Programação em C# com Visual Studio .NET

Professor Leo Larback

Esta apresentação é parte de um material desenvolvido pelo Prof. Alessandro Brawerman – disponível em

http://docb.gratix.com.br/csharp

Trabalhando com o Visual Studio .NET

- Como os programas em Windows são orientados a eventos é comum desativar opções de menus e botões.
- Neste primeiro programa, vamos criar uma espécie de gangorra eletrônica.

Programa 01

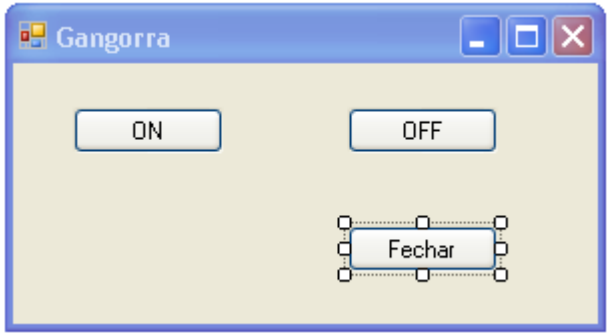
- Criar uma aplicação do tipo Windows Application e chamá-la de Gangorra.

Programa 01 – Passo a Passo

- Mude o título do formulário para Gangorra
  - Propriedade text
- Insira dois botões alinhados na horizontal.
- Mude o texto dos botões para ON e OFF.
  - Propriedade text
- Mude o nome dos botões para bOn e bOff.
  - Propriedade name

Programa 01 – Passo a Passo

- Insira mais um botão abaixo e alinhe-o na vertical com um dos anteriores.
- Chame-o de bFechar e mude seu texto para Fechar.



Programa 01 – Passo a Passo

- O objetivo é que quando o usuário clique em um botão, desabilite o outro, formando assim uma gangorra.
- Na tela de design do form, clique duas vezes no botão ON.
- Isto cria o evento clique no código. Toda vez que este botão for pressionado pelo usuário o código neste método é executado.

Programa 01 – Passo a Passo

- Insira o seguinte código no método click do botão ON:

```
private void bOn_Click(object sender, EventArgs e)
        {
            bOn.Enabled = false;
            bOff.Enabled = true;
        }
```

Programa 01 – Passo a Passo

- Volte para a tela de design do form, use a aba ou apert Shift + F7.
- Duplo clique no botão OFF e insira o seguinte código:

```
private void bOff_Click(object sender, EventArgs e)
        {
            bOff.Enabled = false;
            bOn.Enabled = true;
        }
```

Programa 01 – Passo a Passo

- Volte para a tela de design do form, use a aba ou apert Shift + F7.
- Duplo clique no botão Fechar e insira o seguinte código:

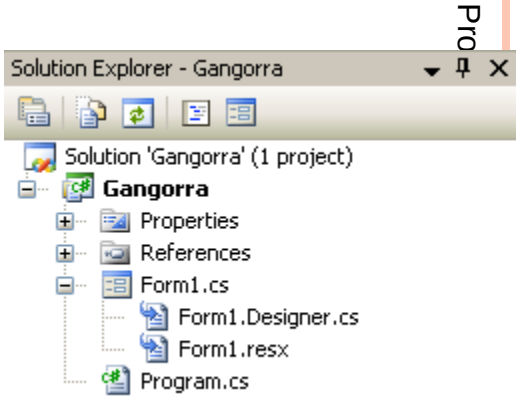
```
private void bFechar_Click(object sender,
EventArgs e)
{
        Close();
}
```

Programa 01 – Passo a Passo

- Antes de executar edite a propriedade Enabled do botão OFF e marque false.
- Execute o programa.
- O que acontece?
  - Ao iniciar o botão OFF está desabilitado.
  - Ao clicar em ON, o botão ON fica desabilitado e o OFF é habilitado.
  - Ao clicar em OFF, o botão OFF fica desabilitado e o ON é habilitado.
  - Ao clicar em Fechar, a aplicações é fechada.

Programa 01 – Passo a Passo

- Examine os arquivos listados no Solution Explorer.
  - Solution "Gangorra"
    - Arquivo de solução de nível superior
    - Um por aplicativo
    - Informa o projeto inicial



(Name)	**Gangorra**
Description	
Path	D:\projects\csharp\curso\Programa01\Gangorra\Gangorra.sln
Startup project	**Gangorra**

Programa 01 – Passo a Passo

- Gangorra
  - Arquivo do projeto do C#.
  - Cada arquivo de projeto referencia um ou mais arquivos que contém o código-fonte.
  - Todos os códigos-fonte de um único projeto devem ser escritos na mesma linguagem.

Project File	Gangorra.csproj
Project Folder	D:\projects\csharp\curso\Programa01\Gangorra\Gangorra\

Programa 01 – Passo a Passo

- Properties
  - Contém um arquivo denominado AssemblyInfo.cs para adicionar atributos ao programa, como nome do autor e outros.
- References
  - Contém referências ao código compilado que seu aplicativo pode usar.
  - Quando o código é compilado é gerado código em assembly que pode ser oferecido a outro desenvolvedor.
- Program.cs
  - Arquivo fonte do C#, ponto de entrada da aplicação.

Identificadores

- Nomes de objetos, variáveis, métodos, classes, etc.
- Primeiro caractere não pode ser um número, deve ser uma letra ou um _.
- Nenhum espaço é permitido.
- Sensíveis a letras minúsculas e maiúsculas (case sensitive).
- A propriedade Name de um botão é um identificador.

Propriedades mais Comuns

- Name
  - Comum a todos os componentes da paleta.
  - Automaticamente nomeados usando o nome da classe do componente e um número sequencial.
  - Quando um componente é renomeado, o Visual Studio atualiza automaticamente todo o código gerado por ele e as propriedades de outros componentes que fazem referência ao componente renomeado.

Propriedades mais Comuns

- Name
  - Não é atualizado o código gerado pelo programador.
  - Portanto, se for mudar o nome do componente, mude logo no início para não dar muito mais trabalho depois.

Propriedades mais Comuns

- Text
  - Todos os componentes que podem apresentar um rótulo têm esta propriedade.
  - Armazena a string que será mostrada quando o componente for desenhado.

Propriedades mais Comuns

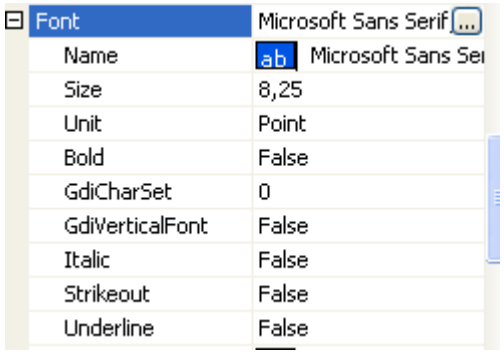
- Location
  - Esquerda – X e topo – Y
  - Armazena a posição do componente em relação ao form ou painel que o contém.
  - Movendo o componente, estas propriedades se atualizam automaticamente.
  - Alterando estas propriedades, o componente é movido.
  - Location(X; Y)

Propriedades mais Comuns

- Size
  - Altura e comprimento.
  - Similar a Location.
  - Size(Height, Width).
- BackColor
  - Cor do componente.
- ForeColor
  - Cor do texto do componente.

Propriedades mais Comuns

- Font
  - Permite selecionar tamanho e tipo da fonte que sera usada para escrever o texto no componente.

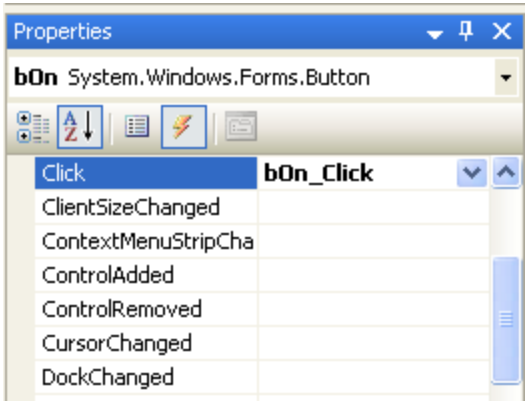


Propriedades mais Comuns

- TabIndex
  - Ordem do componente no form ou painel.
- Image
  - Caminho da imagem que será exibida no componente.
- Visible
  - Determina se o componente é visível ou está escondido.
- Enabled
  - Indica se o componente está habilidato ou desabilitado.

Eventos

- Cada objeto possui uma lista de eventos.
- Para visualizar os eventos de um componente ative a paleta propriedades, selecione o componente desejado e clique sobre o ícone similar a um raio.



Eventos

- Eventos são atendidos por manipuladores (handlers) e inseridos automaticamente no código quando habilitados.

```
private void InitializeComponent()
{
    this.bOn = new System.Windows.Forms.Button();
    this.bOff = new System.Windows.Forms.Button();
    this.bFechar = new System.Windows.Forms.Button();
    this.SuspendLayout();
    //
    // bOn
    //
    this.bOn.Location = new System.Drawing.Point(30, 22);
    this.bOn.Name = "bOn";
    this.bOn.Size = new System.Drawing.Size(75, 23);
    this.bOn.TabIndex = 0;
    this.bOn.Text = "ON";
    this.bOn.UseVisualStyleBackColor = true;
    this.bOn.Click += new System.EventHandler(this.bOn_Click);
```

→ Design do form

Eventos

- A linha:

```
this.bOn.Click += new System.EventHandler(this.bOn_Click);
```

foi gerada automaticamente quando inserido o evento Click na aba de eventos.

- O que o evento deve fazer é por conta do programador.

```
private void bOn_Click(object sender, EventArgs e)
{
    bOn.Enabled = false;
    bOff.Enabled = true;
}
```

Código do form

Eventos mais Comuns

- Click
  - Gerado cada vez que o botão esquerdo do mouse é pressionado e solto em cima do componente.
  - Só ocorre quando o usuário libera o botão.
- KeyPress
  - Gerado quando o usuário pressiona e libera uma tecla no teclado.
  - Muito usado para reconhecimento de teclas em TextBox e ListBox.

Eventos mais Comuns

- Enter
  - Quando o componente se torna o componente ativo na aplicação.
  - Suponha uma tela com vários campos de entrada.
  - Quando a tela é apresentada o foco está sobre o primeiro campo.
  - Após pressionar Tab o usuário passa para o próximo campo.
  - O foco da aplicação passa para o próximo campo.

Eventos mais Comuns

- Leave
  - Gerado imediatamente antes de o foco deixar o componente.
- Resize
  - Gerado quando o tamanho do componente é alterado.
- TextChanged
  - Quando o valor da propriedade Text do componente muda.

Métodos Simples

- Da mesma forma que eventos, cada objeto possui sua própria lista de métodos.
- Show()
  - Ativa o evento de renderização do form.
  - É desenhado e ativado.
- Close()
  - Aplicado geralmente em forms e arquivos.
  - Quando utilizado no form principal, encerra a aplicação.

Métodos Simples

- Refresh()
  - Redesenhar
  - Antes de redesenhar, apaga o componente.
  - Quando aplicado em arquivos, faz com o buffer do mesmo seja recarregado.

Métodos Simples

- Dispose()
  - Libera o endereço de memória alocado com o Create para que o Garbage Collector cuide de sua remoção.
- Hide()
  - Esconde o objeto.

Melhorando a Gangorra

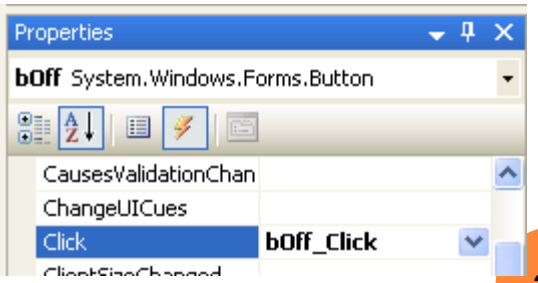
- Vamos otimizar o código da Gangorra.

```
private void bOn_Click(object sender, EventArgs e)
{
        bOn.Enabled = false;
        bOff.Enabled = true;
}
-----
private void bOn_Click(object sender, EventArgs e)
{
        bOn.Enabled = !bOn.Enabled;
        bOff.Enabled = !bOff.Enabled;
}
```

Melhorando a Gangorra

- Operador unário ! – funciona como negação de expressão seguinte.
- Passo seguinte é excluir o evento Click atribuído ao bOff.
- Exclua o código-fonte inserido e depois limpe a caixa de texto ao lado do nome do evento na janela Properties (Events).

```
private void bOff_Click(object sender, EventArgs e)
{
    bOff.Enabled = false;
    bOn.Enabled = true;
}
```



Melhorando a Gangorra

- Execute o programa e certifique-se que não existe mais evento atribuído ao bOff.
- Agora vamos utilizar reusabilidade de código.
- Abra a caixa de texto do evento Click do bOff e atribua o evento já existente bOn_Click.