1 Seja produtivo com C#

Aplicativos visuais em 10 minutos ou menos

Quer criar ótimos programas realmente rápidos?

O C# é uma linguagem de programação poderosa e uma ferramenta valiosa na palma de sua mão. **Com a IDE do Visual Studio você nunca mais gastará horas escrevendo código obscuro para um botão funcionar**. Melhor ainda, você poderá concentrar-se em realizar o seu trabalho, em vez de lembrar qual parâmetro de qual método é o nome para um botão e qual é o seu rótulo. Interessante? Vire a página e vamos começar a programar.

Por que <u>você</u> deve aprender C#

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

O IDE - ou Visual Studio Integrated Development Environment (Ambiente Integrado de Desenvolvimento) - é uma parte importante de trabalhar com C#. É um programa que ajuda a editar seu código, gerenciar seus arquivos e publicar seus projetos.

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());
        }
    }
}
```

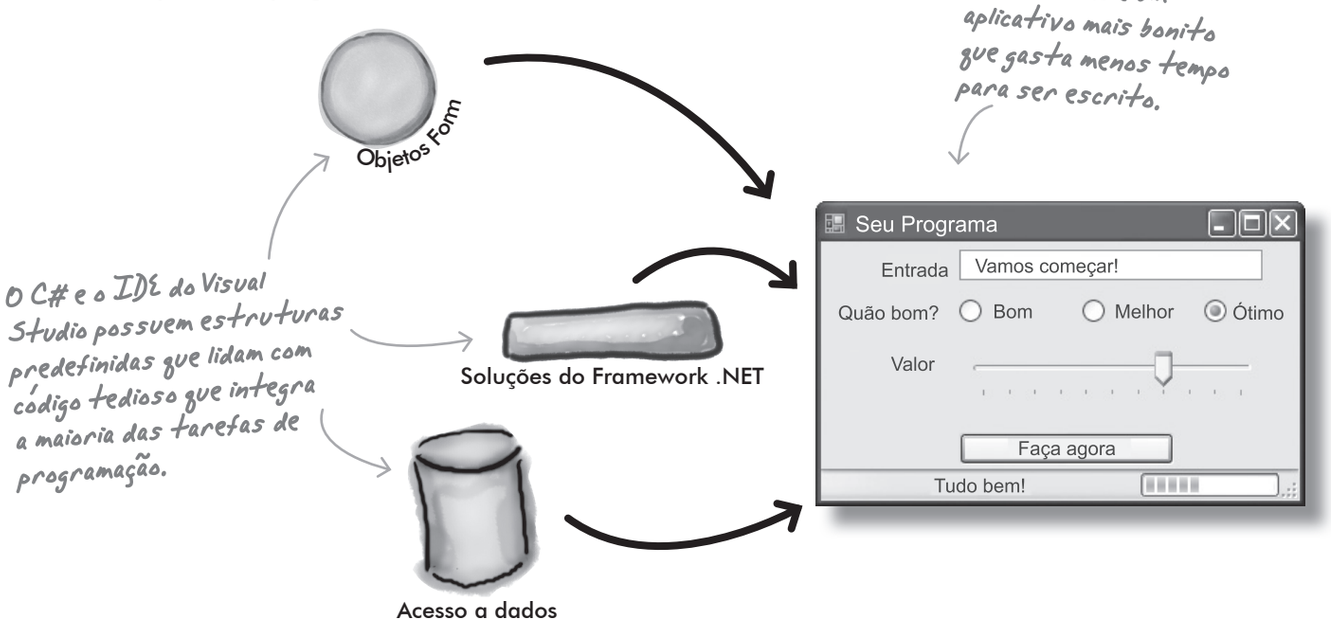
```
private void InitializeComponent()
{
  this.button1 = new System.Windows.Forms.Button();
  this.SuspendLayout();
  //
  // button1
  //
  this.button1.Location = new System.Drawing.Point(105, 56);
  this.button1.Name = "button1";
  this.button1.Size = new System.Drawing.Size(75, 23);
  this.button1.TabIndex = 0;
  this.button1.Text = "button1";
  this.button1.UseVisualStyleBackColor = true;
  this.button1.Click += new System.EventHandler(this.button1_Click);
  //
  // Form1
  //
  this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 16F);
  this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
  this.ClientSize = new System.Drawing.Size(292, 267);
  this.Controls.Add(this.button1);
  this.Name = "Form1";
  this.Text = "Form1";
  this.ResumeLayout(false);
}
```

É preciso de todo este código apenas para desenhar um botão em um formulário. Adicionar alguns elementos visuais adicionais ao formulário poderia precisar até de dez vezes mais código.

O que você consegue com o Visual Studio e C#...

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

O resultado é um aplicativo mais bonito que gasta menos tempo para ser escrito.



O C# e o IDE do Visual Studio possuem estruturas predefinidas que lidam com código tedioso que integra a maioria das tarefas de programação.

O C# e o IDE do Visual Studio facilitam muitas coisas

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

1. **Faça um aplicativo RAPIDAMENTE.** Criar programas em C# leva segundos. A linguagem é poderosa e fácil de aprender; e o IDE do Visual Studio faz muito do trabalho automaticamente para você. Podem-se deixar tarefas de código simples para o IDE e concentrar-se no que o seu código deveria fazer.

2. **Faça uma interface de usuário com boa aparência.** O Form Designer no IDE do Visual Studio é uma das ferramentas de design mais fáceis de usar que existem por aí. Ela faz tanto que você descobrirá que criar interfaces de usuário lindíssimas é umas das partes mais satisfatórias de desenvolver um aplicativo em C#. Você pode fazer programas profissionais com todas as características sem ter de passar horas escrevendo uma interface de usuário gráfica totalmente do zero.

3. **Crie e interaja com bases de dados.** O IDE inclui uma interface simples para construir bases de dados e integra-se perfeitamente ao SQL Server Express, assim como muitos outros sistemas de base de dados populares.

4. **Concentre-se em resolver seus problemas REAIS.** O IDE faz muito, mas você ainda está no controle do que é feito com o C#. Ele permite-lhe concentrar-se em seu programa, em seu trabalho (ou diversão!) e em seus clientes; mas cuida de todo o trabalho repetitivo, como:
   - ★ manter o registro todos os seus projetos;
   - ★ facilitar a edição do código de seu projeto;
   - ★ manter o registro sobre os gráficos, áudio, ícones e outros recursos de seu projeto;
   - ★ gerenciar e interagir com bases de dados.

   Isto significa que todo o tempo gasto para esta programação de rotina pode ser seu e usado para **criar programas muito legais**.

Você verá exatamente o que queremos dizer a seguir.

Ajude o diretor geral a eliminar os papéis

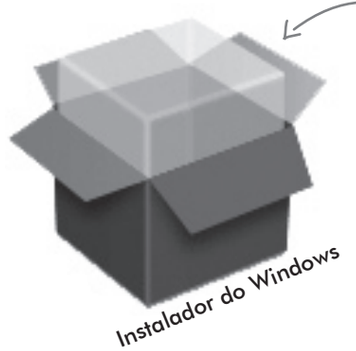
A Empresa de Papel Vila Objeto contratou um novo diretor geral. Ele adora fazer caminhadas, café e a natureza... e ele decidiu ajudar a salvar as florestas; quer ser um executivo "sem papel", começando com seus contatos. Ele está a caminho de Aspen para esquiar no fim de semana e quer ter um novo programa de agenda pronto quando voltar. Caso contrário... bem... não será apenas o antigo diretor geral que estará em busca de um emprego.

É melhor você encontrar rapidamente uma forma destes dados entrarem no notebook do 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 geral 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.

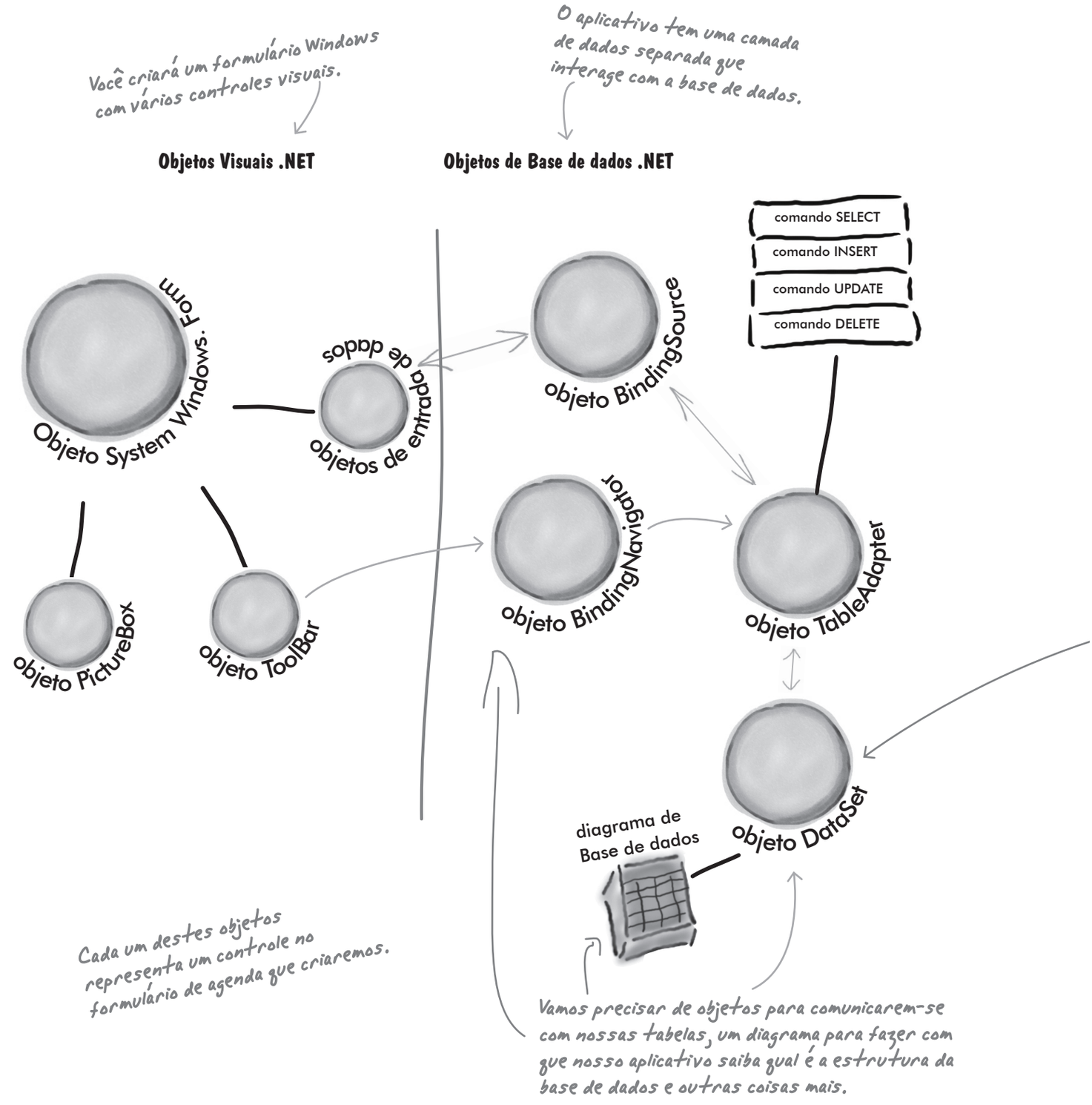
O diretor geral acha que uma base de dados seria a melhor forma para que todos na empresa pudessem ter acesso aos dados dele. Assim, ele pode manter apenas uma cópia de todos os seus contatos.

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 apenas 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 uma base de dados, a própria base de dados e um instalador. Parece muito trabalhoso, mas você fará isto tudo nas próximas páginas.

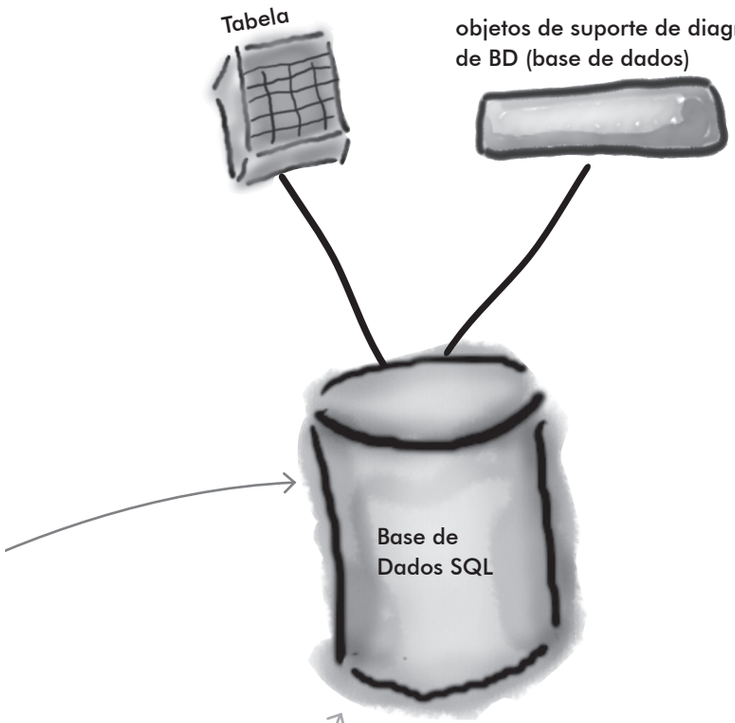
Aqui está a estrutura do programa que criaremos:



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

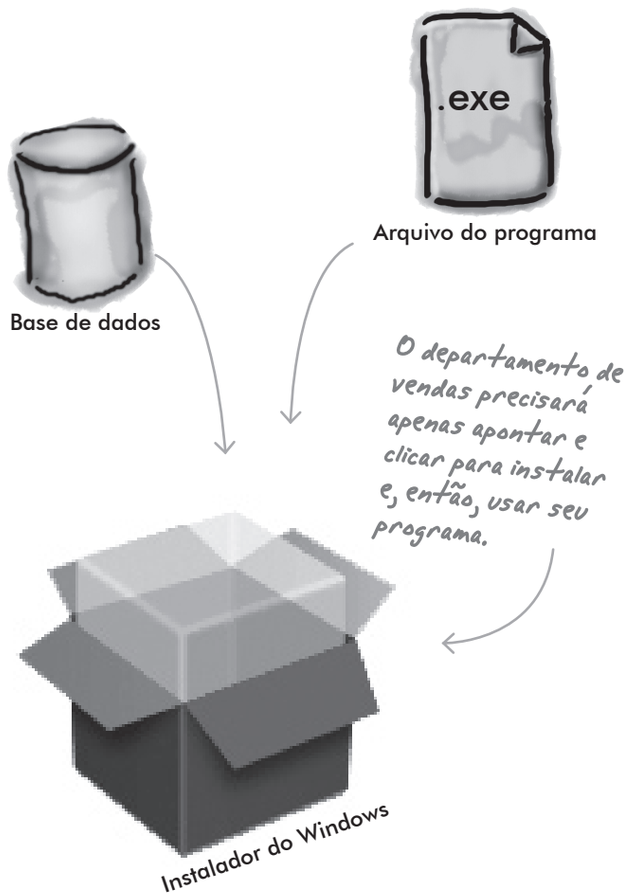
Uma vez que o programa tenha sido feito, ele será incluído num pacote do instalador do Windows.

Armazenamento de Dados



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

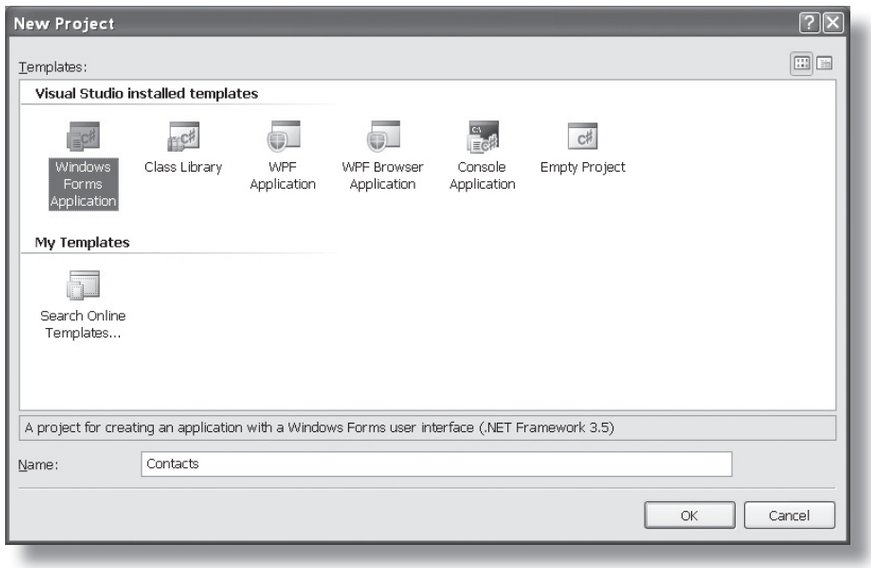
Pacote de Distribuição



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

O que você faz no Visual Studio...

Vá em frente e instale o Visual Studio, se ainda não o fez. Pule a página inicial e selecione New Project (novo projeto) no menu **File** (Arquivo). Nomeie seu projeto como "Contatos" e clique em OK.



Veja bem!

As coisas podem parecer um pouco diferentes em seu IDE.

Esta é a aparência da janela "New Project" (Novo Projeto) no Visual Studio 2008 Express Edition. Se você estiver usando a edição Professional ou Team Foundation, pode ser um pouco diferente. Mas não se preocupe, tudo ainda funciona exatamente da mesma forma.

O que o Visual Studio faz por você...

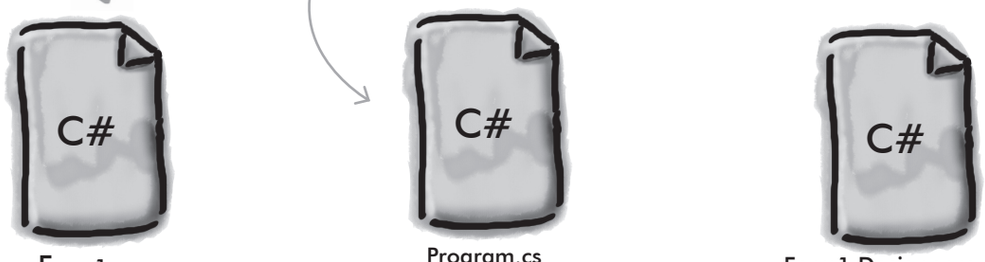
Quando você inicia um novo projeto, assim que você o salva, o IDE cria os arquivos Form1.cs, Form1. Designer.cs, e Program.cs. Ele acrescenta-os à janela Solution Explorer (Navegador de Solução) e, por padrão, coloca-os em `Meus Documentos\Visual Studio 2008\Projects\Contacts\`.

Certifique-se de salvar seu projeto assim que o criar, selecionando Save All (Salvar Tudo) no menu File (Arquivo) – isto salvará todos os arquivos do projeto. Se você selecionar Salvar, somente aquele no qual você está trabalhando é salvo.

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.



Form1.cs

Program.cs

Form1.Designer.cs

O Visual Studio cria estes três arquivos automaticamente.

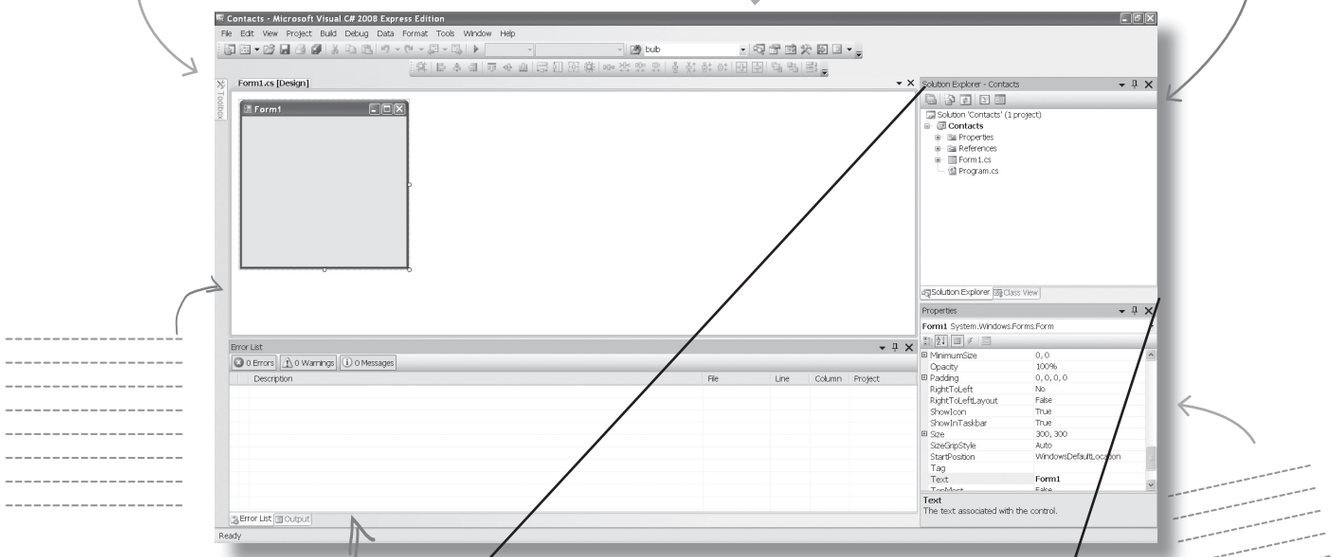
Aponte seu lápis

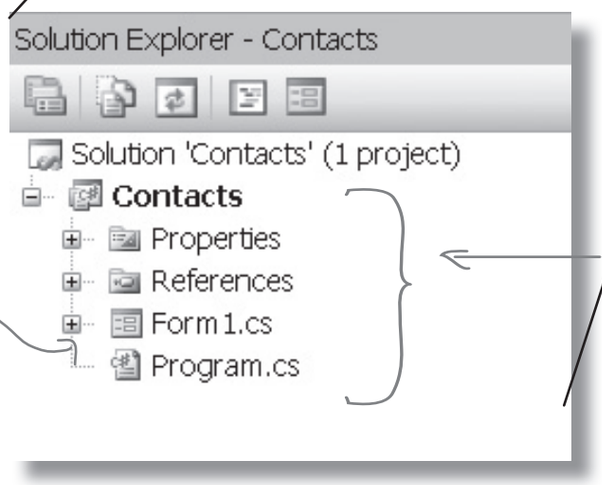
Abaixo vemos como, provavelmente, sua tela está agora. Você já deve ter percebido que a maioria destas janelas e arquivos se baseiam no que você já sabe. Preencha cada um dos espaços em branco com uma anotação, tentando descrever o que aquela parte do IDE faz. Adiantamos seu trabalho fazendo um para você.

Esta barra de ferramentas possui botões correspondentes ao que você está fazendo atualmente no IDE.

Se o seu IDE não parece exatamente com o desta figura, você pode selecionar Reset Window Layout (Reiniciar o Layout das Janelas) no menu Window (Janela).

Ampliamos esta janela abaixo para que você tenha mais espaço.





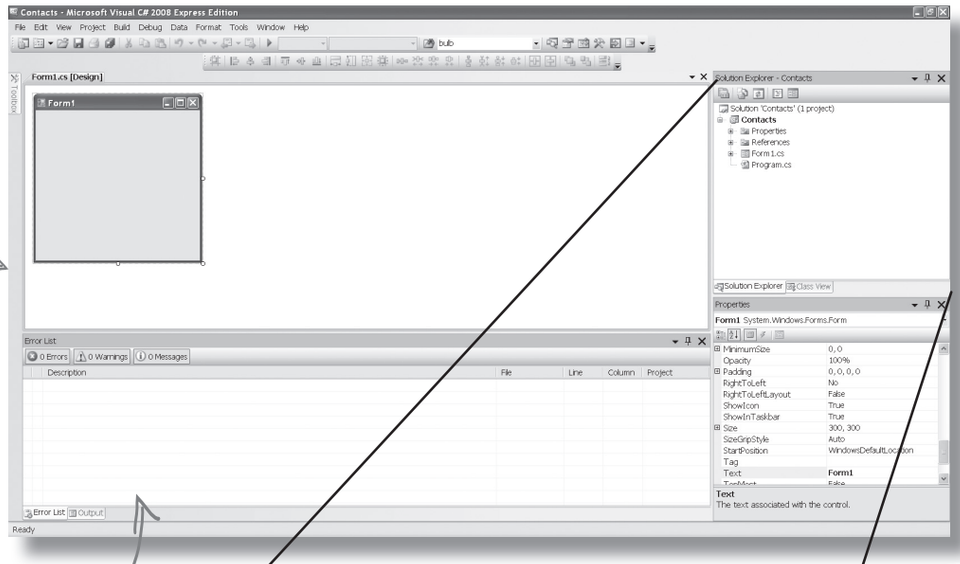
Você também pode configurar a essas janelas, selecionando Solucion Explorer, Propriedade de erro ou a partir da lista Exibir menu.

Aponte seu lápis
Solução

Preenchemos os campos com as anotações sobre as diferentes seções do IDE do Visual Studio C#. Você pode ter escrito algumas coisas diferentes, mas já deve ter percebido o básico sobre as finalidades de cada grupo de janelas do IDE.

Esta barra de ferramentas possui botões que correspondem ao que você está fazendo atualmente no IDE.

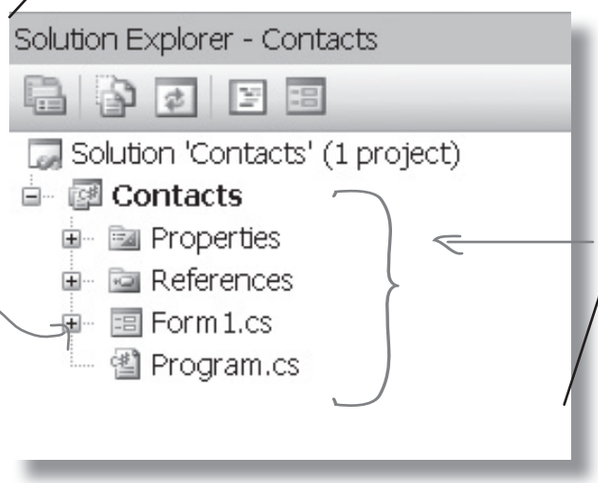
Ampliamos essa janela abaixo para que você tenha mais espaço.



Essa é a caixa de ferramentas. Ela possui vários controles visuais que você pode arrastar para seu formulário.

Este painel abaixo é para depuração. Ele mostra-lhe quando há erros em seu código.

Esta janela mostra todas as propriedades dos controles em seu formulário.



Aparecem no Navegador de Solução os arquivos Form1.cs e Program.cs que o IDE criou quando o novo projeto foi iniciado.

Você pode trocar de arquivos usando o Navegador de Solução do IDE.

não existem Perguntas Idiotas

P: Se o IDE escreve todo esse código para mim, aprender C# é apenas uma questão de aprender como usar o IDE?

R: Não. O IDE é ótimo em gerar automaticamente algum código para você, mas é só isso que ele pode fazer. Há algumas coisas em que ele é muito bom, como estabelecer bons pontos de partida e mudar propriedades de controles automaticamente em seus formulários. Mas a parte difícil da programação – descobrir e implementar o que seu programa precisa fazer – é algo que nenhum IDE pode fazer por você. Embora o IDE do Visual Studio seja um dos ambientes de desenvolvimento mais avançados, ele só pode ir até esse ponto. É você – não o IDE – que escreve o código de ação, ou o código que faz o trabalho.

P: Eu criei um novo projeto no Visual Studio, mas quando entrei na pasta "Projetos", em Meus Documentos, não o vi lá. O que acontece?

R: Em primeiro lugar, você deve estar usando o Visual Studio 2008 – no 2005 isso não acontecia. Quando você cria pela primeira vez um novo projeto no Visual Studio 2008, o IDE cria o projeto em sua pasta `Local Settings\Application Data\TemporaryProjects`. Quando você salva o projeto pela primeira vez, ele pede, através de uma janela, um novo nome de arquivo e salva-o na pasta `Meus Documentos\Visual Studio2008\Projects`. Se você tentar abrir um novo projeto ou fechar o temporário, uma janela se abrirá para perguntar se você quer salvar ou descartar o projeto temporário.

P: E se o IDE criar código que eu não queira em meu projeto?

R: Você pode modificá-lo. O IDE é feito para criar o código baseando-se na forma que o elemento que você arrastou ou adicionou é mais usado comumente. Mas, às vezes, não é exatamente isto que você quer. Tudo que o IDE faz por você – todas as linhas de código que ele cria, todos os arquivos que adiciona – pode ser alterado manualmente, editando-se os arquivos, ou através de uma interface simples de usar no IDE.

P: Tudo bem se eu baixei e instalei o Visual Studio Express? Ou eu preciso usar uma das versões do Visual Studio que não são de graça para fazer o descrito no livro?

R: Não há nada neste livro que você não possa fazer com a versão grátis do Visual Studio (que você pode baixar na página da Microsoft). As principais diferenças entre o Express e as outras edições (Professional e Team Foundation) não atrapalharão a forma de escrever em C# e de criar aplicativos totalmente funcionais e completos.

P: Posso mudar os nomes dos arquivos que o IDE cria?

R: Claro, você pode mudar qualquer aspecto de seu programa. Mas o IDE está programado para nomear seus arquivos de forma coerente. Quando você adiciona um arquivo ao seu projeto, o nome do arquivo que você escolhe afeta a forma na qual o código é gerado e o código criado incluirá esse nome. Em alguns casos, se você renomear o arquivo, ou terá de mudar outras partes em todo o código, ou terá de lidar com a diferença entre o nome do arquivo e do código dentro dele. Já que isto é um pouco incômodo, recomendamos que você não mude os nomes dos arquivos a não ser que tenha uma boa razão para isso.

P: Estou olhando para o IDE neste momento, mas minha tela não parece com a sua! Algumas janelas estão faltando e outras estão no lugar errado. O que acontece?

R: Se você clicar no comando "Reset Window Layout" (Reiniciar Layout de Janelas) no menu "Window" (janela), o IDE deve restaurar o layout padrão das janelas. Então sua tela ficará igualzinha as deste capítulo.

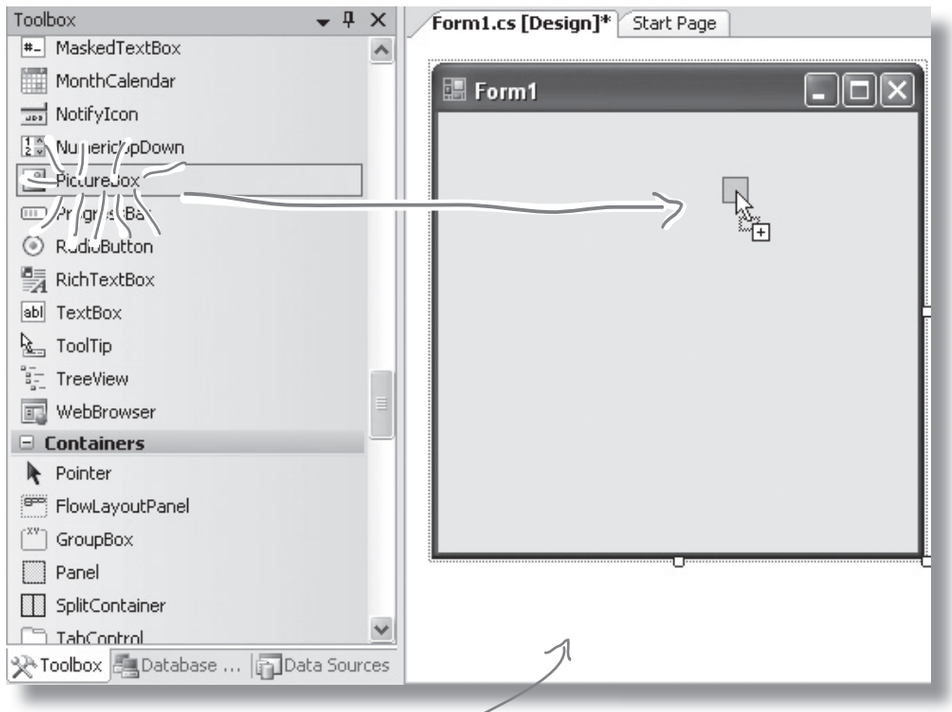
O Visual Studio gerará código que pode ser um ponto de partida para seus aplicativos. Certificar-se de que o aplicativo faz aquilo que deve fazer ainda cabe a você.

Desenvolva a interface de usuário

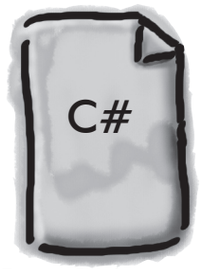
Adicionar controles e arrumar a interface de usuário é tão fácil quanto arrastar e soltar no IDE do Visual Studio. Vamos acrescentar um logo ao formulário:

1. **Utilize o controle PictureBox para acrescentar uma figura.** Clique no controle PictureBox (caixa de imagem) 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.

Se você não estiver vendo a caixa de ferramentas, tente colocar o mouse sobre a palavra "Toolbox" (caixa de ferramentas) que aparece no canto superior esquerdo do IDE. Se não estiver lá, selecione "Toolbox" do menu View (Visualizar) para fazê-la aparecer.



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



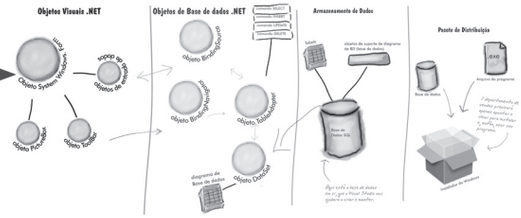
Form1.Designer.cs

Relaxe

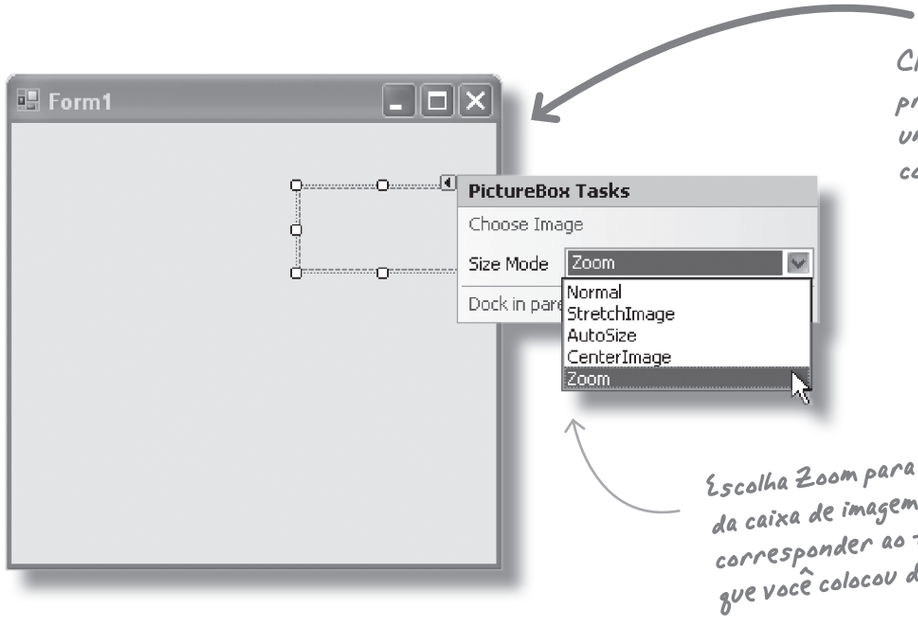
Tudo bem se você não for um profissional em design de interface de usuário.

Falaremos muito mais sobre criar boas interfaces de usuário mais tarde. Por enquanto, vamos apenas colocar o logo e outros controles em seu formulário e preocupar-nos com o comportamento. Adicionaremos mais um pouco de estilo mais tarde.

Você está aqui



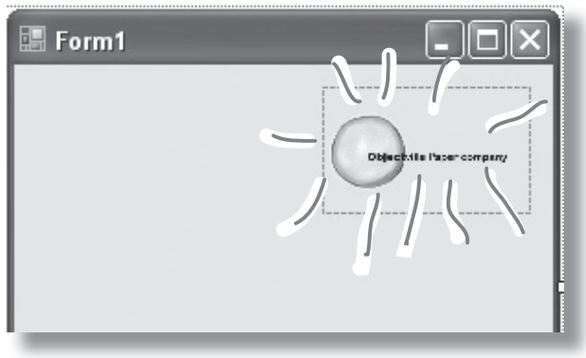
2 **Coloque a caixa de imagem em modo Zoom.**
Todos os controles em seu formulário possuem propriedades ajustáveis. Clique na flechinha preta para um controle para acessá-las. Altere a propriedade Size (Tamanho) da PictureBox para "Zoom" para ver como isto funciona:



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.

3 **Baixe o logo da Empresa de Papel Vila Objeto.**
Baixe o logo da Empresa de Papel Vila Objeto dos laboratórios do Use a Cabeça (`www.altabooks.com.br`) e salve-o em seu disco rígido. Então clique na seta de propriedades da PictureBox e selecione Escolher Imagem. Clique em Importar, encontre seu logo e está tudo pronto:

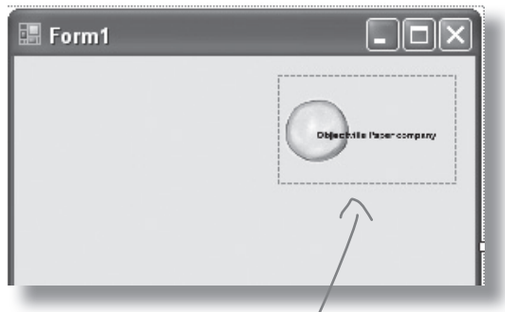


Aqui está o logo da OPC (Objectville Paper Company, Empresa de Papel Vila Objeto). A PictureBox usa o zoom para ficar do tamanho certo.

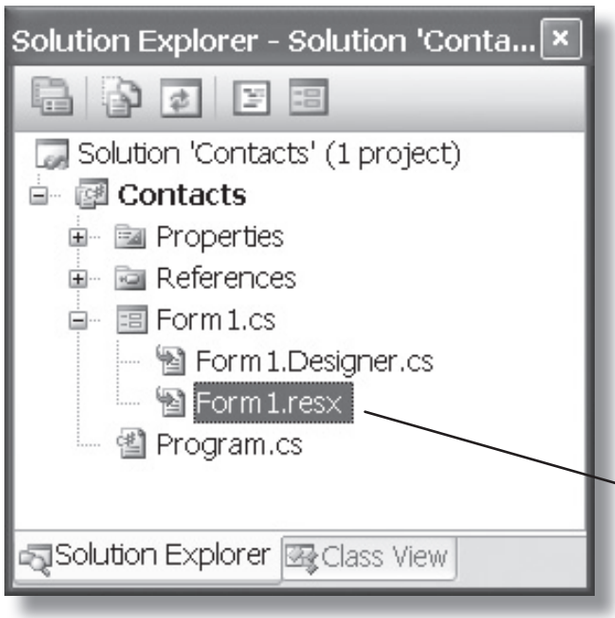
Visual Studio, nos bastidores

Toda vez que você faz algo no IDE do Visual Studio, ele está **escrevendo código para você**. Quando você criou o logo e mandou o Visual Studio usar a imagem baixada, 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 seu aplicativo. O arquivo gráfico fica integrado ao programa, para que, então, quando ele for instalado em outro computador, o gráfico seja instalado junto com ele e a PictureBox possa usá-lo.

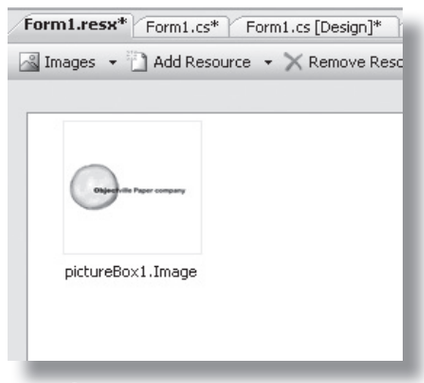
Quando você arrastou o controle PictureBox para o seu formulário, o IDE automaticamente criou um arquivo de recurso chamado Form1.resx para armazená-lo e mantê-lo em seu projeto. Dê um duplo clique neste arquivo e você verá a imagem recém-importada.



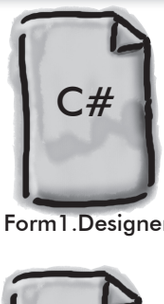
Esta imagem é agora um recurso do aplicativo Lista de Contatos.



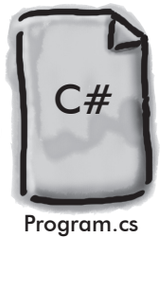
Se você clicar em Form1.resx no Navegador de Solução pode ver a logomarca importada. Este arquivo é o conectado à caixa de imagem; e o IDE adicionou código para fazer a conexão.



Form1.cs



Form1.Designer.cs



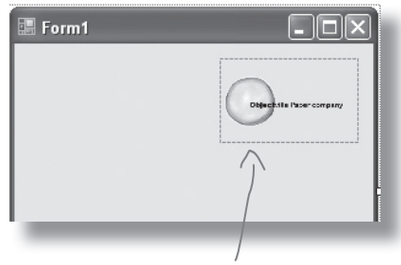
Program.cs

Aqui estão os arquivos que o Visual Studio criou anteriormente.

Form1.resx

Quando você importou a imagem, o IDE criou este arquivo para você. Ele contém todos os recursos (gráficos, vídeo, áudio e outros dados armazenados) associados ao Form1.

Complete o código gerado automaticamente



O IDE cria muito código para você, mas você ainda quer ter acesso a ele e acrescentar-lhe coisas. Vamos usar o logo para mostrar uma caixa de mensagem sobre quando os usuários executarem o programa e clicar cobre o logotipo.

Certifique-se de que seu formulário aparece no IDE e clique duas vezes no controle de caixa de imagem. Você deve ver algum código semelhante ao seguinte 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 controle de caixa de imagem, 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 ideia sobre quando ele executa: quando alguém clica no controle PictureBox.

Quando você clicar duas vezes na caixa de imagem, ela abrirá este código com um cursor piscando bem aqui. Ignore qualquer janela que o IDE mostrar enquanto você digita. Ele está tentando ajudá-lo mas não precisamos disso agora.

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

Uma vez que você tenha digitado a linha de código, salve usando o ícone Save na barra de ferramentas do IDE ou selecionando Save no menu File. Adquira o hábito de clicar em Save All regularmente!

não existem Perguntas Idiotas

P: O que é um método?

R: Um método é apenas um bloco de código com um nome. Falaremos muito mais sobre métodos no capítulo 2.

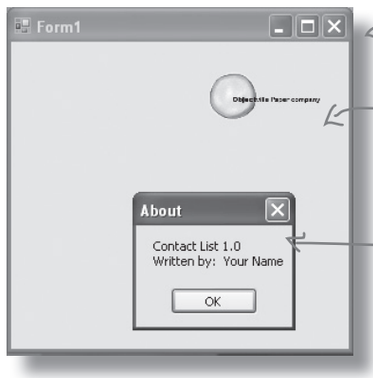
P: O que aquele \n faz?

R: Isto é uma quebra de linha. Ela fala ao C# para colocar "Contact List 1.0" na linha um, e então começar uma nova linha para "Written by:" (Escrito por).

Você já pode executar seu aplicativo

Pressione F5 no seu teclado ou clique no botão com a seta verde (▶) na barra de ferramentas para checar o que você fez até agora. (Isto se chama "Depurar", o que significa apenas executar seu programa usando o IDE). Você pode parar de depurar selecionando "Stop Debugging" (Parar a Depuração) no menu Debug (Depurar) ou clicando neste botão na barra de ferramentas: ■

Esses três botões funcionam – e você não teve que escrever nenhum código para eles.

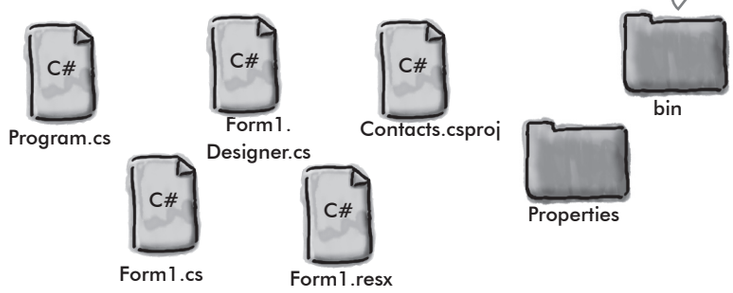


Clicar no logo da OPC faz a caixa Sobre, que você acabou de codificar, aparecer.

Onde estão meus arquivos?

Quando você executa seu programa, o Visual Studio copia todos os seus arquivos para `Meus Documentos\VisualStudio 2008\Projects\Contacts\Contacts\bin\debug`. Você pode acessar rapidamente esse diretório e executar seu programa clicando duas vezes no arquivo .exe que o IDE cria.

O C# transforma seu programa num arquivo que você pode executar, chamado de ***executável****. Você o encontrará aqui, na pasta de depuração.*



Isso não é um erro: existem dois níveis de pastas. A pasta interna possui os arquivos de código C# em si.

não existem Perguntas Idiotas

P: No meu IDE, a seta verde está marcada como "Debug". Isso é ruim?

R: Não. Depurar, pelo menos para nossos propósitos agora, significa apenas executar seu aplicativo dentro do IDE. Falaremos muito mais sobre depuração adiante. Por enquanto, você pode pensar sobre isso como uma forma de executar seu programa.

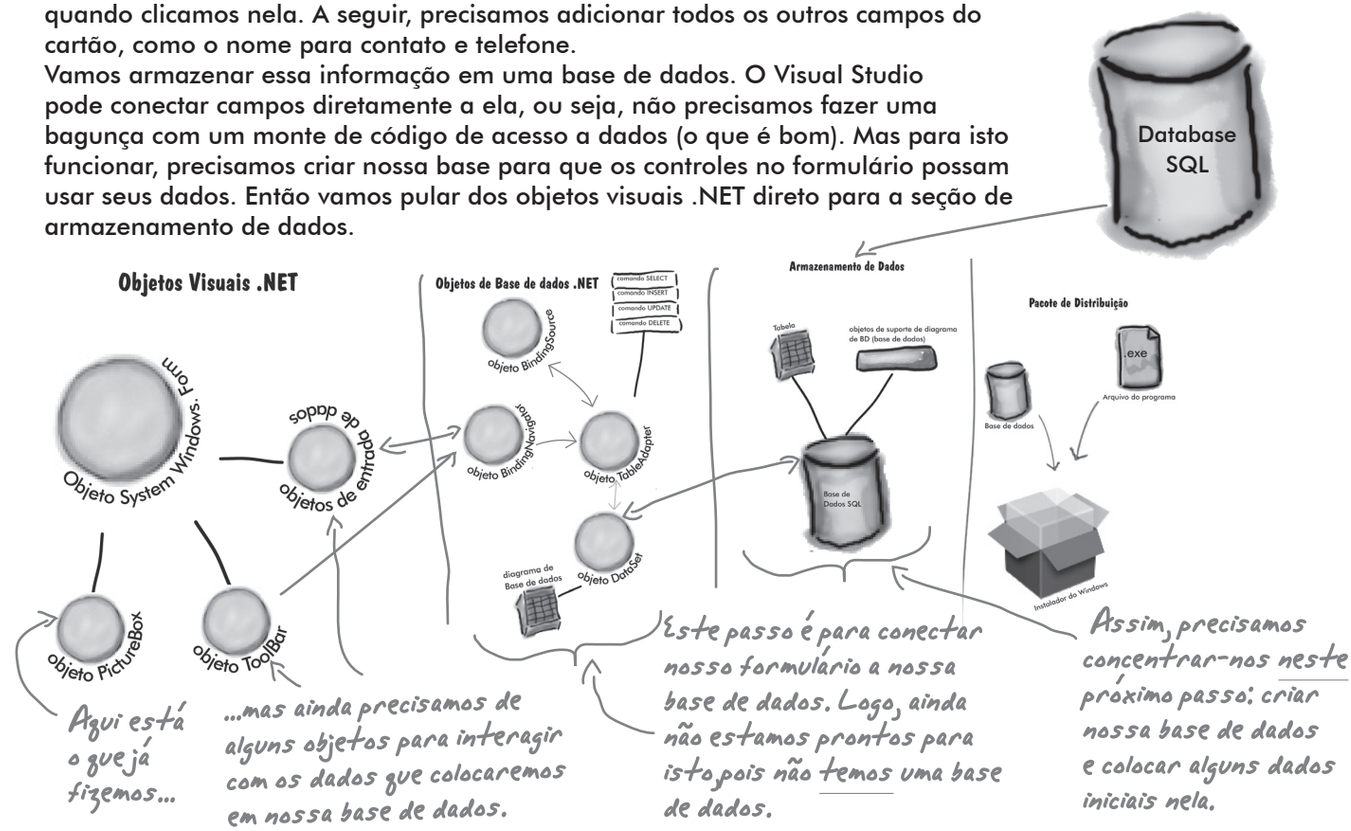
P: Eu não vejo o botão Stop Debugging em minha barra de ferramentas. O que acontece?

R: O botão Stop Debugging aparece apenas em uma barra de ferramentas especial que, por sua vez, aparece somente quando seu programa está em execução. Tente iniciar o aplicativo de novo e veja se ele aparece.

O que já fizemos até agora

Criamos um formulário e um objeto PictureBox que abre uma caixa de mensagem quando clicamos nela. A seguir, precisamos adicionar todos os outros campos do cartão, como o nome para contato e telefone.

Vamos armazenar essa informação em uma base de dados. O Visual Studio pode conectar campos diretamente a ela, ou seja, não precisamos fazer uma bagunça com um monte de código de acesso a dados (o que é bom). Mas para isto funcionar, precisamos criar nossa base para que os controles no formulário possam usar seus dados. Então vamos pular dos objetos visuais .NET direto para a seção de armazenamento de dados.



Aqui está o que já fizemos...

...mas ainda precisamos de alguns objetos para interagir com os dados que colocaremos em nossa base de dados.

Este passo é para conectar nosso formulário a nossa base de dados. Logo, ainda não estamos prontos para isto, pois não temos uma base de dados.

Assim, precisamos concentrar-nos neste próximo passo: criar nossa base de dados e colocar alguns dados iniciais nela.

O Visual Studio pode criar código para conectar seu formulário a uma base de dados, mas você precisa ter a base de dados no lugar certo ANTES de gerar o código

Precisamos de uma base de dados para armazenar nossas informações

Antes de adicionarmos o restante dos campos no formulário, precisamos criar uma base de dados para associar-lhe. O IDE pode criar muito código para conectar nosso formulário com nossos dados, mas precisamos definir a própria base de dados primeiro.

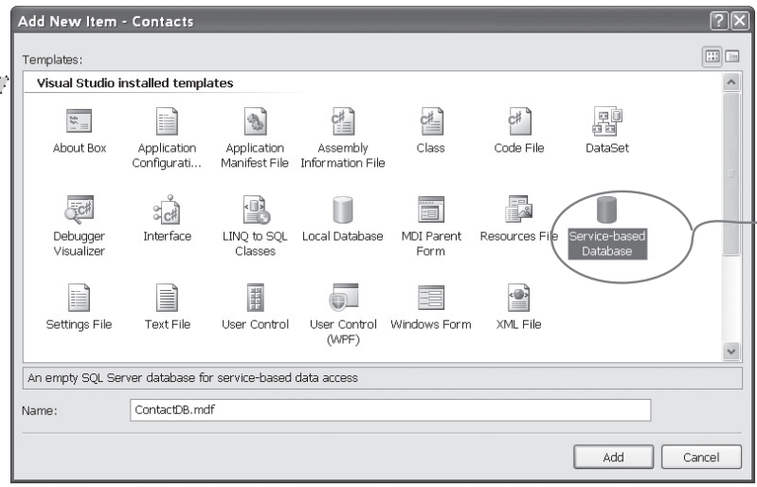
Certifique-se de que parou de depurar antes de continuar.

1 **Adicione uma nova base de dados SQL ao seu projeto.**
No Navegador de Solução, clique com o **botão direito no projeto Contacts** e escolha New Item (Novo Item). Escolha o ícone SQL Database (Base de Dados SQL) e nomeie-o ContactDB.mdf.

Este arquivo é nossa nova base de dados.

ContactDB.mdf

Escolha o ícone certo para a versão que você está usando. Escolha **SQL Database** se você está usando o Visual Studio Express 2005 e **Service-Based Database** (Base de Dados baseada em Serviços) se você está usando o 2008.



O ícone SQL Database funciona apenas se você tiver o SQL Server Express instalado. Volte para o README (Leia-me) se você não tem certeza sobre como fazer isto).

2 **Cancele o Assistente de Configuração de Fonte de Dados.**
Por enquanto, queremos pular a configuração de uma fonte de dados, então clique no botão Cancelar. Voltaremos a isto quando tivermos criado a estrutura de nossa base de dados.

3 **Examine sua base de dados no Navegador de Solução.**
Vá para o Navegador de Solução e você verá que ContactDB foi adicionado à lista de arquivos. Clique duas vezes em ContactDB.mdf e olhe no lado esquerdo de sua tela. A Toolbox mudou para Navegador de Base de Dados.

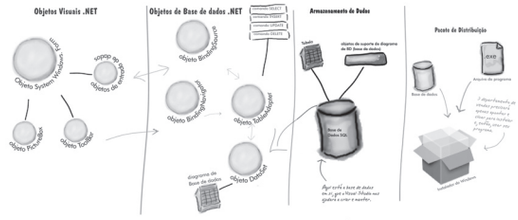
Veja bem!

Se você não está usando a edição Express, verá "Server Explorer", em vez de "Database Explorer".

As edições Professional e Team Foundation do Visual Studio 2008 não possuem uma janela de Navegador de Solução chamada Database Explorer (explorador, ou navegador, de base de dados). Em vez disso, elas possuem uma janela Server Explorer (navegador de servidores), que faz tudo que a Database Explorer faz, mas também lhe permite explorar dados em sua rede.

O IDE criou uma base de dados

Quando você mandou o IDE adicionar uma nova base de dados SQL ao seu projeto, ele criou uma nova para você. Uma **base de dados** SQL é um sistema que armazena dados para você de uma forma organizada e inter-relacionada. O IDE tem todas as ferramentas de que você precisa para manter seus dados e bases.

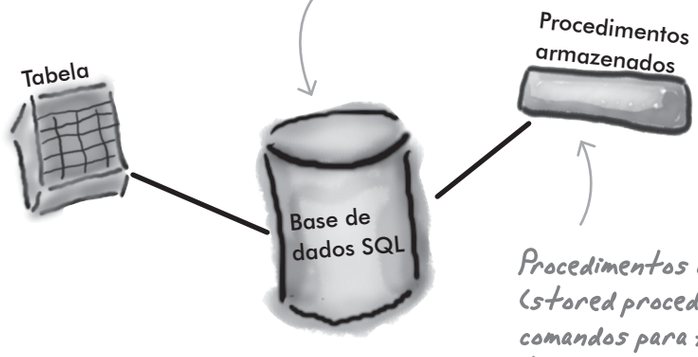


Você está aqui

Os dados na base de dados SQL ficam em tabelas. Por enquanto, você pode pensar em uma tabela como uma planilha. Ela organiza sua informação em colunas e linhas. As colunas são as categorias dos dados, como nome e telefone de contato e cada linha são os dados em si para uma ficha individual.

Uma base de dados SQL armazena seus dados e possui informações sobre como ela é estruturada e código SQL que o ajuda a acessá-los.

Seus dados estão armazenados em tabelas com colunas e linhas, como numa planilha.



Procedimentos armazenados (stored procedures) são comandos para facilitar o trabalho com seus dados.

O SQL é uma linguagem em si mesmo

SQL significa **Linguagem Estruturada de Consulta** (Structured Query Language). É uma linguagem de programação para acessar dados em bases de dados. Possui sua própria sintaxe, palavras-chave e estrutura. O código SQL tem a forma de **comandos** e **consultas**, que acessam e recuperam os dados. Uma base de dados SQL pode possuir **procedimentos armazenados** (stored procedures), que são vários comandos e consultas SQL armazenados na base de dados e podem ser executados a qualquer momento. O IDE gerou comandos SQL e procedimentos armazenados automaticamente para você para permitir ao seu programa acessar os dados na base de dados.

ContactDB.mdf

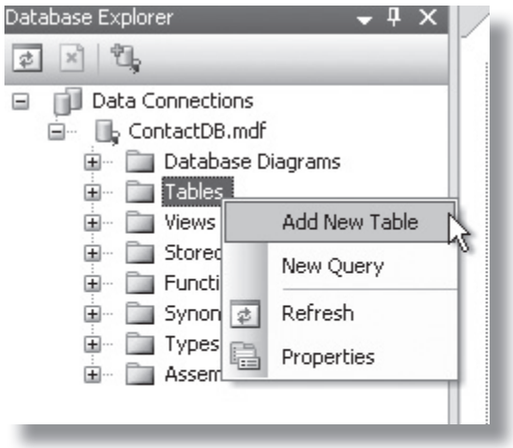
A base de dados SQL está neste arquivo. Agora vamos definir tabelas e dados para ele e tudo isto será armazenado aqui também.

[nota do marketing: podemos colocar uma referência sobre o Use a Cabeça SQL aqui?]

Criando a tabela para a Lista de Contatos

Temos uma base de dados e agora precisamos armazenar informações nela. Mas, na verdade, nossas informações tem de ir para uma tabela, estruturas usadas para guardar porções individuais dos dados. Para nosso aplicativo, vamos criar uma tabela chamada "People" (pessoas) para armazenar todas as informações para contato:

1 **Adicione uma tabela à base de dados ContactDB**
Clique com o botão direito em Tables (tabelas) no Navegador de Base de Dados e selecione Add New Table (Adicionar Nova Tabela). Será aberta uma janela onde você pode definir as colunas na tabela que acabou de criar.



Agora precisamos adicionar colunas à nossa tabela. Primeiro, vamos adicionar uma coluna chamada ContactID na tabela People, para que cada registro de Contato tenha sua própria identidade única.

2 **Adicione uma coluna ContactID à tabela People.**
Digite "ContactID" no campo Column Name (nome da coluna) e selecione Int na caixa de combinação Data Type. Certifique-se de desmarcar a caixa Allow Nulls (Permitir Nulos).

Finalmente, vamos fazer desta coluna a chave primária de nossa tabela. Selecione a coluna ContactID que acabou de criar e clique no botão Primary Key (Chave Primária). Isto informa à base de dados que cada linha terá um valor único de chave primária.

Este é o botão Primary Key. Uma chave primária ajuda sua base de dados a procurar dados rapidamente.

não existem Perguntas Idiotas

P: O que é mesmo uma coluna?

R: Uma coluna é um campo de uma tabela. Então numa tabela chamada Pessoas, você pode ter uma coluna para Primeiro Nome e outra para Sobrenome. Ela sempre terá um tipo de dado, também, como String (seqüência de caracteres), Date (data) ou Bool (booleano).

P: Por que precisamos desta coluna ContactID?

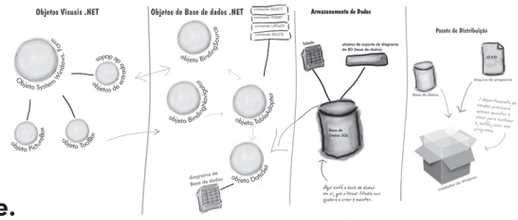
R: Ela ajuda a ter uma única identidade (ID) para cada registro na maioria das tabelas em bases de dados. Já que estamos armazenando informações de contato para pessoas individuais, decidimos criar uma coluna para isso e chamá-la de ContactID.

P: O que aquele Int em Data Type significa?

R: O tipo de dado (data type) informa à base de dados qual tipo de informação ela deve esperar de uma coluna. Int significa apenas um número inteiro. Então a coluna ContactID terá números inteiros nela.

P: Tem muitos detalhes aí. Eu deveria entender tudo isto?

R: Não, tudo bem se você não entender tudo agora. Concentre-se nos passos básicos. Gastaremos muito mais tempo com base de dados nos últimos capítulos. Se você está curioso para saber mais agora, pode sempre ler o ***Use a Cabeça SQL*** junto com este livro.

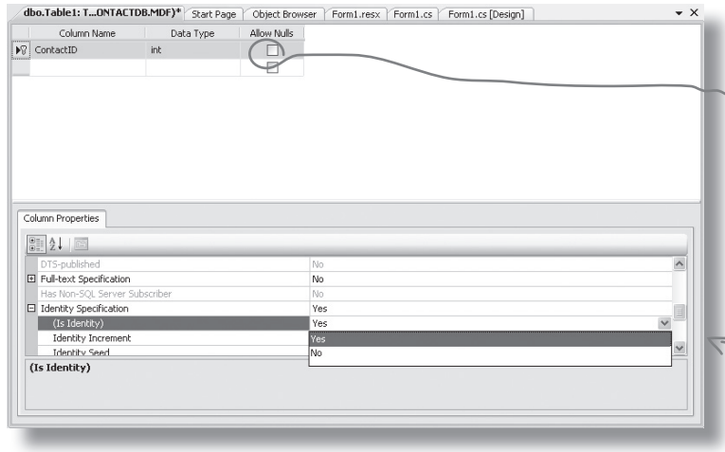


Você está aqui

3 **Mande a base de dados gerar identidades automaticamente.** Já que ContactID é um número que serve para a base de dados e não para nossos usuários, podemos mandar nossa base lidar com a criação e atribuição de identidades automaticamente. Desta forma, não temos que nos preocupar em escrever código para fazer isto.

Nas propriedades abaixo de sua tabela, desça a barra até Identity Specification (Especificação de Identidade), clique no botão + e selecione Yes (sim) perto da propriedade "Is Identity" (É Identidade).

Esta janela é o que você usa para definir sua tabela e os dados que ela armazenará.



É importante que você deixe este controle desmarcado. Como a chave primária é a forma principal pela qual seu programa localizará registros, ela sempre precisa ter um valor.

Isto fará com que o campo ContactID se atualize automaticamente sempre que um novo registro for adicionado.

Os espaços em branco no cartão de contato são colunas na tabela People

Com uma chave primária para a tabela, é preciso definir todos os campos que serão mantidos na base de dados. Cada campo em nosso cartão de contato impresso deve tornar-se uma coluna na tabela People.

Para cada pessoa, queremos armazenar dados: seu nome, empresa, telefone, e-mail, se ela é um cliente da OPC e a data da sua última ligação.

Cada espaço em branco no cartão deve ser mapeado para uma coluna na tabela People.

PODER DO CÉREBRO

Que tipos de problemas poderiam resultar do armazenamento de várias linhas para a mesma pessoa?