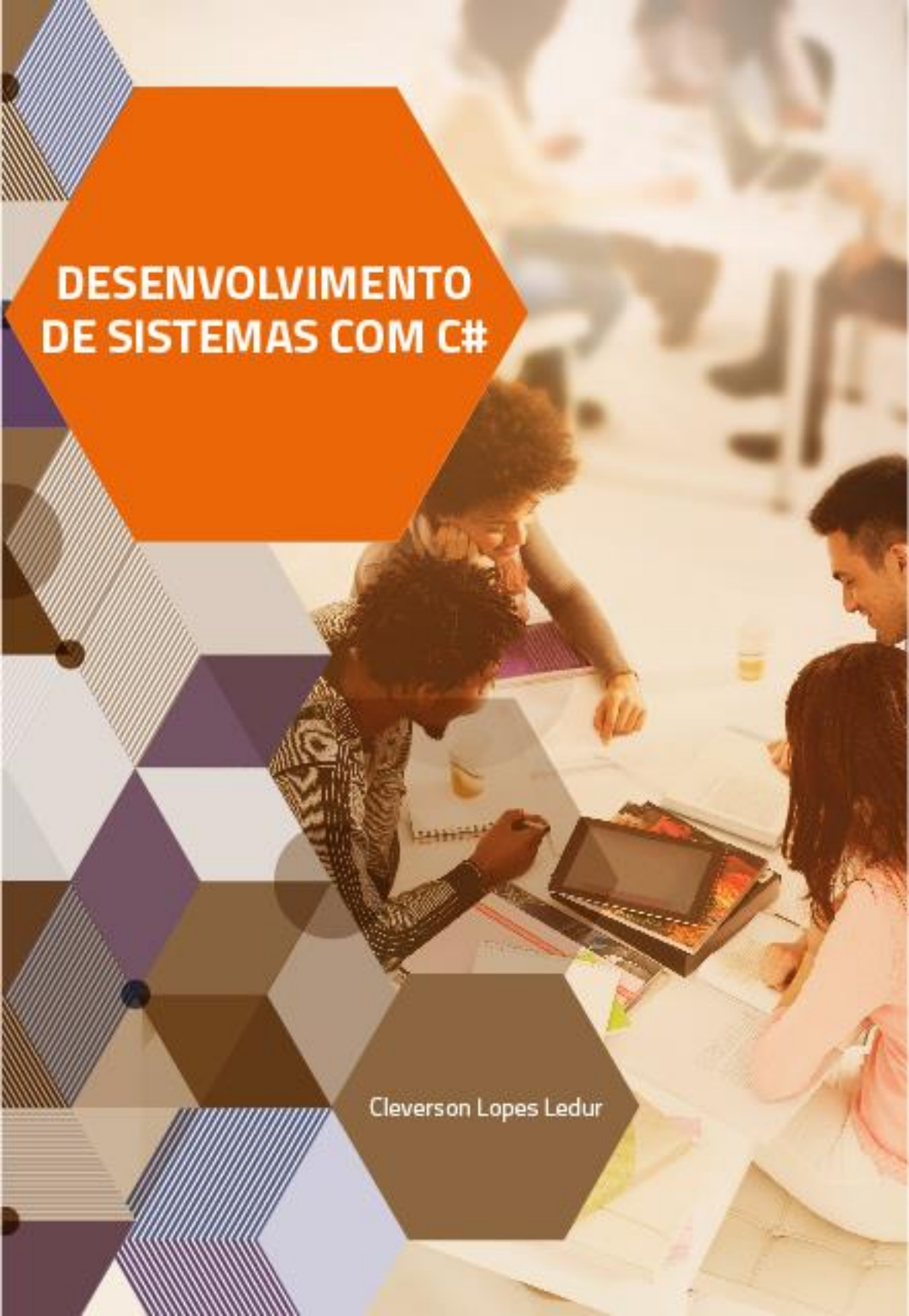
DESENVOLVIMENTO DE SISTEMAS COM C#



Cleverson Lopes Ledur

Revisão técnica:

Jeferson Faleiro Leon
Desenvolvimento de Sistemas
Especialista Formação Pedagógica de Professores
Professor do curso Técnico em informática

L475d Ledur, Cleverson Lopes.
Desenvolvimento de sistemas com #C [recurso eletrônico] / Cleverson Lopes Ledur; [revisão técnica: Jeferson Faleiro Leon]. – Porto Alegre : SAGAH, 2018.

ISBN 978-85-9502-314-7

1. Ciência da computação. 2. Linguagens de programação de computador. I. Título.

CDU 004.43

Catalogação na publicação: Karin Lorien Menoncin CRB-10/2147

Utilizar o Windows Forms para o desenvolvimento de aplicações visuais – I

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Descrever o funcionamento da tecnologia Windows Forms.
- Criar uma aplicação Windows Forms no Visual Studio.
- Identificar a estrutura de um projeto Windows Forms.

Introdução

A criação de sistemas com interfaces gráficas foi um grande desafio da computação no século passado. Com a evolução das linguagens e IDEs, hoje há um grande número de possibilidades para a criação dessas aplicações. O Windows Forms é uma dessas possibilidades que facilitam e aceleram a criação de sistemas de informação. Também conhecido com WinForms, ele é uma biblioteca de classes gráfica (GUI) incluída como parte do Microsoft .NET Framework, fornecendo uma plataforma para escrever aplicativos rich client para computadores de mesa, laptops e tablets PC. Uma aplicação Windows Forms é uma aplicação baseada em eventos suportada pelo .NET Framework da Microsoft. O Windows Forms fornece acesso aos controles comuns nativos da interface do usuário do Windows, envolvendo a API do Windows existente no código gerenciado.

Neste capítulo, você vai conhecer o Windows Forms e a sua forma de funcionamento. Também vai aprender a criar um projeto do tipo Windows Forms e ver como estruturá-lo em camadas.

Aplicações Windows Forms

O Windows Forms é uma biblioteca de classes gráfica (GUI) incluída como parte do Microsoft .NET Framework. Ele fornece uma plataforma para escrever aplicativos rich client para computadores de mesa, laptops e tablets PC. Embora seja visto como um substituto para o anterior, mais complexo e baseado em C ++ Microsoft Foundation Class Library, ele não oferece um paradigma comparável. Assim, só atua como uma plataforma para a camada de interface do usuário em uma solução de várias camadas (MICROSOFT, 2017).

Uma aplicação Windows Forms é uma aplicação baseada em eventos suportada pelo .NET Framework da Microsoft. Ao contrário de um programa em lote, ele passa a maior parte do tempo simplesmente esperando que o usuário faça algo, como preencher uma caixa de texto ou clicar em um botão.

O Windows Forms fornece acesso aos controles comuns nativos da interface do usuário do Windows, envolvendo a API do Windows existente no código gerenciado. Com a ajuda do Windows Forms, o .NET Framework fornece uma abstração mais abrangente acima da API do Win32 do que as fornecidas pelo Visual Basic ou pelo MFC (FREEMAN, 2017).

Todos os elementos visuais na biblioteca de classes do Windows Forms derivam da classe `Controle`. Isso fornece uma funcionalidade mínima de um elemento de interface do usuário, como localização, tamanho, cor, fonte, texto, bem como eventos comuns, como **clique e arraste/solte**. A classe `Controle` também possui suporte de encaixe para permitir que um controle reorganize sua posição sob seu pai. O suporte Microsoft Active Accessibility na classe `Controle` também ajuda os usuários com deficiência a usar o Windows Forms melhor.

Saiba mais

Além de fornecer acesso a controles originais do Windows, como botão, caixa de texto, caixa de seleção e listagem, o Windows Forms adicionou seus próprios controles para hospedagem ActiveX, arranjo de layout, validação e ligação de dados ricos. Esses controles são renderizados usando o GDI +.

Criando uma aplicação Windows Forms

Para criar uma aplicação Windows Forms, você deve primeiramente clicar em **Arquivo > Novo > Projeto**. Alternativamente, você pode utilizar o atalho **Ctlr + Shift + N** para abrir a janela de **Novo Projeto**. Veja, na Figura 1, como realizar essas operações.

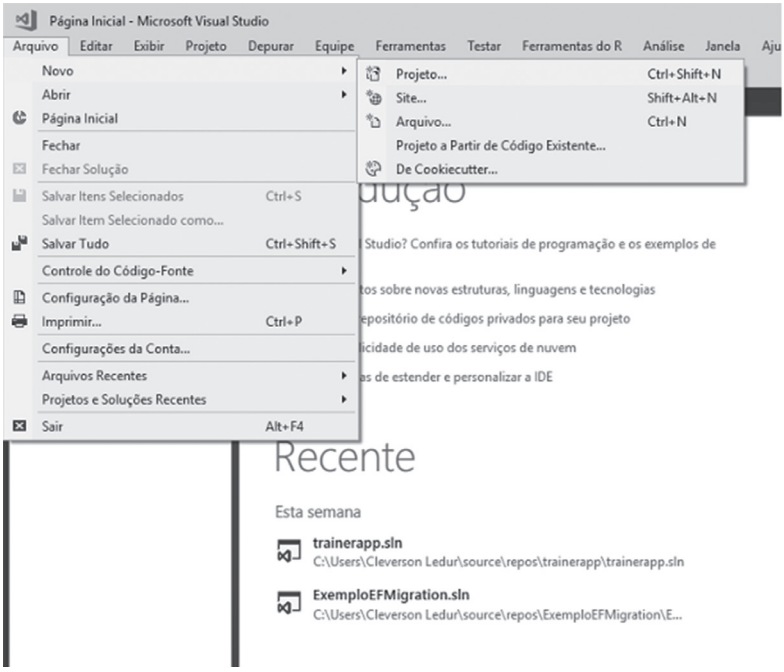


Figura 1. Primeiro passo para criar um novo projeto.

Após a abertura da janela, você poderá selecionar o tipo de projeto que deseja criar. Selecione **Aplicação do Windows Forms (.NET Framework)**. Lembre-se de nomear o projeto no campo **Nome** e depois clicar em **OK** (Figura 2).

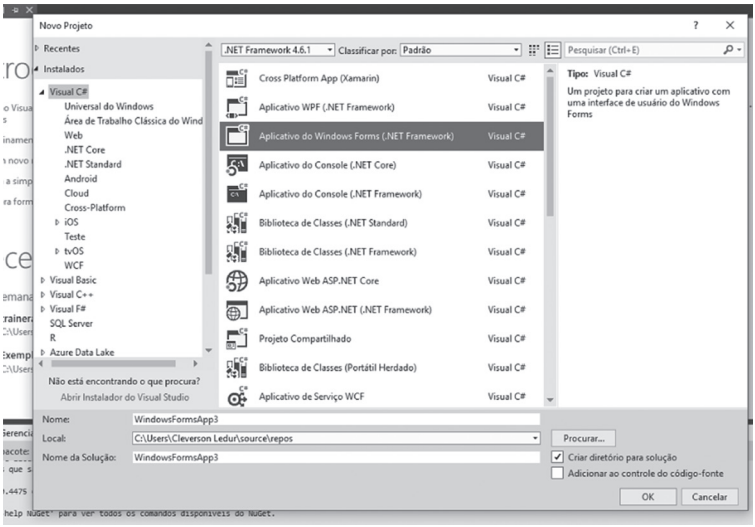


Figura 2. Selecione o tipo de projeto e dê um nome a ele.

O projeto do exemplo recebeu o nome de **SistemaComercial**. Veja, na Figura 3, o local onde você deve inserir o nome do projeto na tela **Novo Projeto**.

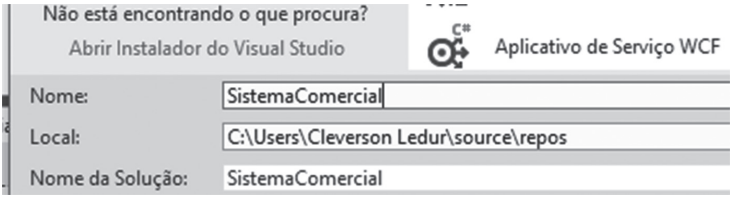


Figura 3. Local para inserção do nome do projeto.

Logo após clicar em **OK**, você precisará aguardar alguns instantes até que o Visual Studio crie os elementos iniciais do projeto. Dessa forma, assim que o Visual Studio finalizar essa etapa, será apresentada uma tela inicial com um form já criado. Esse form inicial é criado e chamado pelo método `Main()`. Você pode criar quantos forms desejar e chamá-los de acordo com a necessidade do sistema. Na Figura 4, você pode ver o primeiro formulário do projeto criado pelo Visual Studio.

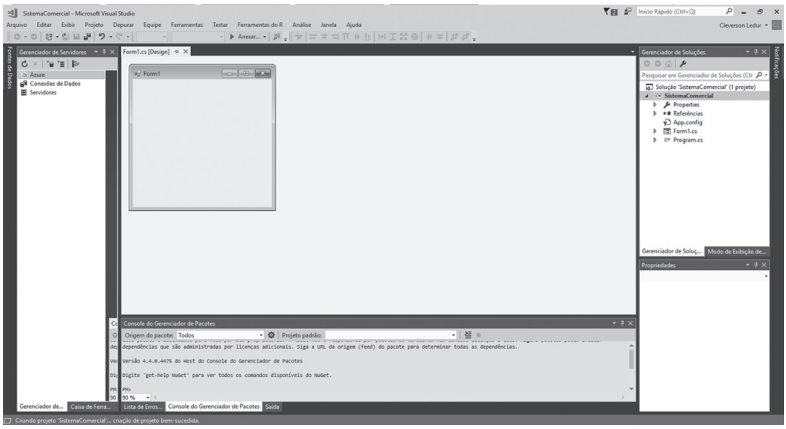


Figura 4. Primeiro formulário do projeto.

No gerenciador de soluções, você pode verificar a estrutura inicial que o projeto possui. Na Figura 5, você pode ver que há uma solução e, dentro dela, o projeto **SistemaComercial**. Na próxima seção, você vai ver como organizar melhor o seu projeto utilizando camadas e dividindo as responsabilidades.

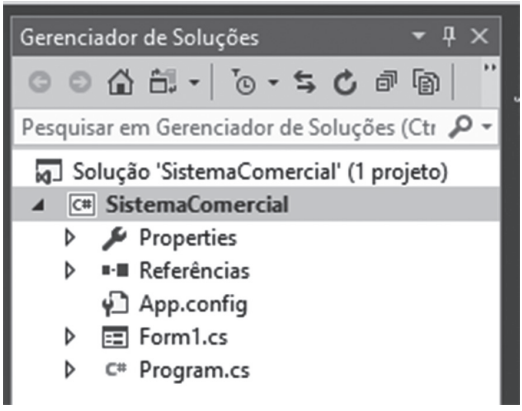


Figura 5. Estrutura inicial do projeto.

Além dos formulários, você precisa de outros elementos para construir uma aplicação, como botões, labels e caixas de inserção de dados. A caixa

de ferramentas exibe ícones para controles e outros itens que você pode adicionar a projetos do Visual Studio. Para abrir a caixa de ferramentas, clique em **Caixa de Ferramentas** na exibição **Menu**. Você pode encaixar a caixa de ferramentas e também pode mantê-la aberta ou defini-la como auto-oculta.

Veja, na Figura 6, como exibir a caixa de ferramentas.

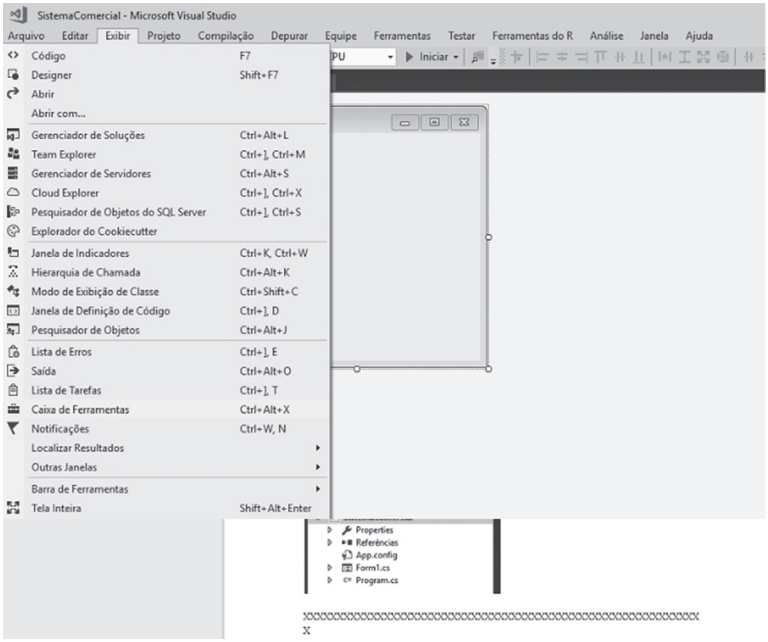


Figura 6. Exibição da caixa de ferramentas.

Os elementos que estão dentro da caixa de ferramentas podem ser arrastados para um modo de exibição de design ou colados em um editor de código. Qualquer ação adiciona o código fundamental para criar uma instância do item no arquivo de projeto ativo.

A caixa de ferramentas exibe apenas os itens que são apropriados para o tipo de arquivo em que você está trabalhando (Figura 7). Você pode pesquisar na caixa de ferramentas para filtrar ainda mais os itens que aparecem. Se o projeto requer um controle que não é compatível com o perfil de cliente, você pode definir seu projeto para a estrutura inteira, editando as propriedades dele (DEITEL; DEITEL, 2016).

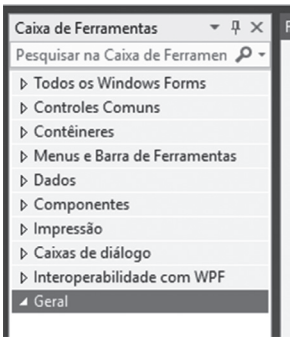


Figura 7. Na caixa de ferramentas, você vê apenas os itens que são apropriados para o tipo de arquivo em que você está trabalhando.

Então, você pode simplesmente arrastar um item da caixa de ferramentas para o form. Na Figura 8, você pode ver como é simples adicionar um novo elemento ao form. Faça um teste utilizando um button (que cria um botão no form).

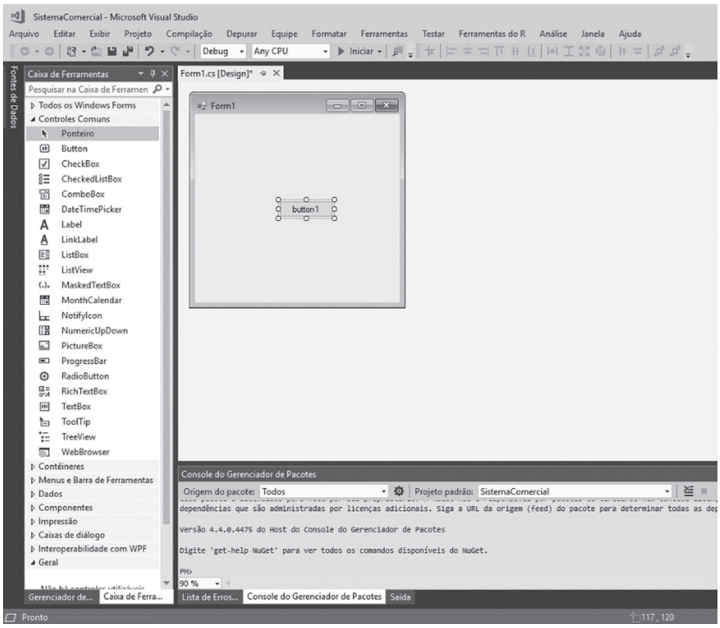


Figura 8. Como adicionar um novo elemento ao form.

Quando você trabalha com Windows Forms, além de mexer na caixa de ferramentas, constantemente tem de utilizar e alterar as propriedades dos elementos. Para isso, deve utilizar o componente **Propriedades** do Visual Studio. Nele, pode modificar as características visuais, comportamentais e estruturais dos seus elementos.

Fique atento

O componente **Propriedades** fica localizado no canto inferior direito do Visual Studio por padrão, mas pode ser movido para outra posição, de acordo com a sua preferência.

Na Figura 9, você pode verificar o componente **Propriedades** exibindo as propriedades de um elemento **button1**. O controle apresenta as propriedades do elemento que está selecionado atualmente.

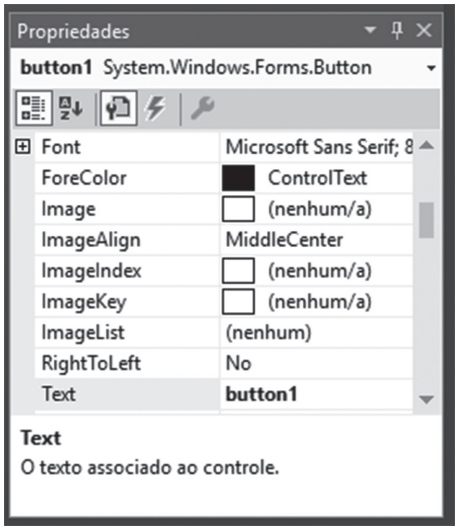


Figura 9. O componente **Propriedades** do Visual Studio.

Estrutura da aplicação

Quando você cria uma nova aplicação, ela apresenta uma estrutura inicial em que todos os arquivos ficam dentro do projeto Windows Forms. O ideal, em uma aplicação mais robusta e complexa, é que as responsabilidades sejam divididas em camadas. Você vai ver como fazer isso nesta seção. Observe, no exemplo a seguir, o método `main()` criado para uma aplicação automaticamente pelo Visual Studio durante a criação de um novo projeto.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace SistemaComercial
{
    static class Program
    {
        /// <summary>
        /// Ponto de entrada principal para o aplicativo.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

Note que há uma classe `Application` que possui métodos que são chamados de acordo com o que é necessário para rodar a aplicação. A chamada final, `Application.Run(new Form1())`, cria uma nova instância do form principal. Quando a instância é criada, caso o form esteja como visível igual a verdadeiro, ele será exibido. Você pode alterar o método `Main` de acordo com o necessário para criar uma aplicação.

Além disso, o ideal em projetos em que você pretende escalar é criar camadas e dividir as responsabilidades. Para isso, você pode criar bibliotecas de classes. Você deve clicar com o botão direito no nome da solução e depois em **Adicionar > Novo Projeto**. Veja como fazer isso na Figura 10.

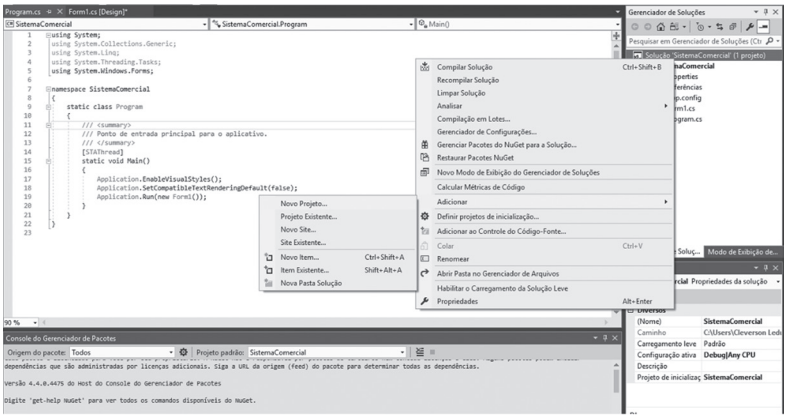


Figura 10. O primeiro passo para criar bibliotecas de classes.

Logo, será exibida uma tela em que você deve selecionar **Biblioteca de Classes (.NET Framework)**. Você pode nomeá-la *DAL* (*Data Access Layer*). Observe esse procedimento na Figura 11.

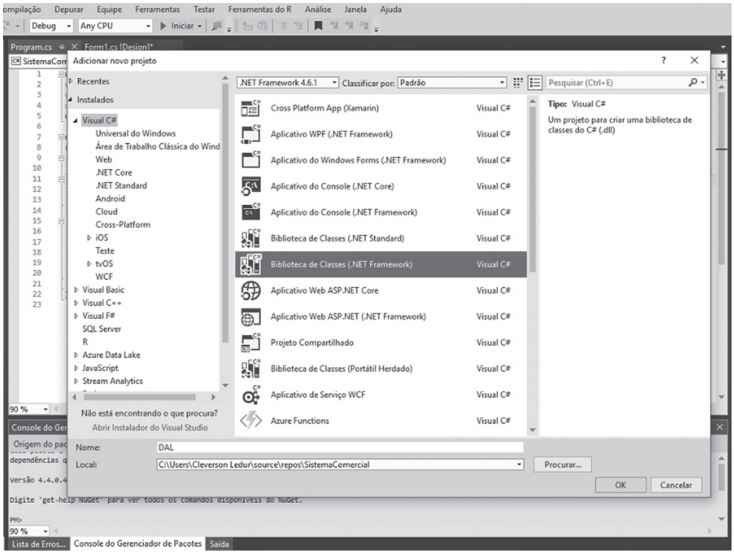


Figura 11. Crie uma biblioteca de classes e dê um nome a ela.

Após, você deve repetir o procedimento para criar também uma camada DAL (*Data Access Layer*), como você pode ver na Figura 12. Nessa camada, ficará toda a lógica de regras de negócio.

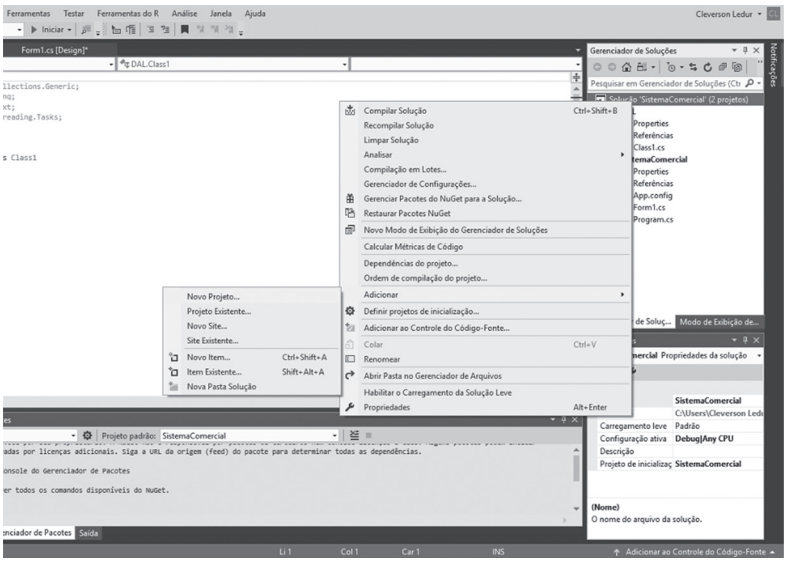


Figura 12. Repita o processo para criar uma camada DAL.

Ao final, você terá as camadas em bibliotecas, como você pode ver na Figura 13. Você também pode utilizar pastas dentro do projeto para separar as responsabilidades.

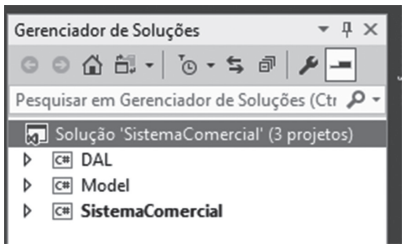


Figura 13. Camadas em bibliotecas.

Neste capítulo, você viu a criação e a organização de um projeto do tipo Windows Forms. O uso de Windows Forms permite criar aplicações de uma maneira muito rápida e fácil, fazendo com que o desenvolvimento de software torne-se mais produtivo.

Referências

DEITEL, P.; DEITEL, H. *Visual C#*: how to program. 6th ed. London: Pearson, 2016.

FREEMAN, A. *Essential C# features*: Pro ASP.NET Core MVC 2. 7th ed. Berkeley: Apress, 2017.

MICROSOFT. *Windows forms*. Redmond, 2017. Disponível em: <https://msdn.microsoft.com/pt-br/library/dd30h2yb(v=vs.110).aspx>. Acesso em: 02 nov. 2017.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo: