

Aula 01 – Introdução ao VS2010 e C#

Profa. Ana Paula Citro Fugarra Rodrigues

Aula 01 – Introdução ao VS2010 e C#

Profa. Ana Paula Citro Fugarra Rodrigues

Prof. Ronaldo Vaqueli de Paula

Por que você deve aprender C#

O C# e o IDE do Visual Studio facilitam o trabalho de escrever código e de desenvolvê-lo rapidamente. Quando você estiver trabalhando com o C# o IDE será seu melhor amigo e companhia constante.

Aqui vemos o que o IDE automatiza para você

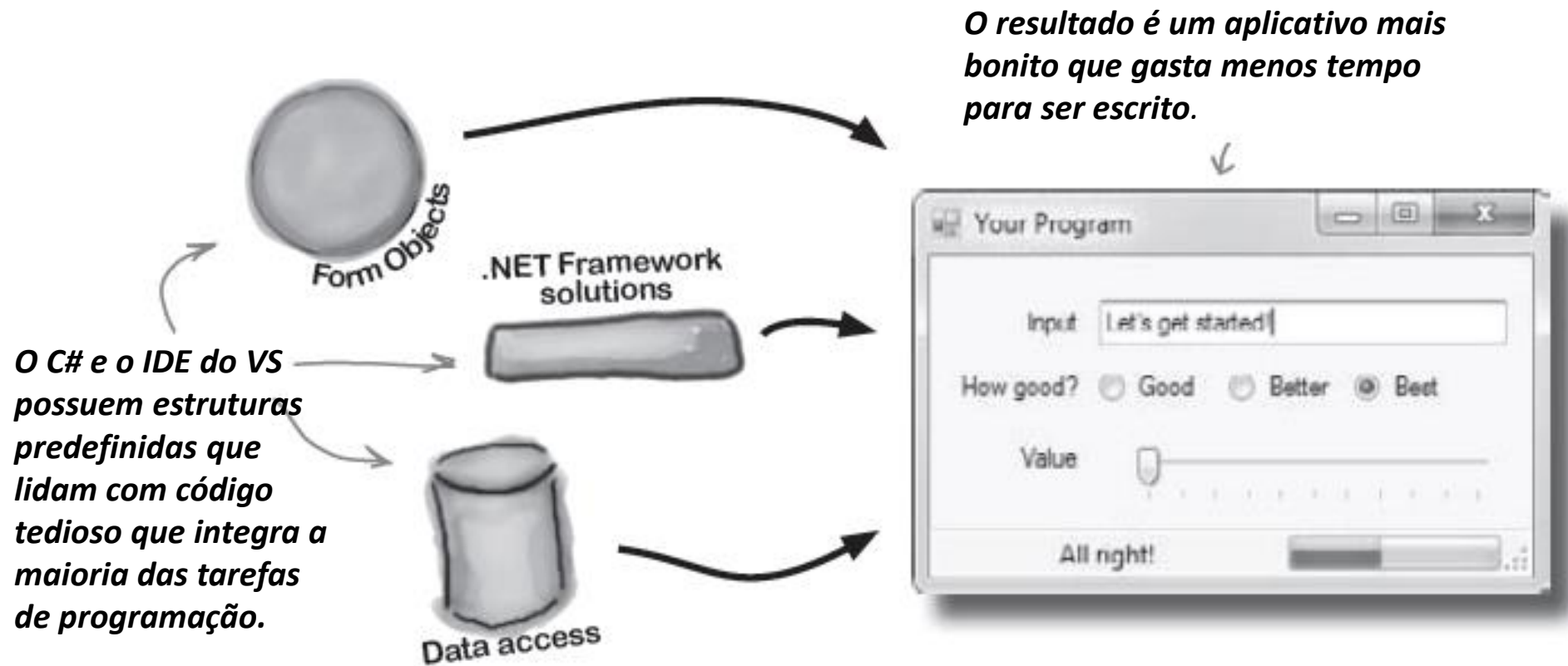
Para escrever um programa ou apenas colocar um botão em um formulário seu programa precisa de um monte de código repetitivo.

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;
namespace A_New_Program
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

```
private void InitializeComponent()
{
    this.button1 = new System.Windows.Forms.Button();
    this.SuspendLayout();
    //
    // button1
    //
    this.button1.Location = new System.Drawing.Point(105, 56);
    this.button1.Name = "button1";
    this.button1.Size = new System.Drawing.Size(75, 23);
    this.button1.TabIndex = 0;
    this.button1.Text = "button1";
    this.button1.UseVisualStyleBackColor = true;
    this.button1.Click += new System.EventHandler(this.button1_Click);
    //
    // Form1
    //
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.None;
    this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 16F);
    this.ClientSize = new System.Drawing.Size(292, 267);
    this.Controls.Add(this.button1);
    this.Name = "Form1";
    this.Text = "Form1";
    this.ResumeLayout(false);
}
```

O que você consegue com o VS e o C#

Com uma linguagem como C#, otimizada para programação em Windows, e com o IDE do VS, você pode focar-se no que o seu programa deve fazer



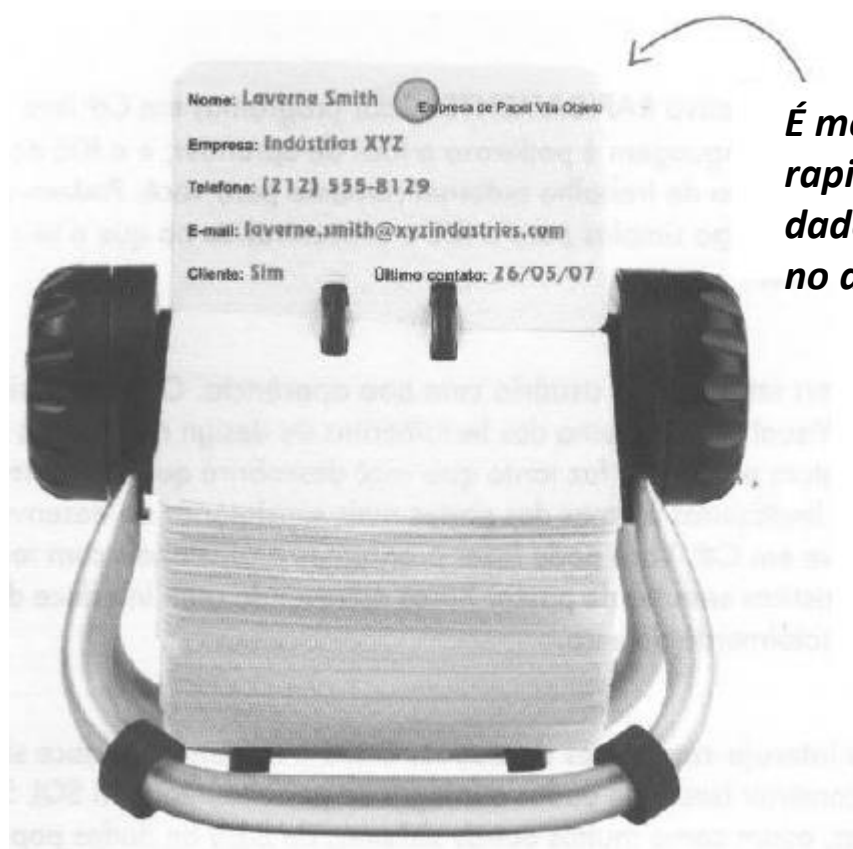
O C# e o IDE do VS facilitam muitas coisas

Quando você usa C# e o VS tem todas estas grandes características ao seu alcance, sem nenhum trabalho extra. Juntos, eles permitem que você:

- 1. Faça um aplicativo RAPIDAMENTE.**
- 2. Faça uma interface de usuário com boa aparência.**
- 3. Crie e interaja com bases de dados.**
- 4. Concentre-se em resolver seus problemas REAIS.**

Ajude o diretor a eliminar os papéis

A Empresa de Papel Vila Objeto contratou um novo diretor. Ele adora fazer caminhadas, café e a natureza ... e ele decidiu ajudar a salvar as florestas; quer ser um executivo “sem papel”, começando pelos seus contatos.

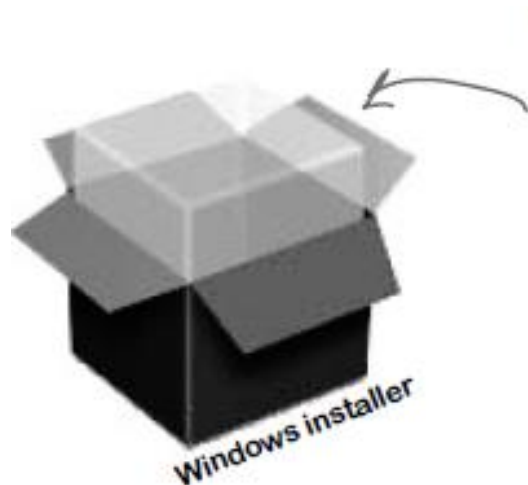


É melhor você encontrar rapidamente uma forma destes dados entrarem no notebook no diretor

Conheça as necessidades dos usuários antes de começar a fazer seu programa

Antes que possamos começar a escrever o aplicativo de agenda – ou qualquer outro programa – precisamos de um minuto para pensar em quem irá usá-lo e o que eles precisam que seja feito.

1. O diretor precisa conseguir executar seu programa de agenda no trabalho e também em seu notebook. Ele precisará de um instalador para ter certeza de que todos os arquivos corretos estejam em cada máquina.

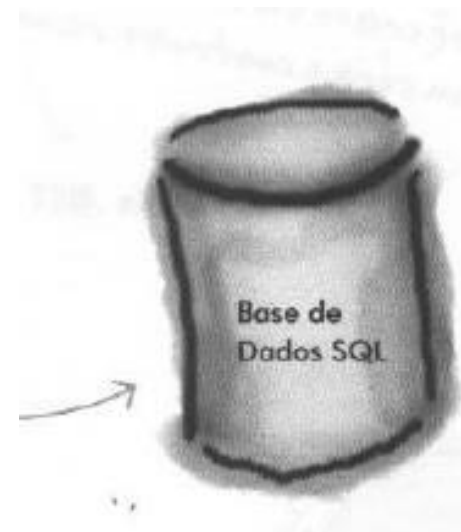


O diretor quer poder executar seu programa no desktop e no notebook, então um instalador é uma necessidade.

2. A equipe de vendas da Empresa de Papel Vila Objeto quer acessar sua agenda também. Eles podem usar seus dados para fazer listas de e-mail para obter mais ordens de compra de papel de seus clientes.

O diretor acha que uma base de dados seria a melhor forma para que todos na empresa pudessem ter acesso aos dados dele. Assim, ele pode manter apenas uma cópia de todos os seus contatos.

Já sabemos que o Visual C# facilita o trabalho com bases de dados. Ter os contatos em uma base de dados permite que o diretor e a equipe de vendas tenham acesso às informações, ainda que não exista uma cópia dos dados.



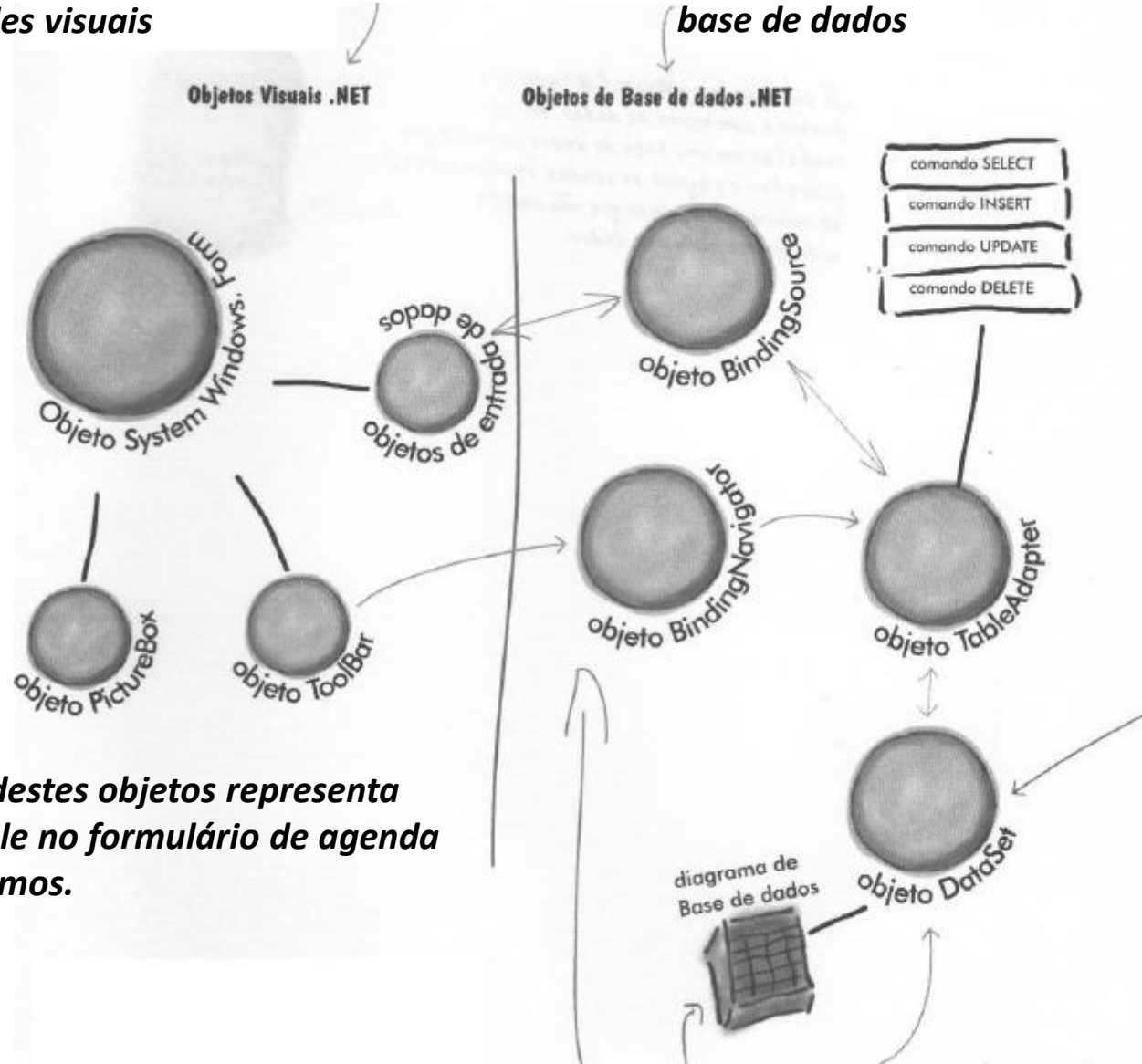
Aqui está o que você vai desenvolver

Você precisará de um aplicativo com uma interface gráfica de usuário, objetos para comunicarem-se com a base de dados, a própria base de dados e um instalador. Parece muito trabalhoso, mas até o final do curso você fará isto tudo.

Aqui está a estrutura do programa que criaremos:

Você criará um formulário Windows com vários controles visuais

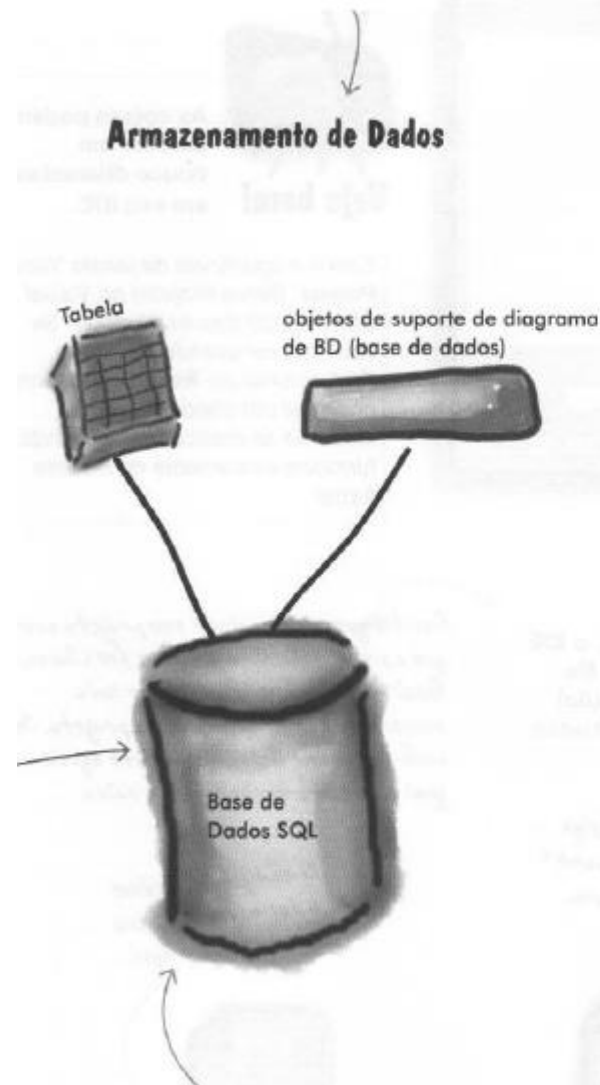
O aplicativo tem uma camada de dados separada que interage com a base de dados



Cada um destes objetos representa um controle no formulário de agenda que criaremos.

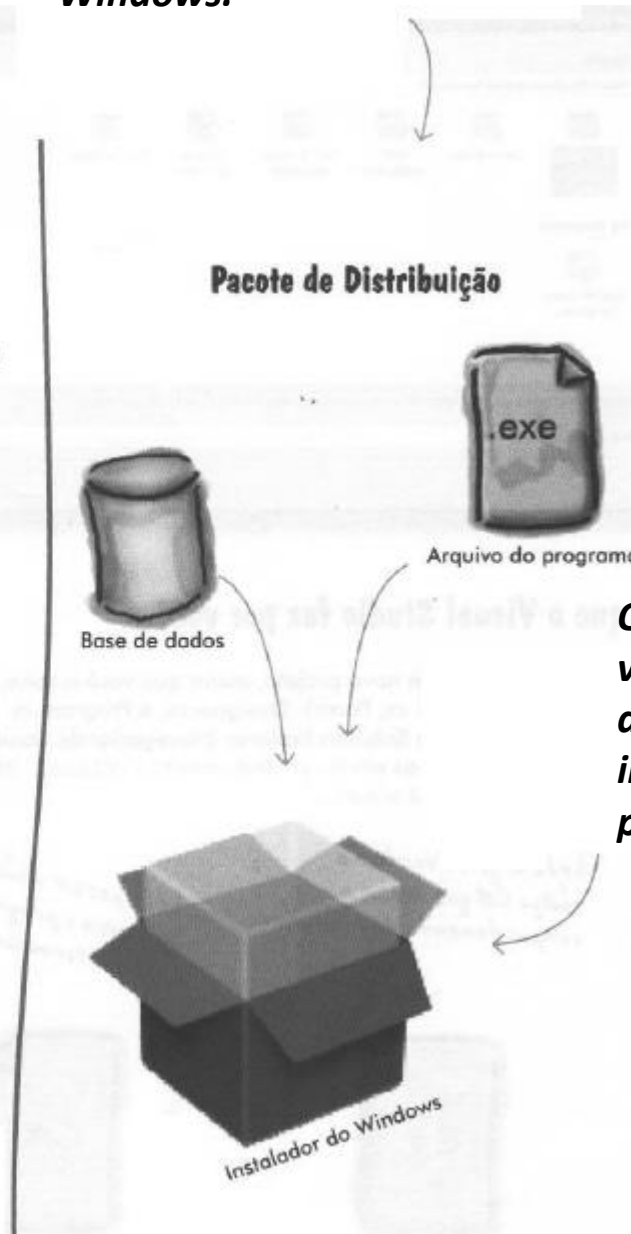
Vamos precisar de objetos para comunicarem-se com nossas tabelas.

Os dados são armazenados em uma tabela na base de dados SQL.



Aqui está a base de dados em si, que o VS nos ajudará a criar e manter.

Uma vez que o programa tenha sido feito, ele será incluído num pacote do instalador do Windows.

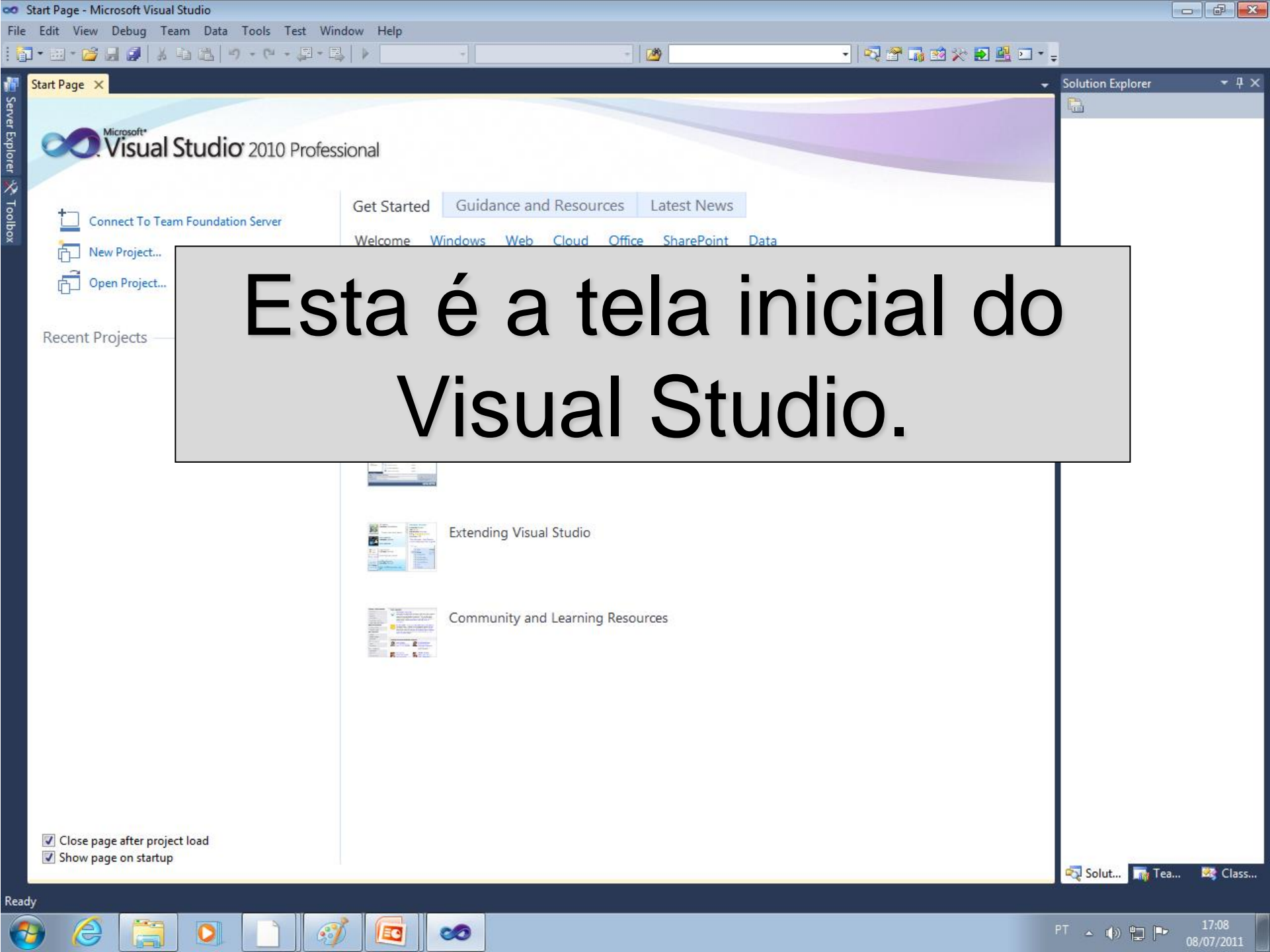


O departamento de vendas precisará apenas apontar e clicar para instalar e, então, usar seu programa.

Inicializando o Visual Studio 2010

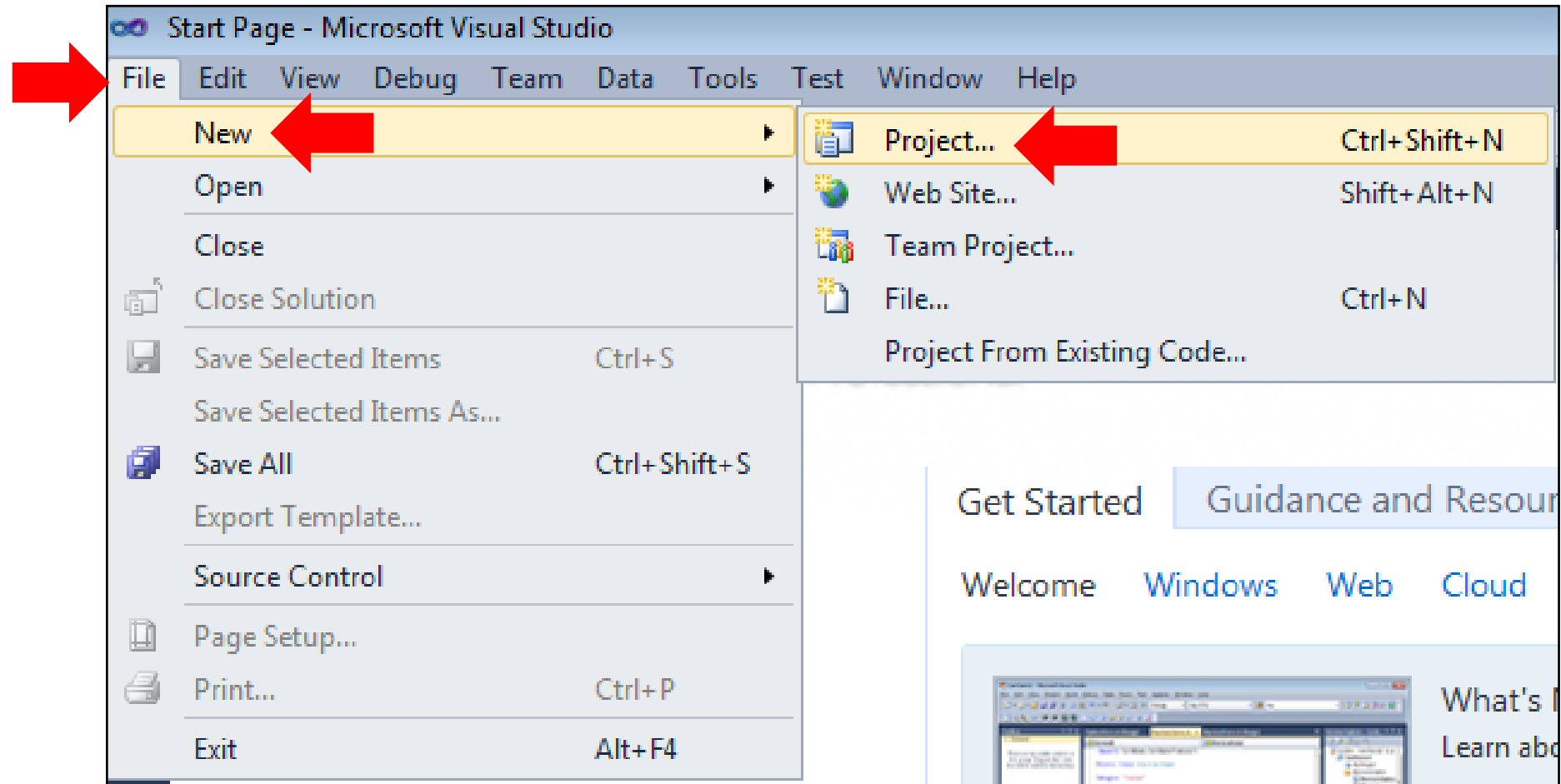
Para inicializar o Ambiente de desenvolvimento Visual Studio 2010 deve-se localizar o ícone abaixo na Área de Trabalho e clicar 2x



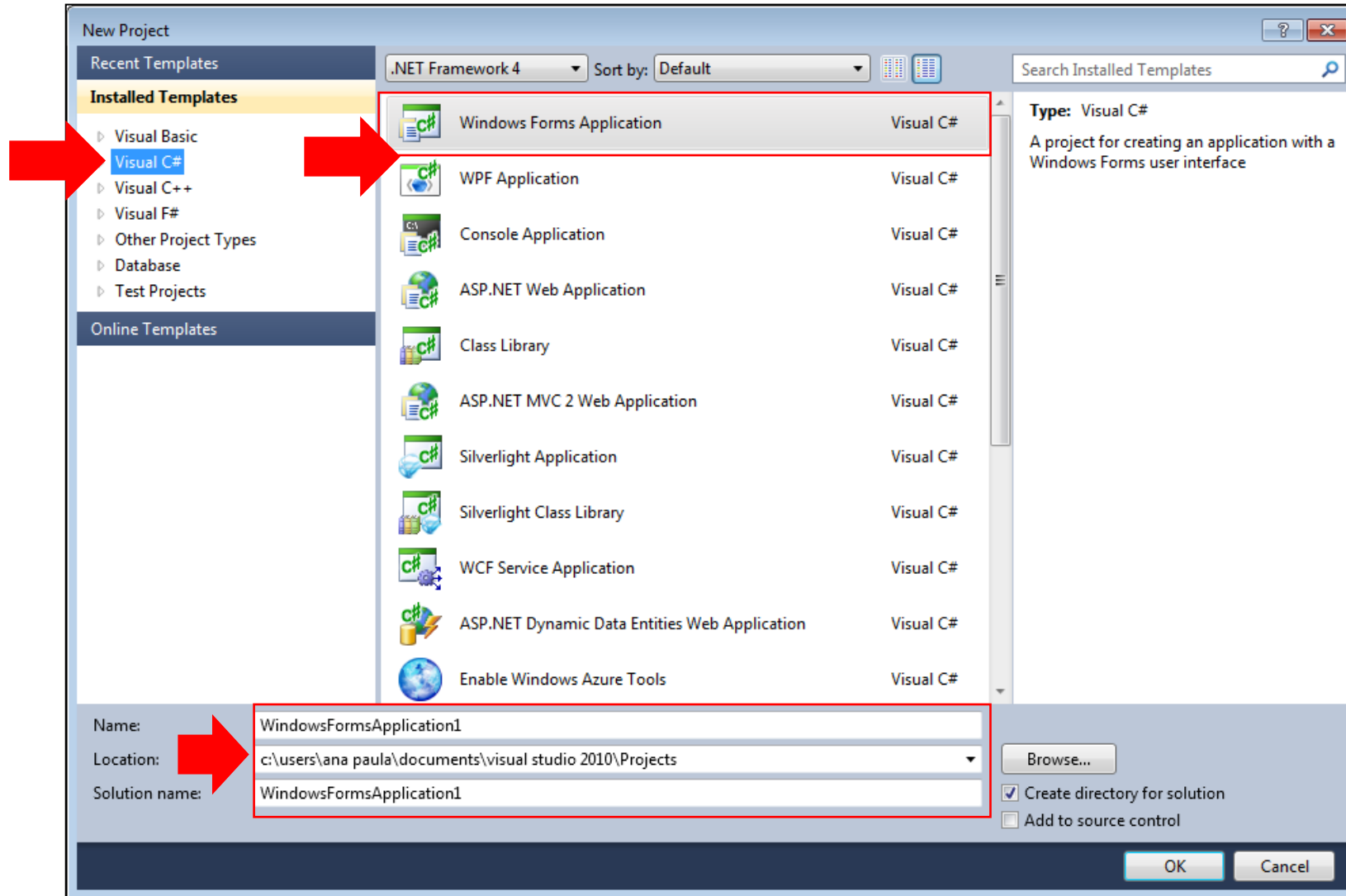


Esta é a tela inicial do
Visual Studio.

Criando um projeto



Criando um projeto



Criando um projeto

Name:	WindowsFormsApplication1	
Location:	c:\users\ana paula\documents\visual studio 2010\Projects	<input type="button" value="Browse..."/>
Solution name:	WindowsFormsApplication1	<input checked="" type="checkbox"/> Create directory for solution <input type="checkbox"/> Add to source control

Name:	Exemplo1	<input type="button" value="Browse..."/>
Location:	E:\Ana Paula\Ano 2011\CDT\Programas\	<input checked="" type="checkbox"/> Create directory for solution <input type="checkbox"/> Add to source control
Solution name:	Exemplo1	<input type="button" value="OK"/> <input type="button" value="Cancel"/>

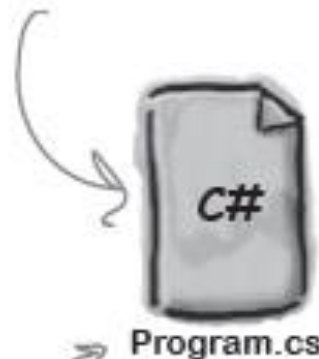
O que o Visual Studio faz por você...

Quando você inicia um novo projeto, assim que você o salva, o IDE cria os arquivos Form1.cs, Form1.Designer.cs, e Program.cs. Ele acrescenta-os à janela Solution Explorer (Navegador de Solução) e, por padrão, coloca-os em **Meus Documentos\Visual Studio 2010\Projects\Exemplo**.

Este arquivo contém o código C# que define o comportamento do formulário.



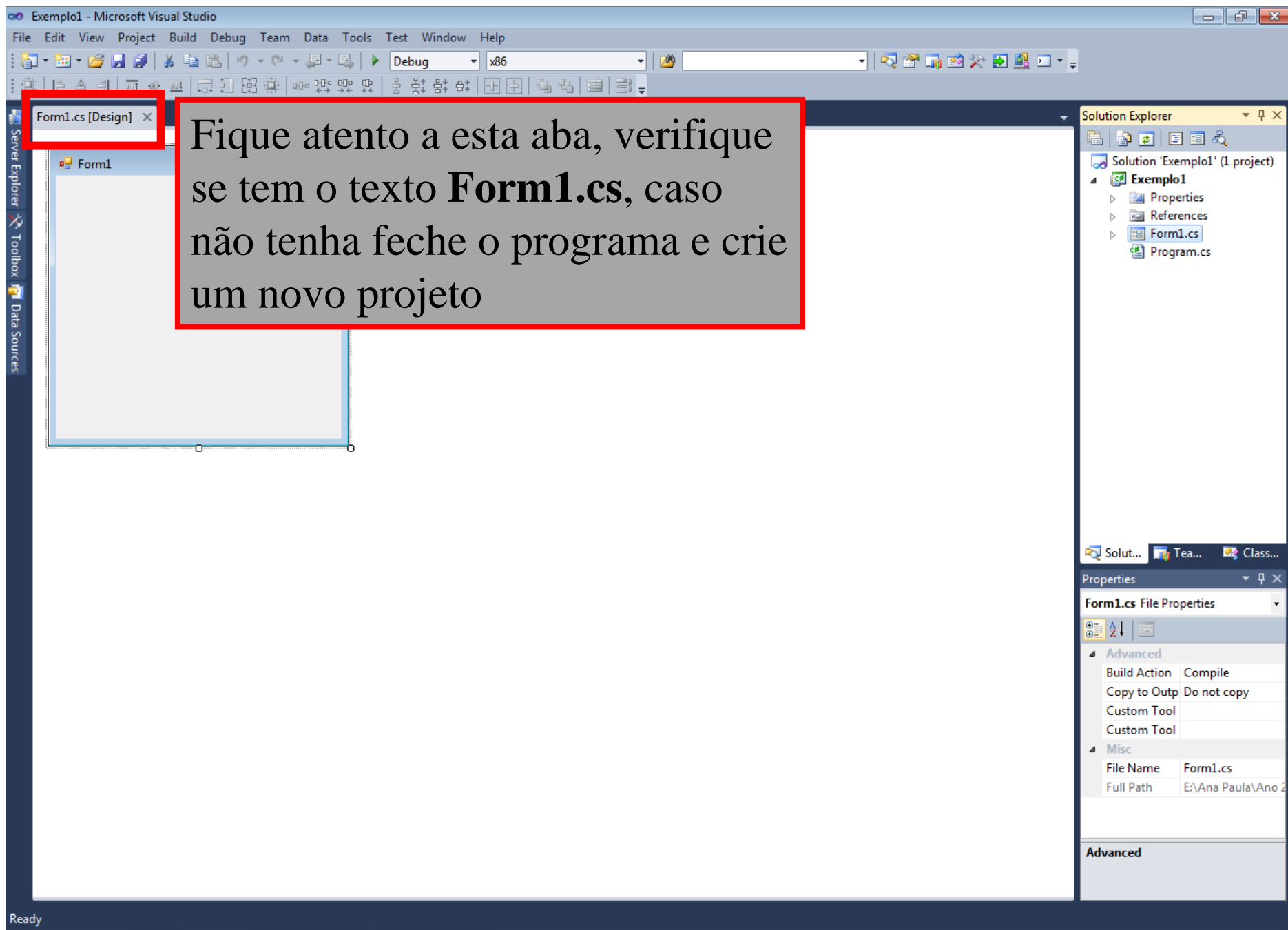
Este possui o código que inicia o programa e exibe o formulário.

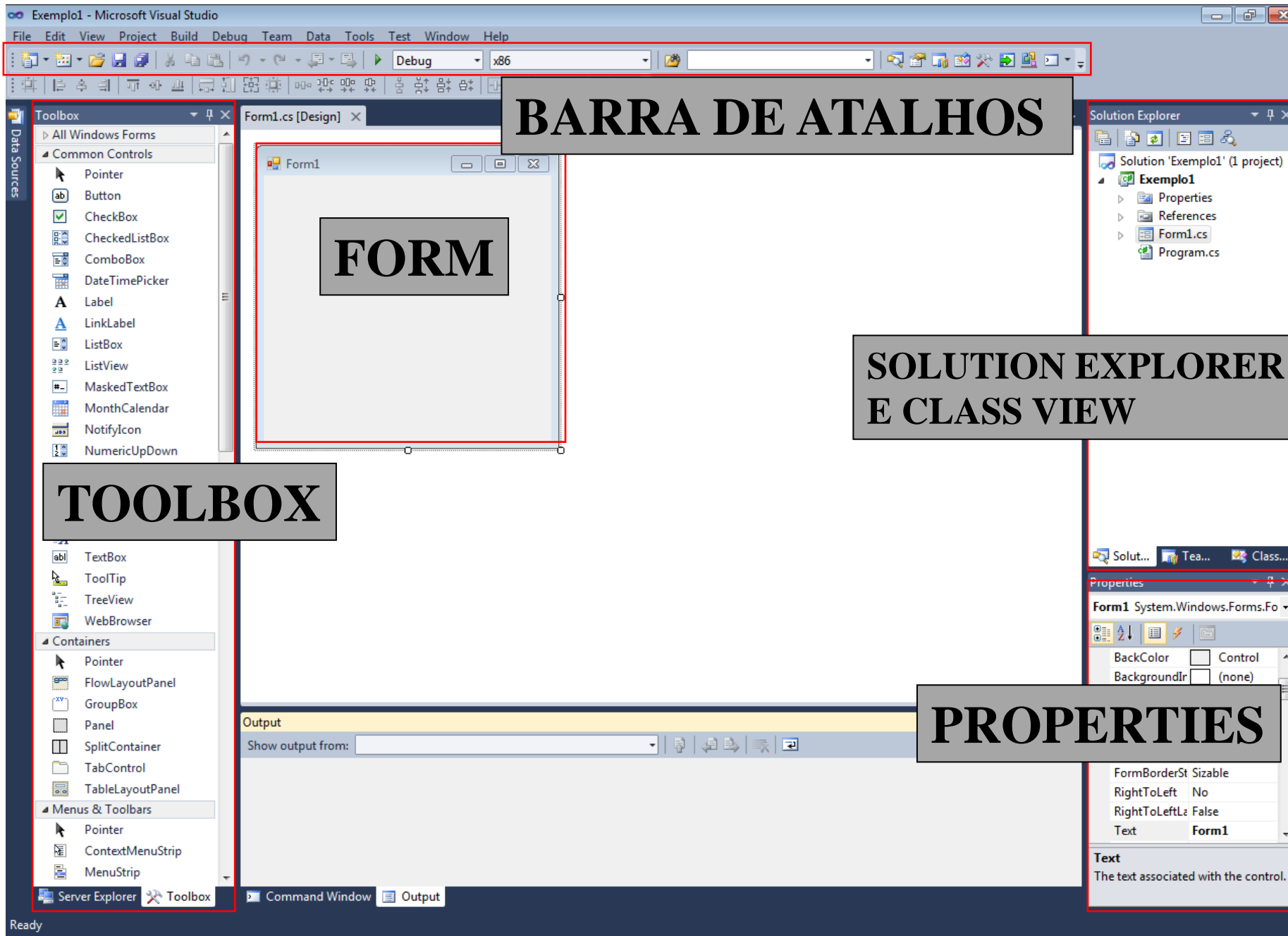


O código que define o formulário e seus objetos está aqui.



O Visual Studio cria estes três arquivos automaticamente.





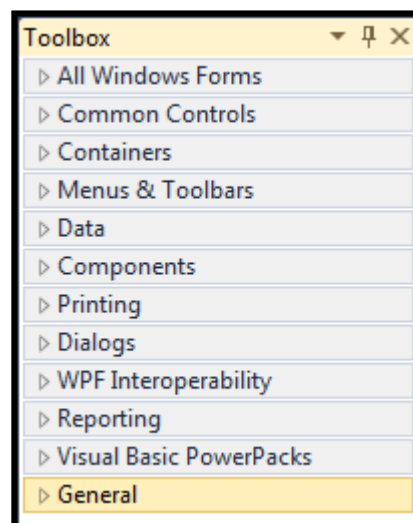
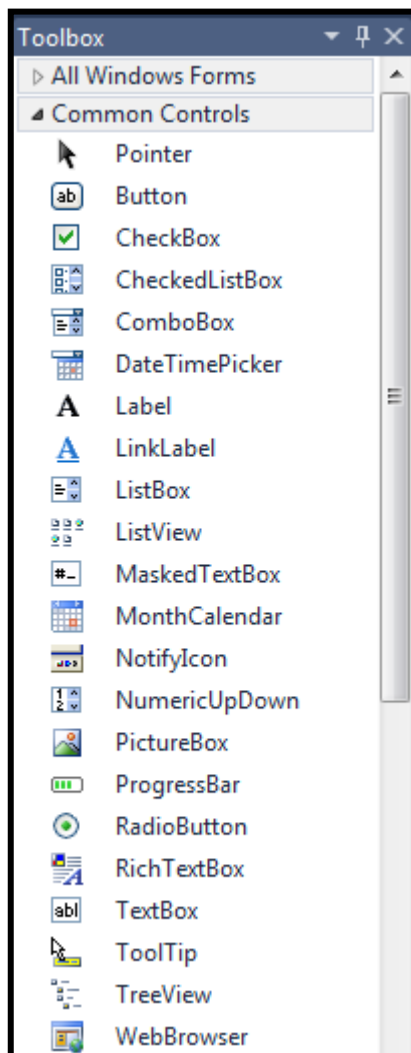
Conhecendo as janelas do VS

TOOLBOX

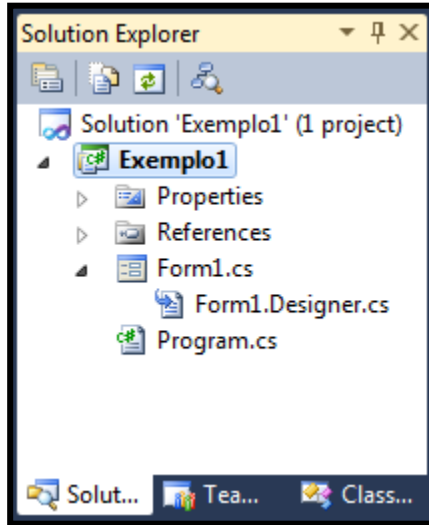
Esta caixa de diálogo é muito importante pois é nela que iremos escolher e pegar os objetos que usaremos no nosso programa.

Ela dividida por grupos de objetos que são separados por tipo.

Neste primeiro momento usaremos somente os objetos do **COMMON CONTROLS**.



Conhecendo as janelas do VS

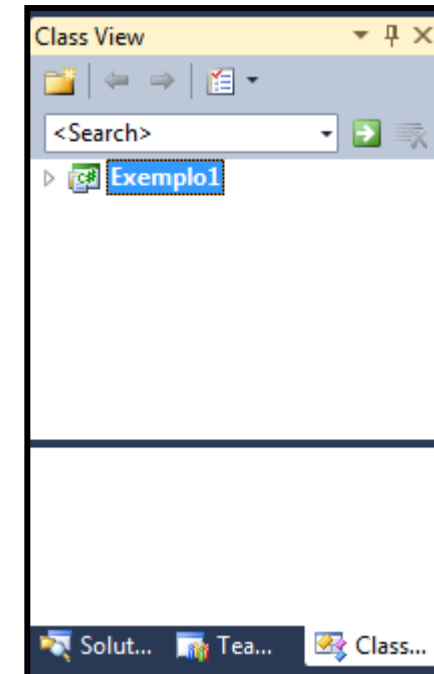


SOLUTION EXPLORER

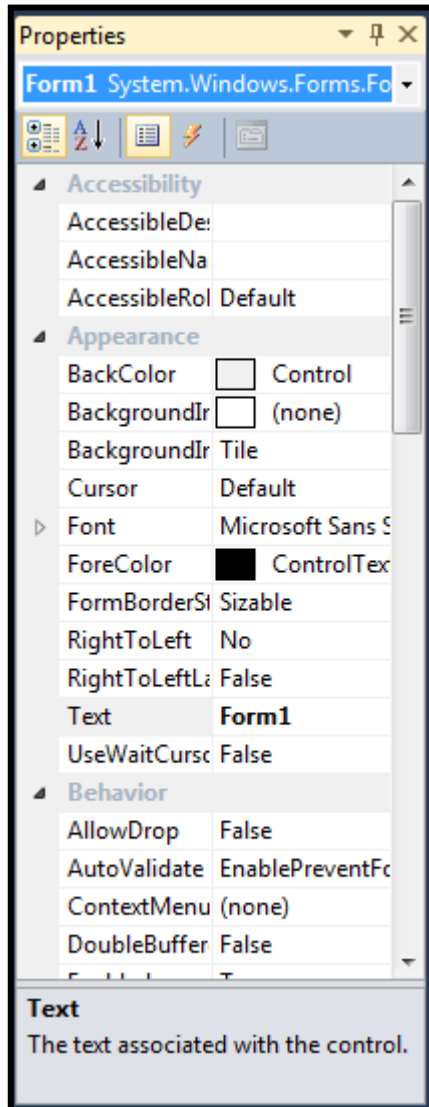
Esta caixa de diálogo é usada para gerenciar os arquivos que foram criados no nosso projeto.

CLASS VIEW

Esta caixa de diálogo é usada para gerenciar as classes criadas no nosso projeto.



Conhecendo as janelas do VS



PROPERTIES

Esta caixa de diálogo é outra muito importante pois é nela que configuraremos os objetos adicionados no nosso programa.

Ela é dividida em 2 partes: Properties e Events, para selecioná-los basta clicar em:



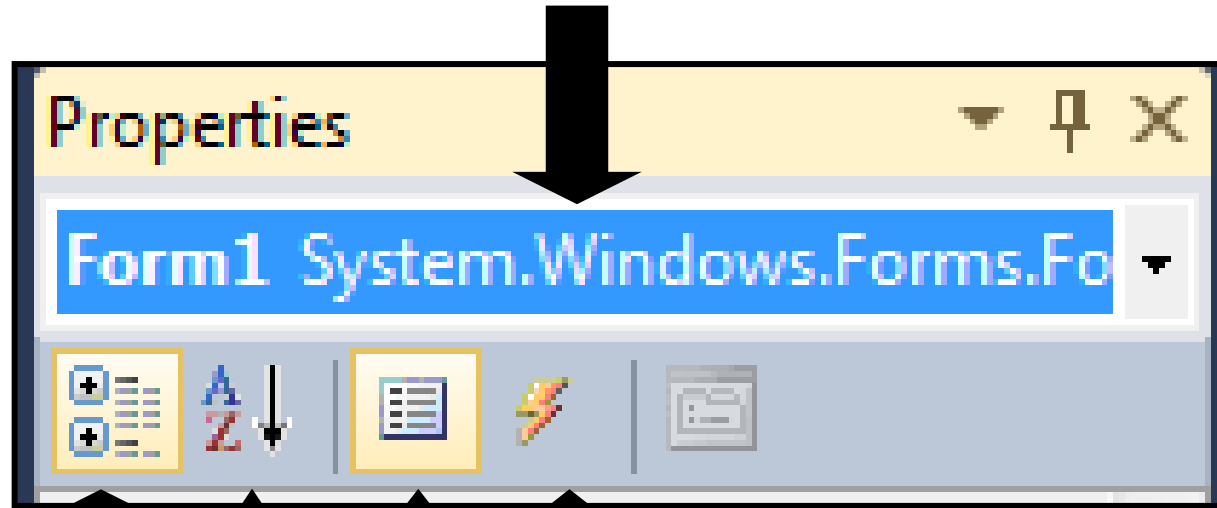
Para exibir as propriedades do objeto selecionado



Para exibir os eventos do objeto selecionado

Conhecendo as janelas do VS

Este campo informa o objeto selecionado



Exibir os itens
separados por tipo

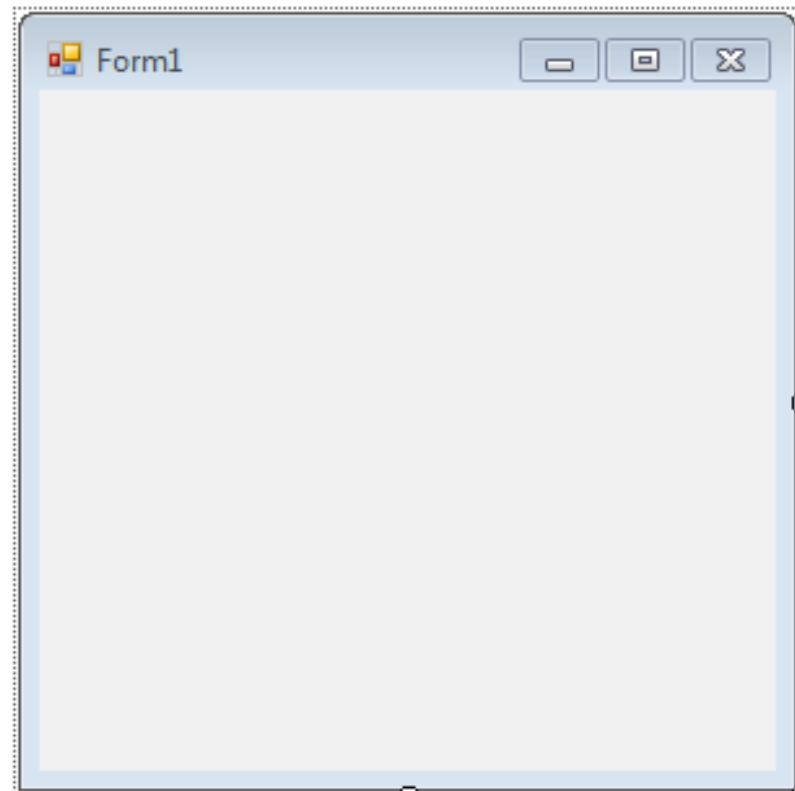
Exibir os itens em
ordem alfabética

Propriedades

Eventos

Conhecendo as janelas do VS

Esta é a tela do nosso programa, conhecida como **FORM** ou **FORMULÁRIO**



Conhecendo as janelas do VS



New Project

Add New Item

Open File

Save File



Start Debbing (F5)



Solution Explorer

Properties Window

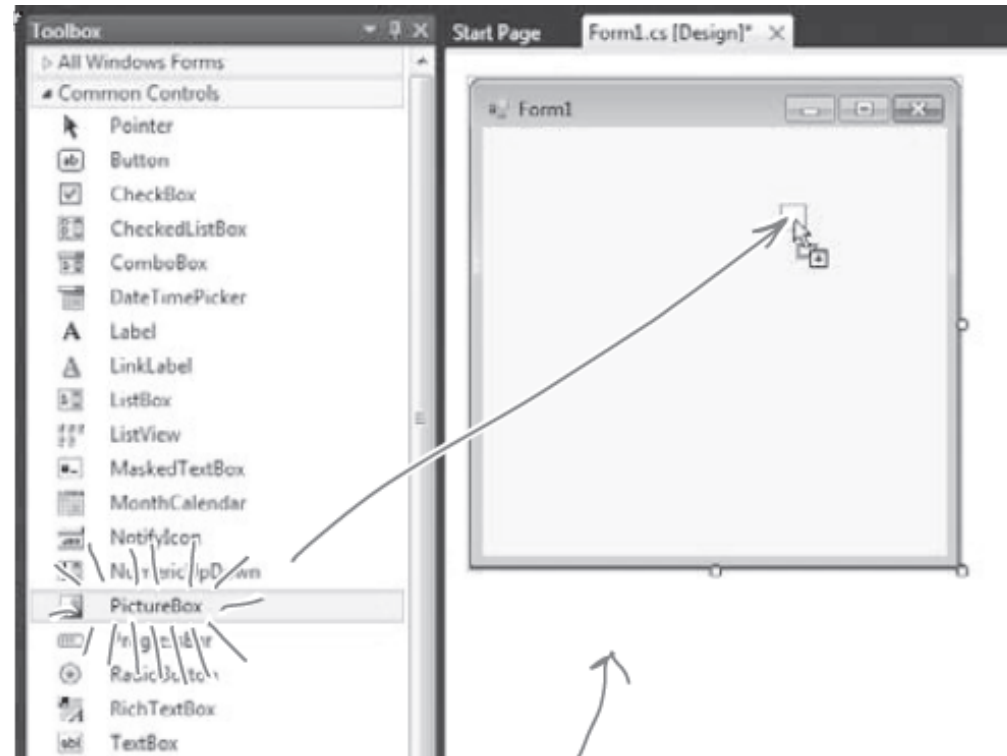
ToolBox Window

Desenvolvendo a interface com o usuário

Adicionar controles e arrumar a interface de usuário é tão fácil quanto arrastar e soltar no IDE do Visual Studio. Vamos acrescentar um logo ao formulário:

- 1. Utilize o controle PictureBox para acrescentar uma figura.**
Clique no controle PictureBox na Caixa de Ferramentas e arraste-o para o seu formulário. Nos bastidores, o IDE adicionou código em `Form1.Designer.cs` para um novo controle de imagens.

Desenvolvendo a interface com o usuário

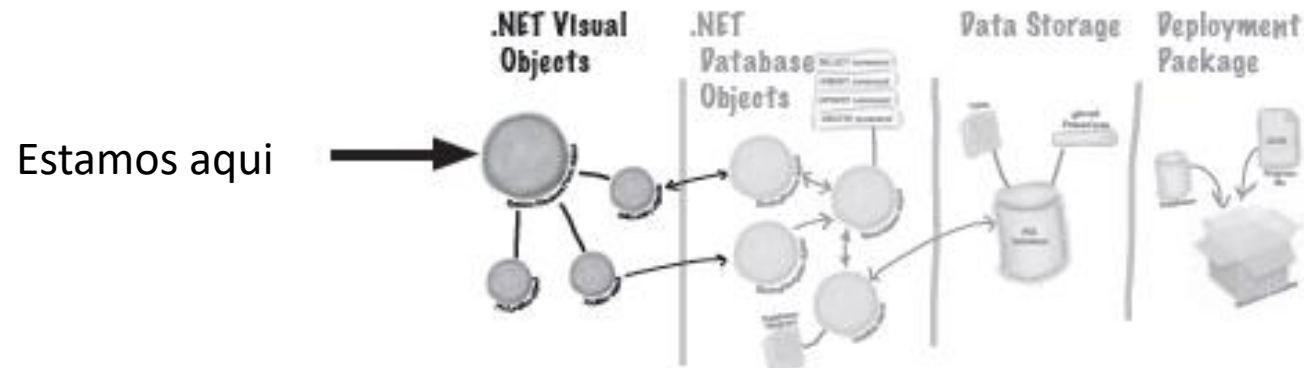


Toda vez que você alterar uma propriedade de controle no formulário, o código em `Form1.Designer.cs` também será mudado pelo IDE



Form1.Designer.cs

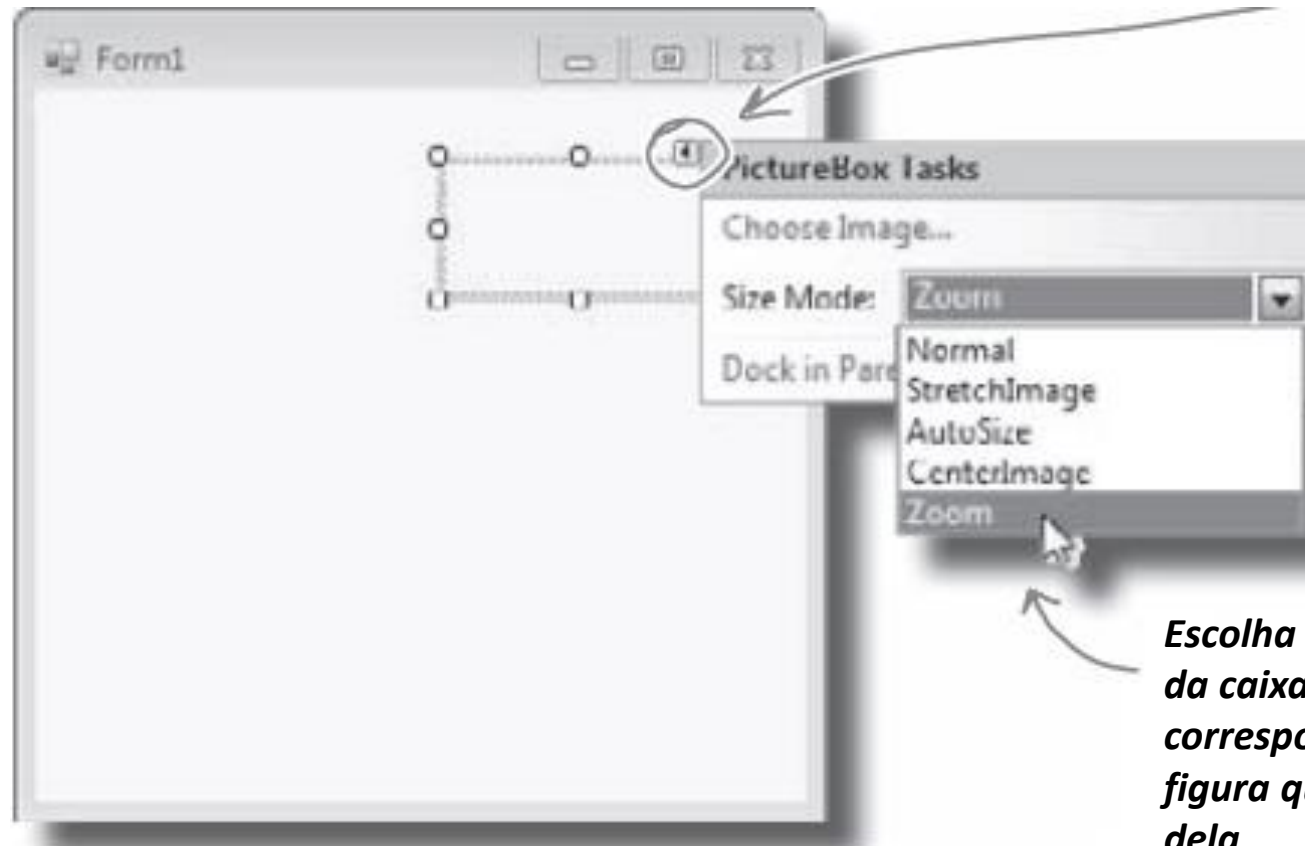
Desenvolvendo a interface com o usuário



2. Coloque a PictureBox em modo Zoom.

Todos os controles em seu formulário possuem propriedades ajustáveis. Clique na flechinha preta para acessá-las. Altere a propriedade Size da PictureBox para "Zoom" para ver como isto funciona:

Desenvolvendo a interface com o usuário



Clique nessa flechinha preta para acessar uma propriedade de um controle

Escolha Zoom para que a borda da caixa de imagem mude para corresponder ao tamanho da figura que você colocou dentro dela

Desenvolvendo a interface com o usuário

3. Adicionar o Logo da Empresa Papel Vila Objeto

Salve o logo no seu disco rígido. Então clique na seta de propriedades da PictureBox e selecione Choose Image. Click em Import ..., encontre seu logo e está tudo pronto:



*Aqui está o logo da OPC –
Empresa de Papel Vila Objeto. A
PictureBox usa o zoom para ficar
do tamanho certo.*

Visual Studio, nos bastidores

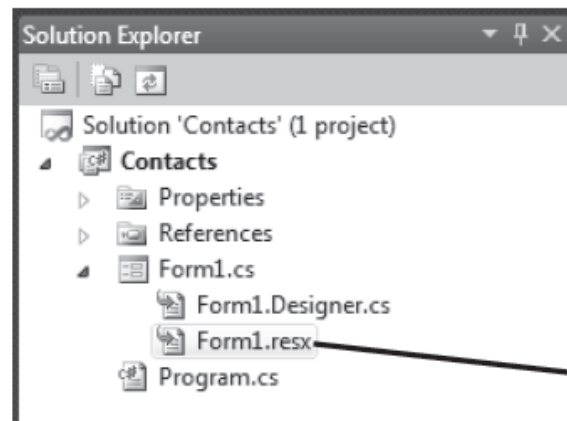
Toda vez que fazemos algo no IDE do Visual Studio, ele está **escrevendo código automaticamente**. Quando criarmos o logo e mandamos o Visual Studio usar a imagem selecionada, ele criou um recurso e associou-o com seu aplicativo. Um **recurso** é qualquer arquivo gráfico, de áudio, ícone ou outro tipo de arquivo de dados embutido no nosso aplicativo. O arquivo gráfico fica integrado ao programa, para eu, então quando ele for instalado em outro computador, o gráfico seja instalado junto com ele e a PictureBox possa usá-lo.



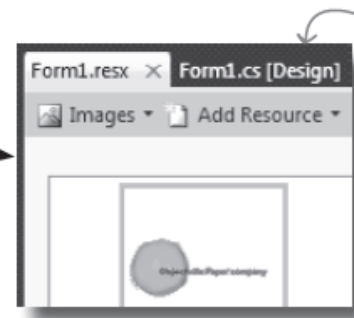
Esta imagem é agora um recurso do aplicativo Exemplo1

Visual Studio, nos bastidores

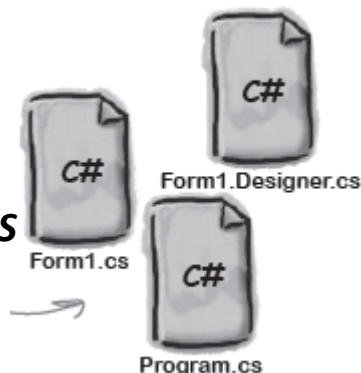
Quando arrastamos o controle PictureBox para o nosso formulário, o IDE automaticamente criou um arquivo de recurso chamado Form1.resx para armazená-lo e mantê-lo em seu projeto. Dê um duplo clique neste arquivo e você verá a imagem importada.



Se clicarmos em Form1.resx no Solution Explorer, veremos a logomarca importada. Este arquivo é conectado à caixa de imagem; e o IDE adicionou código para fazer a conexão.



Estes são os arquivos que o VS criou anteriormente



Complete o código gerado automaticamente

O IDE cria muito código, mas precisamos ter acesso a ele e acrescentar-lhe coisas.

Certifique-se de que o formulário aparece no IDE e clique duas vezes no controle de caixa de imagem. Um código semelhante ao seguinte deve aparecer:

```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
    }
    private void pictureBox1_Click(object sender, EventArgs e)
    {
        MessageBox.Show("Contact List 1.0.\nWritten by: Your Name", "About");
    }
}
```

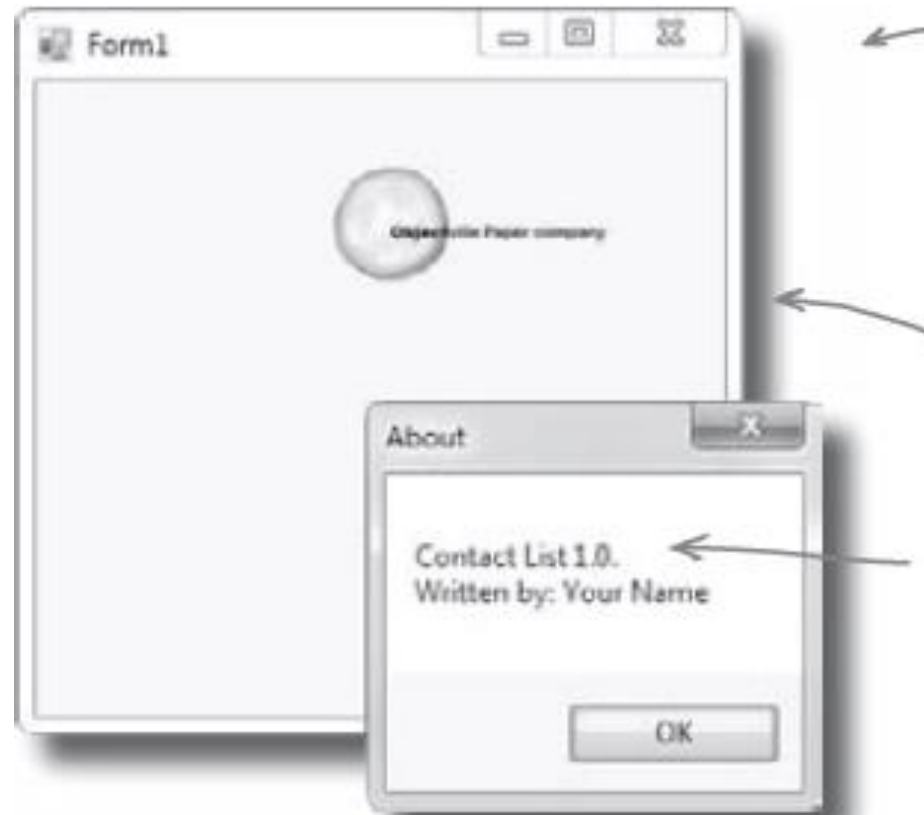
Quando você clicou duas vezes no PictureBox, o IDE criou este método. Ele será executado sempre que um usuário clicar no logo com o aplicativo em execução.

Este nome de método dá uma boa idéia sobre quando ele executa: quando alguém clica no controle PictureBox

Digite esta linha de código. Uma caixa de mensagem aparecerá com o texto que você digitou. A caixa terá o título About (Sobre).

Testando nosso programa

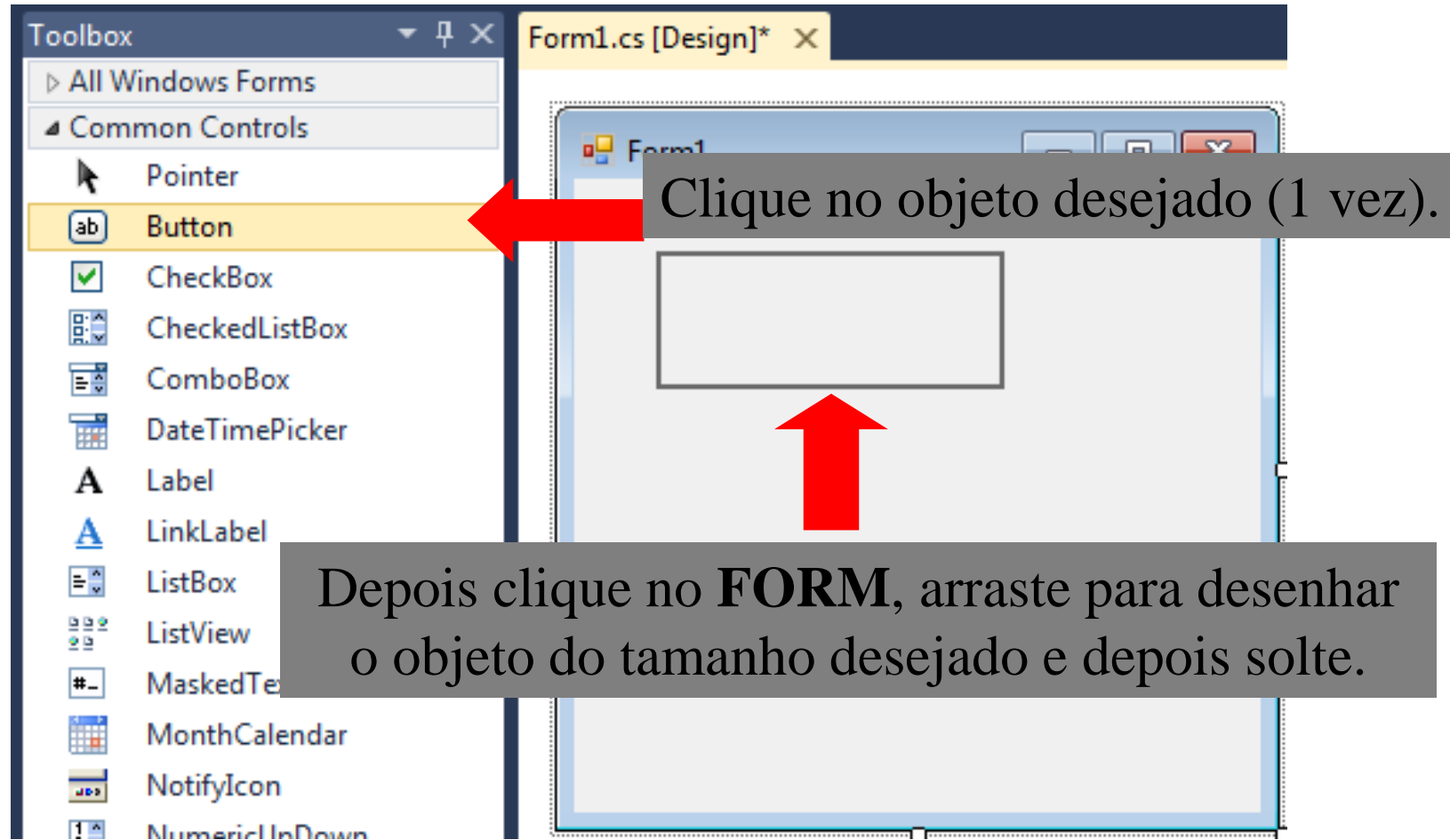
Pressione F5 para executar o programa.



Estes três botões funcionam e não precisamos escrever nenhum código para eles.

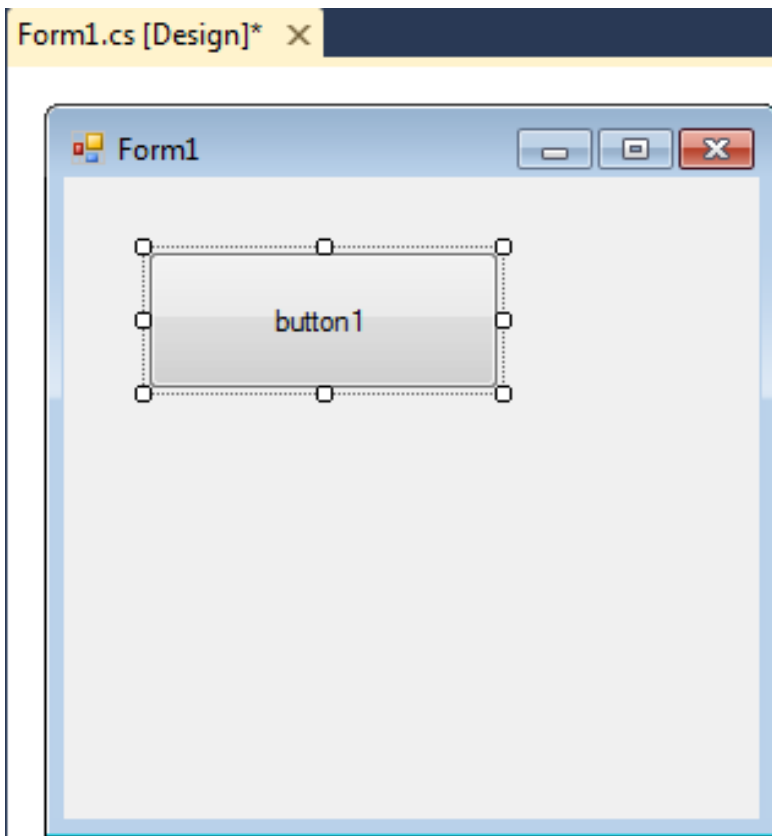
Clicar no logo da OPC faz a caixa Sobre, que acabamos de codificar, aparecer.

Adicionando Controles ao Form



Adicionando Controles ao Form

O * significa que o seu programa ainda não foi salvo



Todos objetos adicionados ao FORM deverão ter alteradas as propriedades NAME e TEXT.

A propriedade NAME serve para identificar o objeto a nível do programador (VARIÁVEL).

A propriedade TEXT serve para identificar o objeto a nível do usuário (RÓTULO).

Adicionando Controles ao Form

Text	button1	▼
------	---------	---

Nesta propriedade podemos usar acentos, caracteres especiais e espaço em branco.

Altere a propriedade para o texto: **Clique Aqui!!!**

Text	Clique Aqui!!!	▼
------	----------------	---

(Name)	button1
--------	---------

Nesta propriedade **NÃO** podemos usar acentos, caracteres especiais e espaço em branco.

Altere a propriedade para o texto: **btnClique**

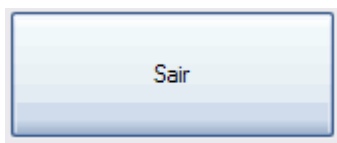
(Name)	btnClique
--------	-----------

Adicionando Controles ao Form

Regra de Nomenclatura

Remover as vogais e adicionar a funcionalidade do objeto.

Exemplos:



Button1 => btnSair

Digite seu Nome:

Label1 => lblNome

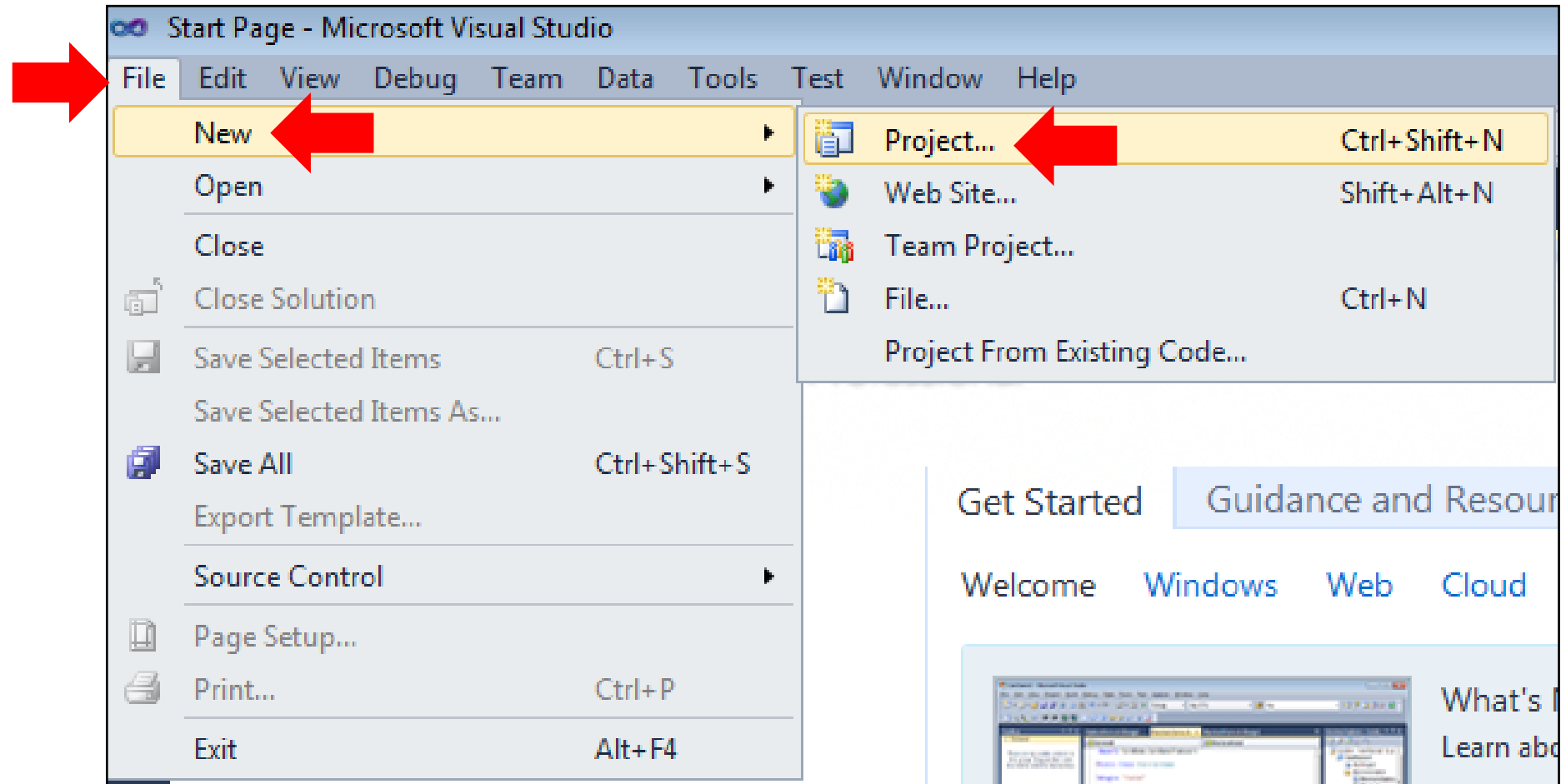
Digite seu endereço:

TextBox1 => txtEndereco

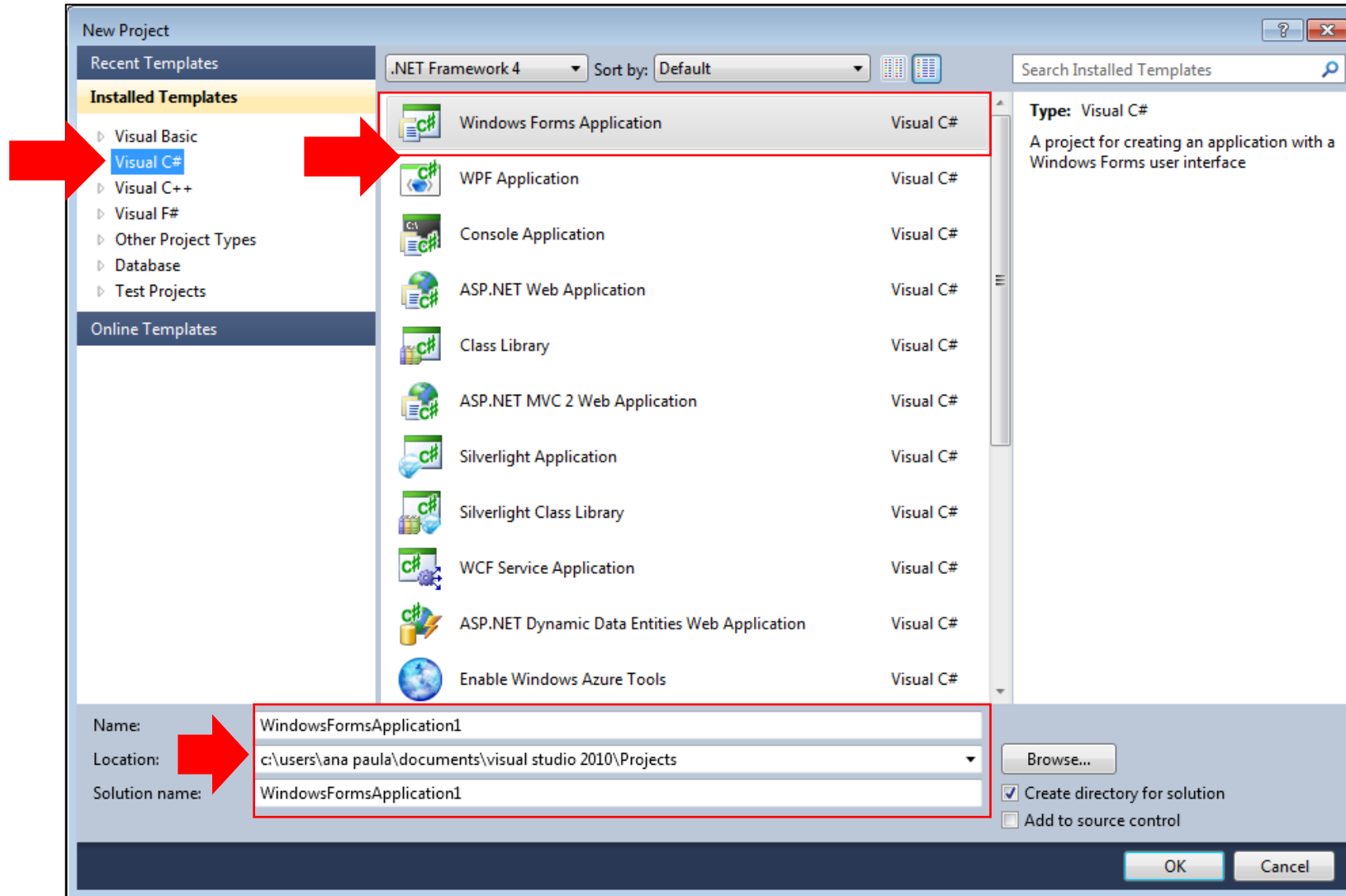
EXEMPLO 1

**FAÇA UM PROGRAMA QUE ESCREVA
A FRASE HELLO WORLD NA TELA
QUANDO UM BOTÃO FOR CLICADO**

1. Passo: Criar um Projeto



1. Passo: Criar um Projeto



1. Passo: Criar um Projeto

Troque o nome para Hello

Escolha um diretório

Name: Exemplo1

Location: E:\Ana Paula\Aho 2011\CDT\Programas\

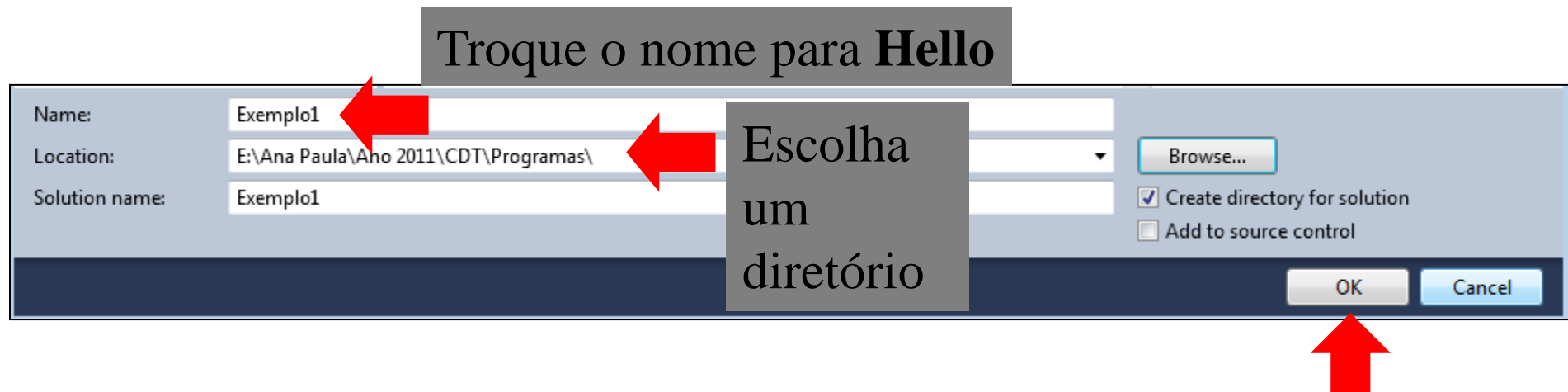
Solution name: Exemplo1

Browse...

☒ Create directory for solution

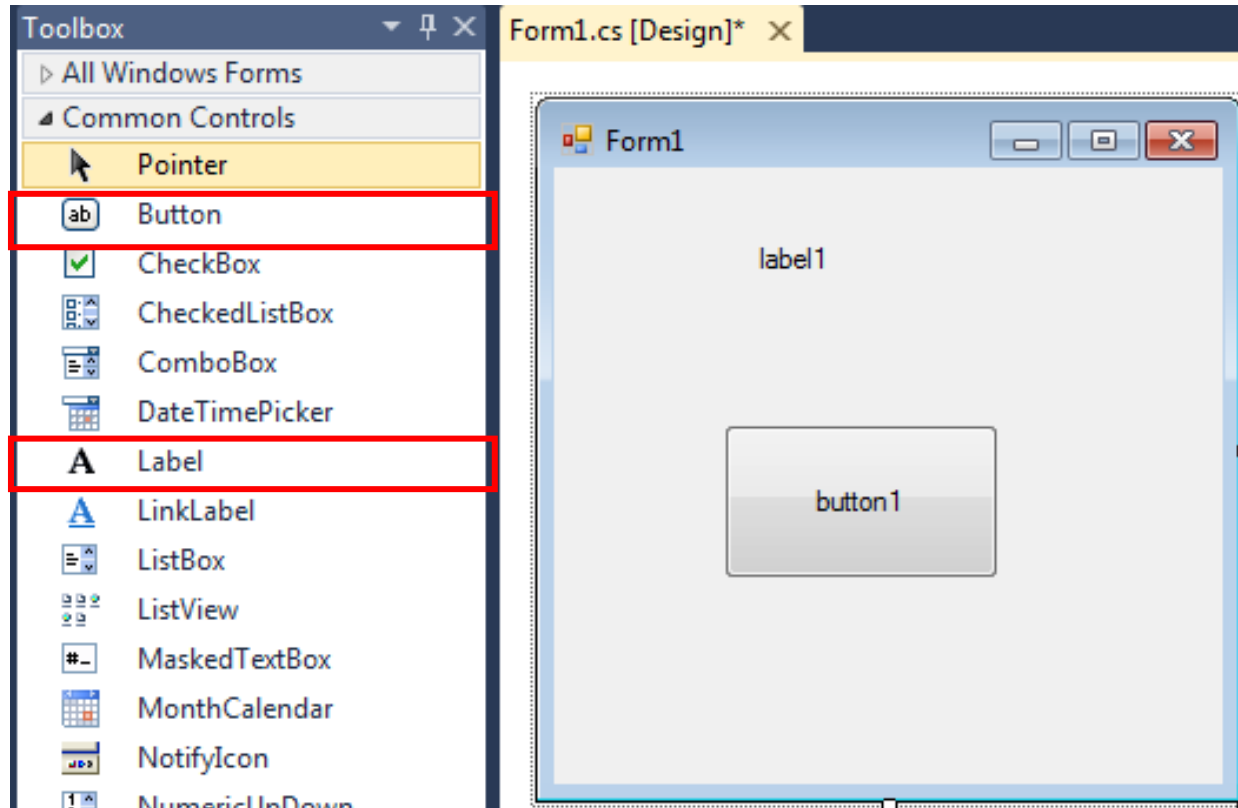
☐ Add to source control

OK Cancel

A screenshot of a project creation dialog box. The dialog has three input fields: 'Name' with 'Exemplo1', 'Location' with 'E:\Ana Paula\Aho 2011\CDT\Programas\' and a dropdown arrow, and 'Solution name' with 'Exemplo1'. To the right of the 'Location' field is a 'Browse...' button. Below these fields are two checkboxes: 'Create directory for solution' (checked) and 'Add to source control' (unchecked). At the bottom right are 'OK' and 'Cancel' buttons. Three red arrows point to the 'Name' field, the 'Location' field, and the 'OK' button. Two grey text boxes with black text are overlaid: 'Troque o nome para Hello' above the 'Name' field and 'Escolha um diretório' to the right of the 'Location' field.

Clique OK

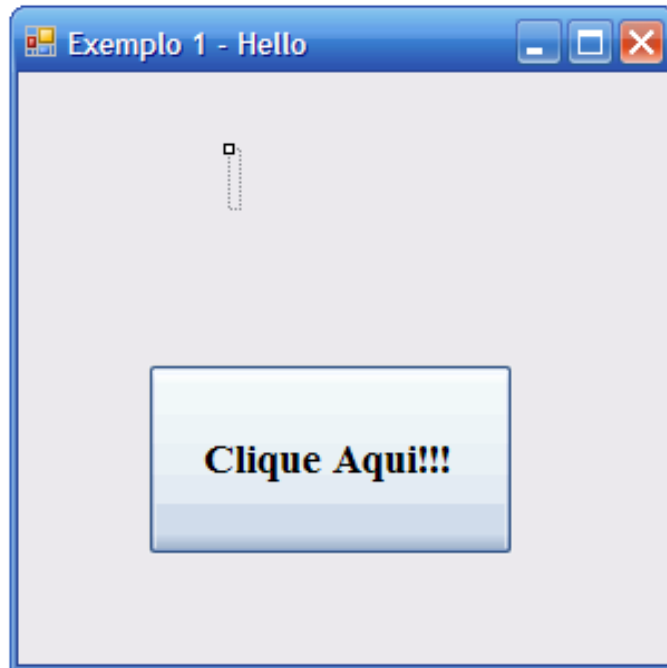
2. Passo: Construir a Tela



**Adicione
1 Label e
1 Button**

2. Passo: Construir a Tela

Para que o FORM fique assim deve-se alterar algumas propriedades dos três objetos: FORM, BUTTON e LABEL.



Tome cuidado para não alterar a propriedade do objeto errado. **CLIQUE 1 VEZ NO OBJETO, PARA SELECIONÁ-LO, E DEPOIS ALTERE AS PROPRIEDADES.**

LABEL

Text	
(Name)	lblMensagem

BUTTON

Text	Clique Aqui!!!
(Name)	btnCliqueAqui

FORM

Text	Exemplo 1 - Hello
(Name)	frmTelaPrincipal

2. Passo: Construir a Tela

As propriedades NAME e TEXT sempre serão alteradas.

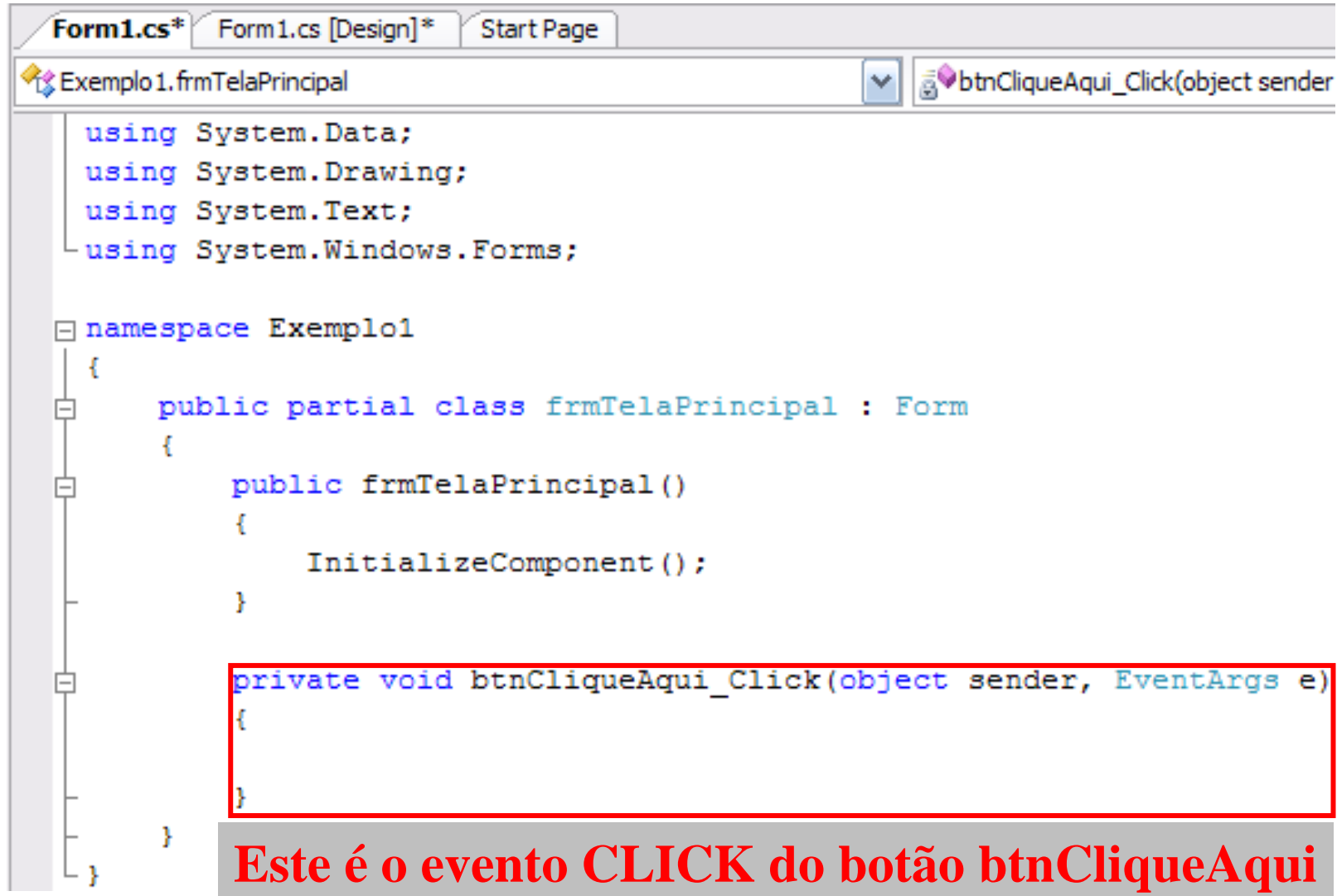
Existem outras propriedades que também são interessantes, teste as seguintes propriedades:

- **BACKCOLOR** – usada para alterar a cor de fundo
- **FONT** – usada para alterar a fonte (estilo, tamanho, entre outros)
- **FORECOLOR** – usada para alterar a cor da fonte

3. Passo: Codificar o Programa



3. Passo: Codificar o Programa



The screenshot shows the Visual Studio code editor with the following code in `Form1.cs`:

```
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace Exemplo1
{
    public partial class frmTelaPrincipal : Form
    {
        public frmTelaPrincipal()
        {
            InitializeComponent();
        }

        private void btnCliqueAqui_Click(object sender, EventArgs e)
        {
        }
    }
}
```

Este é o evento CLICK do botão btnCliqueAqui

3. Passo: Codificar o Programa

```
private void btnCliqueAqui_Click(object sender, EventArgs e)
{
    lblMensagem.Text = "HELLO WORLD";
}
```

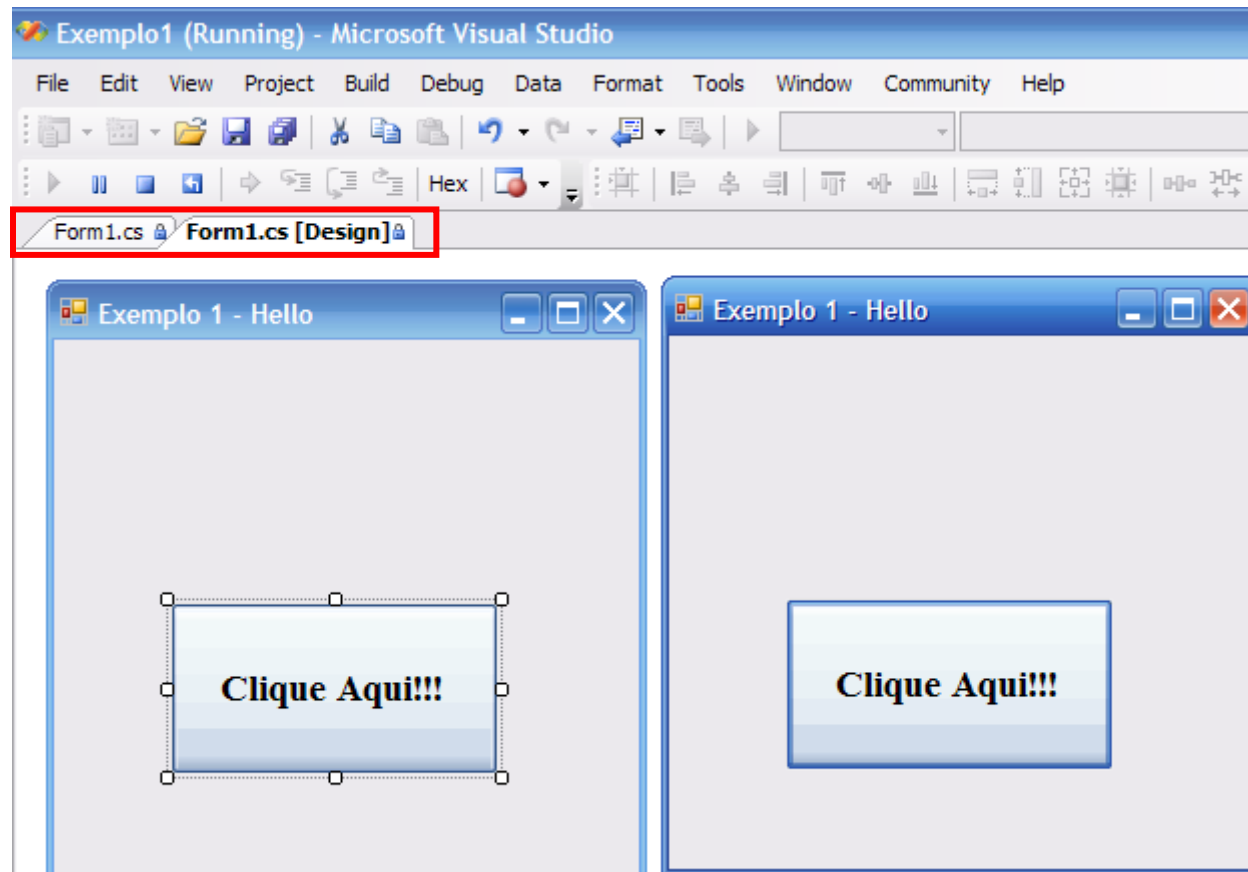
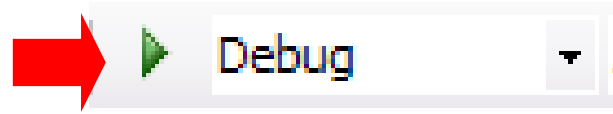


Escreva o código acima.

DICA: Use as teclas Ctrl+Barra de Espaço para facilitar a digitação

3. Passo: Codificar o Programa

Clique na seta verde ou pressione F5



Para diferenciar as duas telas verifique se existe o cadeado azul na frente das ABAS, caso exista feche a tela do programa que está executando antes de alterar qualquer coisa.

3. Passo: Codificar o Programa



CONCLUSÃO

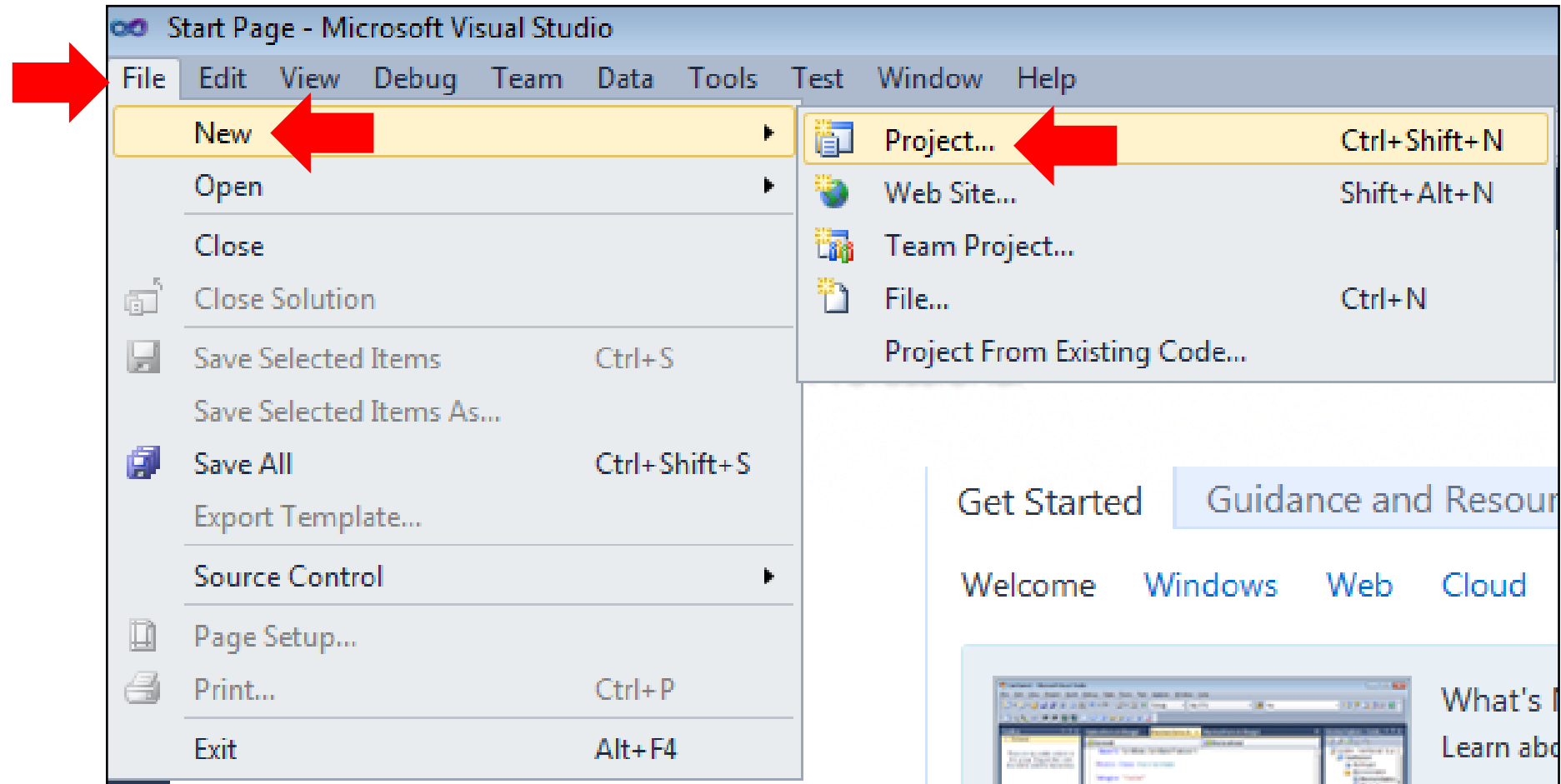
Neste exemplo aprendemos a usar dois objetos (LABEL e BUTTON) e um evento (CLICK).

- **LABEL** – usado para escrever mensagens na tela
- **BUTTON** – usado para executar a funcionalidade do programa. Para isto acionamos o **evento CLICK**, que executa o código através do click do mouse no objeto.

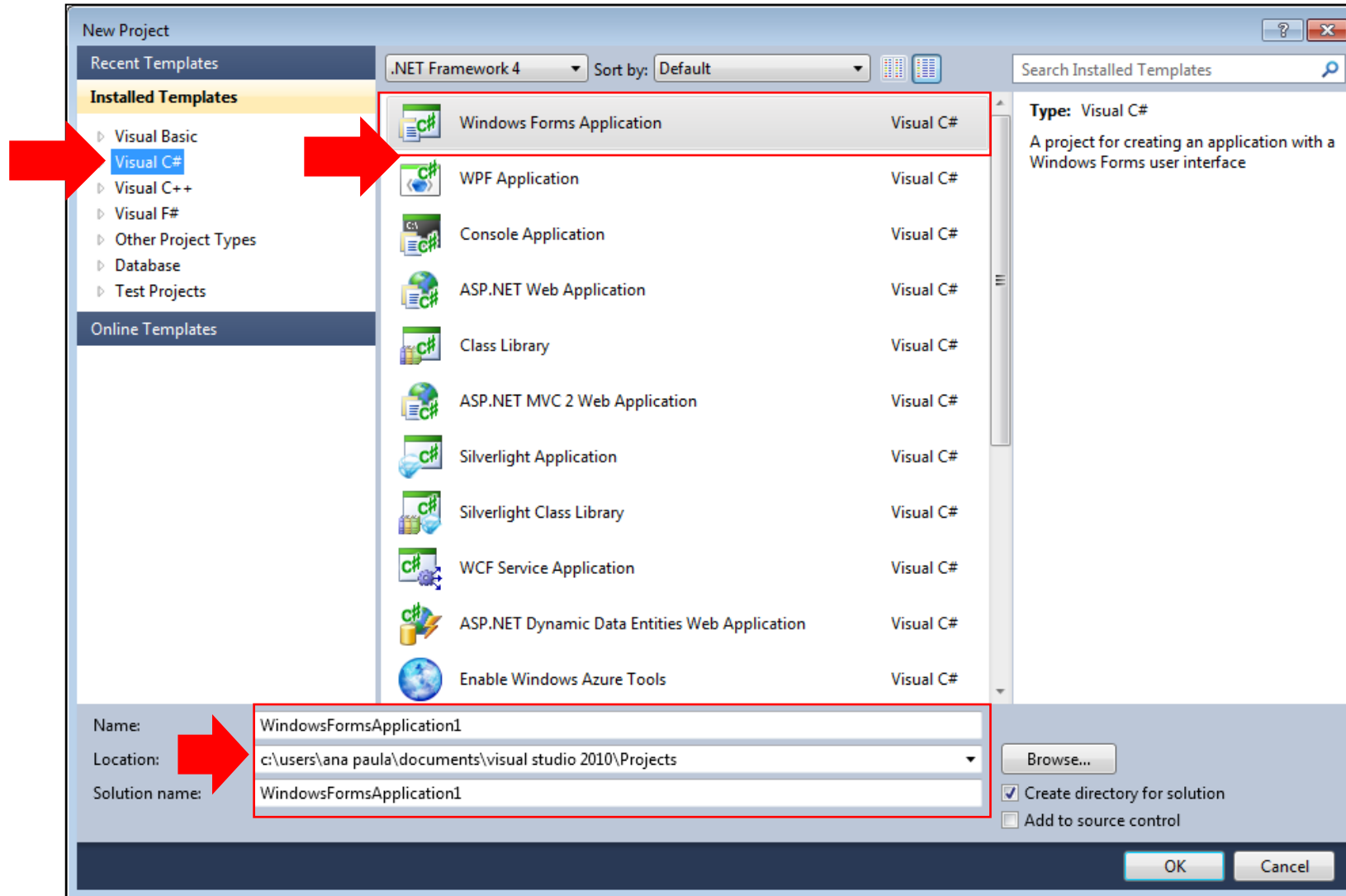
EXEMPLO 2

FAÇA UM PROGRAMA QUE ESCREVA
A FRASE BOM DIA, seu nome NA TELA
QUANDO UM BOTÃO FOR CLICADO

1. Passo: Criar um Projeto



1. Passo: Criar um Projeto



1. Passo: Criar um Projeto

Troque o nome para **Bom Dia**

Escolha um diretório

Name: Exemplo1

Location: E:\Ana Paula\Aho 2011\CDT\Programas\

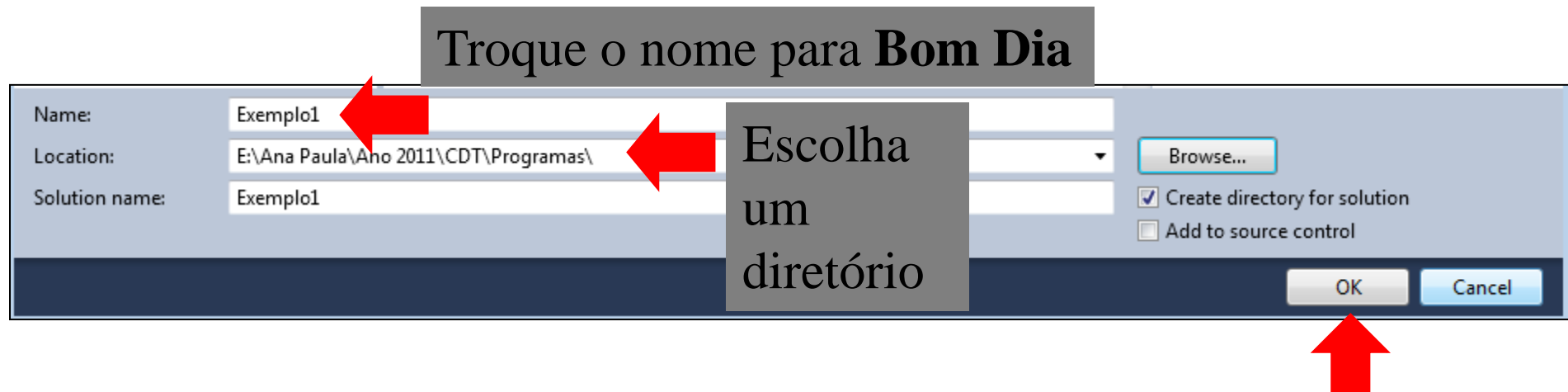
Solution name: Exemplo1

Browse...

☒ Create directory for solution

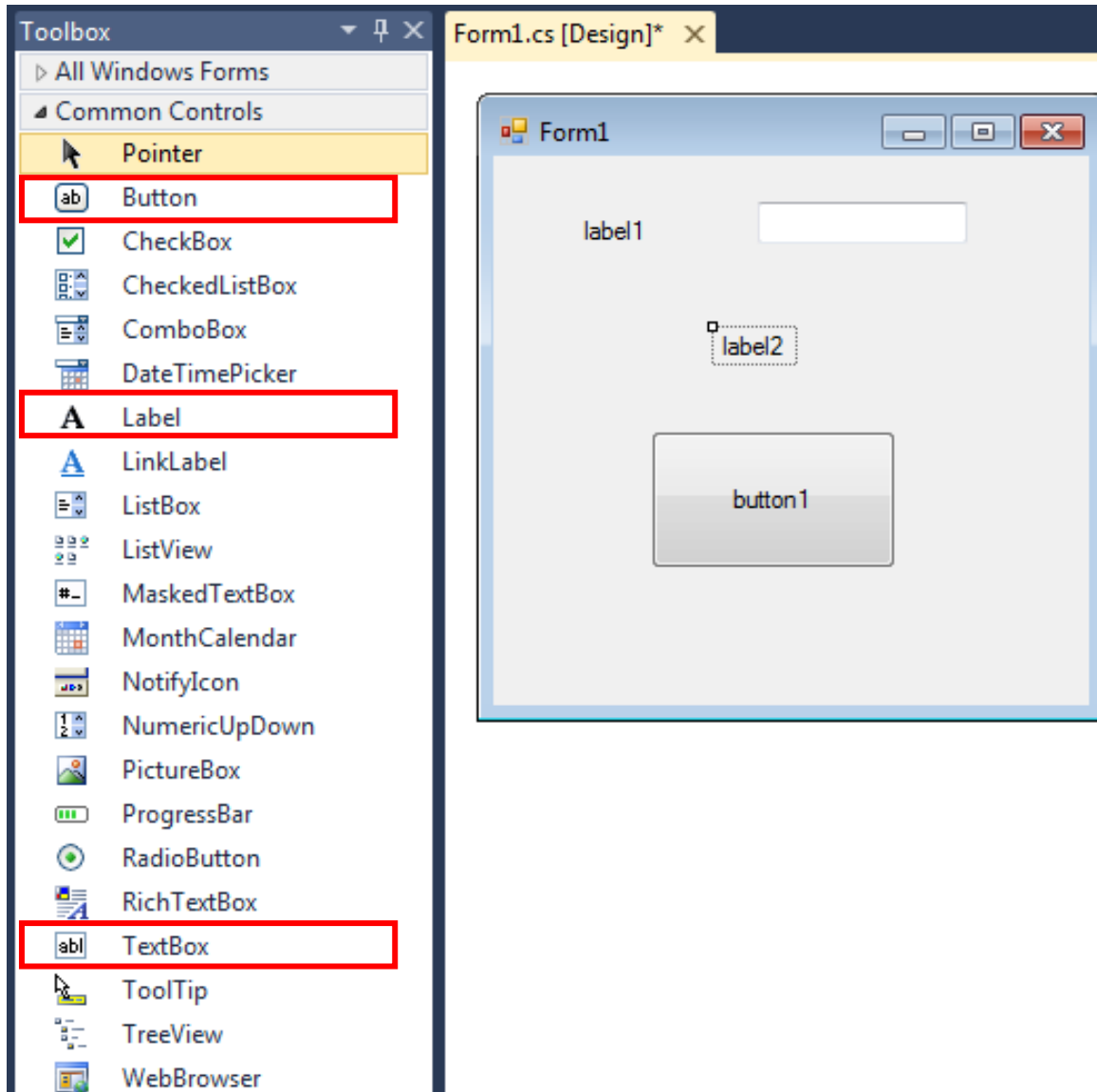
☐ Add to source control

OK Cancel

A screenshot of a project creation dialog box. The dialog has three input fields: 'Name' with 'Exemplo1', 'Location' with 'E:\Ana Paula\Aho 2011\CDT\Programas\' and a dropdown arrow, and 'Solution name' with 'Exemplo1'. To the right of the 'Location' field is a 'Browse...' button. Below these fields are two checkboxes: 'Create directory for solution' (checked) and 'Add to source control' (unchecked). At the bottom right are 'OK' and 'Cancel' buttons. Annotations include a grey box 'Troque o nome para Bom Dia' with a red arrow pointing to the 'Name' field, another grey box 'Escolha um diretório' with a red arrow pointing to the 'Location' field, and a red arrow pointing up to the 'OK' button.

Clique **OK**

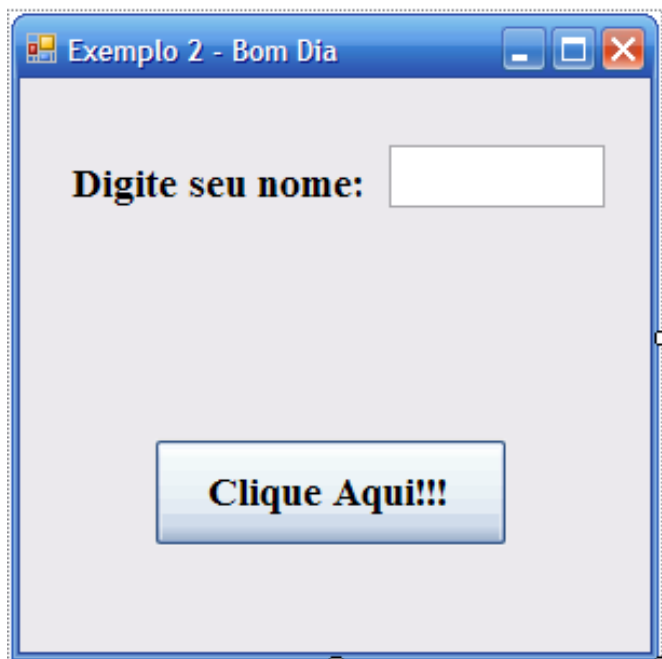
2. Passo: Construir a Tela



Adicione:
2 Labels
1 Button
1 TextBox

2. Passo: Construir a Tela

Para que o FORM fique assim deve-se alterar algumas propriedades dos três objetos: FORM, BUTTON, LABEL e TEXTBOX.



FORM1

Text: Exemplo 2 – Bom Dia

Name: frmTelaPrincipal

Tome cuidado para não alterar a propriedade do objeto errado. **CLIQUE 1 VEZ NO OBJETO, PARA SELECIONÁ-LO, E DEPOIS ALTERE AS PROPRIEDADES.**

LABEL1

Text: Digite seu nome:

Name: lblNome

LABEL2

Text:

Name: lblMensagem

BUTTON1

Text: Clique Aqui!!!

Name: btnCliqueAqui

TEXTBOX1

Text:

Name: txtNome

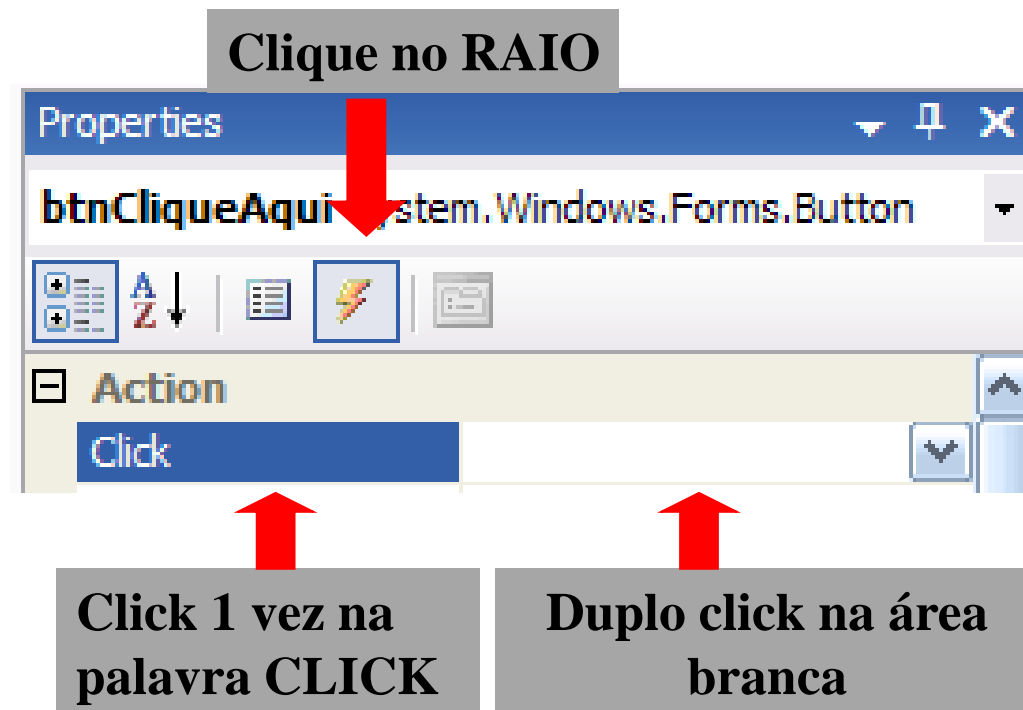
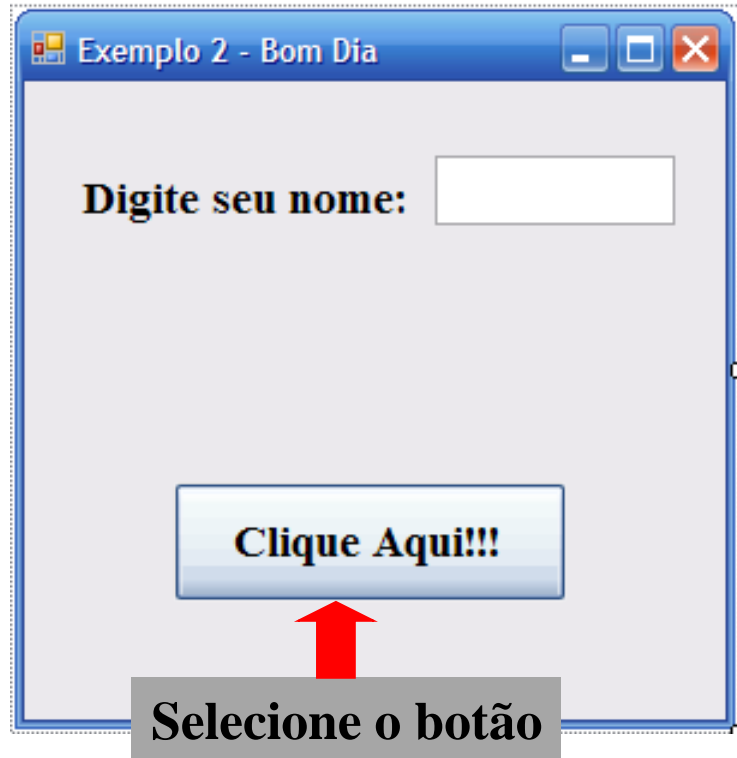
2. Passo: Construir a Tela

As propriedades NAME e TEXT sempre serão alteradas.

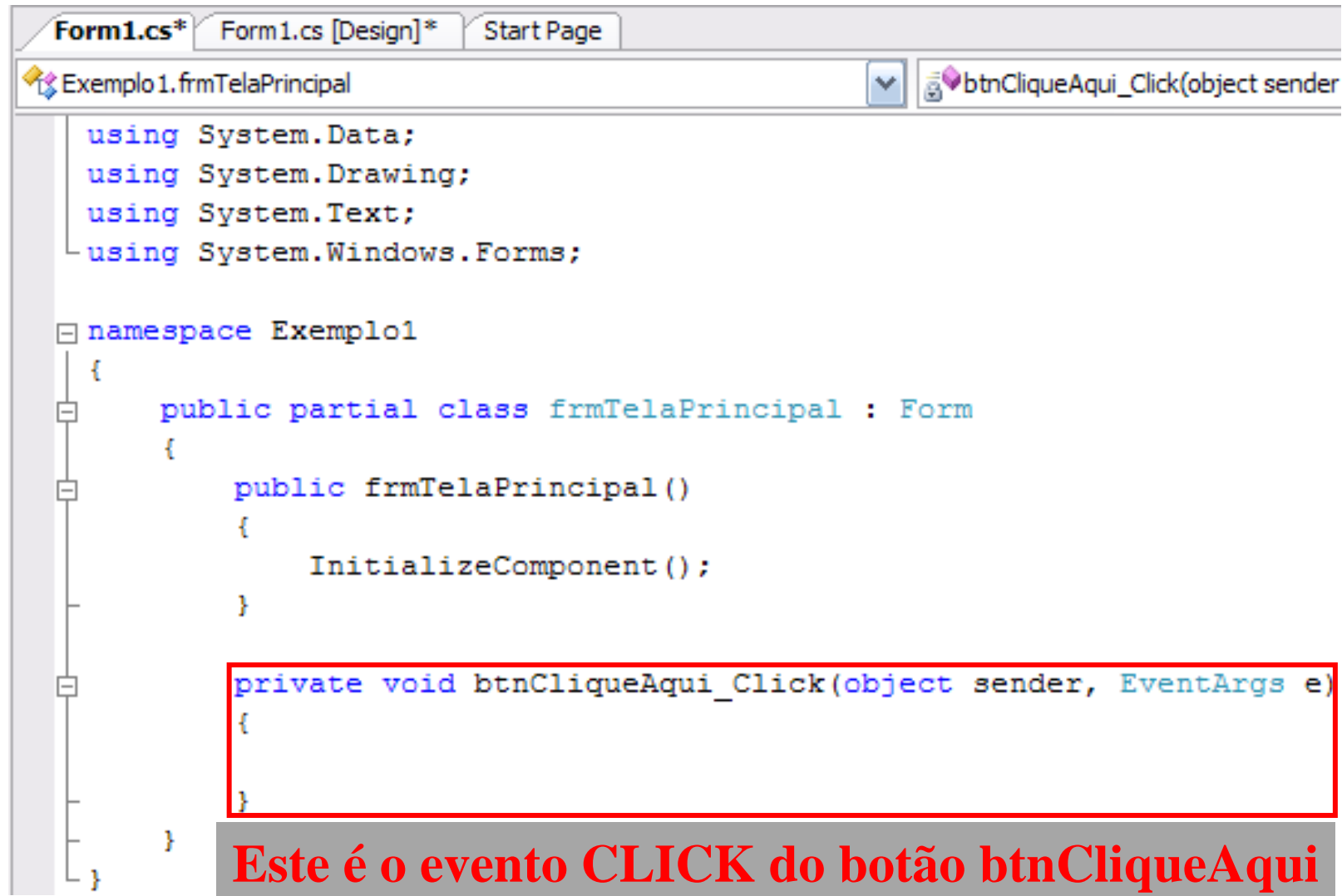
Existem outras propriedades que também são interessantes, teste as seguintes propriedades:

- **BACKCOLOR** – usada para alterar a cor de fundo
- **FONT** – usada para alterar a fonte (estilo, tamanho, entre outros)
- **FORECOLOR** – usada para alterar a cor da fonte

3. Passo: Codificar o Programa



3. Passo: Codificar o Programa



```
Form1.cs*  Form1.cs [Design]*  Start Page
Exemplo1.frmTelaPrincipal  btnCliqueAqui_Click(object sender, EventArgs e)

using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace Exemplo1
{
    public partial class frmTelaPrincipal : Form
    {
        public frmTelaPrincipal()
        {
            InitializeComponent();
        }

        private void btnCliqueAqui_Click(object sender, EventArgs e)
        {
        }
    }
}
```

Este é o evento CLICK do botão btnCliqueAqui

3. Passo: Codificar o Programa

```
private void btnCliqueAqui_Click(object sender, EventArgs e)
{
    lblMensagem.Text = "Bom dia, " + txtNome.Text;
}
```



Escreva o código acima.

DICA: Use as teclas Ctrl+Barra de Espaço para facilitar a digitação

3. Passo: Codificar o Programa

Agora teste este código:

```
private void btnCliqueAqui_Click(object sender, EventArgs e)
{
    String nome;
    nome = txtNome.Text;
    lblMensagem.Text = "Bom dia, " + nome;
}
```

Qual a diferença entre os códigos?

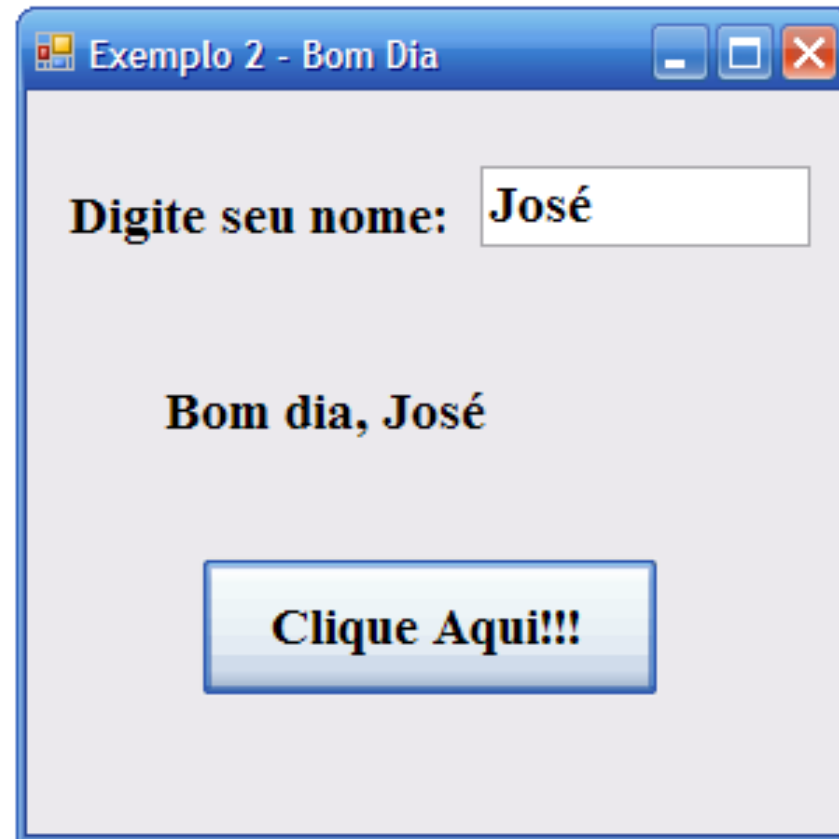
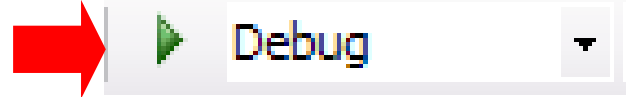
A diferença é que no primeiro não usamos variável e no segundo usamos a variável nome do tipo STRING (texto) para armazenar as informações digitadas pelo usuário.

Funciona sem variável?

Sim, pois a propriedade TEXT armazena automaticamente as informações digitadas. Lembre-se que ela armazena somente Texto (STRING)

3. Passo: Codificar o Programa

Clique na seta verde ou pressione F5



CONCLUSÃO

Neste exemplo aprendemos a usar um objeto novo (TEXTBOX) e variável.

- **TEXTBOX** – usado para capturar as informações digitadas pelo usuário. Estas informações serão **SEMPRE** no formato de texto (STRING). Quando precisarmos de valores numéricos (INT ou DOUBLE) teremos que converter.

- **VARIÁVEL** – Variável é o nome dado a uma posição de memória onde podemos armazenar um tipo de dado definido. Os tipos são:

int – dados do tipo inteiro

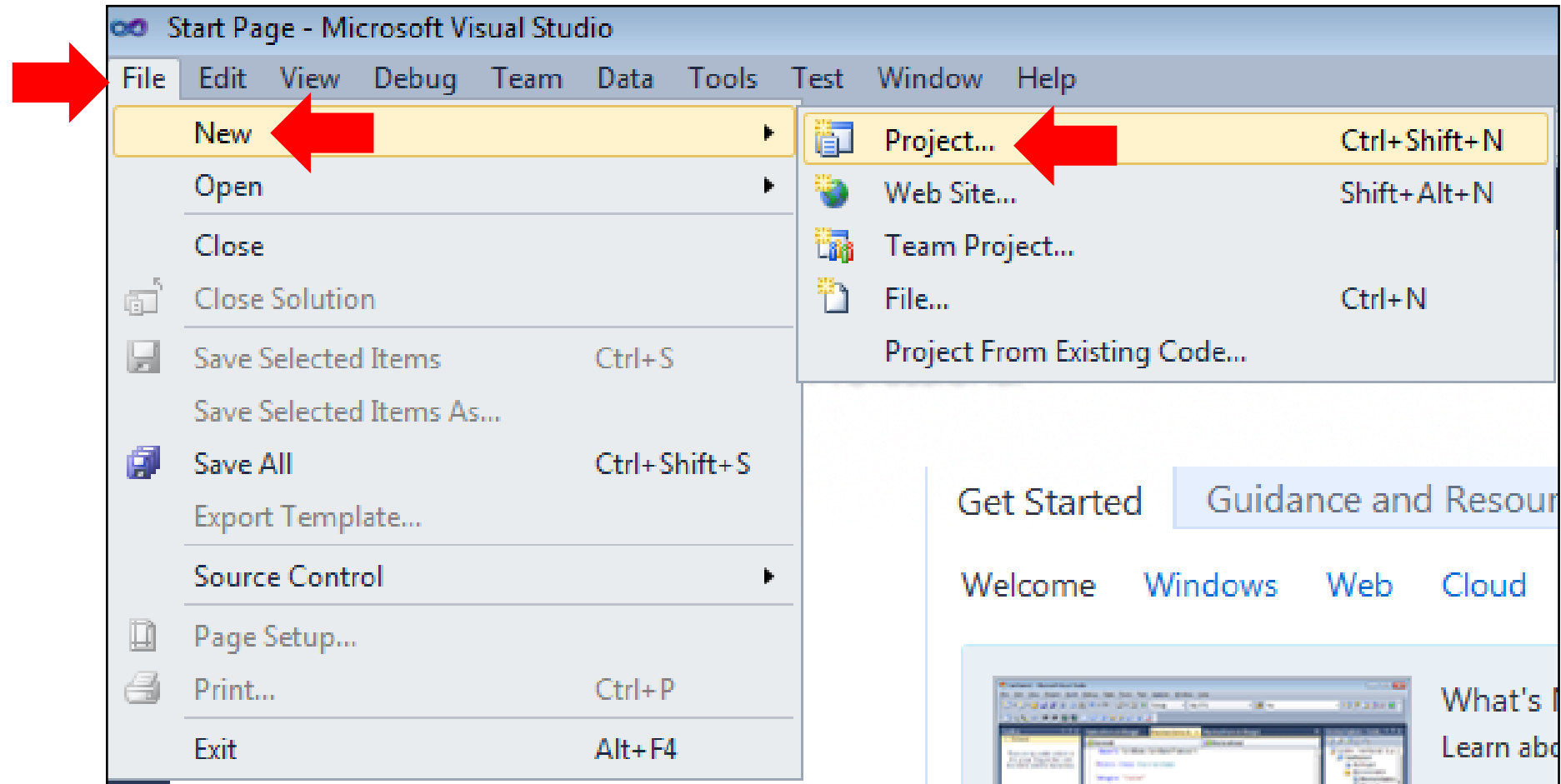
double – dados do tipo real

String ou string – dados do tipo texto

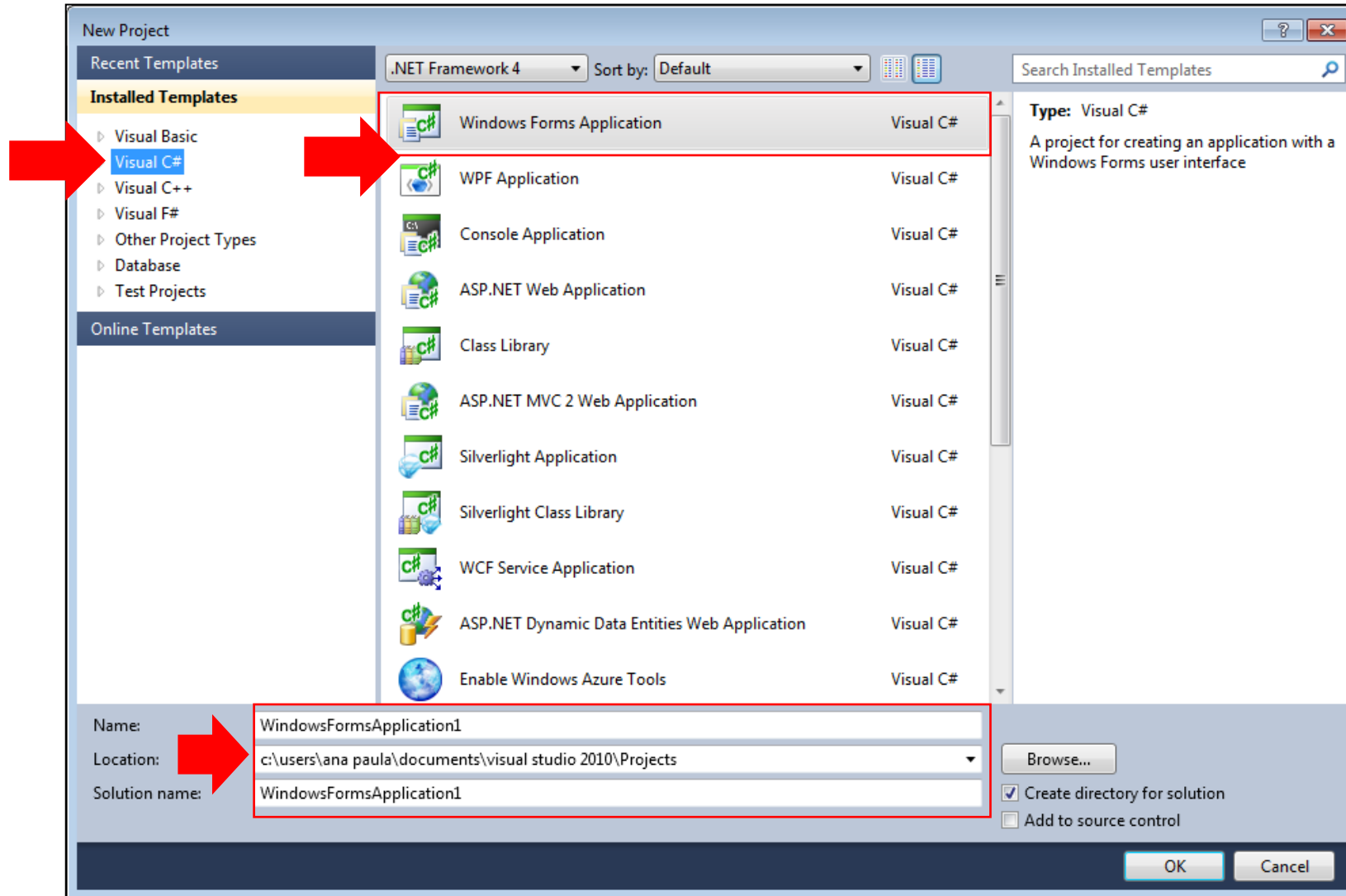
EXEMPLO 3

**FAÇA UM PROGRAMA QUE SIMULE
UMA CALCULADORA DE 4
OPERAÇÕES BÁSICAS (Somar,
Subtrair, Multiplicar e Dividir)**

1. Passo: Criar um Projeto



1. Passo: Criar um Projeto



1. Passo: Criar um Projeto

Troque o nome para **Calculadora**

Escolha um diretório

Name: Exemplo1

Location: E:\Ana Paula\Aho 2011\CDT\Programas\

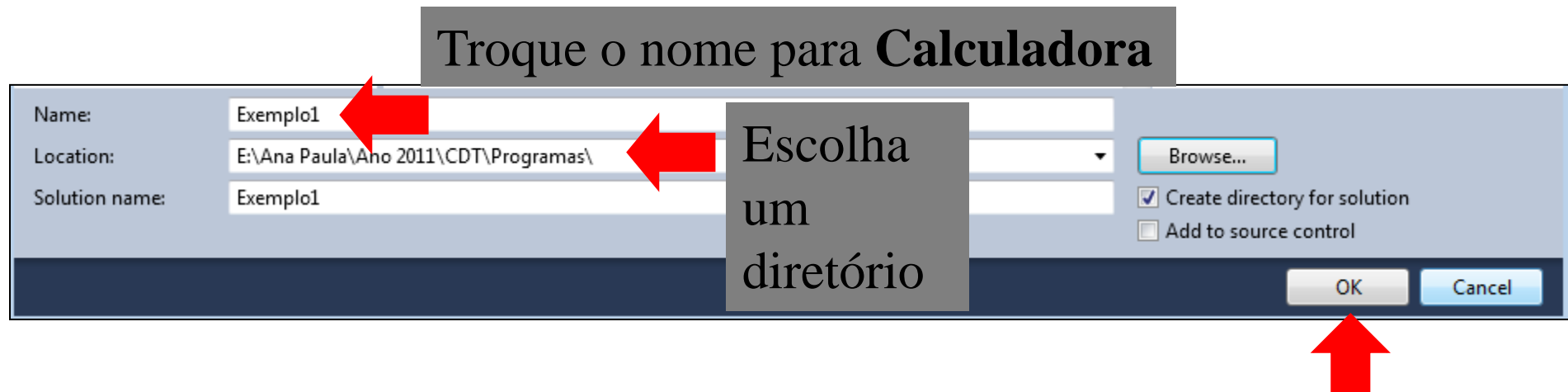
Solution name: Exemplo1

Browse...

☒ Create directory for solution

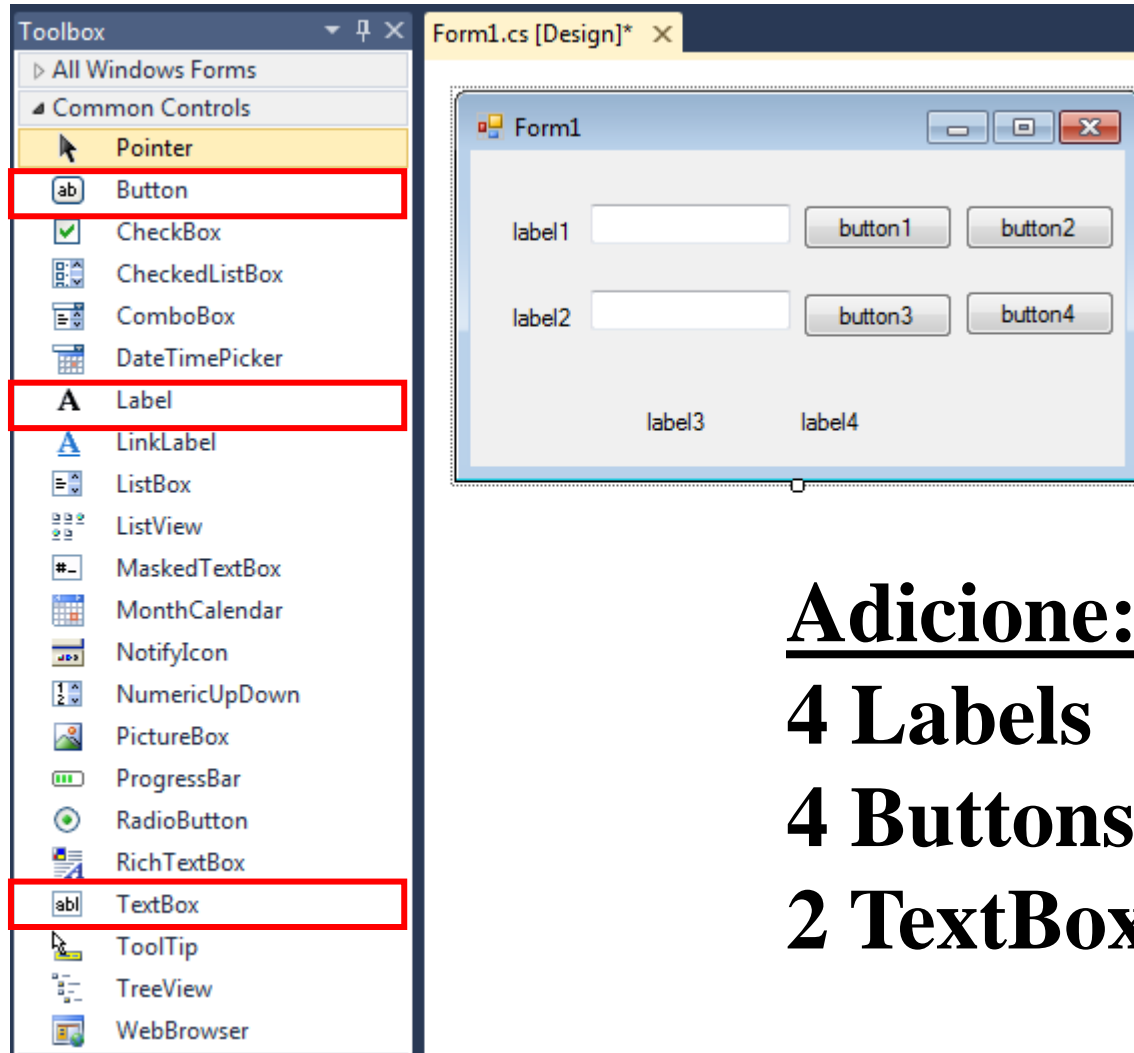
☐ Add to source control

OK Cancel

A screenshot of a project creation dialog box. The dialog has three input fields: 'Name' with 'Exemplo1', 'Location' with 'E:\Ana Paula\Aho 2011\CDT\Programas\' and a dropdown arrow, and 'Solution name' with 'Exemplo1'. To the right of the 'Location' field is a 'Browse...' button. Below these fields are two checkboxes: 'Create directory for solution' (checked) and 'Add to source control' (unchecked). At the bottom right are 'OK' and 'Cancel' buttons. Annotations include a grey box at the top saying 'Troque o nome para Calculadora' with a red arrow pointing to the 'Name' field, another grey box on the right saying 'Escolha um diretório' with a red arrow pointing to the 'Location' field, and a red arrow pointing up to the 'OK' button.

Clique **OK**

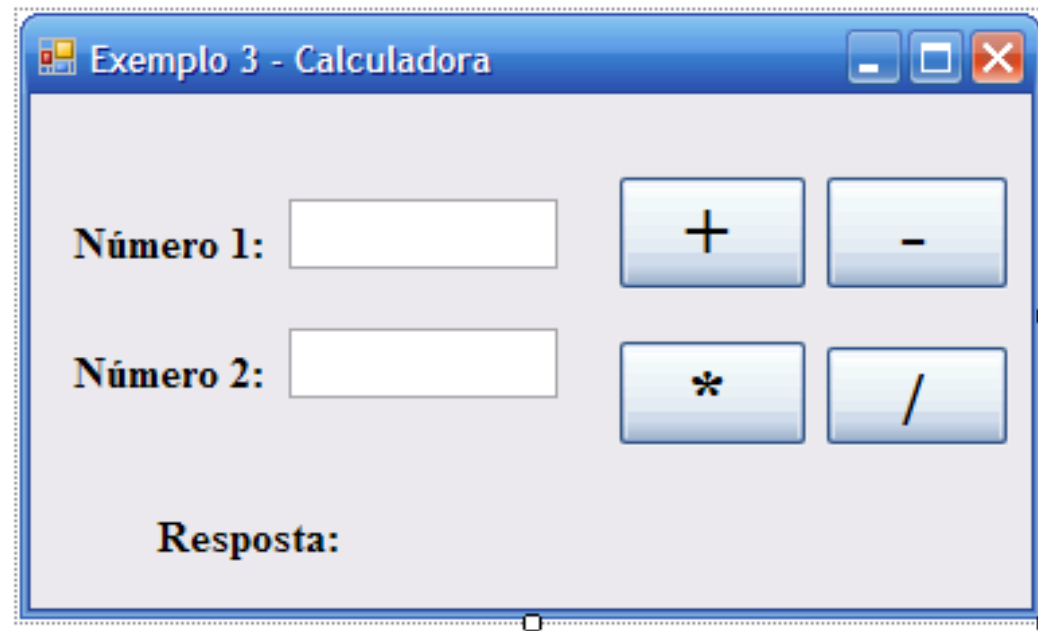
2. Passo: Construir a Tela



Adicione:
4 Labels
4 Buttons
2 TextBoxs

2. Passo: Construir a Tela

Para que o FORM fique assim deve-se alterar algumas propriedades dos três objetos: FORM, BUTTON, LABEL e TEXTBOX.



Tome cuidado para não alterar a propriedade do objeto errado. **CLIQUE 1 VEZ NO OBJETO, PARA SELECIONÁ-LO, E DEPOIS ALTERE AS PROPRIEDADES.**

2. Passo: Construir a Tela

Label1

Text: Número 1

Name: lblNum1

Button1

Text: +

Name: btnSomar

TextBox1

Text:

Name: txtNum1

Label2

Text: Número 2

Name: lblNum2

Button2

Text: -

Name: btnSubtrair

TextBox2

Text:

Name: txtNum2

Label3

Text: Resposta

Name: lblMenResp

Button3

Text: *

Name: btnMultiplicar

Form1

Text: Exemplo 3 –
Calculadora

Name: frmTelaPrincipal

Label4

Text:

Name: lblResposta

Button4

Text: /

Name: btnDividir

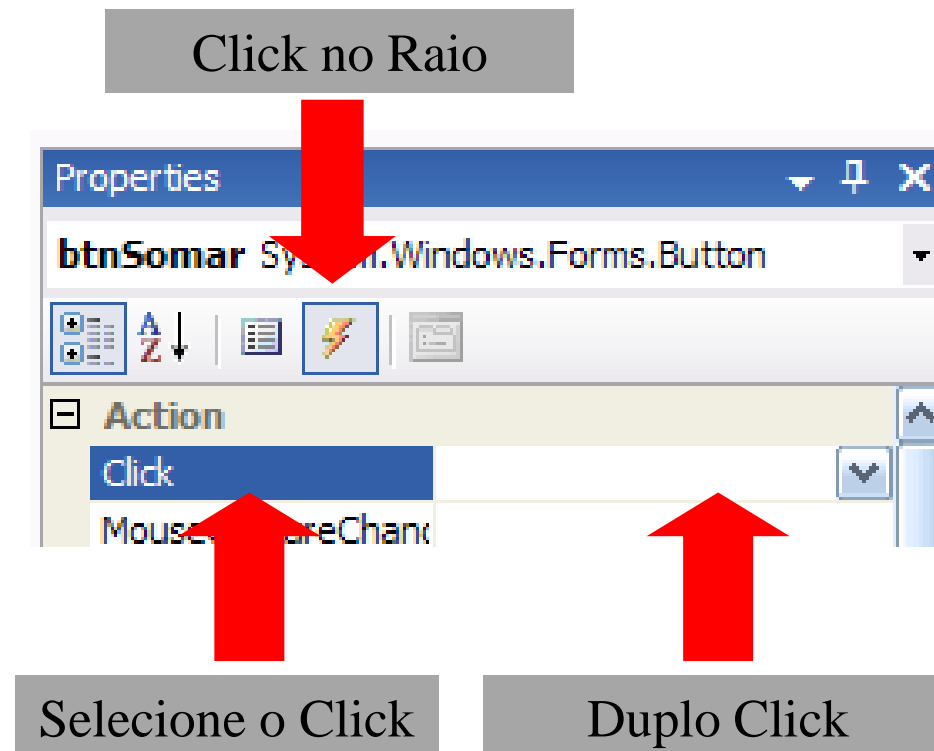
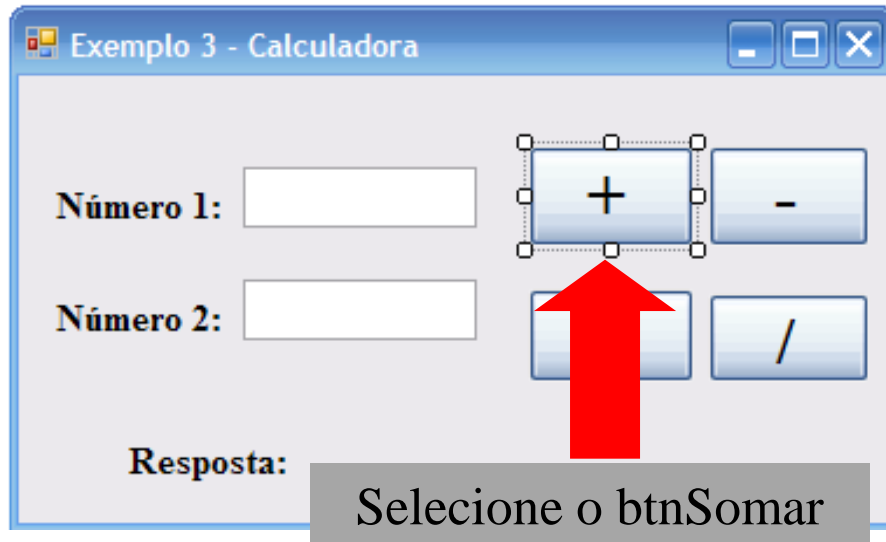
2. Passo: Construir a Tela

As propriedades NAME e TEXT sempre serão alteradas.

Existem outras propriedades que também são interessantes, teste as seguintes propriedades:

- **BACKCOLOR** – usada para alterar a cor de fundo
- **FONT** – usada para alterar a fonte (estilo, tamanho, entre outros)
- **FORECOLOR** – usada para alterar a cor da fonte

3. Passo: Codificar o Programa



3. Passo: Codificar o Programa

```
private void btnSomar_Click(object sender, EventArgs e)
{
    int n1, n2, resp;
    n1 = Convert.ToInt32(txtNum1.Text);
    n2 = Convert.ToInt32(txtNum2.Text);
    resp = n1 + n2;
    lblResposta.Text = resp.ToString();
}
```



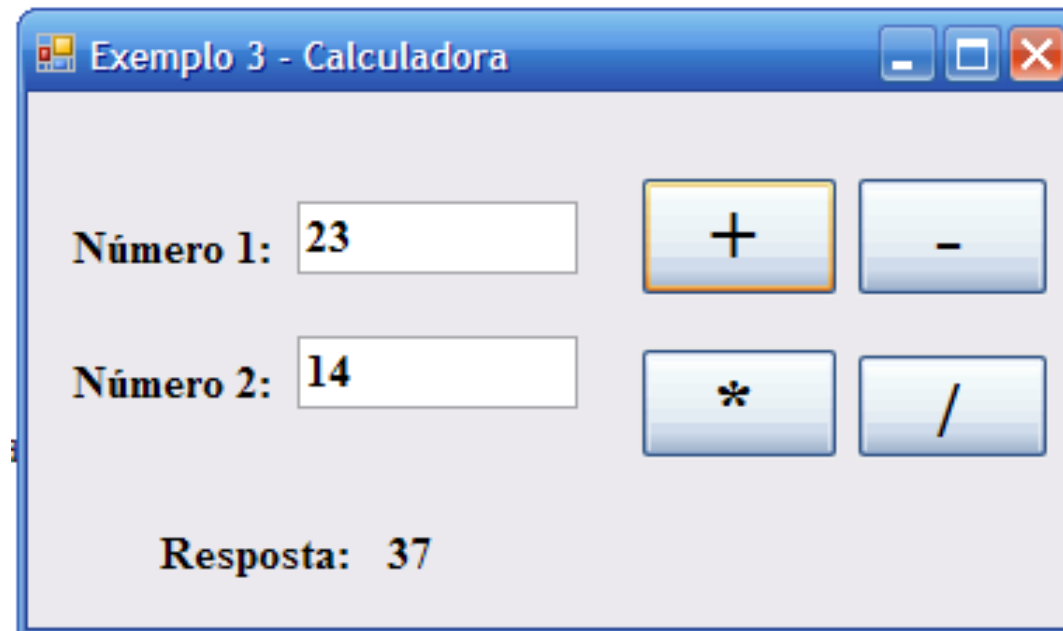
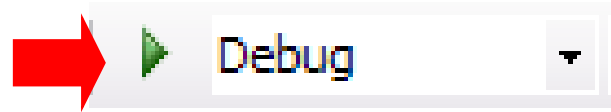
Escreva o código acima.

DICA: Use as teclas Ctrl+Barra de Espaço para facilitar a digitação

**Repita estas operações para os outros botões
btnSubtrair, btnMultiplicar, btnDividir**

3. Passo: Codificar o Programa

Clique na seta verde ou pressione F5



CONCLUSÃO

Neste exemplo aprendemos a usar conversão de tipos.

- **CONVERSÃO DE TIPOS** – método usado para converter o texto digitado pelo usuário nos TEXTBOXS para variáveis numéricas e/ou variáveis numéricas para texto.

Texto para inteiro

Variável Numérica Inteira = **Convert.ToInt32(Texto)**;

Texto para real

Variável Numérica Real = **Convert.ToDouble(Texto)**;

Inteiro ou Real para Texto

Variável Texto = Variável Numérica.**ToString()**;

Por que você deve aprender C#

O C# e o IDE do Visual Studio facilitam o trabalho de escrever código e de desenvolvê-lo rapidamente. Quando você estiver trabalhando com o C# o IDE será seu melhor amigo e companhia constante.

Aqui vemos o que o IDE automatiza para você

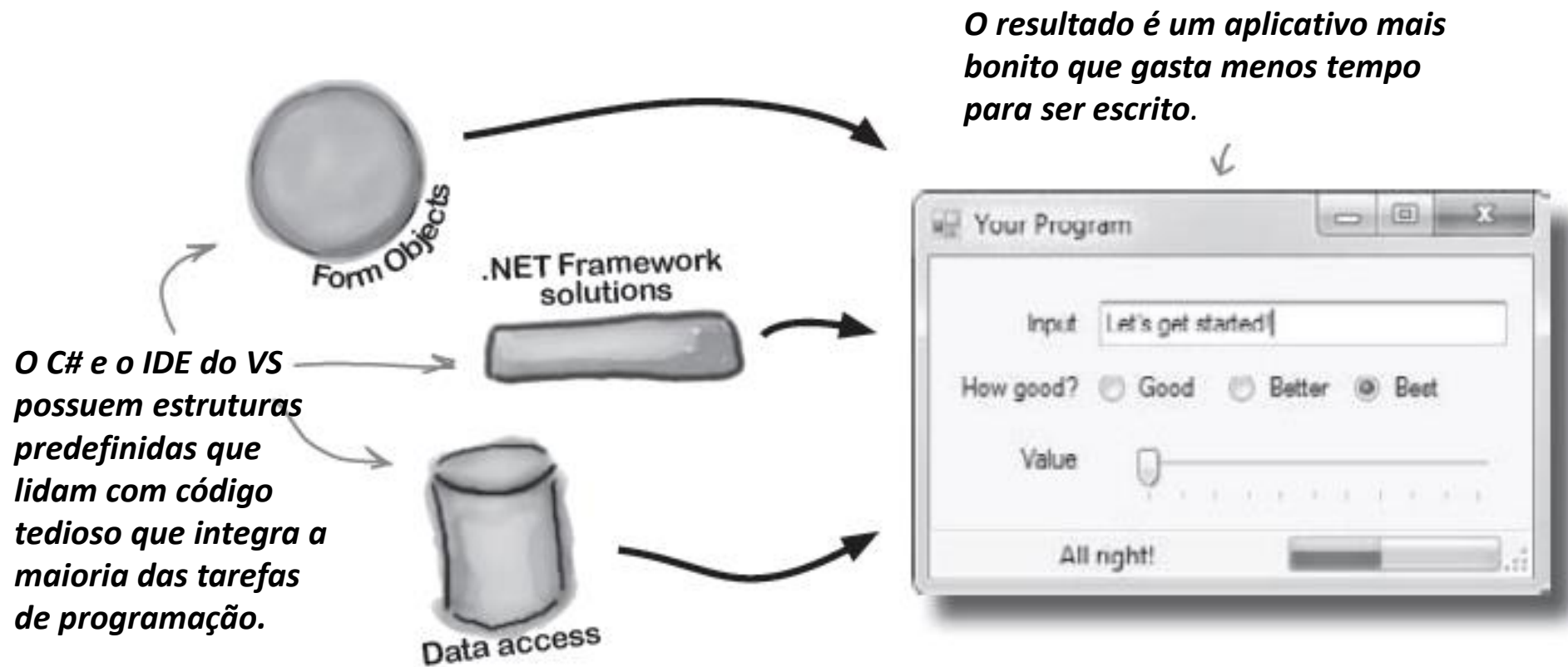
Para escrever um programa ou apenas colocar um botão em um formulário seu programa precisa de um monte de código repetitivo.

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;
namespace A_New_Program
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

```
private void InitializeComponent()
{
    this.button1 = new System.Windows.Forms.Button();
    this.SuspendLayout();
    // button1
    //
    this.button1.Location = new System.Drawing.Point(105, 56);
    this.button1.Name = "button1";
    this.button1.Size = new System.Drawing.Size(75, 23);
    this.button1.TabIndex = 0;
    this.button1.Text = "button1";
    this.button1.UseVisualStyleBackColor = true;
    this.button1.Click += new System.EventHandler(this.button1_Click);
    // Form1
    //
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.None;
    this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 16F);
    this.ClientSize = new System.Drawing.Size(292, 267);
    this.Controls.Add(this.button1);
    this.Name = "Form1";
    this.Text = "Form1";
    this.ResumeLayout(false);
}
```

O que você consegue com o VS e o C#

Com uma linguagem como C#, otimizada para programação em Windows, e com o IDE do VS, você pode focar-se no que o seu programa deve fazer



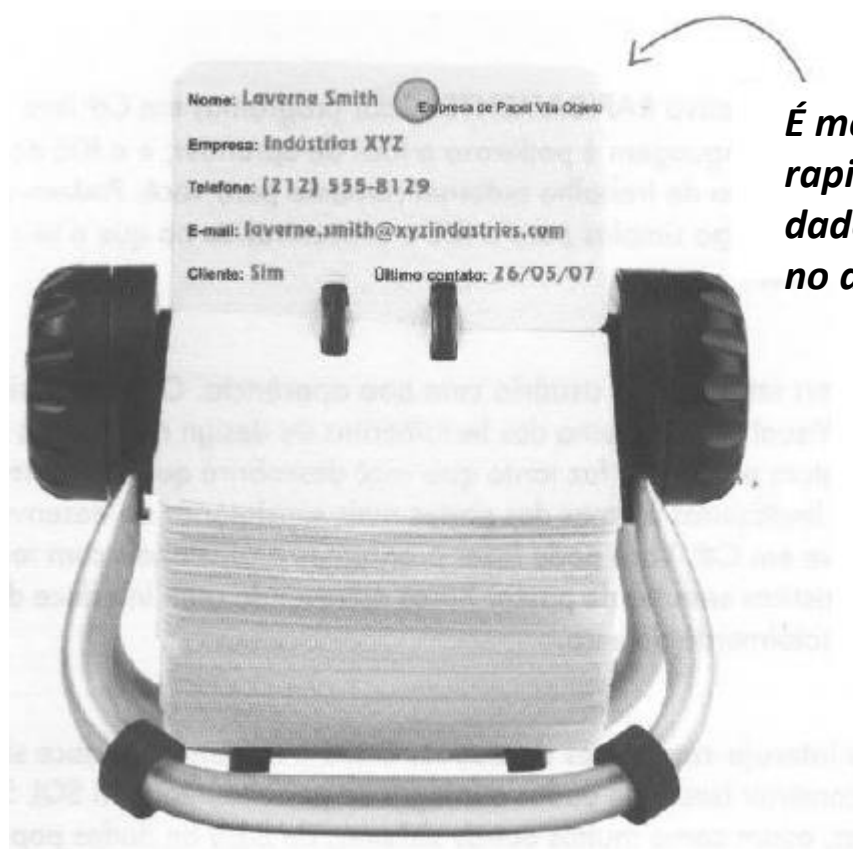
O C# e o IDE do VS facilitam muitas coisas

Quando você usa C# e o VS tem todas estas grandes características ao seu alcance, sem nenhum trabalho extra. Juntos, eles permitem que você:

- 1. Faça um aplicativo RAPIDAMENTE.**
- 2. Faça uma interface de usuário com boa aparência.**
- 3. Crie e interaja com bases de dados.**
- 4. Concentre-se em resolver seus problemas REAIS.**

Ajude o diretor a eliminar os papéis

A Empresa de Papel Vila Objeto contratou um novo diretor. Ele adora fazer caminhadas, café e a natureza ... e ele decidiu ajudar a salvar as florestas; quer ser um executivo “sem papel”, começando pelos seus contatos.

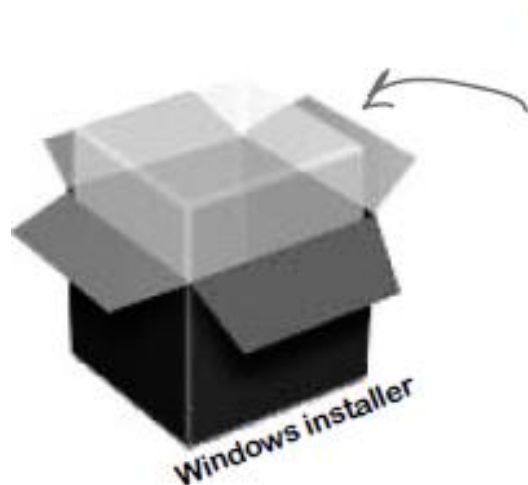


É melhor você encontrar rapidamente uma forma destes dados entrarem no notebook no diretor

Conheça as necessidades dos usuários antes de começar a fazer seu programa

Antes que possamos começar a escrever o aplicativo de agenda – ou qualquer outro programa – precisamos de um minuto para pensar em quem irá usá-lo e o que eles precisam que seja feito.

1. O diretor precisa conseguir executar seu programa de agenda no trabalho e também em seu notebook. Ele precisará de um instalador para ter certeza de que todos os arquivos corretos estejam em cada máquina.

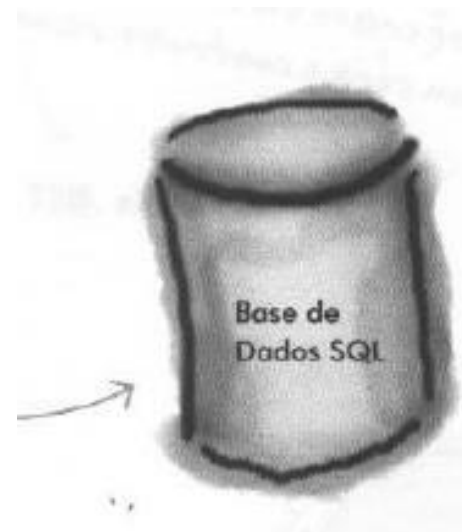


O diretor quer poder executar seu programa no desktop e no notebook, então um instalador é uma necessidade.

2. A equipe de vendas da Empresa de Papel Vila Objeto quer acessar sua agenda também. Eles podem usar seus dados para fazer listas de e-mail para obter mais ordens de compra de papel de seus clientes.

O diretor acha que uma base de dados seria a melhor forma para que todos na empresa pudessem ter acesso aos dados dele. Assim, ele pode manter apenas uma cópia de todos os seus contatos.

Já sabemos que o Visual C# facilita o trabalho com bases de dados. Ter os contatos em uma base de dados permite que o diretor e a equipe de vendas tenham acesso às informações, ainda que não exista uma cópia dos dados.



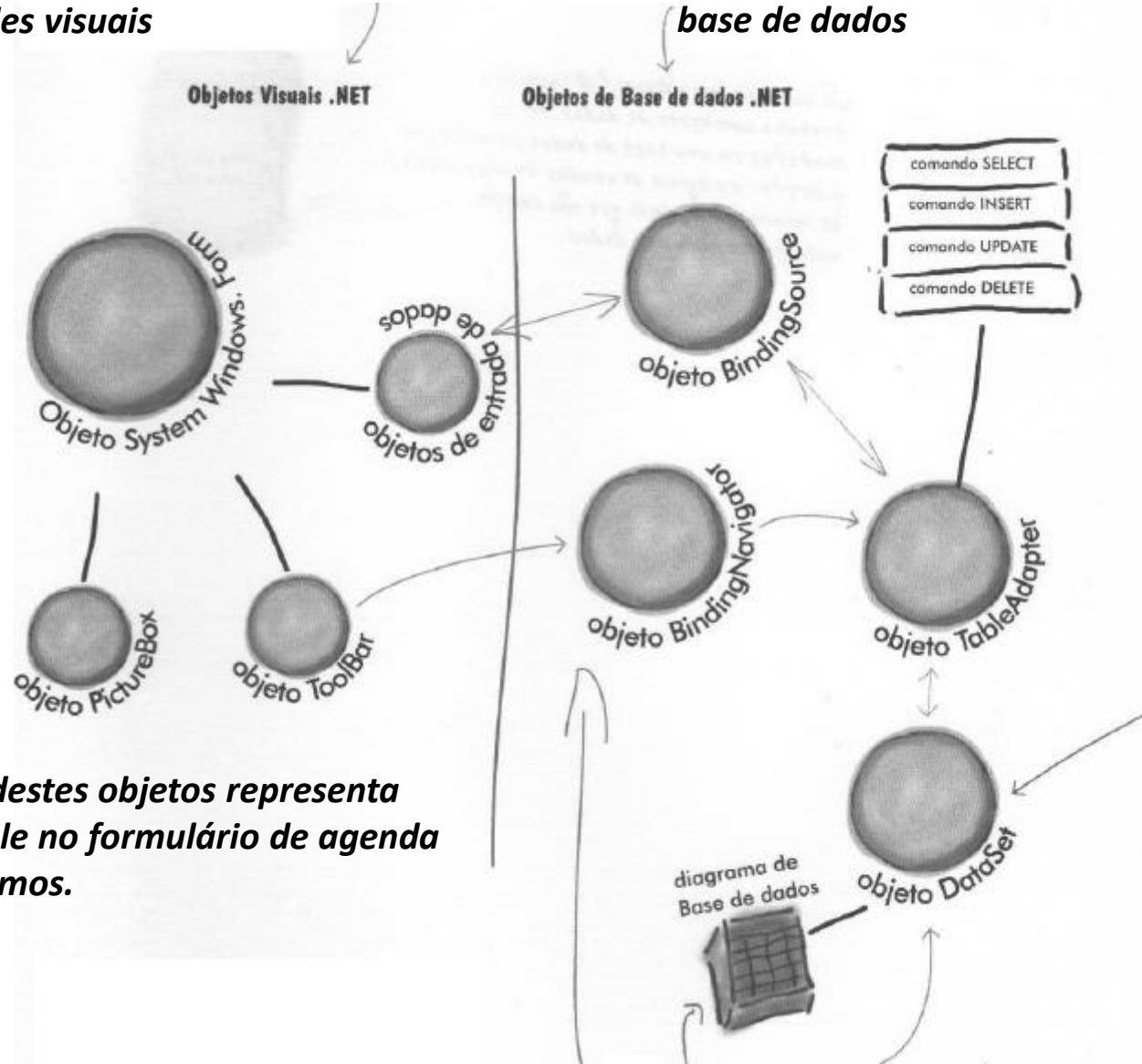
Aqui está o que você vai desenvolver

Você precisará de um aplicativo com uma interface gráfica de usuário, objetos para comunicarem-se com a base de dados, a própria base de dados e um instalador. Parece muito trabalhoso, mas até o final do curso você fará isto tudo.

Aqui está a estrutura do programa que criaremos:

Você criará um formulário Windows com vários controles visuais

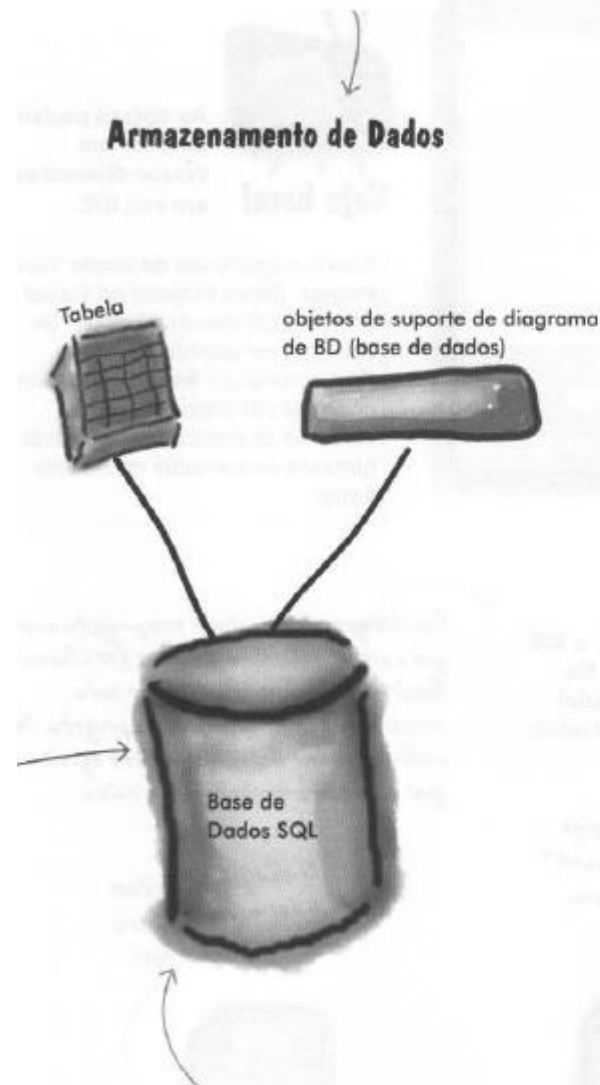
O aplicativo tem uma camada de dados separada que interage com a base de dados



Cada um destes objetos representa um controle no formulário de agenda que criaremos.

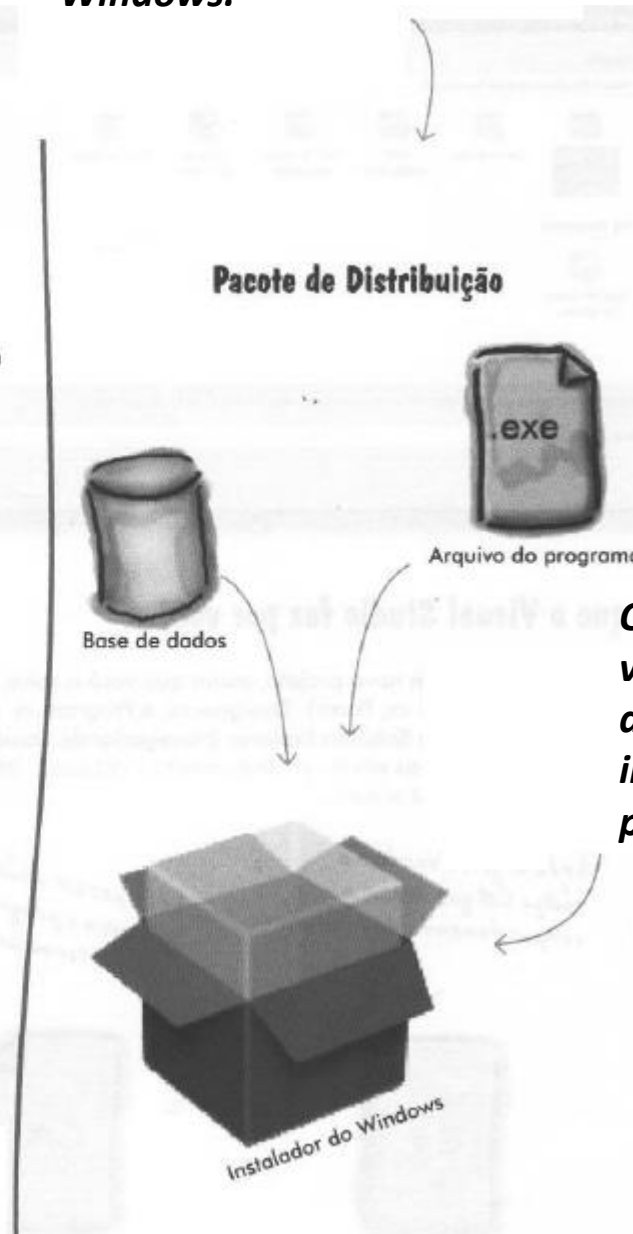
Vamos precisar de objetos para comunicarem-se com nossas tabelas.

Os dados são armazenados em uma tabela na base de dados SQL.



Aqui está a base de dados em si, que o VS nos ajudará a criar e manter.

Uma vez que o programa tenha sido feito, ele será incluído num pacote do instalador do Windows.

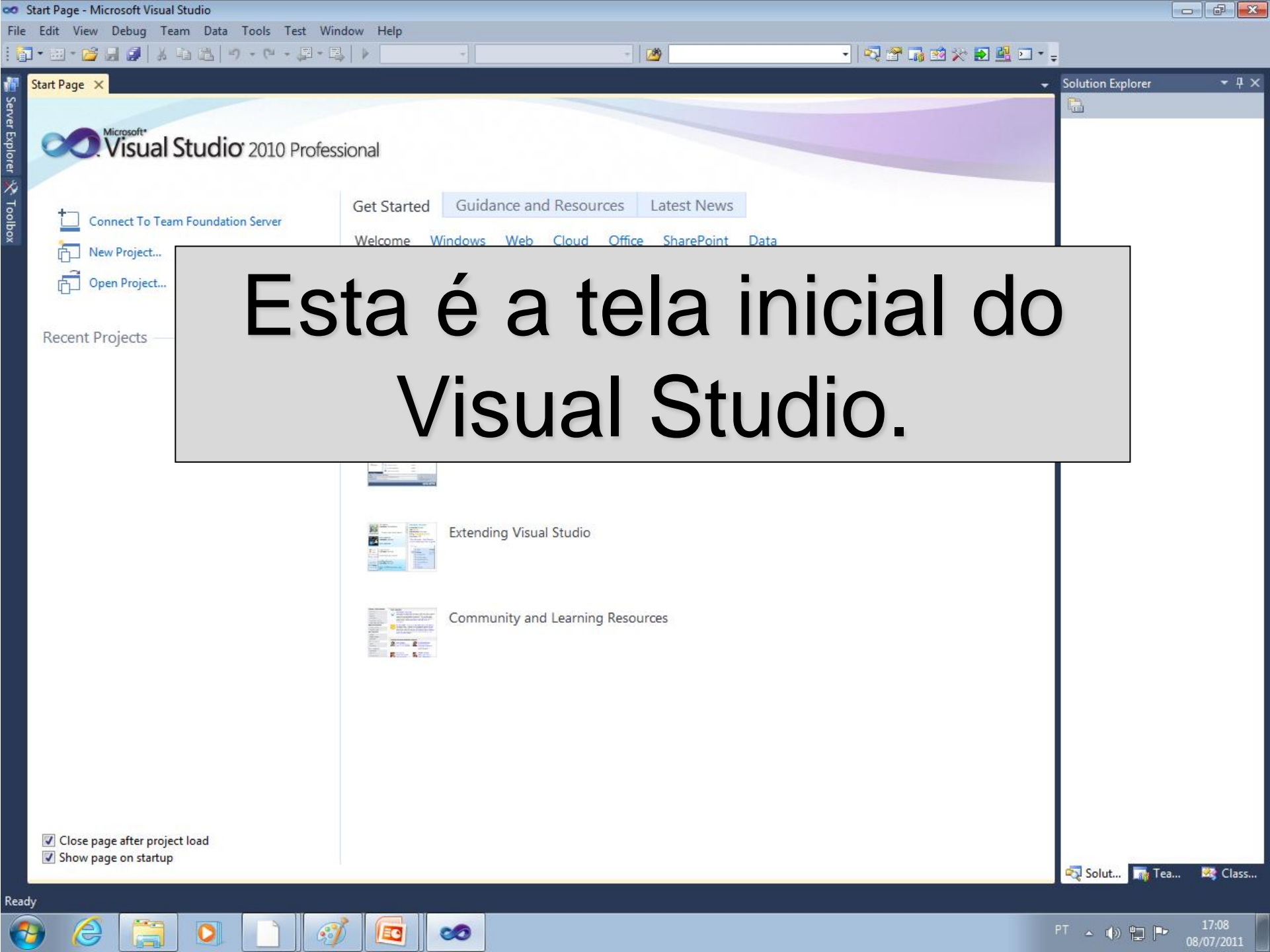


O departamento de vendas precisará apenas apontar e clicar para instalar e, então, usar seu programa.

Inicializando o Visual Studio 2010

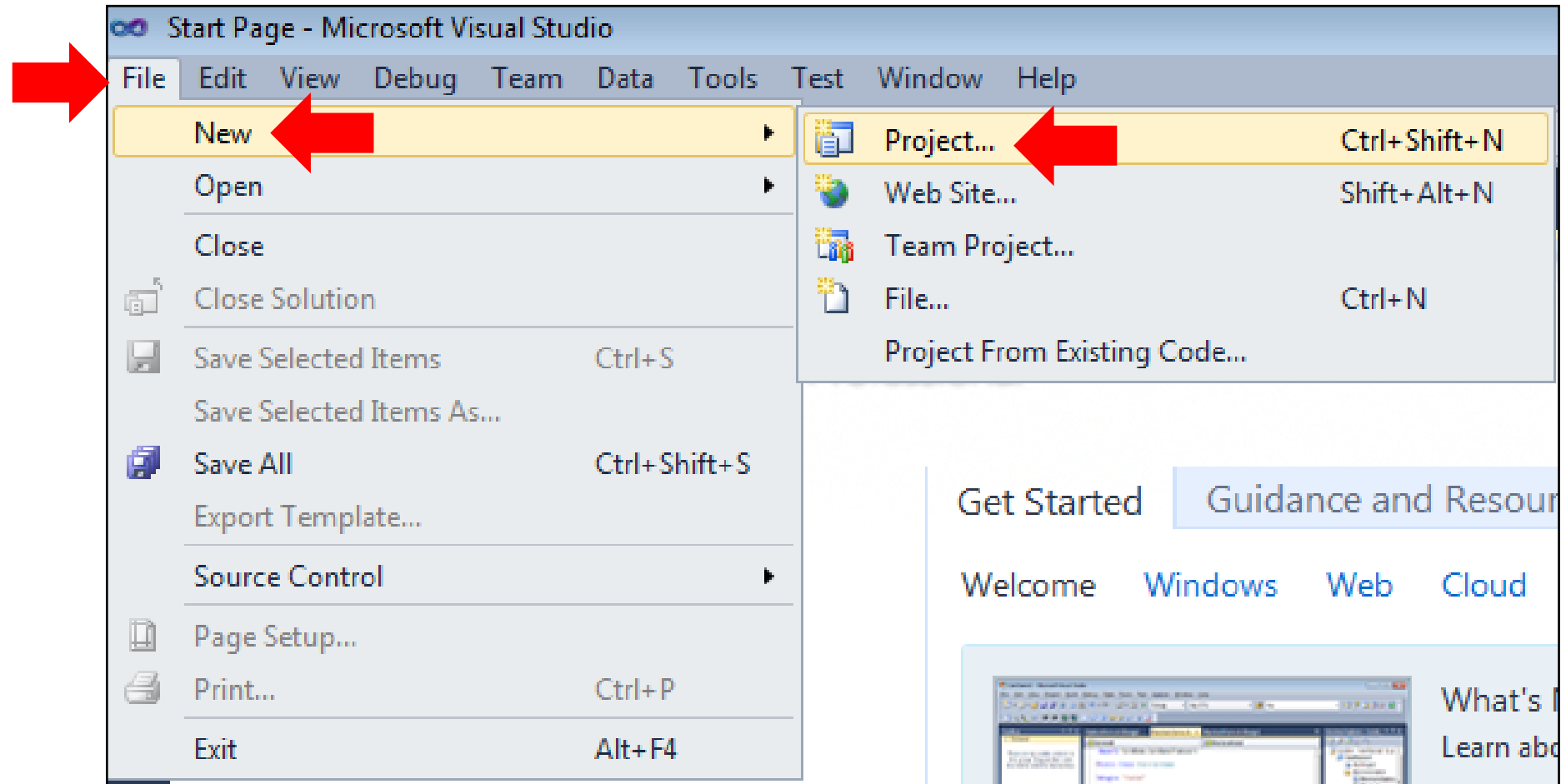
Para inicializar o Ambiente de desenvolvimento Visual Studio 2010 deve-se localizar o ícone abaixo na Área de Trabalho e clicar 2x



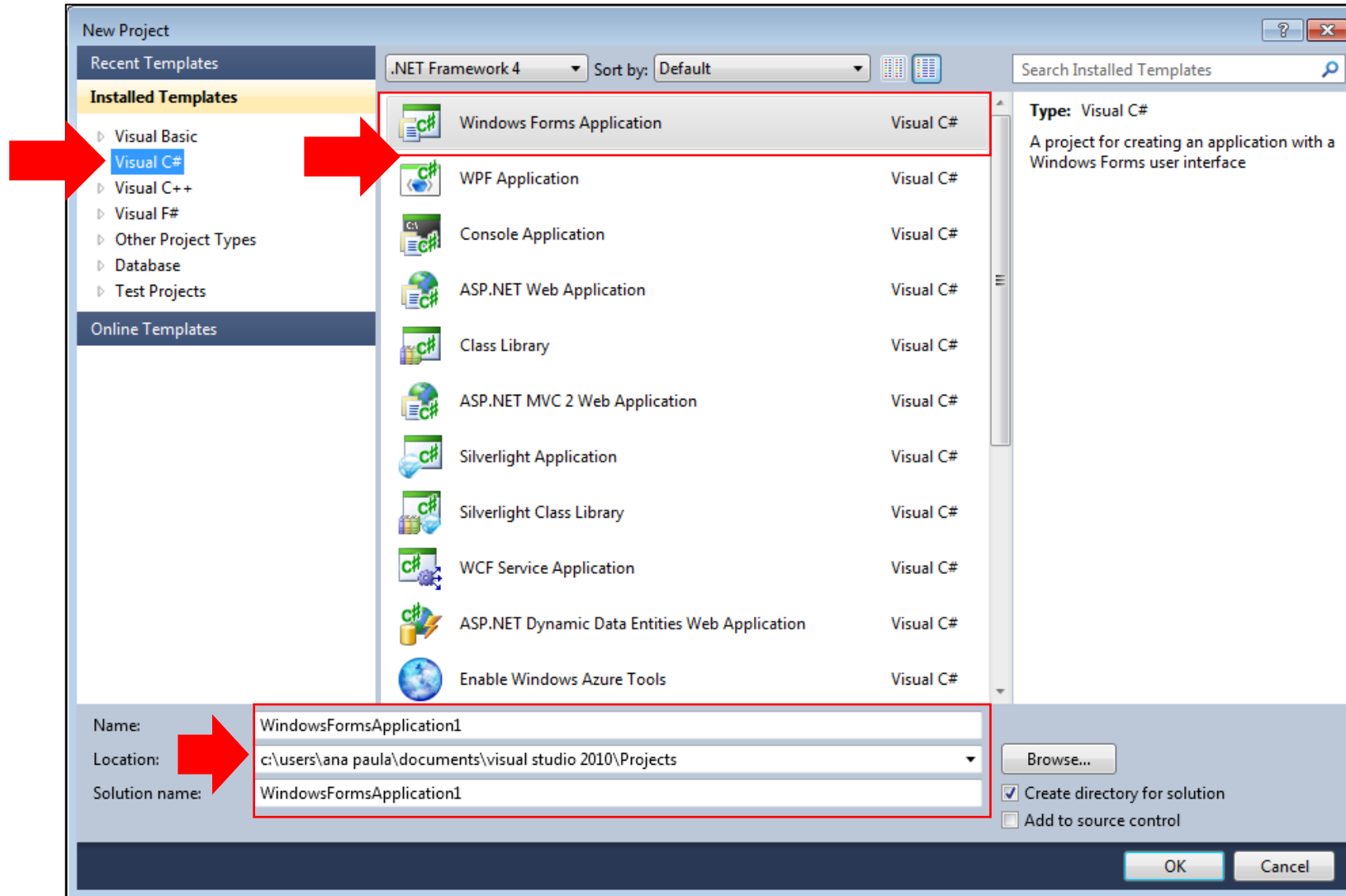


Esta é a tela inicial do
Visual Studio.

Criando um projeto



Criando um projeto



Criando um projeto

Name:	WindowsFormsApplication1	
Location:	c:\users\ana paula\documents\visual studio 2010\Projects	<input type="button" value="Browse..."/>
Solution name:	WindowsFormsApplication1	<input checked="" type="checkbox"/> Create directory for solution <input type="checkbox"/> Add to source control

Name:	Exemplo1	<input type="button" value="Browse..."/>
Location:	E:\Ana Paula\Ano 2011\CDT\Programas\	<input checked="" type="checkbox"/> Create directory for solution <input type="checkbox"/> Add to source control
Solution name:	Exemplo1	<input type="button" value="OK"/> <input type="button" value="Cancel"/>

O que o Visual Studio faz por você...

Quando você inicia um novo projeto, assim que você o salva, o IDE cria os arquivos Form1.cs, Form1.Designer.cs, e Program.cs. Ele acrescenta-os à janela Solution Explorer (Navegador de Solução) e, por padrão, coloca-os em **Meus Documentos\Visual Studio 2010\Projects\Exemplo**.

Este arquivo contém o código C# que define o comportamento do formulário.



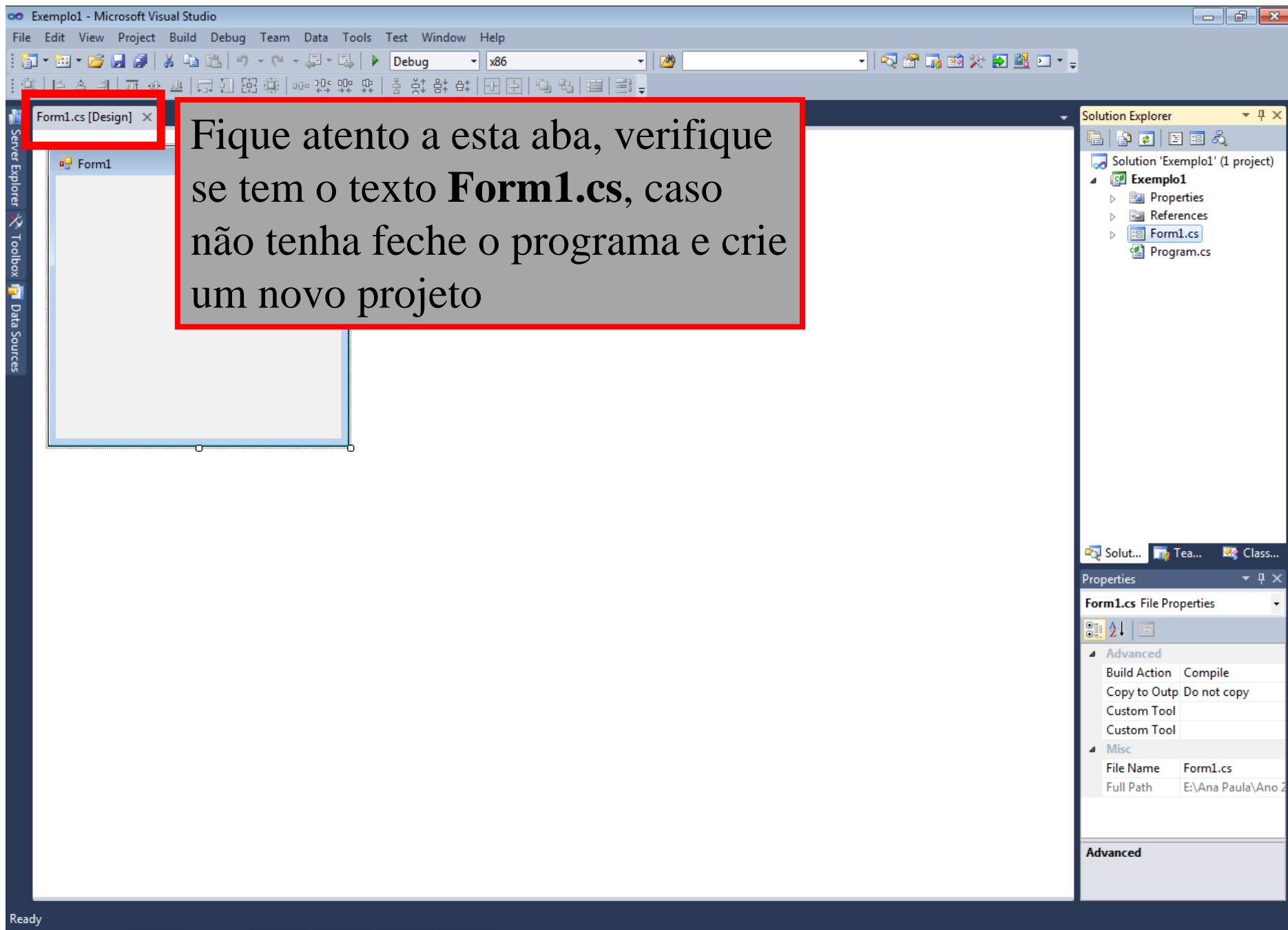
Este possui o código que inicia o programa e exibe o formulário.

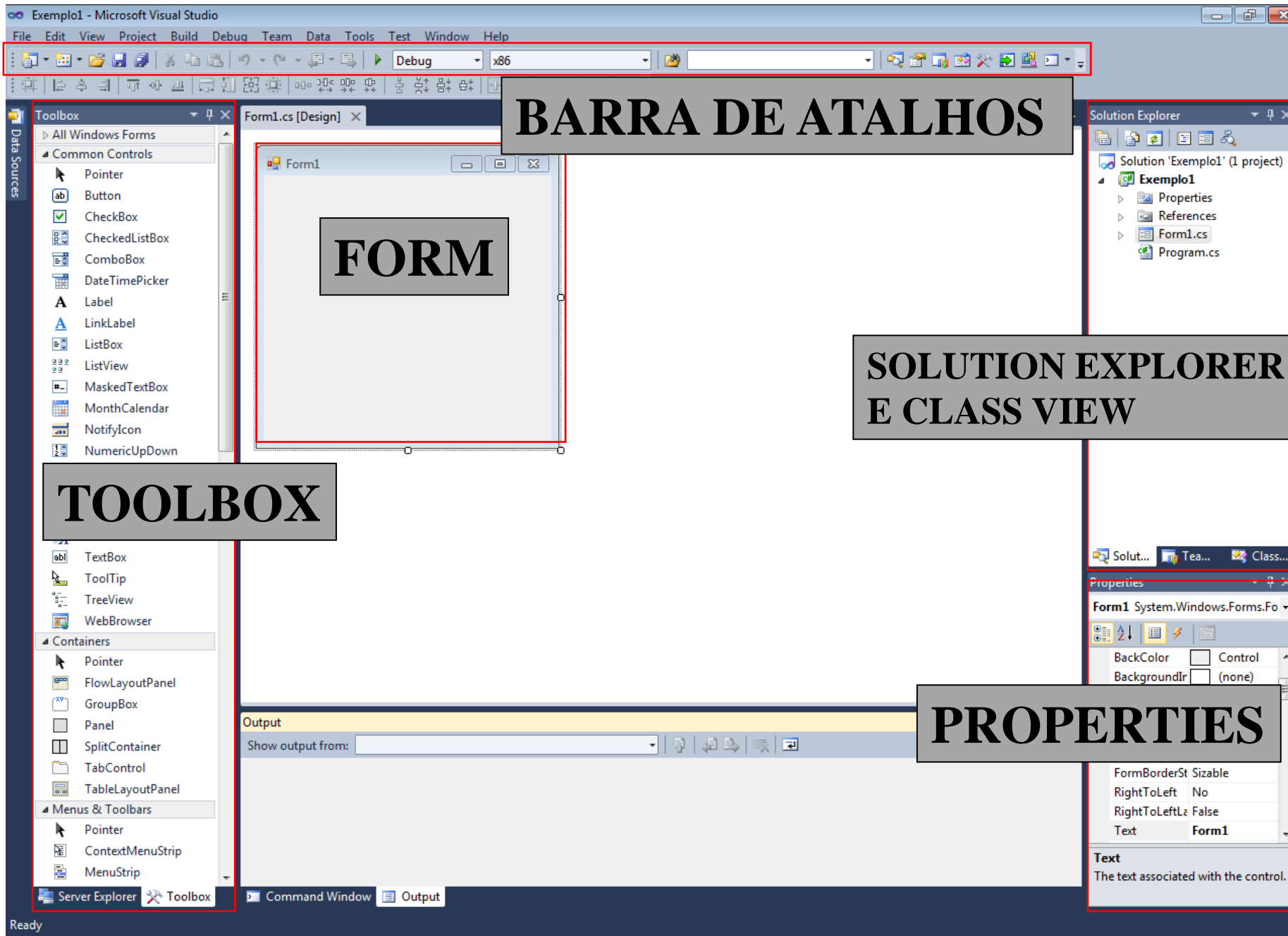


O código que define o formulário e seus objetos está aqui.



O Visual Studio cria estes três arquivos automaticamente.





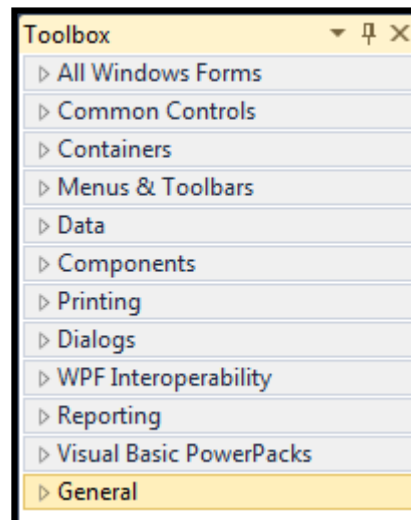
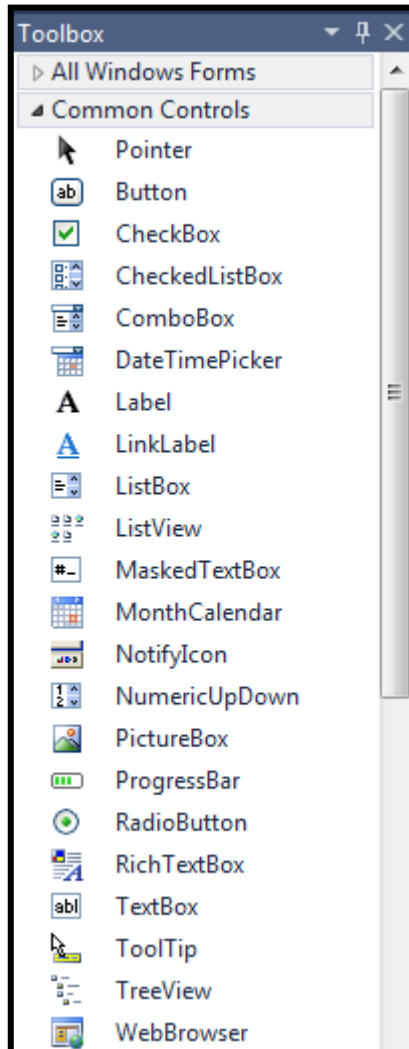
Conhecendo as janelas do VS

TOOLBOX

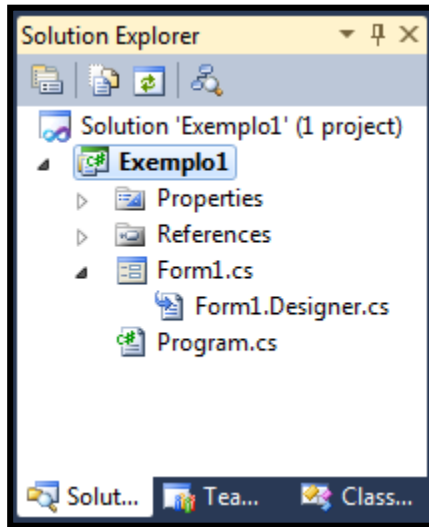
Esta caixa de diálogo é muito importante pois é nela que iremos escolher e pegar os objetos que usaremos no nosso programa.

Ela dividida por grupos de objetos que são separados por tipo.

Neste primeiro momento usaremos somente os objetos do **COMMON CONTROLS**.



Conhecendo as janelas do VS

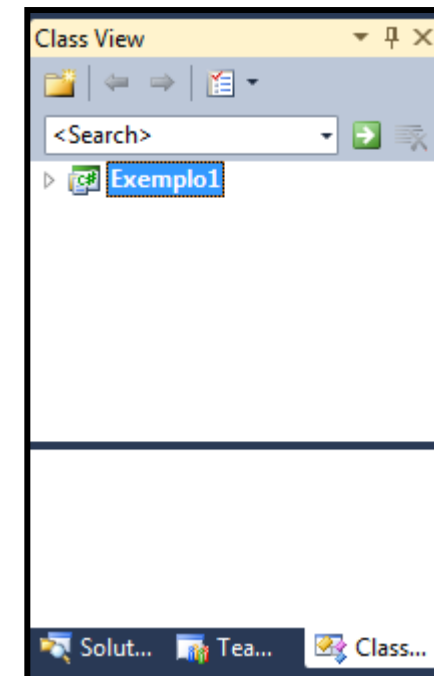


SOLUTION EXPLORER

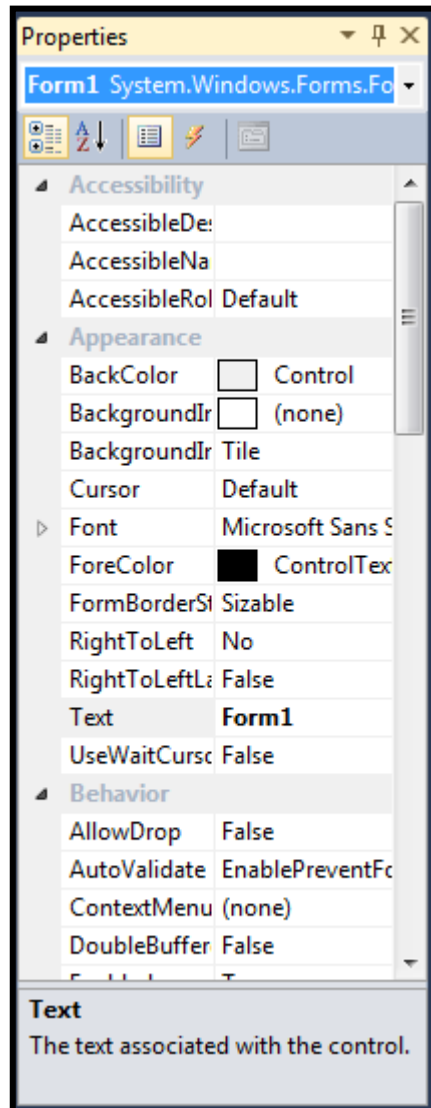
Esta caixa de diálogo é usada para gerenciar os arquivos que foram criados no nosso projeto.

CLASS VIEW

Esta caixa de diálogo é usada para gerenciar as classes criadas no nosso projeto.



Conhecendo as janelas do VS



PROPERTIES

Esta caixa de diálogo é outra muito importante pois é nela que configuraremos os objetos adicionados no nosso programa.

Ela é dividida em 2 partes: Properties e Events, para selecioná-los basta clicar em:



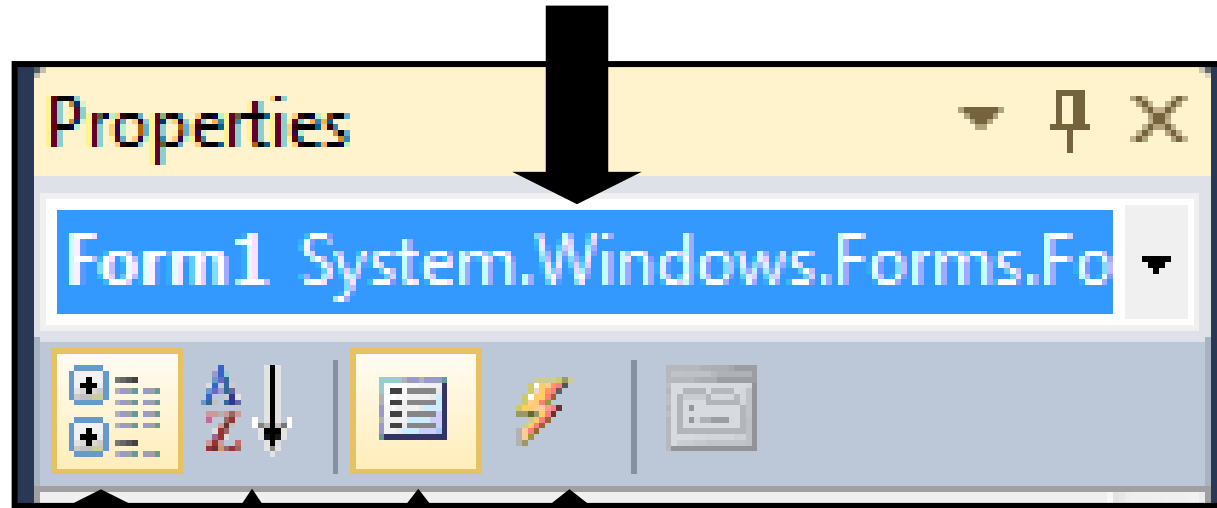
Para exibir as propriedades do objeto selecionado



Para exibir os eventos do objeto selecionado

Conhecendo as janelas do VS

Este campo informa o objeto selecionado



Exibir os itens
separados por tipo

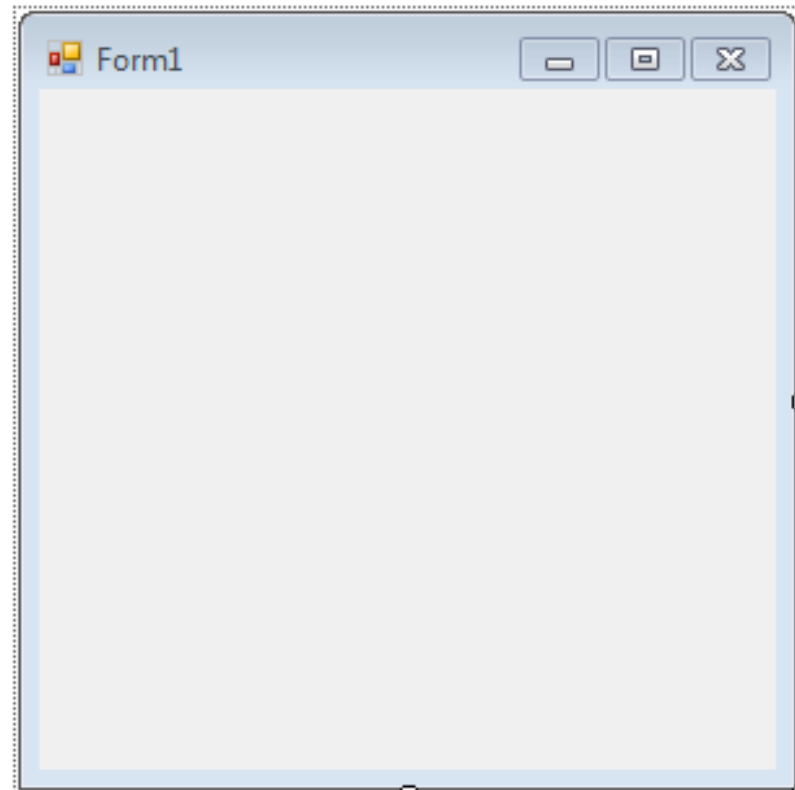
Exibir os itens em
ordem alfabética

Propriedades

Eventos

Conhecendo as janelas do VS

Esta é a tela do nosso programa, conhecida como **FORM** ou **FORMULÁRIO**



Conhecendo as janelas do VS



New Project



Add New Item



Open File



Save File



Start Debbing (F5)



Solution Explorer



Properties Window



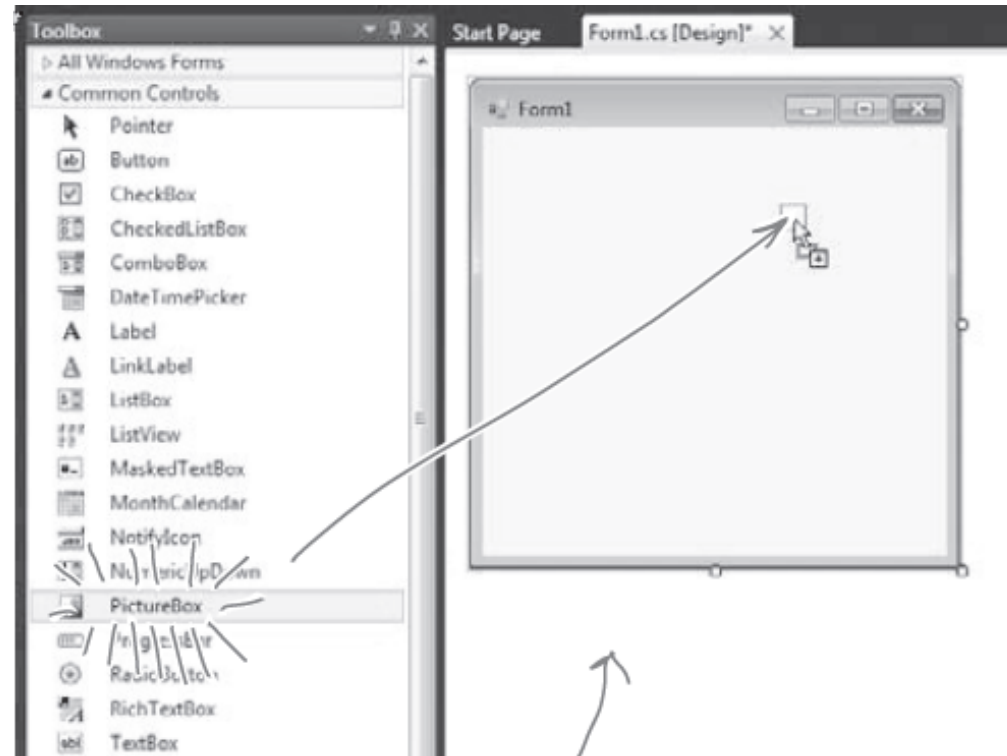
ToolBox Window

Desenvolvendo a interface com o usuário

Adicionar controles e arrumar a interface de usuário é tão fácil quanto arrastar e soltar no IDE do Visual Studio. Vamos acrescentar um logo ao formulário:

- 1. Utilize o controle PictureBox para acrescentar uma figura.**
Clique no controle PictureBox na Caixa de Ferramentas e arraste-o para o seu formulário. Nos bastidores, o IDE adicionou código em `Form1.Designer.cs` para um novo controle de imagens.

Desenvolvendo a interface com o usuário

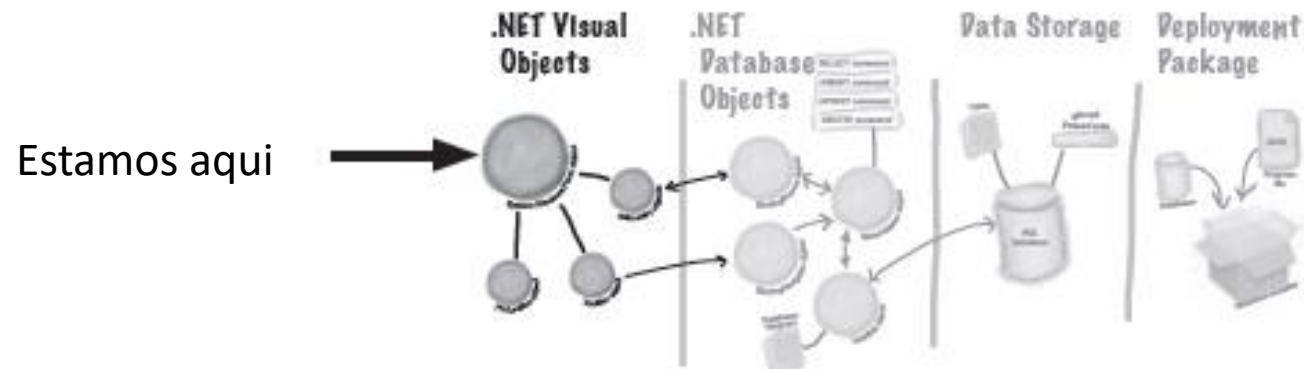


Toda vez que você alterar uma propriedade de controle no formulário, o código em `Form1.Designer.cs` também será mudado pelo IDE



Form1.Designer.cs

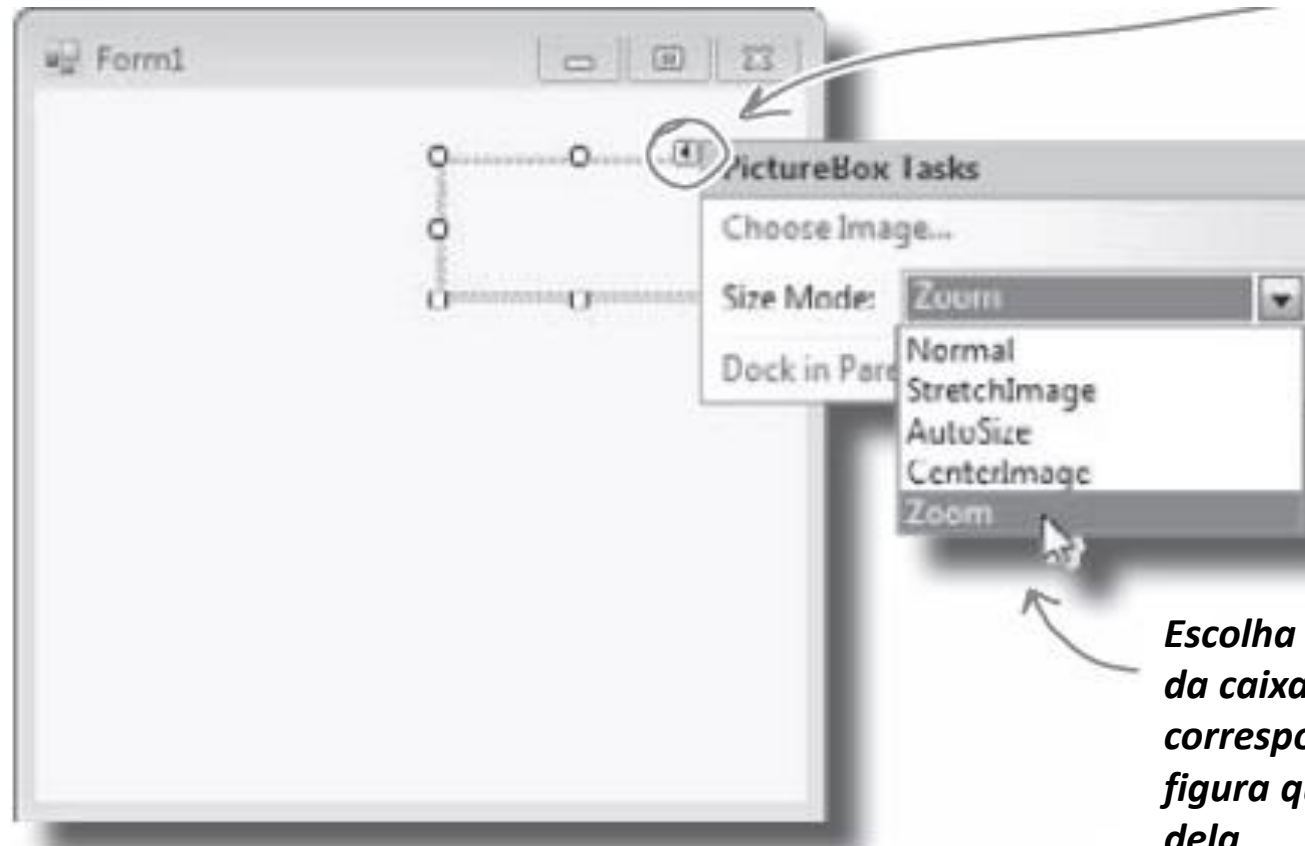
Desenvolvendo a interface com o usuário



2. Coloque a PictureBox em modo Zoom.

Todos os controles em seu formulário possuem propriedades ajustáveis. Clique na flechinha preta para acessá-las. Altere a propriedade Size da PictureBox para "Zoom" para ver como isto funciona:

Desenvolvendo a interface com o usuário



Clique nessa flechinha preta para acessar uma propriedade de um controle

Escolha Zoom para que a borda da caixa de imagem mude para corresponder ao tamanho da figura que você colocou dentro dela

Desenvolvendo a interface com o usuário

3. Adicionar o Logo da Empresa Papel Vila Objeto

Salve o logo no seu disco rígido. Então clique na seta de propriedades da PictureBox e selecione Choose Image. Click em Import ..., encontre seu logo e está tudo pronto:



*Aqui está o logo da OPC –
Empresa de Papel Vila Objeto. A
PictureBox usa o zoom para ficar
do tamanho certo.*

Visual Studio, nos bastidores

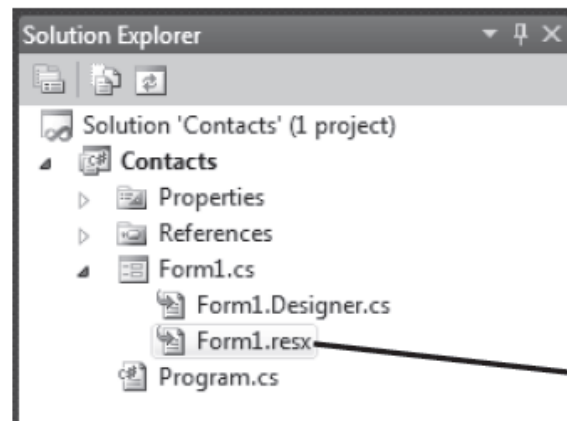
Toda vez que fazemos algo no IDE do Visual Studio, ele está **escrevendo código automaticamente**. Quando criarmos o logo e mandamos o Visual Studio usar a imagem selecionada, ele criou um recurso e associou-o com seu aplicativo. Um **recurso** é qualquer arquivo gráfico, de áudio, ícone ou outro tipo de arquivo de dados embutido no nosso aplicativo. O arquivo gráfico fica integrado ao programa, para eu, então quando ele for instalado em outro computador, o gráfico seja instalado junto com ele e a PictureBox possa usá-lo.



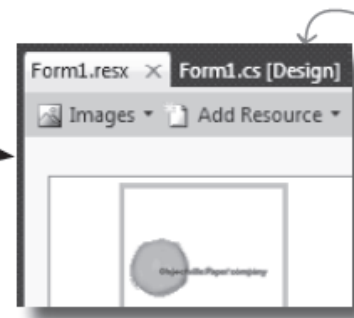
Esta imagem é agora um recurso do aplicativo Exemplo1

Visual Studio, nos bastidores

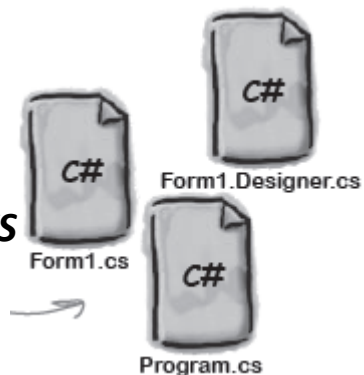
Quando arrastamos o controle PictureBox para o nosso formulário, o IDE automaticamente criou um arquivo de recurso chamado Form1.resx para armazená-lo e mantê-lo em seu projeto. Dê um duplo clique neste arquivo e você verá a imagem importada.



Se clicarmos em Form1.resx no Solution Explorer, veremos a logomarca importada. Este arquivo é conectado à caixa de imagem; e o IDE adicionou código para fazer a conexão.



Estes são os arquivos que o VS criou anteriormente



Complete o código gerado automaticamente

O IDE cria muito código, mas precisamos ter acesso a ele e acrescentar-lhe coisas.

Certifique-se de que o formulário aparece no IDE e clique duas vezes no controle de caixa de imagem. Um código semelhante ao seguinte deve aparecer:

```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
    }
    private void pictureBox1_Click(object sender, EventArgs e)
    {
        MessageBox.Show("Contact List 1.0.\nWritten by: Your Name", "About");
    }
}
```

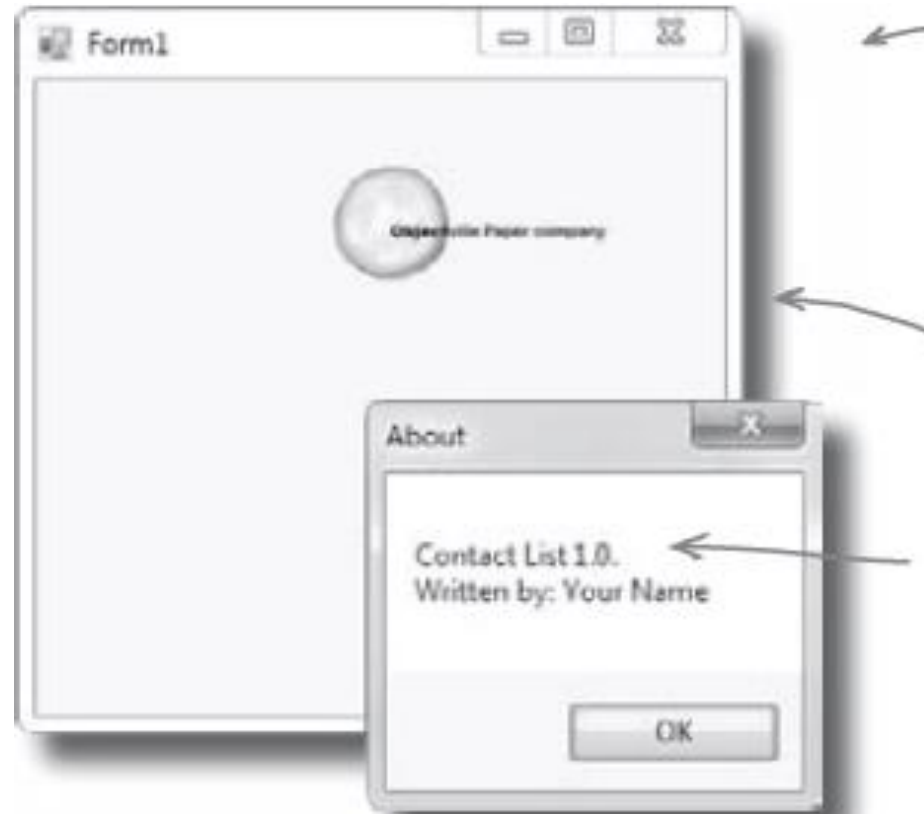
Quando você clicou duas vezes no PictureBox, o IDE criou este método. Ele será executado sempre que um usuário clicar no logo com o aplicativo em execução.

Este nome de método dá uma boa idéia sobre quando ele executa: quando alguém clica no controle PictureBox

Digite esta linha de código. Uma caixa de mensagem aparecerá com o texto que você digitou. A caixa terá o título About (Sobre).

Testando nosso programa

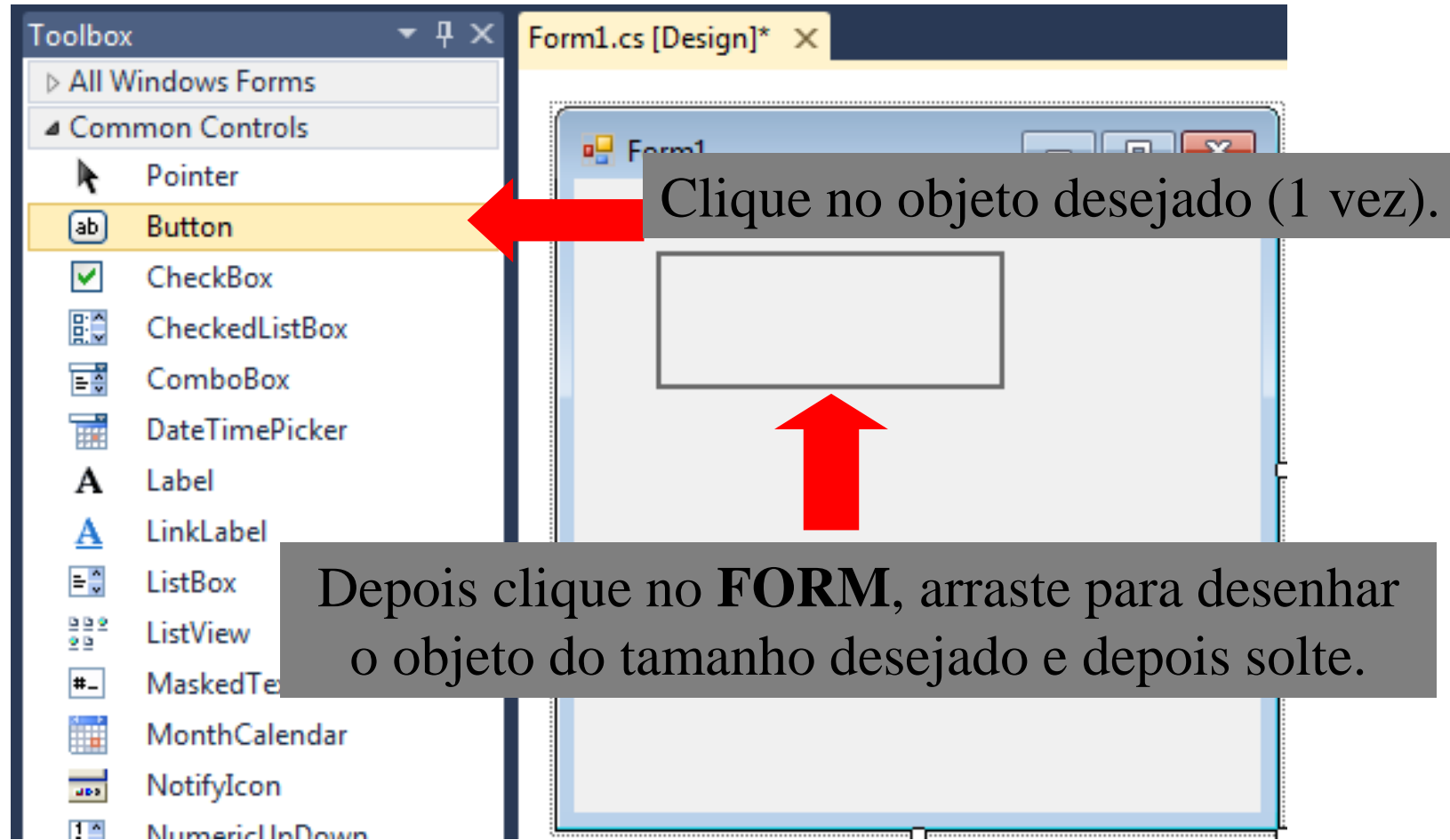
Pressione F5 para executar o programa.



Estes três botões funcionam e não precisamos escrever nenhum código para eles.

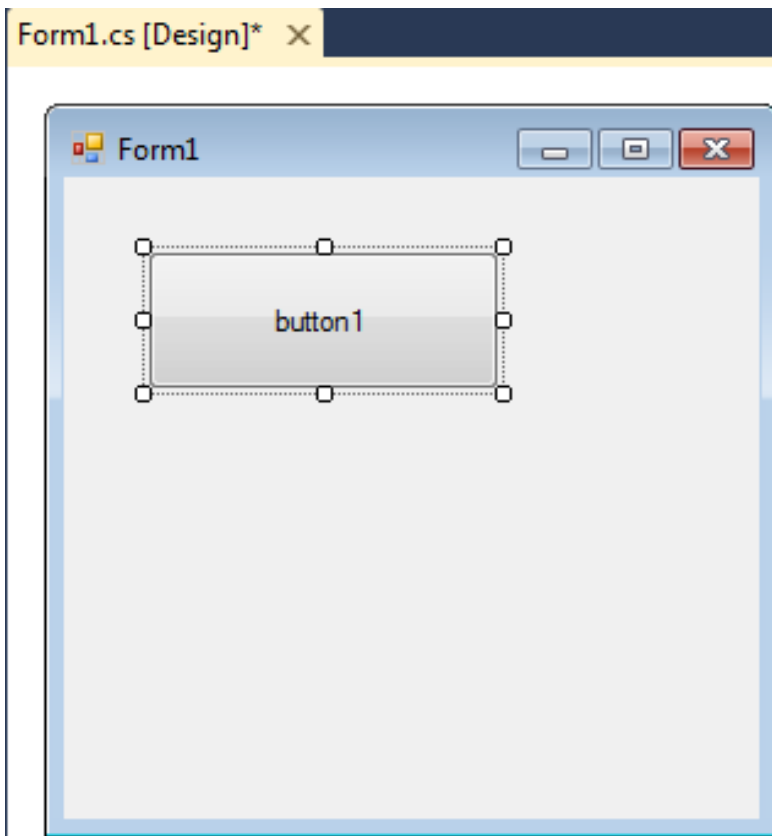
Clicar no logo da OPC faz a caixa Sobre, que acabamos de codificar, aparecer.

Adicionando Controles ao Form



Adicionando Controles ao Form

O * significa que o seu programa ainda não foi salvo



Todos objetos adicionados ao FORM deverão ter alteradas as propriedades NAME e TEXT.

A propriedade NAME serve para identificar o objeto a nível do programador (VARIÁVEL).

A propriedade TEXT serve para identificar o objeto a nível do usuário (RÓTULO).

Adicionando Controles ao Form

Text	button1	▼
------	---------	---

Nesta propriedade podemos usar acentos, caracteres especiais e espaço em branco.

Altere a propriedade para o texto: **Clique Aqui!!!**

Text	Clique Aqui!!!	▼
------	----------------	---

(Name)	button1
--------	---------

Nesta propriedade **NÃO** podemos usar acentos, caracteres especiais e espaço em branco.

Altere a propriedade para o texto: **btnClique**

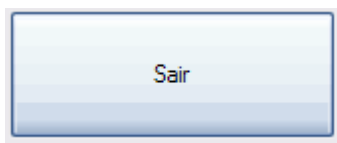
(Name)	btnClique
--------	-----------

Adicionando Controles ao Form

Regra de Nomenclatura

Remover as vogais e adicionar a funcionalidade do objeto.

Exemplos:



Button1 => btnSair

Digite seu Nome:

Label1 => lblNome

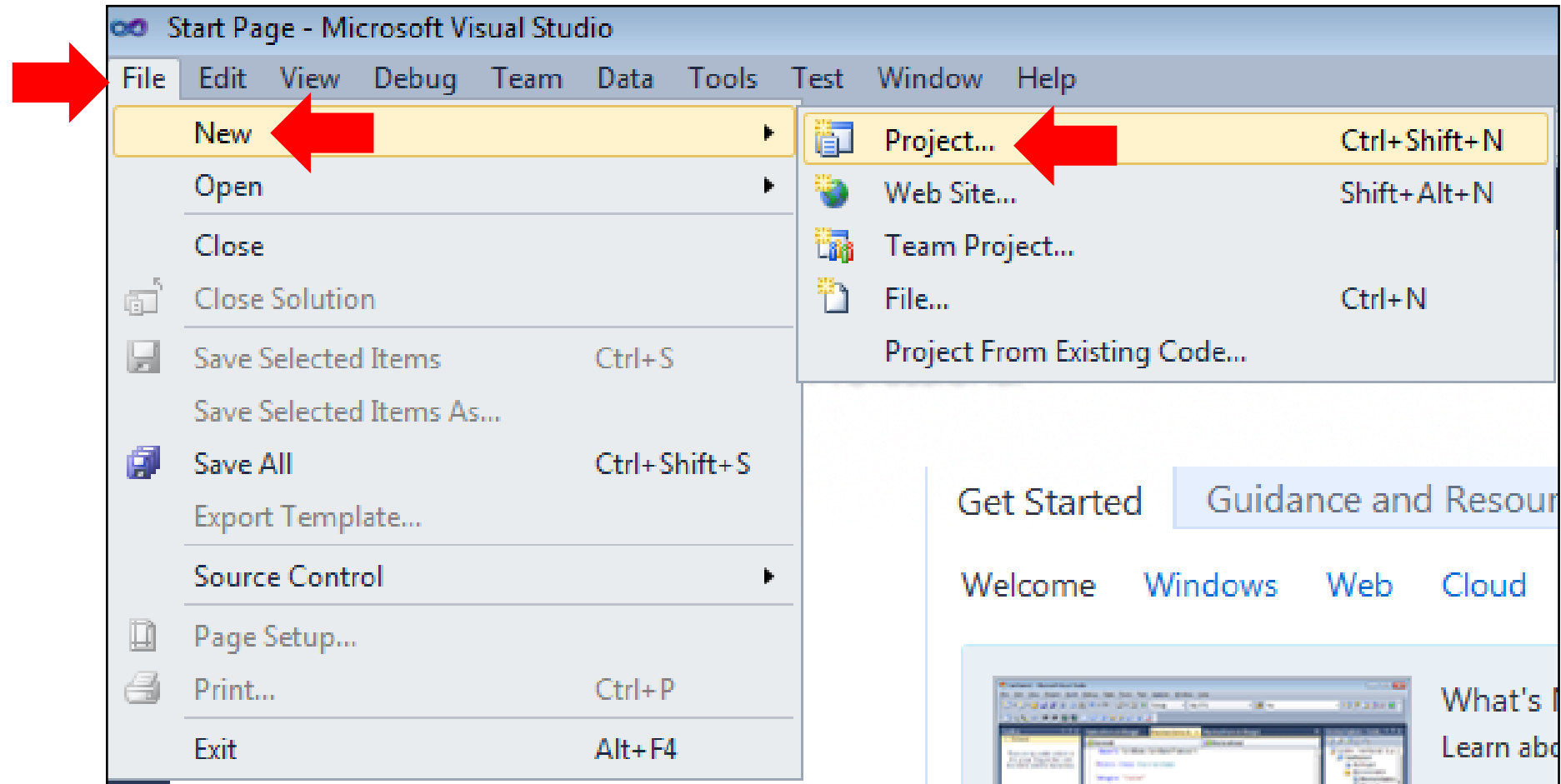
Digite seu endereço:

TextBox1 => txtEndereco

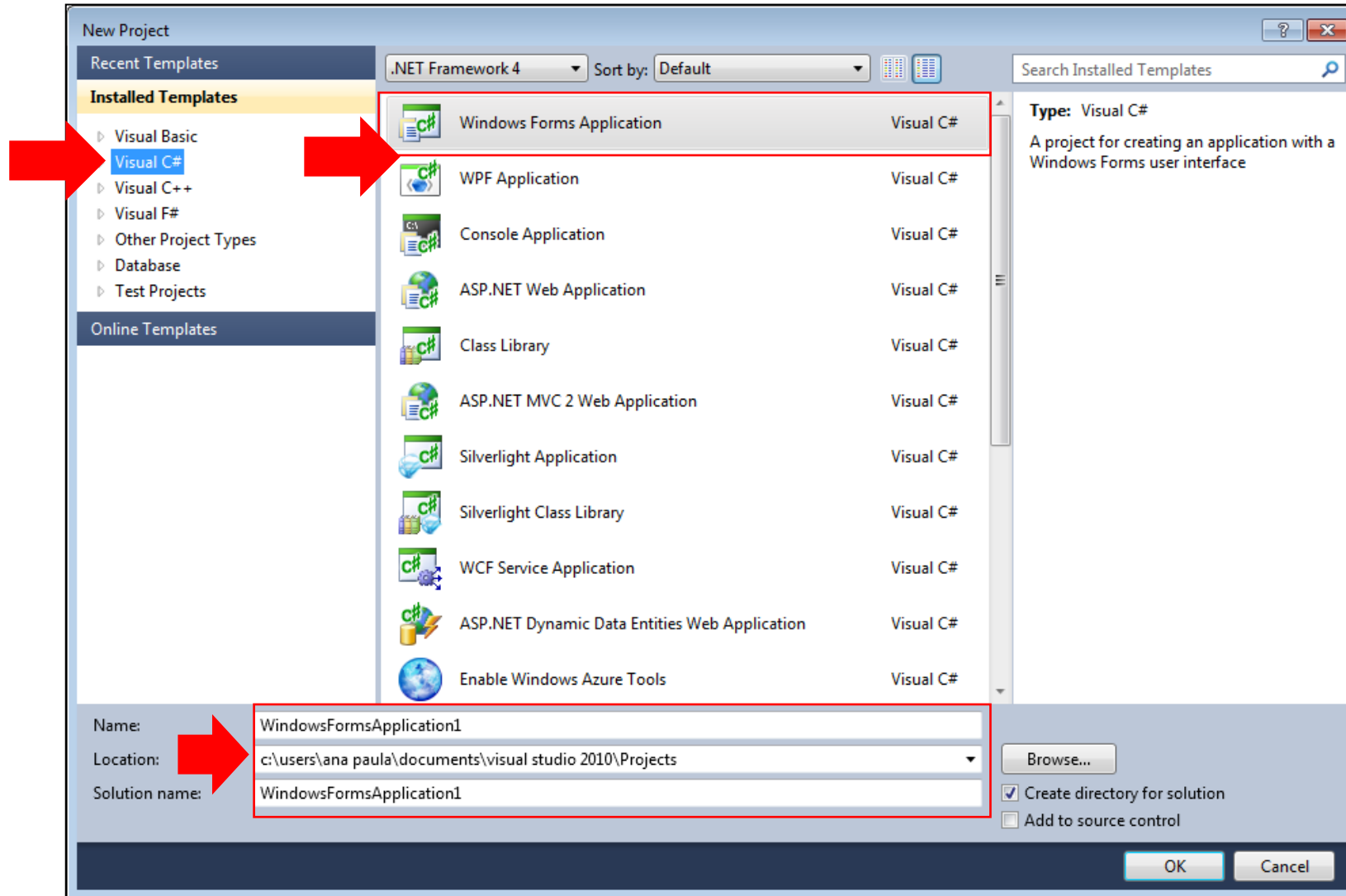
EXEMPLO 1

**FAÇA UM PROGRAMA QUE ESCREVA
A FRASE HELLO WORLD NA TELA
QUANDO UM BOTÃO FOR CLICADO**

1. Passo: Criar um Projeto



1. Passo: Criar um Projeto

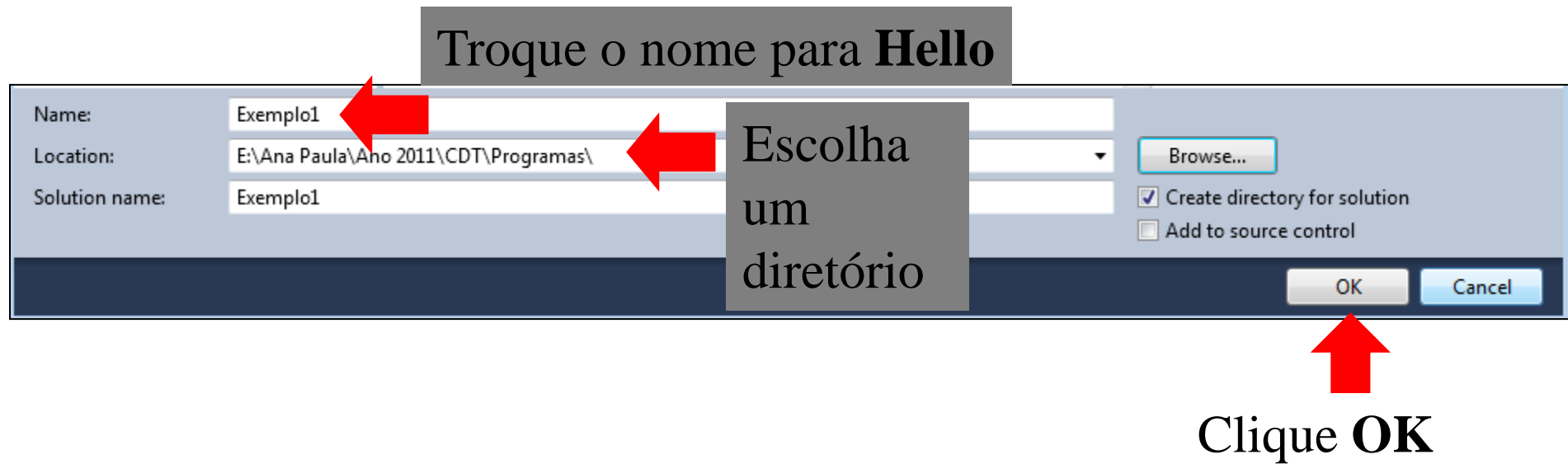


1. Passo: Criar um Projeto

Troque o nome para Hello

Escolha um diretório

Clique OK



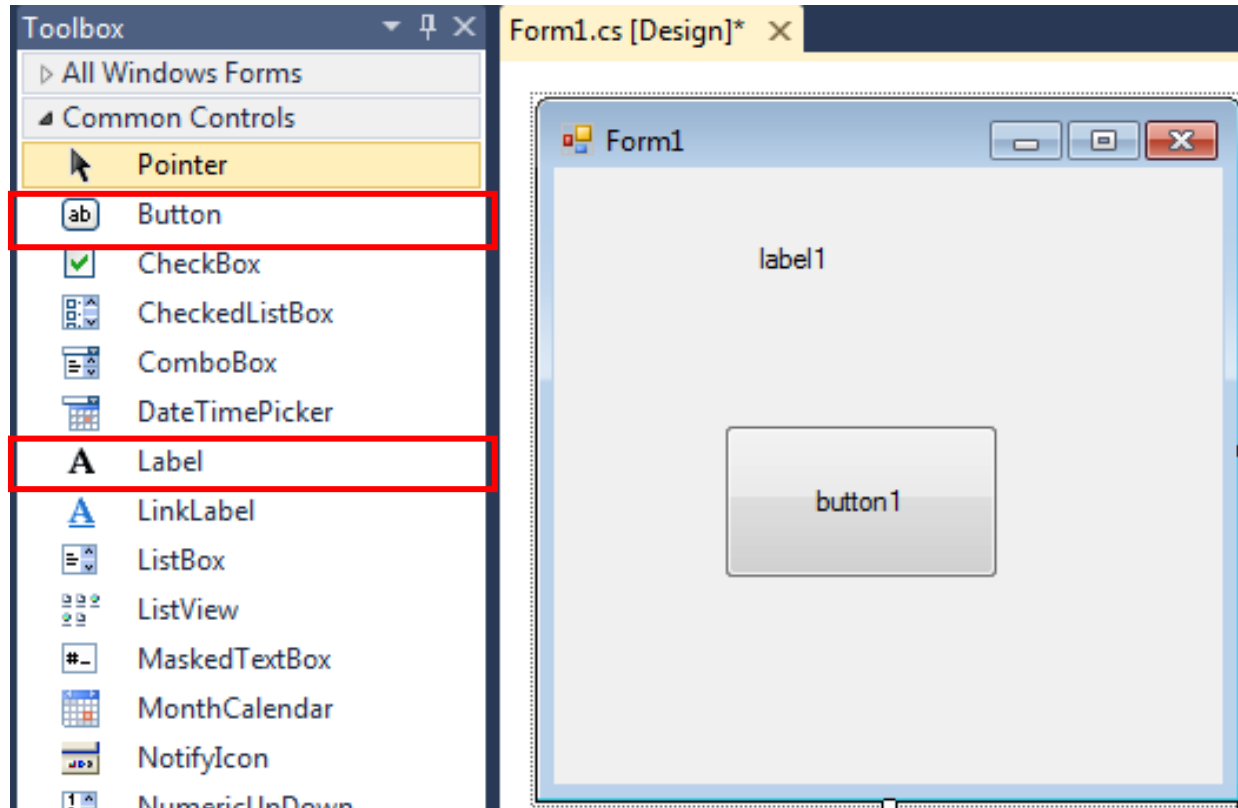
The image shows a 'Create New Project' dialog box from Visual Studio. It has three input fields: 'Name:' with 'Exemplo1', 'Location:' with 'E:\Ana Paula\Aho 2011\CDT\Programas\' and a 'Browse...' button, and 'Solution name:' with 'Exemplo1'. There are two checkboxes: 'Create directory for solution' (checked) and 'Add to source control' (unchecked). At the bottom are 'OK' and 'Cancel' buttons. Red arrows point from text boxes to these elements: one from 'Troque o nome para Hello' to the Name field, one from 'Escolha um diretório' to the Location field, and one from 'Clique OK' to the OK button.

Name:	Exemplo1
Location:	E:\Ana Paula\Aho 2011\CDT\Programas\
Solution name:	Exemplo1

☒ Create directory for solution
☐ Add to source control

OK Cancel

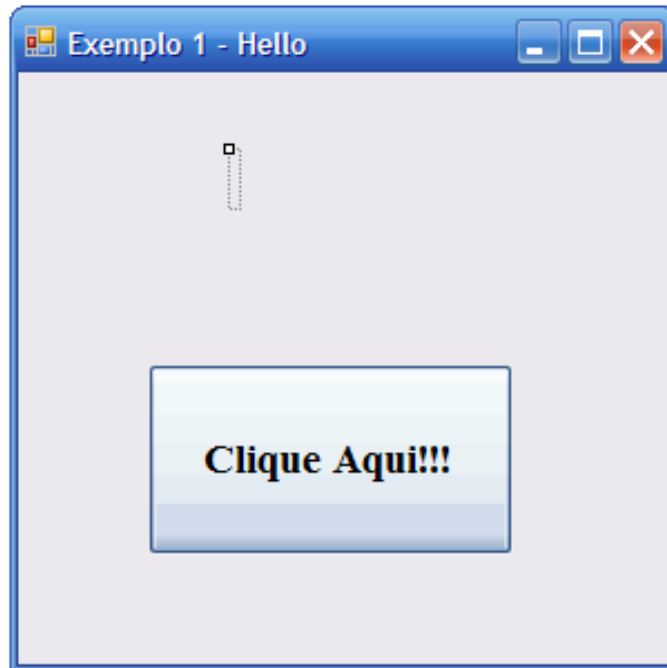
2. Passo: Construir a Tela



Adicione
1 Label e
1 Button

2. Passo: Construir a Tela

Para que o FORM fique assim deve-se alterar algumas propriedades dos três objetos: FORM, BUTTON e LABEL.



Tome cuidado para não alterar a propriedade do objeto errado. **CLIQUE 1 VEZ NO OBJETO, PARA SELECIONÁ-LO, E DEPOIS ALTERE AS PROPRIEDADES.**

LABEL

Text	
(Name)	lblMensagem

BUTTON

Text	Clique Aqui!!!
(Name)	btnCliqueAqui

FORM

Text	Exemplo 1 - Hello
(Name)	frmTelaPrincipal

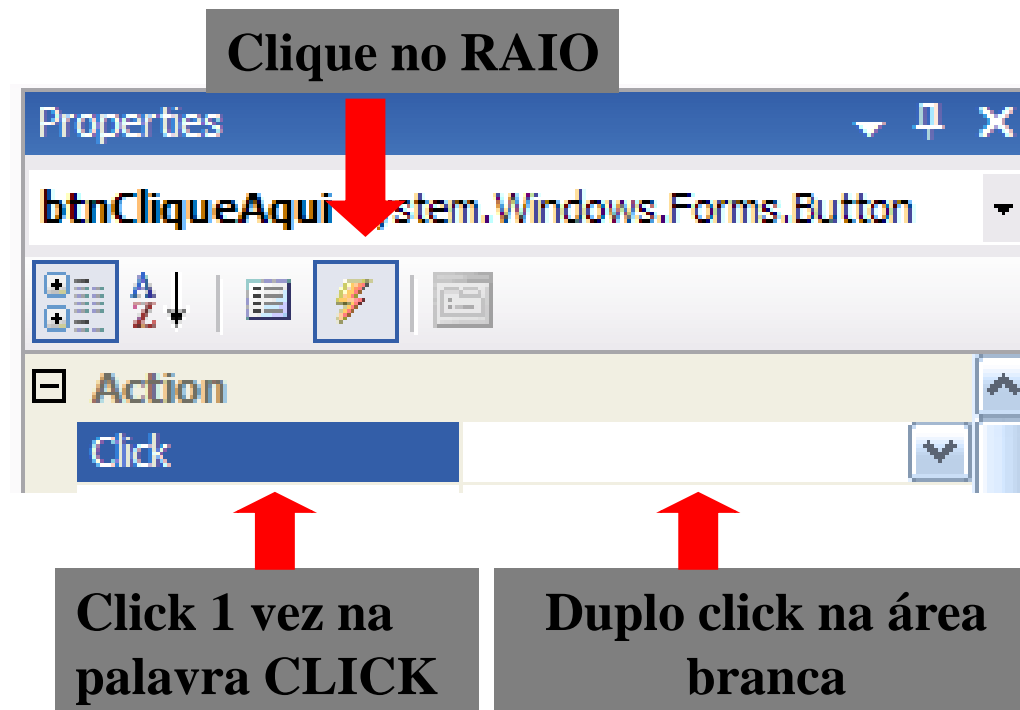
2. Passo: Construir a Tela

As propriedades NAME e TEXT sempre serão alteradas.

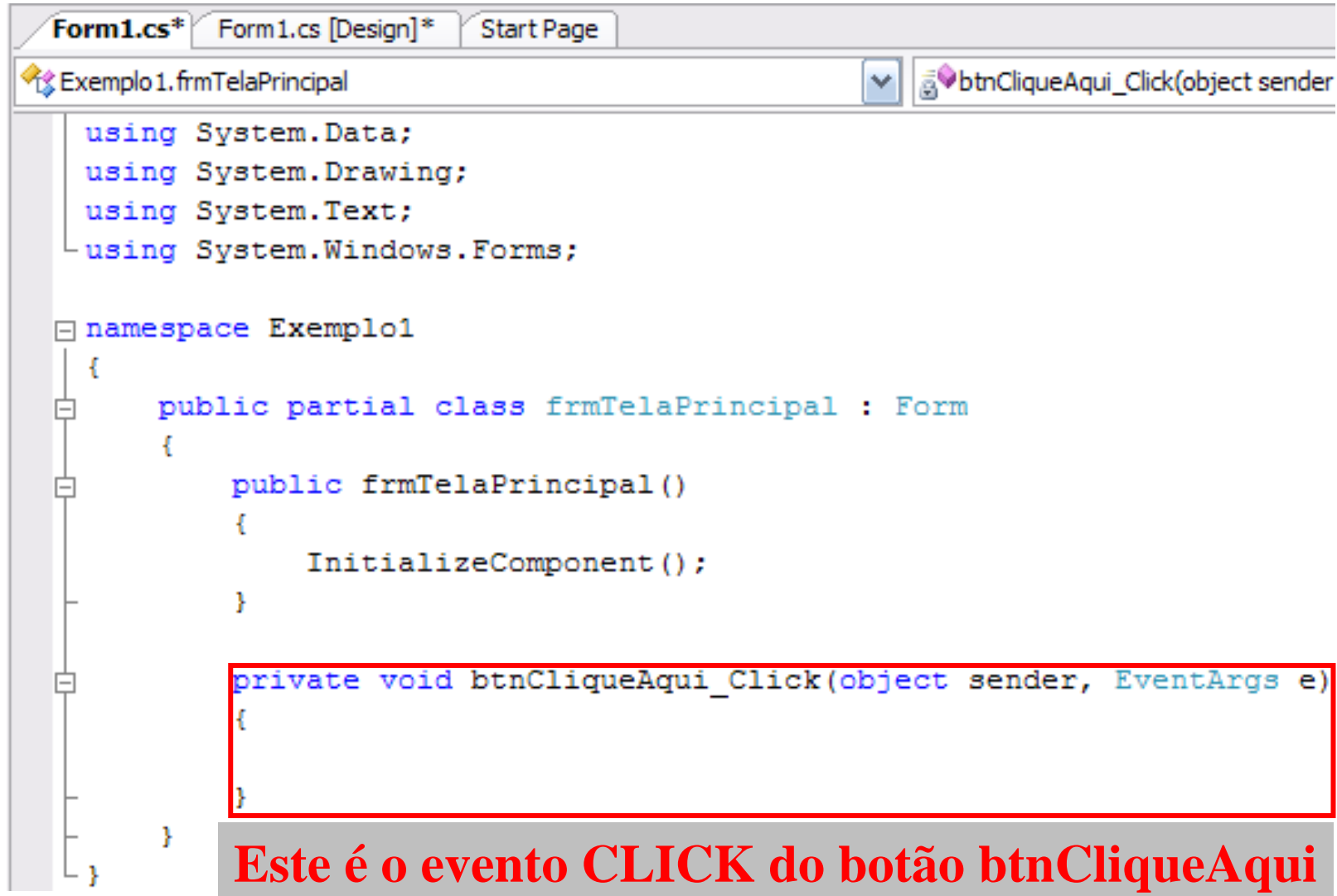
Existem outras propriedades que também são interessantes, teste as seguintes propriedades:

- **BACKCOLOR** – usada para alterar a cor de fundo
- **FONT** – usada para alterar a fonte (estilo, tamanho, entre outros)
- **FORECOLOR** – usada para alterar a cor da fonte

3. Passo: Codificar o Programa



3. Passo: Codificar o Programa



```
Form1.cs*  Form1.cs [Design]*  Start Page
Exemplo1.frmTelaPrincipal  btnCliqueAqui_Click(object sender

using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace Exemplo1
{
    public partial class frmTelaPrincipal : Form
    {
        public frmTelaPrincipal()
        {
            InitializeComponent();
        }

        private void btnCliqueAqui_Click(object sender, EventArgs e)
        {
        }
    }
}
```

Este é o evento CLICK do botão btnCliqueAqui

3. Passo: Codificar o Programa

```
private void btnCliqueAqui_Click(object sender, EventArgs e)
{
    lblMensagem.Text = "HELLO WORLD";
}
```

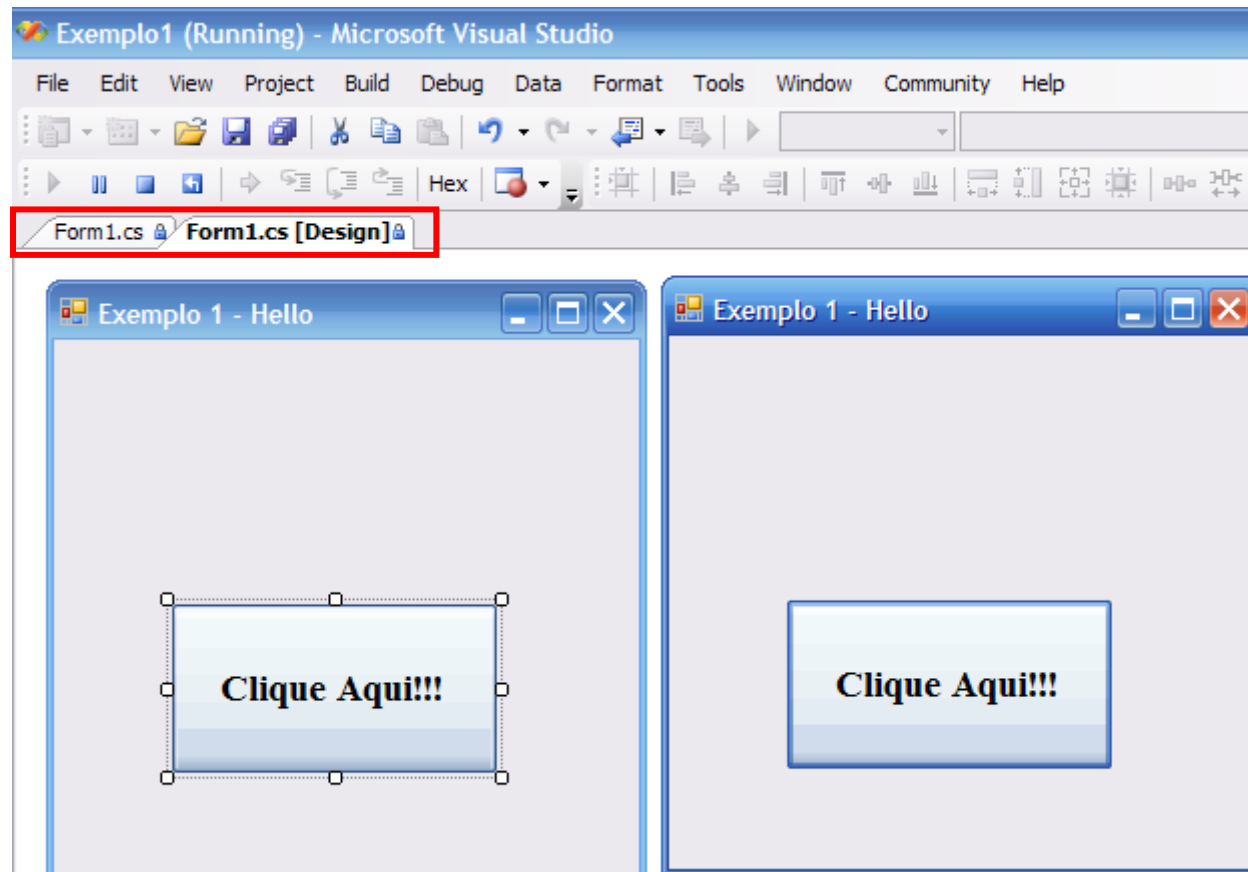
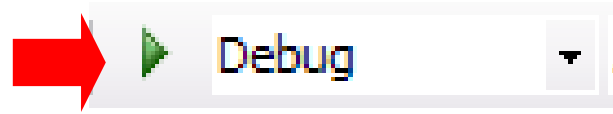


Escreva o código acima.

DICA: Use as teclas Ctrl+Barra de Espaço para facilitar a digitação

3. Passo: Codificar o Programa

Clique na seta verde ou pressione F5



Para diferenciar as duas telas verifique se existe o cadeado azul na frente das ABAS, caso exista feche a tela do programa que está executando antes de alterar qualquer coisa.

3. Passo: Codificar o Programa



CONCLUSÃO

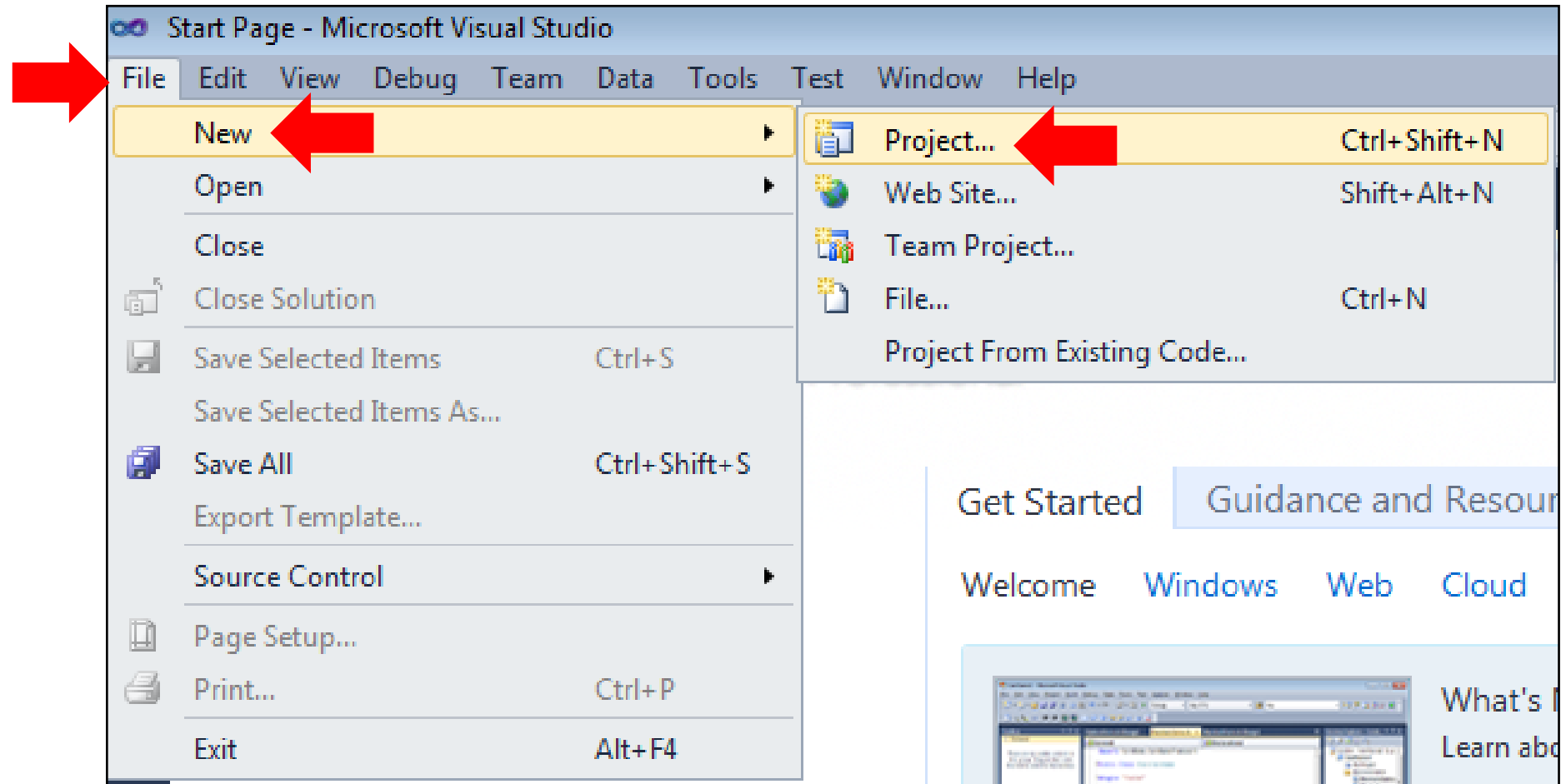
Neste exemplo aprendemos a usar dois objetos (LABEL e BUTTON) e um evento (CLICK).

- **LABEL** – usado para escrever mensagens na tela
- **BUTTON** – usado para executar a funcionalidade do programa. Para isto acionamos o **evento CLICK**, que executa o código através do click do mouse no objeto.

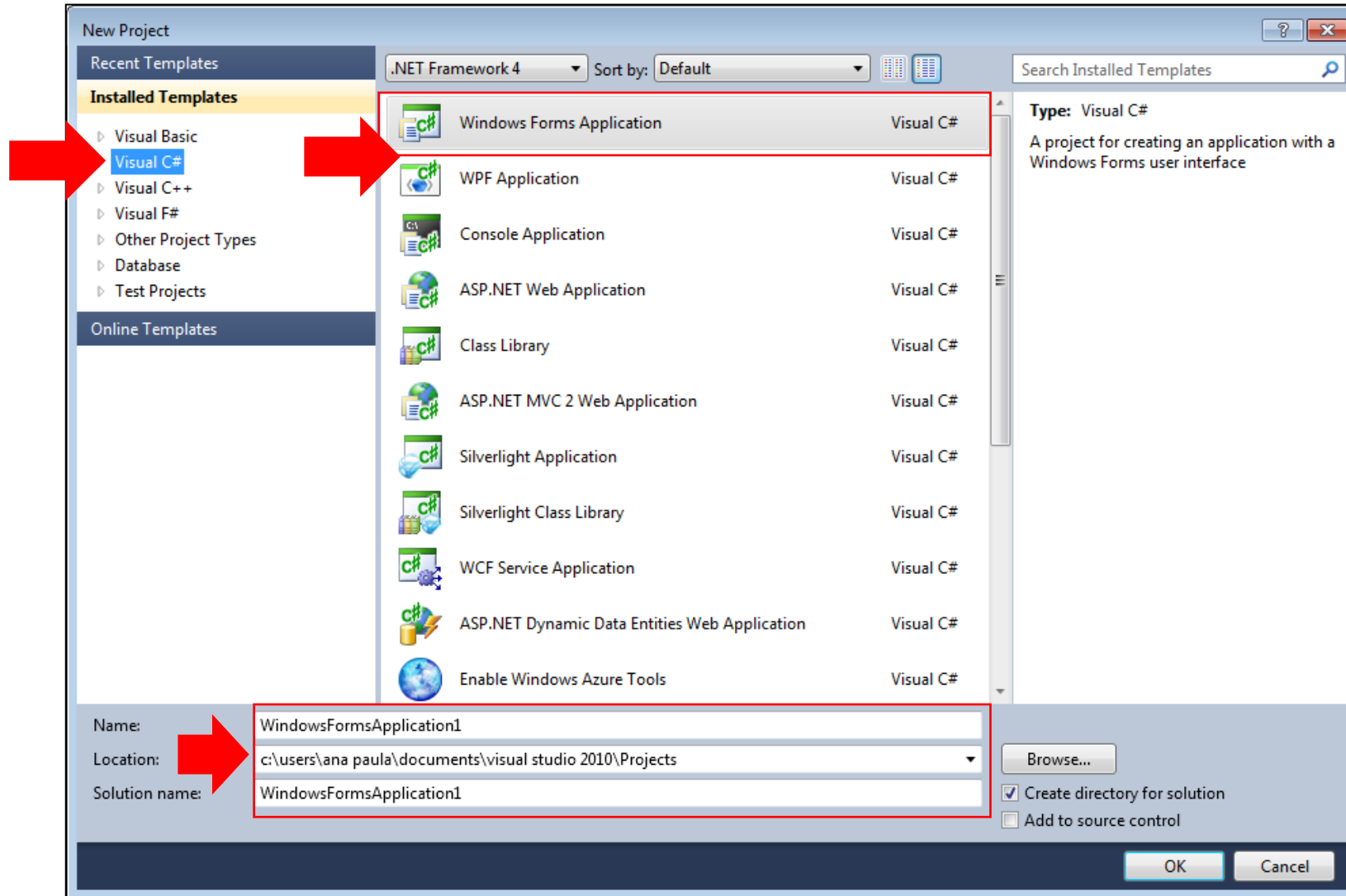
EXEMPLO 2

FAÇA UM PROGRAMA QUE ESCREVA
A FRASE BOM DIA, seu nome NA TELA
QUANDO UM BOTÃO FOR CLICADO

1. Passo: Criar um Projeto



1. Passo: Criar um Projeto



1. Passo: Criar um Projeto

Troque o nome para **Bom Dia**

Escolha um diretório

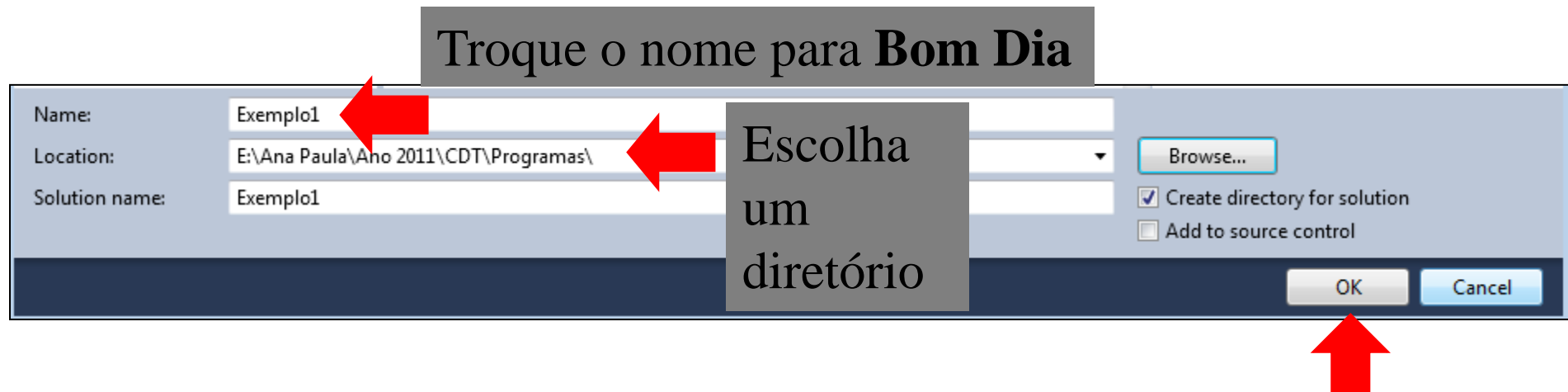
Name: Exemplo1

Location: E:\Ana Paula\Aho 2011\CDT\Programas\ Browse...

Solution name: Exemplo1

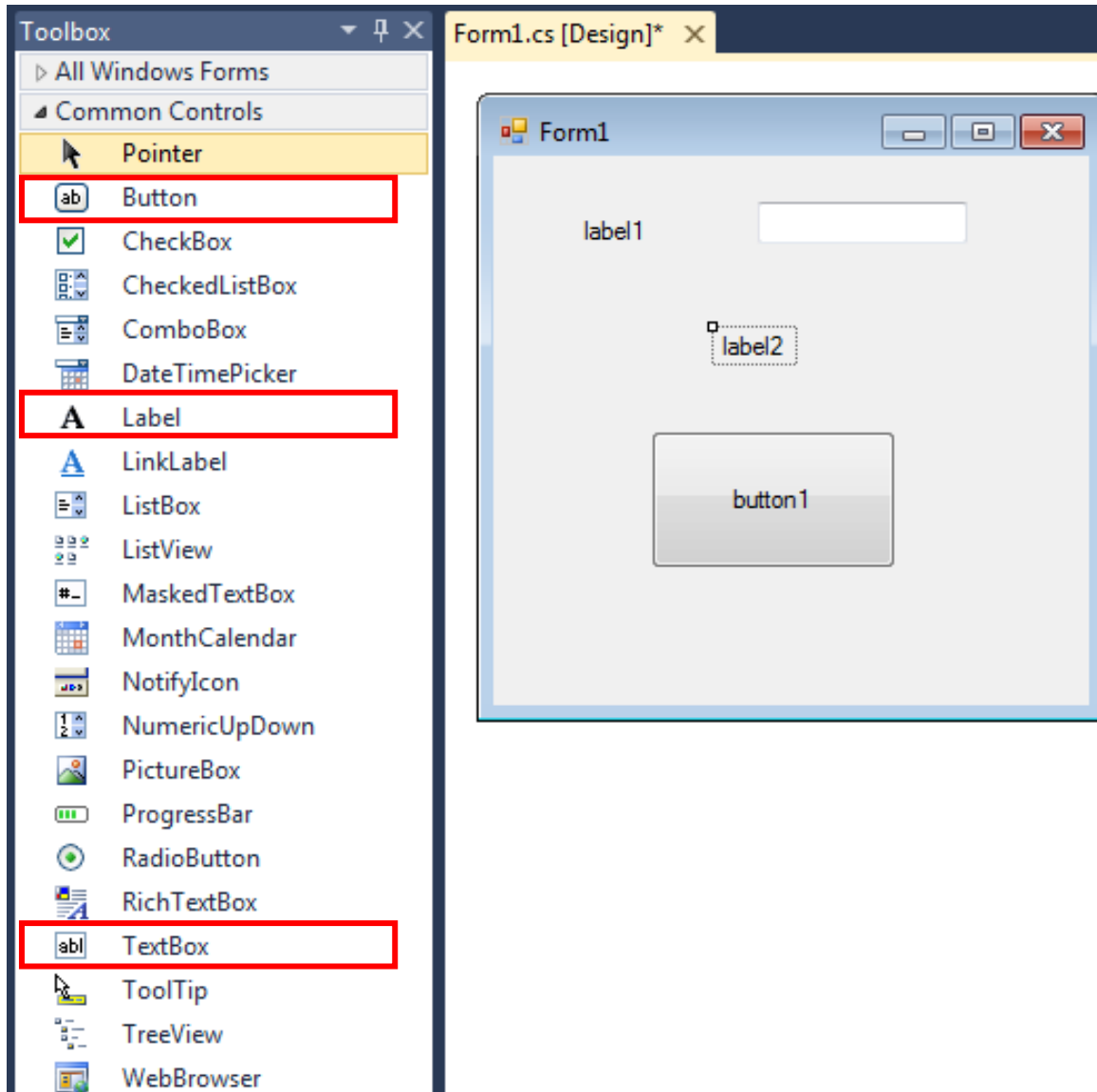
☒ Create directory for solution
☐ Add to source control

OK Cancel

A screenshot of a project creation dialog box. The dialog has a light blue header and a white body. It contains three input fields: 'Name' with 'Exemplo1', 'Location' with 'E:\Ana Paula\Aho 2011\CDT\Programas\' and a 'Browse...' button, and 'Solution name' with 'Exemplo1'. There are two checkboxes: 'Create directory for solution' (checked) and 'Add to source control' (unchecked). At the bottom are 'OK' and 'Cancel' buttons. Red arrows point from text annotations to the 'Name' field, the 'Location' field, and the 'OK' button. A grey box with the text 'Troque o nome para Bom Dia' is positioned above the 'Name' field. Another grey box with the text 'Escolha um diretório' is positioned to the right of the 'Location' field.

Clique **OK**

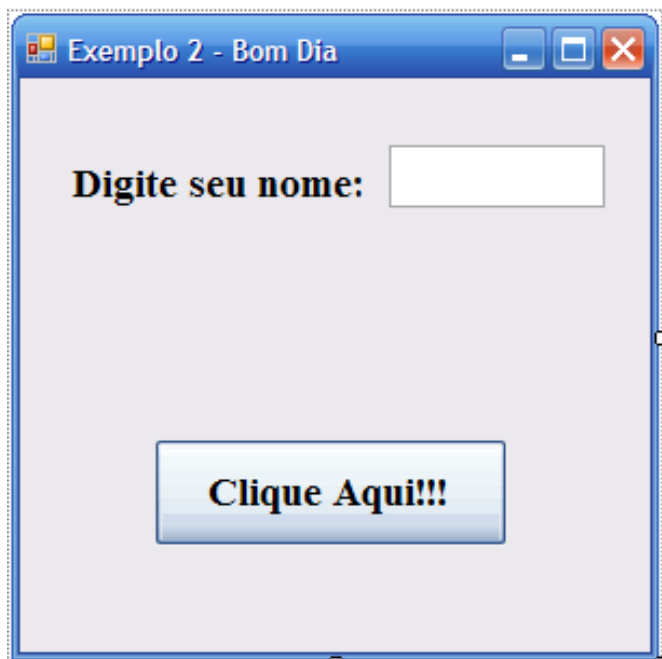
2. Passo: Construir a Tela



Adicione:
2 Labels
1 Button
1 TextBox

2. Passo: Construir a Tela

Para que o FORM fique assim deve-se alterar algumas propriedades dos três objetos: FORM, BUTTON, LABEL e TEXTBOX.



FORM1

Text: Exemplo 2 – Bom Dia

Name: frmTelaPrincipal

Tome cuidado para não alterar a propriedade do objeto errado. **CLIQUE 1 VEZ NO OBJETO, PARA SELECIONÁ-LO, E DEPOIS ALTERE AS PROPRIEDADES.**

LABEL1

Text: Digite seu nome:

Name: lblNome

LABEL2

Text:

Name: lblMensagem

BUTTON1

Text: Clique Aqui!!!

Name: btnCliqueAqui

TEXTBOX1

Text:

Name: txtNome

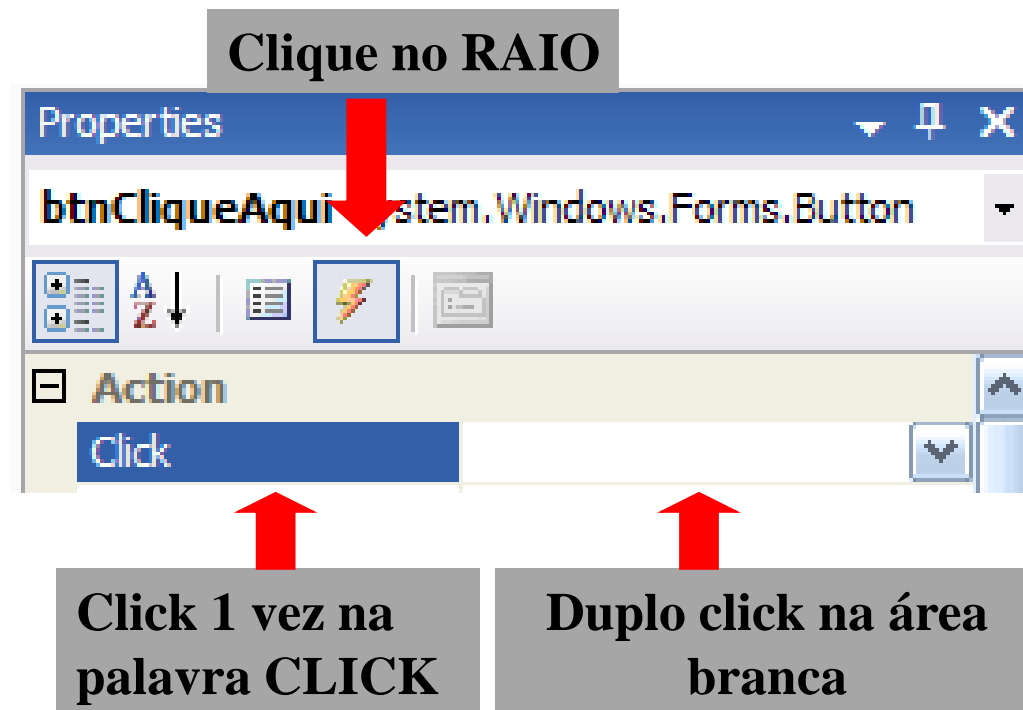
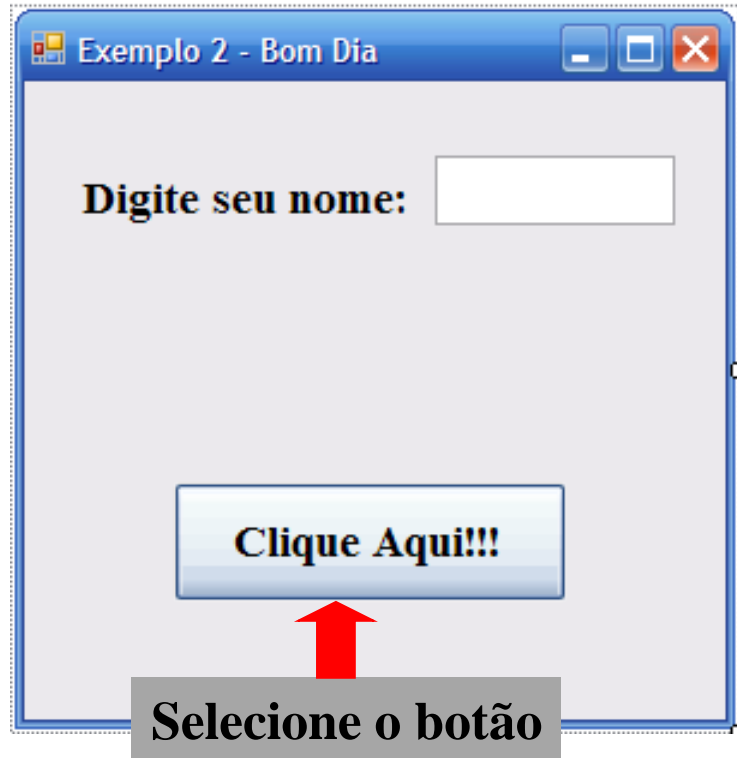
2. Passo: Construir a Tela

As propriedades NAME e TEXT sempre serão alteradas.

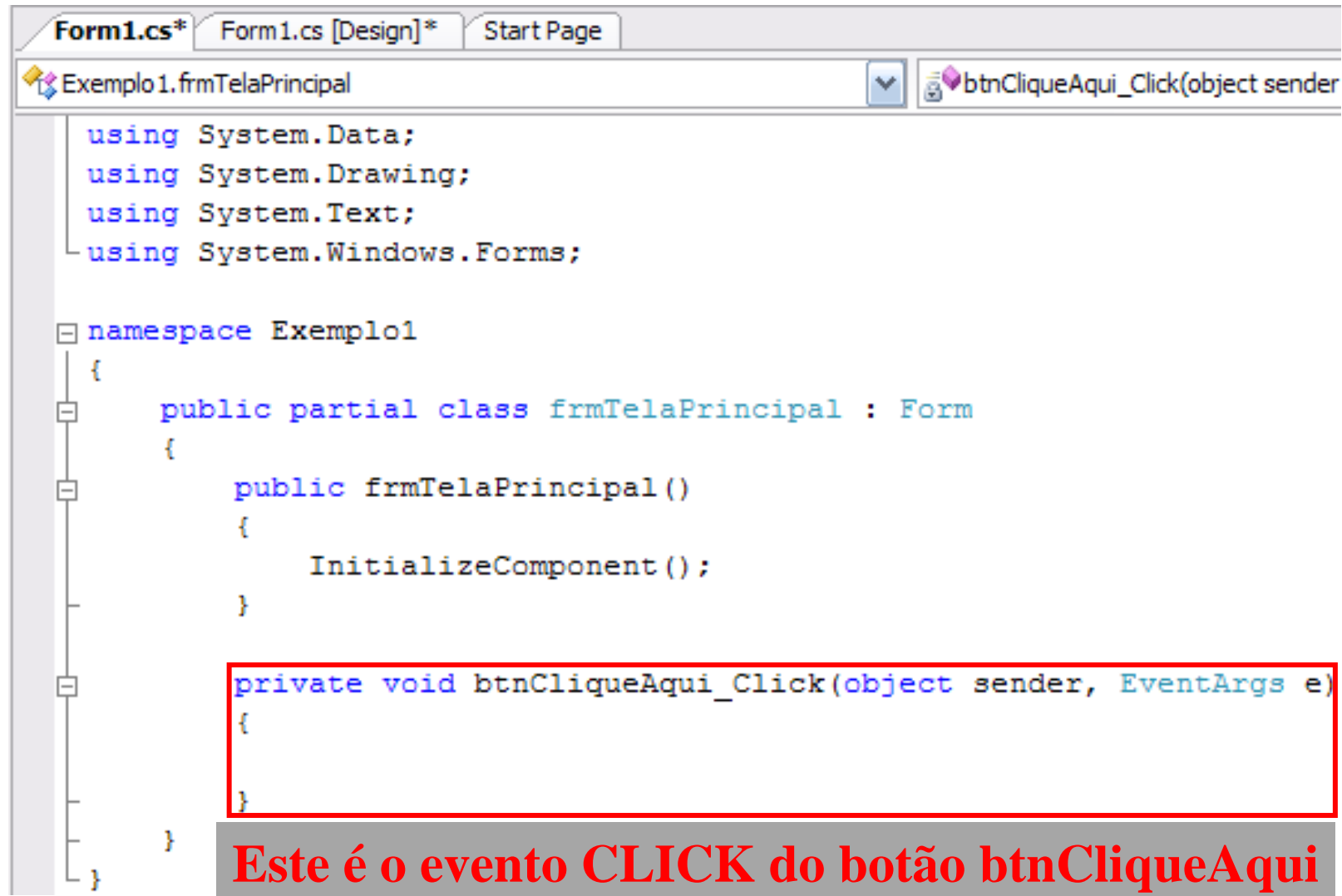
Existem outras propriedades que também são interessantes, teste as seguintes propriedades:

- **BACKCOLOR** – usada para alterar a cor de fundo
- **FONT** – usada para alterar a fonte (estilo, tamanho, entre outros)
- **FORECOLOR** – usada para alterar a cor da fonte

3. Passo: Codificar o Programa



3. Passo: Codificar o Programa



```
Form1.cs*  Form1.cs [Design]*  Start Page
Exemplo1.frmTelaPrincipal  btnCliqueAqui_Click(object sender, EventArgs e)

using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace Exemplo1
{
    public partial class frmTelaPrincipal : Form
    {
        public frmTelaPrincipal()
        {
            InitializeComponent();
        }

        private void btnCliqueAqui_Click(object sender, EventArgs e)
        {
        }
    }
}
```

Este é o evento CLICK do botão btnCliqueAqui

3. Passo: Codificar o Programa

```
private void btnCliqueAqui_Click(object sender, EventArgs e)
{
    lblMensagem.Text = "Bom dia, " + txtNome.Text;
}
```



Escreva o código acima.

DICA: Use as teclas Ctrl+Barra de Espaço para facilitar a digitação

3. Passo: Codificar o Programa

Agora teste este código:

```
private void btnCliqueAqui_Click(object sender, EventArgs e)
{
    String nome;
    nome = txtNome.Text;
    lblMensagem.Text = "Bom dia, " + nome;
}
```

Qual a diferença entre os códigos?

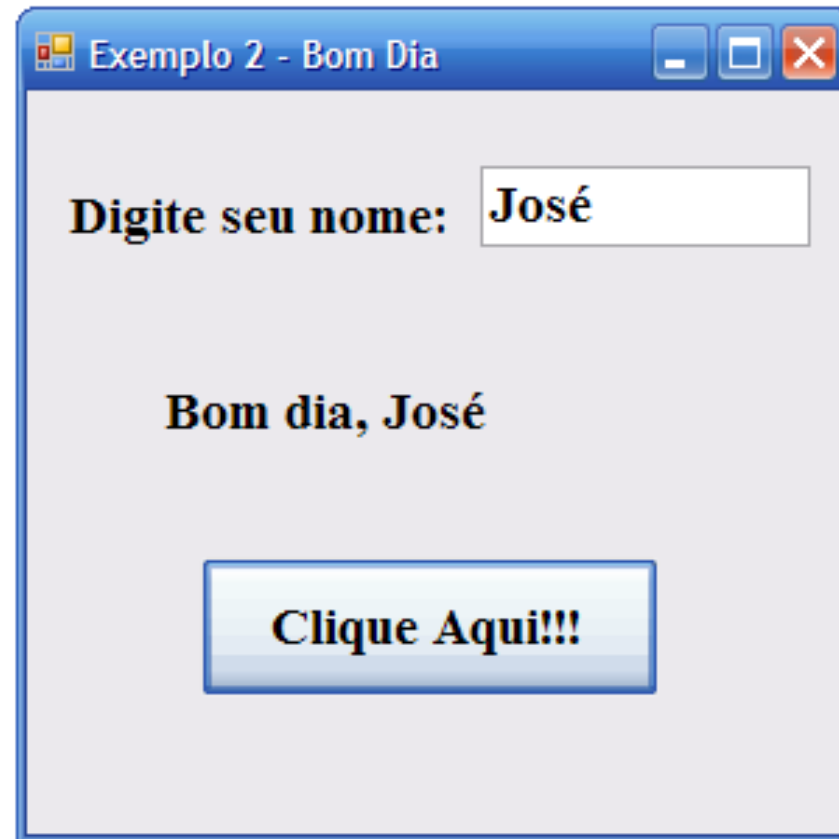
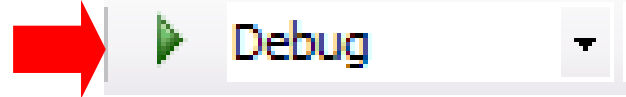
A diferença é que no primeiro não usamos variável e no segundo usamos a variável nome do tipo STRING (texto) para armazenar as informações digitadas pelo usuário.

Funciona sem variável?

Sim, pois a propriedade TEXT armazena automaticamente as informações digitadas. Lembre-se que ela armazena somente Texto (STRING)

3. Passo: Codificar o Programa

Clique na seta verde ou pressione F5



CONCLUSÃO

Neste exemplo aprendemos a usar um objeto novo (TEXTBOX) e variável.

- **TEXTBOX** – usado para capturar as informações digitadas pelo usuário. Estas informações serão **SEMPRE** no formato de texto (STRING). Quando precisarmos de valores numéricos (INT ou DOUBLE) teremos que converter.

- **VARIÁVEL** – Variável é o nome dado a uma posição de memória onde podemos armazenar um tipo de dado definido. Os tipos são:

int – dados do tipo inteiro

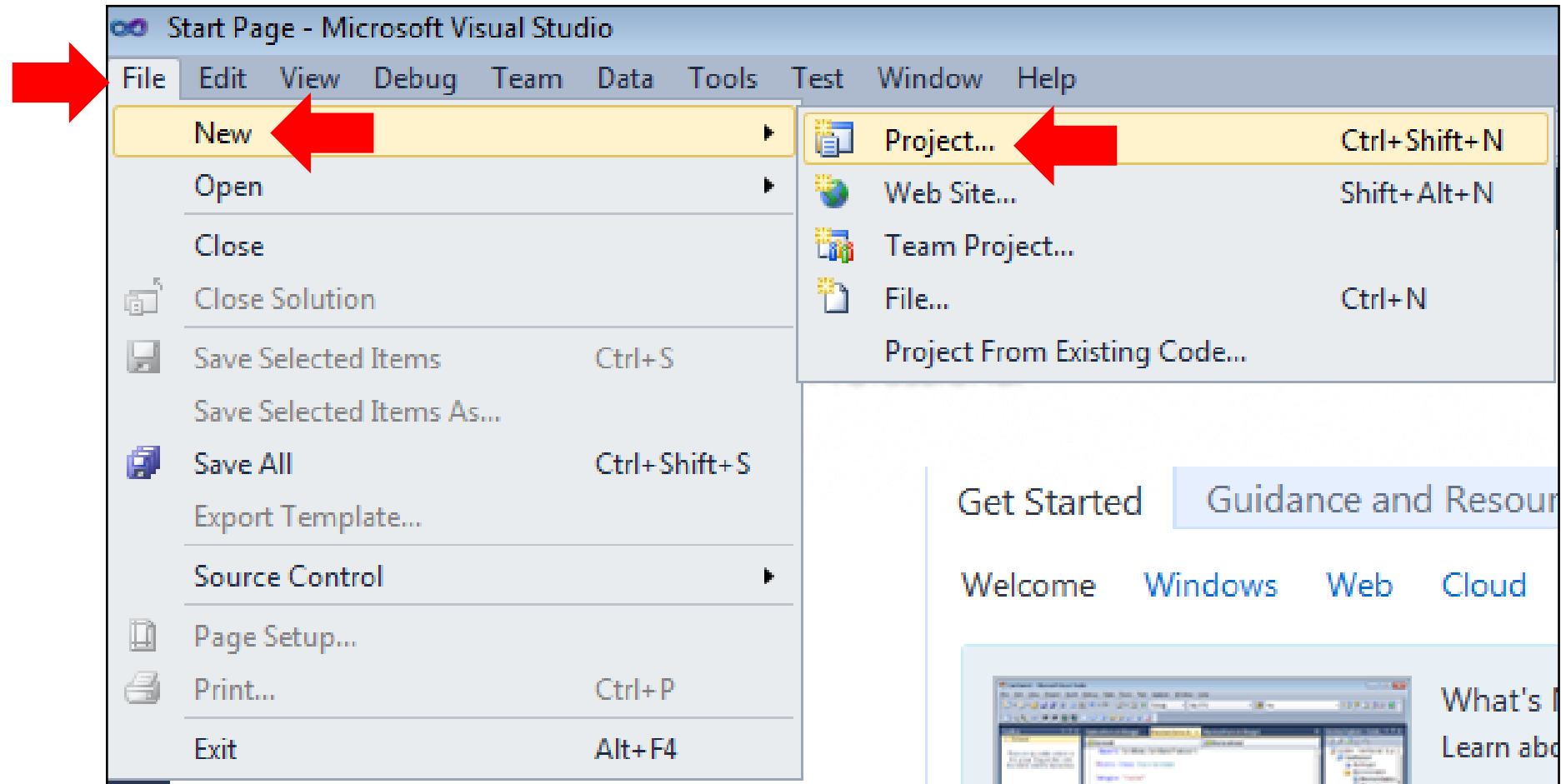
double – dados do tipo real

String ou string – dados do tipo texto

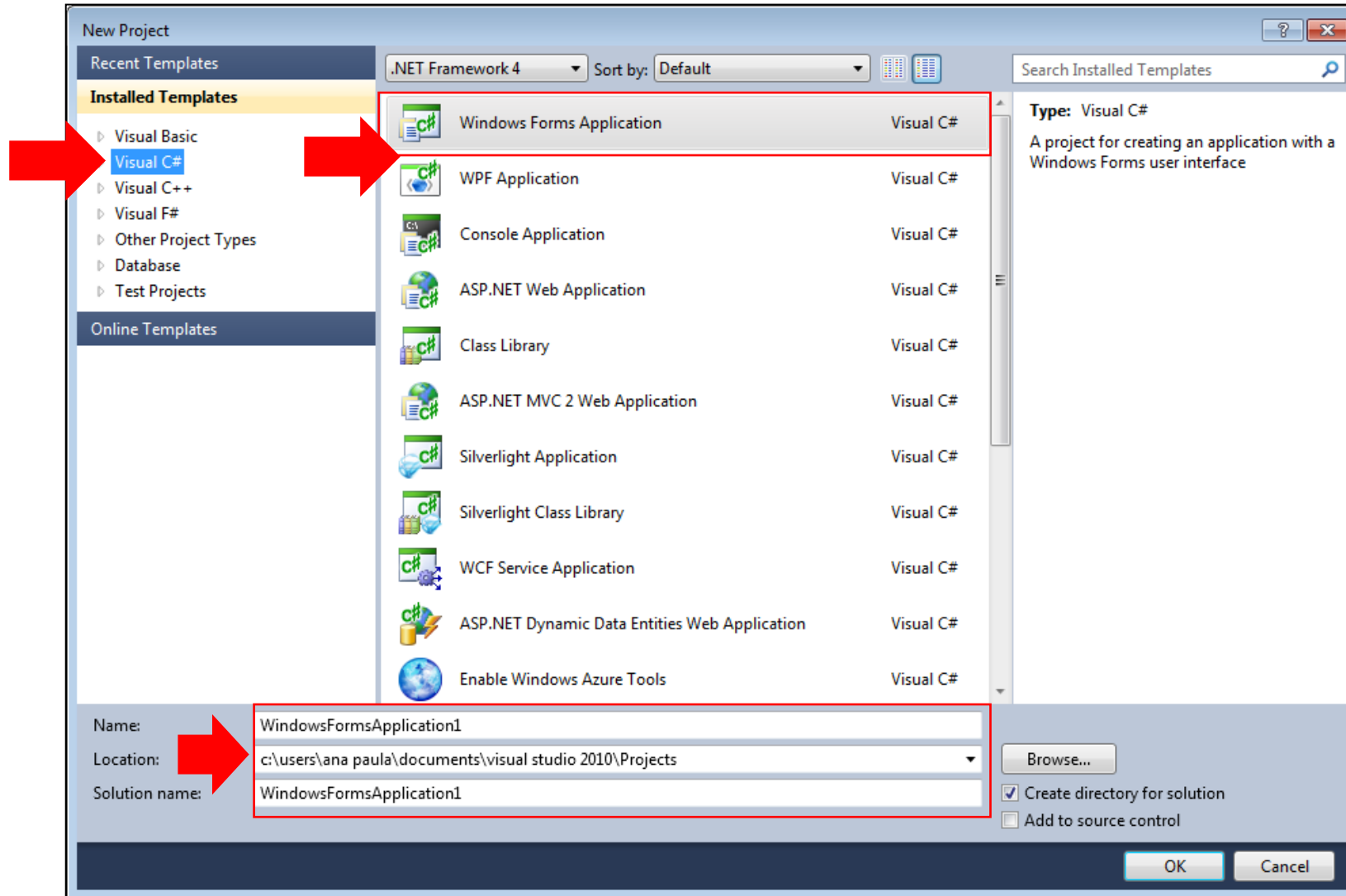
EXEMPLO 3

**FAÇA UM PROGRAMA QUE SIMULE
UMA CALCULADORA DE 4
OPERAÇÕES BÁSICAS (Somar,
Subtrair, Multiplicar e Dividir)**

1. Passo: Criar um Projeto



1. Passo: Criar um Projeto



1. Passo: Criar um Projeto

Troque o nome para Calculadora

Escolha um diretório

Name: Exemplo1

Location: E:\Ana Paula\Aho 2011\CDT\Programas\

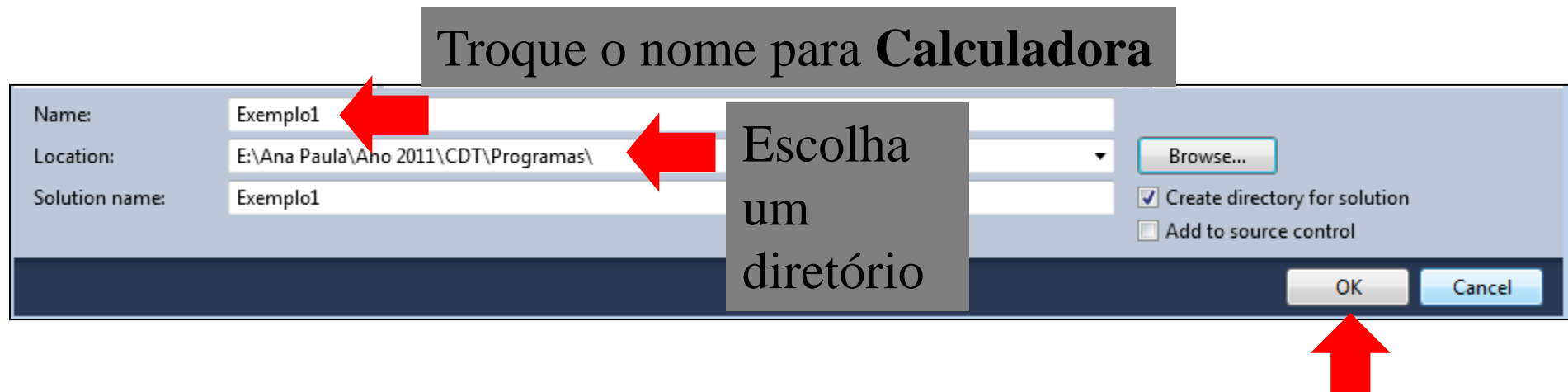
Solution name: Exemplo1

Browse...

☒ Create directory for solution

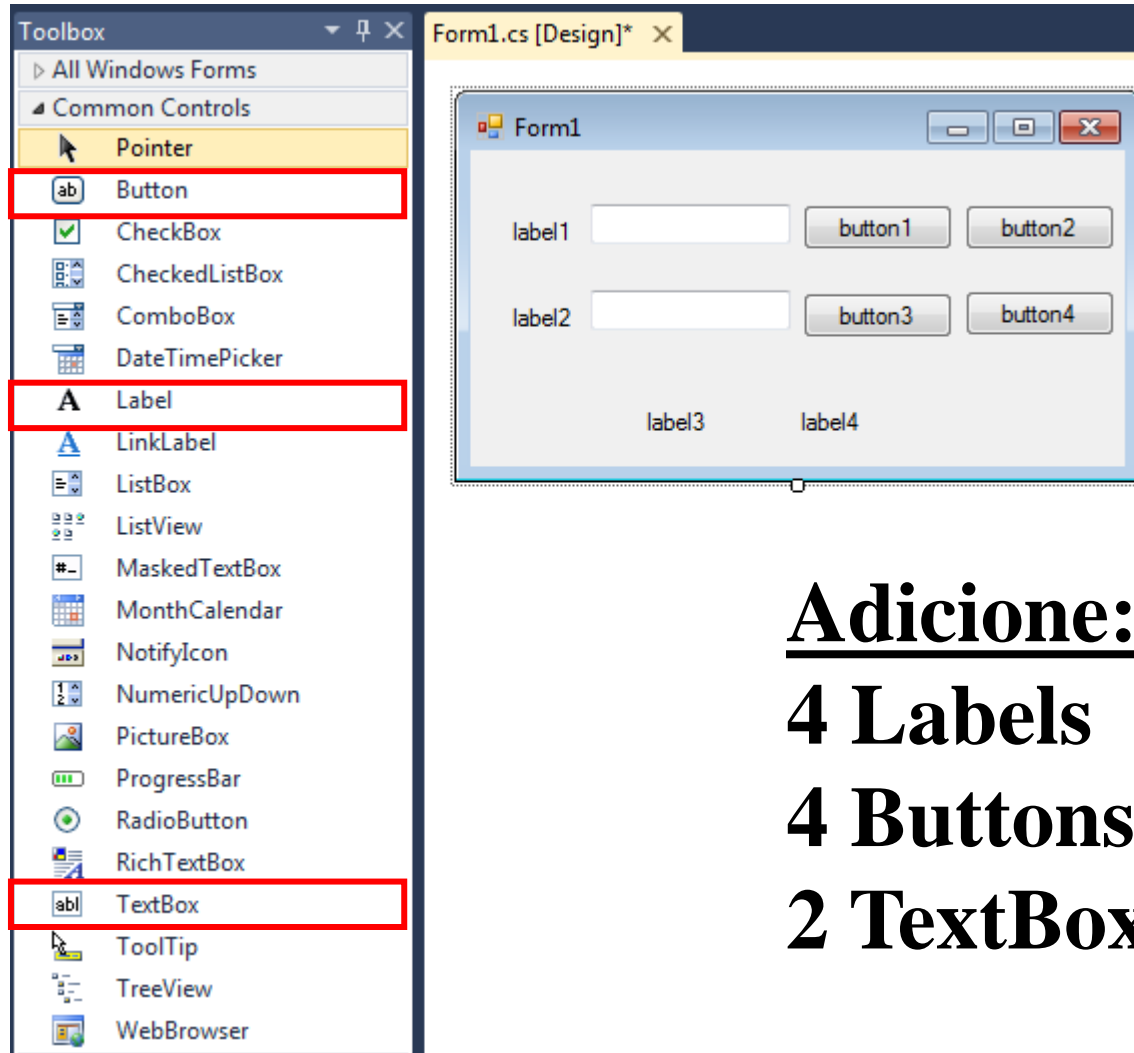
☐ Add to source control

OK Cancel

A screenshot of a project creation dialog box. The dialog has three input fields: 'Name' with 'Exemplo1', 'Location' with 'E:\Ana Paula\Aho 2011\CDT\Programas\' and a dropdown arrow, and 'Solution name' with 'Exemplo1'. To the right of the 'Location' field is a 'Browse...' button. Below these are two checkboxes: 'Create directory for solution' (checked) and 'Add to source control' (unchecked). At the bottom right are 'OK' and 'Cancel' buttons. Annotations include a grey box at the top saying 'Troque o nome para Calculadora' with a red arrow pointing to the 'Name' field, another grey box on the right saying 'Escolha um diretório' with a red arrow pointing to the 'Location' field, and a red arrow pointing up to the 'OK' button.

Clique **OK**

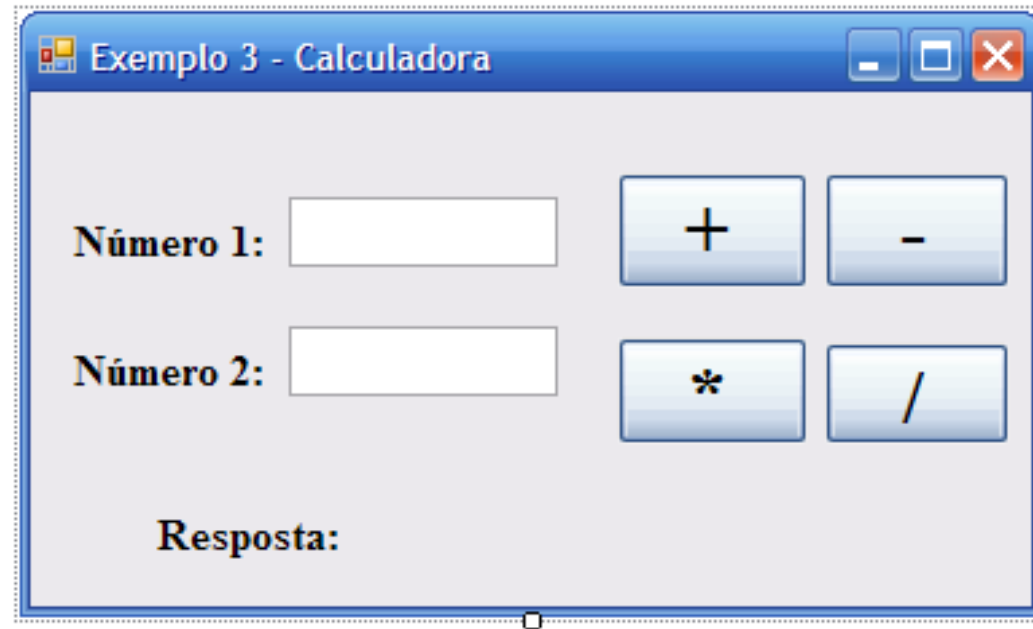
2. Passo: Construir a Tela



Adicione:
4 Labels
4 Buttons
2 TextBoxs

2. Passo: Construir a Tela

Para que o FORM fique assim deve-se alterar algumas propriedades dos três objetos: FORM, BUTTON, LABEL e TEXTBOX.



Tome cuidado para não alterar a propriedade do objeto errado. **CLIQUE 1 VEZ NO OBJETO, PARA SELECIONÁ-LO, E DEPOIS ALTERE AS PROPRIEDADES.**

2. Passo: Construir a Tela

Label1

Text: Número 1

Name: lblNum1

Button1

Text: +

Name: btnSomar

TextBox1

Text:

Name: txtNum1

Label2

Text: Número 2

Name: lblNum2

Button2

Text: -

Name: btnSubtrair

TextBox2

Text:

Name: txtNum2

Label3

Text: Resposta

Name: lblMenResp

Button3

Text: *

Name: btnMultiplicar

Form1

Text: Exemplo 3 –
Calculadora

Name: frmTelaPrincipal

Label4

Text:

Name: lblResposta

Button4

Text: /

Name: btnDividir

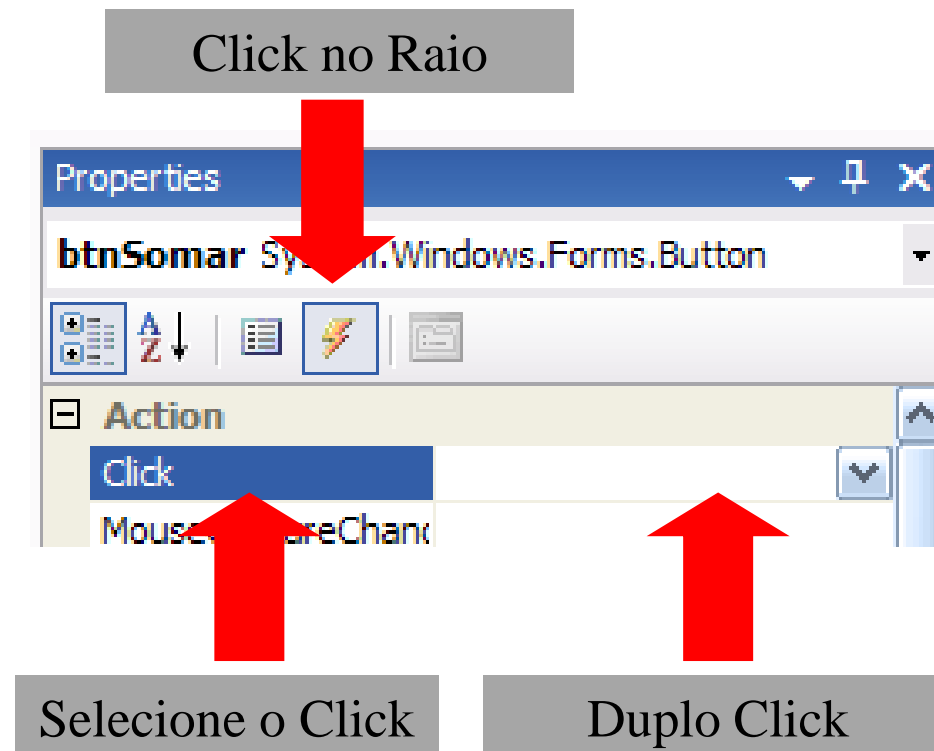
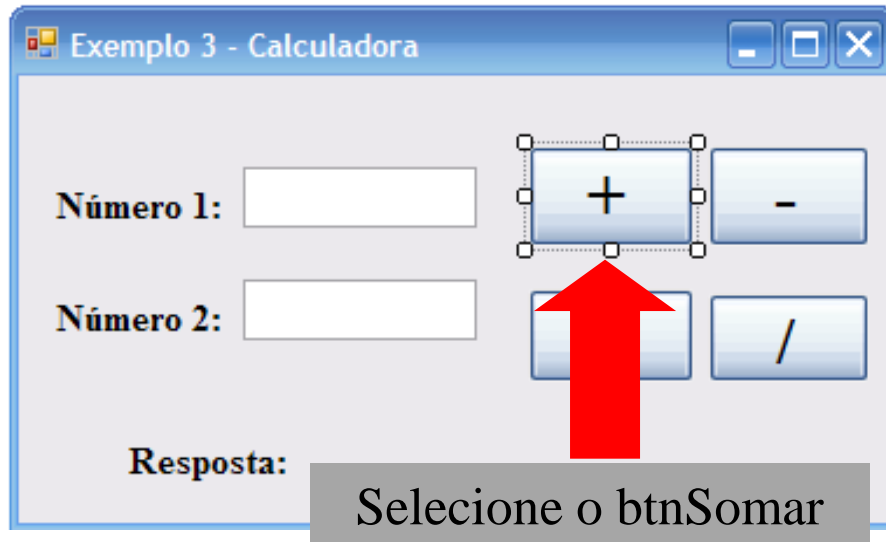
2. Passo: Construir a Tela

As propriedades NAME e TEXT sempre serão alteradas.

Existem outras propriedades que também são interessantes, teste as seguintes propriedades:

- **BACKCOLOR** – usada para alterar a cor de fundo
- **FONT** – usada para alterar a fonte (estilo, tamanho, entre outros)
- **FORECOLOR** – usada para alterar a cor da fonte

3. Passo: Codificar o Programa



3. Passo: Codificar o Programa

```
private void btnSomar_Click(object sender, EventArgs e)
{
    int n1, n2, resp;
    n1 = Convert.ToInt32(txtNum1.Text);
    n2 = Convert.ToInt32(txtNum2.Text);
    resp = n1 + n2;
    lblResposta.Text = resp.ToString();
}
```



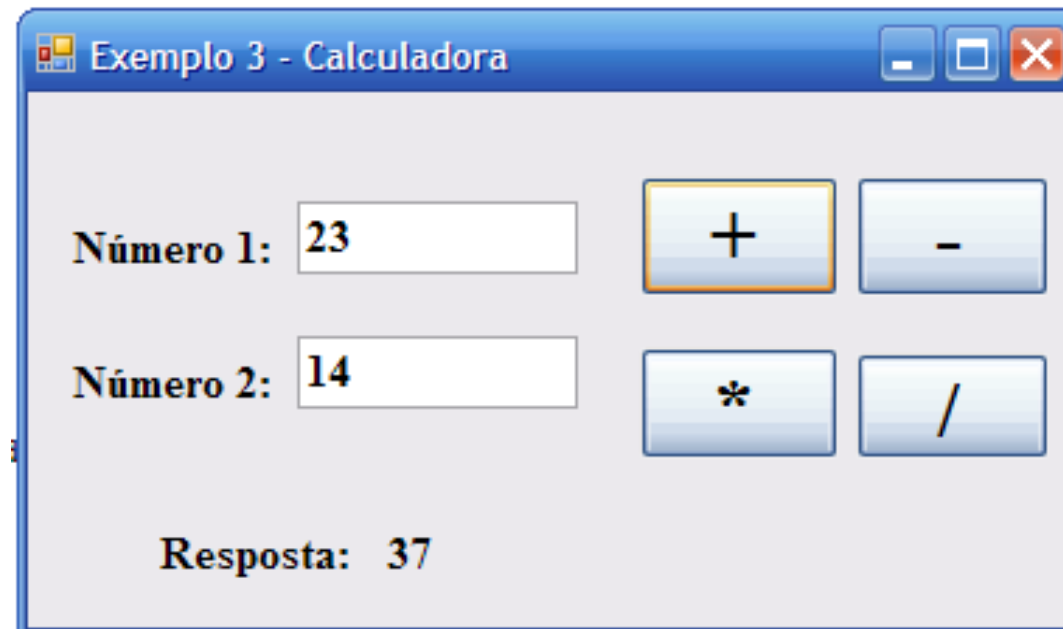
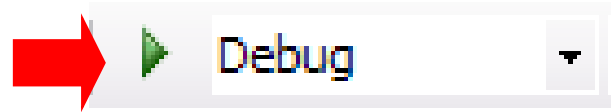
Escreva o código acima.

DICA: Use as teclas Ctrl+Barra de Espaço para facilitar a digitação

**Repita estas operações para os outros botões
btnSubtrair, btnMultiplicar, btnDividir**

3. Passo: Codificar o Programa

Clique na seta verde ou pressione F5



CONCLUSÃO

Neste exemplo aprendemos a usar conversão de tipos.

- **CONVERSÃO DE TIPOS** – método usado para converter o texto digitado pelo usuário nos TEXTBOXS para variáveis numéricas e/ou variáveis numéricas para texto.

Texto para inteiro

Variável Numérica Inteira = **Convert.ToInt32(Texto)**;

Texto para real

Variável Numérica Real = **Convert.ToDouble(Texto)**;

Inteiro ou Real para Texto

Variável Texto = Variável Numérica.**ToString()**;