Esses arquivos estão na janela Solution Explorer.

- Form1.cs: contém o código C# que define o comportamento do formulário.
- Form.Designer.cs: contém o código que define o formulário e seus objetos.
- Program.cs: contém o código que inicia o programa e exibe o formulário.

Conversão de tipos numéricos para string e de string para tipos numéricos:

```
int num;

num = Convert.ToInt32(txtNumero.Text);

//ou

num = Int32.Parse(txtNumero.Text);
```

```
txtNumero.Text = Convert.ToString(num);

//ou

txtNumero.Text = num.ToString();
```

```
txtNumero.Text = Convert.ToString(num);

//ou

txtNumero.Text = num.ToString();
```

Conversão Implícita

O nome já diz: implícito. Ao declarar esse tipo de conversão, o compilador identifica que ambos são de tipos compatíveis, sendo assim nenhuma intervenção na hora do build é feita e há a garantia que nenhuma informação seja perdida durante a conversão.
Exemplo:

```
static void Main(string[] args)
    {
        sbyte a = 10;
        short b = a;
        int c = b;
        long d = c;
        float e = d;
        double f = e;
```

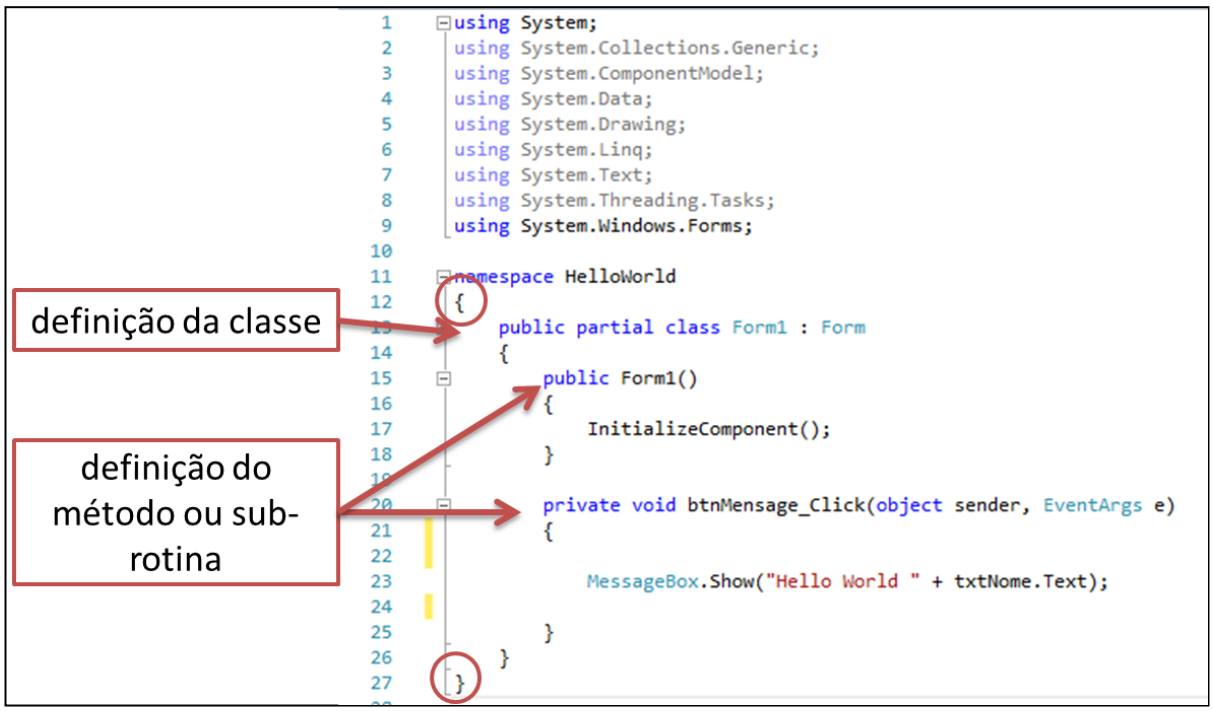
```
        Console.Write(f);
        Console.Read();
    }
```

Conversão Explicita

Essa conversão deixa o código mais legível. A forma de se declarar uma conversão desse tipo facilita no entendimento da necessidade. Para realizar este tipo, você deve declarar à frente da variável da direita e entre parênteses o tipo de dado que deseja.
Exemplo:

```
static void Main(string[] args)
    {
        sbyte a = 10;
        short b = (short)a;
        int c = (int)b;
        long d = (long)c;
        float e = (long)d;
        double f = (float)e;
        Console.Write(f);
        Console.Read();
    }
```

Estrutura da Linguagem C#



Regras da Linguagem C#:

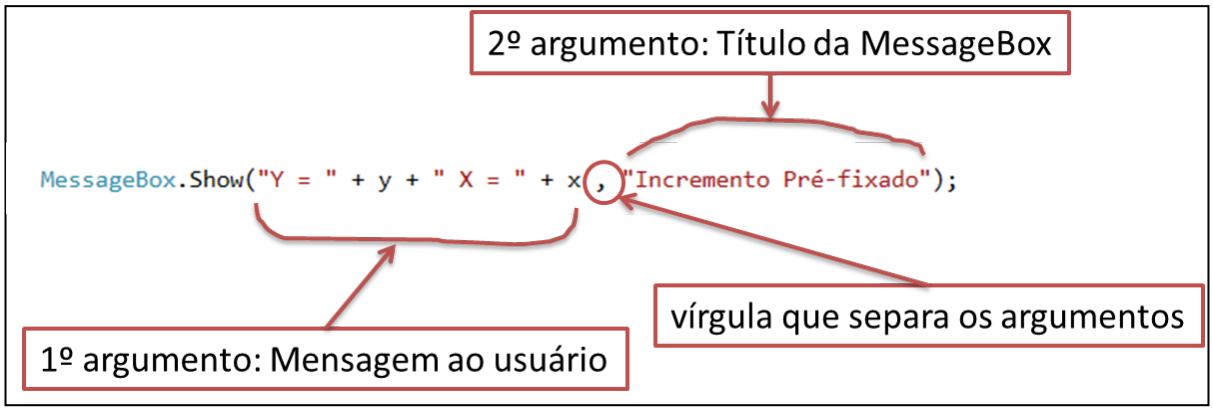
O C# é case-sensitive, ou seja, o C# faz distinção entre letras maiúsculas e minúsculas, portanto, Total é diferente de total, que é diferente de toTal.

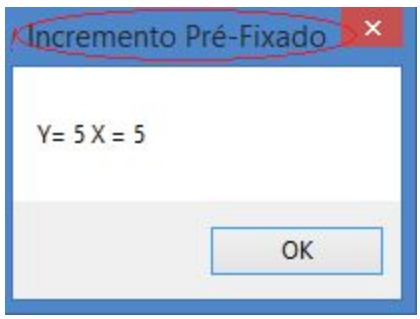
//MessageBox exemplo + concatena;

```
1. MessageBox.Show("Hello, World");//+ uma variável concatena
//Comentário de uma linha
/*Comentario
de várias
linhas*/
```

Exemplo de Concatenação:

```
1. int x, y;      //declaração das variáveis
   .....
MessageBox.Show("Y = "  +  y +  " X = " +  x , "Incremento
Pré-fixado");// A vírgula (,) depois da concatenação (+) Mostra o Título na
MessageBox do Programa
```





Erro de Sintaxe

Quando o compilador não reconhece uma instrução, dizemos que temos um erro de sintaxe. Normalmente o compilador fornece uma mensagem de erro para nos ajudar a localizar e corrigir o erro. Erros de sintaxe são violações das regras da linguagem.

Erro lógico

Erros lógicos são erros que impedem o programa de fazer o que você pretendia fazer. Seu código pode ser compilado e executado sem erros, mas o resultado de uma

operação pode produzir um resultado não esperado, ou seja, houve um erro de lógica.

Erro de tempo de execução

Erros em tempo de execução são erros que ocorrem enquanto o programa é executado. Eles normalmente ocorrem quando o programa tenta realizar uma operação que é impossível de ser executar. Por exemplo, divisão por zero.

Erro de compilação

Erros de compilação, também conhecido como erros de compilador, são erros que impedem o programa de ser executado.

Convenções de Nomenclatura:

Notação	Definição	Para nomear	Exemplos
camel casing	primeira letra de cada palavra em minúsculo	variáveis e parâmetros	nome, nomeCompleto, nomeDoLivro
Pascal casing	cada palavra inicia com letra maiúscula	métodos e outros identificadores	CalcularMedia( )

Controle	Mnemônico	Exemplo
Label	lbl	lblNome
TextBox	txt	txtNome
Button	btn	btnSalvar
Form	frm	frmPrincipal
ComboBox	cbo	cboUF

ListBox	lst	lstDetalhes
RadioButton	rdb	rdbAtivo
CheckBox	chk	chkLazer
DataGrid	dtg	dtgAlunos
Image	img	imgLogotipo
Panel	pnl	pnlDados
Table	tbl	tblCliente

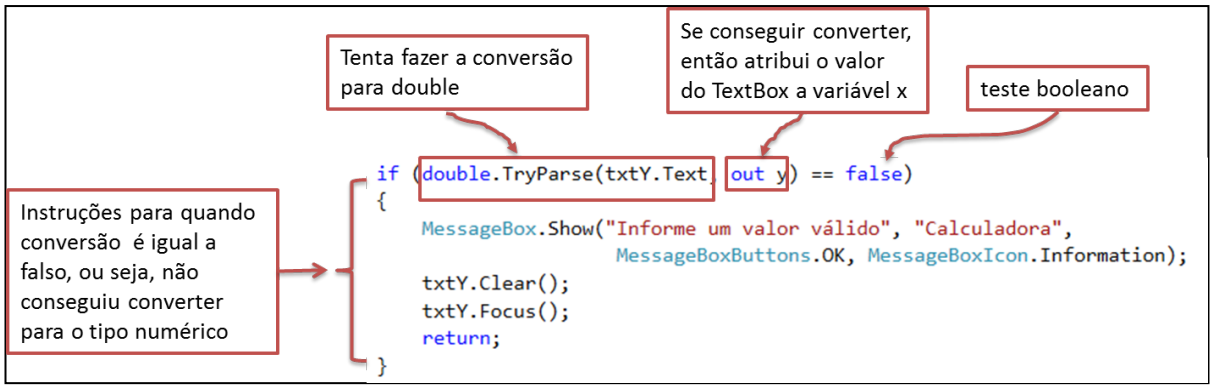
Palavras-chaves são palavras reservadas da linguagem que não podem ser utilizadas para nomear identificadores. Exemplo: **break**; **bool**;**if**;etc…
Palavras-chaves são palavras pré-definidas, que tem um significado especial para o compilador.

If-Else-Else If //Exemplo:

```
1.  if (rdbSomar.Checked == true)
2.  {
3.     total = x + y;
4.  }
5.
6.  else if (rdbSubtrair.Checked)
7.  {
8.     total = x - y;
9.  }
10.else if (rdbMultiplicar.Checked)
11.{
12.   total = x * y;
13.}
14.else
15.{
16.   //verifica se está sendo realizada divisão por zero
17.   if (y == 0)
18.   {
19.      txtTotal.Text = "Divisão por zero";
20.      return;     //sai da sub-rotina, não executa o código que está
     abaixo
21.   }
22.
23.   total = x / y;
24.}
```

No exemplo acima se trocasse os else if por somente if o programa iria validar somente o último if junto com o else, assim mesmo codificado certo somar e subtrair multiplicariam por conta do programa ignorá-los por serem if agora se forem else if não.

Outro Exemplo de If com explicações sobre 'out' em um programa calculadora:



return; //sai da sub-rotina, não executa o código que está abaixo

Entendendo o método TryParse

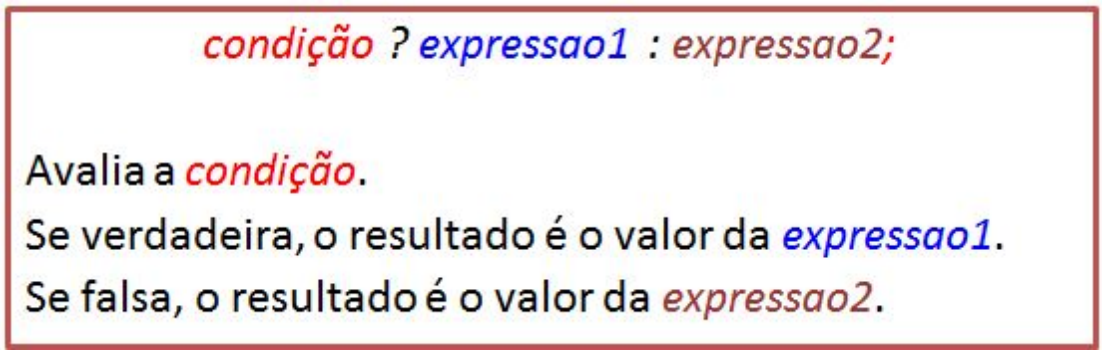
O método TryParse tenta fazer a conversão do texto do TextBox que é String para double. Esse método devolve um valor true (verdadeiro) se foi possível realizar a conversão de dados ou false (falso) caso contrário.

Na instrução condicional if está sendo verificado se o método TryParse retornou um valor falso (false), o que significa que não foi possível converter o texto para double. Nesse caso as instruções que estão dentro do corpo do if serão executadas (exibe o MessageBox, limpa o texto do TextBox, coloca o cursor no TextBox, sai da sub-rotina).

Se o método TryParse retornar verdadeiro (true), significa que foi possível converter o texto do TextBox para double, então o valor convertido será atribuído a variável x. O método TryParse usa a mesa analogia para os outros tipos de dados.

Operador ternário ?:

O operador condicional ternário (?:) está relacionado com a estrutura condicional if-else e é escrito em uma única linha.




```
x = num < 10 ? 0 : 1;
//Se num for menor que 10, a variável x recebe o valor 0, senão a variável x recebe o valor 1.
```

Estrutura switch

A estrutura de seleção switch consiste em uma *expressão de controle,*uma série de rótulos case e em um case default opcional. Cada rótulo (case ou default) contém instruções para serem executadas, caso esse rótulo seja selecionado.

Sintaxe da instrução switch em C#:

```
switch (expressãoControle)
{
    case condição:
        instruções;
        break;
    case condição:
        instruções;
        break;
    case condição:
        instruções;
        break;
    default:
        instruções;
        break;
}
```

A instrução switch só pode ser utilizada com os tipos de dados int, char ou string. Para os outros tipos de dados, inclusive os tipos double e float, você deverá utilizar a instrução if.

A estrutura switch não permite a utilização de dados do tipo float, nem operadores relacionais e lógicos.

Quando a condição é encontrada, as instruções desse case são executadas. A estrutura switch termina imediatamente com a instrução break.

default executa o bloco de código associado se nenhuma condição anterior for verdadeira.

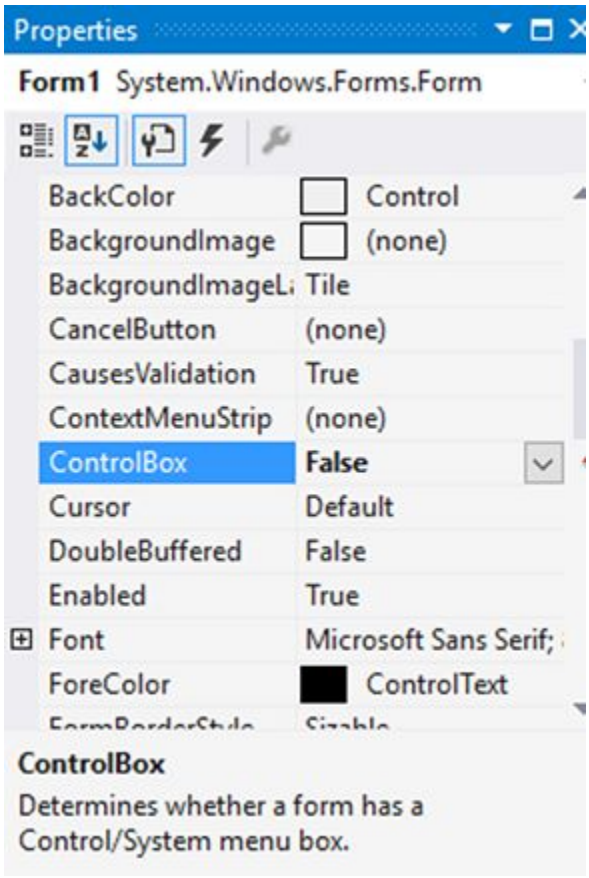
Quando vários case devem executar as mesmas instruções, utilizamos case o nome do case junto de dois pontos (:).

Exemplo:

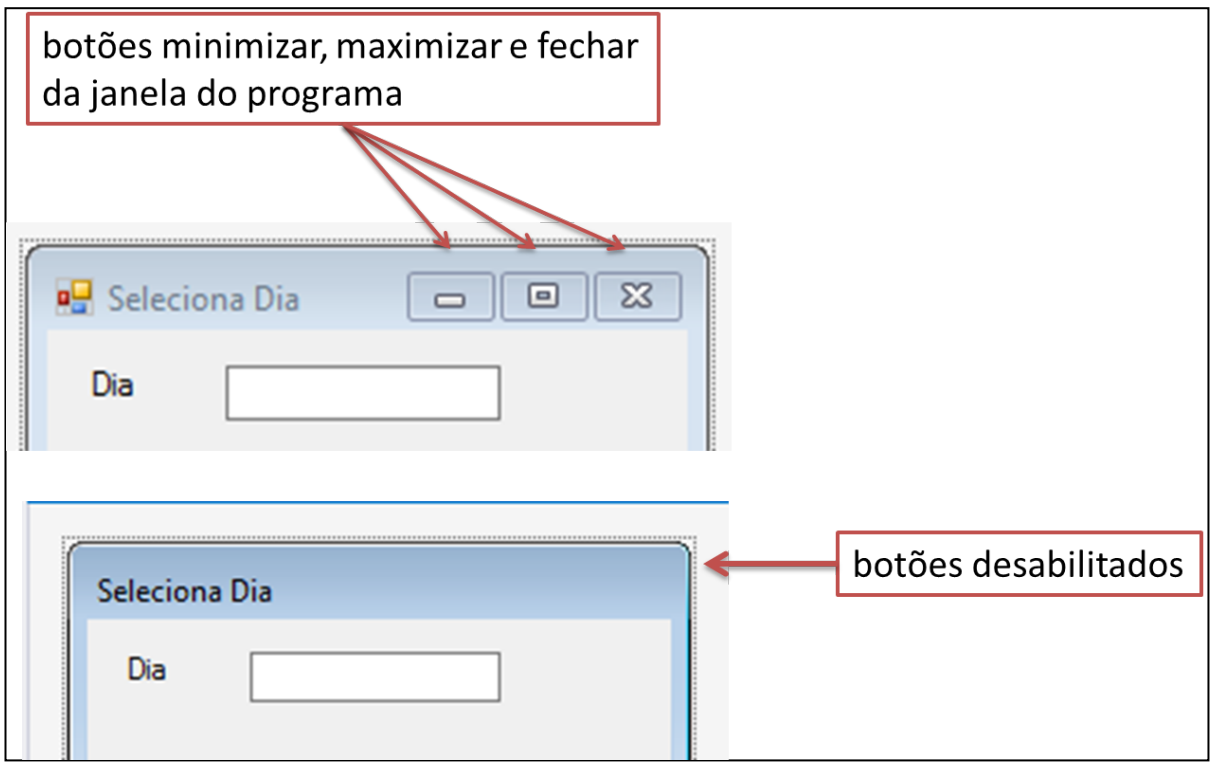
```
string animal="cavalo";
switch (animal)
{
    case "cavalo":
    case "gato":
        MessageBox.Show("Esse animal tem quatro patas");
        break;
    case "cobra":
        MessageBox.Show("Esse animal tem duas patas");
        break;
}
```

Propriedade ControlBox do Form

Para habilitar ou desabilitar os botões maximizar, minimizar e fechar de um Form (janela do Windows), alteramos a propriedade ControlBox do Form. O valor True, deixa habilitado os botões; o valor False, desabilita os botões.

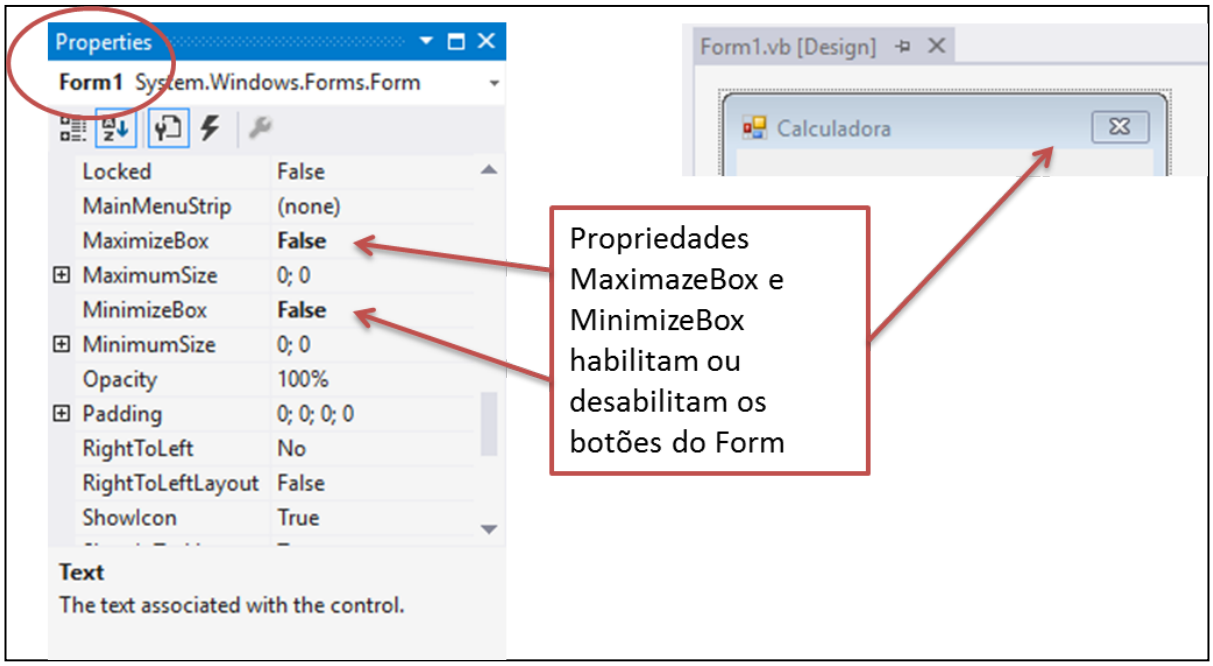


Depois disso a janela do form deverá ficar com os botões minimizar, maximizar e fechar desabilitados. Veja a figura a seguir.



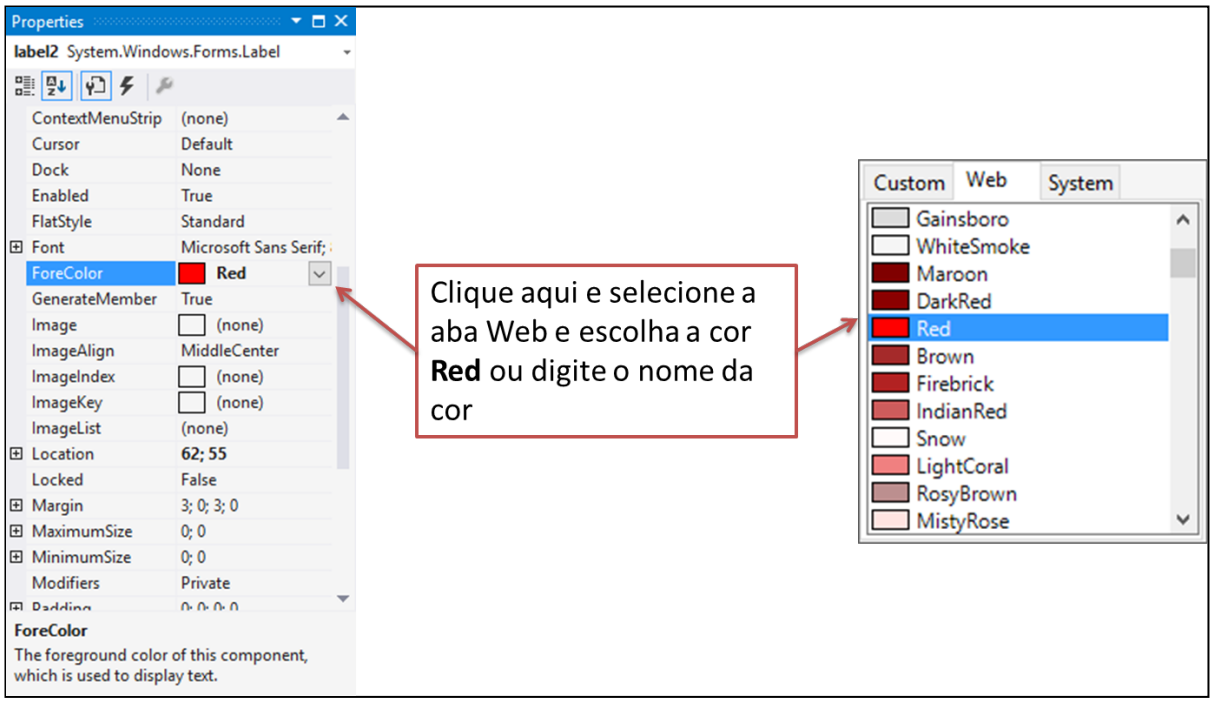
Propriedades MaximizeBox e MinimizeBox do Form

Você deve ter percebido que os botões maximizar e minimizar do Form não estão habilitados, porém, desta vez, o botão fechar está habilitado. As propriedades MaximizeBox e MinimizeBox são responsáveis por habilitar ou desabilitar os botões Maximizar e Minimizar do nosso Form. Como configuramos para false, desativamos.

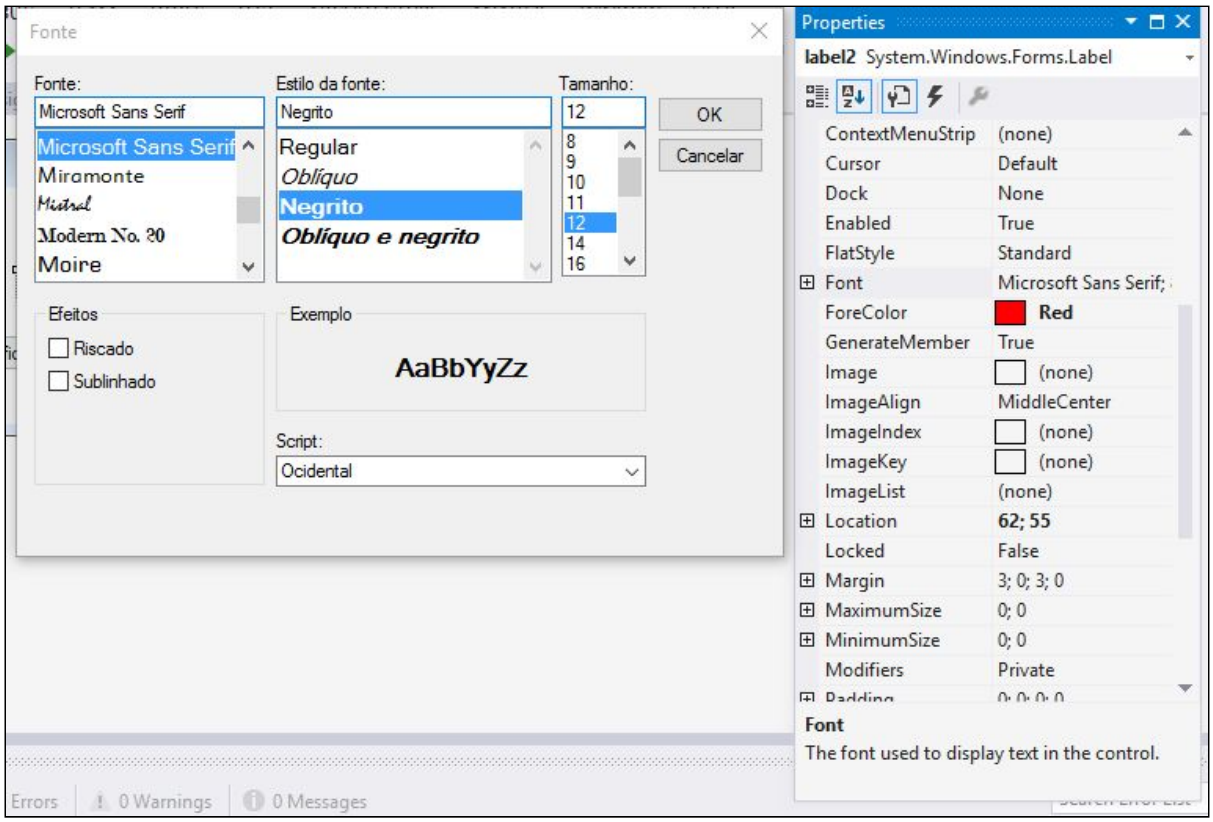


Propriedades Font e ForeColor do controle Label

A propriedade Font define a fonte, o estilo de fonte e tamanho da fonte para os controles. A propriedade ForeColor define a cor da fonte a ser utilizada.



Em Font da para alterar:



Codificando o botão Fechar:

Application.Exit(); ou Close();

Codificando o Botão Limpar:

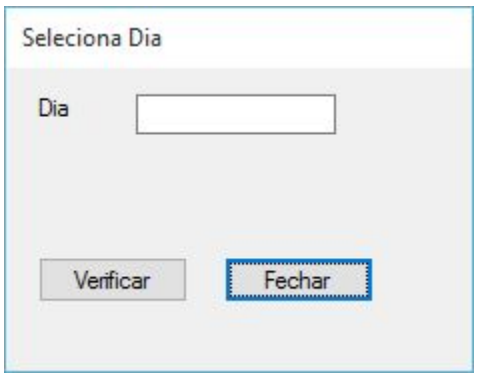
```
1. Limpar.ResetText();//O método ResetText() do controle Label redefine a propriedade Text
   para o valor padrão.
2. //ou
3. Limpar.Clear();//Limpa
//ou
4. Limpar.Text = String.Empty;//Esvazia a String
//ou
5. Limpar.Text = (" ");//Preenche com nada dentro
//Todas as funções acima servem para limpar a textbox.
6. txtVariavel.Focus(); // Coloca o cursor do mouse de volta na textbox escolhida.
```

Evento Load do Form

Quando o programa for executado não queremos que o texto Label2 do controle label2 fique aparente para o usuário.

Duplo clique no form e limpar a label de resultado, isso é o evento load.
Exemplo

```
1. lblDiaSemana.ResetText();
```



O evento Load ocorre antes que o Form seja exibido pela primeira vez.

Controle GroupBox
O controle GroupBox é um controle do tipo container utilizado para organizar, agrupar um conjunto de controles tais como RadioButton e CheckBox, Button, etc, contendo uma

legenda e uma borda fina que envolve o conjunto dos controles. Um Form pode conter vários GroupBox.

Controle RadioButton

O controle RadioButton é utilizado quando desejamos que o usuário escolha uma entre duas ou mais opções mutuamente exclusivas, como por exemplo, amarelo ou verde. Normalmente existem vários botões RadioButtons agrupados.

Para agrupar os controles RadioButton que fazem parte do mesmo conjunto de opções colocamos esses controles dentro de um Groupbox. Um Form pode ter vários conjuntos de RadioButton separados em GroupBox.

Propriedade Checked do controle RadioButton:

Por meio da propriedade Checked do controle RadioButton podemos verificar se um determinado controle está selecionado ou não.

Exemplo no código:

```
rdbVariavel.Checked=true;//rdb = radiobutton
```

Controle CheckBox

O controle CheckBox é utilizado quando desejamos que o usuário escolha uma ou mais opções. Por exemplo, dada as opções de esportes futebol, natação, vôlei, basquete, atletismo, o usuário poderá escolher um ou mais esporte de sua preferência.

Para agrupar os controles CheckBox que fazem parte do mesmo conjunto de opções colocamos esses controles dentre de um GroupBox e se necessário podemos ter vários conjuntos de CheckBox separados em GroupBox.

Controle ComboBox

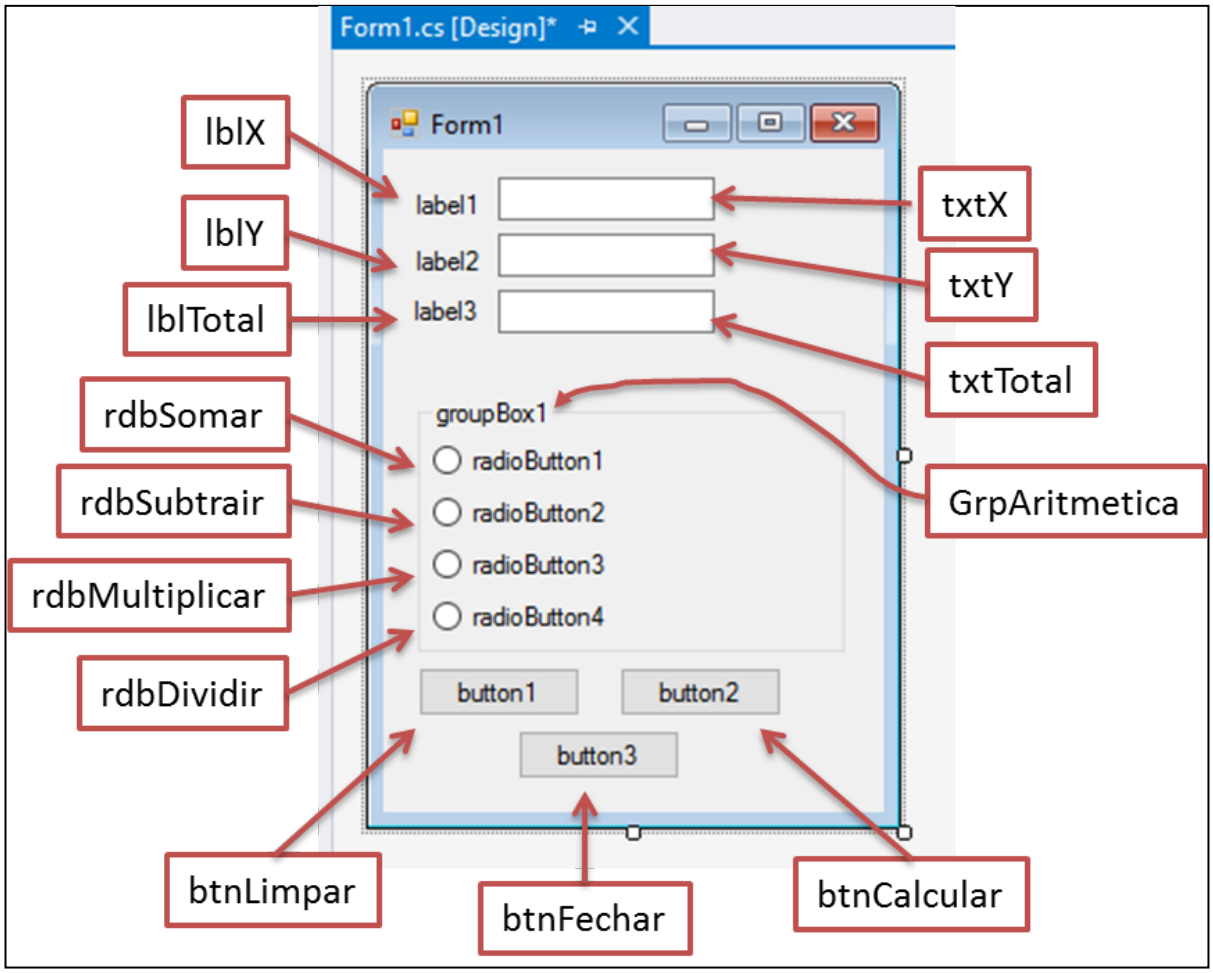
O controle ComboBox (caixa de combinação) combina recursos de TextBox com uma lista suspensa que contém uma lista a partir da qual valores podem ser escolhidos.

A os itens da lista podem ser adicionados na propriedade Items, em linhas de programação utilizando o método Add( ), ou utilizando o editor String Collection Editor. Veremos outros detalhes mais adiante.

Controle PictureBox

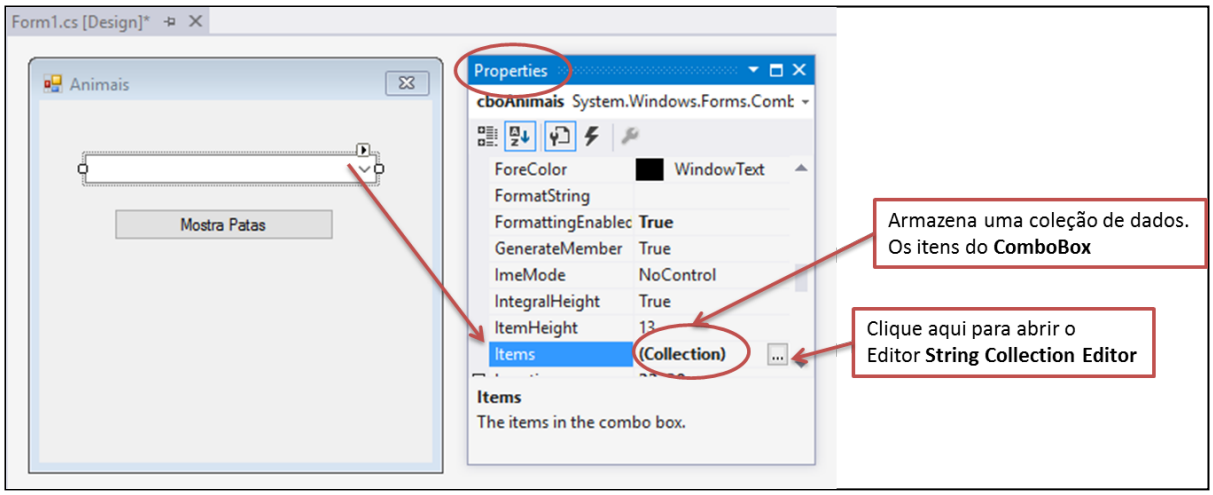
O controle PictureBox (caixa de figura) exibe uma imagem. A imagem pode estar no formato de bitmap (.bmp), .gif, .jpg, ícone ou metaarquivo. A propriedade Image configura a imagem a ser usada; a propriedade SizeMode configura o modo como a imagem é apresentada podendo assumir os valores Normal, StretchImage, AutoSize ou CenterImage.

Exemplo usando Tudo Isso:



Propriedade Items do controle ComboBox

A propriedade Items armazena a coleção de itens no controle ComboBox. Você pode adicionar os itens que serão apresentados na lista suspensa do ComboBox utilizando a propriedade Items e clicando no botão ao lado para utilizar o String Collection Editor, ou por meio do programa Visual Studio utilizando o método Add( ). Nós vamos adicionar os itens por meio do programa, ou seja, codificando os itens do ComboBox.



Método Add( ) do ComboBox

Quando o usuário executar o nosso programa Mostra Patas, o ComboBox já deve estar preenchido com os itens. Como eu mencionei, vamos adicionar os itens em linha de comando. Você deve fazer essa codificação no evento Load do Form, portanto, dê duplo clique no Form e digite o código a seguir.

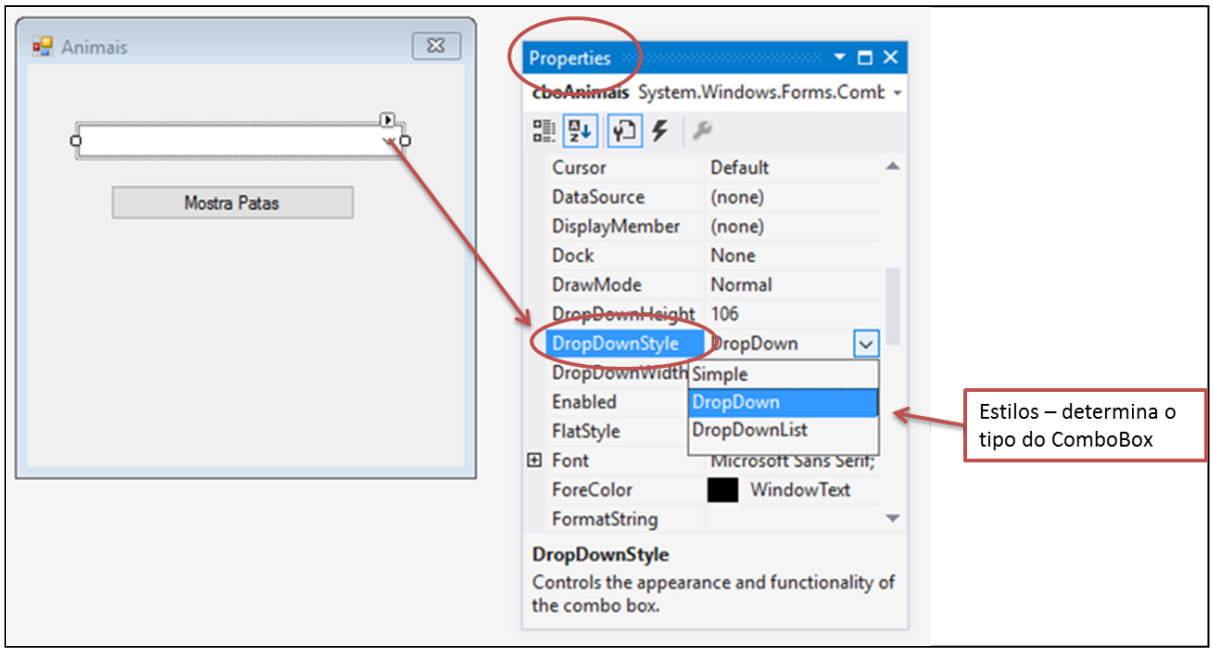
```
private void Form1_Load(object sender, EventArgs e)
{
    //limpa os itens adicionados no ComboBox
    cboAnimais.Items.Clear();

    //adiciona os Itens no ComboBox
    //por meio do méto Add( )
    cboAnimais.Items.Add("Cachorro");
    cboAnimais.Items.Add("Cavalo");
    cboAnimais.Items.Add("Gato");
    cboAnimais.Items.Add("Centopeia");
    cboAnimais.Items.Add("Cobra");
    cboAnimais.Items.Add("Pássaro");
}
```

O método Add( ) recebe uma String como item a ser adicionado no ComboBox.

Propriedade DropDownStyle do ComboBox

Essa propriedade determina o tipo do ComboBox. Os valores para essa propriedade são: Simple: o texto pode ser editado e a lista é visível; DropDown (default) o texto pode ser editado mas o usuário tem que clicar no botão do ComboBox para ver a lista; e

DropDownList: o texto não pode ser editado e o o usuário tem que clicar no botão do ComboBox para ver a lista.



Botão Mostra Patas:

```
private void btnPatas_Click(object sender, EventArgs e)
{
    //declara variável
    String strAnimal;

    //variável recebe o texto do item do ComboBox Selecionado
    strAnimal = cboAnimais.Text;

    //instrução de seleção
    switch (strAnimal)
    {
        //agrupando vários casos na mesma seção
        case "Cavalo":
        case "Gato":
        case "Cachorro":
            MessageBox.Show("Este aminal tem 4 patas", "Animais",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            break;
```

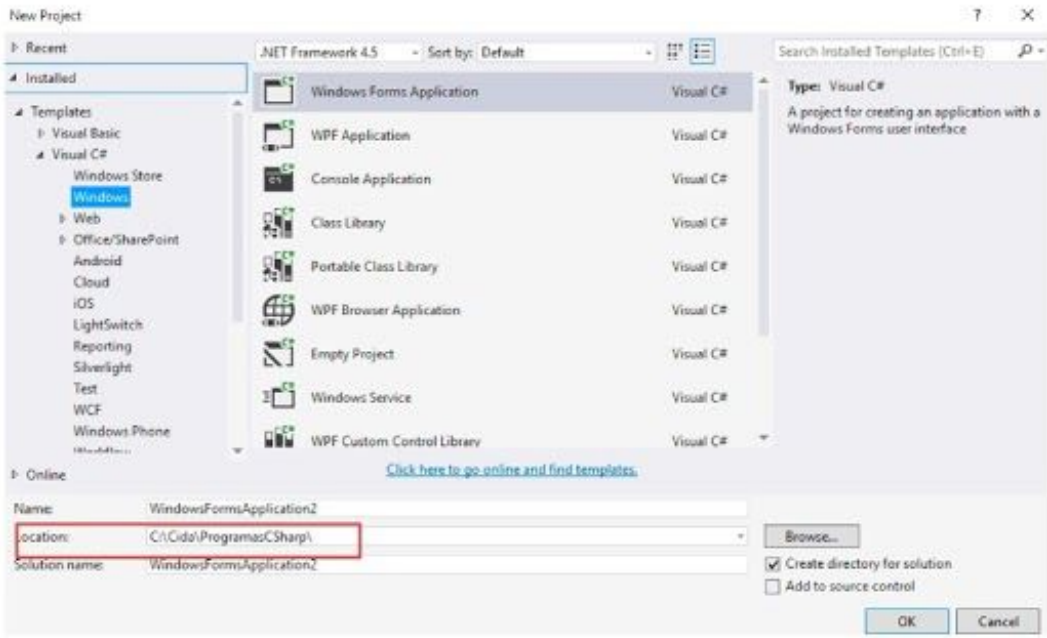
Upgrade Projeto Mostra Patas

Quando o usuário selecionar um animal no cboAnimais será apresentada a figura correspondente a esse animal. Para fazer essa evolução no programa, você precisa ter uma imagem de cada animal. Pesquise na Internet e faça o download dessas imagens.

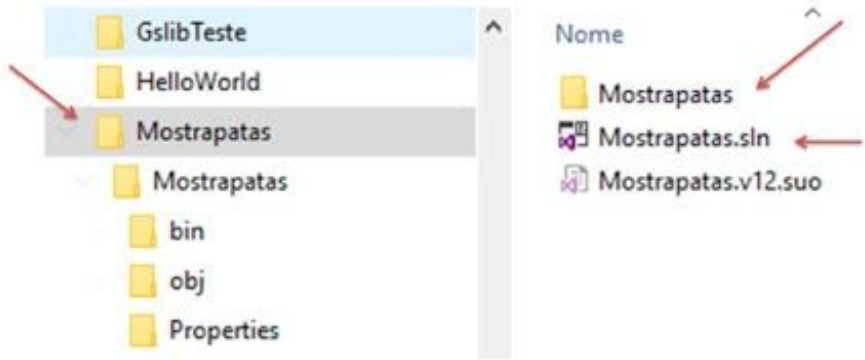
Você pode estar se perguntando, onde salvar as imagens? As imagens deverão ser salvas na pasta Debug. Quando você cria um novo projeto no Visual Studio, você define o local no qual o seu projeto será salvo (location). Abrir a pasta principal que tem o nome do projeto Mostrapatas, abrir a segunda pasta (Mostrapatas), abrir a pasta bin, abrir a pasta Debug. Note que o Visual Studio criou várias pastas e arquivos automaticamente. Nessa mesma pasta Debug, encontra-se o arquivo executável da aplicação, mas para o Visual Studio gerar esses arquivo você tem que ter construído a solução (menu Build – Build Solution ou Ctrl+Shift-B), mas toda vez que você executa o programa (play ou F5) o Visual Studio compila o seu programa e gera/atualiza o seu executável.

Veja a sequência de pastas ilustrado a seguir.

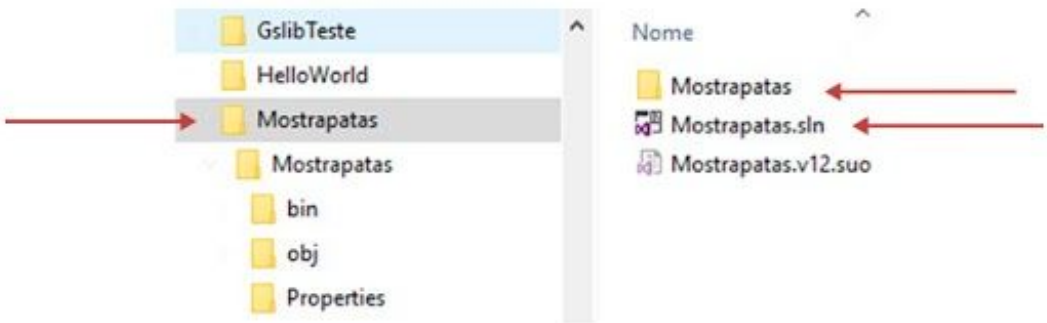
Localizando a pasta Debug para salvar as imagens do projeto Mostra Patas



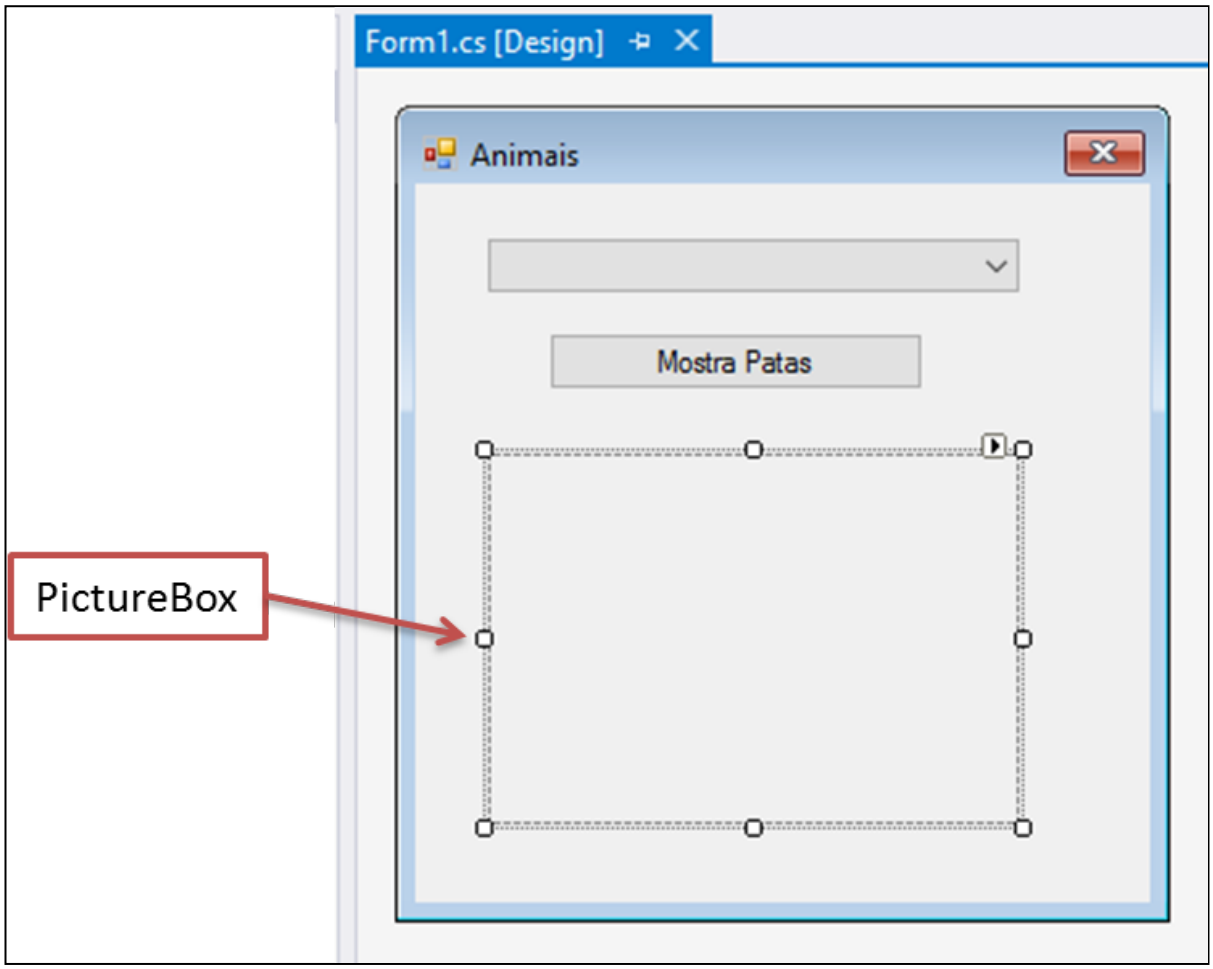
Quando você cria um novo projeto no Visual Studio, você define o local no qual o seu projeto será salvo (location).



No Windows Explorer, localize a pasta onde você salvou o projeto.
Note que o Visual Studio criou várias pastas e arquivos.
Dentro dessa pasta está o arquivo Mostrapatas.sln que é o arquivo do projeto.
Você pode abrir o projeto dando duplo clique nesse arquivo.
Abra a pasta Mostrapatas do lado direito.



Agora, inclua um controle PictureBox no Form. O layout deve estar parecido com a figura a seguir.



Altere a propriedade name do PictureBox para pctAnimais e propriedade SizeMode para StretchImage.

A propriedade SizeMode controla a dimensão e posicionamento da imagem no PictureBox. Os possíveis valores são: Normal, CenterImage, StretchImage, AutoSize e Zoom.

Valor	Descrição
AutoSize	Redimensiona o **PictureBox** em tamanho igual ao tamanho da imagem que ele contém.

CenterImage	A imagem é exibida no centro se o **PictureBox** é maior do que a imagem. Se a imagem for maior que o **PictureBox**, a imagem é colocada no centro da **PictureBox** e as bordas externas são cortadas.
Normal	A imagem é colocada no canto superior esquerdo do **PictureBox**. A imagem será cortada se ela for maior que o **PictureBox**.
StretchImage	Redimensiona a imagem para caber dentro do **PictureBox**.
Zoom	O tamanho da imagem aumenta ou diminui para manter a proporção de tamanho.

Altere a propriedade Sort do ComboBox para True. Dessa forma os itens serão colocados em ordem alfabética.

Codificando o botão Mostra Patas agora com Imagens

Você vai utilizar o método Load( ) para carregar as imagens no PictureBox dependendo do animal selecionado no cboAnimais. Será preciso alterar o código do botão Mostra Patas.

Dê duplo clique no botão Mostra Patas e apague o código existente, ou se preferir, deixe o código anterior comentado. Você também poderá desenvolver outro programa para ficar com as duas versões do Mostra Patas. Digite o código a seguir.

```
1. //declara variável
2. String strAnimal;
3.
4. //variável recebe o texto do item do ComboBox Selecionado
5. strAnimal = cboAnimais.Text;
6.
7. //instrução de seleção
8. switch (strAnimal)
```

```
9.  {
10. case "Cachorro":
11.
12.    //carrega o arquivo que está na pasta Debug
13.    pctAnimais.Load("cachorro.png");
14.    break;
15. case "Cavalo":
16.    pctAnimais.Load("cavalo.png");
17.             MessageBox.Show("Este aminal tem 4 patas", "Animais",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
18.    break;
19. case "Centopeia":
20.    pctAnimais.Load("centopeia.png");
21.           MessageBox.Show("Este aminal tem 100 patas", "Animais",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
22.    break;
23. case "Cobra":
24.    pctAnimais.Load("cobra.png");
25.           MessageBox.Show("Este aminal não tem patas", "Animais",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
26.    break;
27. case "Gato":
28.    pctAnimais.Load("gato.png");
29.             MessageBox.Show("Este aminal tem 4 patas", "Animais",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
30.    break;
31. case "Pássaro":
32.    pctAnimais.Load("passaro.png");
33.             MessageBox.Show("Este aminal tem 2 patas", "Animais",
    MessageBoxButtons.OK, MessageBoxIcon.Information);
34.    break;
35. default:
```

```
36.    //limpa o PictureBox
37.    pctAnimais.ResetText();
38.                MessageBox.Show("Selecione um Animal", "Animais",
       MessageBoxButtons.OK, MessageBoxIcon.Error);
39.    break;
40. }
```

Lembre-se que o nome do arquivo e sua extensão que está dentro do método Load tem que corresponder exatamente ao nome do arquivo existente na pasta Debug.