Central de Help em Excel / VBA
http://www.bestexcel.eng.br

Excel_Vba_Aula13

User Forms

O que é uma User Form e para que servem? Essencialmente as User Forms nascem da necessidade que tem o usuário de interagir com as procedures de maneira a mais amigável possivel. Podemos usa-las para nos devolver informações ou, o que é mais comum, usa-las para entrada de dados. Até aqui, e nos inumeros exemplos e exercicios apresentados, lançamos mão das funções **MsgBox** e **InputBox**, funções essas que, de certa maneira, se impoem como preciosos recursos do VBA, no tocante a interação entre o meio exterior e o ambiente de programação. Contudo, como se verá, as User Forms nos conduz a uma nova dimensão, fazendo com que nossas procedures incorporem todas as facilidades a que estamos acostumados no ambiente Windows.

Como criar uma User Form ?

Para criar uma User Form é necessário que nos mudemos de ambiente acessando o (**V**)isual (**B**)asic (**E**)ditor. Para faze-lo, estando no Excel, basta
(1)manter calcada a tecla Alt enquanto aciona a tecla (F11).
(2)Outra maneira de acessar o VBE é seguir os passos
<Tools; <Macro; <Visual Basic Editor
Estando agora no VBE, acionamos na barra de ferramentas Standard a opção **Project Explorer** que nos permitirá navegar pelas planilhas, modulos, user forms e outros elementos do nosso projeto que, a titulo de exemplo, apelidamos **BookX.xls**. Portanto os passos seguintes no VBE serão :
<Project Explorer; ><Insert User Form
Ou, o que é mais seguro :
<Project Explorer; <VBA Project(BookX.xls); ><Insert; <User Form
O proximo passo será introduzir na User Form recem criada, Labels, TextBoxes, ListBoxes, CheckBoxes, CommandButtons, e muitos outros elementos que chamaremos **ActiveX controls**. Cada um desses controles reagem de forma peculiar respondendo a uma variedade de eventos.

Introduzindo controles ActiveX na User Form

Uma User Form não teria utilidade se não fossem os controles **ActiveX** que podemos lhe acrescentar. Vejamos então como se acrescenta uma **TextBox** numa User Form. Qualquer outro controle a ser acrescentado seguirá o mesmo caminho. Primeiramente vamos tornar disponiveis os controles ActiveX :
(1)no VBE, clicando na opção ToolBox, da barra de ferramentas Standard, ou
(2)na barra de ferramentas Standard acionando <View; <Toolbox

Os vários controles **ActiveX** após esses passos, estarão disponiveis. Ao passearmos o mouse pelos icones da Toolbox, automaticamente, o **help** informará ao usuário do que se trata. Se por exemplo queremos inserir uma TextBox na User Form basta acompanhar uma das duas sequencias :
(1)Dois cliques : um clique no ícone TextBox da barra de ferramentas ActiveX e outro na User Form.
(2)Clicar no icone correspondente à TextBox com o (**L**)eft (**M**)ouse (**B**)utton. Mantendo o LMB calcado arraste o icone selecionado para a User Form e libere o LMB.

Redimensionando e reposicionando o Objeto

Qualquer controle recem instalado na User Form estará primeiramente no estado de edição. Esse estado peculiar está caracterizado pelas 8 alças de arraste no contorno do objeto. Com o controle em edição podemos (1) Redimensiona-lo (em termos de comprimento e altura) usando qualquer alça de arraste (2) Calcando o LMB no contorno do objeto, mas fora das alças de arraste, conseguimos reposiciona-lo na User Form onde melhor nos convem. O leitor deve observar que o mouse se modifica quando o posicionamos nas alças de arraste ou fora delas.
Para fazer com que o objeto saia do estado de edição, basta clicarmos com o LMB na User Form. Se queremos que o objeto retorne ao estado de edição, clicamos com o LMB no controle ActiveX

UserForm1

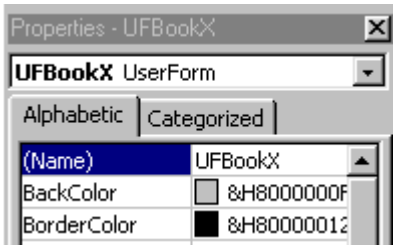
Ao introduzirmos uma UserForm no nosso projeto, o VBA provisóriamente providencia um nome para esse objeto: UserForm1. Se esse nome não nos satisfaz, devemos substitui-lo por outro mais adequado. Por exemplo UFBookX, atrelando o nome do arquivo ativo à User Form. Que providencias devemos tomar para alterar o nome User Form1 para UFBookX ? Chamamos a atenção para o fato de que (1)o nome do controle é uma **propriedade** do mesmo. Portanto, alterar o nome atribuido inicialmente pelo VBA, significará alterarmos a propriedade correspondente. (2)Ao introduzirmos no nosso projeto uma UserForm, as propriedades **Name** (nome segundo o qual o VBA saberá identificar o objeto User Form) e a propriedade **Caption** se confundem (ambas tem a denominação UserForm1)

Como exibir a lista de propriedades do objeto User Form?

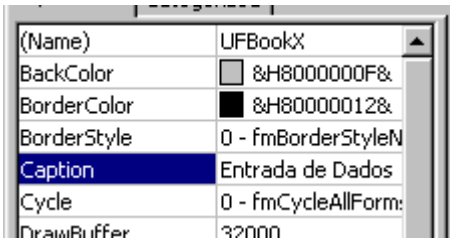
(1) Clicamos na UserForm; ><Properties;
e a lista de propriedades da User Form se apresenta. Onde está indicado (Name) trocamos para **UFBookX** e onde está indicado Caption, trocamos para **Entrada de Dados**

Podemos indicar essas ações escrevendo :
><Properties; (Name) : UFBookX; <Caption : Entrada de Dados
(2)Podemos tambem exibir a lista de propriedades da User Form clicando na opção **Properties Window** da barra de ferramentas Standard :
<Properties Window
As ilustrações abaixo mostram as modificações introduzidas nas propriedades Name e Caption

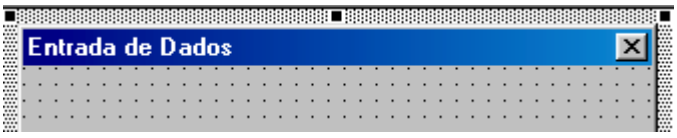
Alterando a propriedade Name na lista de propriedades da UserForm



Alterando a propriedade Caption na lista de propriedades da UserForm



Resposta à modificação introduzida na Propriedade Caption



Montando uma UserForm para interagir com o Usuário.

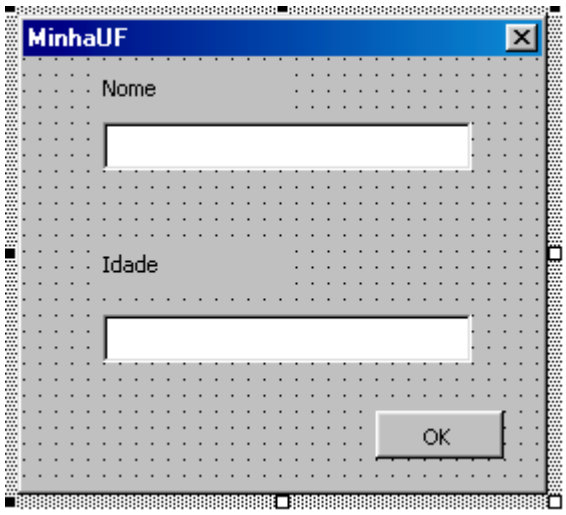
Coloquemos a seguinte questão : queremos montar um novo esquema que (1)peça ao usuário o nome e a idade de uma mesma pessoa e (2)que transfira esses dados para a Sheet1. Para atender a essa questão, será necessário preparar uma User Form com os controles ActiveX adequados. O quadro abaixo mostra com detalhe quais são esses elementos.

Quadro de Controles ActiveX inseridos na UserForm

Controle ActiveX	Name	Caption	ControlSource
TextBox	txNome	-------	sheet1!b2
TextBox	txIdade	-------	sheet1!c2
Label	LbNome	Nome	-------
Label	LbIdade	Idade	-------
CommandButton	cbOK	OK	-------

A propriedade **ControlSource** das duas textboxes formam um link de mão dupla entre os conteudos das textboxes e das celulas b2 e c2 da Sheet1. Com isso, qualquer alteração no conteudo dessas celulas afetará o conteudo das textboxes e vice versa. Nem sempre linkar os controles da User Form às celulas é conveniente. Ganhamos mais independencia e flexibilidade se deixamos essa questão por conta do nosso codigo.

Aspecto da User Form após a inserção dos controles ActiveX

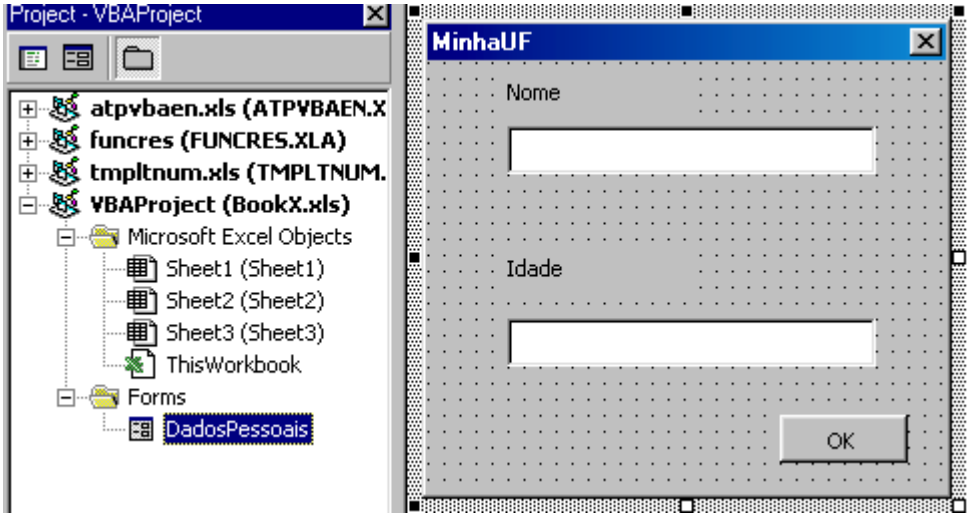


No objeto UserForm, alteramos: (1)a propriedade Name de UserForm1 para **DadosPessoais** (2)a propriedade Caption de UserForm1 para **MinhaUF** conforme mostra a ilustração. Alteramos ainda (3)a propriedade Name do controle CommandButton de CommandButton1 para **cbOK** e (4)a propriedade Caption desse controle de CommandButton1 para **OK** (vide "Quadro de Controles ActiveX inseridos na User Form" e o Quadro Resumo abaixo)

User Form - Quadro Resumo

	(1)Propriedade Name	(1)Propriedade Caption	(2)Propriedade Name	(2)Propriedade Caption
User Form	**UserForm1**	**UserForm1**	**DadosPessoais**	**MinhaUF**
CommandButton	**CommandButton1**	**CommandButton1**	**cbOK**	**OK**
(1) Valores das propriedades atribuidas pelo VBA				
(2 Valores das propriedades modificadas pelo Usuário				

A ilustração abaixo nos mostra um aspecto do ambiente VBE



Atingimos um estágio no qual estamos com nossa User Form montada com os controles ActiveX que nos interessam no momento. Seria comodo para o usuário, para chamar a User Form que acabamos de criar, ter adicionalmente, no ambiente Excel, um CommandButton inserido em Sheet1.

Para tanto, a titulo de recordação apresentamos a sequencia de ações a serem seguidas para criar em Sheet1 esse novo objeto :

<Microsoft Excel; <View; <Toolbars; <Control Toolbox; <Na barra de ferramentas Control Toolbox, clique no icone **CommandButton**; <Clique na planilha mantendo o LMB calcado. Trace um retangulo (cujo comprimento deve ser maior que a altura); </ (vide aula01) libere o LMB. Surgirá então o objeto CommandButton1 circundado por 8 alças de arraste indicativas de que o mesmo está no modo de edição. O leitor deve recordar-se que “modo de edição” e “Design Mode” são na realidade a mesma coisa. Estando o objeto em Design Mode, podemos desenvolver a “event procedure”. Event Procedures em geral, tratam do seguinte problema : Qual deve ser a reação, ou o que deve ocorrer quando o Usuário clicar em determinado objeto, ou tomar essa ou aquela atitude? (1)Clicando-se na planilha, fora portanto do objeto CommandButton1 recem criado, encerramos o processo de edição. (2)Clicando-se no objeto

CommandButton1, o mesmo retorna ao modo de edição (Design Mode). (3)É conveniente estando CommandButton1 em edição, alterarmos a propriedade Caption para algo mais amigavel, por exemplo : **Exibir MinhaUF**. Para faze-lo bastam as ações : >CommandButton1; <Properties ; <Caption :**Exibir MinhaUF**. (4)Alteramos tambem a propriedade Name de CommandButton1 para **cbBookX** (5)Estando esse objeto no modo de edição (Design Mode) , um clique duplo no mesmo vai nos transportar do ambiente Excel para o VBE onde já nos esperam as duas instruções abaixo :

```
Private Sub cbBookX_Click()
End Sub
```

Eventos

Como será que o VBA interpreta o fato de que o usuário clicou no objeto CommandButton para obter determinada resposta? Esse fato, assim como muitos outros apresentados ao longo desse curso, é interpretado pelo VBA como sendo um evento. Em resumo, o fato de clicarmos num objeto com determinada finalidade constitui-se num evento.

Consequentemente as duas instruções anteriores devem responder pelo seguinte evento: Clicando-se no objeto commandbutton cuja propriedade Name é : **cbBookX,** queremos trazer para a tela a User Form **DadosPessoais.** Para isso inserimos a instrução **DadosPessoais.Show**

```
Private Sub CommandButton1_Click()
   DadosPessoais.Show
End Sub
```

A titulo de teste, tendo encerrado o modo de edição, clique no commandbutton **cbBookX** (inserido na Sheet1), e a UserForm **DadosPesoais** que acabamos de criar será então exibida na tela. Contudo ainda é necessário tomarmos algumas providencias. Por exemplo, o CommandButton OK da User Form ainda não está funcionando.

Ativando o controle OK da UserForm

No ambiente VBE e na User Form DadosPessoais, dê um clique duplo no controle OK. Como anteriormente, o VBA lhe prepara as duas instruções abaixo:

```
Private Sub cbOK_Click()
End Sub
```

Caberá então ao codificador inserir a instrução **DadosPessoais.Hide** conforme fizemos abaixo. Essa instrução fará com que a UserForm **DadosPessoais** seja removida da tela

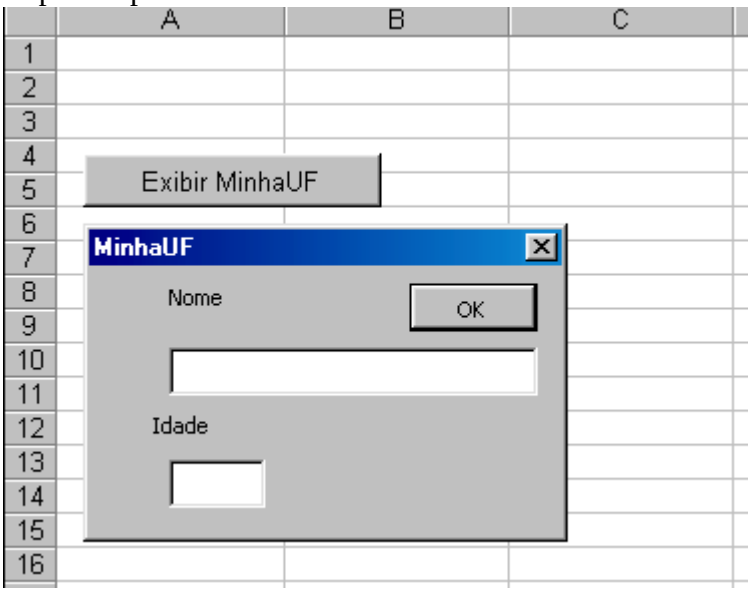
```
Private Sub cbOK_Click()
   DadosPessoais.Hide
End Sub
```

Temos até o momento duas procedures. (1)Aquela que chama a UserForm DadosPessoais para a tela e (2)aquela que faz com que nos descartemos da UserForm DadosPessoais. É importante que o leitor perceba o seguinte fato : a User Form vem para a tela, e aguarda uma atitude qualquer por parte do usuário. Por exemplo, preencher as duas textboxes e clicar no commandbutton cbOK.

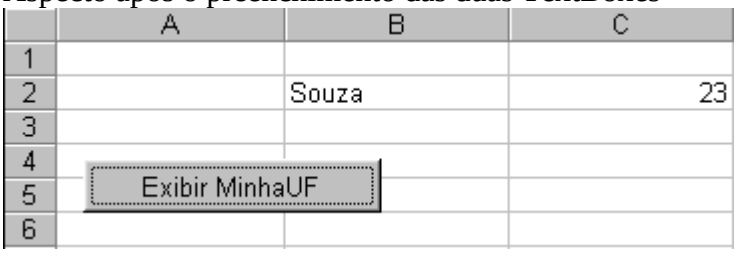
Vamos introduzir uma alteração na procedure cbBookX_click( ). Queremos que a User Form DadosPessoais ao se apresentar esteja com as duas textboxes vazias para permitir a entrada de novos dados.

```
Private Sub cbBookX_Click()          ' propriedade Caption : Exibir MinhaUF
  DadosPessoais.txNome.Value = ""
  DadosPessoais.txIdade.Value = ""
  DadosPessoais.Show
End Sub

Private Sub cbOK_Click()             'propriedade Caption: OK
   DadosPessoais.Hide
End Sub
```

Aspecto após clicarmos no commandbutton Exibir MinhaUF



Aspecto após o preenchimento das duas TextBoxes



Usando Prefixos na identificação dos controles ActiveX

O leitor já deve ter percebido que estamos usando prefixos para os controles ActiveX de tal forma que seja possivel identifica-los rapidamente. Por exemplo **tx**Nome; **tx**Idade(TextBoxes); **cb**OK(CommandButtons); **lb**Idade(Label) etc...

Usando o commandbutton Cancel

Vamos montar uma nova User Form com as seguintes facilidades: Um commandbutton **OK**; um commandbutton **Cancel**; uma TextBox para entrada de textos(dados), um Label para orientação do usuário. Na Sheet2 vamos criar um commandbutton para chamar essa User Form.

Antes de nos lançarmos a montagem dessa User Form vamos nos decidir sobre as novos valores que as propriedades devem assumir.

	Propriedade Name	Propriedade Caption	Propriedade Default	Propriedade Cancel
CommandButton1 (Sheet2)	cbIntroducao	Exibe Introducao	----------	----------
UserForm1	UFIntroducao	Introducao	----------	----------
Controles ActiveX	----------	----------	----------	----------
ComandButton1	cbOK	OK	TRUE	FALSE
ComandButton2	cbCancel	Cancel	FALSE	TRUE
TextBox1	txNome		----------	----------
Label1	LbNome	Seu Nome ?	----------	----------

Criada a UserForm com os controles ActiveX seguindo mesmos passos estudados anteriormente, e tendo introduzido em Sheet2 o commandbutton cbIntroducao, apresentamos a seguir as duas event procedures relativas aos controles cbOK e cbCancel. O leitor deve observar que (1)no quadro acima, deixamos de informar o valor da propriedade ControlSource da textbox txNome (2)que criamos uma variavel booleana que denominamos **cancelo** cujo escopo é Public.

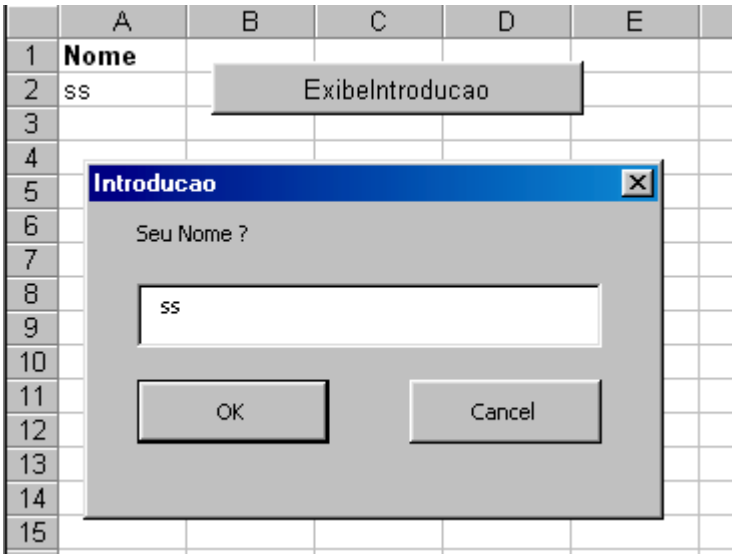
```
Public cancelo As Boolean          'criando a variavel booleana cancelo
```

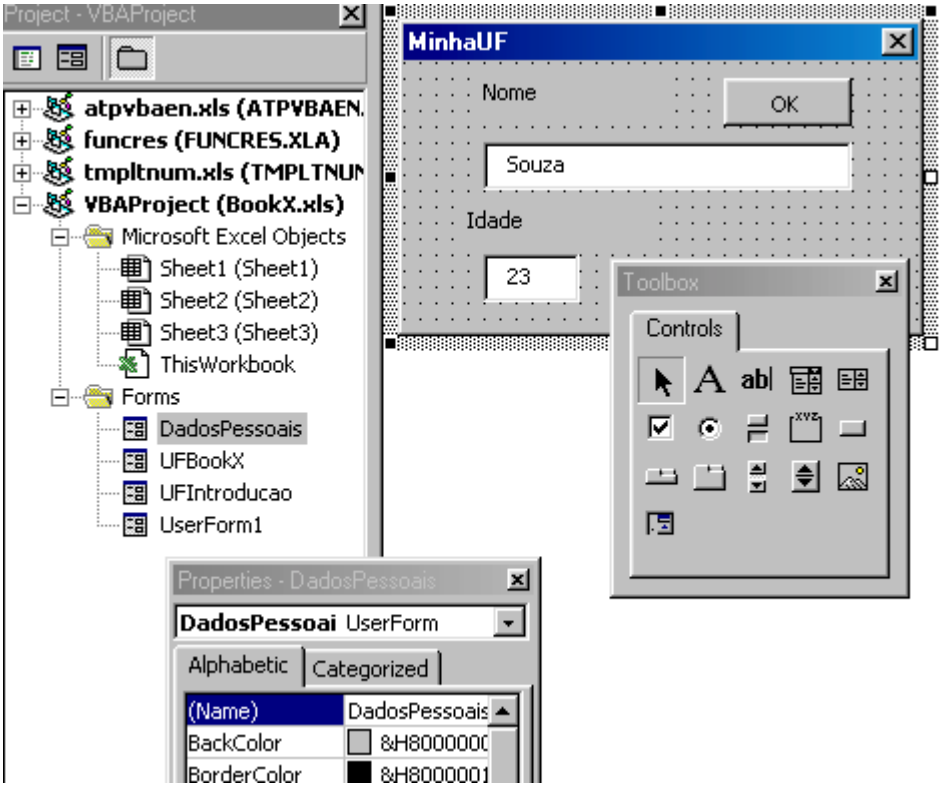
```
Private Sub CbCancel_Click()
    txNome.Value = ""
    cancelo = True
   UFIntroducao.Hide
End Sub

Private Sub CbOK_Click()
   cancelo = False
   Sheets("Sheet2").Range("a2").Value = txNome.Value
  UFIntroducao.Hide
End Sub
```

A event procedure correspondente ao commandbutton cbIntroducao responsavel pela chamada da UserForm será :

```
Private Sub cbIntroducao_Click()
      UFIntroducao.Show
End Sub
```

Operação:
(1)Clicamos no commandbutton cbIntroducao (caption: ExibeIntroducao), (2) a User Form UFIntroducao (caption: Introducao) é chamada para a planilha ativa (Sheet2), (3)propositalmente, não providenciamos que a textbox txNome viesse vazia, consequentemente ao ser carregada na tela o referido controle se apresenta com dados. (4)se teclamos no objeto cbCancel (caption Cancel) a propriedade Value da textbox txNome assumirá o valor Empty (vide procedure cbCancel_Click( )).(5)se teclamos no commandbutton cbOK os dados da txNome são transferidos para a planilha e a User Form é descartada da tela. (6)se teclamos no commandbutton cbCancel a User Form é descartada sem afetar o conteudo da celula a2.

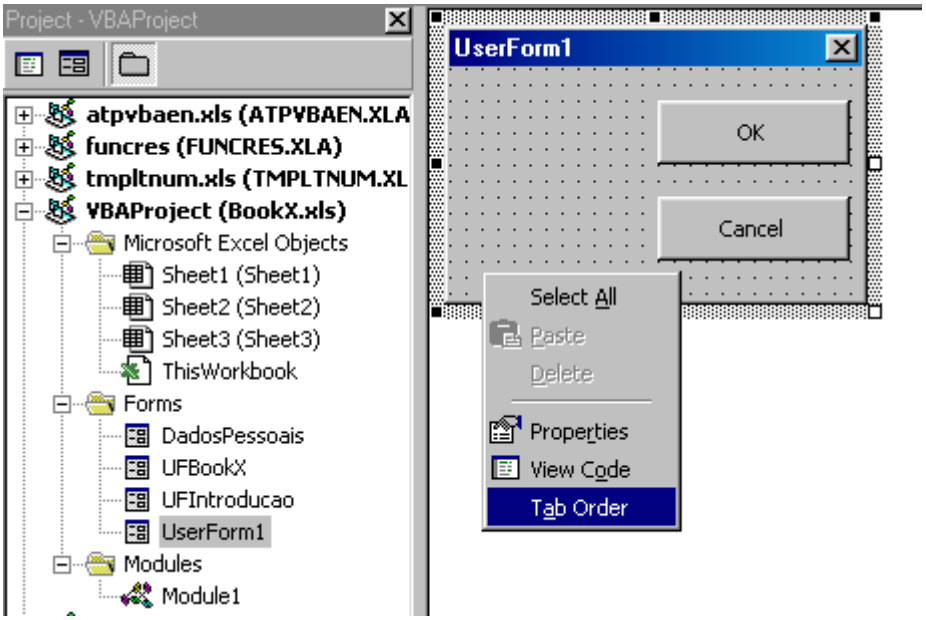




A ilustração acima tem por objetivo chamar a atenção do leitor para os seguintes fatos : (1)Ao criarmos a User Form DadosPessoais a mesma aparecerá na sua view particular (Observe que nosso VBAProject(BookX.xls) conta, sob o folder Forms, com 4 User Forms) (2)Ao ser a User Form DadosPessoais criada, sob o folder Forms (View Project Explorer) a barra de ferramentas Toolbox estará disponivel para que o Usuário selecione e aplique os controles ActiveX que lhe interessarem. (3)A lista de propriedades da User Form DadosPessoais pode ser obtida fazendo : >Clique na User Form DadosPessoais com o (R)ight (M)ouse (B)utton; <com o LMB selecione a opção Properties, ou então, se a UserForm estiver no modo de edição (as alças de arraste estarão presentes no seu contorno) acione <View; <Properties Window (4)O valor da propriedade Name foi alterado de UserForm1 para DadosPessoais. Esse alteração se reflete e fica registrada na view Project Explorer como mostra a ilustração acima. (5)Uma vez aplicado um controle na User Form, para acessarmos sua lista de propriedades, tambem há dois caminhos : >ActiveX (o controle entra no modo de edição); <Properties ou então : <ActiveX; <View; <Properties Window

Compreendendo a utilidade da propriedade Tag.

A propriedade **Tag** retem o resultado de um evento, propriedade ou qualquer outra informação que diga respeito a User Form. O exemplo a seguir esclarece : Acesse o VBE; <Project Explorer; <VBAProject(BookX.xls); >Insert Module;
Com esses passos, na view Project Explorer surgirá um folder que o VBA apelidou Modules e sob esse folder estará o modulo Module1.
A partir da Toolbox acrescentamos dois controles ActiveX(commandbuttons) na UserForm1. Em seguida alteramos a propriedade Caption desses dois commandbuttons para OK e Cancel.



A ilustração acima nos mostra a view Project Explorer com os 3 folders Microsoft Excel Objects; Forms e Modules. Na User Form1 (sob o folder Forms), aplicamos dois controles ActiveX. Ao clicarmos na UserForm1 com o RMB um menu auxiliar se apresenta. No referido menu selecionamos a opção Tab Order (a ilustração exibe esse menu auxiliar)
Na view Tab Order estabelecemos a ordem de prioridade para as ações do Usuário sobre os controles ActiveX. Isso é absolutamente necessário. Imaginem uma User Form na qual aplicamos vários controles ActiveX. Por onde começar? Essa preocupação fica no entanto resolvida quando estabelecemos prioridades. Numa UserForm com várias textboxes por exemplo, uma vez estabelecida a prioridade, tendo preenchido uma delas, saberemos qual será a próxima a ser preenchida. Costuma-se usar a tecla Tab (keyboard tab) para verificar a ordem de prioridade. Ao faze-lo observamos que o foco sobre o controle ActiveX vai se alterando na medida em que Tab é acionado.

Na UserForm1 do exemplo, estabelecemos o controle OK em primeiro lugar e em segundo lugar o controle Cancel. Criada a UserForm e os controles ActiveX que nos interessam, podemos cuidar das procedures. Chamamos a atenção do leitor para os seguintes fatos: (1)Não podemos numa mesma procedure usar os métodos Show e Hide. Isso porque quando invocamos o metodo Show a procedure sofre um break, uma interrupção, para que o Usuário execute determinadas ações. Será necessário que o Usuário acesse outros controles para dar prosseguimento ao processo, que pode ser o de descarte da User Form. Fica então claro que na mesma instancia que convocou a User Form para a tela, não podemos descarta-la. (2)Os controles ActiveX devem ter por traz suas event procedures, tema que já estudamos.

Em module1 introduzimos a procedure TestandoUserForm1( )

```
Sub TestandoUserForm1( )
UserForm1.Show
If UserForm1.Tag = vbOK Then
    MsgBox "Voce teclou OK"
Else
   MsgBox "Voce teclou Cancel"
End If
End Sub
```

As instruções abaixo são a resposta do VBA ao clique duplo no controle ActiveX OK

```
Private Sub CommandButton1_Click()
End Sub
```

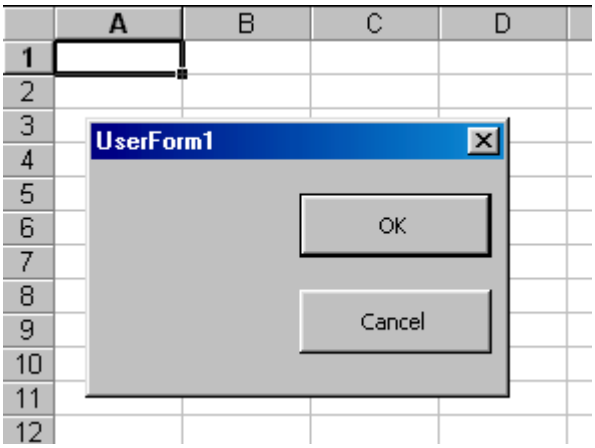
Dando prosseguimento, vamos introduzir nessa “event procedure” as instruções (1)que descartem a UserForm1 da tela e (2)que retenham na propriedade Tag a ação do Usuário quando a UserForm estiver visivel. (3)Essa procedure corresponderá ao evento “clicar no controle OK”

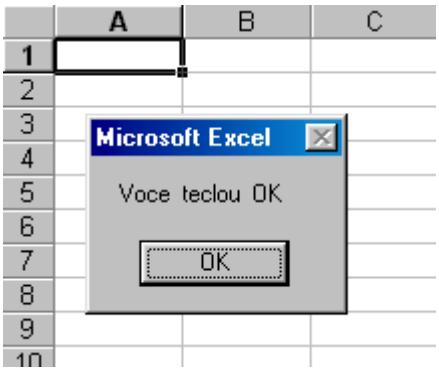
```
Private Sub CommandButton1_Click()
UserForm1.Hide
UserForm1.Tag = vbOK
End Sub
```

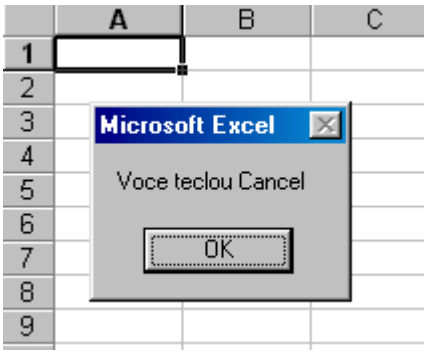
Em analogia aos procedimentos anteriores, tomamos as mesmas medidas em relação ao controle ActiveX Cancel.

```
Private Sub CommandButton2_Click()
UserForm1.Hide
UserForm1.Tag = vbCancel
End Sub
```

Observe o foco sobre o controle OK como resultado das ações que definiram as prioridades







As tres ilustrações acima mostram como funcionou a procedure TestandoUserForm1( ) no module1, em conjunto com as duas event procedures.

Dando sequencia aos exemplos, vamos montar uma User Form que capte o nome da pessoa e o seu nivel de escolaridade e em seguida, transfere esses dados para a planilha. Vide detalhes no preenchimento do quadro abaixo, fase que corresponde ao planejamento, onde se prevê os elementos que participarão do projeto, as propriedades que sofrerão alterações, etc...

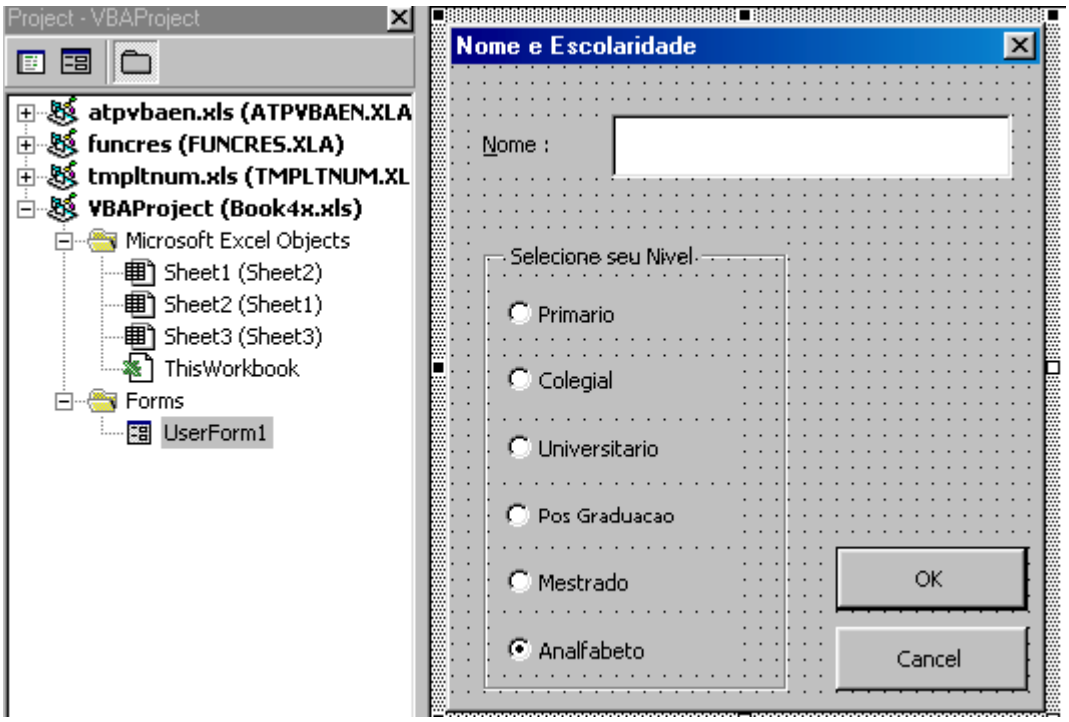
	Propriedade Name	Propriedade Caption	Propriedade Accelerator	Propriedade Default	Propriedade Cancel	Propriedade Value	Tab Index
CommandButton1	cbChamaUserForm	Escolaridade	----------	----------	----------	----------	----
UserForm1	UserForm1	Nome e Escolaridade	----------	----------	----------	----------	----
Controles ActiveX			----------	----------	----------		
CommandButton1	CbOK	OK	----------	TRUE	FALSE		6
CommandButton2	CbCancel	Cancel	----------	FALSE	TRUE		7
TextBox1	TextNome	----------	----------	----------	----------		1
Label1	Lb1	Nome :	N	----------	----------		0
Frame Control	Frame1	Selecione seu Nivel	----------	----------	----------		2
OptionButton1	obPrimario	Primario	P	----------	----------		0
OptionButton2	obColegial	Colegial	C	----------	----------		1
OptionButton3	obUniversitario	Universitario	U	----------	----------		2
OptionButton4	obPosGraduado	Pos Graduacao	G	----------	----------		3
OptionButton5	obMestrado	Mestrado	M	----------	----------		4
OptionButton6	obAnalfabeto	Analfabeto	A	----------	----------	TRUE	5

Como parte desse planejamento acrescentamos (1)que o objeto cbChamaUserForm deve ser criado na Sheet2 do arquivo Book4x.xls, (2)que a event procedure cbOK_Click( ) relativa ao controle ActiveX deve responder pela transferencia dos dados da User Form para a planilha.

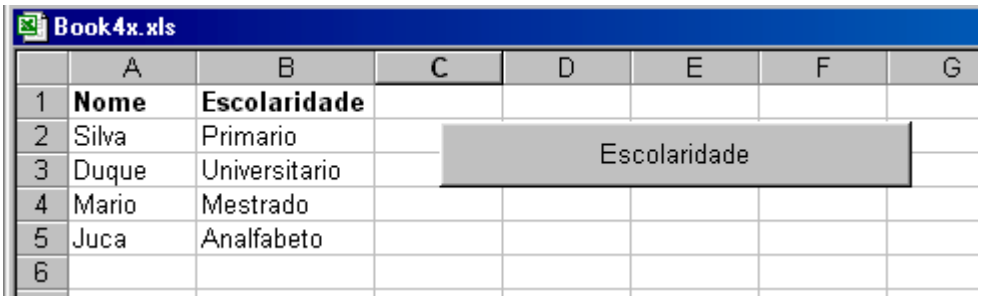
Os próximos passos serão na ordem : (1)criar na Sheet2 o objeto cbChamaUserForm, a partir da barra de ferramentas Control Toolbox, (2)criar a UserForm1 no VBE; (3)aplicar os controles ActiveX segundo o quadro acima, (4)Alterar as propriedades desses controles segundo o quadro acima, (5)Estabelecer as prioridades com Tab Order, (6)Estabelecer as prioridades internamente ao controle Frame, (7)Escrever as 3 procedures que finalizam o projeto.

```
Private Sub cbChamaUserForm_Click()
  UserForm1.Show
End Sub

Private Sub cbCancel_Click()
 Unload  UserForm1
End Sub

Private Sub cbOK_Click()
Sheets("sheet2").Activate
linha_seguinte = _
Application.WorksheetFunction.CountA(Range("A:A")) + 1
Cells(linha_seguinte, 1) = txNome.Text
If obPrimario Then Cells(linha_seguinte, 2) = "Primario"
If obColegial Then Cells(linha_seguinte, 2) = "Colegial"
If obUniversitario Then Cells(linha_seguinte, 2) = "Universitario"
If obPosGraduado Then Cells(linha_seguinte, 2) = "Pos Graduado"
If obMestrado Then Cells(linha_seguinte, 2) = "Mestrado"
If obAnalfabeto Then Cells(linha_seguinte, 2) = "Analfabeto"
txNome.SetFocus  : txNome.Text = "" :  obAnalfabeto = True
End Sub
```

O leitor poderá visualizar na ilustração abaixo, a view Project Explorer com os folders **Microsoft Excel Objects**, o folder **Forms** sob o qual inserimos o objeto UserForm1 e finalmente a UserForm1 com os controles ActiveX já com suas propriedades alteradas



A ilustração abaixo nos mostra o projeto já na sua fase de execução, na qual os dados inseridos na User Form foram transferidos para a Worksheet



Observações :
(1)Estando na view Object (onde montamos a User Form, seus controles, definimos as propriedades e estabelecemos as prioridades) se acionamos (F7) ganhamos acesso as procedures (view Code) (2)Estando na view Code onde editamos as procedures, se acionamos Shft (F7) voltamos a view Object

A procedure abaixo faz o mesmo serviço que a anterior. Contudo ao invés de usarmos a instrução **if** para controle do fluxo, preferimos usar, por razões didáticas, a instrução **Select Case**.

```
Private Sub cbOK_Click()
Dim ob As Object
Sheets("sheet2").Activate
Linha_Seguinte = _
Application.WorksheetFunction.CountA(Range("a:a")) + 1
Cells(Linha_Seguinte, 1) = txNome.Text

Select Case True
  Case obPrimario.Value
    Cells(Linha_Seguinte, 2).Value = "Primario"
  Case obColegial.Value
    Cells(Linha_Seguinte, 2).Value = "Colegial"
  Case obUniversitario.Value
    Cells(Linha_Seguinte, 2).Value = "Universitario"
  Case obPosGraduado.Value
    Cells(Linha_Seguinte, 2).Value = "Pos Graduado"
  Case obMestrado.Value
    Cells(Linha_Seguinte, 2).Value = "Mestrado"
  Case obAnalfabeto.Value
    Cells(Linha_Seguinte, 2).Value = "Analfabeto"
End Select

txNome.SetFocus
txNome.Text = ""
obAnalfabeto = True
End Sub
```

Trabalhando com outros Eventos

Começamos abrindo um novo arquivo, selecionamos a opção SaveAs salvando um novo projeto que vamos denominar "**Eventos nas User Forms.xls**"
Na sheet1 inserimos dois Buttons (a partir da barra de ferramentas **Forms**). Trocamos o nome do primeiro objeto de Button1 para **Load User Form1**. Trocamos o nome do segundo objeto de Button2 para **Show User Form1.** Essas duas ações são executadas a partir do menu auxiliar (o menu auxiliar se apresenta quando clicamos no objeto com o RMB) onde selecionamos a opção Edit Text. O quadro ilustrativo a seguir mostra em detalhes os elementos constitutivos desse novo projeto, que chamamos fase de planejamento.

"Eventos nas User Forms.xls"		
Button1	Sheet1	Load User Form1
Button2	Sheet1	Show User Form1
Controles ActiveX	**Name**	**Caption**
UserForm1	UserForm1	UserForm1
CommandButton1	cbShow	Mostrar outra UserForm
CommandButton2	cbHide	Hide
CommandButton3	cbUnLoad	UnLoad
UserForm2	UserForm2	UserForm2
CommandButton1	cbOK	OK

No VBE, inserimos Module1, ambiente onde vamos escrever as duas procedures relativas aos botões "Load User Form1" e "Show User Form1"

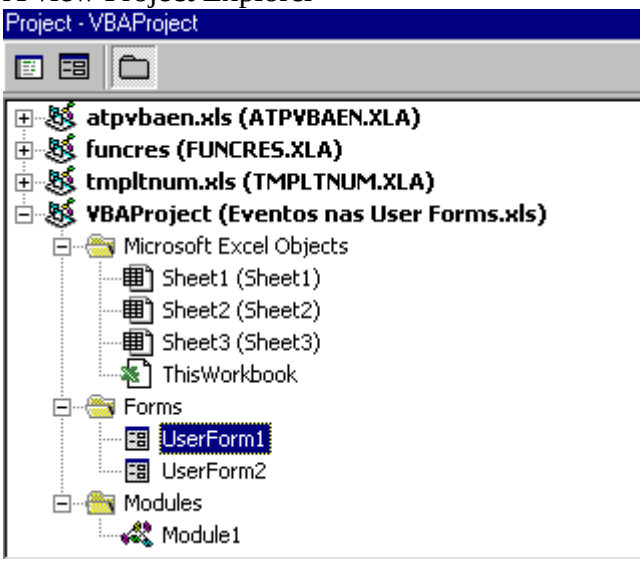
```
Sub LoadDialog( )
   Load UserForm1
End Sub

Sub ShowDialog( )
   UserForm1.Show
End Sub
```

A conexão entre os 2 botões "Load UserForm1" e "Show UserForm1" e as macros se faz atraves do menu auxiliar, acessando a opção Assign Macro.

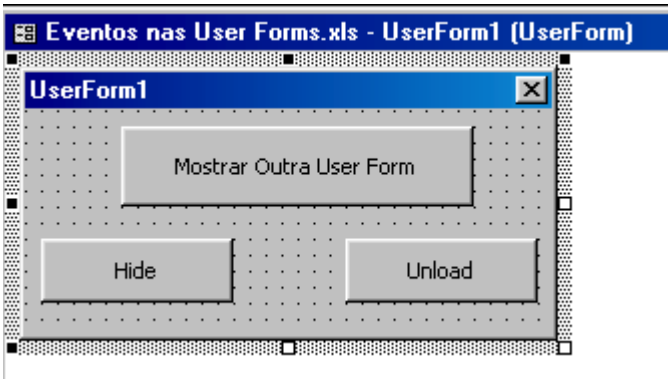
O leitor deve observar (1) que a instrução Load UserForm1 carrega na memória o objeto UserForm1 sem no entanto exibi-lo na tela. (2)que a instrução UserForm1.Show foi empregada para exibir o objeto UserForm1 na tela sem no entanto carrega-lo inicialmente na memória. Ocorre que o metodo Show, quando Load está ausente, faz automaticamente os dois serviços : Carrega na memória e exibe.(3)O objeto UserForm1 continua na memoria enquanto não dermos instrução para descarrega-lo.

Os próximo passo será montar as duas User Forms e tomar as demais providencias, assim como escrever as "event procedures" ligadas aos controles ActiveX usados. Na view Project Explorer teremos 3 folders a saber : Microsoft Excel Objects, Forms e Modules. Sob o folder Forms temos duas UserForms (UserForm1, UserForm2) e sob o folder Modules temos Module1.

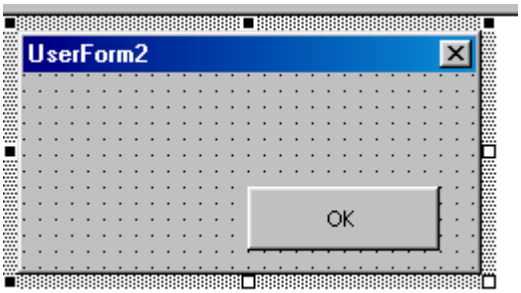
As ilustrações a seguir mostram em detalhes os elementos desse projeto
A view Project Explorer



A UserForm1



A UserForm2



Executamos esse projeto (“Eventos nas User Forms.xls”) obtendo sucessivamente (1)Ao clicarmos no objeto “Load User Form1” a mensagem “Initialize Event ocorreu” se apresenta. Clicamos no botão OK da MessageBox. (2)Clicando-se agora no botão “Show User Form1” a mensagem “Active Event ocorreu” se apresenta. Clicando-se no botão OK dessa MessageBox e a UserForm1 surge na tela. (3)Clicando-se no controle “Mostrar outra User Form” a mensagem “Deactivate ocorreu” aparece. Clicamos no botão OK dessa

MessageBox, quando então surge na tela a UserForm2. (4)Clicando-se no controle OK dessa UserForm2, a mensagem “Active Event ocorreu” se apresenta. (5)Clicando-se no controle Unload da UserForm1, a mensagem “Query Close Event ocorreu” se apresenta. Clicando-se no botão OK dessa MessageBox, a mensagem “Terminate event ocorreu” se apresenta.

```
Private Sub cbUnload_Click( )
   Unload Me                        'O mesmo que Unload UserForm1
End Sub

Private Sub cbHide_Click( )
   Me.Hide                          'O mesmo que UserForm1.Hide
End Sub

Private Sub cbShow_Click( )
   UserForm2.Show
End Sub

Private Sub UserForm_Activate()
   MsgBox "O Activate Event Ocorreu", vbInformation
End Sub

Private Sub UserForm_Deactivate()
   MsgBox "O Deactivate Event Ocorreu", vbInformation
End Sub

Private Sub UserForm_Initialize()
   MsgBox "O Initialize Event Ocorreu", vbInformation
End Sub

Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)
   MsgBox "O QueryClose Event Ocorreu", vbInformation
End Sub

Private Sub UserForm_Terminate()
   MsgBox "O Terminate Event Ocorreu, vbInformation"
End Sub
```

Com um clique duplo no Controle OK da UserForm2 preparamos a procedure que elimina da memoria a UsertForm2.

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

Usando a propriedade Tag

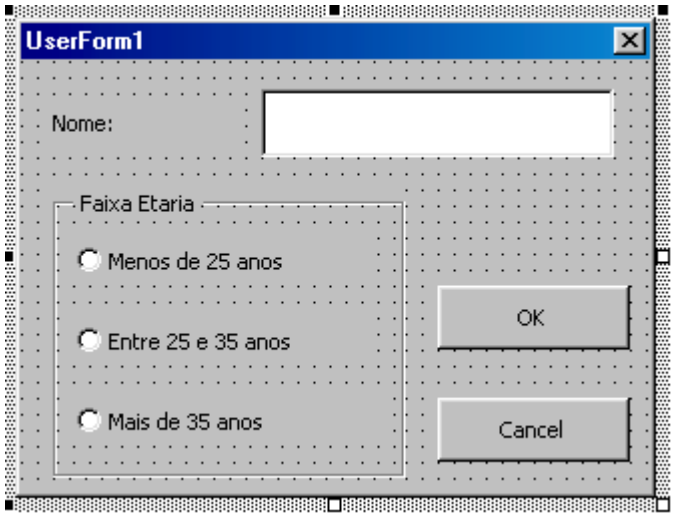
Seja montar uma UserForm de tal forma que uma vez preenchidos os seus requisitos (commandbuttons, textboxes, checkboxes, optionbuttons, etc...) nos apresente uma MessageBox com os dados desse preenchimento.
Abrimos um novo arquivo que denominamos Book7x.xls. Na Sheet1 inserimos um button (a partir da barra de ferramentas Forms; Caption: Faixa Etaria) que

chamará a procedure MinhaProc.xls (Module1). As duas “event procedures” correspondentes aos dois commandbuttons cbOK e cbCancel aplicados na UserForm são tambem necessárias para encerrar o serviço.

O quadro abaixo resume os elementos que farão parte desse projeto.

Objeto	Name	Caption
Button		Faixa etaria
UserForm	UserForm1	UserForm1
Controles ActiveX		
Label	Lb1	Nome:
TextBox	txNome	
Frame	Frame1	Faixa Etaria
OptionButton1	obMenos25	Menos de 25 anos
OptionButton2	obEntre2535	Entre 25 e 35 anos
OptionButton3	obMais35	Mais de 35 anos
CommandButton1	cbOK	OK
CommandButton2	cbCancel	Cancel

Aspecto da UserForm apos introduzir os controles e alterar as propriedades conforme planejado. Em termos de prioridades há que se estabelecer duas classes : A primeira classe diz respeito aos controles: label, textbox, frame e os dois commandbuttons. A segunda classe diz respeito aos tres controles optionbuttons internos ao controle frame. Consequentemente voce terá que acessar duas vezes a view Tab Order para estabelecer como caminhará o foco durante o preenchimento.



A ordem de execução é a seguinte : Clique no botão Faixa Etaria (Sheet1) para trazer UserForm1 para a tela. Com a User Form visivel voce pode modificar o texto entrando com o nome adequado e selecionando a faixa etária correspondente. Observe que a procedure não tem nenhum comando de interrupção, mas no momento em que o usuário deve se definir, as coisas

ocorrem como se tivessemos colocado um break na procedure. Ao clicarmos em quaisquer dos dois controles OK ou Cancel, a propriedade Tag é carregada e só então a procedure MinhaProc( ) se conclui.

```
Sub MinhaProc()
Dim Mensagem As String
   With UserForm1
      .txNome.Text = "Digite Seu Nome Aqui”
      .obMenos25.Value = True
      .Show
      If .Tag = vbOK Then
         Mensagem = "Nome: " & .txNome.Text & Chr(13)
         If .obMenos25.Value Then
            Mensagem = Mensagem & "Idade: " & .obMenos25.Caption
         ElseIf .obEntre2535.Value Then
            Mensagem = Mensagem & "Idade: " & .obEntre2535.Caption
         Else
            Mensagem = Mensagem & "Idade: " & .obMais35.Caption
         End If
         MsgBox Mensagem
      End If
   End With
End Sub

Private Sub cbOK_Click()
Me.Hide                          ‘o mesmo que UserForm1.Hide
Me.Tag = vbOK                    ‘o mesmo que UserForm1. Tag
End Sub

Private Sub cbCancel_Click()
Me.Hide
Me.Tag = vbCancel
End Sub
```

Estaria errado se tivessemos programado Unload UserForm1 para as event procedures cbOK_Click e cbCancel_Click, porque não teriamos como capturar a propriedade Tag. A instrução Hide elimina a UserForm da tela mas não da memória.

Ajustando os controles na User Form

Ao inserirmos uma UserForm, observamos que a mesma vem com uma grade de pontos, elementos necessários e que servem de guia para aplicarmos nossos controles alinhados, com as mesmas dimensões e igualmente espaçados visando obter um resultado final dotado de certa estética. Apesar disso não é muito simples faze-lo. Eis algumas dicas para resolver essa questão. (1)Se voce aplicar 2 ou mais OptionButtons, segure a tecla Shift enquanto clica nos mesmos. Com esses controles selecionados, acesse a opção Format no Menu para alinha-los e espaça-los igualmente. (2)Se voce vai aplicar varios OptionButtons, aplique o primeiro e ajuste o seu tamanho. A partir desse modelo use o método Copy Paste para inserir os demais. (3)Vamos supor que voce vai aplicar 3 Option

Buttons na sua User Fom. Esses controles, num primeiro momento, estarão desalinhados e espaçados um dos outros de forma irregular. Segure a tecla Shift e mantendo-a calcada, clique no primeiro, segundo e terceiro OptionButton. Observará que esses 3 controles ficaram selecionados. Observará tambem que o OptionButton selecionado em primeiro lugar tem as alças de arraste brancas, enquanto os demais tem as alças de arraste pretas. Ao comandar o alinhamento, o mesmo será providenciado tomando por base o que tem as alças brancas. (4)O que foi dito em relação ao controle OptionButton se aplica aos demais controles.

DoEvents

Há casos em que o codificador deseja que a User Form saia de cena tão logo acione um controle. Neste caso as instruções **UserForm1.Hide:DoEvents** deverão se localizar no topo da procedure. O exemplo a seguir esclarece.

```
Private Sub CommandButton1_Click( )
   UserForm1.Hide                    ‘ou Me.Hide
   DoEvents
   For Linha = 1 to 3
     Cells(Linha, 1) = Linha
   Next
   Unload UserForm1
End Sub
```

Validando dados de entrada

Em muitos casos há necessidade de testar os dados introduzidos pelo usuário ao preencher o controle textbox. Para tanto basta introduzir na procedure o teste:

```
Private Sub CommandButton1_Click()
  UserForm1.Hide                         'ou Me.Hide
  DoEvents
  If TextBox1.Text = "" Then
     MsgBox "Entre com um Nome"
     Exit Sub
  End If
  Unload UserForm1
End Sub
```

Chamo a atenção do leitor (1)para a instrução If TextBox1.Text = “” Then ...Se ao invés de “”(empty) escrevemos “ “(string de comprimento nulo) a instrução MsgBox não será executada pois a TextBox já vem preenchida com um string de comprimento nulo.(2) Ao ser executada a procedure, a UserForm1 entra em cena, aguardando que o usuário preencha a textbox. Mas não há nenhum comando para que essa UserForm entre em cena. Se o usuário entrar com o numérico 2100 preenchendo uma textbox que pede o seu salario, esse dado será um string enquanto não for convertido. Este fato pode ser comprovado na procedure :

```
Private Sub CommandButton1_Click()
UserForm1.Hide                    'Me.Hide
DoEvents
MsgBox TypeName(TextBox1.Text)
Unload UserForm1
End Sub
```

A procedure abaixo converte em numérico o string fornecido pelo usuário

```
Private Sub CommandButton1_Click()
UserForm1.Hide                    'Me.Hide
DoEvents
salario = Val(TextBox1.Text)
MsgBox TypeName(salario)
Unload UserForm1
End Sub
```

O Evento Initialize

Abra um novo projeto (Book9x.xls) e insira duas User Forms(UserForm1 e UserForm2). Com um clique duplo do mouse na User Form1 o VBA escreverá na view Book9x.xls-UserForm1(Code): Private Sub UserForm_Click( ) : End Sub. Complemente essa procedure :

```
Private Sub UserForm_Click( )   ´(1)
Unload UserForm1
UserForm2.Hide
UserForm1.Show
End Sub
```

Com um clique duplo do mouse na User Form1(em edição) o VBA escreverá na view UserForm1(Code): Private Sub UserForm_Click( ) : End Sub. Abandone essas duas instruções e clique na opção **Initialize** da combobox da direita(procedure) e o VBA lhe escreve :
Private Sub UserForm_Initialize( ) : End Sub. Complemente essa proc com as instruções Load UserForm2 : UserForm2.Show

```
Private Sub UserForm_Initialize( )  ‘(2)
  Load UserForm2
  UserForm2.Show
End Sub
```

Com um clique duplo na User Form2(em edição) o VBA escreverá na view Book9x.xls-UserForm2(Code):

```
Private Sub UserForm_Click()        ‘(3)
   UserForm2.Hide
End Sub.
```

Em resumo, temos 3 procedures, duas relativas a UserForm1 e uma relativa a UserForm2. Observe que se estivésssemos escrito as tres procedures na mesma

view UserForm1(Code) incorreríamos num erro de ambiguidade (duas procs com o mesmo nome). Com o mouse na proc (1) clicamos na tecla (F5) para executa-la. Com isso entrará em cena a UserForm2. Clicamos na UserForm2, e com isso entrará em cena a UserForm1. Clicando-se na UserForm1 entrará em cena a UserForm2. E assim por diante. Para encerrar o processo clicar em X (Close)

Os eventos Initialize e Activate

Quando usamos o metodo Show os eventos **Initialize** e **Activate** são automaticamente invocados nesta ordem. Para comprovar essa afirmação, abrimos um novo arquivo (Book11x.xls), inserimos os objetos Module1 e UserForm1. Na view Book11x.xls-Module1(Code) escrevemos a procedure Teste( ) abaixo. Obs: A instrução Unload UserForm1 pode ser removida da procedure Teste( ) e inserida na procedure Private Sub UserForm_Activate( ) substituindo a instrução UserForm1.Hide

```
Sub Teste( )
  Unload UserForm1
  UserForm1.Show
End Sub
```

Com um clique duplo na UserForm1, o VBA nos prepara as instruções Private Sub UserForm_Click( ) : End Sub, que abandonamos. Na combobox da direita clicamos na opção Initialize. O VBA irá escrever Private Sub UserForm_Initialize( ) : End Sub, que deve ser complementada com as intruções adequadas. Como queremos demonstrar que essa procedure é ativada pelo metodo Show, essa complementação pode ser feita atraves de uma mensagem que elimine duvidas. Siga os mesmos passos em relação a procedure Private Sub UserForm_Activate( )

```
Private Sub UserForm_Initialize( )
    MsgBox “Inicializando UserForm1”
End Sub

Private Sub UserForm_Activate( )
   MsgBox “Ativada a UserForm1”
   UserForm1.Hide
End Sub
```

Com (F5) acionamos a procedure Teste( ).Diante do metodo Show a mensagem “Inicializando UserForm1” entra em cena na Sheet1. Clicamos no botão OK e a mensagem “Ativada a UserForm1” entra em cena assim como a UserForm1.

A propriedade Tag e os Eventos Initialize e Activate

Abrimos um novo arquivo (Book12x.xls) onde vamos inserir duas User Forms : UserForm1 e UserForm2 e um modulo(Module1). Introduzimos na propriedade

tag da userForm1o texto Initialize_Tag e na propriedade tag da UserForm2 o texto Activate_Tag. Na view Book12x.xls-Module1(Code) escrevemos :

```
Sub Teste( )
   UserForm1.Show
End Sub
```

Com dois cliques na UserForm1, o VBA prepara na view Book12x.xls – User Form1(Code) as instruções Private Sub UserForm_Click( ) : End Sub que abandonamos. Clicamos nas opções Initialize e Activate da combobox da direita e complementamos essas duas event procedures conforme abaixo:

```
Private Sub UserForm_Initialize
  MsgBox UserForm1.tag
End Sub

Private Sub UserForm_Activate
  MsgBox UserForm2.tag
End Sub
```

Operação :
Executamos a Procedure Teste( ). A mensagem Initialize_Tag entra em cena mostrando que o método Show acionou a event procedure Private Sub User Form1_Initialize. Quando clicamos no botão OK dessa messagebox entra em cena a mensagem Activate_Tag concomitantemente com a UserForm1. Fica assim comprovado mais uma vez que o metodo Show invoca dois eventos : Initialize e Activate nesta ordem.

A instrução Load e a Event procedure Initialize

Ao se deparar com a instrução Load UserForm, o VBA executará primeiramente a event procedure Private Sub UserForm_Initialize( ). Para comprovar esse fato façamos algumas alterações nas procedures da demonstração anterior :

```
Sub Teste( )
   Load UserForm1
End Sub

Private Sub UserForm_Initialize()
MsgBox "Voce acaba de carregar a UserForm1 na memoria"
End Sub
```

Ao executarmos Teste( ) entrará em cena a mensagem “Voce acaba de carregar a UserForm1 na memória”. Executando-se Teste( ) novamente, nada irá ocorrer devido ao fato de que a UserForm1 já estará na memória. Para contornar esse problema, acrescentamos a instrução Unload UserForm1 na procedure Teste( ).

Enfatizamos o planejamento prévio

Queremos montar uma UserForm para anotar endereços de médicos. Trata-se de um caso especial porque esse profissional pode ter vários endereços (endereço residencial, endereço do consultório, endereço do hospital onde opera na parte da manhã e do hospital onde opera na parte da tarde, endereço do local onde atende as Segundas e Quartas feiras à tarde, onde pode ser achado nos fins de semana, etc….). Para simplificar a questão vamos tornar obrigatorio o fornecimento de pelo menos dois endereços (endereço da residencia e do consultorio) e deixar como opção, fornecer ou não um terceiro endereço. Portanto nessa UserForm devemos ter no mínimo 3 textboxes. Como sempre fazemos, abrimos um novo arquivo: Book14x.xls e planejamos os elementos que farão parte desse projeto.

	Name	Caption	Tag	Outros
Button		Teste		
UserForm	UserForm1	UserForm1		
Controles ActiveX				
CommandButton1	CommandButton1	CommandButton1		Opcional
textbox1	txEndereco1	--------	Obrigatorio	
textbox2	txEndereco2	--------	Obrigatorio	
textbox3	txEndereco3	--------		
Label1	LbResid	Residencia		
Label2	LbEscr	Escritorio		
Label3	LbOutros	End. Opcional		

O quadro acima deixa claro que na Sheet1 devemos ter um button (criado a partir da Barra de Ferramentas Forms) que chame a procedure Teste( ) (a ser desenvolvida na view Book14x.xls-Module1(Code)). Na view Project Explorer, inserimos uma UserForm(UserForm1) e um Modulo(Module1). Na UserForm aplicamos os controles indicados, observando (1)que a propriedade Tag, para duas primeiras textboxes foi preenchida com o string "Obrigatorio". A finalidade será usar este lance na event procedure. (2)Que o controle CommandButton1 sendo opcional neste projeto, o deixamos de lado. (3)Que as instruções InputBox, conforme demonstramos, podem ser posicionadas na Sheet1 adequadamente, desde que providenciemos suas coordenadas de forma a evitar a superposição de objetos. (4)Que o metodo Show deflagra as events procedures Initialize e Activate, nesta ordem, e que nos aproveitamos desse fato. (5)A event procedure Initialize tem por finalidade preparar o preenchimento, no sentido de tornar mais eficiente ou acelerar as ações que se seguirem. (6)Que incluimos a instrução For Each …Next ao final da event procedure Activate( ) para demonstrar como é possivel percorrer uma coleção de controles capturando, nesse percurso, elementos de interesse do usuário. (7)Que incluimos na MsgBox final um ícone (vbInformation) e um título para essa caixa de mensagem. (8)Que a instrução MsgBox ao final, embora desnecessária, tem por objetivo interromper a execução deixando à mostra a UserForm preenchida com os dados fornecidos pelo usuário.

Na view Book14x.xls-Module1(Code) escrevemos

```
Sub Teste()
   Sheets("Sheet1").Activate
   UserForm1.Show
End Sub
```

Na view Book14x.xls-UserForm1(Code) escrevemos

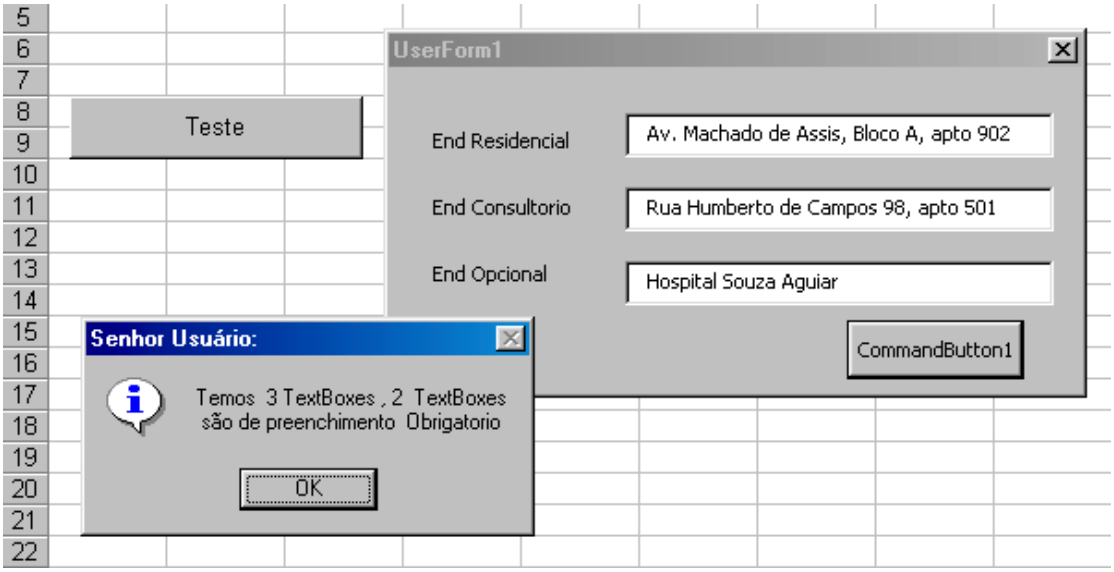
```
Private Sub UserForm_Initialize()
TxEndereco1.Text = "Obrigatorio"
TxEndereco2.Text = "Obrigatorio"
TxEndereco3.Text = "Opcional"
End Sub

Private Sub UserForm_Activate()
Do While UserForm1.TxEndereco1.Tag = UserForm1.TxEndereco1.Text
 end1 = InputBox("Seu endereco Residencial?", "Case Sensitive", , 100, 100)
    If UserForm1.TxEndereco1.Tag <> end1 And end1 <> "" Then
      UserForm1.TxEndereco1.Text = end1
      Exit Do
      End If
Loop
Do While UserForm1.TxEndereco2.Tag = UserForm1.TxEndereco2.Text
end2 = InputBox("Endereço do seu Escritorio?", "Case Sensitive", , 100, 100)
  If UserForm1.TxEndereco2.Tag <> end2 And end2 <> "" Then
      UserForm1.TxEndereco2.Text = end2
      Exit Do
    End If
Loop
Do While UserForm1.TxEndereco3.Text = "Opcional"
  end3 = InputBox("Este endereço é Opcional", , , 100, 100)
  If end3 = "" Or end3 <> "" Then
     UserForm1.TxEndereco3.Text = end3
     Exit Do
  End If
Loop
contador = 0
contatag = 0
For Each controles In UserForm1.Controls
If TypeName(controles) = "TextBox" Then
  If controles.Tag = "Obrigatorio" Then
    contatag = contatag + 1
  End If
contador = contador + 1
End If
Next controles
Messg = "Temos  " & contador & " TextBoxes , " & contatag
Messg = Messg & "  TextBoxes " & Chr(13) & _
" são de preenchimento  Obrigatorio"
MsgBox Messg, vbInformation, "Senhor Usuário:"
Unload UserForm1
End Sub
```

Uma segunda versão para esse projeto seria usarmos o contrôle CommandButton1 para deflagrar os diálogos com o usuário, necessários ao preenchimento dos dados. Neste caso, voltamos à UserForm e incluimos esse controle (A partir da barra de ferramentas Toolbox) restabelecendo a Tab Order. Na view Book14x.xls-UserForm1(Code) e na combobox da esquerda selecionamos CommandButton1, na combobox da direita clicamos na opção Click. O VBA irá escrever : Private Sub CommandButton1_Click() : End Sub.

Finalmente, inserimos as instruções abaixo e aproveitamos para relembrar o uso da estrutura With…End With que facilita a escrita e a leitura do código.

```
Private Sub CommandButton1_Click()
With UserForm1
Do While .TxEndereco1.Tag = .TxEndereco1.Text
 end1 = InputBox("Seu endereco Residencial?", "Case Sensitive", , 100, 100)
    If .TxEndereco1.Tag <> end1 And end1 <> "" Then
       .TxEndereco1.Text = end1
        Exit Do
   End If
Loop

Do While .TxEndereco2.Tag = .TxEndereco2.Text
end2 = InputBox("Endereço do seu Escritorio?", "Case Sensitive", , 100, 100)
  If .TxEndereco2.Tag <> end2 And end2 <> "" Then
     .TxEndereco2.Text = end2
      Exit Do
  End If
Loop

Do While .TxEndereco3.Text = "Opcional"
  end3 = InputBox("Este endereço é Opcional", , , 100, 100)
  If end3 = "" Or end3 <> "" Then
     .TxEndereco3.Text = end3
   Exit Do
   End If
Loop
End With

contador = 0
contatag = 0
For Each controles In UserForm1.Controls
If TypeName(controles) = "TextBox" Then
  If controles.Tag = "Obrigatorio" Then
       contatag = contatag + 1
  End If
contador = contador + 1
End If
Next controles

Messg = "Temos  " & contador & " TextBoxes , " & contatag
Messg = Messg & "  TextBoxes " & Chr(13) & _
" são de preenchimento  Obrigatorio"
MsgBox Messg, vbInformation, "Senhor Usuário:"
Unload UserForm1
End Sub
```

Aspecto final do projeto após o preenchimento dos dados



Fim da Aula13