

**Tópicos desta Aula**

1. Caixas de verificação e botões de opção .....	2
2. Caixas de verificação .....	2
3. Botões de opção .....	3
4. Trabalhando com CheckedListBox .....	6
5. Adicionando novos itens .....	7
6. Trabalhando com Caixas de Listas.....	9
7. Controle ListBox.....	9
8. Controle ComboBox .....	10
9. Exemplo Caixa de Lista .....	11
10. Adicionando, excluindo e limpando a lista via programação.....	12
11. Métodos para remover e limpar a lista .....	14
12. Avaliação da aula .....	15

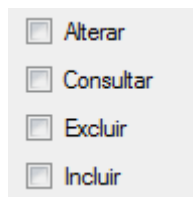
### Caixas de verificação e botões de opção

Quando você cria a interface do usuário para seu programa, você geralmente precisa de uma maneira para apresentar opções para que o usuário faça a escolha.

Por exemplo, em um sistema para controle de estoque, alguns usuários poderão apenas consultar o estoque, outros além de consultar poderão alterar, excluir ou incluir novos produtos.

### Caixas de verificação

O controle CheckBox fornece uma representação visual que facilita criar essa opção.



O controle CheckBox consiste em uma caixa e um rótulo de texto que o usuário pode selecionar. Quando o usuário clica na caixa, uma marca de seleção aparece na caixa.

Se a caixa for clicada novamente, a marca de seleção será removida.

O status da caixa de seleção pode ser recuperado usando a propriedade **Checked**.

Se a caixa exibe uma marca de seleção, a propriedade retornará True.

Se nenhuma seleção for exibida, a propriedade retorna False.

### Exemplo

```
14 Dim permissoes As String = ""
15 If (chkAlterar.Checked) Then
16     permissoes = permissoes + chkAlterar.Text + " "
17 End If
18 If (chkConsultar.Checked) Then
19     permissoes = permissoes + chkConsultar.Text + " "
20 End If
21 If (chkExcluir.Checked) Then
22     permissoes = permissoes + chkExcluir.Text + " "
23 End If
24 If (chkIncluir.Checked) Then
25     permissoes = permissoes + chkIncluir.Text + " "
26 End If
27 lblPermissoesAtribuidas.Text = "Permissões: " + permissoes
```

Observe no código acima que basta utilizar um If avaliando a propriedade Checked.

#### Botões de opção

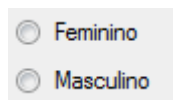
Você acabou de aprender como permitir que um usuário escolha qualquer uma das várias opções.

Mas e se você deseja que o usuário escolha apenas uma das várias opções? Nesse caso, você pode usar o controle **RadioButton**.

Ao contrário das caixas de seleção, botões de opção sempre funcionam como parte de um grupo. Selecionar um botão de opção imediatamente limpa todos os outros botões de opção no grupo.

Você pode usar grupos de controles RadioButton para permitir que os usuários escolham entre opções exclusivas.

Por exemplo pode ser utilizado para definir qual é o sexo do usuário.

A screenshot of a Windows form showing two radio buttons. The first radio button is selected and is labeled 'Feminino'. The second radio button is unselected and is labeled 'Masculino'.

Da mesma forma que o CheckBox, você pode verificar a propriedade Checked.

```
29 Dim sexo As String = ""
30 If (rbFeminino.Checked) Then
31     sexo = "Feminino"
32 End If
33 If (rbMasculino.Checked) Then
34     sexo = "Masculino"
35 End If
36 lblSexo.Text = "Sexo: " + sexo
```

#### Exemplo

Crie o seguinte formulário

## Visual Basic 2008 - Módulo Básico

### Aula 06

**Incluir Usuário**

**Incluir novo Usuário**

Nome do Usuário  
Sergio Kadluba

Incluir

**Permissões**

☒ Alterar  
☒ Consultar  
☐ Excluir  
☐ Incluir

**Sexo**

☐ Feminino  
☒ Masculino

Sergio Kadluba incluído.  
Permissões: Alterar Consultar  
Sexo: Masculino

Fechar

Código fonte do botão Incluir

## Visual Basic 2008 - Módulo Básico

### Aula 06

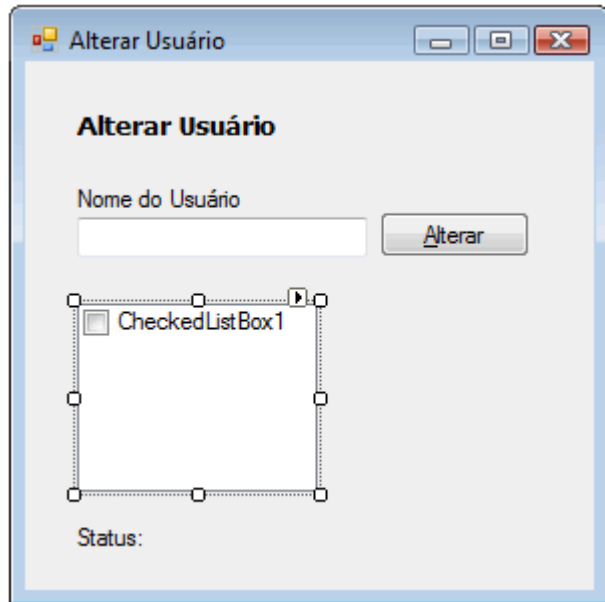
```
Private Sub cmdIncluir_Click(ByVal sender As System.Object, ByVal  
    Dim status As String = ""  
    Dim permissoes As String = ""  
    Dim sexo As String = ""  
  
    status = txtNomeUsuario.Text + " incluído."  
    lblStatus.Text = status  
  
    If (chkAlterar.Checked) Then  
        permissoes = permissoes + chkAlterar.Text + " "  
    End If  
    If (chkConsultar.Checked) Then  
        permissoes = permissoes + chkConsultar.Text + " "  
    End If  
    If (chkExcluir.Checked) Then  
        permissoes = permissoes + chkExcluir.Text + " "  
    End If  
    If (chkIncluir.Checked) Then  
        permissoes = permissoes + chkIncluir.Text + " "  
    End If  
    lblPermissoesAtribuidas.Text = "Permissões: " + permissoes  
  
    If (rbFeminino.Checked) Then  
        sexo = "Feminino"  
    End If  
    If (rbMasculino.Checked) Then  
        sexo = "Masculino"  
    End If  
    lblSexo.Text = "Sexo: " + sexo  
  
End Sub
```

### Trabalhando com CheckedListBox

O controle CheckedListBox contém uma coleção de itens, onde cada item representará uma caixa de verificação.

#### Exemplo

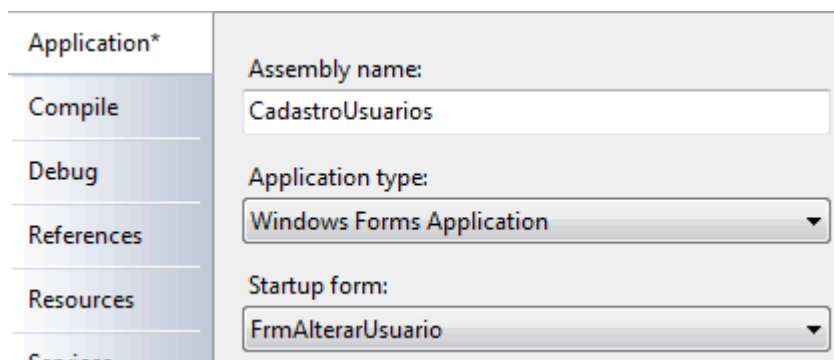
Crie o formulário abaixo e adicione um CheckedListBox conforme a figura abaixo



Altere a propriedade name do controle CheckedListBox para clbPermissoes

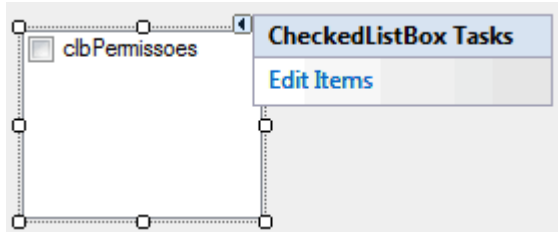
Altere o Start up Form para este formulário.

Caso tenha esquecido como se faz, clique com o botão direito do mouse sobre o nome do teu projeto dentro do Solution Explorer e selecione Properties



## Adicionando novos items

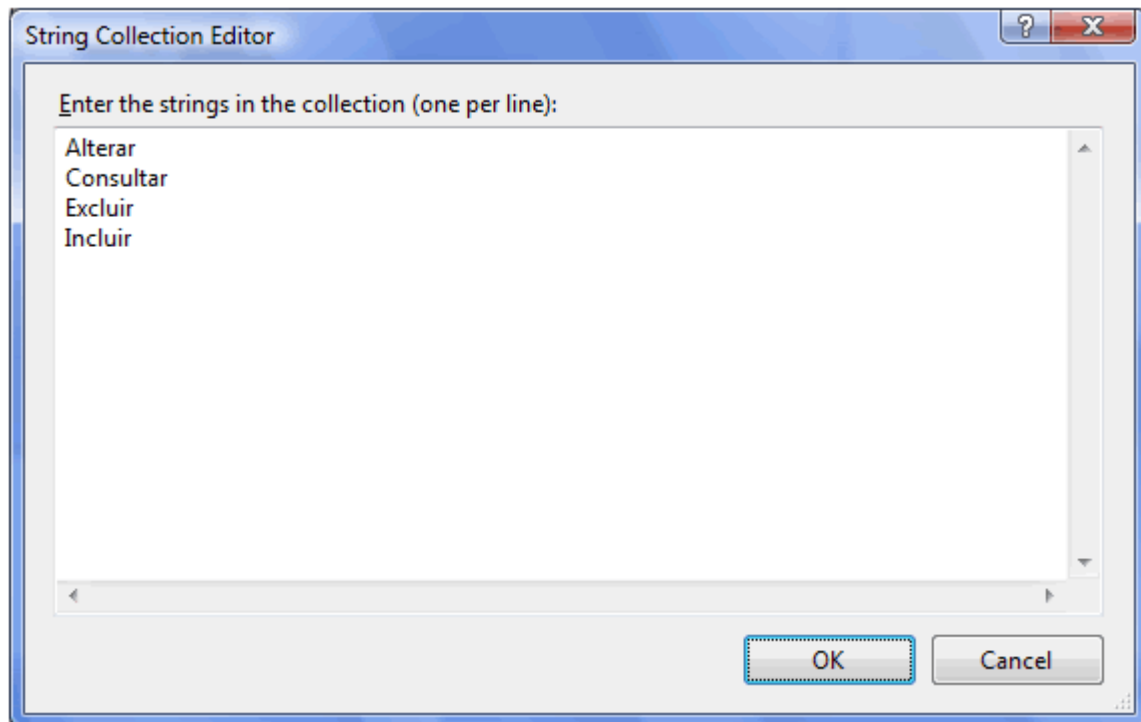
Para adicionar novos items, clique na seta, na parte superior direita do controle para que o menu de tarefas seja aberto, conforme a figura abaixo.



Clique na opção **Edit Items**

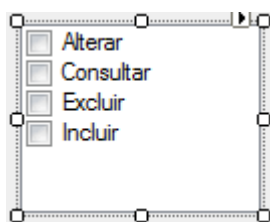
A janela **String Collection Editor** será aberta.

Adicione alguns itens conforme a figura abaixo



Clique em OK

Observe que cada item adicionado equivale à um CheckBox



Você pode alterar as propriedades BackColor e BorderStyle se preferir

## Visual Basic 2008 - Módulo Básico

### Aula 06

BackColor	<input type="checkbox"/>	ButtonFace
BorderStyle		None

Para obter os itens selecionados basta fazer um loop até o total de itens selecionados, subtrair 1 pois a coleção começa a partir do zero e depois obter o valor dos itens marcados.

#### Exemplo

```
lblStatus.Text = "Permissões: "  
For i = 0 To clbPermissoes.CheckedItems.Count - 1  
    lblStatus.Text = lblStatus.Text + clbPermissoes.CheckedItems(i) + " "  
Next
```



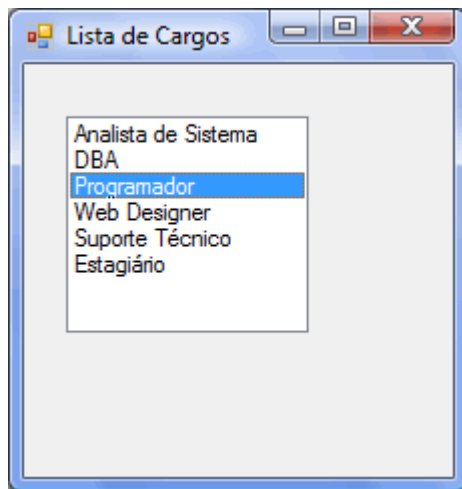
## Trabalhando com Caixas de Listas

Quando você deseja fornecer aos usuários uma lista de opções, você pode exibir a lista de itens em um controle **ListBox** ou em um controle **ComboBox**.

### Controle ListBox

Um controle ListBox (caixa de lista) permite que você exiba vários itens ao mesmo tempo, permitindo que os usuários percorram esta lista para selecionar o item desejado.

Quando um usuário seleciona um item, ele fica realçado na lista, como mostra a imagem abaixo

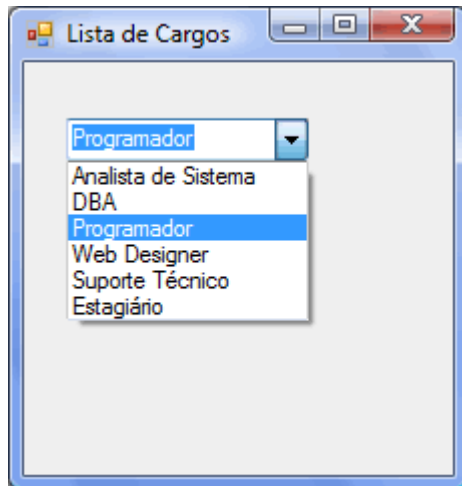


#### Controle ComboBox

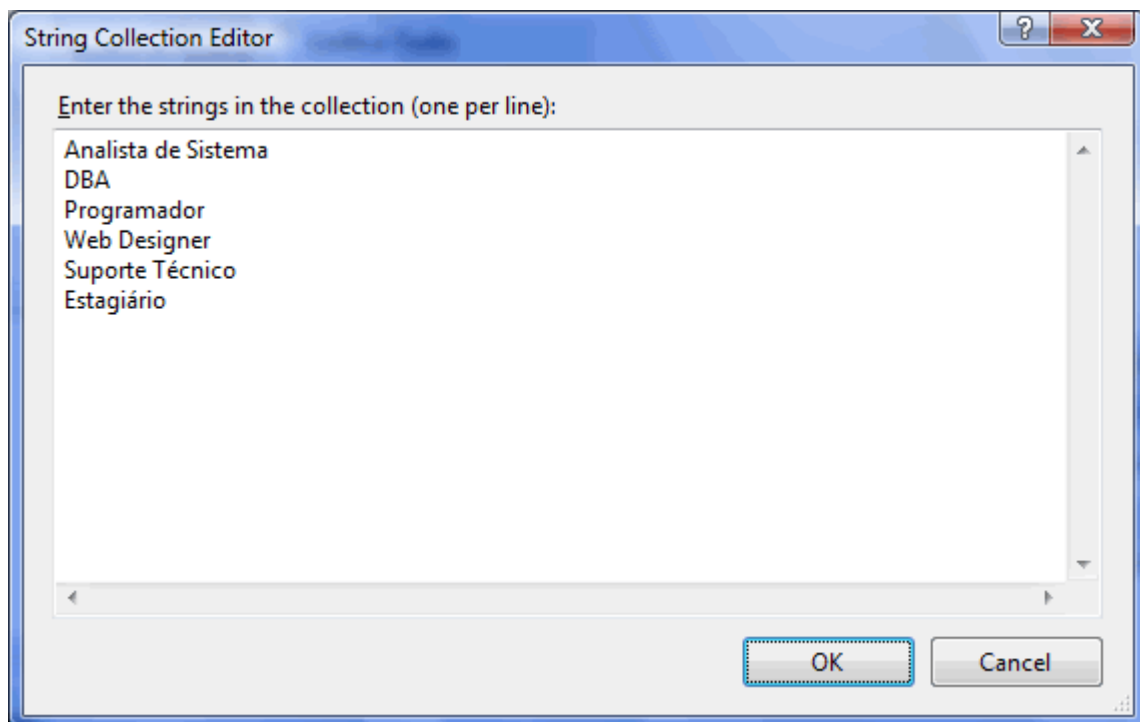
Um controle ComboBox é uma combinação de uma caixa de texto e uma caixa de lista.

Por padrão, a caixa de combinação é exibida como um caixa de texto, mas quando os usuários clicam na seta a direita, é exibida uma lista.

Quando um usuário seleciona um item, ele fica realçado e está visível na exibição padrão, como mostra a imagem abaixo.



O processo para adicionar itens a caixas de lista e caixas de combinação são semelhantes. Você pode utilizar o String Collection Editor para adicionar os itens.

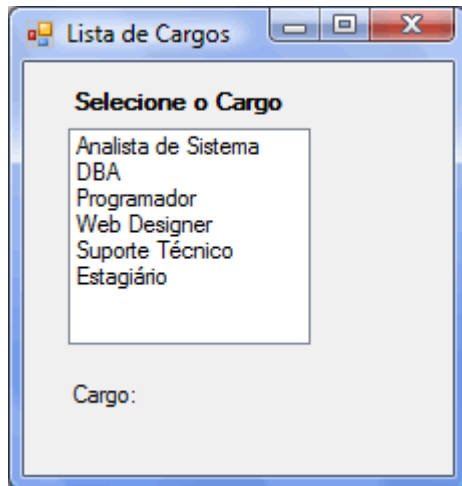


## Exemplo Caixa de Lista

Crie um novo formulário denominado **FrmListaCargos**

Configure este formulário como o formulário inicial (StartupForm)

Adicione os controles conforme a figura abaixo



Altere o Text do formulário para **Lista de Cargos**

Altere o nome do ListBox para **lbCargo**

Altere o nome do Label superior para **lblDescrição** e o Text para **Selecione o Cargo**

Altere o nome do Label inferior para **lblCargo** e o Text para **Cargo:**

De um duplo clique na caixa de lista para criar o procedimento de evento

Para obter o item selecionado basta utilizar a propriedade SelectedItem

```
1 Public Class FrmListaCargos
2
3     Private Sub lbCargo_SelectedIndexChanged(ByVal sender As Object, ByVal e As EventArgs) Handles lbCargo.SelectedIndexChanged
4         lblCargo.Text = "Cargo:" + lbCargo.SelectedItem
5     End Sub
6
7 End Class
```

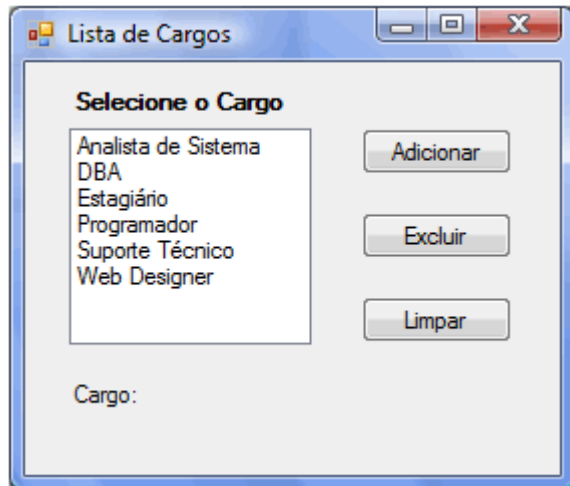
### Observação

O evento SelectedIndexChanged ocorre sempre que o item selecionado é alterado

## Adicionando, excluindo e limpando a lista via programação

Criando o formulário

Adicione os botões conforme a figura abaixo



Altere os nomes para **cmdAdicionar**, **cmdExcluir** e **cmdLimpar**

Altere a propriedade **Sorted** do ListBox para **True**  
A lista ficará ordenada.

### Método Add

Para adicionar um item via programação utilize o método **Add**

Exemplo

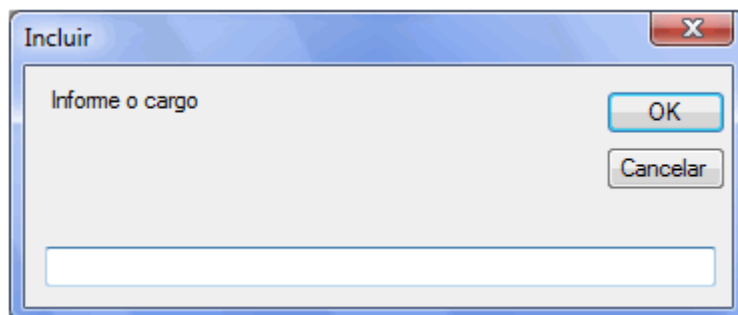
```
7 Private Sub cmdAdicionar_Click(ByVal sender As System.Object, ByVal e As EventArgs)
8     Dim cargo As String
9     cargo = InputBox("Informe o cargo", "Incluir")
10    lbCargo.Items.Add(cargo)
11 End Sub
```

Ao clicar no botão **Adicionar**

Na linha 8 declaramos a variável cargo como do tipo String

Na linha 9 atribuímos a variável cargo o retorno da função InputBox.

Esta função chama uma janela para a entrada de dados, sendo o primeiro parâmetro é a mensagem e o segundo parâmetro é o título



Você digita o valor na sua caixa de texto e clica em **OK**

O valor digitado será atribuído a variável cargo.

Se você clicar em **Cancelar** a janela do InputBox retornará uma string em branco

Na linha 10 adicionamos o conteúdo da variável cargo para a caixa de lista.

Observe que mesmo clicando em cancelar uma linha em branco esta sendo adicionada

Você poder corrigir este problema verificando se a variável cargo é diferente de uma string vazia

Exemplo

```
7 | Private Sub cmdAdicionar_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles cmdAdicionar.Click
8 |     Dim cargo As String
9 |     cargo = InputBox("Informe o cargo", "Incluir")
10 |     If cargo <> "" Then
11 |         lbCargo.Items.Add(cargo)
12 |     End If
13 | End Sub
```

### Métodos para remover e limpar a lista

#### Método Remove

Para remover um item via programação utilize o método **Remove**

Exemplo

```
15 Private Sub cmdExcluir_Click(ByVal sender As Sy
16     lbCargo.Items.Remove(lbCargo.SelectedItem)
17 End Sub
```

Observe que estamos mandando remover o item selecionado

#### Método Clear

Para remover todos os itens via programação utilize o método **Clear**

Exemplo

```
19 Private Sub cmdLimpar_Click
20     lbCargo.Items.Clear()
21 End Sub
```

Estes métodos são exatamente os mesmos para a caixa combo.

#### Verificando se um item na lista existe

Quando você adiciona itens a uma lista, geralmente não deseja duplicar um item existente. Você pode usar o método **Contains** para determinar se o item já está na caixa de combinação ou na caixa de listagem.

Exemplo

```
7 Private Sub cmdAdicionar_Click(ByVal sender As System.Object, I
8     Dim cargo As String
9     cargo = InputBox("Informe o cargo", "Incluir")
10    If Not lbCargo.Items.Contains(cargo) And cargo <> "" Then
11        lbCargo.Items.Add(cargo)
12    End If
13 End Sub
```

Observe que no If da linha 10 estamos verificando duas condições utilizando o operador **And**:

- 1- Se na lista não contem um valor igual ao contido na variável cargo e
- 2- Se o conteúdo da variável cargo não esta vazio.

### **Avaliação da aula**

Para a avaliação desta aula e para que a próxima aula seja liberada, você deve enviar o projeto contendo os exemplos desta aula, compactado, para o email do seu tutor.

O prazo para correção e conseqüente liberação da próxima aula é de até 24 horas.

#### **Observação**

Alguns provedores tem barrado emails que contenham anexos com arquivos compactados que contenham executáveis dentro (.exe ou .dll)

Sugerimos que para evitar problemas renomeie a extensão do arquivo de .zip ou .rar para .txt