

**Programação para computadores - C#**

## **Semana 1**

* Introdução à programação visual (C#)
* Download e instalação do software Microsoft Visual Studio
* Conhecendo o ambiente de desenvolvimento
* Criar e salvar um projeto
* Formulários
* Executando um projeto

## **Introdução ao Visual Studio**

Visual Studio é um conjunto completo de ferramentas de desenvolvimento para construção de aplicações Web ASP.NET, serviços Web XML, aplicações desktop e aplicativos móveis.

Visual Basic, Visual C# e Visual C++ todos usam o mesmo ambiente de desenvolvimento integrado (IDE), que permite o compartilhamento de ferramentas e facilita a criação de soluções de linguagens

Além disso, essas linguagens usam a funcionalidade do *.NET Framework*, que fornece acesso às tecnologias chaves que simplificam o desenvolvimento de aplicativos Web em ASP e serviços Web XML.

## **O que posso fazer com o .NET?**

O .NET permite desenvolver soluções como:

* Aplicativos Web
* Aplicativos para Servidores
* Aplicativos *Smart Client*
* Aplicativos de Console
* Aplicativos de Banco de Dados
* Serviços Windows
* Serviços Web

## **TERMOS DA PLATAFORMA**

**CLR** - Sigla de *Common Language Runtime*. Base comum a todas as linguagens .NET, o CLR é o ambiente que gerencia a execução de código escrito em qualquer linguagem. Faz parte do *Framework*.

**FRAMEWORK** - É o modelo da plataforma .NET para construir, instalar e rodar qualquer aplicação, no desktop ou na Internet. Para executar um programa .NET, é preciso ter o *Framework* instalado.

## **TERMOS DA PLATAFORMA**

**IDE COMPARTILHADO** - Ambiente integrado de programação (*Integrated Development Environment*) do Visual Studio.NET. Diferentes linguagens usam o mesmo editor de código e depurador e compilam executáveis na linguagem MSIL. Além das linguagens da Microsoft, já há mais de 20 outras (Perl, Cobol, Pascal, etc) que podem usar esse ambiente.

**MSIL** - *Microsoft Intermediate Language*. Quando se compila uma aplicação .NET, ela é convertida para uma linguagem intermediária, a MSIL, um conjunto de instruções independentes de CPU. Na hora de executar o programa, um novo compilador, chamado *Just-in-time* (JIT) *Compiler*, o converte para o código nativo, ou seja, específico para o processador da máquina.

**MANAGED CODE** - Código administrado, ou seja, código escrito para rodar com o *runtime* do VS.NET. No VS.NET, somente o C++ produz programas que não dependem do *runtime*, o chamado *Unmanaged code*.

## **Visão geral do .NET Framework**

É uma tecnologia que dá suporte à compilação e à execução da próxima geração de aplicativos e serviços Web XML. O *.NET Framework* foi criado para atender os seguintes objetivos:

* Para fornecer um ambiente de programação orientada a objetos consistente, quer o código objeto seja armazenado e executado localmente ou remotamente.
* Fornecer um ambiente de execução que minimize conflitos de versionamento de publicação.
* Fornecer um ambiente de execução que promova a execução segura de código criado por desconhecidos ou código de terceiros com baixo nível de confiança
* Para fornecer um ambiente de execução que elimina os problemas de desempenho dos ambientes interpretados ou com scripts.
* Para tornar a experiência do desenvolvedor consistente, através dos diversos tipos de aplicativos, como aplicativos baseados no Windows e Web.
* Para executar toda comunicação usando padrões da indústria, assim garantindo que códigos baseados no *.NET Framework* possam se integrar a qualquer outro código.

## **CLR (Common Language Runtime)**

O .NET Framework tem um ambiente de tempo de execução chamado de *Common Language Runtime*, que executa o código e provê serviços que tornam o processo de desenvolvimento mais fácil.

Compiladores e ferramentas expõem as funcionalidades do CLR e habilitam você escrever código que se beneficia desse ambiente de execução gerenciado.

Código que você desenvolve com um compilador de linguagem que tem como alvo o *runtime* é chamado de código gerenciado; ele se beneficia de recursos como integração entre linguagens, tratamento de exceção entre linguagens, segurança aprimorada, suporte a versionamento e implantação, um modelo simplificado para interação entre componentes, e serviços de depuração e de perfil.

## **CLR (Common Language Runtime)**

O *runtime* automaticamente trata o leiaute de objetos e gerencia referências a objetos, liberando-os quando eles não estão sendo usados. Objetos cujos tempos de vida são gerenciados dessa forma são chamados de dados gerenciados.

A coleta de lixo elimina perdas de memória, bem como alguns outros erros de programação comuns. Se o código for gerenciado, você poderá usar dados gerenciados, dados não gerenciados ou ambos no seu aplicativo do *.NET Framework*.

Devido ao fato de compiladores de linguagens fornecerem seus próprios tipos, como tipos primitivos, você nem sempre pode saber (ou precisa saber) se seus dados estão sendo gerenciados.

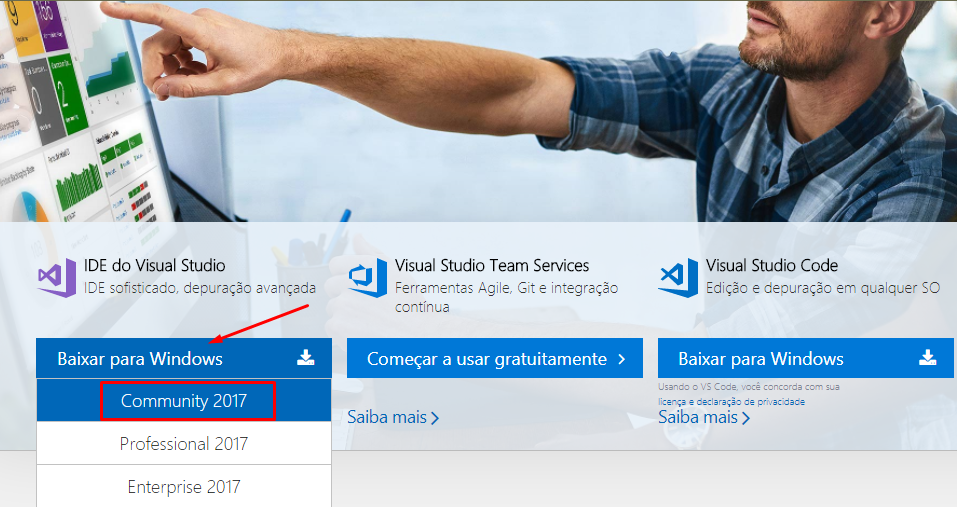
## **CLR (Common Language Runtime)**

O tempo de execução oferece os seguintes benefícios:

* Melhorias de desempenho.
* A capacidade de utilizar facilmente componentes desenvolvidos em outras línguas.
* Tipos extensíveis fornecidos por uma biblioteca de classes.
* Recursos da linguagem como a herança, interfaces, e sobrecarga para programação orientada a objeto.
* Suporte a definição explícita de threads que permite a criação de aplicações *multithread* escalonáveis .
* Suporte a manipulação estruturada de exceções .
* Suporte a atributos personalizados.
* Coleta de lixo.
* Uso de *delegates* em vez de ponteiros de função para aumentar a segurança de tipos e segurança.

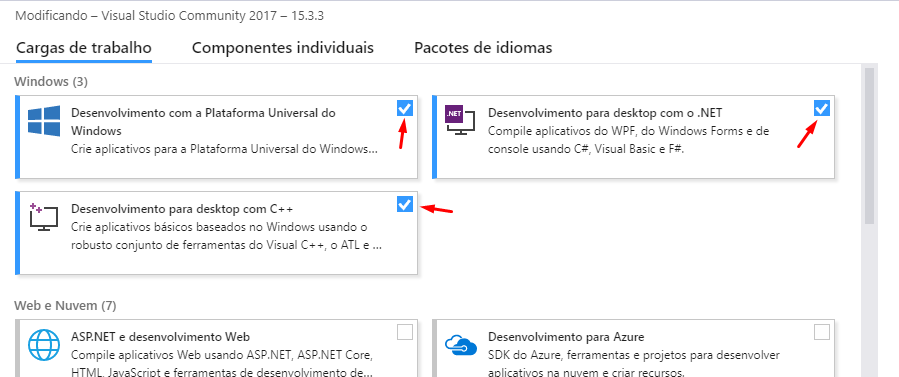
## **Download do Visual Studio**

Para realizar o download do Microsoft Visual Studio, basta acessar [https://www.visualstudio.com/pt-br](https://www.visualstudio.com/pt-br/), clicar em [IDE do Visual Studio](https://www.visualstudio.com/pt-br/vs/" \t "_self)

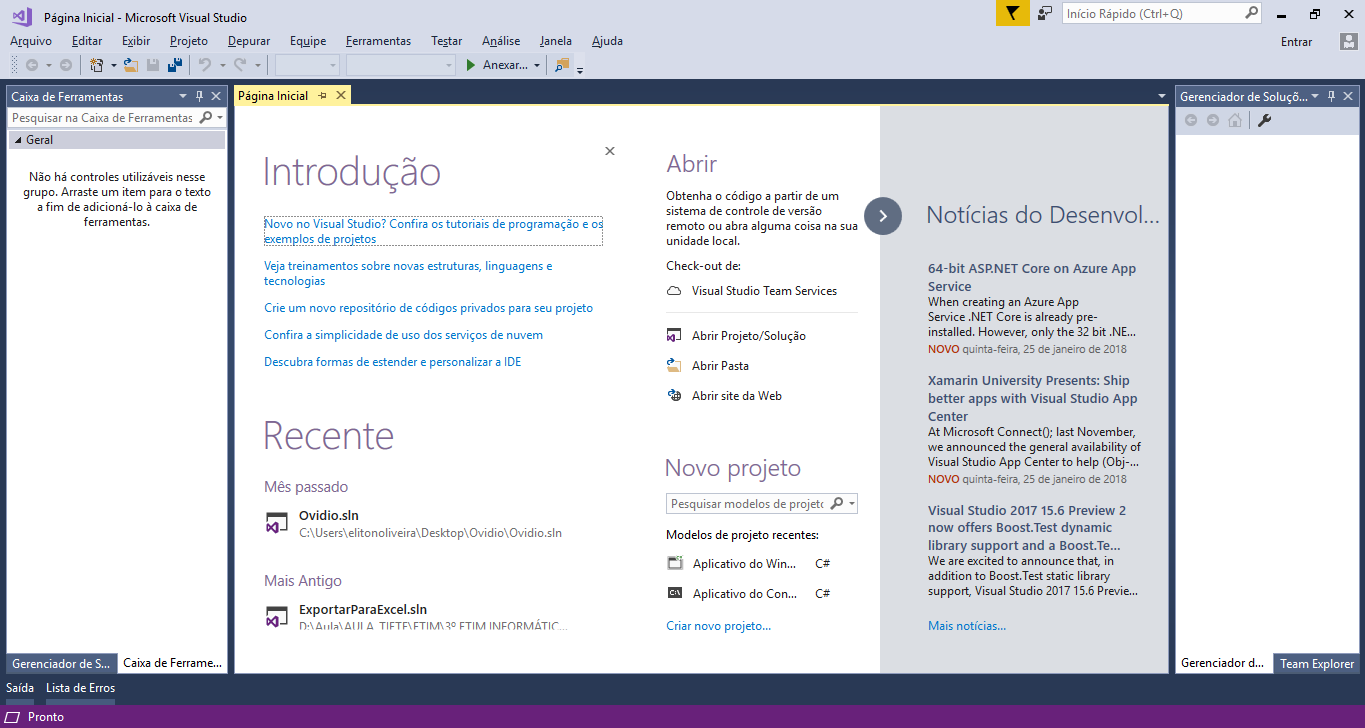


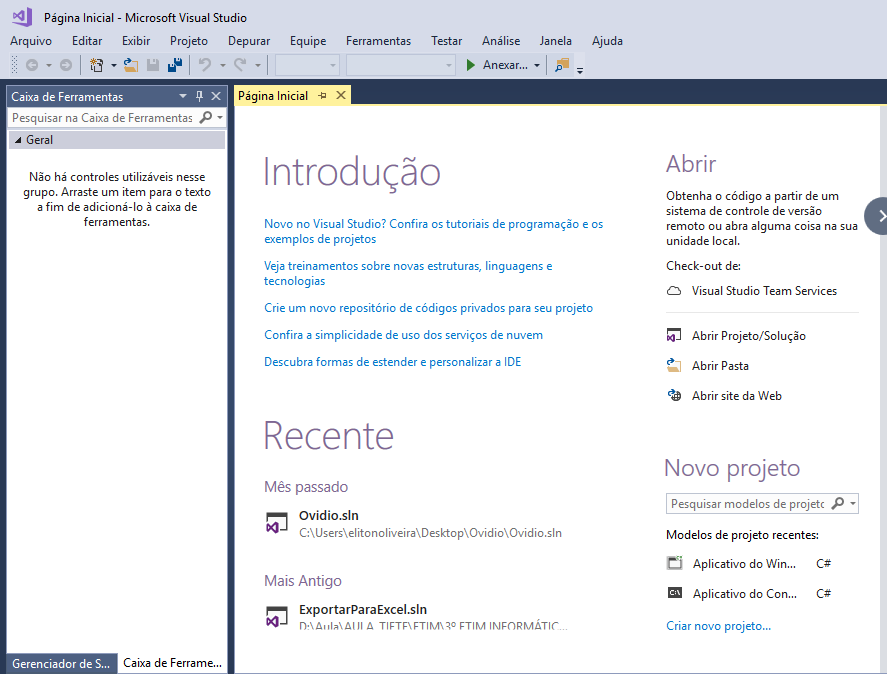
## **Instalação do Visual Studio**

Habilite os seguintes componentes



## **Conhecendo o ambiente – Tela inicial**





Barra de Menu e

Barra de

Ferramentas

Informações,

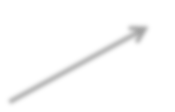
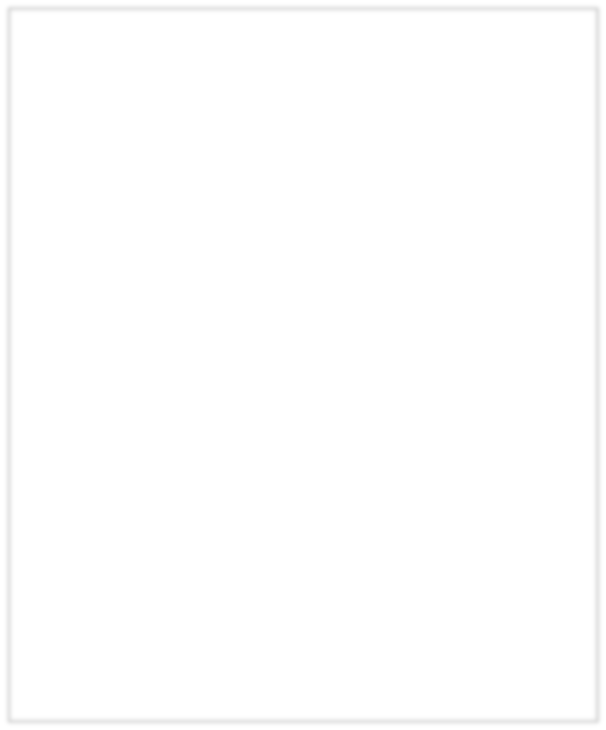
dicas e

referências de

documentação

sobre as

tecnologias



## **Criando o primeiro projeto**

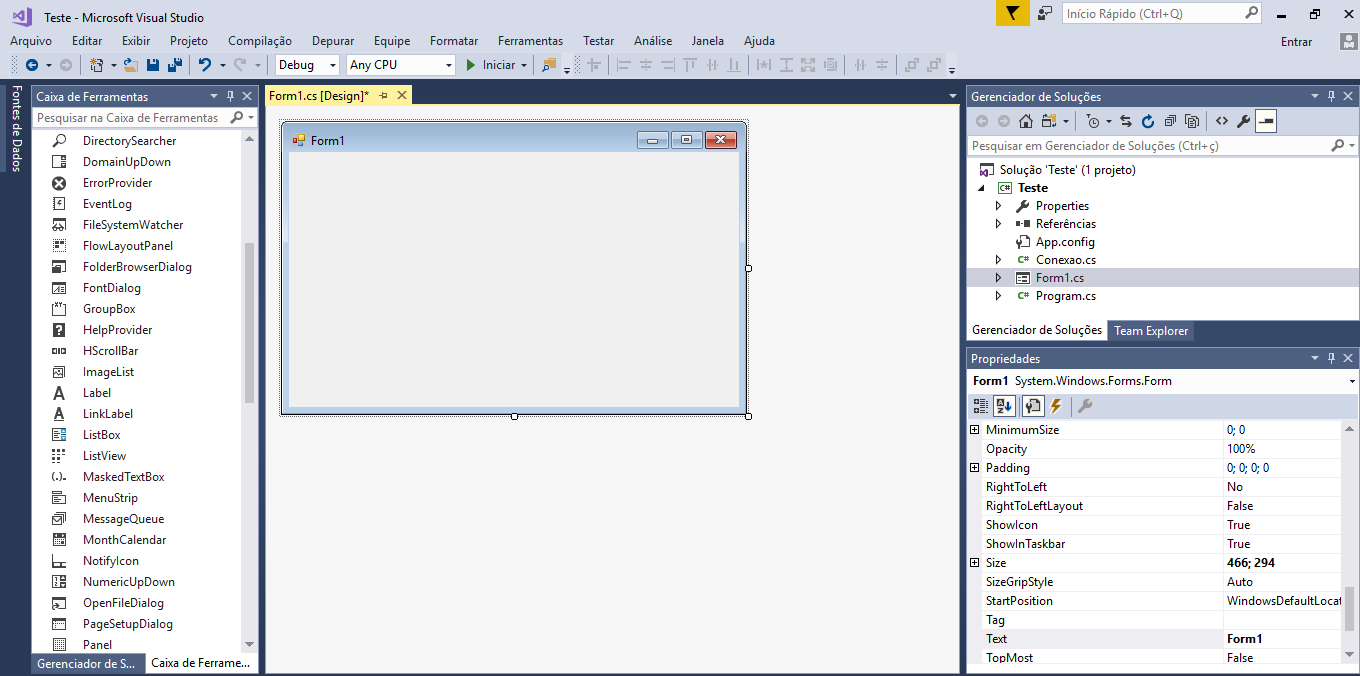
## **Criando o primeiro projeto**



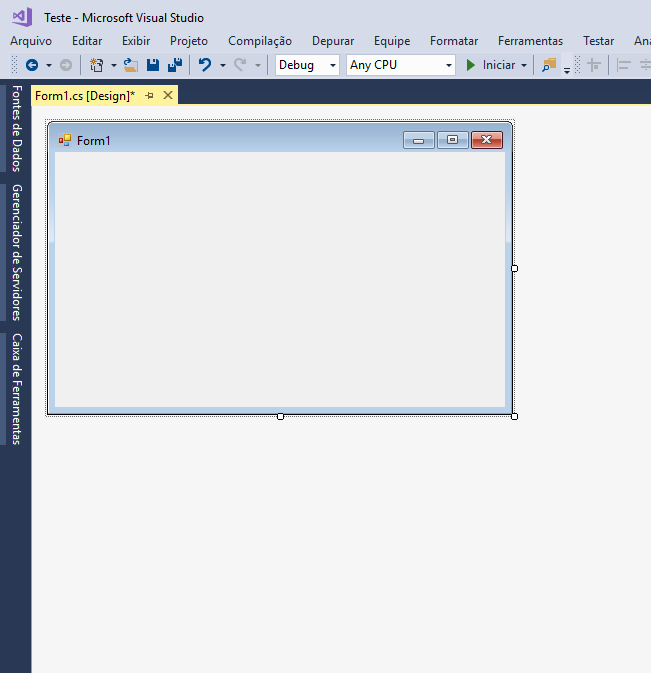
## **Criando o primeiro projeto**

1. Selecione **Visual C#**
2. Selecione ***Windows***
3. Selecione ***Windows Forms Aplication***
4. Coloque um nome para o projeto e indique o local para salvá-lo.
5. Selecione ***Create diretory for solution*** (Deixar selecionado para criação automática da estrutura do projeto).
6. Depois clique em ***OK***

## **Ambiente do primeiro projeto**



## **Exibir Caixa de ferramentas**

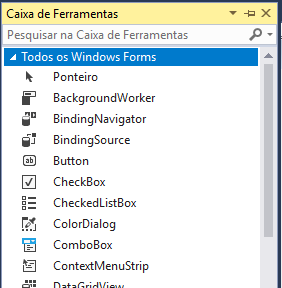
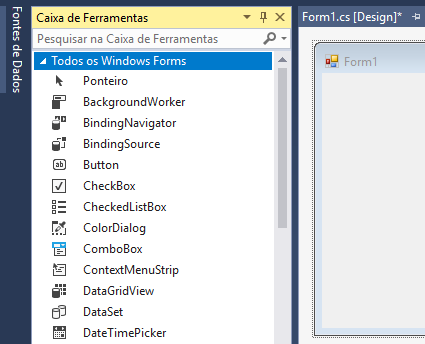


Clique para abrir a caixa de

ferramentas, onde constam os

componentes.

## **Exibir Caixa de ferramentas**





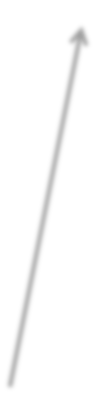
Clique em

*auto hide*

, para fixar

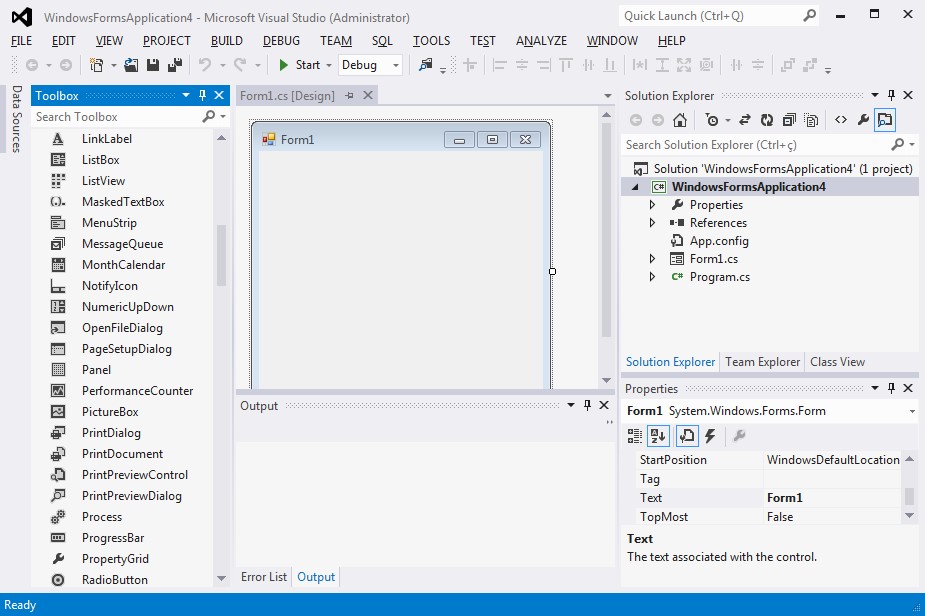
a barra de ferramentas na

janela

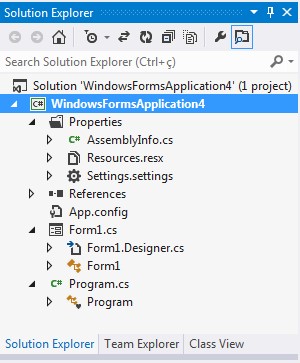


Logo após o clique, a barra de ferramentas ficará fixada na janela.

## **Identificando as áreas do ambiente**



## **Solution Explorer**

Esta janela mostra a estrutura de pastas e os arquivos que fazem parte do seu projeto.

Seu comportamento é como o

Explorer do Windows, permitindo criar, excluir e importar arquivos.

O arquivo principal de uma aplicação é armazenado no disco como um arquivo do tipo “.sln”.

Um projeto é armazenado em um arquivo do tipo “.csproj”.

A seção Solution Explorer de um projeto possui quatro itens:

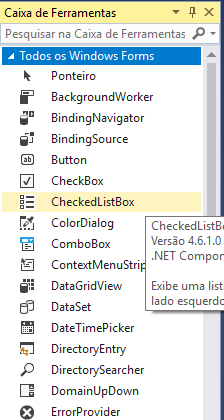
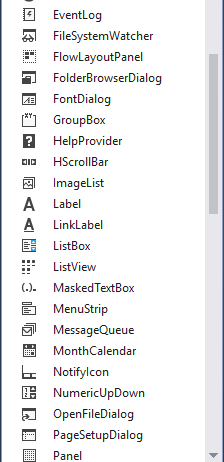
* + **Properties** (propriedades) – contém arquivos de configuração da solução e do projeto como o AssemblieInfo.cs, que define informações de configuração do projeto.

* + **References** (referências) – a lista de fragmentos de códigos compilados (assemblies) referenciados pelo projeto.

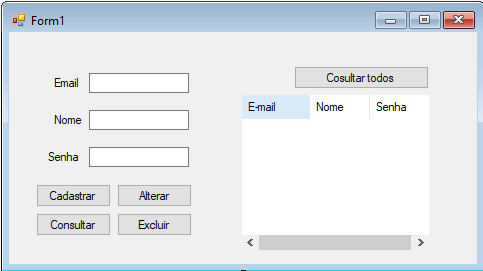
* + **Form1.cs** – um arquivo contendo a classe Form criada por padrão para a aplicação.

* + **Program.cs** – Um arquivo contendo a classe do programa criada por padrão para a aplicação.

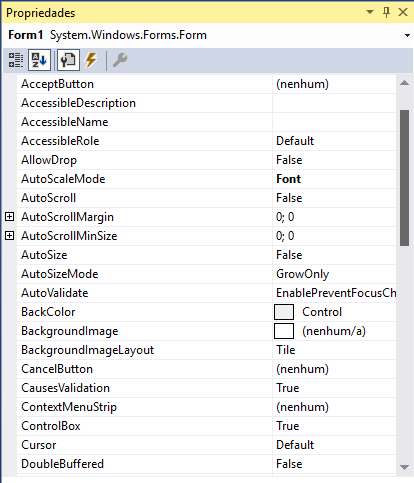
## **Toolbox**

Esta janela contém os componentes necessários para o desenvolvimento de formulários.

## **Form**

Esta janela é a nossa aplicação, a qual receberá a programação e os componentes da toolbox.

## **Properties**

Esta janela permite alterar as propriedades dos componentes, as quais poderão estar organizadas por categoria, ordem alfabética, propriedades ou eventos.

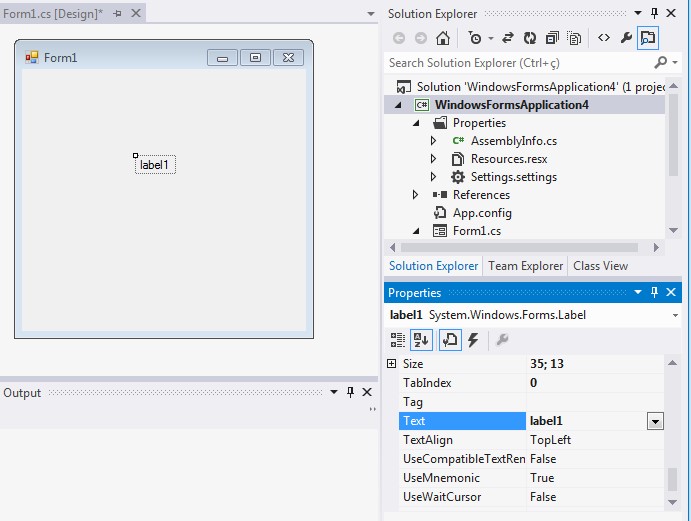
## **Inserindo um Componente - componente Label.**



Para inserir um componente, basta selecionar algum na Caixa de Ferramentas (Toolbox) e depois clicar dentro do formulário, ou um duplo clique para inserir diretamente. Neste exemplo foi escolhido o componente label.

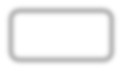
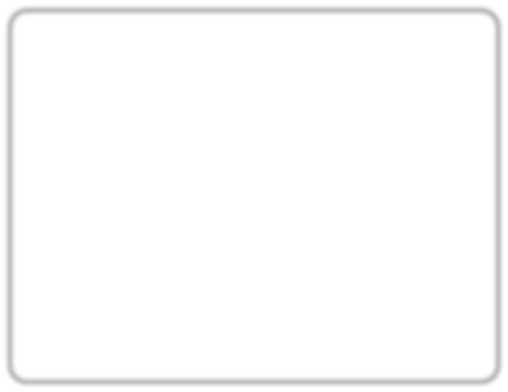


## **Propriedades do Componente**



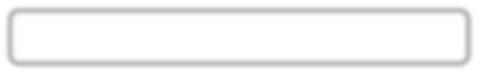
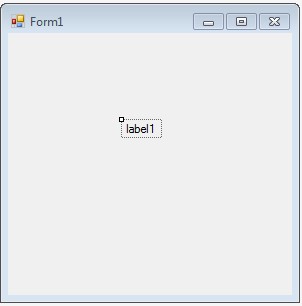
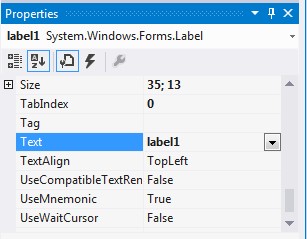
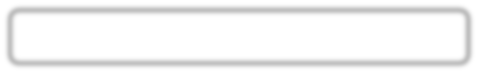
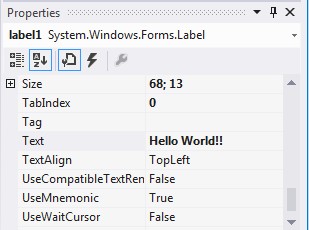
Quando selecionar o componente dentro do formulário, todas suas propriedades são apresentadas na janela ***Properties***.

.



## **Alterando a propriedade Text**

Vamos alterar o texto deste ***Label***,



colocando o famoso “Hello Word!!!”.

Para isso na caixa de propriedades basta

alterar o campo ***Text***, conforme indicado.

## **Algumas propriedades do Label**

**AutoSize** – alterar para ***False*** permitindo que o tamanho do ***Label*** possa ser definido pelo programador.

**(name)** – para definir um nome para o componente, que visa facilitar a identificação durante a programação **BackColor** – define a cor de fundo do componente

**Font** – define a fonte, estilo e tamanho do texto

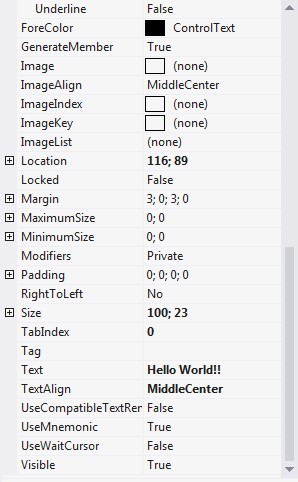
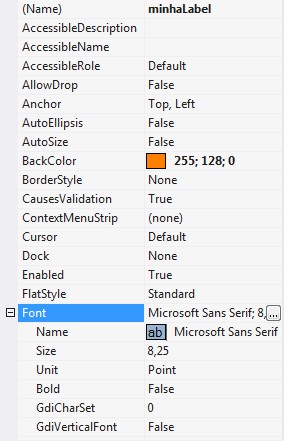
**ForeColor** – define a cor da letra do componente

**TextAlign** – define o alinhamento do texto dentro do label

**Image** – permite incluir uma imagem no label

## **Alterando algumas propriedades**

Alterar as seguintes propriedades com os respectivos valores: **AutoSize = False**



**(name) = minhaLabel**

**BackColor = selecione a cor Laranja**

**Font = Size = 20**

**ForeColor = selecione a cor Preta**

**TextAlign = MiddleCenter**

## **Executando o Programa**

Para que possamos executar

o programa, basta clicar em

*Start*

na barra de

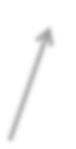
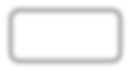
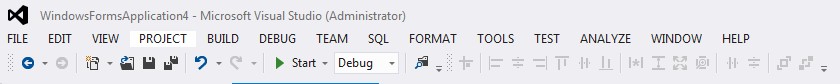
ferramentas.

E se estiver tudo OK, será

exibida a seguinte janela,

contendo o formulário

Form1.



## **Aprimorando o Hello World!!!**

Para testar suas habilidades, modifique algumas propriedades deste formulário, como por

exemplo, alterar a cor de fundo do formulário, o texto do

formulário, a cor do texto da

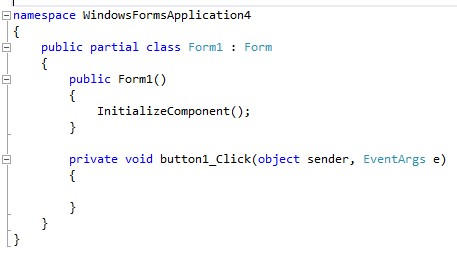
Label, o tipo de fonte do texto entre outras.

Na sequência adicione um botão, que servirá para finalizar a aplicação, como por exemplo…..

## **Evento no Botão**

Clicando duas vezes no botão, abrirá a área de codificação do componente.

Nesta área iremos programar o evento (ação) click que servirá para finalzar a aplicação.

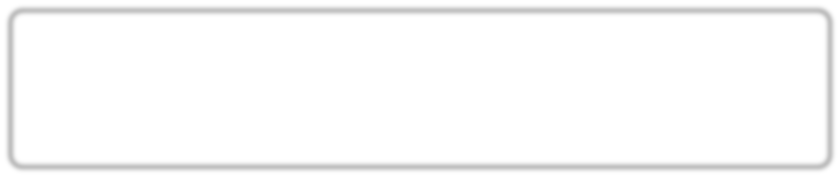
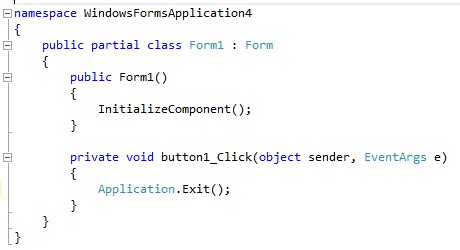


## **Evento no Botão**

Dentro do evento Click, iremos colocar o comando

**Application.Exit();**

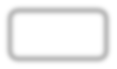
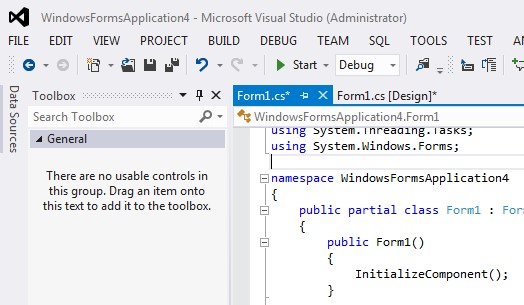
que será responsável por finalizar a aplicação.



## **Salvando e executando o programa**

Clique em Salvar todos.

Em seguida clique em **Start** ou pressione a tecla **F5** para executar. Dessa vez, quando clicar em **Sair** a aplicação será finalizada.



## **A propriedade (name)**

A propriedade (name) dos componentes serve para determinarmos um nome específico que remos adotar como referência ao componente, para o tratamento de eventos e alteração de alguma propriedade através da codificação.

Não podemos definir um mesmo nome para mais de um componente.

Sendo esta a maneira de tornar cada componente, único na aplicação.

## **Mais Eventos**

Vamos agora criar outro projeto, onde deverá conter os seguintes componentes, de acordo com a imagem.



***Label***

**Alterar as propriedades:**

**name) = minhaLabel**

**(**

**Text = Hello World!!!**

***Button***

**Alterar as propriedades:**

**(**

**name) = btnSair**

**Text = Sair**

**name) = btnLimpar**

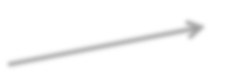
**(**

**Text = Limpar**

**name) = btnExibir**

**(**

**Text = Exibir**



## **Aplicando Ações**

As ações que serão aplicadas aos eventos dos respectivos botões são:

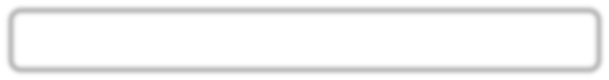
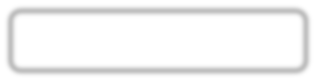
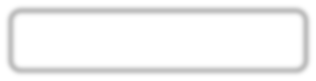
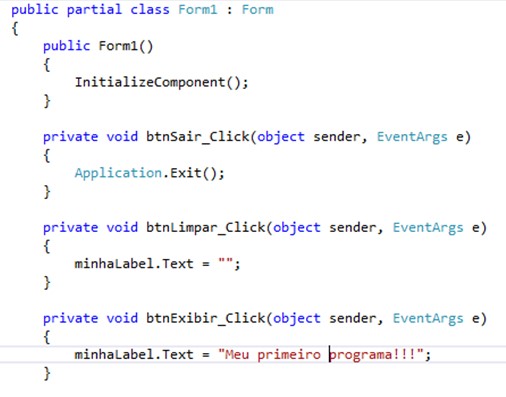
**Sair**: Deverá ser encerrada a aplicação

**Limpar**: Deverá limpar o conteúdo da minhaLabel

**Exibir**: Deverá exibir o texto “Meu primeiro programa!!!” na minhaLabel

## **Aplicando Ações**

Dando um duplo clique em cada botão, abrirá a área de codificação para implementarmos as seguintes ações.



## **Estudos dos componentes: MessageBox, Label, TextBox, Button, e seus métodos, eventos e propriedades.**

## **MessageBox**

Exibe uma caixa de mensagem que pode conter texto, botões e símbolos que possam informar e instruir o usuário.

Crie

um

formulário

Quando

clicar

em

mensagem

contendo

cada

um

dos

botões,

esta

interface

gráfica

.

uma

caixa

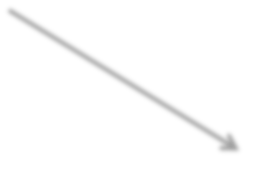
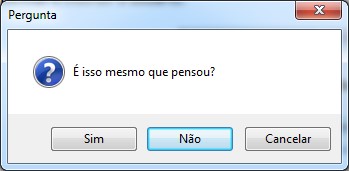
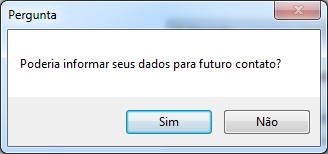
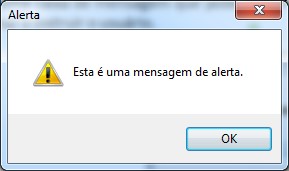
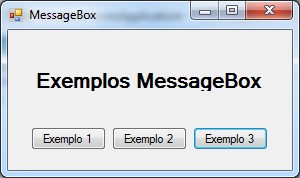
de

mensagem

será

exibida

.



Quando clicar em Exemplo 1, estará sendo acionado o seguinte evento:

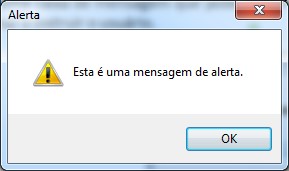
private void btnEx1\_Click(object sender, EventArgs e)

{

MessageBox.Show("Esta é uma mensagem de alerta.", "Alerta",

MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}



Quando clicar em Exemplo 2, estará sendo acionado o seguinte evento:

private void btnEx2\_Click(object sender, EventArgs e)

{

string message = "Poderia informar seus dados para futuro contato?";

string caption = "Pergunta";

MessageBoxButtons buttons = MessageBoxButtons.YesNo;

DialogResult result;

result = MessageBox.Show(message, caption, buttons);

if (result == System.Windows.Forms.DialogResult.Yes)

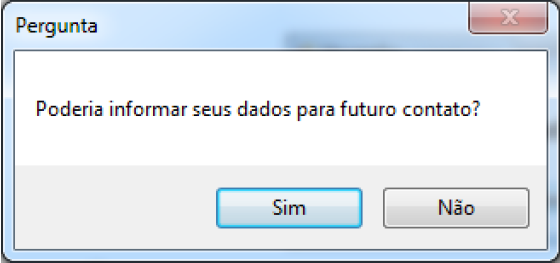
{

MessageBox.Show("OK Obrigado! Em breve entrarei em contato!", "Resposta SIM", MessageBoxButtons.OK);

}

else

{

MessageBox.Show("OK Sem problemas! ", "Resposta NÃO", MessageBoxButtons.OK);

}

}

Quando clicar em Exemplo 3, estará sendo acionado o seguinte evento:

private void btnEx3\_Click(object sender, EventArgs e)

{

DialogResult resultado = MessageBox.Show("É isso mesmo que pensou?",

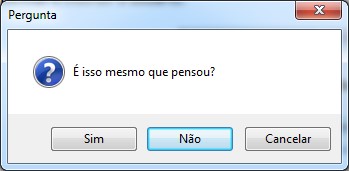
"Pergunta",

MessageBoxButtons.YesNoCancel,

MessageBoxIcon.Question,

MessageBoxDefaultButton.Button2);

}



## **MessageBox**

|  |  |
| --- | --- |
| **Métodos** | **Descrição** |
| Show(String) | Exibe uma caixa de mensagem com o texto especificado. |
| Show(IWin32Window, String, String, MessageBoxButtons, MessageBoxIcon, MessageBoxDefaultButton, MessageBoxOptions, String, HelpNavigator, Object) | Exibe uma caixa de mensagem com o texto especificado, legenda, botões, ícone, botão padrão, opções e botão de Ajuda, usando o arquivo de Ajuda especificado, HelpNavigatore o tópico da Ajuda. |
| Show(String, String) | Exibe uma caixa de mensagem com o texto especificado e a legenda. |
| Show(String, String, MessageBoxButtons) | Exibe uma caixa de mensagem com botões, legenda e texto especificado. |
| Show(IWin32Window, String, String) | Exibe uma caixa de mensagem na frente do objeto  especificado e com o texto especificado e a legenda. |

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.messagebox.show%28v=vs.90%29.aspx>

## **MessageBoxButtons**

|  |  |
| --- | --- |
| **Membros** | **Descrição** |
| OK | A caixa de mensagem contém um botão OK. |
| OKCancel | A caixa de mensagem contém botões OK e Cancelar. |
| AbortRetryIgnore | A caixa de mensagem contém botões Abortar, repetir e ignorar. |
| YesNoCancel | A caixa de mensagem conterá Sim, não e Cancelar botões. |
| YesNo | A caixa de mensagem contém botões Sim e não. |
| RetryCancel | A caixa de mensagem contém botões de Repetir e Cancelar. |

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxbuttons%28v=vs.90%29.aspx>

## **MessageBoxIcon**

|  |  |
| --- | --- |
| **Membros** | **Descrição** |
| Question | Símbolo que consiste em um ponto de interrogação em um círculo. |
| Exclamation | Símbolo que consiste em um ponto de exclamação dentro de um triângulo com um plano de fundo amarelo. |
| Stop | Símbolo que consiste em X branco em um círculo com um plano de fundo vermelho. |
| Error | Símbolo que consiste em X branco em um círculo com um plano de fundo vermelho. |
| Warning | Símbolo que consiste em um ponto de exclamação dentro de um triângulo com um plano de fundo amarelo. |
| Information | Símbolo consiste em uma letra em minúsculas i em um círculo. |

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxicon%28v=vs.90%29.aspx>

## **MessageBoxDefaultButton**



|  |  |
| --- | --- |
| **Membros** | **Descrição** |
| Button1 | O primeiro botão na caixa de mensagem é o botão padrão. |
| Button2 | O segundo botão na caixa de mensagem é o botão padrão. |
| Button3 | O terceiro botão na caixa de mensagem é o botão padrão. |

[http://msdn.microsoft.com/pt-](http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxdefaultbutton%28v=vs.90%29.aspx)

[br/library/system.windows.forms.messageboxdefaultbutton%28v=vs.90%29.aspx](http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxdefaultbutton%28v=vs.90%29.aspx)



Especifica identificadores para indicar o valor retornado de uma caixa de diálogo.

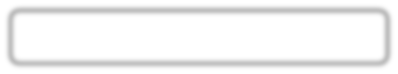
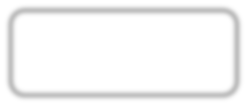
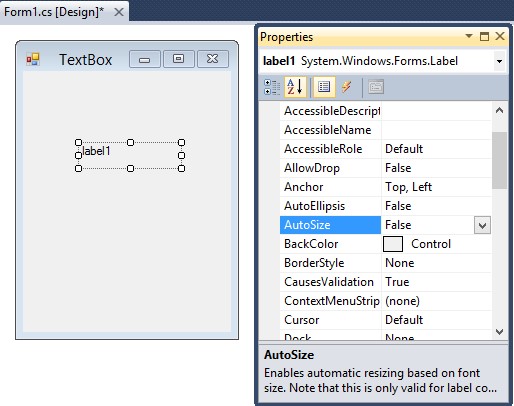
|  |  |
| --- | --- |
| **Membros** | **Descrição** |
| None | Nothing é retornado pelo formulário da caixa de diálogo. |
| OK | O valor retornado de caixa de diálogo é OK |
| Cancel | O valor retornado de caixa de diálogo é Cancel |
| Abort | O valor retornado de caixa de diálogo é Abort |
| Retry | O valor retornado de caixa de diálogo é Retry |
| Ignore | O valor retornado de caixa de diálogo é Ignore |

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.dialogresult%28v=vs.90%29.aspx>

## **Label**



O Label é usado para adicionar texto como forma de adicionar etiquetas para identificar a funcionalidade de outros controles. Arraste um controle de rótulo da caixa de ferramentas para o formulário. Por padrão, ele terá um texto inicial **label1**.



## **Label**

|  |  |
| --- | --- |
| **Propriedade** | **Descrição** |
| AutoSize | Se for verdade, o tamanho das fronteiras do rótulo no designer será redimensionada automaticamente dependendo do texto dentro dele. | |
| BorderStyle | Especifica o tipo de borda ao redor do rótulo. | |
| Font | Usado para alterar as propriedades de fonte do texto dentro do controle de rótulo. | |
| Text | O texto da etiqueta. | |
| TextAlign | O alinhamento do texto dentro do controle Label | |

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.label(v=vs.110).aspx>

## **Label**

Neste exemplos estão sendo utilizadas as propriedades:

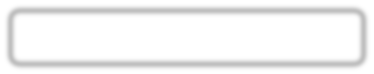
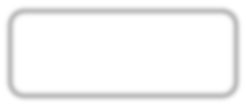
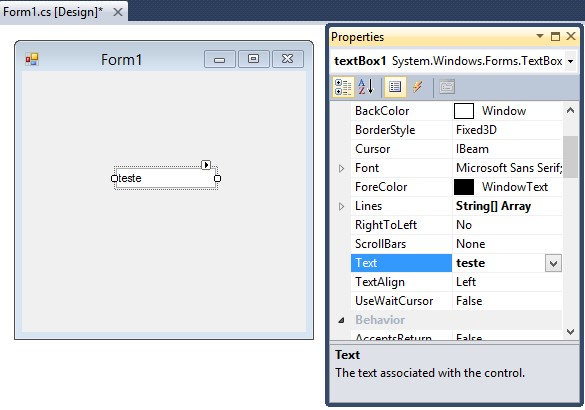


* AutoSize
* BorderLayout
* BackColor
* Font
* ForeColor
* Size
* TextAlign

### 

## **TextBox**

É um componente básico de entrada de dados em formulário. Basta o usuário digitar os dados de entrada dentro da caixa de texto. O texto digitado pode ser acessado usando a propriedade **Text**.



|  |  |
| --- | --- |
| **Propriedade** | **Descrição** |
| AcceptsReturn | Usado com várias linhas. Diz se a tecla de retorno está incluída na entrada. O retorno será convertido em uma seqüência de escape \n. |
| Enabled | Selecionado como false para deixar a caixa de texto somente como leitura. |
| Font | As propriedades de fonte que serão usadas pela caixa de texto. |
| Lines | As linhas de texto em uma caixa de texto de várias linhas. |
| Multiline | Defina como true para permitir várias linhas em uma caixa de texto. |
| Text | Texto dentro da caixa de texto. |
| PasswordChar | Aceita um caractere que será usado para mascarar cada caractere digitado pelo usuário. |
| ReadOnly | Determina se o texto pode ser editado. |
| Visible | Determina se a caixa de texto é visível dentro do formulário. |
| WordWrap | Usado com várias linhas. Defina como true para permitir a quebra automática de palavras. |

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.textbox(v=vs.110).aspx>

## **TextBox**

Neste exemplo foram alteradas as propriedades:



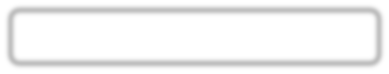
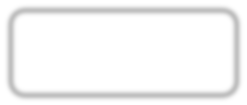
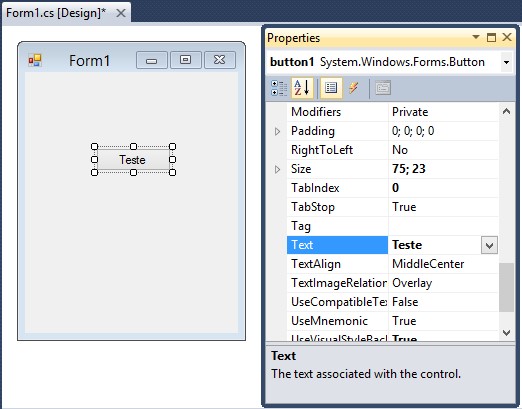
* BackColor
* Enable
* ForeColor
* PasswordChar
* Text
* WordWrap

### 

## **Button**



O Button é geralmente usado para executar comandos quando é clicado. Quando um botão é clicado, você determina os códigos que serão usados. São normalmente utilizados para confirmar ou cancelar uma ação, para executar ações diferentes.



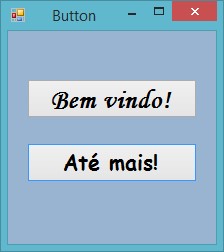
|  |  |
| --- | --- |
| **Propriedade** | **Descrição** |
| AutoEllipsis | Especifica se é necessário acrescentar pontos (...) quando o texto é muito longo e não pode cabe dentro do botão. |
| AutoSize | Especifica se o botão irá redimensionar automaticamente para caber seu conteúdo. |
| FlatStyle | Obtém ou define aparência plana ao controle botão. |
| Enabled | Se definido como false, o botão não pode ser clicado ou receber foco. |
| Image | Uma imagem opcional que você pode colocar dentro do componente. |
| ImageAlign | O alinhamento da imagem no botão. |
| Text | O texto dentro do botão. |
| Visible | Diz se o botão está visível ou não. |

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.button_properties(v=vs.110).aspx>

|  |  |
| --- | --- |
| **Evento** | **Descrição** |
| Click | Ocorre quando você clica no botão. |
| Leave | Ocorre quando o ponteiro do mouse sai do controle. |
| LocationChanged | Ocorre quando a localização do botão é alterado. |
| MouseDown | Ocorre quando o ponteiro do mouse está sobre o controle e um botão do mouse é pressionado. |
| MouseEnter | Ocorre quando o ponteiro do mouse entra no controle. |
| MouseHover | Ocorre quando o ponteiro do mouse é posicionado sobre o controle. |
| MouseUp | Ocorre quando o ponteiro do mouse está sobre o controle e um botão do mouse seja liberado. |

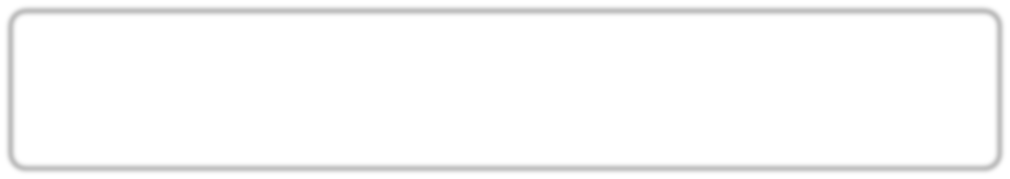
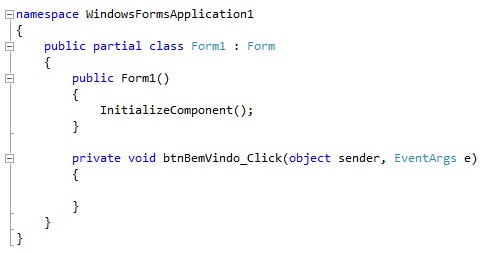
<http://msdn.microsoft.com/pt-br/library/system.windows.forms.button_events(v=vs.110).aspx>

## **Button**

Neste exemplo foram alteradas as propriedades:

* (name)
* Font
* Size
* Text

Para criar um evento quando o botão for clicado, basta dar um duplo clique no botão, que na sequencia aparecerá uma área para codificação. No caso dê um duplo clique no botão Aparecerá a seguinte tela:



É nesta área que

iremos codificar a

ação do botão

***Bem Vindo!***

Para criar um evento quando o botão for clicado, basta dar um duplo clique no botão, que na sequencia aparecerá uma área para codificação. No caso dê um duplo clique no botão

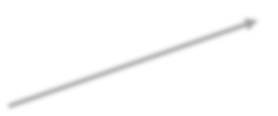
Aparecerá

a

seguinte

tela

:



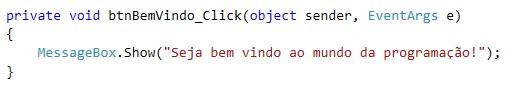
Este é o nome atributo na

propriedade

***(***

***name)***

do botão



Este é o comando que será executado quando o botão for clicado

Agora você poderá criar uma ação para o botão 

## **Referências**

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.messagebox.show%28v=vs.90%29.aspx>

[http://msdn.microsoft.com/pt-](http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxbuttons%28v=vs.90%29.aspx)

[br/library/system.windows.forms.messageboxbuttons%28v=vs.90%29.aspx](http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxbuttons%28v=vs.90%29.aspx)

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxicon%28v=vs.90%29.aspx>

[http://msdn.microsoft.com/pt-](http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxdefaultbutton%28v=vs.90%29.aspx)

[br/library/system.windows.forms.messageboxdefaultbutton%28v=vs.90%29.aspx](http://msdn.microsoft.com/pt-br/library/system.windows.forms.messageboxdefaultbutton%28v=vs.90%29.aspx)

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.dialogresult%28v=vs.90%29.aspx>

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.label(v=vs.110).aspx>

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.textbox(v=vs.110).aspx>

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.button_properties(v=vs.110).aspx>

<http://msdn.microsoft.com/pt-br/library/system.windows.forms.button_events(v=vs.110).aspx>

[http://msdn.microsoft.com/ptbr/library/zw4w595w%28v=vs.110%29.aspx](http://msdn.microsoft.com/pt-br/library/zw4w595w%28v=vs.110%29.aspx)