



Log In / Cadastre-se

Pesquisar



HOME DESENVOLVIMENTO FRONT-END BANCO DE DADOS EM DESTAQUE TODOS

PUBLIQUE

Desenvolvimento - C#

Guia prático para o desenvolvimento de Aplicações C# em Camadas - parte 9 (Final)

Este é o nono e último 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



9 – User Interface – Formulário de Produtos em Falta (FINAL)

Este é o último artigo da série Guia Prático para o desenvolvimento de Aplicações C# em Camadas. Nele vamos criar o Formulário de Produtos em Falta.

- Abra o Microsoft Visual Studio;

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

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

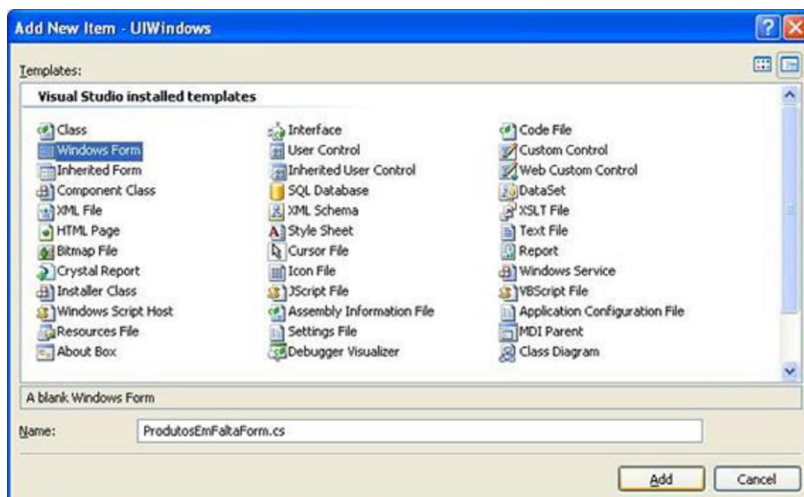
Vamos adicionar um formulário chamado *ProdutosEmFaltaForm.cs*.

- 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: **ProdutosEmFaltaForm**



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

Será exibido o formulário *ProdutosEmFaltaForm*:



Publicidade

Compuware dynaTrace
É muito simples monitorar **100%** das transações do seu App

Experimente a Versão FREE TRIAL

Compuware **APM**

REVISTAS DEVMEDIA



.net Mag 116



Easy .net mag 37

VER TODAS

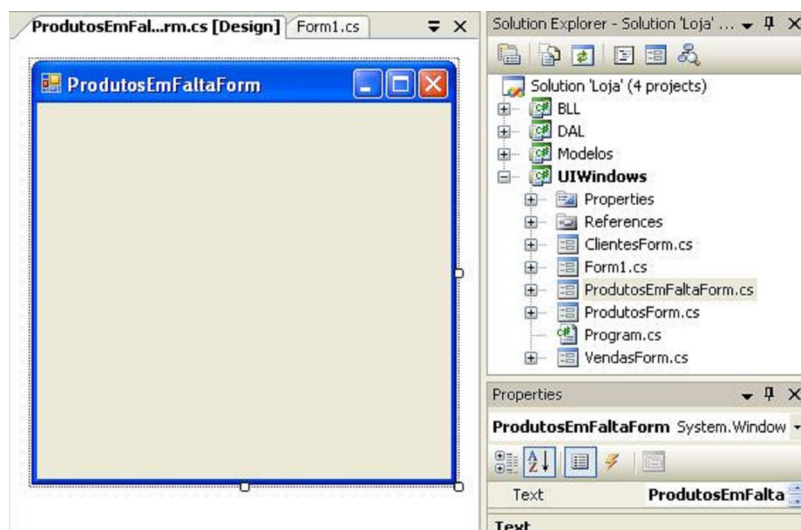
ASSINE

TOP 10 - ARTIGOS

TOP 10 - AUTORES

- 1 HTML Básico
- 2 Menu em CSS - Menu dropdown horizontal com HTML5 e CSS3
- 3 Comandos básicos em SQL - insert, update, delete e select
- 4 Criando um sistema de cadastro e login com PHP e MySQL
- 5 Código para background HTML e CSS
- 6 Copiando dados com o Robocopy
- 7 Trabalhando com Div em HTML
- 8 Excel: Como verificar se existe valores duplicados
- 9 Botão com CSS 3: Como criar um botão sem imagens
- 10 HTML Avançado

VER TODOS



- Altere a propriedade **Size** para **510; 310**

Vamos criar 1 label e 1 DataGridView.

- Arraste e solte 1 Label para o nosso formulário de produtos em falta e configure-o com as seguintes características:

Name: produtosEmFaltaLabel
Location: 9; 21
Text: Relatório de Produtos em Falta:

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

Name: produtosEmFaltaDataGridView
Anchor: Top, Bottom, Left, Right
Location: 12; 58
Size: 474; 206

Neste momento, nosso formulário de produtos em falta estará com esta aparência:



Vamos carregar a relação de produtos em falta sempre que o formulário for carregado.

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

- Copie e cole o código abaixo:

```
ProdutosBLL produto = new ProdutosBLL();

produtosEmFaltaDataGridView.DataSource = produto.ProdutosEmFalta();
```

Não se esqueça de usar

```
using Loja.BLL;
```

```
using Loja.DAL;
```

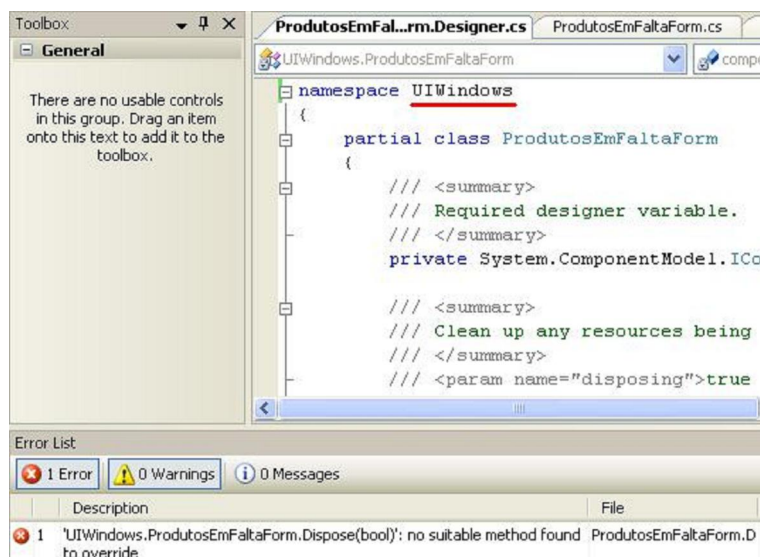
```
using Loja.Modelos;
```

no início do code behind do ProdutosEmFaltaForm.cs

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

```
/*  
 * File....: ProdutosEmFaltaForm.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 ProdutosEmFaltaForm : Form  
    {  
        public ProdutosEmFaltaForm()  
        {  
            InitializeComponent();  
        }  
  
        private void ProdutosEmFaltaForm_Load(object sender, EventArgs e)  
        {  
            ProdutosBLL produto = new ProdutosBLL();  
  
            produtosEmFaltaDataGridView.DataSource = produto.ProdutosEmFalta();  
        }  
    }  
}
```

- Compile o projeto para nos certificarmos de que tudo foi implementado corretamente.



Se ocorrer o erro *"no suitable method found to override"* como na imagem acima, corrija o namespace do código colocando **"Loja."** Antes do nome do projeto UIWindows.

- Compile o projeto novamente para ter certeza de que tudo está ok.

Agora vamos fazer a chamada para este formulário a partir do menu principal do projeto Loja .Net.

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

```
ProdutosEmFaltaForm obj = new ProdutosEmFaltaForm();
```

```
obj.MdiParent = this;
```

```
obj.Show();
```

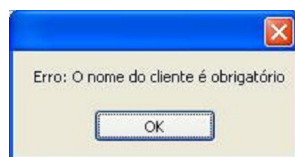
Acabamos de implementar o Formulário de Produtos em Falta do nosso projeto Loja .Net.

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

Realize vendas até que um determinado produto fique com menos de 10 unidades em estoque. Esse produto deverá aparecer no relatório de produtos em falta.

Testando as Regras de Negócio

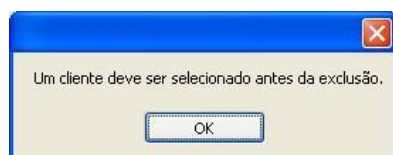
Na camada BLL implementamos algumas regras de negócio. Tente incluir um cliente com o campo nome em branco e você receberá o alerta a seguir:



Após clicar em OK, a inclusão não é permitida e você estará de volta ao formulário de inclusão.

Outra regra **implícita** que colocamos, ou seja, não está visível ao usuário, é que o campo e-mail é armazenado em letras minúsculas. Experimente incluir um cliente escrevendo o e-mail em letras maiúsculas e você verá que a nossa camada BLL se encarregará de armazenar esse campo em letras minúsculas.

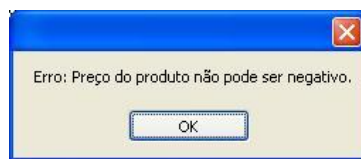
Se você estiver com o cursor numa linha do DataGridView que não tenha registro e clicar no botão Excluir, receberá o aviso:



Um alerta semelhante será exibido ao clicar no botão Alterar, pois definimos como regra que um registro

precisa estar selecionado para realizarmos estas operações.

Mesmo que uma informação seja fornecida, implementamos algumas especificações que devem ser respeitadas no nosso projeto. Por exemplo, se tentarmos cadastrar um valor negativo no preço de um produto, receberemos o aviso:



Dessa forma, vemos que a nossa camada de regras de negócio vai além da tarefa de manter a integridade dos dados, evitando o armazenamento de campos em branco. Ela também nos permite garantir que as regras especificadas para o projeto sejam respeitadas, como no exemplo da inclusão do preço do produto que não pode ser um número negativo.

Estrutura de Arquivos

Na pasta C:\Loja\UIWindows\UIWindows\bin\Debug você encontrará os arquivos:

- Modelos.dll
- DAL.dll
- BLL.dll
- UIWindows.exe

Os arquivos Modelos.dll e DAL.dll referem-se a camada de Banco de dados;

O arquivo BLL.dll refere-se a camada de regras de negócio;

E o arquivo UIWindows.exe é a interface com o usuário que roda no MS-Windows.

Dicas:

- 1) Se você copiar somente esses 4 arquivos em qualquer pasta do microcomputador de um usuário, bastará ele executar o aplicativo UIWindows.exe para iniciar o sistema Loja.Net e utilizar todas as funcionalidades.
- 2) Se você precisar desenvolver uma interface para a Web, por exemplo, poderá reutilizar as camadas de Banco de dados e de regras de negócio sem a necessidade de qualquer alteração.

Vantagens do modelo de desenvolvimento em Camadas

Agora imagine que você é o Gerente de Projeto do sistema Loja .Net e recebe o seguinte e-mail de um de seus clientes:

Prezado Sr.

Como usuário do Sistema Loja .Net desenvolvido pela sua empresa, informo que precisamos que o Sr. nos envie uma atualização para atender as nossas novas necessidades.

Nossa empresa está em contínuo crescimento e em breve contrataremos uma empresa de Call Center que oferecerá promoções especiais aos nossos clientes. Dessa forma, precisamos fazer com que não seja permitido o cadastramento de clientes sem o número de telefone, pois essa informação será muito importante para o nosso sucesso.

Por favor nos notifique assim que a nova versão do sistema estiver disponível para testes.

Atenciosamente,

Carlos Camacho

Analista de Sistemas

Banco Nossa Caixa S.A.

Departamento de Tecnologia da Informação

carlos.camacho@nossacaixa.com.br

Algumas perguntas:

- Qual camada do Sistema Loja .Net você vai precisar alterar para atender a solicitação do cliente?

Isso mesmo, você vai precisar alterar somente a camada BLL.

- Depois que você realizar as atualizações do Sistema, precisará disponibilizar os 4 arquivos que compõem o sistema para o cliente?

Não, é só você enviar o arquivo referente a camada que foi atualizada. Neste caso, bastaria enviar o arquivo BLL.dll com instruções para que o cliente substitua o arquivo atual por este novo arquivo para realizar a atualização do sistema.

Conclusões

Nesta série de artigos vimos como construir uma aplicação .Net utilizando o conceito de desenvolvimento em camadas.

Neste projeto você teve a oportunidade de praticar todas as etapas da implementação do desenvolvimento de uma aplicação .Net, adquirindo conhecimentos sobre:

- Comunicação entre as camadas;
- Vantagens do modelo de desenvolvimento em camadas;
- Controle de transações do banco de dados com o ADO .Net;
- Construção de uma aplicação para Windows.

Implementou também as três camadas:

- Camada de acesso à dados ou Data Access Layer (DAL);
- Camada de regras de negócio ou Business Logic Layer (BLL); e
- Camada de interface do usuário ou User Interface (UI).

Você adquiriu habilidades na construção de aplicativos capazes de reaproveitar a camada de acesso à dados e a camada de regras de negócio. Se por um acaso você estiver trabalhando em um projeto complexo, com uma grande infra-estrutura de banco de dados e diversos formulários, certamente se sentirá seguro nas suas atribuições. Se você for o responsável pela manutenção do sistema, seja para a alteração ou criação de novos formulários, terá confiança no seu trabalho pois entende como as camadas se relacionam e tem know-how no processo de criação dos objetos mais comuns de um formulário.

É importante lembrar que para ter sucesso na área de TI você deve manter-se sempre atualizado com as tecnologias mais recentes. Concordo que não é uma tarefa fácil pois em um curto espaço de tempo surgem inovações incríveis (...). Por outro lado, investir no constante aperfeiçoamento é um mal necessário para quem almeja o sucesso.

Muito obrigado por ter acompanhado essa série de artigos desde o início. Desejo que os conhecimentos adquiridos sejam úteis no seu dia-a-dia.

Um forte abraço,

Carlos Camacho

www.camachojunior.com.br



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#

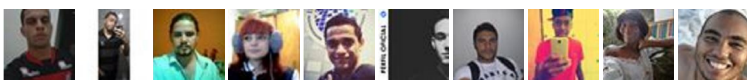
Estamos aqui:    

Linha de Código faz parte do grupo Web-03

[Política de privacidade e de uso](#) | [Anuncie](#) | [Cadastre-se](#) | [Fale conosco](#)

**Linha de Código**
Like

9,981 people like [Linha de Código](#).



Facebook social plugin

© 2014 Linha de Código. Todos os direitos reservados