Desenvolvimento - C#

Guia prático para o desenvolvimento de Aplicações C# em Camadas - parte 8

Este é o oitavo artigo da série onde vamos demonstrar passo-a-passo a construção de uma aplicação .Net utilizando o conceito de desenvolvimento em camadas.

por Carlos Camacho

8 – User Interface – Formulário de Vendas

Neste artigo vamos criar o Formulário de Vendas do nosso projeto.

- Abra o Microsoft Visual Studio;

- Clique em **File** > **Open** > **Project/Solution...**

- Na janela *Open Project*, selecione o arquivo da nossa Solution (**C:\Loja\Modelos\Loja.sln**) e clique em **Open** para abrir a solução.

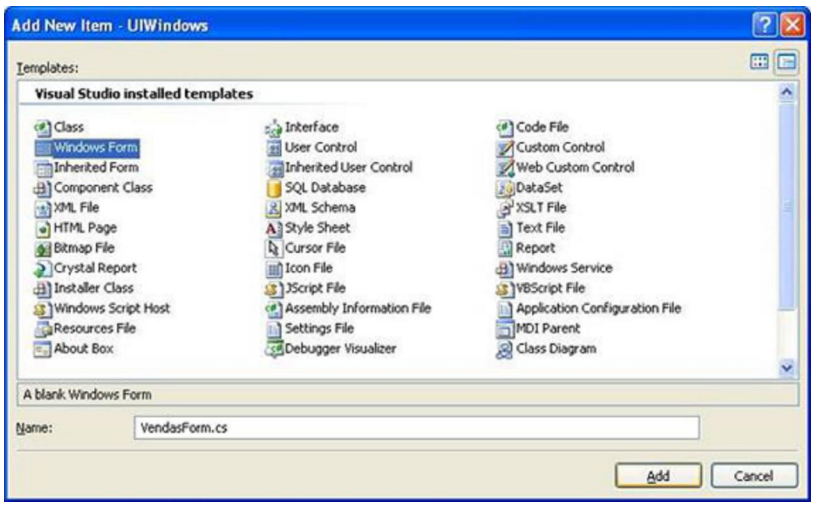
Vamos adicionar um formulário chamado VendasForm.cs.

- Clique com o botão direito no projeto UIWindows e escolha **Add** > **Windows Form...**

- Na janela "Add New Item" vamos informar os seguintes dados:

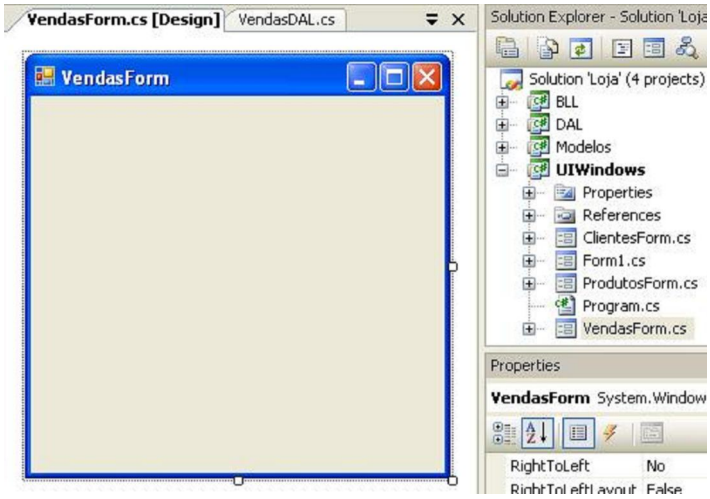
Template: **Windows Form**

Name: **VendasForm**



- Clique em **Add** para adicionar o formulário.

Será exibido o formulário VendasForm:



- Altere a propriedade **Size** para **470; 200**

Vamos criar 7 objetos neste formulário. Estes objetos podem ser arrastados para o formulário a partir da "Common Controls" da Toolbox.

- Arraste e solte 3 objetos do tipo Label para o nosso formulário de vendas e configure-os com as seguintes características:

Name: **clienteLabel**
Text: **Cliente:**
Location: **35; 37**

Name: **produtoLabel**
Text: **Produto:**
Location: **35; 64**

Name: **quantidadeLabel**
Text: **Quantidade:**
Location: **35; 95**

- Arraste e solte 2 objetos do tipo combobox para o nosso formulário de vendas e defina as seguintes propriedades:

Name: **clienteComboBox**
DisplayMember: **Nome**
DropDownStyle: **DropDownList**
DropDownWidth: **300**
Location: **106; 34**
Size: **300; 21**
ValueMember: **Codigo**

Name: **produtoComboBox**
DisplayMember: **Nome**
DropDownStyle: **DropDownList**
DropDownWidth: **300**

Location: **106; 64**
Size: **300; 21**
ValueMember: **Codigo**

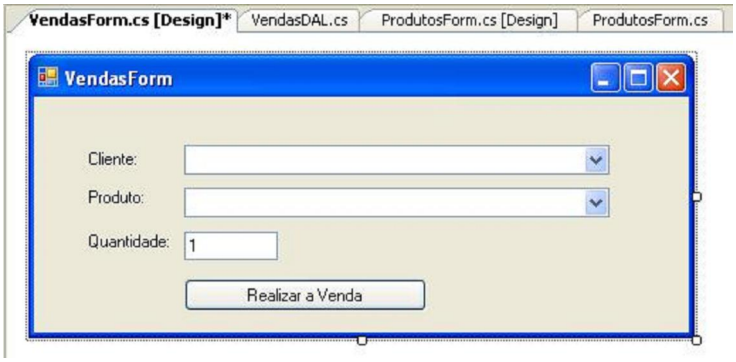
- Arraste e solte um objeto do tipo TextBox e defina as seguintes propriedades:

Name: **quantidadeTextBox**
Location: **106; 95**
Size: **66; 20**
Text: **1**

- Arraste e solte um objeto do tipo Button e defina as seguintes propriedades:

Name: **incluirVendaButton**
Location: **106; 128**
Size: **171; 23**
Text: **Realizar a Venda**

Neste momento, nosso formulário de vendas estará com esta aparência:



Vamos carregar dados nos dois objetos combobox sempre que o formulário de vendas for carregado.

- Dê um duplo clique em uma área livre do formulário para exibir o método VendasForm_Load().

- Copie e cole o código abaixo:

```
VendasBLL obj = new VendasBLL();

clienteComboBox.DataSource = obj.ListaDeClientes;

produtoComboBox = obj.ListaDeProdutos;
```

Observação: Devemos usar

```
using Loja.BLL;

using Loja.DAL;

using Loja.Modelos;
```

no início do code behind do VendasForm.cs

- No modo de design do VendasForm, dê um duplo clique no botão “Realizar a Venda”;

- Copie e cole o código abaixo:

```
try

{
```

```
            VendaInformation venda = new VendaInformation();
            venda.Quantidade = int.Parse(quantidadeTextBox.Text);
            venda.CodigoCliente = (int)clienteComboBox.SelectedValue;
            venda.CodigoProduto = (int)produtoComboBox.SelectedValue;
            venda.Data = DateTime.Now;
            venda.Faturado = false;
            VendasBLL obj = new VendasBLL();
            obj.Incluir(venda);
            MessageBox.Show("A venda foi realizada com sucesso!");
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
```

O code behind completo do nosso VendasForm.cs ficou assim:

```
/*
 * File...: VendasForm.cs
 * Author.: Carlos Camacho - www.camachojunior.com.br
 * Content: Formulário de Vendas
 * Subject: Guia prático para o desenvolvimento de Aplicações C# em Camadas
*/
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Loja.BLL;
using Loja.DAL;
using Loja.Modelos;
namespace Loja.UIWindows
{
    public partial class VendasForm : Form
    {
        public VendasForm()
        {
            InitializeComponent();
        }
```

```
        private void VendasForm_Load(object sender, EventArgs e)
        {
            VendasBLL obj = new VendasBLL();
            clienteComboBox.DataSource = obj.ListaDeClientes;
            produtoComboBox.DataSource = obj.ListaDeProdutos;
        }
        private void incluirVendaButton_Click(object sender, EventArgs e)
        {
            try
            {
                VendaInformation venda = new VendaInformation();
                venda.Quantidade = int.Parse(quantidadeTextBox.Text);
                venda.CodigoCliente = (int)clienteComboBox.SelectedValue;
                venda.CodigoProduto = (int)produtoComboBox.SelectedValue;
                venda.Data = DateTime.Now;
                venda.Faturado = false;
                VendasBLL obj = new VendasBLL();
                obj.Incluir(venda);
                MessageBox.Show("A venda foi realizada com sucesso!");
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
    }
}
```

- Abra o formulário principal Form1 no modo de design e dê um duplo clique na opção Vendas do menu para codificarmos a chamada para este formulário com o código abaixo:

```
        VendasForm obj = new VendasForm();
        obj.MdiParent = this;
        obj.Show();
```

Acabamos de implementar o Formulário de Vendas do nosso projeto Loja .Net.

Execute o projeto e teste as funcionalidades que implementamos neste artigo.

Perceba que a quantidade de produtos adquirida por um cliente na venda deverá se refletir automaticamente na quantidade de produtos em estoque. Após realizar uma venda, abra o formulário de produtos e veja que o estoque foi devidamente atualizado.

No próximo e último artigo desta série vamos implementar o formulário de Produtos em Falta.

Até o próximo artigo!

Carlos Camacho - Carlos Olavo de Azevedo Camacho Júnior é mestrando em Tecnologias da Inteligência e Design Digital pela Pontifícia Universidade Católica de São Paulo PUCSP. Pós-graduado em Análise e Projeto de Sistemas pela Universidade Paulista UNIP. Bacharel em Ciência da Computação pela Universidade Paulista UNIP e possui Licenciatura Plena em Matemática pelas Faculdades Oswaldo Cruz.
MCP .Net, MCP SQL Server, Carlos Camacho leciona disciplinas técnicas na área de Ciências Exatas e é Consultor em Tecnologia da Informação para Instituições Financeiras.

Leia também

Lambda Expressions x SQL: Comparando a sintaxe de consultas comuns
C#

List: trabalhando com listas genéricas em C#
C#

Criando Gráficos usando C# e API do Google
C#

Imprimindo um Panel com C#
C#

Consumindo um Web API em C#
C#