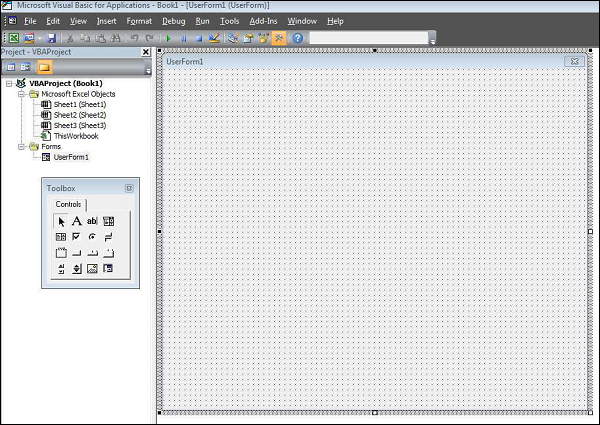
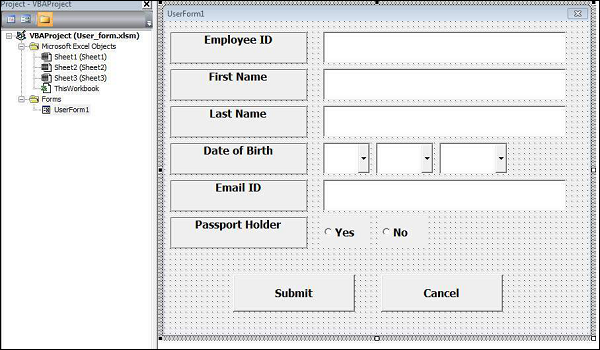
1. What are Userforms? Why are they used? How to fill a list box using for loop.

A **User Form** is a custom-built dialog box that makes a user data entry more controllable and easier to use for the user. In this chapter, you will learn to design a simple form and add data into excel.

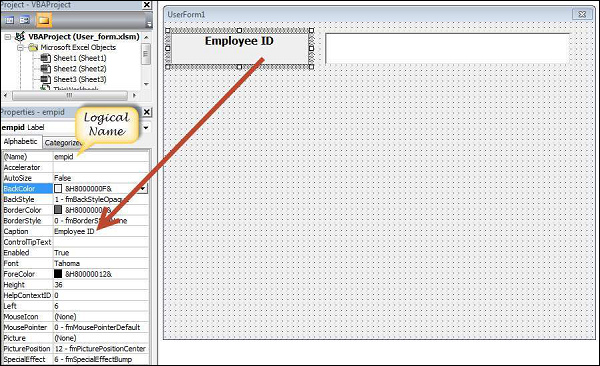
**Step 1** − Navigate to VBA Window by pressing Alt+F11 and Navigate to "Insert" Menu and select "User Form". Upon selecting, the user form is displayed as shown in the following screenshot.



**Step 2** − Design the forms using the given controls.



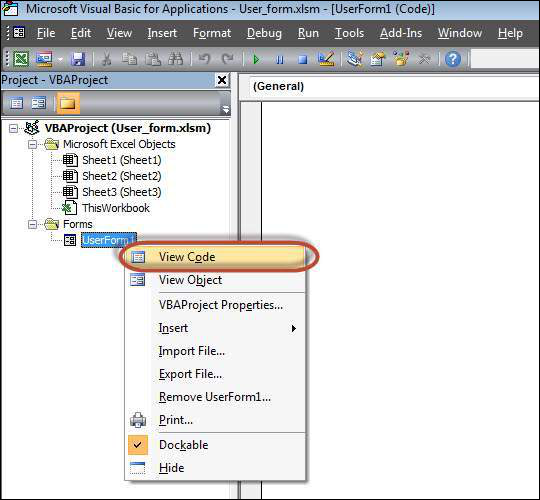
**Step 3** − After adding each control, the controls have to be named. Caption corresponds to what appears on the form and name corresponds to the logical name that will be appearing when you write VBA code for that element.



**Step 4** − Following are the names against each one of the added controls.

|  |  |  |
| --- | --- | --- |
| **Control** | **Logical Name** | **Caption** |
| From | frmempform | Employee Form |
| Employee ID Label Box | empid | Employee ID |
| firstname Label Box | firstname | First Name |
| lastname Label Box | lastname | Last Name |
| dob Label Box | dob | Date of Birth |
| mailid Label Box | mailid | Email ID |
| Passportholder Label Box | Passportholder | Passport Holder |
| Emp ID Text Box | txtempid | NOT Applicable |
| First Name Text Box | txtfirstname | NOT Applicable |
| Last Name Text Box | txtlastname | NOT Applicable |
| Email ID Text Box | txtemailid | NOT Applicable |
| Date Combo Box | cmbdate | NOT Applicable |
| Month Combo Box | cmbmonth | NOT Applicable |
| Year Combo Box | cmbyear | NOT Applicable |
| Yes Radio Button | radioyes | Yes |
| No Radio Button | radiono | No |
| Submit Button | btnsubmit | Submit |
| Cancel Button | btncancel | Cancel |

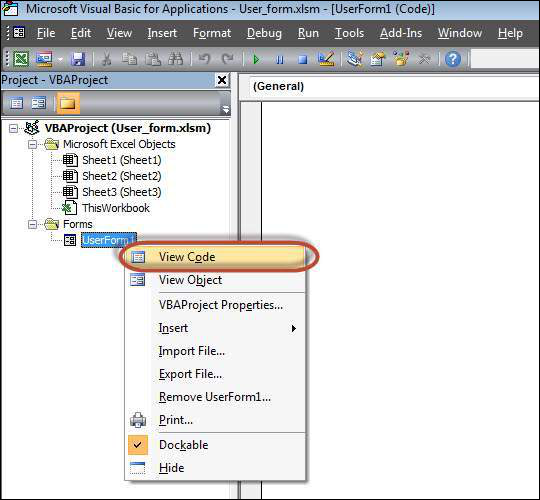
**Step 5** − Add the code for the form load event by performing a right-click on the form and selecting 'View Code'.



**Step 4** − Following are the names against each one of the added controls.

|  |  |  |
| --- | --- | --- |
| **Control** | **Logical Name** | **Caption** |
| From | frmempform | Employee Form |
| Employee ID Label Box | empid | Employee ID |
| firstname Label Box | firstname | First Name |
| lastname Label Box | lastname | Last Name |
| dob Label Box | dob | Date of Birth |
| mailid Label Box | mailid | Email ID |
| Passportholder Label Box | Passportholder | Passport Holder |
| Emp ID Text Box | txtempid | NOT Applicable |
| First Name Text Box | txtfirstname | NOT Applicable |
| Last Name Text Box | txtlastname | NOT Applicable |
| Email ID Text Box | txtemailid | NOT Applicable |
| Date Combo Box | cmbdate | NOT Applicable |
| Month Combo Box | cmbmonth | NOT Applicable |
| Year Combo Box | cmbyear | NOT Applicable |
| Yes Radio Button | radioyes | Yes |
| No Radio Button | radiono | No |
| Submit Button | btnsubmit | Submit |
| Cancel Button | btncancel | Cancel |

**Step 5** − Add the code for the form load event by performing a right-click on the form and selecting 'View Code'.



'Empty all other text box fields

txtfirstname.Value = ""

txtlastname.Value = ""

txtemailid.Value = ""

'Clear All Date of Birth Related Fields

cmbdate.Clear

cmbmonth.Clear

cmbyear.Clear

'Fill Date Drop Down box - Takes 1 to 31

With cmbdate

.AddItem "1"

.AddItem "2"

.AddItem "3"

.AddItem "4"

.AddItem "5"

.AddItem "6"

.AddItem "7"

.AddItem "8"

.AddItem "9"

.AddItem "10"

.AddItem "11"

.AddItem "12"

.AddItem "13"

.AddItem "14"

.AddItem "15"

.AddItem "16"

.AddItem "17"

.AddItem "18"

.AddItem "19"

.AddItem "20"

.AddItem "21"

.AddItem "22"

.AddItem "23"

.AddItem "24"

.AddItem "25"

.AddItem "26"

.AddItem "27"

.AddItem "28"

.AddItem "29"

.AddItem "30"

.AddItem "31"

End With

'Fill Month Drop Down box - Takes Jan to Dec

With cmbmonth

.AddItem "JAN"

.AddItem "FEB"

.AddItem "MAR"

.AddItem "APR"

.AddItem "MAY"

.AddItem "JUN"

.AddItem "JUL"

.AddItem "AUG"

.AddItem "SEP"

.AddItem "OCT"

.AddItem "NOV"

.AddItem "DEC"

End With

'Fill Year Drop Down box - Takes 1980 to 2014

With cmbyear

.AddItem "1980"

.AddItem "1981"

.AddItem "1982"

.AddItem "1983"

.AddItem "1984"

.AddItem "1985"

.AddItem "1986"

.AddItem "1987"

.AddItem "1988"

.AddItem "1989"

.AddItem "1990"

.AddItem "1991"

.AddItem "1992"

.AddItem "1993"

.AddItem "1994"

.AddItem "1995"

.AddItem "1996"

.AddItem "1997"

.AddItem "1998"

.AddItem "1999"

.AddItem "2000"

.AddItem "2001"

.AddItem "2002"

.AddItem "2003"

.AddItem "2004"

.AddItem "2005"

.AddItem "2006"

.AddItem "2007"

.AddItem "2008"

.AddItem "2009"

.AddItem "2010"

.AddItem "2011"

.AddItem "2012"

.AddItem "2013"

.AddItem "2014"

End With

'Reset Radio Button. Set it to False when form loads.

radioyes.Value = False

radiono.Value = False

End Sub

**Step 8** − Now add the code to the Submit button. Upon clicking the submit button, the user should be able to add the values into the worksheet.

Private Sub btnsubmit\_Click()

Dim emptyRow As Long

'Make Sheet1 active

Sheet1.Activate

'Determine emptyRow

emptyRow = WorksheetFunction.CountA(Range("A:A")) + 1

'Transfer information

Cells(emptyRow, 1).Value = txtempid.Value

Cells(emptyRow, 2).Value = txtfirstname.Value

Cells(emptyRow, 3).Value = txtlastname.Value

Cells(emptyRow, 4).Value = cmbdate.Value & "/" & cmbmonth.Value & "/" & cmbyear.Value

Cells(emptyRow, 5).Value = txtemailid.Value

If radioyes.Value = True Then

Cells(emptyRow, 6).Value = "Yes"

Else

Cells(emptyRow, 6).Value = "No"

End If

End Sub

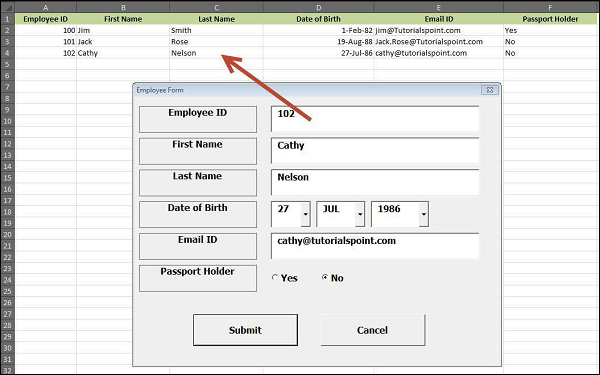
**Step 9** − Add a method to close the form when the user clicks the Cancel button.

Private Sub btncancel\_Click()

Unload Me

End Sub

**Step 10** − Execute the form by clicking the "Run" button. Enter the values into the form and click the 'Submit' button. Automatically the values will flow into the worksheet as shown in the following screenshot.



2.What is an array? Write a VBA code to enter students and their marks from the below table.

This post provides an**in-depth look at the VBA array**which is a very important part of the Excel VBA programming language. It covers everything you need to know about the VBA array.

We will start by seeing what exactly is the VBA Array is and why you need it.

Below you will see a quick reference guide to **using the VBA Array**.  Refer to it anytime you need a quick reminder of the VBA Array syntax.

The rest of the post provides the most complete guide you will find on the VBA array.

| **Task** | **Static Array** | **Dynamic Array** |
| --- | --- | --- |
| **Declare** | **Dim** arr(0 **To**5) **As Long** | **Dim** arr() **As Long** **Dim** arr **As Variant** |
| **Set Size** | See Declare above | **ReDim** arr(0 **To**5)**As Variant** |
| **Get Size(number of items)** | See [ArraySize](https://excelmacromastery.com/excel-vba-array/" \l "Get_the_Array_Size) function below. | See [ArraySize](https://excelmacromastery.com/excel-vba-array/" \l "Get_the_Array_Size) function below. |
| **Increase size (keep existing data)** | Dynamic Only | **ReDim** **Preserve** arr(0 **To** 6) |
| **Set values** | arr(1) = 22 | arr(1) = 22 |
| **Receive values** | total = arr(1) | total = arr(1) |
| **First position** | **LBound**(arr) | **LBound**(arr) |
| **Last position** | **Ubound**(arr) | **Ubound**(arr) |
| **Read all items(1D)** | **For** i = **LBound**(arr)**To UBound**(arr) **Next**i Or **For** i = **LBound**(arr,1)**To UBound**(arr,1) **Next**i | **For** i = **LBound**(arr)**To UBound**(arr) **Next**i Or **For** i = **LBound**(arr,1)**To UBound**(arr,1) **Next**i |
| **Read all items(2D)** | **For** i = **LBound**(arr,1)**To UBound**(arr,1)   **For** j = **LBound**(arr,2)**To UBound**(arr,2)   **Next**j **Next**i | **For** i = **LBound**(arr,1)**To UBound**(arr,1)   **For** j = **LBound**(arr,2)**To UBound**(arr,2)   **Next**j **Next**i |
| **Read all items** | **Dim** item **As Variant** **For Each** item **In** arr **Next** item | **Dim** item **As Variant** **For Each** item **In** arr **Next** item |
| **Pass to Sub** | **Sub** MySub(**ByRef** arr()**As String**) | **Sub** MySub(**ByRef** arr()**As String**) |
| **Return from Function** | **Function** GetArray() **As Long**()     **Dim** arr(0**To**5)**As Long**     GetArray = arr **End Function** | **Function** GetArray() **As Long**()     **Dim** arr() **As Long**     GetArray = arr **End Function** |
| **Receive from Function** | Dynamic only | **Dim** arr() **As Long** Arr = GetArray() |
| **Erase array** | **Erase** arr \*Resets all values to default | **Erase** arr \*Deletes array |
| **String to array** | Dynamic only | **Dim** arr **As Variant** arr = Split("James:Earl:Jones",":") |
| **Array to string** | **Dim** sName **As String** sName = Join(arr, ":") | **Dim** sName **As String** sName = Join(arr, ":") |
| **Fill with values** | Dynamic only | **Dim** arr **As Variant** arr = Array("John", "Hazel", "Fred") |
| **Range to Array** | Dynamic only | **Dim** arr **As Variant** arr = Range("A1:D2") |
| **Array to Range** | Same as dynamic | **Dim** arr **As Variant** Range("A5:D6") = arr |

3. Write step by step procedure to protect your workbook using a password.

To prevent other users from viewing hidden worksheets, adding, moving, deleting, or hiding worksheets, and renaming worksheets, you can protect the structure of your Excel workbook with a password.

**Note:**The screen shots in this article were taken in Excel 2016. If you have a different version your view might be slightly different, but unless otherwise noted, the functionality is the same.

Example of Workbook structure

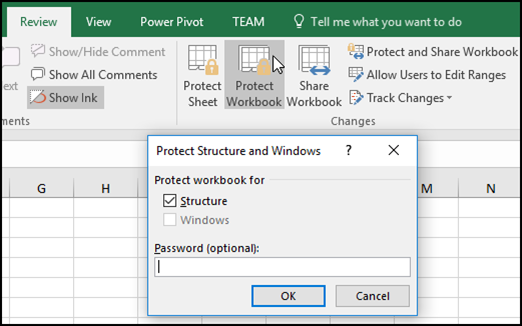
**Notes:**Protecting the workbook is not the same as protecting an Excel file or a worksheet with a password. See below for more information:

* To lock your file so that other users can't open it, see [Protect an Excel file](https://support.microsoft.com/en-us/office/protect-an-excel-file-7359d4ae-7213-4ac2-b058-f75e9311b599).
* To protect certain areas of the data in your worksheet from other users, you have to protect your worksheet. For more information, see [Protect a worksheet](https://support.microsoft.com/en-us/office/protect-a-worksheet-3179efdb-1285-4d49-a9c3-f4ca36276de6).
* To know the difference between protecting your Excel file, workbook, or a worksheet, see [Protection and security in Excel](https://support.microsoft.com/en-us/office/protection-and-security-in-excel-be0b34db-8cb6-44dd-a673-0b3e3475ac2d).

Protect the workbook structure

To protect the structure of your workbook, follow these steps:

1. Click **Review** > **Protect Workbook**.



**Note:**The **Windows** option is available only in Excel 2007, Excel 2010, Excel for Mac 2011, and Excel 2016 for Mac. Select the **Windows** option if you want to prevent users from moving, resizing, or closing the workbook window, or hide/unhide windows.

1. Enter a password in the **Password** box.

**Important:**The password is optional. If you do not supply a password, any user can unprotect and change the workbook. If you do enter a password, make sure that you choose a password that is easy to remember. Write your passwords down and store them someplace safe. If you lose them, Excel cannot recover them for you.

1. Select **OK**, re-enter the password to confirm it, and then select **OK** again.