Noções de VBScript no Software Elipse E3: Lição 1 - Caixa de Diálogo

Uma Message Box mostra uma caixa de diálogo e espera que o usuário clique num botão, retornando um valor indicador do botão que foi clicado.

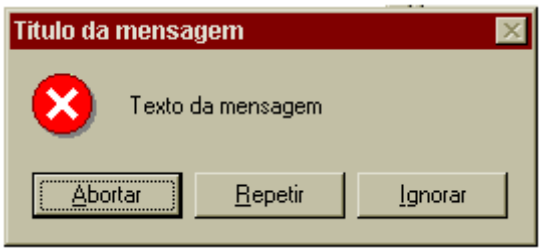


Figura 1

A sintaxe é a seguinte:

```
MsgBox(Mensagem, Botões, Título)
```

Onde:
Mensagem: Texto a ser exibido na caixa de diálogo
Botões: Ver tabela abaixo
Título: Título da caixa de diálogo

BOTÕES

CONSTANTE	VALOR	DESCRIÇÃO
vbOKOnly	0	Mostra o botão OK
vbOKCancel	1	Mostra botões de Ok e Cancel
vbAbortRetryIgnore	2	Mostra botões de Abortar, Tentar, Ignorar
vbYesNoCancel	3	Mostra botões de Sim, Não, Cancel
vbYesNo	4	Mostra botões de Sim e Não
vbRetryCancel	5	Mostra botões de Tentar e Cancelar
VbCritical	16	Mostra o ícone de Perigo
vbQuestion	32	Mostra o ícone de Interrogação
vbExclamation	48	Mostra o ícone de Exclamação
vbInformation	64	Mostra o ícone de Informação
vbDefaultButton1	0	Coloca como padrão o 1º botão
vbDefaultButton2	256	Coloca como padrão o 2º botão
vbDefaultButton3	512	Coloca como padrão o 3º botão
vbDefaultButton4	768	Coloca como padrão o 4º botão
vbApplicationModal	0	Aplicação modal. O utilizador deve responder à caixa de mensagem antes de continuar a trabalhar na aplicação.
vbSystemModal	4096	Sistema modal. Todas as aplicações são suspensas até que o utilizador responda à caixa de mensagens.

A função MsgBox retorna os seguintes valores:

VALORES

CONSTANTE	VALOR	DESCRIÇÃO
vbOK	1	OK
vbCancel	2	Cancel
vbAbort	3	Abort
vbRetry	4	Retry
vbIgnore	5	Ignore
vbYes	6	Yes
vbNo	7	No

Para o exemplo da Figura 1, escrevemos a seguinte linha de código:

```
MsgBox "Texto da mensagem", vbAbortRetryIgnore + vbCritical, "Título da mensagem"
```

Caso deseje guardar a resposta do usuário, os parâmetros da função devem estar entre parênteses:

```
'Pergunta
resp = MsgBox("Texto da mensagem", vbAbortRetryIgnore + vbCritical, "Título da mensagem")
'Resposta
MsgBox "O usuário respondeu " & resp
```

Exercícios:

1. Escreva o código para as seguintes caixas de diálogo:

Figura 2

Figura 3

2. Escreva o código para as seguintes caixas de diálogo e mostre em uma segunda mensagem a resposta do usuário:

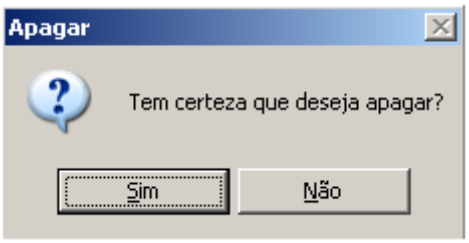


Figura 4

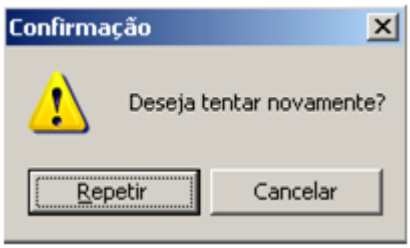


Figura 5

Application

A palavra Application representa a aplicação como um todo e pode indicar tanto funções que são executadas no E3 Viewer quanto no servidor. No caso, o objeto Application sabe de antemão quais funções devem ser executadas tanto para um quanto para outro caso. Não é possível, entretanto, executar funções de E3 Viewer dentro do servidor, assim como também não é possível executar funções de servidor dentro do E3 Viewer.

Exemplo:

Application.ChangePassword(): Função do Viewer que permite alterar a senha do usuário atual.
Application.Trace (mensagem): Função do Servidor que escreve em um arquivo texto.

Exercícios:

Viewer
1. Cadastre pelo menos 2 usuários, sendo um deles Administrador.
2. Crie um botão para fazer o login (método Login)
3. Crie um botão para alterar a senha do usuário (método ChangePassword).
4. Crie um botão para chamar a administração de usuários (método UserAdministration)
5. Crie um botão para confirmar a senha (método PasswordConfirm)
6. Crie um botão para exibir o nome completo do usuário (método GetFullUserName)

Servidor
7. Crie um tag Interno.
8. No evento OnStartRunning do tag Interno use o método Trace do Servidor para criar um arquivo texto com a mensagem "Tag iniciado"

Objetos do Servidor

Para se acessar um objeto que está sendo executado no servidor a partir de um Objeto de Tela ou um ElipseX, deve-se usar a diretiva Application.GetObject.

Exemplo:

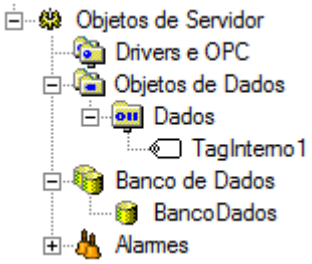


Figura 6

Application.GetObject("Dados.TagInterno1"): Tag Interno do Servidor Dados.
Application.GetObject("BancoDados"): Banco de Dados.

Exercícios:

9. Insira um tag demo e um tag interno na aplicação.
10. Mostre em uma MessageBox o nome do tag interno criado no exercício anterior.
11. Mostre em uma MessageBox o nome do tag demo criado no exercício anterior.

Item

O método Item retorna a referência para o objeto-filho do objeto que o chamou. Este método pode buscar um objeto tanto pelo nome quanto pelo índice (inteiro). Se o índice ou o nome especificado for válido, o método Item() retorna a referência do objeto. Caso contrário, o método retorna um erro de "parâmetro inválido".

Exemplo:

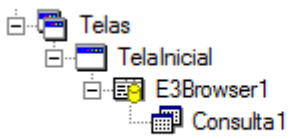
Figura 7

Screen.Item("E3Browser1"): Objeto E3Browser que está dentro da tela.
Screen.Item("E3Browser1").Item("Consulta1"):Objeto Consulta que está dentro do E3Browser.

Exercícios:

12. Crie dois textos na tela com a mensagem que desejar.

13. Agrupe os textos.
14. Mostre em uma MessageBox o nome do grupo criado no exercício anterior.
15. Mostre em uma MessageBox o nome texto criado no exercício anterior.
16. Qual das alternativas é o modo correto de se referenciar ao objeto Consulta1?



( ) a. Screen.Item("Consulta1")

( ) b. Item("TelaInicial").Item("Consulta1")

( ) c. Screen.Item("E3Browser1").Item("Consulta1")

( ) d. Item("E3Browser1").Item("Consulta1")

Figura 8

Comando Set

O VBScript implementa o conceito das linguagens orientadas a objeto, permitindo que uma variável do tipo Variant assuma a forma de um objeto qualquer, através do comando Set. Deste modo, a variável funciona como um ponteiro do objeto desejado, permitindo acessar seus métodos e propriedades.

Exemplo:

```
Set retangulo = Screen.Item("Retangulo1")
retangulo.BackgroundColor = RGB (255,0,0)
```

Sem o comando Set, a mesma chamada teria que ser:

```
Screen.Item("Retangulo1").BackgroundColor = RGB(255,0,0)
```

Aparentemente, não existe vantagem neste caso, pois se pode fazer tudo em uma única linha de código. Porém, se logo abaixo no mesmo script, outras operações forem necessárias, o processo se torna mais simples e rápido se a chamada do método Item() não tiver sido colocada em todas as linhas.

Exemplo ruim:

```
Screen.Item("Retangulo1").BackgroundColor = RGB(212,208,20)
Screen.Item("Retangulo1").Height = 500
Screen.Item("Retangulo1").Width = 500
```

Exemplo melhor:

```
Set Retangulo = Screen.Item("Retangulo1")
Retangulo.BackgroundColor = RGB(212,208,20)
Retangulo.Height = 500
Retangulo.Width = 500
```

Exercícios:

17. Insira um gráfico na tela com 1 pena. Configure-o para mostrar Legenda.
18. Ao clicar na legenda do gráfico, mostre em uma caixa de mensagem (msgbox) o nome e a cor da pena. Use o comando SET.
19. Ao clicar no gráfico, exiba a posição X e Y da pena em dois displays na tela.

Comando If...Else...ElseIf...End If

Permite a tomada de decisões durante a execução de um script. A sintaxe é a seguinte:

```
If condição Then
código que será executado se a condição for verdadeira.
Else
código que será executado se a condição NÃO for verdadeira
End if
```

Exemplos:

```
If Motor=0 Then
    Texto="Motor desligado"
Else
    Texto="Motor Ligado"
End if
```

Mais de uma condição pode ser verificada em um mesmo comando:

```
If <condição1> Then
    código que será executado se a condição1 for verdadeira.
Elseif <condição2> Then
    código que será executado se a condição2 for verdadeira.
Elseif <condição3> Then
    código que será executado se a condição3 for verdadeira.
Else
    código que será executado se nenhuma das condições for verdadeira
End if
```

Exemplos:

```
If Motores=0 then
    Texto="Motores desligados"
Elseif Motores=1 then
    Texto="Motor 1 ligado"
Elseif Motores=2 then
    Texto="Motor 2 ligado"
Elseif Motores=3 then
    Texto="Motores 1 e 2 ligados"
Else
    Texto="Erro de Status. Verifique os motores"
End if
```

Exercícios:

1. Escreva o código para as seguintes caixas de diálogo e mostre em uma segunda mensagem a resposta do usuário (Sim ou Não) e (Repetir ou Cancelar):

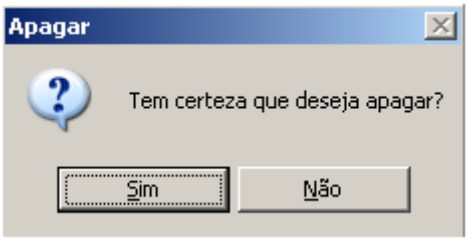


Figura 9

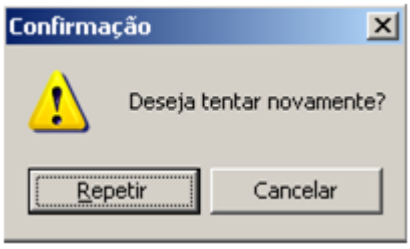


Figura 10

2. Crie um setpoint onde só seja possível digitar valores.
3. Ao mudar o valor do setpoint, altere a cor de um retângulo por script, conforme a tabela abaixo. Para qualquer valor fora da tabela o retângulo deverá ser preto.

Mínimo	Máximo	Cor
0	10	Azul
10	50	Verde
50	70	Amarelo
70	100	Vermelho

Comando Select Case

Executa um dos vários grupos de instruções. A sintaxe é:

```
Select Case <expressão>
    Case <Valor1>
        <bloco de instruções>
    Case <Valor2>
        <bloco de instruções>
End Select
```

Exemplos:

```
Select Case Motores
Case 0
    Texto="Motores desligados"
Case 1
    Texto="Motor 1 ligado"
Case 2
    Texto="Motor 2 ligado"
Case 3
    Texto="Motores 1 e 2 ligados"
Case else
    Texto="Erro de Status. Verifique os motores"
End select
```

Exercícios:

4. Insira duas Telas na sua aplicação.
5. Crie um menu com o nome de todas as telas usando o método SelectMenu do Viewer.
6. Ao clicar sobre o nome da tela no menu, abra a tela selecionada.

Eventos de Telas

KeyDown (KeyCode, Shift)

Ocorre no momento em que uma tecla é pressionada, independentemente do foco na Tela.
KeyCode: Número inteiro que identifica o caractere ASCII da tecla que foi pressionada
Shift: Mostra a tecla pressionada juntamente com o mouse: 4 = Tecla [Shift]; 8 = Tecla [Ctrl]; 12 = Teclas [Ctrl] + [Shift].

Exemplo:

```
Sub Tela1_KeyDown(KeyCode, Shift)
'Mostra uma caixa de mensagem quando o usuário pressiona uma tecla
MsgBox "Código da tecla: " & KeyCode
End Sub
```

Exercícios:

1. Quando o usuário apertar o Esc na Tela, sair do Viewer.
2. Quando a combinação Ctrl+Y for apertada, aparecer uma caixa de mensagem.

MouseDown (Button, ShiftState, MouseX, MouseY)

Ocorre quando se pressiona qualquer botão do mouse sobre a Tela. Utilize o evento MouseDown para determinar ações específicas quando a Tela for clicada pelo usuário.
Button: 1 - O botão do mouse pressionado é o esquerdo. 2 - O botão do mouse pressionado é o direito.
ShiftState: Mostra a tecla pressionada juntamente com o mouse: 4 = Tecla [Shift]; 8 = Tecla [Ctrl]; 12 = Teclas [Ctrl] + [Shift].
MouseX: Mostra a posição X onde o mouse foi clicado na Tela
MouseY: Mostra a posição Y onde o mouse foi clicado na Tela

Exemplo:

```
MsgBox "Coordenada X: " & MouseX & vbNewLine & "Coordenada Y: " & MouseY
```

Exercícios:

3. Ao clicar com o botão esquerdo na tela, exibir um calendário na coordenada x e y do clique.
4. Ao clicar com o botão direito do mouse na tela, abrir a paleta de cores (ShowColorPicker) e alterar a cor de fundo da tela.

OnPreShow (Arg)

Ocorre antes da Tela ser mostrada. A variável de evento **Arg** recebe o conteúdo do parâmetro Arg do método OpenScreen(), que gera esse evento.

Exemplo:

Em um botão na tela inicial, inserimos o script abaixo onde a palavra "Recado" é o parâmetro Arg.

```
Sub CommandButton1_Click()
Application.GetFrame("").OpenScreen("Tela1"), "Recado"
End Sub
```

No evento OnPreShow da Tela1, exibimos o Arg em uma caixa de mensagem:

```
Sub Tela1_OnPreShow(Arg)
    MsgBox Arg
End Sub
```

Exercícios:

5. Em uma tela, crie um setpoint e um botão para chamar uma segunda tela. Passe através do parâmetro Arg o que foi digitado no setpoint.
6. A segunda tela deve exibir em um display o parametro Arg.
7. Crie um XControl com um desenho de Motor, chamado XMotor.
8. Insira 3 XMotores na tela.
9. Ao clicar no XMotor deve ser aberta uma tela (sempre a mesma tela) e um display desta segunda tela deve mostrar o nome do XMotor clicado.

Eventos do Setpoint

Validate (Cancel, NewValue)

Ocorre após os testes de limites do SetPoint e antes do valor do SetPoint ser enviado para o tag. A finalidade deste evento é permitir que o usuário

cancele o envio do valor do SetPoint para o tag. O valor antigo pode ser acessado pela propriedade Value do SetPoint.
Cancel: booleano que indica se a operação de atribuição do valor do SetPoint ao tag deve ser cancelada (Cancel = True). O padrão é False
NewValue: valor que está sendo avaliado.

Exemplo:

```
Sub Setpoint1_Validate(Cancel, NewValue)
' Mostra uma MsgBox que pergunta ao usuário se ele deseja usar o novo valor digitado no SetPoint
mensagem = "Valor atual: " & Value & vbnewline & "Valor novo: " & NewValue & vbnewline & "Aceita o novo valor?"
If MsgBox(mensagem, vbQuestion+vbYesNo, "Evento Validate")=vbNo Then
    Cancel = True
End If
End Sub
```

Exercícios:

10. Insira um setpoint e um display na tela. Associe ambos ao mesmo tag interno.
11. Se o valor digitado no setpoint for diferente de 10, 20 ou 30, deve-se exibir uma mensagem ao usuário informando que este valor não é permitido. O valor não deve ser passado ao tag.

Eventos do E3Browser

OnDrawRow (Selected, Line, RowTextColor, RowBackColor)

Selected: indica se a linha está selecionada;
Line: indica o número da linha sendo desenhada;
RowTextColor: indica a cor do texto da linha;
RowBackColor: indica a cor de fundo do texto.
Se a cor for modificada dentro deste evento, esta modificação será usada pelo E3Browser no desenho da linha. Se o método GetColumnValue() for chamado de dentro do evento, os valores retornados serão os da linha sendo desenhada, e não os da linha selecionada.

Exemplo:

```
Estado = GetColumnValue(1)
If Estado=0 then
    RowTextColor = RGB(255,0,0) 'vermelho
Else
RowTextColor = RGB(255,0,0) 'verde
End if
```

Exercícios:

12. Insira um objeto Banco de Dados na aplicação e configure-o.
13. Crie um tag de demonstração.
14. Insira um alarme para o tag de demonstração.
15. Salve os alarmes no banco de Dados.
16. Exiba os dados de alarmes no E3Browser
17. Se o registro for de entrada de alarme, deve ter a cor vermelha. Se for um reconhecimento, cor azul. Se for um retorno, preto.

Eventos do E3Chart

OnLegendClick(Row, Col, RowData)

Ocorre quando o usuário clica em uma linha da legenda. Os parâmetros **Row** e **Col** indicam, respectivamente, a linha e a coluna clicadas. O parâmetro **RowData** é o índice da pena da legenda onde ocorreu o clique.

Exemplo:

```
Sub E3Chart1_OnLegendClick(Row, col, RowData)
MsgBox "Nome da pena: " & Pens.Item(RowData).name
End Sub
```

Exercícios:

18. Crie um gráfico na tela com pelo menos 3 penas. Mostre a legenda.
19. No evento OnStartRunning do E3Chart, insira o código abaixo para que todas as penas sempre apareçam:

```
Sub E3Chart1_OnStartRunning()
    Legend.ShowAllPens = True
End Sub
```

20. Ao clicar em uma pena na legenda, mostre um menu com as opções Visivel e Cor (Select Menu)
21. Ao clicar na opção Visível, a pena deve aparecer/desaparecer.
22. Ao clicar na opção Cor, deve abrir uma paleta de cores onde é possível escolher a nova cor da pena.

Comando For...Next

Repete um bloco de instruções um determinado número de vezes, do <ínicio> ao <fim>:

```
For <contador> = <início> To <fim>
<bloco de instruções>
Next
```

Exemplos:

```
'Zera os displays
For i=1 to 5
Screen.Item("Display"&i).Value = 0
next
```

Exercícios:

1. Crie três tags de demonstração.
2. Insira na tela um gráfico.
3. Ao clicar em um botão (Adicionar), cria uma pena para cada tag.
4. Ao clicar em um botão (Remover), remove todas as penas.

Comando For Each ... Next

Um laço For Each ... Next é similar a um laço For ... Next. Ao invés de repetir as instruções de um bloco um número específico de vezes, um laço For Each ... Next repete um grupo de comandos para cada item numa coleção. Esse comando é bastante útil quando não sabemos a quantidade de elementos.

```
For each <objeto> in <Colecao>
<bloco de instruções>
Next
```

Exemplos:

```
'Exibe o nome de todas as tags
For each tag in Application.GetObject("Dados")
MsgBox tag.Name
next
```

Exercícios:

5. Crie três tags de demonstração.
6. Insira na tela um gráfico.
7. Ao clicar em um botão (Adicionar), cria uma pena para cada tag.

Comando While...Wend

Executa um bloco de instruções enquanto uma determinada condição é verdadeira:

```
While <condição>
<bloco de instruções>
Wend
```

Exemplos:

```
'Soma 1 enquanto for menor que 10
While Tag.Value < 10
Tag.Value = Tag.Value + 1
Wend
```

Exercícios:

8. Crie na tela um retângulo horizontal para ser uma barra de progresso.
9. Faça a propriedade HorizontalPercentFill do retângulo variar de 0 a 100 dentro de um comando while. Dica: Para atualizar a tela, utilize o comando Refresh do Frame.

Lista de Seleção (ComboBox)

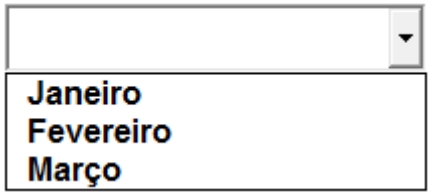


Figura 11

O objeto Lista de seleção exibe/esconde uma lista de itens configurados pelo usuário. Para preencher a lista utilizamos o evento OnStartRunning.

Exemplo:

```
Sub ComboBox1_OnStartRunning()
    'Preenche a ComboBox
    Clear()
    AddItem "Janeiro"
    AddItem "Fevereiro"
    AddItem "Março"
End Sub
```

Exercícios:

1. Crie uma ComboBox na Tela e preencha-a com todos os dias da semana.
2. Insira um objeto Banco de Dados no projeto e configure-o.
3. Crie 3 tags internos chamados: vermelho, verde e azul.
4. Insira um objeto Fórmula que guarde a quantidade de vermelho, verde e azul de uma cor. Use os tags internos para carregar e salvar os valores da formula. Cadastre pelo menos 4 cores.
5. Preencha a ComboBox com todas as cores cadastradas. (Utilize o método GetFormulaValueDataObj do Viewer)
6. Crie na tela um botão para inserir um novo valor na Fórmula. Escreva o nome da nova cor em um InputBox. O valor inserido deve aparecer na ComboBox.
7. Crie um botão para remover a cor selecionada na ComboBox da Fórmula. Somente a cor apagada deve ser removida da ComboBox.

Lista (ListBox)

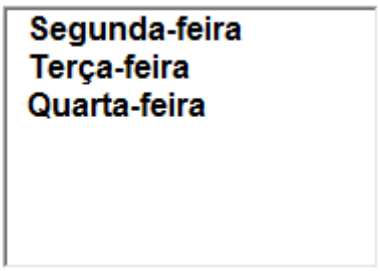


Figura 12

O objeto Lista é similar ao objeto ComboBox e exibe uma lista de itens configurados pelo usuário. Para preencher a lista utilizamos o evento OnStartRunning.

Exemplo:

```
Sub ListBox1_OnStartRunning()
    'Preenche a ListBox
    Clear()
    AddItem "Segunda-feira"
    AddItem "Terça-feira"
    AddItem "Quarta-feira"
End Sub
```

Exercícios:

1. Crie uma ListBox na Tela e preencha-a com todos os dias da semana.
2. Exiba em uma caixa de mensagem o nome do dia selecionado.

Noções de VBScript no Software Elipse E3: Lição 7 - GetADORecordSet

O método GetADORecordSet da consulta retorna um Recordset do tipo ADO (ActiveX Data Object), resultante da execução da consulta configurada.

O objeto ADO RecordSet é usado para acessar os registros de uma tabela do banco de dados. Possui as seguintes propriedades e métodos:

Propriedades:
BOF: Retorna verdadeiro se a posição do ponteiro é anterior ao primeiro registro.
EOF: Retorna verdadeiro se a posição do ponteiro é posterior ao último registro.
RecordCount: Retorna o número de registros na tabela.

Métodos:
MoveFirst: Move o ponteiro para o primeiro registro
MoveLast: Move o ponteiro para o último registro
MoveNext: Move o ponteiro para o próximo registro
MovePrevious: Move o ponteiro para o registro anterior

Para acessar os registros individualmente, utilizamos o comando Fields("NomeCampo").

Exemplo:

```
set RS = Screen.Item("Consulta1").GetADORecordset()
for i=1 to RS.RecordCount
    Campo1 = RS.Fields("Campo1").Value
    Campo2 = RS.Fields("Campo2").Value
    campo3 = RS.Fields("Campo3").Value
    MsgBox Campo1 &vbTab& Campo2 &vbTab& Campo3
next
```

Exercícios:

1. Insira um objeto Banco de Dados no projeto e configure-o.
2. Crie 3 tags de demonstração e grave-as no banco utilizando o objeto histórico. No histórico, configure o tempo de gravação para 0 segundos.
3. Insira um botão na tela que ao clicar grave um registro no histórico (WriteRecord)
4. Insira na tela um objeto consulta que busque os dados salvos pelo histórico.
5. Crie um botão na tela que ao clicar, mostre linha por linha do histórico. Utilize o comando While para repetir o script enquanto não chegar ao fim da tabela.

Algumas vezes é conveniente atribuir mais de um valor relacionado a uma única variável. Para isso, pode-se criar uma variável que contém uma série de valores, uma variável do tipo array, ou vetor.

Para declarar uma variável explicitamente, utiliza-se o comando Dim. A declaração de um array utiliza parênteses contendo sua dimensão.

Exemplo:

```
Dim A(10)
```

Pode-se atribuir dados a cada um dos elementos de um array usando-se um índice começando do zero e terminando no tamanho declarado (o número de elementos do array é sempre o número mostrado nos parênteses mais um).

Exemplo:

```
A(0) = 256
A(1) = 324
A(2) = 100
...
A(10) = 55
```

Também é possível criar um array utilizando o método Array(arglist) do VBScript. Este comando retorna uma variável do tipo Variant que contém um array. Os valores deverão ser separados por vírgula.

Exemplo:

```
A = Array(10,20,30)
MsgBox A(0)
MsgBox A(1)
MsgBox A(2)
```

Split(expression, [delimiter], [count], [compare])

Retorna um array unidimensional baseada em zero, contendo um número especificado de substrings.
Expression: Obrigatório. Expressão de String contendo subseqüências e delimitadores.
Delimiter: Opcional. Qualquer caractere singular usado para identificar limites subseqüência de caracteres. Se Delimiter for omitido, o caractere de espaço (" ") é assumido como o delimitador.
Count: Opcional. Número máximo de substrings no qual a seqüência de caracteres de entrada deve ser dividida. O padrão, - 1, indica que a seqüência de caracteres de entrada deve ser dividida em cada ocorrência da seqüência Delimiter.
Compare: Opcional. Valor numérico indicando a comparação para usar ao avaliar substrings.

Exemplo:

```
MyString = "Os que muito falam, pouco fazem de bom"
MyArray = Split(MyString, " ")
' MyArray(0) contem "Os"
' MyArray(1) contem "que"
' MyArray(2) contem "muito"
' MyArray(3) contem "falam,"
' MyArray(4) contem "pouco"
' MyArray(5) contem "fazem"
' MyArray(6) contem "de"
' MyArray(7) contem "bom."
for i=0 to 7
    MsgBox MyArray(i)
Next
```

UBound(arrayname)

Retorna a dimensão do array (arrayname).

Exemplo:

```
Dim MyArray(3)
MsgBox UBound(MyArray)
```

Aparece uma caixa de mensagem com o valor 3.

Configuração de Driver em execução

Para modificar em execução qualquer propriedade do dialog do driver de comunicação deve ser criado um array com os parâmetros que deseja alterar. Os mais comuns são:

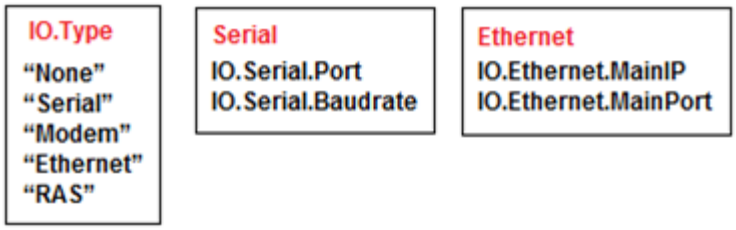


Figura 13

Exemplo:

```
Dim arr(6)
'configura os elementos do array
arr(1) = "IO.Type"
```

```
arr(2) = "Serial"
arr(3) = "IO.Serial.Port"
arr(4) = 1
arr(5) = "IO.serial.BaudRate"
arr(6) = 19200
```

Um jeito mais elegante de escrever seria:

```
Dim arr(3)
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.serial.BaudRate", 19200)
```

Exercícios:

1. Ao clique de um botão, crie um array com todos os dias da semana. Guarde o array em um tag interno.
2. Ao clicar em um botão, leia cada posição do array criado no exercício anterior e exiba cada um em uma caixa de mensagem. Para saber o tamanho do array, use o comando UBound.
3. Insira uma lista (Listbox). No OnStartRunning digite o seguinte script:

```
'Preenche a ListBox
    Clear()
    AddItem "Manhã,Tarde,Noite"
    AddItem "violeta;anil;azul;verde;amarelo;laranja;vermelho"
    AddItem "1;2;3;4"
```

4. Crie um botão que separe cada palavra e exiba-as em uma caixa de mensagem. Cuidado que cada linha contém uma quantidade diferente de palavras.
5. Escreva o array necessário para configurar o driver com os seguintes parâmetros:

a) Ethernet / 127.0.0.1 / Porta 502
b) Serial / COM1 / Baudrate 19200

AddObject(strClassName, bActivate)

O método AddObject() adiciona um novo objeto à aplicação.
strClassName: tipo de objeto que será criado.
bActivate: indica se o objeto será ativado após a criação.
Quando o objeto estiver ativado, os links e os scripts ficam habilitados. Se o objeto for criado com bActivate em False, mais tarde ele pode ser ativado pelo método Activate().

Exemplo:

```
set retangulo = Screen.AddObject("DrawRect", True)
retangulo.X = 200
retangulo.Y = 200
retangulo.ForegroundColor = vbRed
```

With...End With

Múltiplas propriedades do mesmo objeto podem ser alteradas utilizando o With..End With. O código é executado mais rapidamente já que o objeto é referenciado apenas uma vez.

Exemplo:

```
With Screen.Item("Retangulo1")
    .BackgroundColor = vbRed
    .BorderColor = vbBlack
    .Effect3D = 1
    .Visible = True
End With
```

TypeName(varname)

Retorna um valor String contendo informações sobre uma variável de tipo de dados. O tipo do objeto pode ser visto na lista de propriedades.

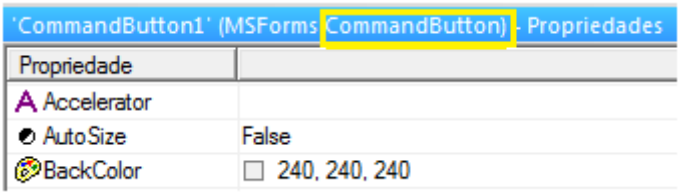


Figura 14

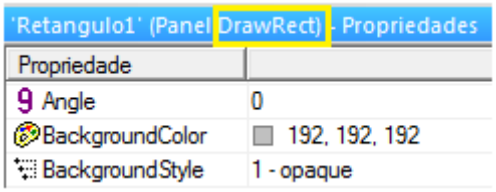


Figura 15

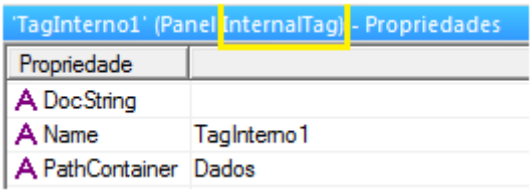


Figura 16

Exemplo:

```
set Tag = Application.GetObject("Dados.TagInterno1")
MsgBox TypeName(Tag)
'Será exibida uma caixa de mensagem com o texto "InternalTag"
set Botao = Screen.Item("CommandButton1")
MsgBox TypeName(Botao)
'Será exibida uma caixa de mensagem com o texto "CommandButton"
```

Exercícios:

1. Ao clicar na tela, crie um retângulo azul na posição do clique.
2. Crie um botão que crie um tag interno e um tag demo na pasta Dados em execução.
3. Crie uma associação por tabela por script da cor do retângulo com o tag demo obedecendo à seguinte tabela de cores:

0	25	Verde
25	50	Azul
50	75	Amarelo
75	100	Vermelho

4. Crie uma lista de seleção (ComboBox). Essa lista deve exibir o nome de todos os tags internos existentes na pasta Dados. A atualização da lista pode ser feita em um botão (Atualizar). Para diferenciar os tags demo e interno, utilize o comando TypeName.
5. Ao selecionar um tag na lista de seleção, um setpoint deve ser criado na tela e associado ao tag selecionado. Use a coordenada X=0 e Y varia de acordo com a quantidade de setpoints já criados.
6. Ao clicar o botão direito na tela, um menu com o nome de todos os setpoints existentes na tela deve ser listado. Para saber se o objeto da tela é um setpoint, utilize o comando TypeName.
7. Ao selecionar um setpoint no menu da tela apagar o objeto. Exibir uma mensagem de confirmação antes de apagar.

Noções de VBScript no Software Elipse E3: Lição 10 - Procedimentos e Funções

Procedimentos são linhas de código, usadas para responder a eventos ou executar algumas ações. Os procedimentos não retornam valores e são identificados pela palavra Sub na sua declaração. Todos os procedimentos devem começar com uma letra, e podem conter letras, números e o caractere underscore.

Funções são usadas quando se pretende executar cálculos ou testar valores e retorna sempre algo. Um exemplo é somar dois valores e retornar o valor da soma.

Exemplo:

```
Sub CommandButton1_Click()
    'Function soma
    resposta = Soma(10,20)
    MsgBox resposta

    'Sub exibir mensagem
    ExibirMensagem("Mensagem a ser exibida")
End Sub

Function Soma(a,b)
    Soma = a + b
End Function

Sub ExibirMensagem(mensagem)
    MsgBox mensagem
End Sub
```

Exercícios:

1. Crie um botão que contenha uma Function que some, multiplique e divida dois números e exiba cada resposta em uma caixa de mensagem.
2. Crie a seguinte estrutura de alarmes:

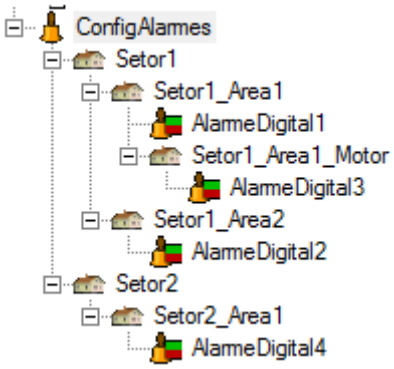


Figura 17

3. Associe cada alarme digital a um tag interno. Para alterar o valor do tag interno, utilize na tela o ToggleButton.
4. Crie um botão que exiba uma caixa de mensagem para todos os objetos de alarmes. A mensagem deve informar se é uma área ou alarme digital e o nome do objeto. Dica: para cada objeto, verifique o TypeName. Crie uma Sub para objetos do tipo área e outra sub para objetos do tipo fonte.