Solução Aula 05 – Projeto1

Utilizando o controle Timer

Através do controle **Timer** você pode executar ações periodicamente.

Eventualmente você necessitará executar ações repetidamente, por exemplo, salvar um arquivo a cada 3 minutos ou atualizar a interface do usuário a cada 1 segundo .

O Timer difere dos controles que você usou até aqui, pois ele não tem uma representação visual em tempo de execução.

O Timer tem duas propriedades e um evento que são usados na maioria das vezes. A propriedade **Enabled** determina se o componente Timer esta ativo ou não.

Se Enabled estiver como True, Timer está ativo.
Se Enabled estiver como False, Timer está inativo.

A propriedade **Interval** determina o intervalo de execução em milissegundos.

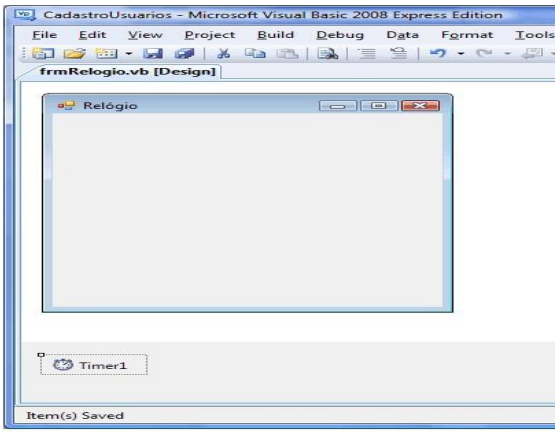
Por exemplo, se a propriedade Interval é definida como 1000, o componente Timer irá disparar o evento a cada 1000 milissegundos, ou a cada segundo.

Definindo as propriedades Enabled e Interval e adicionando código ao procedimento de evento, o mesmo será executado em intervalos definidos sem a necessidade de ação do usuário.

Exemplo do uso do componente Timer

Crie um novo formulário denominado **frmRelogio** e adicione um componente Timer ao mesmo.

Observação
O componente Timer se encontra na aba Components da ToolBox



Observe que o componente Timer não é exibido no formulário em si, mas na barra de componentes abaixo do formulário.
Isso ocorre porque o temporizador não tem uma representação visual.

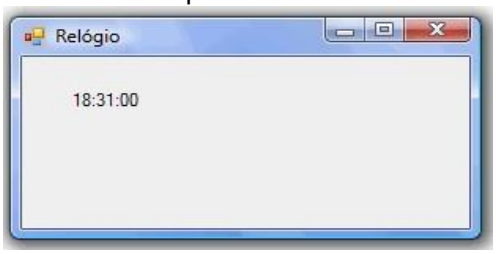
Selecione o componente Timer e em seguida, na janela Properties, defina a propriedade **Enabled** como True e **Interval** como 1000.

Clique duas vezes no componente Timer1 para abrir o Editor de Códigos.

Adicione código a seguir.

```
private void timer1_Tick(object sender, EventArgs e)
{
    label1.Text = DateTime.Now.ToString("hh:mm:ss");
}
```

Ao executar o aplicativo.



O texto no rótulo é atualizado a cada segundo (mil milisegundos) com a hora atual.

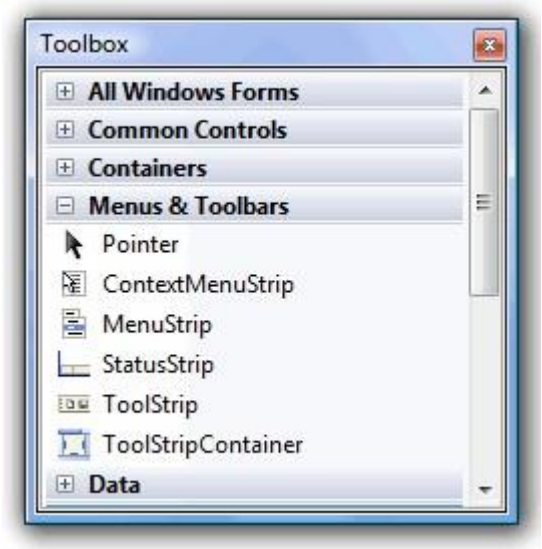
Solução Aula 05 – Projeto2

Trabalhando com Menus

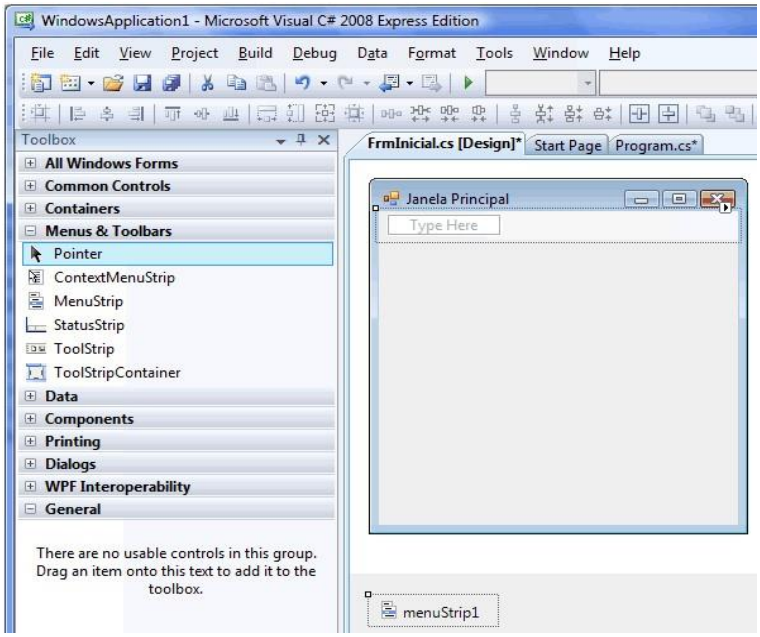
O Visual C# facilita a implementação de menus através de alguns componentes prontos.
Você pode usar o controle **MenuStrip** para criar menus graficamente.

Exemplo

Crie um novo formulário chamado de FrmInicial
Na Toolbox abra a aba Menus & Toolbars



Arraste para o formulário o controle MenuStrip
Quando arrastado para um formulário, o controle **MenuStrip** aparece como uma caixa que contém as palavras **"Type Here"** localizadas na parte superior do formulário.



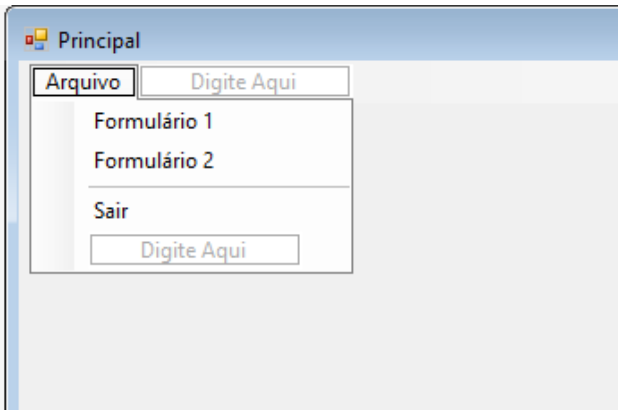
Você pode clicar na caixa e digitar dentro dela para criar títulos de menu.

Itens adicionais ao menu

Quando o título de um item de menu é definido, os itens de menu adicionais podem ser criados abaixo e à direita do primeiro.

Isso permite que você estenda o menu com quantos itens adicionais ou sub-items você desejar.

Adicione alguns itens e uma linha separadora
Para adicionar uma linha separadora clique na seta a direita do Type Here. Selecione a opção Separator



Programando os menus

Quando a aparência de seu menu for concluída, você pode criar procedimentos de eventos para cada item.

Antes de adicionar código não se esqueça de alterar a propriedade Name de cada item de menu

Faça as seguintes alterações

Menu Arquivo altere para mnuArquivo
Menu Abrir altere para mnuFormulario1
Menu Fechar altere para mnuFormulario2

Menu Sair altere para mnuSair

Clique duas vezes no menu Sair para abrir o Editor de Códigos e adicione o método Close.

```
private void mnuSair_Click(object sender, EventArgs e)
{
    Close();
}
```

Execute a aplicação e de um clique no menu Arquivo -> Sair

Criando e Ativando ou Desativando os formulários pelo menu

Em alguns casos as opções do menu devem estar disponíveis apenas em determinados momentos.

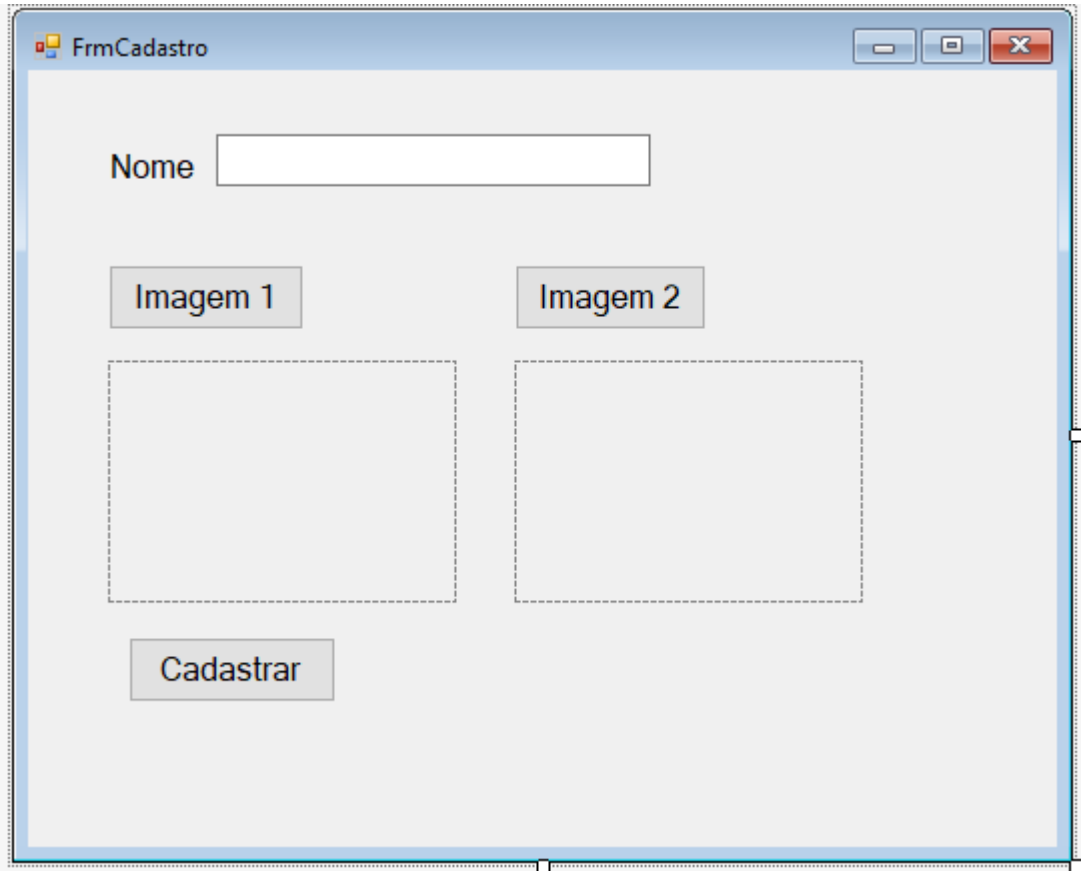
Quando um item de menu está desativado, a cor do texto de menu é alterado para cinza, e clicar no item de menu não fará nada.

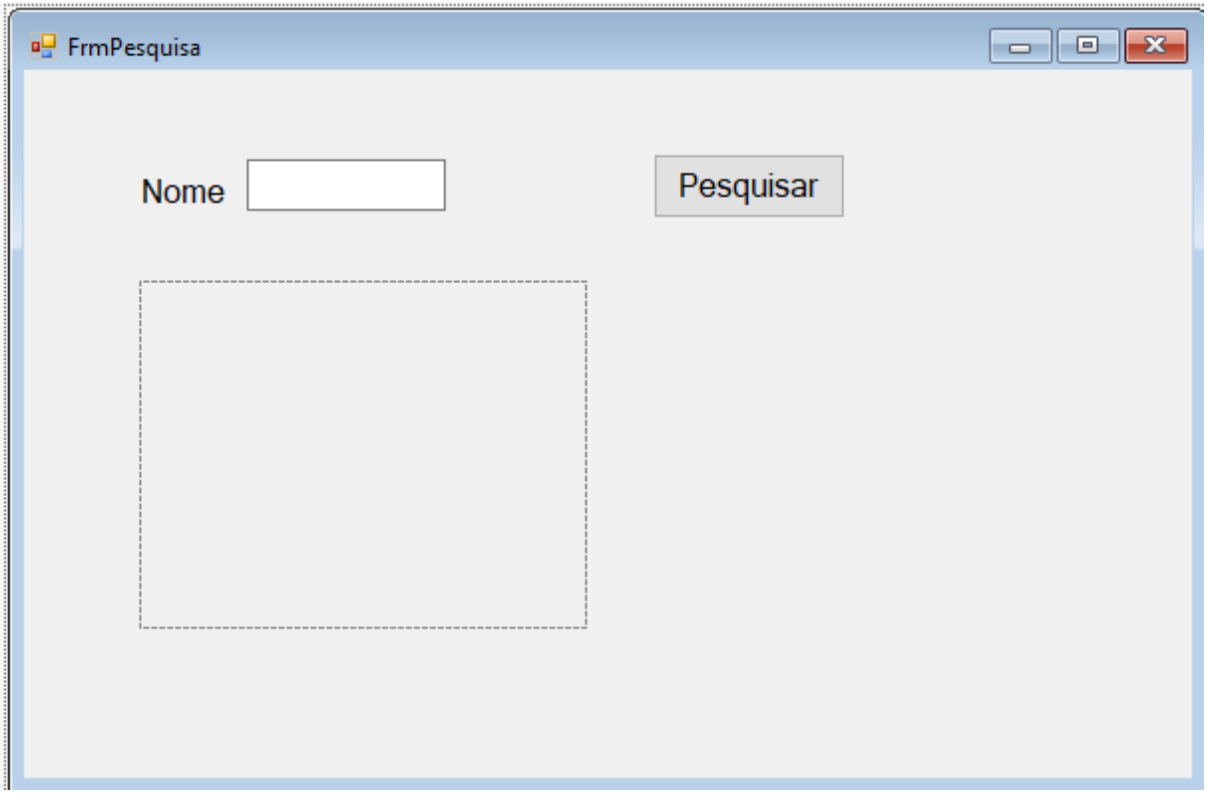
Para desativar ou ativar os itens de menu basta configurar a propriedade **Enabled**.

```
private void mnuFormulario1_Click(object sender, EventArgs e)
{

    Form1 f1 = new Form1();
    f1.ShowDialog();
    mnuFormulario2.Enabled = true;
}

1 referência
private void mnuFormulario2_Click(object sender, EventArgs e)
{
    Form2 f2 = new Form2();
    f2.ShowDialog();
}
```

LTPII - Aula 05

Exercício Resolvido – Dentro da Solução Aula05 criar o Projeto 3






```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Drawing;

namespace Projeto3
{
    8 referências
    class Cadastro
    {
        private string nome;
        private Bitmap img1;
        private Bitmap img2;

        2 referências
        public string Nome { get => nome; set => nome = value; }
        2 referências
        public Bitmap Img1 { get => img1; set => img1 = value; }
        2 referências
        public Bitmap Img2 { get => img2; set => img2 = value; }
    }
}
```

```
public partial class FrmPrincipal : Form
{
    1 referência
    public FrmPrincipal()
    {
        InitializeComponent();
    }
    private FrmCadastro FC = new FrmCadastro();

    1 referência
    private void mnuCadastro_Click(object sender, EventArgs e)
    {

        FC.ShowDialog();
    }

    1 referência
    private void mnuPesquisa_Click(object sender, EventArgs e)
    {
        FrmPesquisa FP = new FrmPesquisa();
        FP.Lista = FC.Lista;
        FP.ShowDialog();

    }
}
```

```
public partial class FrmCadastro : Form
{
    1 referência
    public FrmCadastro()
    {
        InitializeComponent();
    }
    private Bitmap img1, img2;
    private Cadastro c;
    private List<Cadastro> lista = new List<Cadastro>();
    2 referências
    internal List<Cadastro> Lista { get => lista; set => lista = value; }
    1 referência
    private void cmdImagem2_Click(object sender, EventArgs e)
    {
        if (openFileDialog1.ShowDialog() == DialogResult.OK)
        {
            img2 = new Bitmap(openFileDialog1.FileName);
            pictureBox2.Image = img2;
        }
    }

    1 referência
    private void cmdCadastrar_Click(object sender, EventArgs e)
    {
        c = new Cadastro();
        c.Nome = txtNome.Text;
        c.Img1 = img1;
        c.Img2 = img2;
        Lista.Add(c);
        MessageBox.Show("Cadastrado com sucesso");
    }


    1 referência
    private void cmdImagem1_Click(object sender, EventArgs e)
    {
        if(openFileDialog1.ShowDialog() == DialogResult.OK)
        {
            img1 = new Bitmap(openFileDialog1.FileName);
            pictureBox1.Image = img1;
        }
    }
}
```

```
public FrmPesquisa()
{
    InitializeComponent();
}
private List<Cadastro> lista = null;
2 referências
internal List<Cadastro> Lista { get => lista; set => lista = value; }
private Bitmap img1, img2;
private int flag = 1;
1 referência
private void timer1_Tick(object sender, EventArgs e)
{
    if (flag == 1)
    {
        pictureBox1.Image = img1;
        flag = 2;
    }
    else
    {
        pictureBox1.Image = img2;
        flag = 1;
    }

}
1 referência
private void cmdPesquisar_Click(object sender, EventArgs e)
{
    foreach (Cadastro item in Lista)
    {
        if(item.Nome == txtNome.Text)
        {
            img1 = item.Img1;
            img2 = item.Img2;
            timer1.Enabled = true;
        }
    }
}
```