appears to be setting up default images and text to be displayed on a UserForm. Let's break down what it does:

Default Image Setup:

Dim defaultImagePath As String: Declares a variable defaultImagePath as a string. This variable will store the file path of the default image.

defaultImagePath = "E:\MyPicture\Default1.jpg": Assigns the file path of the default image to defaultImagePath. This path should point to an image file (in this case, "Default1.jpg") located at the specified location ("E:\MyPicture").

If Len(Dir(defaultImagePath)) > 0 Then: Checks if the file specified by defaultImagePath exists using the Dir function. If the length of the result of Dir is greater than zero, it means the file exists.

Image1.Picture = LoadPicture(defaultImagePath): If the default image file exists, it loads and displays this image in an Image control named Image1 on the UserForm.

Label Caption:

Label6.Caption = "Search and Filter UserForm: Implement a search and filter feature that allows users to search for images based on keywords, locations, or other criteria. This UserForm can include search fields and filter options.": This line sets the caption (text) of a Label control named Label6 on the UserForm. It provides a description or informational text about the UserForm's purpose.

In summary, the byDefaultPicture procedure is responsible for setting up the default image and providing an initial description or instruction text on a UserForm. This is a common practice to give users context or guidance when they first interact with a form or application.

डिफ़ॉल्ट छवि सेटअप:

स्ट्रिंग के रूप में मंद defaultImagePath: एक चर defaultImagePath को एक स्ट्रिंग के रूप में घोषित करता है। यह वेरिएबल डिफ़ॉल्ट छवि के फ़ाइल पथ को संग्रहीत करेगा।

defaultImagePath = "E:\MyPicture\Default1.jpg": डिफ़ॉल्ट छवि का फ़ाइल पथ defaultImagePath को निर्दिष्ट करता है। यह पथ निर्दिष्ट स्थान ("E:\MyPicture") पर स्थित एक छवि फ़ाइल (इस मामले में, "Default1.jpg") को इंगित करना चाहिए।

यदि लेन(Dir(defaultImagePath)) > 0 तब: Dir फ़ंक्शन का उपयोग करके जांच करता है कि defaultImagePath द्वारा निर्दिष्ट फ़ाइल मौजूद है या नहीं। यदि Dir के परिणाम की लंबाई शून्य से अधिक है, तो इसका मतलब है कि फ़ाइल मौजूद है।

Image1.Picture = LoadPicture(defaultImagePath): यदि डिफ़ॉल्ट छवि फ़ाइल मौजूद है, तो यह इस छवि को UserForm पर Image1 नामक छवि नियंत्रण में लोड और प्रदर्शित करता है।

लेबल कैप्शन:

लेबल6.कैप्शन = "उपयोगकर्ता प्रपत्र खोजें और फ़िल्टर करें: एक खोज और फ़िल्टर सुविधा लागू करें जो उपयोगकर्ताओं को कीवर्ड, स्थान या अन्य मानदंडों के आधार पर छवियों को खोजने की अनुमति देती है। इस उपयोगकर्ता प्रपत्र में खोज फ़ील्ड और फ़िल्टर विकल्प शामिल हो सकते हैं।": यह पंक्ति कैप्शन सेट करती है (पाठ) यूज़रफॉर्म पर लेबल6 नामक लेबल नियंत्रण का। यह यूज़रफॉर्म के उद्देश्य के बारे में विवरण या सूचनात्मक पाठ प्रदान करता है।

संक्षेप में, बायडिफॉल्टपिक्चर प्रक्रिया डिफ़ॉल्ट छवि स्थापित करने और यूजरफॉर्म पर प्रारंभिक विवरण या निर्देश पाठ प्रदान करने के लिए जिम्मेदार है। जब उपयोगकर्ता पहली बार किसी फॉर्म या एप्लिकेशन के साथ इंटरैक्ट करते हैं तो उन्हें संदर्भ या मार्गदर्शन देना एक आम बात है।

```
Private Sub ComboLocation_Change()
    Dim selectedValue As String
```

```
    Dim ws As Worksheet
    Dim i As Long

    ' Clear the list box
    ListBoxImages.Clear

    ' Get the selected item from the combo box
    selectedValue = ComboLocation.value

    ' Set your worksheet (change the sheet name if needed)
    Set ws = ThisWorkbook.Sheets("ImageData")

    ' Loop through the data in the worksheet
    For i = 2 To ws.Cells(ws.Rows.Count, "A").End(xlUp).row
        If ws.Cells(i, 2).value = selectedValue Then
            ' Add matching items to the list box
            ListBoxImages.AddItem ws.Cells(i, 1).value
        End If
    Next i
End Sub
```

The code is a VBA event handler for a change event of a ComboBox control named ComboLocation. Let's break down what this code does:

Event Trigger: This code is executed when the value of the ComboLocation ComboBox is changed by the user. In other words, it runs when the user selects a different item from the ComboBox.

कोड कॉम्बोलोकेशन नामक कॉम्बोबॉक्स नियंत्रण के परिवर्तन इवेंट के लिए एक वीबीए इवेंट हैंडलर है। आइए देखें कि यह कोड क्या करता है:

इवेंट ट्रिगर: यह कोड तब निष्पादित होता है जब उपयोगकर्ता द्वारा कॉम्बोलोकेशन कॉम्बोबॉक्स का मान बदल दिया जाता है। दूसरे शब्दों में, यह तब चलता है जब उपयोगकर्ता कॉम्बोबॉक्स से एक अलग आइटम का चयन करता है।

Variable Declarations:

Dim selectedValue As String: Declares a variable selectedValue as a string. This variable will store the currently selected item in the ComboBox.

Dim ws As Worksheet: Declares a variable ws as a Worksheet object. This variable will be used to reference a worksheet in the Excel workbook.

Dim i As Long: Declares a variable i as a long integer. This variable will be used as a counter for a loop.

Clear List Box:

ListBoxImages.Clear: Clears the content of a ListBox control named ListBoxImages. This is done to remove any previous items in the list before populating it with new data.

Get Selected Item:

selectedValue = ComboLocation.Value: Retrieves the value (text) of the currently selected item in the ComboBox and stores it in the selectedValue variable.

Worksheet Reference:

Set ws = ThisWorkbook.Sheets("ImageData"): Sets the ws variable to refer to a worksheet named "ImageData" in the current workbook (ThisWorkbook).

Data Loop:

For i = 2 To ws.Cells(ws.Rows.Count, "A").End(xlUp).Row: Initiates a loop that goes through the data in the "ImageData" worksheet. It starts from row 2 and continues until the last non-empty cell in column A of the worksheet.

एक लूप आरंभ करता है जो "इमेजडेटा" वर्कशीट में डेटा से होकर गुजरता है। यह पंक्ति 2 से शुरू होता है और वर्कशीट के कॉलम ए में अंतिम गैर-रिक्त सेल तक जारी रहता है।

If ws.Cells(i, 2).Value = selectedValue Then: Checks if the value in column B of the current row (ws.Cells(i, 2).Value) matches the selectedValue (the value selected in the ComboBox).

जाँचता है कि क्या वर्तमान पंक्ति (ws.Cells(i, 2).Value) के कॉलम B में मान चयनितValue (कॉम्बोबॉक्स में चयनित मान) से मेल खाता है।

ListBoxImages.AddItem ws.Cells(i, 1).Value: If there's a match, it adds the value from column A of the same row to the ListBox control named ListBoxImages. This effectively populates the ListBox with items that match the selected value from the ComboBox.

In summary, this code listens for changes in the ComboLocation ComboBox and, when a change occurs, it clears the ListBox and populates it with items from a worksheet based on the selected value in the ComboBox. This is a common technique for dynamically updating a list or data display based on user input.

यदि कोई मेल है, तो यह उसी पंक्ति के कॉलम ए से मान को ListBoxImages नामक लिस्टबॉक्स नियंत्रण में जोड़ता है। यह प्रभावी रूप से लिस्टबॉक्स को उन आइटमों से भर देता है जो कॉम्बोबॉक्स से चयनित मान से मेल खाते हैं।

संक्षेप में, यह कोड कॉम्बोलोकेशन कॉम्बोबॉक्स में परिवर्तनों को सुनता है और, जब कोई परिवर्तन होता है, तो यह लिस्टबॉक्स को साफ़ करता है और कॉम्बोबॉक्स में चयनित मान के आधार पर इसे वर्कशीट से आइटम के साथ पॉप्युलेट करता है। यह उपयोगकर्ता इनपुट के आधार पर किसी सूची या डेटा डिस्प्ले को गतिशील रूप से अपडेट करने की एक सामान्य तकनीक है।

```
Private Sub ListBoxImages_Click()
    ' Get the selected item from the listbox
   Dim selectedItem As String
   selectedItem = ListBoxImages.value

   ' Construct the image file path based on the selected item
   Dim imagePath As String
   imagePath = "E:\MyPicture\" & selectedItem & ".jpg"
```

```
    ' Load and display the image in an Image control (assuming you have an Image control named Image1)
    If Len(Dir(imagePath)) > 0 Then
        Image1.Picture = LoadPicture(imagePath)
    Else
        ' Clear the Image control if the image file does not exist
        Image1.Picture = LoadPicture("") ' This will clear the image control
    End If

    Dim ws As Worksheet
    Dim i As Long

    ' Set your worksheet (change the sheet name if needed)
    Set ws = ThisWorkbook.Sheets("ImageData")

    ' Loop through the data in the worksheet
    For i = 2 To ws.Cells(ws.Rows.Count, "A").End(xlUp).row
        If ws.Cells(i, 1).value = selectedItem Then
            ' Add matching items to the list box
            Label5.Caption = "Category : " & ws.Cells(i, 3).value
            Label6.Caption = "Introduction : " & ws.Cells(i, 4).value
        End If
    Next i
End Sub
```

The code is a VBA event handler for the Click event of a ListBox control named ListBoxImages. Let's break down what this code does:

Event Trigger: This code is executed when the user clicks on an item in the ListBoxImages ListBox.

कोड ListBoxImages नामक ListBox नियंत्रण के क्लिक इवेंट के लिए एक VBA इवेंट हैंडलर है। आइए देखें कि यह कोड क्या करता है:

इवेंट ट्रिगर: यह कोड तब निष्पादित होता है जब उपयोगकर्ता ListBoxImages ListBox में किसी आइटम पर क्लिक करता है।

Variable Declarations:

Dim selectedItem As String: Declares a variable selectedItem as a string. This variable will store the text of the item selected in the ListBox.

Dim imagePath As String: Declares a variable imagePath as a string. This variable will store the file path of an image based on the selected item.

Dim ws As Worksheet: Declares a variable ws as a Worksheet object. This variable will be used to reference a worksheet in the Excel workbook.

Dim i As Long: Declares a variable i as a long integer. This variable will be used as a counter for a loop.

Get Selected Item:

selectedItem = ListBoxImages.Value: Retrieves the value (text) of the currently selected item in the ListBoxImages ListBox and stores it in the selectedItem variable.

Construct Image Path:

imagePath = "E:\MyPicture\" & selectedItem & ".jpg": Constructs the file path of an image based on the selected item. It assumes that images are stored in the "E:\MyPicture" directory with file names corresponding to the items in the ListBox.

Load and Display Image:

If Len(Dir(imagePath)) > 0 ***Then***: Checks if the image file exists at the constructed imagePath.

If the image file exists, it loads and displays the image in an Image control named Image1 using the LoadPicture method.

If the image file does not exist, it clears the Image control by loading an empty picture.

Worksheet Reference and Data Loop:

It sets the ws variable to refer to a worksheet named "ImageData" in the current workbook (ThisWorkbook).

It then initiates a loop that goes through the data in the "ImageData" worksheet, starting from row 2 and continuing until the last non-empty cell in column A of the worksheet.

Inside the loop, it checks if the value in column A of the current row (ws.Cells(i, 1).Value) matches the selectedItem (the value selected in the ListBox).

If there's a match, it updates the captions of two Label controls (Label5 and Label6) with information from columns C and D of the same row. This allows you to display additional information related to the selected item.

In summary, this code responds to a click event in the ListBoxImages ListBox. When an item is clicked, it updates an Image control (Image1) to display the corresponding image and updates Label controls (Label5 and Label6) to display additional information about the selected item from an Excel worksheet. This is a common technique for creating interactive interfaces in VBA user forms.

जांचता है कि छवि फ़ाइल निर्मित इमेजपाथ पर मौजूद है या नहीं।

यदि छवि फ़ाइल मौजूद है, तो यह लोडपिक्चर विधि का उपयोग करके Image1 नामक छवि नियंत्रण में छवि को लोड और प्रदर्शित करता है।

यदि छवि फ़ाइल मौजूद नहीं है, तो यह एक खाली चित्र लोड करके छवि नियंत्रण को साफ़ कर देती है।

वर्कशीट संदर्भ और डेटा लूप:

यह वर्तमान कार्यपुस्तिका (यह वर्कबुक) में "इमेजडेटा" नामक कार्यपत्रक को संदर्भित करने के लिए ws वैरिएबल सेट करता है।

इसके बाद यह एक लूप शुरू करता है जो "इमेजडेटा" वर्कशीट में डेटा के माध्यम से जाता है, पंक्ति 2 से शुरू होता है और वर्कशीट के कॉलम ए में अंतिम गैर-रिक्त सेल तक जारी रहता है।

लूप के अंदर, यह जाँचता है कि वर्तमान पंक्ति (ws.Cells(i, 1).Value) के कॉलम A में मान चयनित आइटम (लिस्टबॉक्स में चयनित मान) से मेल खाता है या नहीं।

यदि कोई मेल है, तो यह एक ही पंक्ति के कॉलम सी और डी की जानकारी के साथ दो लेबल नियंत्रणों (लेबल 5 और लेबल 6) के कैप्शन को अपडेट करता है। यह आपको चयनित आइटम से संबंधित अतिरिक्त जानकारी प्रदर्शित करने की अनुमति देता है।

संक्षेप में, यह कोड ListBoxImages ListBox में एक क्लिक इवेंट पर प्रतिक्रिया करता है। जब किसी आइटम पर क्लिक किया जाता है, तो यह संबंधित छवि प्रदर्शित करने के लिए एक छवि नियंत्रण (छवि 1) को अपडेट करता है और एक्सेल वर्कशीट से चयनित आइटम के बारे में अतिरिक्त जानकारी प्रदर्शित करने के लिए लेबल नियंत्रण (लेबल 5 और लेबल 6) को अपडेट करता है। यह VBA उपयोगकर्ता प्रपत्रों में इंटरैक्टिव इंटरफ़ेस बनाने की एक सामान्य तकनीक है।

```
Private Sub CommandButtonSearch_Click()
    Dim ws As Worksheet
    Set ws = ThisWorkbook.Sheets("ImageData")

    Dim keyword As String
    Dim location As String

    keyword = TextBoxKeyword.Text
    location = ComboLocation.value

    ' Clear the ListBox to show new search results
    ListBoxImages.Clear

    ' Loop through the data and add matching images to the ListBox
    Dim i As Long
    For i = 2 To ws.Cells(ws.Rows.Count, "A").End(xlUp).row
        If (location = "" Or ws.Cells(i, "B").value = location) And _
           (InStr(1, ws.Cells(i, "C").value, keyword, vbTextCompare) > 0) Then
            ListBoxImages.AddItem ws.Cells(i, "A").value
```

```
    End If
  Next i
End Sub
```

This code is an event handler for the Click event of a command button named CommandButtonSearch. It is responsible for performing a search and filtering operation based on user-provided criteria (a keyword and a location) and updating the contents of a ListBox named ListBoxImages. Let's break down how it works:

यह कोड CommandButtonSearch नामक कमांड बटन के क्लिक इवेंट के लिए एक इवेंट हैंडलर है। यह उपयोगकर्ता द्वारा प्रदत्त मानदंडों (एक कीवर्ड और एक स्थान) के आधार पर खोज और फ़िल्टरिंग ऑपरेशन करने और ListBoxImages नामक लिस्टबॉक्स की सामग्री को अपडेट करने के लिए जिम्मेदार है। आइए देखें कि यह कैसे काम करता है:

Event Trigger: This code is executed when the user clicks the CommandButtonSearch button.

Variable Declarations:

Dim ws As Worksheet: Declares a variable ws as a Worksheet object. This variable is used to reference a worksheet in the Excel workbook where the data is stored.

Dim keyword As String: Declares a variable keyword as a string. This variable will store the user-entered keyword for searching.

Dim location As String: Declares a variable location as a string. This variable will store the selected location from a ComboBox named ComboLocation.

Data Retrieval:

Set ws = ThisWorkbook.Sheets("ImageData"): Sets the ws variable to refer to a worksheet named "ImageData" in the current workbook (ThisWorkbook).

keyword = TextBoxKeyword.Text: Retrieves the text entered by the user into a TextBox named TextBoxKeyword and stores it in the keyword variable.

location = ComboLocation.Value: Retrieves the selected value from a ComboBox named ComboLocation and stores it in the location variable. This represents the location the user wants to filter by.

ListBox Clearing:

ListBoxImages.Clear: Clears the contents of the ListBoxImages ListBox to prepare it for the updated search results.

Data Loop:

The code initiates a loop that goes through the data in the "ImageData" worksheet, starting from row 2 and continuing until the last non-empty cell in column A of the worksheet.

Inside the loop, it checks each row for two conditions:

(location = "" Or ws.Cells(i, "B").Value = location): This condition checks if the selected location (location) matches the location in the current row (ws.Cells(i, "B").Value) or if no location is selected (location = ""). This condition ensures that either the selected location matches or all locations are included.

(InStr(1, ws.Cells(i, "C").Value, keyword, vbTextCompare) > 0): This condition checks if the keyword entered by the user (keyword) is found within the text in column C of the current row (ws.Cells(i, "C").Value) using a case-insensitive comparison. If the keyword is found, this condition is satisfied.

If both conditions are met for a particular row, it adds the value from column A of that row (ws.Cells(i, "A").Value) to the ListBoxImages ListBox. This means the image associated with this row matches the search criteria, so it's added to the list.

Result:

After the loop completes, the ListBoxImages ListBox is populated with the names of the images that match the search criteria.

In summary, this code performs a search and filter operation in response to a button click. It checks each row of data for a match with the user's specified criteria (keyword and location) and adds matching items to a ListBox for display. This allows the user to filter images based on specific criteria.

इवेंट ट्रिगर: यह कोड तब निष्पादित होता है जब उपयोगकर्ता CommandButtonSearch बटन पर क्लिक करता है।

परिवर्तनीय घोषणाएँ:

वर्कशीट के रूप में डिम ws: एक वेरिएबल ws को वर्कशीट ऑब्जेक्ट के रूप में घोषित करता है। इस वेरिएबल का उपयोग एक्सेल वर्कबुक में वर्कशीट को संदर्भित करने के लिए किया जाता है जहां डेटा संग्रहीत होता है।

स्ट्रिंग के रूप में मंद कीवर्ड: एक वैरिएबल कीवर्ड को एक स्ट्रिंग के रूप में घोषित करता है। यह वेरिएबल खोज के लिए उपयोगकर्ता द्वारा दर्ज किए गए कीवर्ड को संग्रहीत करेगा।

स्ट्रिंग के रूप में मंद स्थान: एक चर स्थान को एक स्ट्रिंग के रूप में घोषित करता है। यह वेरिएबल चयनित स्थान को कॉम्बोलोकेशन नामक कॉम्बोबॉक्स से संग्रहीत करेगा।

डेटा की पुनःप्राप्ति:

सेट ws = ThisWorkbook.Sheets("ImageData"): वर्तमान कार्यपुस्तिका (ThisWorkbook) में "ImageData" नामक वर्कशीट को संदर्भित करने के लिए ws वैरिएबल सेट करता है।

कीवर्ड = TextBoxKeyword.Text: उपयोगकर्ता द्वारा दर्ज किए गए टेक्स्ट को TextBoxKeyword नामक टेक्स्टबॉक्स में पुनर्प्राप्त करता है और इसे कीवर्ड वेरिएबल में संग्रहीत करता है।

स्थान = कॉम्बोलोकेशन.वैल्यू: कॉम्बोलोकेशन नामक कॉम्बोबॉक्स से चयनित मान प्राप्त करता है और इसे स्थान चर में संग्रहीत करता है। यह उस स्थान का प्रतिनिधित्व करता है जिसे उपयोगकर्ता फ़िल्टर करना चाहता है।

लिस्टबॉक्स समाशोधन:

ListBoxImages.Clear: अद्यतन खोज परिणामों के लिए इसे तैयार करने के लिए ListBoxImages ListBox की सामग्री को साफ़ करता है।

डेटा लूप:

कोड एक लूप शुरू करता है जो "इमेजडेटा" वर्कशीट में डेटा के माध्यम से जाता है, पंक्ति 2 से शुरू होता है और वर्कशीट के कॉलम ए में अंतिम गैर-रिक्त सेल तक जारी रहता है।

लूप के अंदर, यह प्रत्येक पंक्ति को दो स्थितियों के लिए जांचता है:

(स्थान = "" या ws.Cells(i, "B"). मान = स्थान): यह स्थिति जाँचती है कि क्या चयनित स्थान (स्थान) वर्तमान पंक्ति में स्थान से मेल खाता है (ws.Cells(i, "B") .मान) या यदि कोई स्थान चयनित नहीं है (स्थान = "")। यह शर्त सुनिश्चित करती है कि या तो चयनित स्थान मेल खाता है या सभी स्थान शामिल हैं।

(InStr(1, ws.Cells(i, "C").Value, कीवर्ड, vbTextCompare) > 0): यह स्थिति जांचती है कि उपयोगकर्ता द्वारा दर्ज किया गया कीवर्ड (कीवर्ड) वर्तमान के कॉलम C में टेक्स्ट के भीतर पाया जाता है या नहीं पंक्ति (ws.Cells(i, "C").Value) केस-असंवेदनशील तुलना का उपयोग करते हुए। यदि कीवर्ड मिल जाता है, तो यह शर्त पूरी हो जाती है।

यदि किसी विशेष पंक्ति के लिए दोनों शर्तें पूरी होती हैं, तो यह उस पंक्ति के कॉलम ए (ws.Cells(i, "A").Value) से मान को ListBoxImages ListBox में जोड़ता है। इसका मतलब है कि इस पंक्ति से जुड़ी छवि खोज मानदंड से मेल खाती है, इसलिए इसे सूची में जोड़ा गया है।

परिणाम:

लूप पूरा होने के बाद, ListBoxImages ListBox उन छवियों के नामों से भर जाता है जो खोज मानदंड से मेल खाते हैं।

संक्षेप में, यह कोड एक बटन क्लिक के जवाब में एक खोज और फ़िल्टर ऑपरेशन करता है। यह उपयोगकर्ता के निर्दिष्ट मानदंड (कीवर्ड और स्थान) के साथ मिलान के लिए डेटा की प्रत्येक पंक्ति की जांच करता है और प्रदर्शन के लिए मेल खाने वाले आइटम को लिस्टबॉक्स में जोड़ता है। यह उपयोगकर्ता को विशिष्ट मानदंडों के आधार पर छवियों को फ़िल्टर करने की अनुमति देता है।

```
Private Sub RefreshAll()

 ' Clear the ComboBox
    ComboLocation.Clear
    ' Clear the ListBox
    ListBoxImages.Clear
    ' Clear the TextBox
    TextBoxKeyword.Text = ""

    ' Reinitialize the ComboBox with data (assuming you have a UserForm_Initialize event)
    UserForm_Initialize
End Sub
```

This RefreshAll subroutine is designed to clear and reset the controls on your UserForm to their initial state. Let's break down what it does step by step:

ComboBox Clearing: ***ComboLocation.Clear***

This line clears the contents of a ComboBox control named ComboLocation. It removes all the items from the ComboBox.

ListBox Clearing: ***ListBoxImages.Clear***

This line clears the contents of a ListBox control named ListBoxImages. It removes all the items from the ListBox.

TextBox Clearing: ***TextBoxKeyword.Text = ""***

This line sets the text property of a TextBox control named TextBoxKeyword to an empty string (""). It effectively clears the text entered by the user in this TextBox.

Reinitialization: ***UserForm_Initialize***

This line calls the UserForm_Initialize procedure. This is likely an event handler for the initialization of your UserForm. It's used to set up initial values and configurations when the UserForm is first loaded.

By calling UserForm_Initialize after clearing the controls, you effectively reset your UserForm to its initial state as if it were just opened. This can be useful when you want to allow the user to start a new search or filtering operation from scratch.

In summary, the RefreshAll subroutine is used to clear the user inputs (ComboBox, ListBox, and TextBox) and reset the UserForm to its initial state for a fresh interaction or search.

यह RefreshAll सबरूटीन आपके UserForm पर नियंत्रणों को साफ़ करने और उनकी प्रारंभिक स्थिति में रीसेट करने के लिए डिज़ाइन किया गया है। आइए चरण दर चरण यह बताएं कि यह क्या करता है:

कॉम्बोबॉक्स क्लियरिंग: कॉम्बोलोकेशन.क्लियर

यह पंक्ति कॉम्बोलोकेशन नामक कॉम्बोबॉक्स नियंत्रण की सामग्री को साफ़ करती है। यह कॉम्बोबॉक्स से सभी आइटम हटा देता है।

लिस्टबॉक्स क्लियरिंग: ListBoxImages.Clear

यह पंक्ति ListBoxImages नामक ListBox नियंत्रण की सामग्री को साफ़ करती है। यह ListBox से सभी आइटम हटा देता है।

टेक्स्टबॉक्स क्लियरिंग: TextBoxKeyword.Text = ""

यह पंक्ति TextBoxKeyword नामक टेक्स्टबॉक्स नियंत्रण की टेक्स्ट प्रॉपर्टी को एक खाली स्ट्रिंग ("") पर सेट करती है। यह इस टेक्स्टबॉक्स में उपयोगकर्ता द्वारा दर्ज किए गए टेक्स्ट को प्रभावी ढंग से साफ़ करता है।

पुनर्प्रारंभीकरण: UserForm_Initialize

यह पंक्ति UserForm_Initialize प्रक्रिया को कॉल करती है। यह संभवतः आपके यूजरफॉर्म के आरंभीकरण के लिए एक इवेंट हैंडलर है। जब UserForm पहली बार लोड किया जाता है तो इसका उपयोग प्रारंभिक मान और कॉन्फ़िगरेशन सेट करने के लिए किया जाता है।

नियंत्रण साफ़ करने के बाद UserForm_Initialize को कॉल करके, आप प्रभावी रूप से अपने UserForm को उसकी प्रारंभिक स्थिति में रीसेट कर देते हैं जैसे कि इसे अभी खोला गया हो। यह तब उपयोगी हो सकता है जब आप उपयोगकर्ता को स्क्रैच से एक नई खोज या फ़िल्टरिंग ऑपरेशन शुरू करने की अनुमति देना चाहते हैं।

संक्षेप में, रिफ्रेशऑल सबरूटीन का उपयोग उपयोगकर्ता इनपुट (कॉम्बोबॉक्स, लिस्टबॉक्स और टेक्स्टबॉक्स) को साफ़ करने और नए इंटरैक्शन या खोज के लिए यूजरफॉर्म को उसकी प्रारंभिक स्थिति में रीसेट करने के लिए किया जाता है।

Private Sub CommandButton1_Click()

```
    RefreshAll
End Sub

Private Sub CommandButton2_Click()
    Unload Me
    MainMenuForm.Show
End Sub

Private Sub Image2_Click()
    Unload Me
End Sub
```

These are event handler procedures for different buttons and an image on your UserForm. Let's go through each one:

CommandButton1_Click()

This event handler is associated with a button named "CommandButton1." It's executed when the user clicks this button.

When this button is clicked, it calls the RefreshAll subroutine. As explained earlier, this subroutine clears and resets various controls on your UserForm to their initial state.

CommandButton2_Click()

This event handler is associated with a button named "CommandButton2." It's executed when the user clicks this button.

When this button is clicked, it does the following:

Unload Me: Unloads (closes) the current UserForm (the one containing these event handlers).

MainMenuForm.Show: Shows another UserForm named "MainMenuForm." This typically takes the user back to the main menu or another form in your application.

Image2_Click()

This event handler is associated with an image control named "Image2." It's executed when the user clicks on this image.

When this image is clicked, it does the following:

Unload Me: Unloads (closes) the current UserForm (the one containing these event handlers).

These event handlers provide functionality for the buttons and image on your UserForm. CommandButton1 resets the form, CommandButton2 navigates to another form, and Image2 closes the current form. These are common operations in user interface design to allow users to interact with your application.

ये आपके यूजरफॉर्म पर विभिन्न बटनों और एक छवि के लिए इवेंट हैंडलर प्रक्रियाएं हैं। आइए प्रत्येक के बारे में जानें:

CommandButton1_क्लिक करें()

यह ईवेंट हैंडलर "कमांडबटन1" नामक बटन से संबद्ध है। जब उपयोगकर्ता इस बटन पर क्लिक करता है तो यह निष्पादित हो जाता है।

जब इस बटन पर क्लिक किया जाता है, तो यह RefreshAll सबरूटीन को कॉल करता है। जैसा कि पहले बताया गया है, यह सबरूटीन आपके यूजरफॉर्म पर विभिन्न नियंत्रणों को साफ़ करता है और उनकी प्रारंभिक स्थिति में रीसेट करता है।

CommandButton2_Click()

यह ईवेंट हैंडलर "CommandButton2" नामक बटन से संबद्ध है। जब उपयोगकर्ता इस बटन पर क्लिक करता है तो यह निष्पादित हो जाता है।

जब इस बटन पर क्लिक किया जाता है, तो यह निम्न कार्य करता है:

मुझे अनलोड करें: वर्तमान यूजरफॉर्म (जिसमें ये ईवेंट हैंडलर शामिल हैं) को अनलोड (बंद) करता है।

MainMenuForm.Show: "MainMenuForm" नामक एक अन्य उपयोगकर्ता प्रपत्र दिखाता है। यह आमतौर पर उपयोगकर्ता को आपके एप्लिकेशन के मुख्य मेनू या किसी अन्य फॉर्म पर वापस ले जाता है।

Image2_क्लिक करें()

यह ईवेंट हैंडलर "इमेज2" नामक छवि नियंत्रण से संबद्ध है। जब उपयोगकर्ता इस छवि पर क्लिक करता है तो इसे निष्पादित किया जाता है।

जब इस छवि पर क्लिक किया जाता है, तो यह निम्न कार्य करता है:

मुझे अनलोड करें: वर्तमान यूजरफॉर्म (जिसमें ये ईवेंट हैंडलर शामिल हैं) को अनलोड (बंद) करता है।

ये इवेंट हैंडलर आपके यूजरफॉर्म पर बटन और छवि के लिए कार्यक्षमता प्रदान करते हैं। CommandButton1 फॉर्म को रीसेट करता है, CommandButton2 दूसरे फॉर्म पर नेविगेट करता है, और Image2 वर्तमान फॉर्म को बंद कर देता है। उपयोगकर्ताओं को आपके एप्लिकेशन के साथ इंटरैक्ट करने की अनुमति देने के लिए उपयोगकर्ता इंटरफ़ेस डिज़ाइन में ये सामान्य ऑपरेशन हैं।

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 Google Pay

9748327614