Visual Basic .NET

Menus, Built-in Dialog Boxes e Printing

Professor: Danilo Giacobo

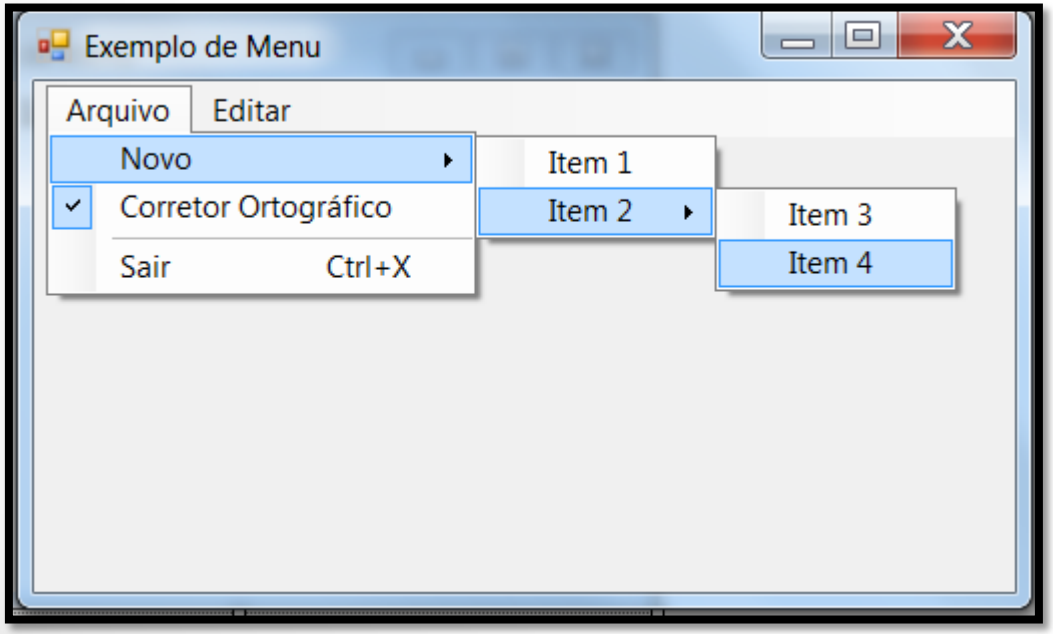
Página pessoal: www.danilogiacobo.eti.br
E-mail: danilogiacobo@gmail.com

Objetivos da aula

- ✓ Trabalhar com o controle **MenuStrip**
- ✓ Trabalhar com o controle **ContextMenuStrip**
- ✓ Trabalhar com o controle **OpenFileDialog**
- ✓ Trabalhar com o controle **SaveFileDialog**
- ✓ Trabalhar com o controle **FontDialog**
- ✓ Trabalhar com o controle **ColorDialog**
- ✓ Trabalhar com o controle **PrintDocument**
- ✓ Trabalhar com o controle **PrintDialog**
- ✓ Trabalhar com o controle **PrintPreviewControl**
- ✓ Trabalhar com o controle **PageSetupDialog**

Menus

- Todo usuário do sistema operacional Windows tem familiaridade com menus.
- O principal controle de menu do VB .NET é o **MenuStrip**.
- Cada item do menu é um objeto do tipo **ToolStripMenuItem**.
- Você pode criar menus e sub-menus, mostrar marcações, criar separadores, atribuir teclas de atalho e até mudar a aparência do mesmo.



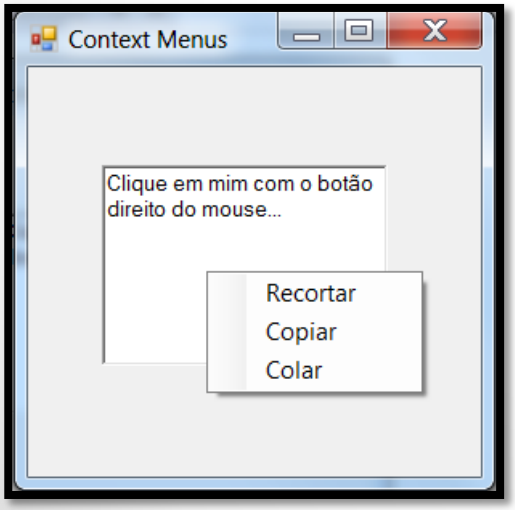
Na pasta:
MenuStrip.sln

Menu Items

- Os itens do menus tais como **Arquivo** e **Editar** do slide anterior pertencem a classe **ToolStripMenuItem**.
- Ele permite inclusive a criação de check boxes, text boxes, combo boxes e separadores no menu.
- A propriedade **Checked** controla a marcação que fica ao lado do item.
- A propriedade **ShortcutKeys** permite que você atribua uma tecla de atalho para um item de menu. Para ela aparecer no menu a propriedade **ShowShortcutKeys** deve estar como **True**.
- A propriedade **Text** configura o texto exibido pelo item do menu. Colocando um "-" você transforma ele em um separador. Inserindo um "&" no texto do menu você coloca a tecla de acesso do mesmo.
- Você pode habilitar e desabilitar itens de menu usando a propriedade **Enabled**.
- Você esconder e mostrar ele usando a propriedade **Visible**.
- O evento mais importante desta classe é o **Click**.

Context Menus

- Tipo de menu bastante popular na plataforma Windows.
- Ele é utilizado para mostrar as opções mais usadas pela pessoa.
- Geralmente é mostrado por meio do clique do botão direito do mouse.
- Você associa este menu a outro controle usando a propriedade **ContextMenuStrip**.
- Você pode adicionar os itens ao menu usando a propriedade **Items**.
- Eles podem ser desabilitados, escondidos e deletados (métodos **Show** e **Hide**).
- O evento mais importante desta classe é o **Click**.



Na pasta:
Context_Menus.sln

As Caixas de Diálogo embutidas

- O VB .NET disponibiliza várias caixas de diálogo para as mais variadas funções comumente usadas em aplicações visuais.
- Os tipos de caixas de diálogos mais interessantes são:

- ✓ **Open File dialogs**
- ✓ **Save File dialogs**
- ✓ **Font dialogs**
- ✓ **Color dialogs**
- ✓ **Print Preview dialogs**
- ✓ **Page Setup dialogs**
- ✓ **Print dialogs**

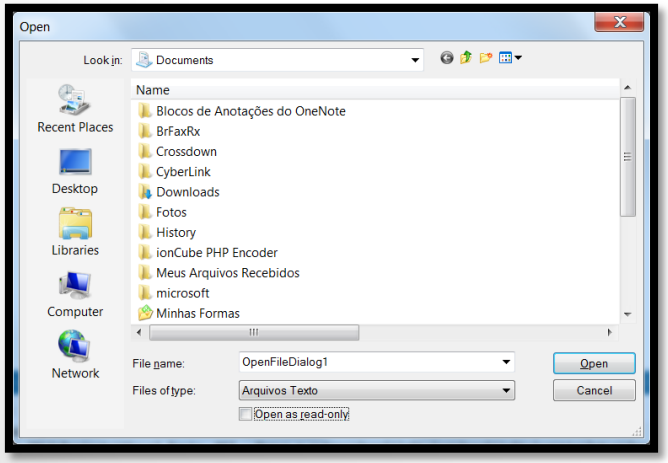
As Caixas de Diálogo embutidas

- Você usa o método **ShowDialog** para apresentar a mesma em tempo de execução e pode verificar seu valor de retorno usando a enumeração **DialogResult** que possui os seguintes valores:

- **Abort**
- **Cancel**
- **Ignore**
- **No**
- **None**
- **OK**
- **Retry**
- **Yes**

Open File Dialogs

- Permite ao usuário selecionar um arquivo para ser aberto.
- É a mesma caixa de diálogo usado pelo próprio Windows.
- O nome da classe é **OpenFileDialog**.
- Você pode deixar a pessoa selecionar vários arquivos configurando a propriedade **Multiselect** para **True**.
- A propriedade **ShowReadOnly** configura um checkbox de apenas leitura.
- A propriedade **ReadOnlyChecked** indica se o checkbox foi selecionado.
- A propriedade **Filter** configura as opções de arquivo que a pessoa pode selecionar.
- O nome e o caminho do arquivo selecionado ficam armazenados na propriedade **FileName**.
- Você pode usar o método **OpenFile** para abrir diretamente o arquivo.

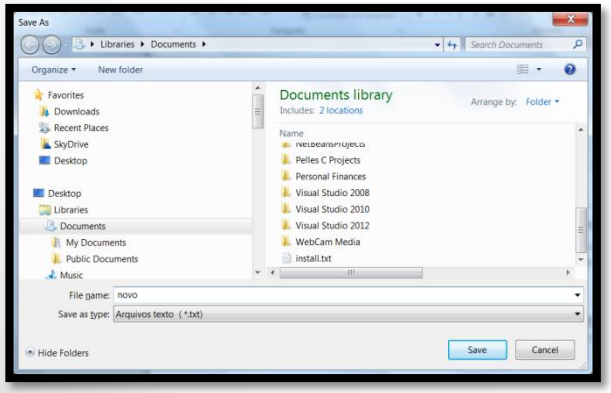


Na pasta:
OpenFileDialog.sln

Save File Dialogs

- A classe desse controle se chama **SaveFileDialog**.
- Esta caixa de diálogo permite ao usuário especificar o nome de um arquivo para salvar dados nele.
- Este diálogo é o mesmo que é utilizado pelo Windows.
- O método **ShowDialog** é usado para mostrar ele em tempo de execução.
- Você pode usar a propriedade **FileName** para recuperar o arquivo que a pessoa selecionou e abrir um arquivo em modo de leitura/escrita com o método **OpenFile**.

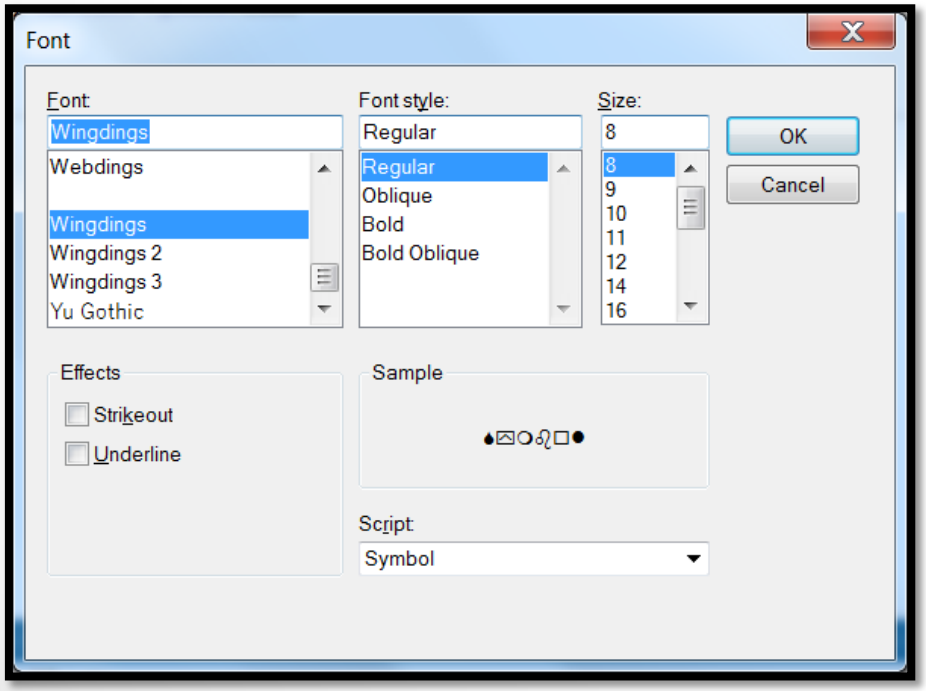
Dica: As propriedades **CheckFileExists** e **CheckPathExists** permitem verificar se um arquivo ou diretório já existem ou se eles devem ser criados.



Na pasta:
SaveFileDialog.sln

Font Dialogs

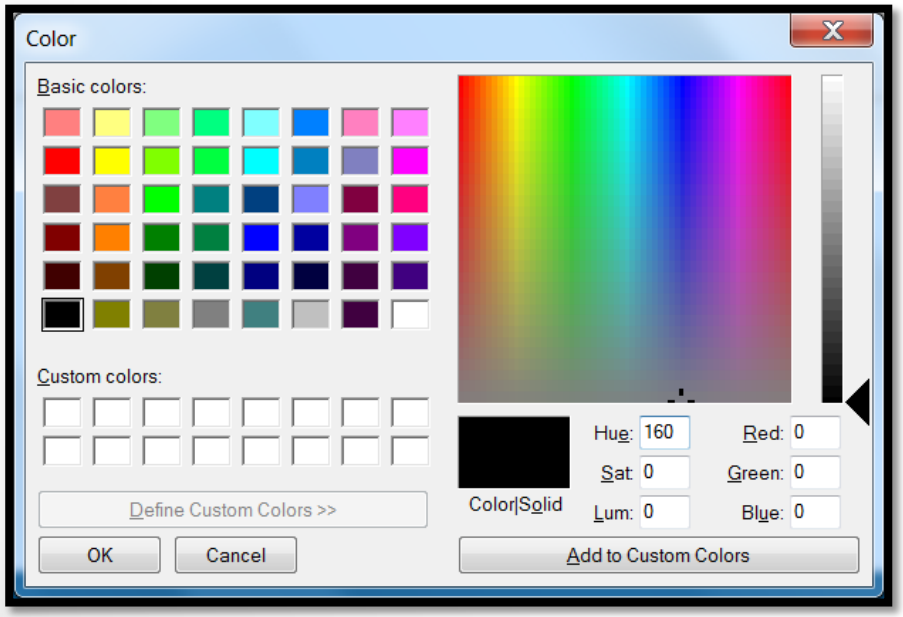
- Este tipo de diálogo permite que a pessoa selecione o tamanho, tipo, cor, estilo e demais elementos relacionados a uma fonte de texto.
- Eles retornam objetos **Font** e **Color** diretamente para você facilmente trabalhar com eles em código.
- O método **ShowDialog** é usado para mostrar ele em tempo de execução.



Na pasta:
FontDialog.sln

Color Dialogs

- Permitem que o usuário escolha uma cor de uma forma fácil.
- A principal propriedade que você usará para este tipo de diálogo é a **Color** que retorna um objeto de mesmo tipo, pronto para uso.
- Você pode usar a propriedade **AllowFullOpen** para desabilitar o botão de customização de cores (False).
- O método **ShowDialog** é usado para mostrar ele em tempo de execução.



Na pasta:
ColorDialog.sln

Imprimindo documentos

- O componente **PrintDocument** é o objeto que controla a impressão de documentos em um form.
- O método **Print** inicia a impressão de um documento.
- A propriedade **PrinterSettings** contém as configurações da impressão de um documento.

Exemplo:

```
1  Public Class Form1
2      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3          PrintDocument1.DocumentName = "Arquivo de Teste"
4          PrintDocument1.PrinterSettings.PrintFileName = "C:\textosalvo.rtf"
5          PrintDocument1.PrinterSettings.PrinterName = "Brother MFC-8952DW Printer"
6          PrintDocument1.Print()
7      End Sub
8  End Class
```

Na pasta:
PrintDocument.sln

Imprimindo documentos com a classe Process

- Para realizar a simples impressão de um documento você pode usar a classe **Process** configurando algumas propriedades da mesma.

Exemplo:

```
1  Public Class Form1
2      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3          Dim print As New Process()
4          With print
5              .StartInfo.CreateNoWindow = True
6              .StartInfo.Verb = "print"
7              .StartInfo.FileName = "D:\Documentos\Ukelele\Cifras\somewhere.pdf"
8              .Start()
9              .Close()
10         End With
11     End Sub
12 End Class
```

Na pasta:
Print_PDF.sln

Imprimindo documentos com o Shell

- Com a função **Shell** é possível executar comandos externos do sistema operacional passando inclusive parâmetros para o mesmo.
- O programa abaixo cria um arquivo texto e imprime o mesmo chamando o aplicativo **Notepad** e acessando a opção de impressão do mesmo.

Exemplo:

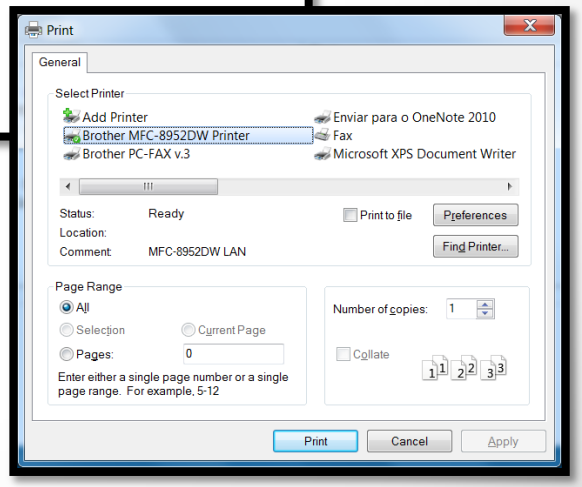
```
1  Imports System.IO
2
3  Public Class Form1
4      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
5          Dim strTexto As String
6
7          strTexto = Chr(9) & "Meu Texto 1" & vbCrLf
8          strTexto = strTexto & Chr(9) & "Meu Texto 2" & vbCrLf
9          strTexto = strTexto & Chr(9) & "Meu Texto 3" & vbCrLf
10         strTexto = strTexto & Chr(9) & "Meu Texto 4" & vbCrLf
11         strTexto = strTexto & Chr(9) & "Meu Texto 5" & vbCrLf
12         strTexto = strTexto & Chr(9) & "Meu Texto 6" & vbCrLf
13
14         Dim strCaminho As String = "\tprint.txt"
15
16         File.WriteAllText(strCaminho, strTexto)
17
18         Dim strComando As String
19         strComando = "start /min notepad /P " + strCaminho
20         Shell("cmd.exe /c " & strComando, AppWinStyle.Hide)
21     End Sub
22 End Class
```

Na pasta:
Print_TXT.sln

Print Dialogs

- Permitem que a pessoa imprima documentos.
- Você precisa adicionar um objeto **PrintDocument** ao projeto e associar ele ao controle **Print Dialog** usando a propriedade **Document**. O mesmo deve ser feito com a propriedade **PrinterSettings**.

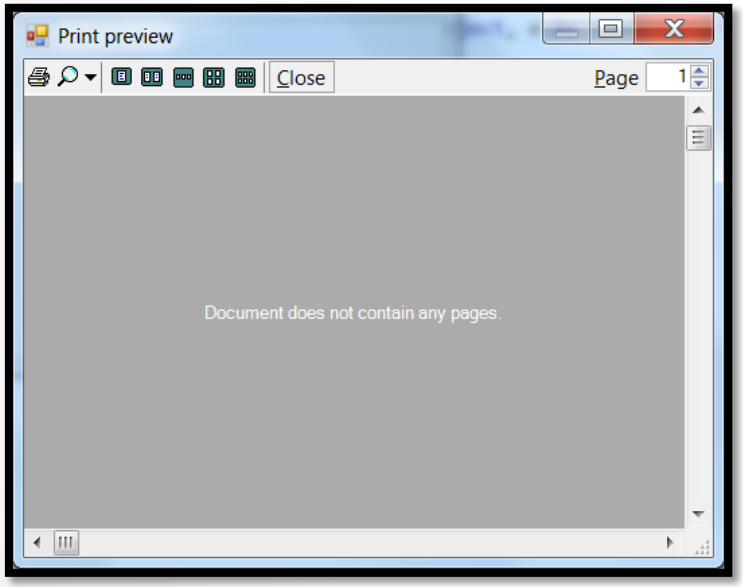
```
1  Public Class Form1
2      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3          PrintDialog1.Document = PrintDocument1
4          PrintDialog1.PrinterSettings = PrintDialog1.PrinterSettings
5          PrintDialog1.AllowSomePages = True
6
7          If PrintDialog1.ShowDialog = DialogResult.OK Then
8              PrintDialog1.PrinterSettings = PrintDialog1.PrinterSettings
9              PrintDocument1.Print()
10         End If
11     End Sub
12 End Class
```



Na pasta:
PrintDialogs.sln

Print Preview Dialogs

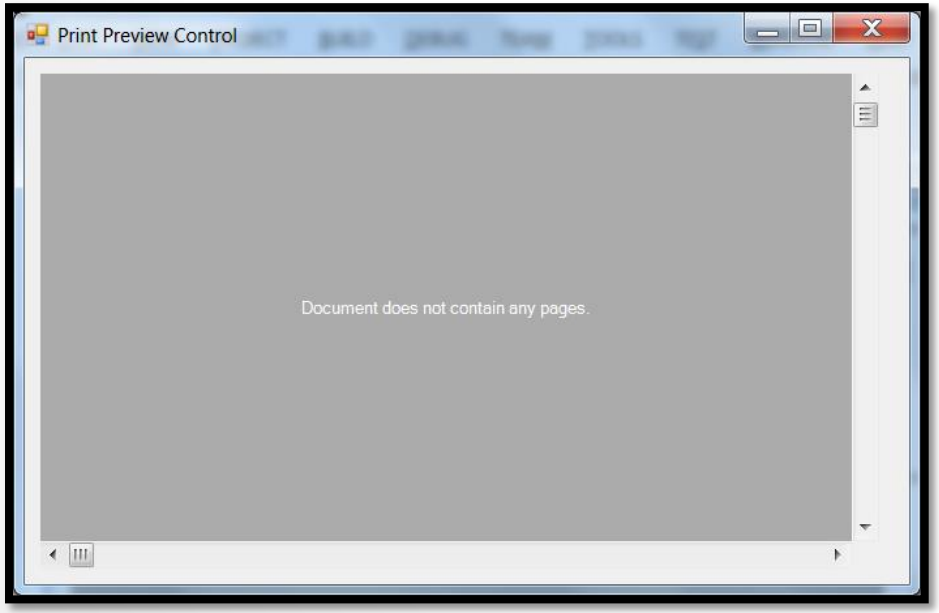
- Permitem que a pessoa veja como será o documento quando este for impresso.
- Você precisa adicionar um objeto **PrintDocument** ao projeto e associar ele ao controle **Print Preview Dialog** usando a propriedade **Document**.



Na pasta:
PrintPreviewDialog.sln

O controle Print Preview

- Você pode usar o controle **PrintPreviewControl** para mostrar como um documento (**PrintDocument**) será impresso.
- Este controle não possui botões e elementos de interface.
- Ele é usado apenas se você quiser construir sua própria interface de visualização de impressão.

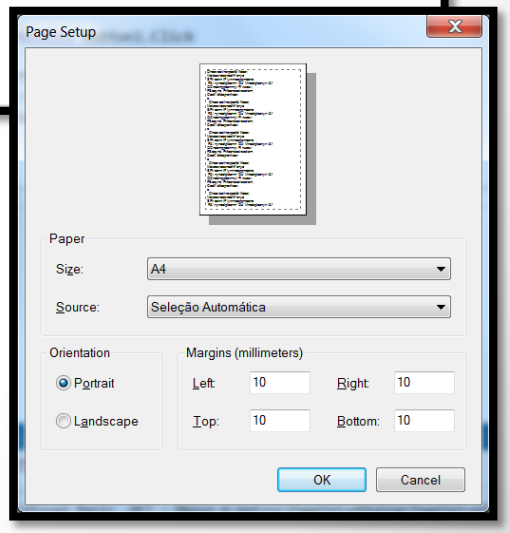


Na pasta:
PrintPreviewControl.sln

Page Setup Dialogs

- Usado para especificar detalhes da página a ser impressa.
- Você pode deixar a pessoa ajustar as bordas e margens do documento bem como o cabeçalho e o rodapé e a orientação do mesmo (retrato ou paisagem).

```
1  Public Class Form1
2      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3          PageSetupDialog1.Document = PrintDocument1
4          PageSetupDialog1.PrinterSettings = PrintDocument1.PrinterSettings
5          If PageSetupDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
6              PageSetupDialog1.PrinterSettings = PrintDocument1.PrinterSettings
7          End If
8      End Sub
9  End Class
```

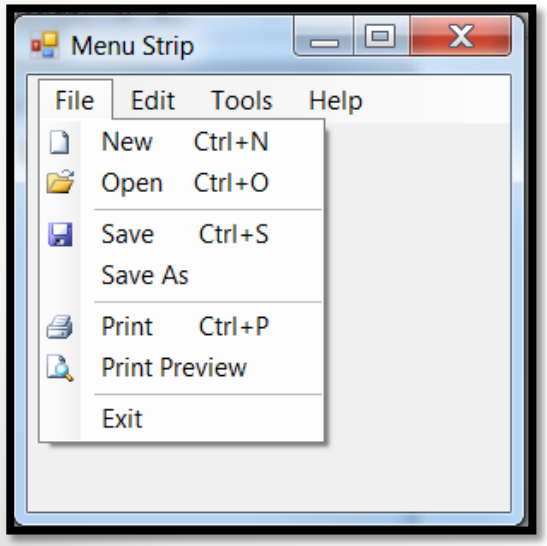


Na pasta:
PageSetupDialog.sln

Usando a classe MenuStrip

- Este é o tipo de menu mais usado em aplicações Windows.
- Os itens do menu são objetos da classe **ToolStripMenuItem**.
- A propriedade **Items** contém os elementos do menu.
- Se você clicar com o botão direito do mouse em cima do componente e selecionar a opção **Insert Standard Items** ele irá colocar em seu menu os elementos principais de qualquer aplicação Windows.

Exemplo:



Usando a classe ToolStripMenuItem

- Os itens adicionados ao menu pertencem a esta classe.
- Você pode adicionar um item de menu normal como um texto ou alguns outros elementos especiais como check boxes e separadores.
- Como o menu é considerado um componente o mesmo irá aparecer na bandeja de componentes do Visual Studio.
- O principal evento desta classe é o **Click**.
- Este controle é fácil de usar pois você precisa apenas saber quais são e como serão organizados seus itens do menu.

Exemplo:

```
1  Public Class Form1
2      Private Sub NovoToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles NovoToolStripMenuItem.Click
3          MsgBox("Você clicou no item de menu: " & sender.Text)
4      End Sub
5
6      Private Sub ArquivoToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ArquivoToolStripMenuItem.Click
7          MsgBox("Você clicou no item de menu: " & sender.Text)
8      End Sub
9
10     Private Sub SairToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles SairToolStripMenuItem.Click
11         End
12     End Sub
13 End Class
```

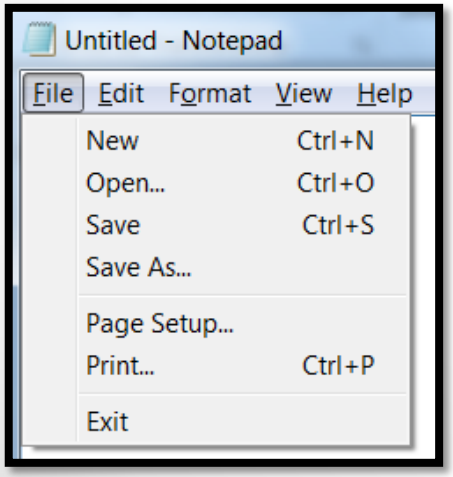
Criando um Menu

Atividade prática:

Vamos criar um pequeno menu usando o componente **MenuStrip**.

Como desafio será utilizado uma aplicação Windows qualquer para testar o conhecimento de criação de menus parecidos com os que são usados diariamente.

Tente criar no Visual Studio o menu do aplicativo **Notepad**.



Adicionando marcações nos itens do menu

- Você pode adicionar marcações (check boxes) em seus itens de menu usando a propriedade **Checked**.
- Ela pode ser usada em modo de design e tempo de execução.
- Ela é bastante útil para a pessoa habilitar/desabilitar alguma opção de configuração do aplicativo que você definiu.

Exemplo:

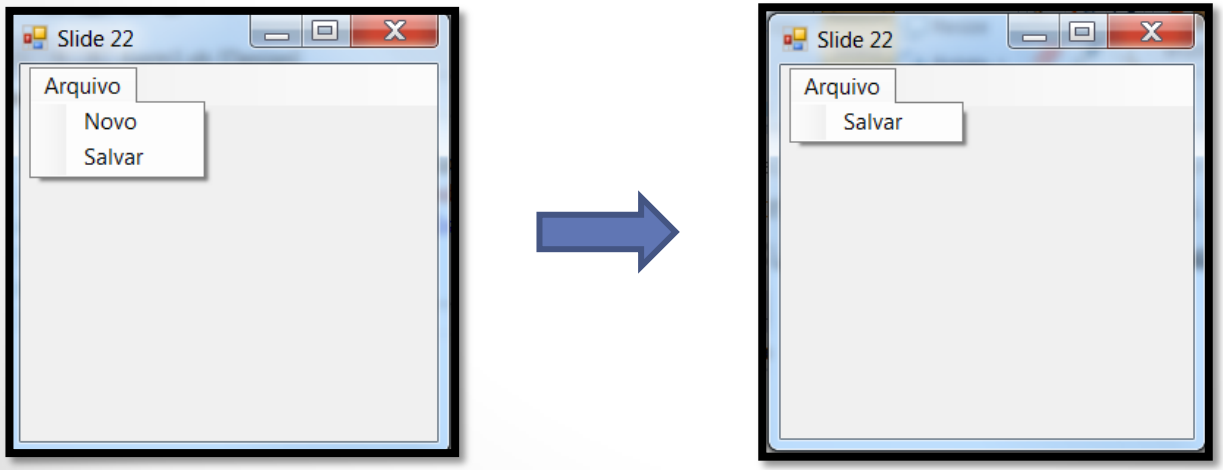
```
1  Public Class Form1
2      Private Sub CorretorOrtográficoToolStripMenuItem_Click(sender As Object, e As EventArgs) _
3          Handles CorretorOrtográficoToolStripMenuItem.Click
4          CorretorOrtográficoToolStripMenuItem.Checked = Not CorretorOrtográficoToolStripMenuItem.Checked
5      End Sub
6  End Class
```

Mostrando e escondendo itens de menu

- Para mostrar e esconder itens de menu você pode usar a propriedade **Visible**.
- Esse controle não possui os métodos **Show** e **Hide**.

Exemplo:

```
1  Public Class Form1
2      Private Sub NovoToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles NovoToolStripMenuItem.Click
3          MsgBox("Vou me esconder em 3, 2, 1...")
4          sender.Visible = False
5      End Sub
6  End Class
```

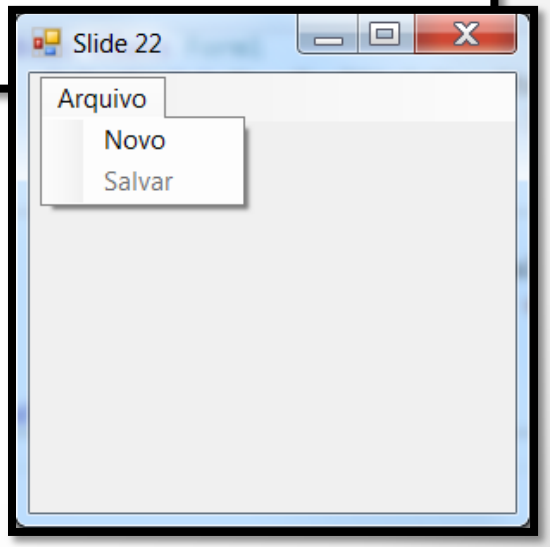


Desabilitando itens de menu

- Para desabilitar itens de menu você pode usar a propriedade **Enabled** alterando o valor dela ar **False**.
- O item do menu ficará com um tom cinza indicando que o mesmo não está acessível.

Exemplo:

```
Private Sub SalvarToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles SalvarToolStripMenuItem.Click
    MsgBox("Vou me desabilitar em 3, 2, 1...")
    sender.Enabled = False
End Sub
```



Criando menus via código

Exemplo:

```
1  Public Class Form1
2      Dim mainMenu1 As New MainMenu()
3
4      Dim WithEvents menuItem1 As New MenuItem()
5      Dim WithEvents menuItem2 As New MenuItem()
6      Dim WithEvents menuItem3 As New MenuItem()
7      Dim WithEvents menuItem4 As New MenuItem()
8
9      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
10         menuItem1.Text = "Arquivo"
11         menuItem2.Text = "Novo"
12         menuItem3.Text = "Arquivo texto..."
13         menuItem3.Checked = True
14         menuItem3.Shortcut = Shortcut.CtrlT
15         menuItem4.Text = "Imagem..."
16         menuItem4.Shortcut = Shortcut.CtrlI
17         menuItem2.MenuItems.Add(menuItem3)
18         menuItem2.MenuItems.Add(menuItem4)
19         AddHandler menuItem3.Click, AddressOf MenuItem3_Click
20         menuItem1.MenuItems.Add(menuItem2)
21         mainMenu1.MenuItems.Add(menuItem1)
22         Menu = mainMenu1
23     End Sub
24
25     Private Sub MenuItem3_Click(ByVal sender As System.Object, _
26                                 ByVal e As System.EventArgs) Handles menuItem3.Click
27         MsgBox("Você clicou em mim!")
28     End Sub
29 End Class
```

Usando a classe ContextMenuStrip

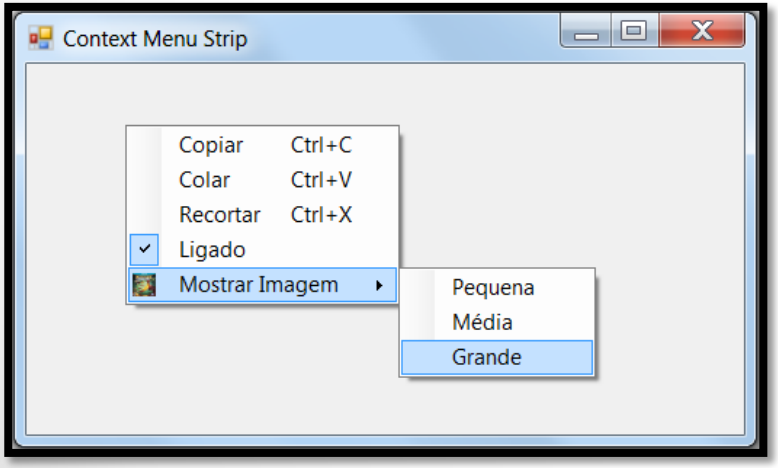
- Um tipo de menu que é usualmente acessível por meio do botão direito do mouse.
- Você usa a propriedade **ContextMenuStrip** de outro controle para associar um menu a ele.
- O evento **Click** é utilizado para gerenciar o código do item do menu clicado.
- As propriedades mais usadas desse controle são:
  - **Items**
  - **LayoutStyle**
  - **ShowCheckMargin**
  - **ShowImageMargin**
  - **ShowItemToolTips**
- A hierarquia de classes do controle **ContextMenuStrip** é a seguinte:

System
  Windows
    Forms
      ContextMenuStrip

Criando um controle ContextMenuStrip

- Para criar um menu deste tipo basta arrastar um controle do tipo **ContextMenuStrip**. Ele fica na categoria **Menus & Toolbars** da barra de ferramentas do Visual Studio.
- Digite os itens de menu desejados e associe o menu ao controle onde ele será usado.
- Você pode colocar imagens, teclas de atalho, caracteres de acesso, checkboxes e tools tips nos itens do menu e até incluir sub-menus.

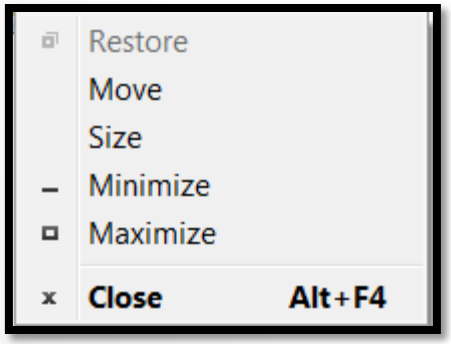
Exemplo:



Na pasta:
ContextMenuStrip.sln

ContextMenuStrip - Atividade Prática

- Crie um novo projeto visual e no form disponível coloque um controle do tipo **ContextMenuStrip**.
- Realize as alterações necessárias para que ele fique parecido com o da imagem abaixo.
- Os ícones não precisam ser necessariamente os mesmos; você poderá usar os seus ou achar alguns parecidos na Internet.



Usando a classe OpenFileDialog

- Um tipo de diálogo que é usado para deixar a pessoa escolher um ou vários arquivos de acordo com as configurações definidas no mesmo.
- Você usa o método **ShowDialog** para mostrar o mesmo.
- As propriedades mais usadas desse controle são:
  - **FileName e FileNames**
  - **Filter**
  - **InitialDirectory**
  - **Multiselect**
  - **RestoreDirectory**
  - **Title**
- A hierarquia de classes do controle **OpenFileDialog** é a seguinte:

System
  Windows
    Forms
      OpenFileDialog

Criando um Open File Dialog

- O exemplo do slide seguinte apresenta a criação de um **OpenFileDialog** que exibirá em um controle **PictureBox** a imagem selecionada pela pessoa.
- Para especificar que apenas imagens do tipo JPEG e GIF sejam selecionadas eu uso a propriedade **Filter**.
- O formato de texto da propriedade **Filter** é o seguinte:

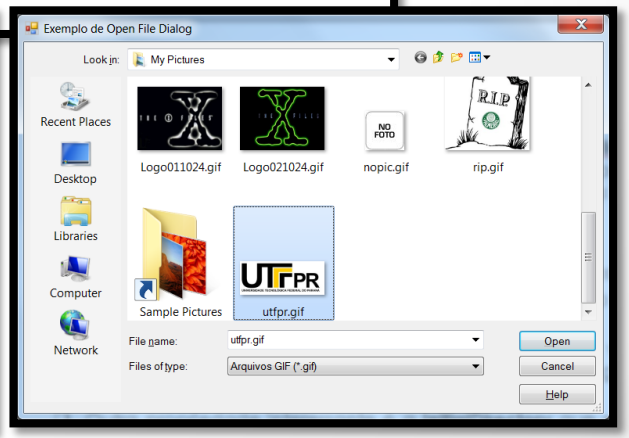
  Arquivos JPEG (*.jpg)|*.jpg|Arquivos GIF (*.gif)|*.gif|Todos os arquivos (*.*)|*.*

- Se a pessoa não clicar no botão Cancelar, o nome do arquivo poderá ser obtido usando a propriedade **FileName**.
- Se a propriedade **Multiselect** estiver igual a **True** então você deverá usar a propriedade **FileNames** para obter um array de strings com os arquivos selecionados pelo usuário.
- Outra propriedade interessante é a **InitialDirectory** que permite configurar o diretório que será aberto quando o método **ShowDialog** for chamado.
- Para mudar o título da caixa de diálogo use a propriedade **Title**.
- Para exibir o botão de ajuda use a propriedade **ShowHelp** e o evento **HelpRequest** para processar o click do botão.

Criando um Open File Dialog

Exemplo:

```
1  Public Class Form1
2      Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
3          PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
4          OpenFileDialog1.FileName = ""
5          OpenFileDialog1.Filter = "Arquivos JPEG (*.jpg)|*.jpg|Arquivos GIF (*.gif)|*.gif|Todos os arquivos (*.*)|*.*"
6          OpenFileDialog1.InitialDirectory = "C:\Users\Danilo Giacobo\Pictures"
7          OpenFileDialog1.ShowHelp = True
8          OpenFileDialog1.Title = "Exemplo de Open File Dialog"
9      End Sub
10
11     Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
12         If OpenFileDialog1.ShowDialog Then
13             PictureBox1.Image = Image.FromFile(OpenFileDialog1.FileName)
14         End If
15     End Sub
16
17     Private Sub OpenFileDialog1_HelpRequest(sender As Object, e As EventArgs) Handles OpenFileDialog1.HelpRequest
18         'Evento gerado quando a pessoa clica no botão de ajuda (Help)
19         MsgBox("Essa caixa de diálogo não tem ajuda!")
20     End Sub
21 End Class
```



Na pasta:
OpenFileDialog2.sln

Usando a classe SaveFileDialog

- Um tipo de diálogo que é usado para deixar a pessoa digitar o nome do arquivo que ela quer salvar de acordo com as configurações definidas no mesmo.
- Você usa o método **ShowDialog** para mostrar o mesmo.
- As propriedades mais usadas desse controle são:
  - **FileName**
  - **Filter**
  - **InitialDirectory**
  - **RestoreDirectory**
  - **Title**
- A hierarquia de classes do controle **SaveFileDialog** é a seguinte:

System
    Windows
        Forms
            SaveFileDialog

Criando um Save File Dialog

- O exemplo abaixo apresenta a criação de um **SaveFileDialog** que exibirá uma mensagem com o nome do arquivo selecionado pela pessoa.
- Você testa o retorno do método **ShowDialog** usando a enumeração **DialogResult**.
- **Dica:** você pode usar a propriedade **CreatePrompt** para perguntar ao usuário se ele deseja criar o arquivo caso o mesmo não exista. A propriedade **OverwritePrompt** quando **True** pergunta ao usuário se ele quer sobrescrever o arquivo.

Exemplo:

Na pasta:
SaveFileDialog2.sln

```
1  Public Class Form1
2      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3          SaveFileDialog1.Title = "Salvar como..."
4          SaveFileDialog1.CreatePrompt = True
5          SaveFileDialog1.OverwritePrompt = True
6          If SaveFileDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel Then
7              MsgBox("Você escolheu o arquivo: " & SaveFileDialog1.FileName)
8          End If
9      End Sub
10 End Class
```

Usando a classe FontDialog

- Um tipo de diálogo que permite ao usuário escolher uma fonte de acordo com as configurações definidas no mesmo.
- Você usa o método **ShowDialog** para mostrar o mesmo.
- Ele retorna um objeto do tipo **Font** na propriedade **Font**.
- Ele retorna um objeto do tipo **Color** na propriedade **Color**.
- As propriedades mais usadas desse componente são:
  - **MinSize e MaxSize**
  - **Color e Font**
  - **ShowApply e ShowColor**
- A hierarquia de classes do controle **FontDialog** é a seguinte:

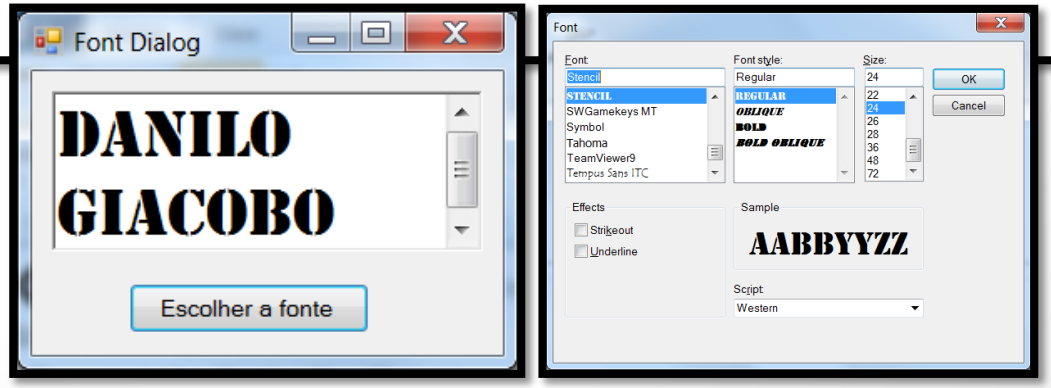
System
  Windows
    Forms
      FontDialog

Criando um Font Dialog

- O exemplo abaixo apresenta a criação de um **FontDialog** que exibirá em um controle **RichTextBox** a fonte e a cor selecionada pela pessoa.
- O programa configura as propriedades **Font** e **ForeColor** do rich text box para mudar a aparência do texto de acordo com os valores selecionados.

Exemplo:

```
1  Public Class Form1
2      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3          If FontDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel Then
4              RichTextBox1.Font = FontDialog1.Font
5              RichTextBox1.ForeColor = FontDialog1.Color
6          End If
7      End Sub
8  End Class
```



Na pasta:
FontDialog2.sln

Usando a classe ColorDialog

- Um tipo de diálogo que permite ao usuário escolher uma cor de acordo com as configurações definidas no mesmo.
- Você usa o método **ShowDialog** para mostrar o mesmo.
- Ele retorna um objeto do tipo **Color** na propriedade **Color**.
- As propriedades mais usadas desse controle são:
  - **AllowFullOpen e FullOpen**
  - **AnyColor**
  - **SolidColorOnly**
  - **ShowHelp**
- A hierarquia de classes do controle **ColorDialog** é a seguinte:

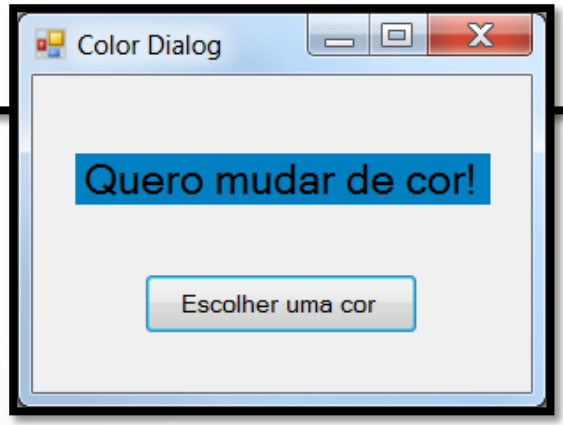
System
  Windows
    Forms
      ColorDialog

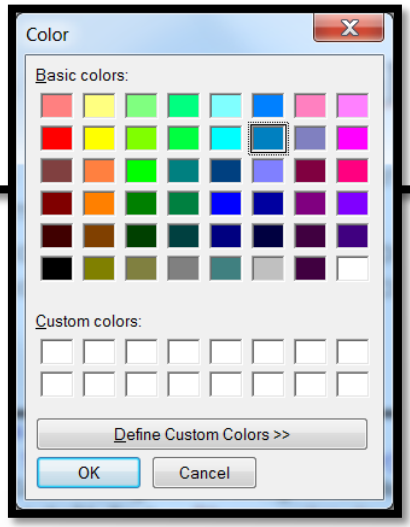
Criando um Color Dialog

- O exemplo abaixo permite ao usuário escolher uma cor para mudar o fundo de um controle do tipo **Label**.
- O programa configura as propriedades **Text** e **BackColor** do label para mudar a aparência do mesmo de acordo com a cor selecionada.

Exemplo:

```
1  Public Class Form1
2      Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
3          If ColorDialog1.ShowDialog <> Windows.Forms.DialogResult.Cancel Then
4              Label1.Text = "Quero mudar de cor!"
5              Label1.BackColor = ColorDialog1.Color
6          End If
7      End Sub
8  End Class
```





Na pasta:
ColorDialog2.sln

Usando a classe PrintDocument

- Objetos do tipo **PrintDocument** suportam eventos e operações relacionados a impressão de documentos.
- O **PrintPage** é o evento principal desta classe.
- O método **Print** é usado para imprimir um documento.
- Quando trabalho de impressão começa o evento **BeginPrint** ocorre, seguido pelo evento **PrintPage** (para cada página a ser impressa) e por último o evento **EndPrint**.
- Use a propriedade **HasMorePages** para saber se há mais páginas a serem impressas.
- A hierarquia de classes do controle **PrintDocument** é a seguinte:

System
    Drawing
        Printing
            PrintDocument

Usando a classe PrinterSettings

- A classe **PrinterSettings** é usada para configurar como o documento será impresso - em qual impressora, quantas cópias, de qual página até qual página e assim em diante.
- As propriedades de destaque desta classe são:
  - **Copies**
  - **DefaultPageSettings**
  - **FromPage**
  - **PrinterName**
  - **ToPage**
- A hierarquia de classes da classe **PrinterSettings** é a seguinte:

System
  Drawing
    Printing
      PrinterSetting

Usando a classe PrintDialog

- Este diálogo permite a pessoa configurar as opções de impressão de um documento e imprimir o mesmo por meio do botão Imprimir.
- Você usa o método **ShowDialog** para mostrar o diálogo para a pessoa.
- As propriedades de destaque desta classe são:
  - **AllowPrintToFile**
  - **AllowSelection**
  - **AllowSomePages**
  - **Document**
  - **PrinterSettings**
  - **PrintToFile**
- A hierarquia de classes da classe **PrintDialog** é a seguinte:

System
    Windows
        Forms
            PrintDialog

Imprimindo...

```
Public Class Form1
    Dim intNumeroPagina As Integer

    Private Sub ImprimirToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ImprimirToolStripMenuItem.Click
        PrintDialog1.Document = PrintDocument1
        PrintDialog1.PrinterSettings = PrintDocument1.PrinterSettings
        PrintDialog1.AllowSomePages = True
        If PrintDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
            PrintDocument1.PrinterSettings = PrintDialog1.PrinterSettings
            PrintDocument1.Print()
        End If
    End Sub

    Private Sub PrintDocument1_BeginPrint(sender As Object, e As Printing.PrintEventArgs) Handles PrintDocument1.BeginPrint
        intNumeroPagina = 0
    End Sub

    Private Sub PrintDocument1_EndPrint(sender As Object, e As Printing.PrintEventArgs) Handles PrintDocument1.EndPrint
        MsgBox("Impressão finalizada.")
    End Sub

    Private Sub PrintDocument1_PrintPage(sender As Object, e As Printing.PrintPageEventArgs) Handles PrintDocument1.PrintPage
        intNumeroPagina += 1
        Select Case intNumeroPagina
            Case 1
                e.Graphics.FillRectangle(Brushes.Red, New Rectangle(200, 200, 500, 500))
                e.HasMorePages = True
            Case 2
                e.Graphics.FillRectangle(Brushes.Blue, New Rectangle(200, 200, 500, 500))
                e.HasMorePages = False
        End Select
    End Sub
End Class
```

Na pasta:
Print.sln

Usando a classe PrintPreviewDialog

- Este diálogo permite a pessoa verificar como será impresso o documento.
- Você usa o método **ShowDialog** para mostrar o diálogo para a pessoa.
- As propriedades de destaque desta classe são:
  - **Document**
  - **Modal**
  - **Name**
  - **ShowIcon**
- A hierarquia de classes da classe **PrintPreviewDialog** é a seguinte:

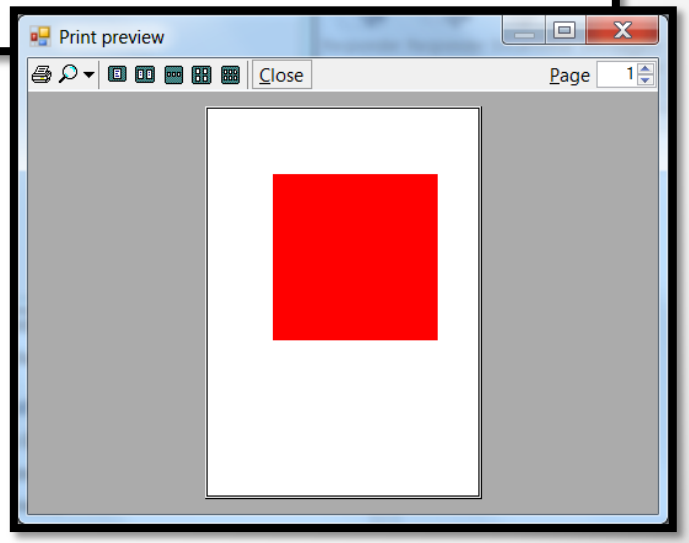
System
  Windows
    Forms
      PrintPreviewDialog

Criando um Print Preview Dialog

- Para mostrar a caixa de diálogo de visualização da impressão na tela basta atribuir um objeto do tipo **PrintDocument** para a propriedade **Document** do controle de diálogo e usar o método **ShowDialog** para exibir o mesmo.

Exemplo:

```
Private Sub VisualizarImpressãoToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles _
    VisualizarImpressãoToolStripMenuItem.Click
    PrintPreviewDialog1.Document = PrintDocument1
    PrintPreviewDialog1.ShowDialog()
End Sub
```



Usando a classe PrintPreviewControl

- Controle usado para criar sua própria customização da visualização de impressão de documentos.
- As propriedades de destaque desta classe são:
  - **AutoZoom**
  - **Columns**
  - **Rows**
  - **StartPage**
  - **Zoom**
- A hierarquia de classes da classe **PrintPreviewControl** é a seguinte:

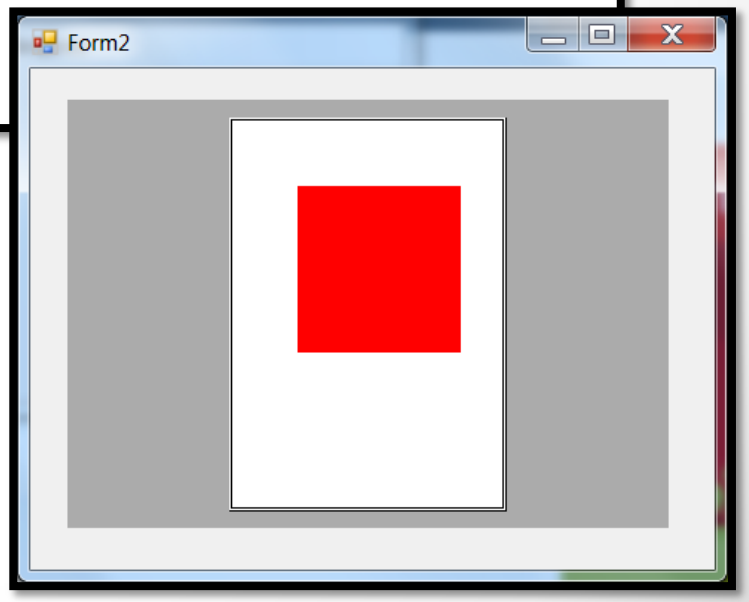
System
    Windows
        Forms
            PrintPreviewControl

Criando um Print Preview Control

- Para usar este controle é necessário atribuir um objeto do tipo **PrintDocument** para a propriedade **Document** do controle de diálogo.
- No exemplo abaixo o controle foi colocado em um novo form e foi chamado do form principal da aplicação usando o método **Show**.

Exemplo:

```
Private Sub CustomizarAVisualizaçãoDaImpressãoToolStripMenuItem_Click(sender As Object, e As EventArgs) _
    Handles CustomizarAVisualizaçãoDaImpressãoToolStripMenuItem.Click
    Dim frmPreview As New Form2()
    frmPreview.PrintPreviewControl1.Document = PrintDocument1
    frmPreview.Show()
End Sub
```



Usando a classe PageSetupDialog

- Caixa de diálogo usada para especificar a orientação da página, tamanho do papel, tamanho da margem e outras configurações.
- O método **ShowDialog** mostra a caixa de diálogo na tela e o método **Reset** retorna as configurações iniciais do diálogo.
- As propriedades de destaque desta classe são:
  - **AllowMargins**
  - **AllowOrientation**
  - **AllowPaper**
  - **AllowPrinter**
  - **Document**
  - **ShowHelp**
  - **ShowNetwork**
- A hierarquia de classes da classe **PageSetupDialog** é a seguinte:

System
    Windows
        Forms
            PageSetupDialog

Criando um Page Setup Dialog

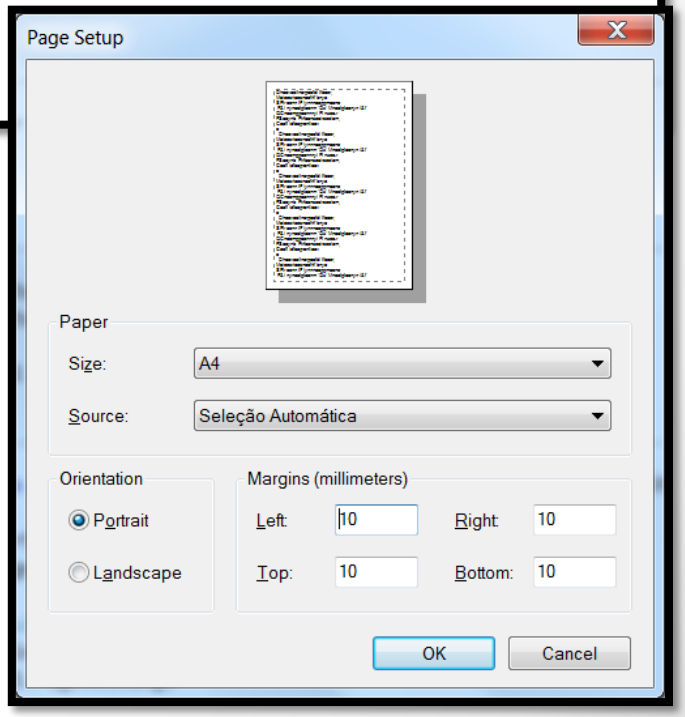
- As configurações deste tipo de diálogo são armazenadas em um objeto do tipo **PageSettings** do componente **PrintDocument**.
- As propriedades dignas de nota da classe **PageSettings** são:
  - Bounds
  - Color
  - Landscape
  - Margins
  - PaperSize
  - PaperSource
  - PrinterResolution
  - PrinterSettings

Criando um Page Setup Dialog

Exemplo:

```
Private Sub ConfigurarPáginaToolStripMenuItem_Click(sender As Object, e As EventArgs) Handles ConfigurarPáginaToolStripMenuItem.Click
    PageSetupDialog1.Document = PrintDocument1
    PageSetupDialog1.PrinterSettings = PrintDocument1.PrinterSettings
    PageSetupDialog1.PageSettings = PrintDocument1.DefaultPageSettings

    If PageSetupDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
        PrintDocument1.PrinterSettings = PageSetupDialog1.PrinterSettings
        PrintDocument1.DefaultPageSettings = PageSetupDialog1.PageSettings
    End If
End Sub
```



Referências Bibliográficas

- HOLZNER, Steven. **Visual basic.NET: black book**. Arizona: Coriolis Group Books, 2002. xxxviii, 1144 p ISBN 1-57610-835-X.