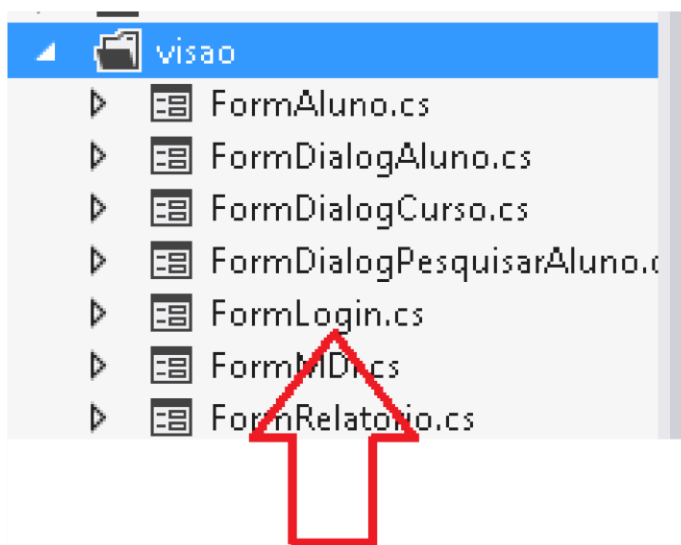


Etec Dr Demétrio Azevedo Junior – Itapeva-SP

BANCO DE DADOS - C SHARP

Sistema de Tela de Login para C#

Crie o seguinte formulário em visão: *FormLogin.cs*



Desenhe os seguintes componentes:

A diagram of a login form titled 'LOGIN DO SISTEMA'. The form has a light blue border and a gray background. It contains the following components:

- A label 'Acesso ao Sistema:' followed by a group box containing:
 - A label 'LOGIN:' followed by a text input field.
 - A label 'SENHA:' followed by a text input field.
 - A button labeled 'CONECTAR'.
 - A button labeled 'SAIR'.

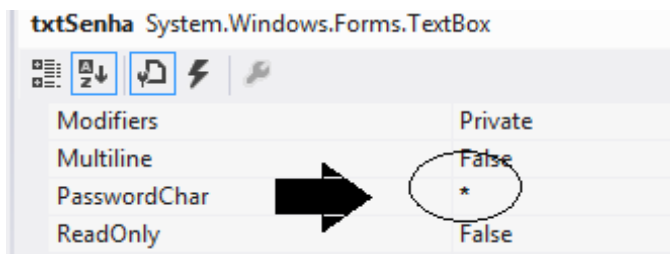
Desativar a propriedade *ControlBox* do formulário para desaparecer com o botão de fechar

Nome dos componentes:

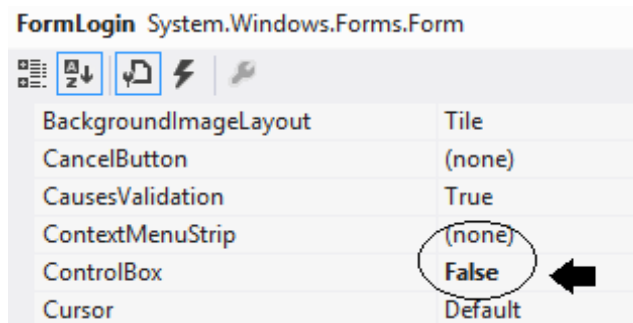
Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
Label label1  
TextBox txtLogin  
GroupBox gbAcesso  
Button btSair  
Button btConectar  
Label label2  
TextBox txtSenha
```

A propriedade *passwordChar* do campo *txtSenha* deve estar preenchida com asterisco.



O formulário deve ter a caixa de controle desativada:



Vamos inserir no banco de dados um usuário, já que ainda não possuímos rotinas de cadastro de usuários em nosso projeto:

```
mysql> insert into usuarios values (1,"admin","123",1);  
Query OK, 1 row affected (0.07 sec)
```

Crie em controle a classe *UsuariosDB*:



Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Codifique a seguinte estrutura na classe:

```
class UsuarioDB
{
    Conexao con = new Conexao("localhost", "biblioteca",
                                "root", "minas");

    public bool validar(string nome, string senha) {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            // conecta no banco de dados
            banco.Database.Connection.ConnectionString = con.open();
            // consulta usando a linguagem LINQ
            var query = (from linha in banco.usuarios
                          where linha.login.Equals(nome) &&
                               linha.senha.Equals(senha)
                          select linha).FirstOrDefault();
            if (query == null) return false;
            return true;
        }
    }
}
```

Codificar o botão conectar do formulário de Login para verificar se o usuário digitou uma senha e usuário correto:

```
private void btConectar_Click(object sender, EventArgs e)
{
    controle.UsuarioDB uDB = new controle.UsuarioDB();
    if (uDB.validar(txtLogin.Text, txtSenha.Text))
    {
        this.Dispose();
    }
    else
        MessageBox.Show("Conexão inválida");
}
```

Criar a opção de encerrar o programa no botão sair:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void btnSair_Click(object sender, EventArgs e)
{
    // encerra o programa
    Environment.Exit(0);
}
```

Crie no evento *load* de *FormMDI* uma chamada a tela de Login:

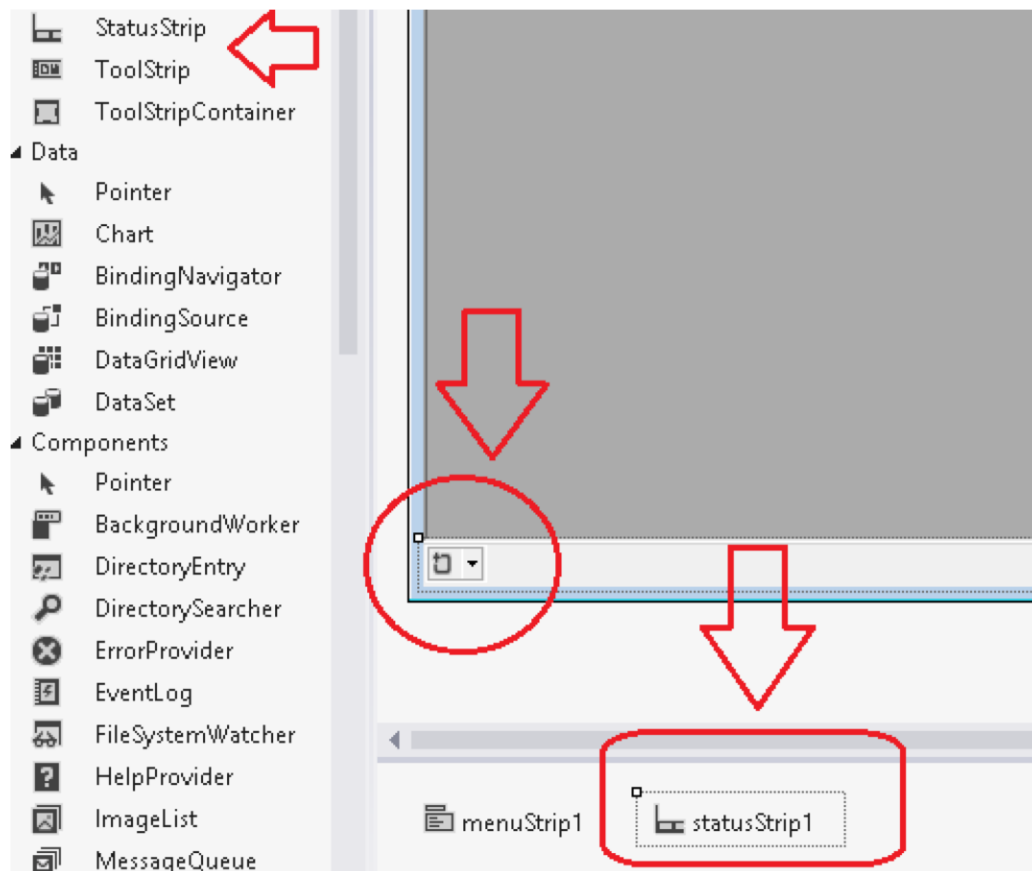
```
private void FormMDI_Load(object sender, EventArgs e)
{
    visao.FormLogin fr = new visao.FormLogin();
    fr.ShowDialog();
}
```

Teste a rotina de conexão:



Sistema de Barra de Status em Formulários MDI

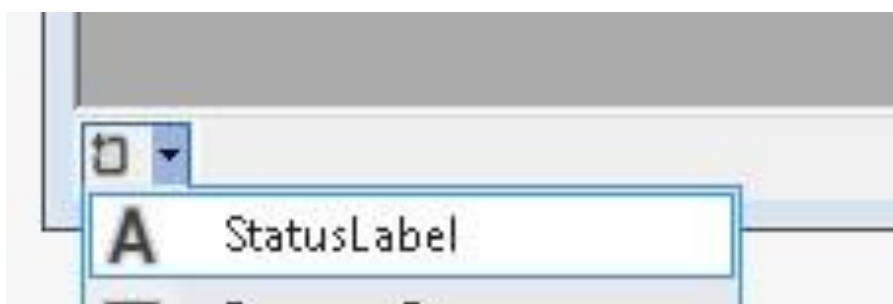
Vamos acrescentar uma barra de status em nosso sistema. Acrescente o seguinte componente *statustrip* em *FormMDI*:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Renomear o componente para *barraStatus*:

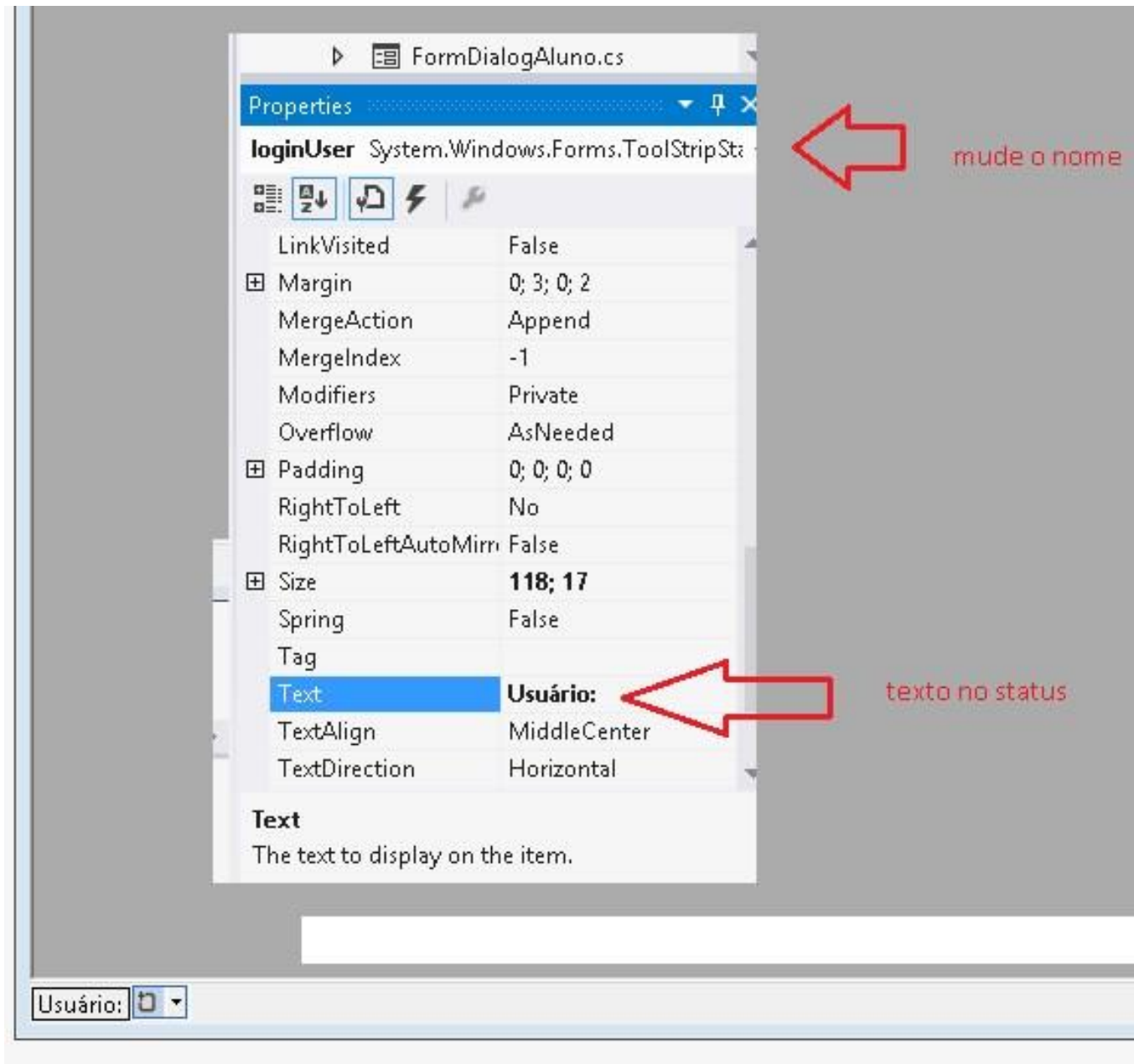


Adicione um status *label* e chame o mesmo de *loginUser*:

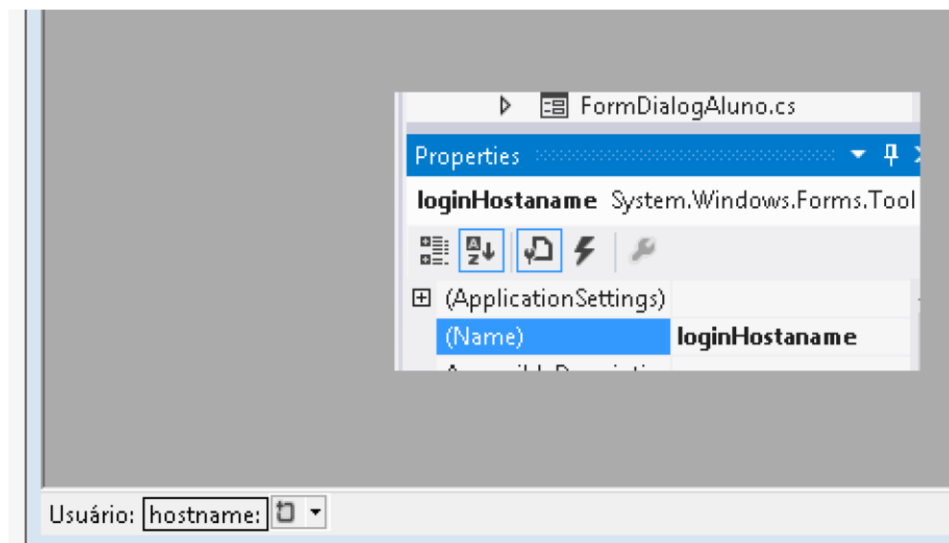


Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Ficará deste jeito:



Crie mais um com o texto *hostname* e nome *loginHostname*:

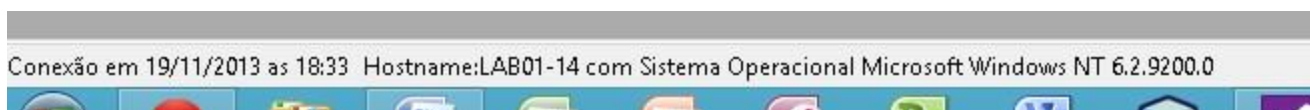
Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Voltemos ao evento *load* do *FormMDI* vamos preencher os dados da barra de status:

```
private void FormMDI_Load(object sender, EventArgs e)
{
    visao.FormLogin fr = new visao.FormLogin();
    fr.ShowDialog();
    loginUser.Text = "Conexão em " + DateTime.Now.ToShortDateString() +
        " as " + DateTime.Now.ToShortTimeString();

    loginHostname.Text = "Hostname:" + Environment.MachineName +
        " com Sistema Operacional " + Environment.OSVersion;
}
```

Resultado da execução:



Ajuste as propriedades acrescente ícones nos status para melhorar a estética:



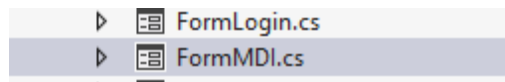
Tendo algo visual como:



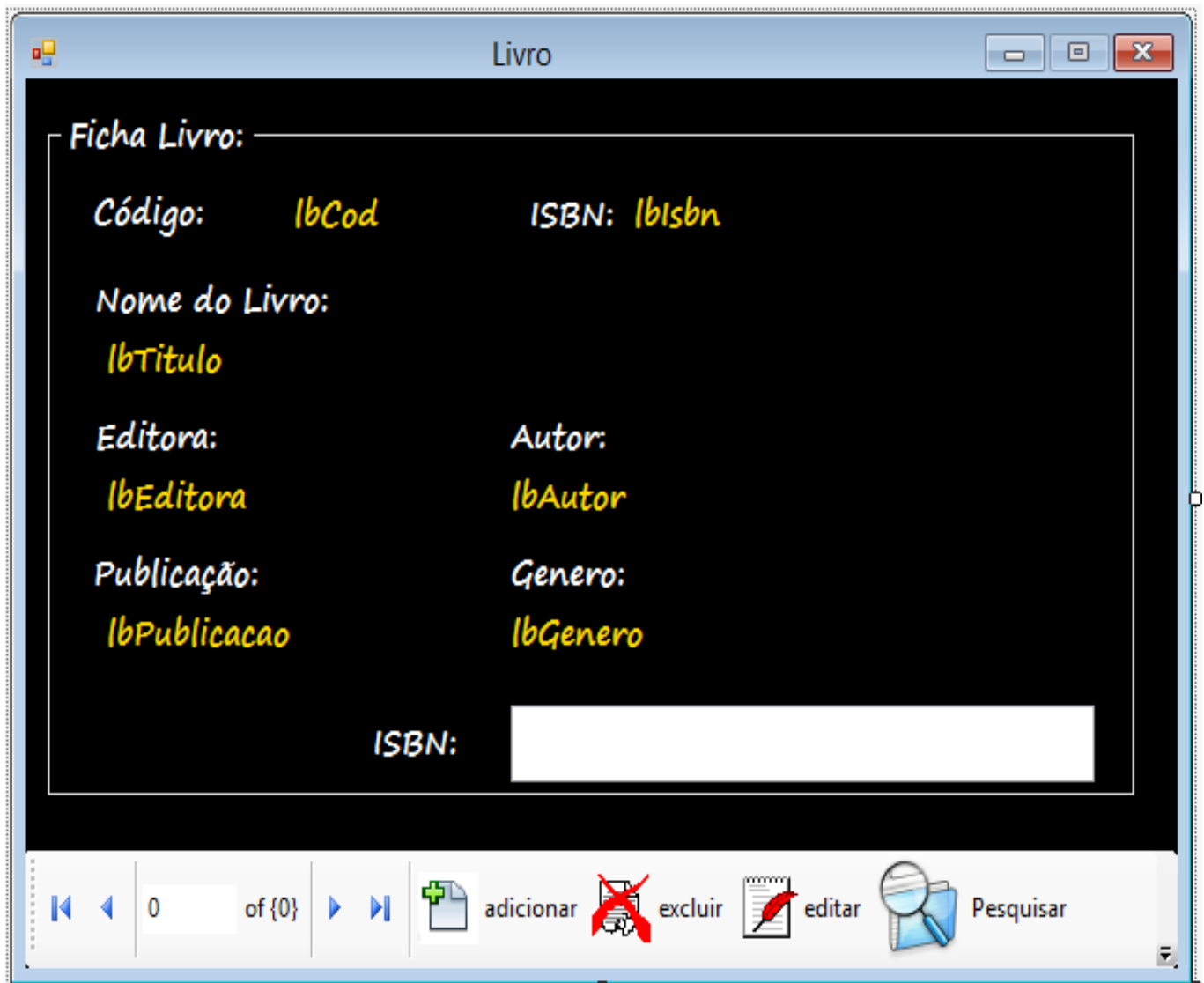
Etec Dr Demétrio Azevedo Junior – Itapeva-SP

DESENVOLVENDO O MÓDULO DE CADASTRO DE LIVROS

Dando continuidade ao projeto biblioteca, vamos desenvolver o módulo de cadastro de livros. Devemos criar o formulário **FormLivro** em nosso projeto na pasta **visao**:



Desenhe no formulário a seguinte estrutura, baseada no formulário aluno já criado:



The screenshot shows a Windows application window titled "Livro". Inside the window, there is a form titled "Ficha Livro:". The form contains several fields and labels:

- Código:** *lbCod* (yellow text)
- ISBN:** *lbIsbn* (yellow text)
- Nome do Livro:** *lbTitulo* (yellow text)
- Editora:** *lbEditora* (yellow text)
- Autor:** *lbAutor* (yellow text)
- Publicação:** *lbPublicacao* (yellow text)
- Genero:** *lbGenero* (yellow text)

At the bottom of the form, there is a label **ISBN:** followed by a text input field.

Below the form, there is a toolbar with the following elements:

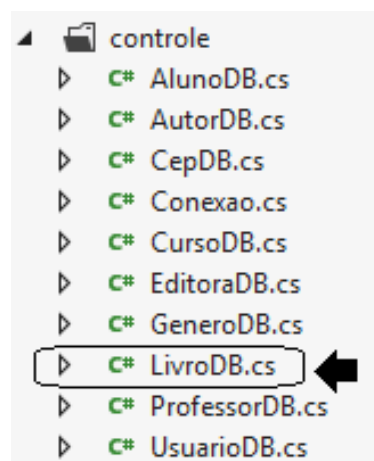
- Navigation buttons: back, forward, and a list of items.
- A status bar showing "0 of {0}" items.
- Buttons: "adicionar" (add), "excluir" (delete, crossed out with a red X), "editar" (edit), and "Pesquisar" (search).

Nome dos campos do formulário:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
.ToolStripButton btAdd;  
.ToolStripButton btDel;  
.ToolStripButton btEdit;  
.ToolStripButton btnPesquisar;  
.ToolStripButton btnRelatorio;  
.GroupBox gbLivro;  
.Label lbIsbn;  
.Label lbAutor;  
.Label lbGenero;  
.Label lbPublicacao;  
.Label lbEditora;  
.Label lbTitulo;  
.Label lbCod;  
.Label label2;  
.Label label6;  
.Label label5;  
.Label label8;  
.Label label4;  
.Label label3;  
.Label label1;  
.TextBox txtISBN;  
.Label label7;
```

Vamos agora codificar o evento Load do Formulário para carregar os dados do livro. Para isso será necessário criar a classe **LivroDB** na pasta de controle:



Codifique conforme abaixo o método conexão na classe LivroDB:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
Conexao con = new Conexao("localhost", "biblioteca",
                           "root", "minas");
public void consultar(System.Windows.Forms.BindingSource bs)
{
    // simplifica a chamada a biblioteca entidades
    using (var banco = new modelo.bibliotecaEntidades())
    {
        // conecta no banco de dados
        banco.Database.Connection.ConnectionString = con.open();
        // consulta usando a linguagem LINQ
        var query = from linha in
                     banco.livro
                     .Include("autor").Include("editora")
                     .Include("genero").Include("autor")
                     orderby linha.idLivro
                     select linha;

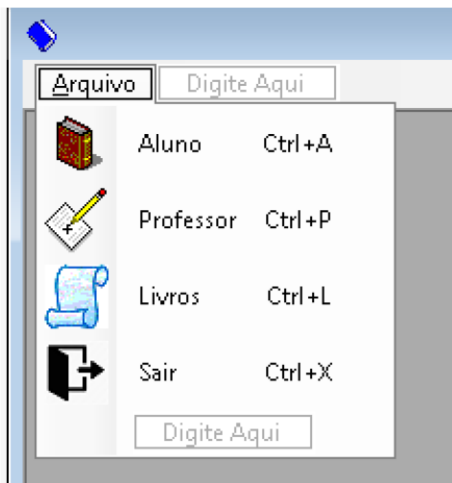
        // transforma em uma lista
        bs.DataSource = query.ToList();
    }
}
```

Vamos agora codificar a rotina de chamada da classe no FormAluno, usando o evento **Load**:

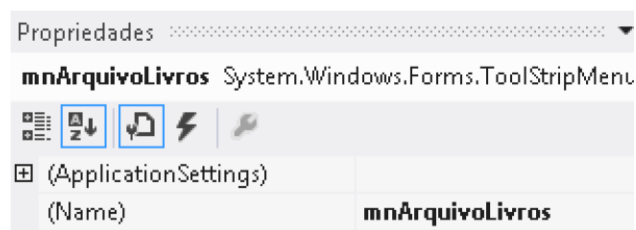
```
private void FormLivro_Load(object sender, EventArgs e)
{
    controle.LivroDB aDb = new controle.LivroDB();
    aDb.consultar(bs);

    lbCod.DataBindings.Add(new Binding("text", bs, "idLivro"));
    lbTitulo.DataBindings.Add(new Binding("text", bs, "titulo"));
    lbAutor.DataBindings.Add(new Binding("text", bs, "autor.nome"));
    lbGenero.DataBindings.Add(new Binding("text", bs, "genero.descricao"));
    lbEditora.DataBindings.Add(new Binding("text", bs,
        "editora.descricao"));
    lbPublicacao.DataBindings.Add(new Binding("text", bs, "publicacao"));
    lbIsbn.DataBindings.Add(new Binding("text", bs, "codisbn"));
}
```

Necessário agora criar a chamada ao formulário livro no formulário principal, logo crie uma nova opção de menu:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Chame o item de menu conforme a figura abaixo:



O passo seguinte é criar uma variável global para o formulário de livros em **FormMDI**:

```
public partial class FormMDI : Form
{
    public visao.FormAluno frAluno = null;

    // aqui a criação do formulário livro
    public visao.FormLivro frLivro = null;

    public FormMDI()
    {
        InitializeComponent();
    }
}
```



Agora o evento clique para o item de menu livro:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void mnArquivoLivros_Click(object sender, EventArgs e)
{
    if (frLivro == null)
    {
        frLivro = new visao.FormLivro();
        frLivro.MdiParent = this;
        frLivro.Show();
    }
}
```

Escreva também a rotina de fechamento do formulário livro em FormLivro:

```
private void FormLivro_FormClosing(object sender, FormClosingEventArgs e)
{
    FormMDI fr = (FormMDI)this.MdiParent;
    fr.frLivro = null;
}
```

Com os dados inseridos acima podemos testar o código do formulário livro.

Ficha Livro:

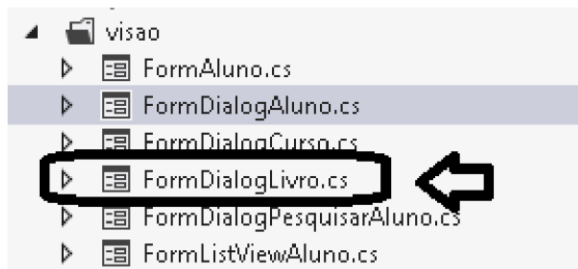
Código:	1	ISBN:	8765
Nome do Livro: DOM CASMURRO			
Editora:	EDITORA	Autor:	ANÔNIMO
Publicação:	24/06/2016	Gênero:	ROMANCE DE
ISBN:		<input type="text"/>	

1 de 1 | adicionar | ~~excluir~~ | editar | Pesquisar

Formulário de Cadastro de Livro

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

A etapa seguinte é projetar o formulário de cadastro de livro, chamado **FormDialogLivro**



Crie o formulário conforme a seguir:

A screenshot of a Windows form titled "Livro". The form contains the following fields and controls:

- Título:** A text input field.
- Publicação:** A date input field showing "24/06/2016" with a calendar icon.
- Resumo:** A large text area.
- ISBN:** A text input field.
- Páginas:** A text input field.
- Edição Número:** A text input field.
- ISBN:** A button with a magnifying glass icon and the text "ISBN".
- Genero:** A dropdown menu with options: [Novo...](#), [Editar...](#), [Excluir...](#).
- Autor:** A dropdown menu with options: [Novo...](#), [Editar...](#), [Excluir...](#).
- Editora:** A dropdown menu with options: [Novo...](#), [Editar...](#), [Excluir...](#).
- Buttons:** Two buttons at the bottom right: "gravar" and "cancelar".

Nome dos componentes na caixa de diálogo:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
TextBox txtTitulo
Label label1
DateTimePicker dtPublicacao
Label label12
MaskedTextBox txtPaginas
Label label5
Label label2
MaskedTextBox txtEdicao
Label label3
TextBox txtResumo
Label label4
TextBox txtISBN
Button btProcurar
ComboBox cbGenero
Label label6
Label label7
ComboBox cbAutor
Label label8
ComboBox cbEditora
Button btCancelar
Button btGravar
LinkLabel lnkEditarEditora
LinkLabel lnkExcluirEditora
LinkLabel lnkEditarGenero
LinkLabel lnkExcluirGenero
LinkLabel lnkNovoGenero
LinkLabel lnkEditarAutor
LinkLabel lnkNovoAutor
LinkLabel lnkExcluirAutor
LinkLabel lnkNovoEditora
```

Crie as referências para **BindingSource** e **livro** no formulário:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public partial class FormDialogLivro : Form
{
    private BindingSource bs;

    public BindingSource Bs
    {
        get { return bs; }
        set { bs = value; }
    }

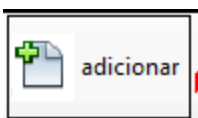
    private modelo.livro livro;

    public modelo.livro Livro
    {
        get { return livro; }
        set { livro = value; }
    }

    public FormDialogLivro()
    {
        InitializeComponent();
    }
}
```



Codificar o botão adicionar no formulário FormLivro para chamar a caixa de diálogo:



Evento click:

```
private void btAdd_Click(object sender, EventArgs e)
{
    FormDialogLivro fr = new FormDialogLivro();
    fr.Livro = null;
    fr.Bs = bs;
    fr.ShowDialog();

    if (fr.Livro != null)
    {
        bs.ResetBindings(false);
        controle.LivroDB ldb = new controle.LivroDB();
        ldb.consultar(bs);
        bs.MoveLast();
    }
}
```

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Desta forma o formulário está preparadora para ser chamado, porém os **combobox** da editora, gênero e autor estarão vazios. Vamos codifica-los. Começaremos com o autor. Crie a classe **AutorDB** na pasta **controle**:

```
class AutorDB
{
    Conexao con = new Conexao("localhost", "biblioteca",
        "root", "minas");

    public void listar(System.Windows.Forms.ComboBox cb)
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            // conecta ao banco
            banco.Database.Connection.ConnectionString =
                con.open();
            var query = from linha in banco.autor
                        orderby linha.idAutor
                        select linha;
            cb.DataSource = query.ToList();
            // exibe a descrição do curso no combobox
            cb.DisplayMember = "nome";
            // porém ao selecionar retorna o código
            cb.ValueMember = "idAutor";
        }
    }
}
```

Podemos chamar esta rotina no evento Load do **FormDialogLivro**:

```
private void FormDialogLivro_Load(object sender, EventArgs e)
{
    controle.AutorDB aDb = new controle.AutorDB();
    aDb.listar(cbAutor);
}
```



O resultado que teremos será:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Repetir o processo agora para a editora. Crie a classe **EditoraDB** na pasta controle:

```
class EditoraDB
{
    Conexao con = new Conexao("localhost", "biblioteca",
        "root", "minas");

    public void listar(System.Windows.Forms.ComboBox cb)
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            // conecta ao banco
            banco.Database.Connection.ConnectionString =
                con.open();
            var query = from linha in banco.editora
                        orderby linha.idEditora
                        select linha;
            cb.DataSource = query.ToList();
            // exibe a descrição do curso no combobox
            cb.DisplayMember = "descricao";
            // porém ao selecionar retorna o código
            cb.ValueMember = "idEditora";
        }
    }
}
```

Repetir o processo agora para o gênero. Crie a classe **GeneroDB** na pasta controle:

```
class GeneroDB
{
    Conexao con = new Conexao("localhost", "biblioteca",
        "root", "minas");

    public void listar(System.Windows.Forms.ComboBox cb)
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            // conecta ao banco
            banco.Database.Connection.ConnectionString =
                con.open();
            var query = from linha in banco.genero
                        orderby linha.idGenero
                        select linha;
            cb.DataSource = query.ToList();
            // exibe a descrição do curso no combobox
            cb.DisplayMember = "descricao";
            // porém ao selecionar retorna o código
            cb.ValueMember = "idGenero";
        }
    }
}
```

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Acrescente a chamada no evento **Load** do **FormDialogLivro**:

```
private void FormDialogLivro_Load(object sender, EventArgs e)
{
    controle.AutorDB aDb = new controle.AutorDB();
    aDb.listar(cbAutor);

    controle.GeneroDB gDb = new controle.GeneroDB();
    gDb.listar(cbGenero);
    controle.EditoraDB eDb = new controle.EditoraDB();
    eDb.listar(cbEditora);

    if (Livro == null)
    {
        cbGenero.SelectedIndex = 0;
        cbEditora.SelectedIndex = 0;
        cbAutor.SelectedIndex = 0;
    }
    else {
        cbEditora.SelectedValue = Livro.idEditora;
        cbGenero.SelectedValue = Livro.idGenero;
        cbAutor.SelectedValue = Livro.idAutor;
        txtTitulo.Text = Livro.titulo;
        txtEdicao.Text = Livro.edicao.ToString();
        txtPaginas.Text = Livro.nrpaginass.ToString() ;
        txtResumo.Text = Livro.resumo ;
        dtPublicacao.Value = Livro.publicacao;
        txtISBN.Text = Livro.codisbnn;
    }
}
```

O resultado visual será:



The screenshot shows a Windows application window titled "Livro". It contains several input fields and dropdown menus. The "Titulo" field is filled with "DOM CASMURRO". The "Resumo" field is filled with "TESTE". The "Páginas" field is filled with "89" and the "Edição Número:" field is filled with "24". The "Genero:" dropdown menu is set to "ROMANCE DE EPOCA". The "Autor:" dropdown menu is set to "ANÔNIMO". The "Editora:" dropdown menu is set to "EDITORIA RECORD S.A". Below the "Editora:" dropdown menu, there is a list of editors: "EDITORIA RECORD S.A" and "GLOBO EDITORA SA".

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Agora temos a estrutura do formulário pronta para seleção de dados. Podemos iniciar os procedimentos de gravação de dados na tabela livro e posteriormente implantar também a lógica de cadastro dos botões para autor, editora e gênero.

Vá para a classe **LivroDB** e codifique a rotina de inserir livro:

```
public void inserir(modelo.livro novo)
{
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            // conecta ao banco
            banco.Database.Connection.ConnectionString = con.open();
            // adiciona o novo livro a lista
            banco.livro.Add(novo);
            // salva no banco de dados
            banco.SaveChanges();
        }
    }
    catch (Exception err)
    {
        // em caso de erro exibir a mensagem
        System.Windows.Forms.MessageBox.Show("Erro:" + err.Message);
    }
}
```

Codifique também a rotina editar em **LivroDB**:

```
public void editar(modelo.livro reg)
{
    using (var banco = new modelo.bibliotecaEntidades())
    {
        banco.Database.Connection.ConnectionString = con.open();
        modelo.livro livro =
            banco.livro.Single(qr => qr.idLivro == reg.idLivro);
        livro.idEditora = reg.idEditora;
        livro.idGenero = reg.idGenero;
        livro.idAutor = reg.idAutor;
        livro.titulo = reg.titulo;
        livro.edicao = reg.edicao;
        livro.nrpaginas = reg.nrpaginas;
        livro.resumo = reg.resumo;
        livro.publicacao = reg.publicacao;
        livro.codisbn = reg.codisbn;
        banco.SaveChanges();
    }
}
```

Codifique também a rotina excluir em **LivroDB**:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public void excluir(modelo.livro reg)
{
    using (var banco = new modelo.bibliotecaEntidades())
    {
        // conecta ao banco
        banco.Database.Connection.ConnectionString = con.open();
        // seleciona o registro a ser deletado usando o método single
        modelo.livro livro=
            banco.livro.Single(qr => qr.idLivro == reg.idLivro);
        // remove o registro selecionado
        banco.livro.Remove(livro);
        // salvavolva a informação no banco de dados
        banco.SaveChanges();
    }
}
```

Codifique a rotina próximo código livre em **LivroDB**

```
public int proximoCodigo()
{
    int t = 0;
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            banco.Database.Connection.ConnectionString =
                con.open();
            var query = (from linha in banco.livro
                        select linha.idLivro).Max();
            t = Convert.ToInt16(query.ToString());
        }
        return t + 1;
    }
    catch (Exception err)
    {
        System.Console.WriteLine(err.Message);
        return 1;
    }
}
```

Por fim, a rotina de procurar registro de livro em **LivroDB**:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public modelo.livro procurar(modelo.livro reg)
{
    using (var banco = new modelo.bibliotecaEntidades())
    {
        // conecta ao banco
        banco.Database.Connection.ConnectionString = con.open();
        // seleciona o registro a ser procurado usando o método single
        modelo.livro livro =
            banco.livro.Single(qr => qr.idLivro == reg.idLivro);
        // retorna se encontrado
        return livro;
    }
}
```

Já podemos desenvolver a rotina de cadastro de livros em **FormDialogLivro**. Vamos codificar o botão gravar:

```
private void btGravar_Click(object sender, EventArgs e)
{
    if (Livro == null)
    {
        novo();
    }
    else
    {
        editar();
    }

    MessageBox.Show("Registro gravado com sucesso!");
    this.Dispose();
}
```

Agora vamos codificar a rotina **novo**:

```
private void novo()
{
    controle.LivroDB lDB = new controle.LivroDB();
    Livro = new modelo.livro() {
        idLivro = lDB.proximoCodigo(),
        idEditora = Convert.ToInt16(cbEditora.SelectedValue),
        idGenero = Convert.ToInt16(cbGenero.SelectedValue),
        idAutor = Convert.ToInt16(cbAutor.SelectedValue),
        titulo = txtTitulo.Text.ToUpper(),
        edicao = Convert.ToInt16(txtEdicao.Text),
        nrpaginas = Convert.ToInt16(txtPaginas.Text),
        resumo = txtResumo.Text.ToUpper(),
        publicacao = dtPublicacao.Value,
        codisbn = txtISBN.Text
    };

    lDB.inserir(Livro);
}
```

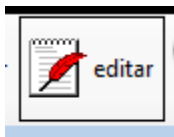
Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Codificar a rotina **editar**:

```
private void editar()
{
    controle.LivroDB ldb = new controle.LivroDB();

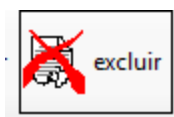
    Livro.idEditora = Convert.ToInt16(cbEditora.SelectedValue);
    Livro.idGenero = Convert.ToInt16(cbGenero.SelectedValue);
    Livro.idAutor = Convert.ToInt16(cbAutor.SelectedValue);
    Livro.titulo = txtTitulo.Text.ToUpper();
    Livro.edicao = Convert.ToInt16(txtEdicao.Text);
    Livro.nrpaginass = Convert.ToInt16(txtPaginas.Text);
    Livro.resumo = txtResumo.Text.ToUpper();
    Livro.publicacao = dtPublicacao.Value;
    Livro.codisbn = txtISBN.Text;
    ldb.editar(Livro);
}
```

Passemos agora para o formulário **FormLivro.cs** e vamos codificar a rotina do botão editar da barra de navegação:



```
private void btEdit_Click(object sender, EventArgs e)
{
    if (bn.PositionItem.Text.Equals("0"))
    {
        MessageBox.Show("Cadastro Vazio", "Mensagem",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    FormDialogLivro fr = new FormDialogLivro();
    // registro atual
    modelo.livro reg = (modelo.livro)bs.Current;
    // conecta a tabela
    controle.LivroDB ldb = new controle.LivroDB();
    // Retorna a pesquisa na própria variável 'reg'
    reg = ldb.procurar(reg);
    //Passa a varfiavel reg para FormDialog
    fr.Livro = reg;
    fr.ShowDialog();
    // se atualizou
    if (fr.Livro != null)
    {
        // reconstroi o bindingsource
        ldb.consultar(bs);
        bs.ResetBindings(false);
    }
}
```

Para terminar codificar o botão excluir em **FormLivro.cs**:

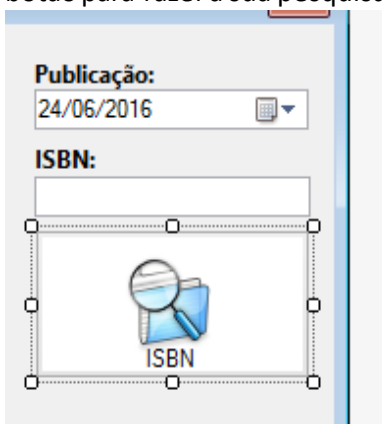


Etec Dr Demétrio Azevedo Junior – Itapeva-SP

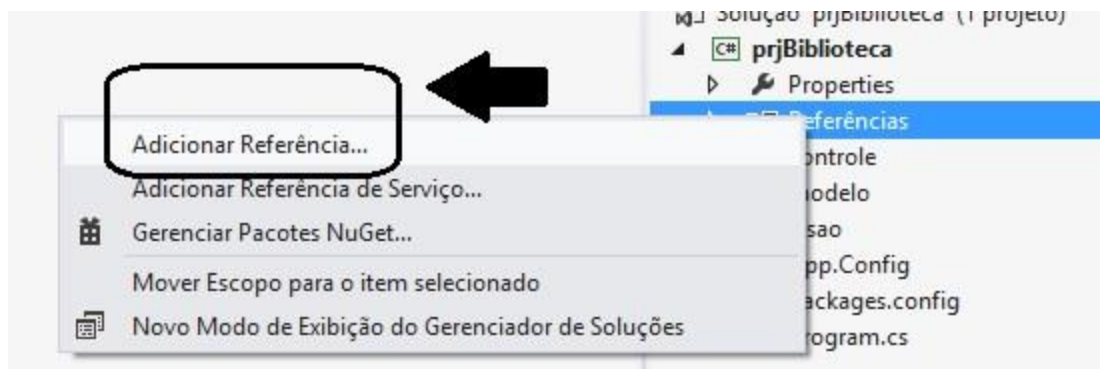
```
private void btDel_Click(object sender, EventArgs e)
{
    // não tem registros cadastrados?
    if (bn.PositionItem.Text.Equals("0"))
    {
        MessageBox.Show("Cadastro Vazio", "Mensagem",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    // conecta a tabela
    controle.LivroDB aDb = new controle.LivroDB();
    // mensagem de exclusao
    if (MessageBox.Show("Remover " + lbAutor.Text,
        "Sistema", MessageBoxButtons.YesNo,
        MessageBoxIcon.Question) == DialogResult.Yes)
    {
        // pega o registro atual
        modelo.livro reg = (modelo.livro)bs.Current;
        // exclui do banco de dados
        aDb.excluir(reg);
        // remove da lista o registro
        bn.BindingSource.RemoveCurrent();
    }
}
```

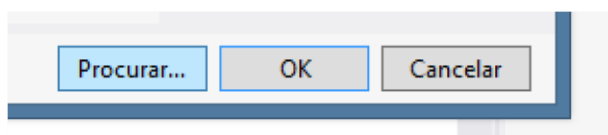
A rotina de cadastro de livro está pronta. Agora poderemos nos concentrar em dois pontos técnicos de nosso projeto: o código do livro ISBN. Para isso crie um campo a mais no formulário para armazenar o código do ISBN e um botão para fazer a sua pesquisa:



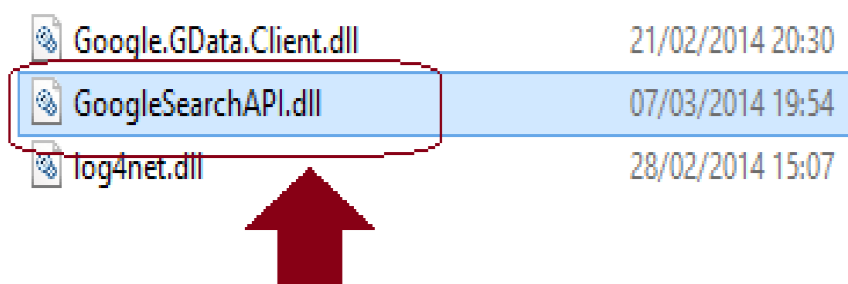
O próximo passo é adicionar a biblioteca de consulta a base de dados de livros do google. Vá em seu projeto e adicione a seguinte biblioteca:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Clique no botão procurar:



Localize a seguinte **dll** em seu disco rígido, fornecido pelo professor:



Essa biblioteca fornece suporte ao google books, usamos as sus bibliotecas para realizar a pesquisa dos dados do livro na internet e preencher o formulário em nosso sistema. Para isso vá em **LivroDB** e acrescente as seguintes referências:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Google.API.Search;

namespace prjBiblioteca.controle
{
    class LivroDB
    {
        Conexao con = new Conexao("biblioteca", "localhost", "root", "minas");
    }
}
```

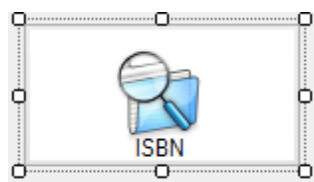
Acrescente na classe o seguinte método:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public void procurarLivroGoogle(string isbn, modelo.livro livro)
{
    GbookSearchClient cliente =
        new GbookSearchClient("www.etecitapeva.com.br");
    // pesquisa o livro e retorna um resultado:
    string nomeautor = "";
    IList<IBookResult> resultados = cliente.Search(isbn, 1);
    if (resultados.Count == 0)
    {
        System.Windows.Forms.MessageBox.Show(
            "ISBN não localizado para o livro");
        livro = null;
        return;
    }

    try
    {
        foreach (IBookResult resultado in resultados)
        {
            // preenche os dados do livro
            livro.titulo = resultado.Title.ToString();
            nomeautor = resultado.Authors.ToString();
            livro.resumo = "Livro de " + nomeautor;
            livro.codisbn = resultado.BookId.ToString();
            livro.publicacao = new
                DateTime(Int16.Parse(resultado.PublishedYear)
                    , 1, 1);
            livro.nrpaginas = Int16.Parse(resultado.PageCount.ToString());
        }
    }
    // as vezes não há pagina nem ano de publicação para o livro
    catch (FormatException)
    {
        livro.publicacao = DateTime.Today; // data de hoje
        livro.nrpaginas = 0; //zero páginas
    }
}
```

Codificar o botão Procurar:



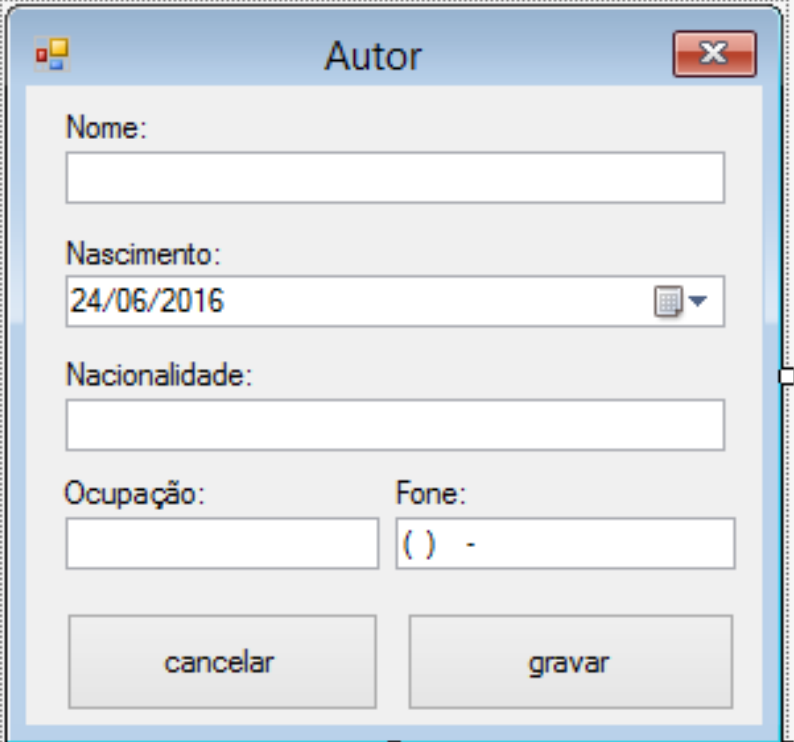
Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void btProcurar_Click(object sender, EventArgs e)
{
    try
    {
        controle.LivroDB ldb = new controle.LivroDB();
        modelo.livro reg = new modelo.livro();
        ldb.procurarLivroGoogle(txtISBN.Text, reg);
        if (reg == null) return;

        txtTitulo.Text = reg.titulo;
        txtEdicao.Text = reg.edicao.ToString();
        txtPaginas.Text = reg.nrpaginas.ToString();
        txtResumo.Text = reg.resumo;
        dtPublicacao.Value = reg.publicacao;
    }
    catch (Exception err) {
        MessageBox.Show("Erro:" + err.Message);
    }
}
```

ROTINA DE CADASTRO DE AUTORES

É necessário agora codificar a rotina de cadastro de autores para poder inserir o livro corretamente no banco de dados. Para isso iremos codificar o botão adicionar e exclui livro. Primeiro crie um formulário para o cadastro de autor chamado **FormDialogAutor**:



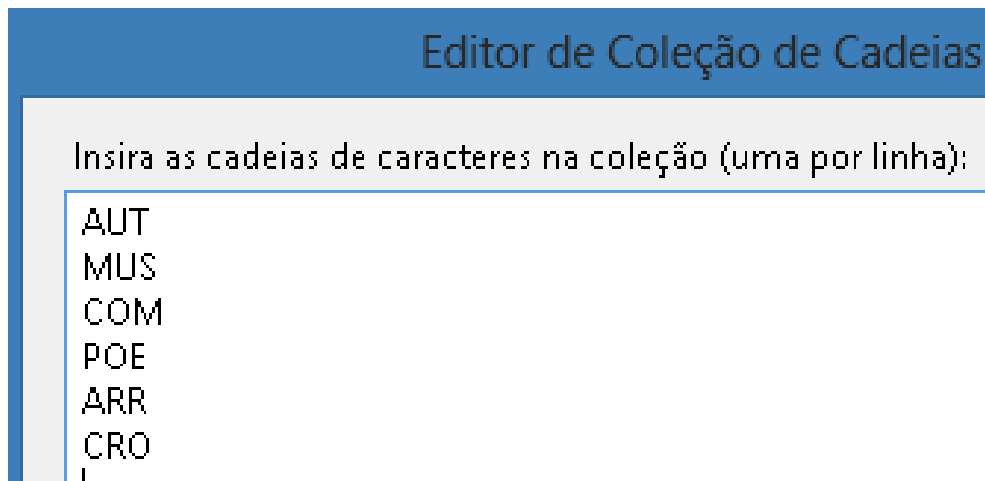
The image shows a Windows form titled "Autor" with a standard Windows XP-style title bar. The form contains several input fields and two buttons at the bottom. The fields are labeled "Nome:", "Nascimento:", "Nacionalidade:", "Ocupação:", and "Fone:". The "Nascimento:" field contains the date "24/06/2016" and has a calendar icon to its right. The "Fone:" field has a placeholder "() -". At the bottom, there are two buttons labeled "cancelar" and "gravar".

Nome dos campos:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
.DateTimePicker dtNascimento;  
.Label label12  
.TextBox txtNome  
.Label label1  
.Label label2  
.TextBox txtNacionalidade  
.Label label3  
.TextBox txtOcupacao  
.MaskedTextBox txtFone  
.Label label11  
.Button btCancelar  
.Button btGravar
```

Preencher ocupação com os seguintes valores padrões:




Codificar no formulário um `FormDialogAutor` um campo para receber os dados do formulário `FormDialogLivro`:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public partial class FormDialogAutor : Form
{
    private modelo.autor autor;

    public modelo.autor Autor
    {
        get { return autor; }
        set { autor = value; }
    }

    public FormDialogAutor()
    {
        InitializeComponent();
    }
}
```



Codificar o evento **load** de FormDialogAutor:

```
private void FormDialogAutor_Load(object sender, EventArgs e)
{
    if (Autor != null) { // é edição?
        // carrega campos no formulário

        txtNome.Text = Autor.nome;
        txtNacionalidade.Text = Autor.nacionalidade;
        txtFone.Text = Autor.telefone;
        cbOcupacao.Text = Autor.ocupacao;
        dtNascimento.Value = Autor.nascimento;
    }
}
```

Criar o evento click do botão gravar:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void btGravar_Click(object sender, EventArgs e)
{
    if (Autor == null) novo();
    else editar();
    this.Dispose();
}
```

Método novo, será responsável por gravar o autor no banco de dados, gerando um novo código e passando os campos do formulário para os campos da tabela do banco de dados via EDMX:

```
private void novo()
{
    controle.AutorDB aDb = new controle.AutorDB();
    Autor = new modelo.autor() {
        idAutor = aDb.proximoCodigo(),
        nome = txtNome.Text,
        nacionalidade = txtNacionalidade.Text,
        ocupacao = txtOcupacao.Text,
        telefone = txtFone.Text,
        nascimento = dtNascimento.Value
    };
    aDb.inserir(Autor);
}
```

Em **AutorDB** codificar o método de próximo código:

```
public int proximoCodigo()
{
    int t = 0;
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            banco.Database.Connection.ConnectionString =
                con.open();
            var query = (from linha in banco.autor
                        select linha.idAutor).Max();
            t = Convert.ToInt16(query.ToString());
        }
        return t + 1;
    }
    catch (Exception err)
    {
        System.Console.WriteLine(err.Message);
        return 1;
    }
}
```

Em **AutorDB** codificar o método inserir:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public void inserir(modelo.autor novo)
{
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            // conecta ao banco
            banco.Database.Connection.ConnectionString = con.open();
            // adiciona na lista
            banco.autor.Add(novo);
            // salva no banco de dados
            banco.SaveChanges();
        }
    }
    catch (Exception err)
    {
        // em caso de erro exibir a mensagem
        System.Windows.Forms.MessageBox.Show("Erro: " + err.Message);
    }
}
```

Em **FormDialogAutor** codificar o método editar, o objetivo é carregar os campos da mesma forma que na rotina novo, não será modificado o código do autor.

```
private void editar()
{
    controle.AutorDB aDb = new controle.AutorDB();
    Autor.nome = txtNome.Text;
    Autor.nacionalidade = txtNacionalidade.Text;
    Autor.ocupacao = txtOcupacao.Text;
    Autor.telefone = txtFone.Text;
    Autor.nascimento = dtNascimento.Value;
    aDb.editar(Autor);
}
```

Em **AutorDB** codificar o método editar:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public void editar(modelo.autor reg)
{
    using (var banco = new modelo.bibliotecaEntidades())
    {
        banco.Database.Connection.ConnectionString = con.open();
        modelo.autor autor =
            banco.autor.Single(qr => qr.idAutor == reg.idAutor);
        autor.nome = reg.nome;
        autor.nacionalidade = reg.nacionalidade;
        autor.nascimento = reg.nascimento;
        autor.ocupacao = reg.ocupacao;
        autor.telefone = reg.telefone;
        banco.SaveChanges();
    }
}
```

Em **FormDialogLivre** codificar o link novo:



```
private void lnkNovoAutor_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)
{
    FormDialogAutor fr = new FormDialogAutor();
    fr.Autor = null;
    fr.ShowDialog();
    if (fr.Autor != null)
    {
        controle.AutorDB gDb = new controle.AutorDB();
        gDb.listar(cbAutor);
    }
}
```

Agora codificar o botão excluir:

```
private void lnkExcluirGenero_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)
{
    controle.GeneroDB eDb = new controle.GeneroDB();
    modelo.genero genero =
        (modelo.genero)cbGenero.SelectedItem;
    eDb.excluir(genero);
    eDb.listar(cbGenero);
    MessageBox.Show("Genero excluido com sucesso!");
}
```

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Em **AutorDB** codificar o método excluir:

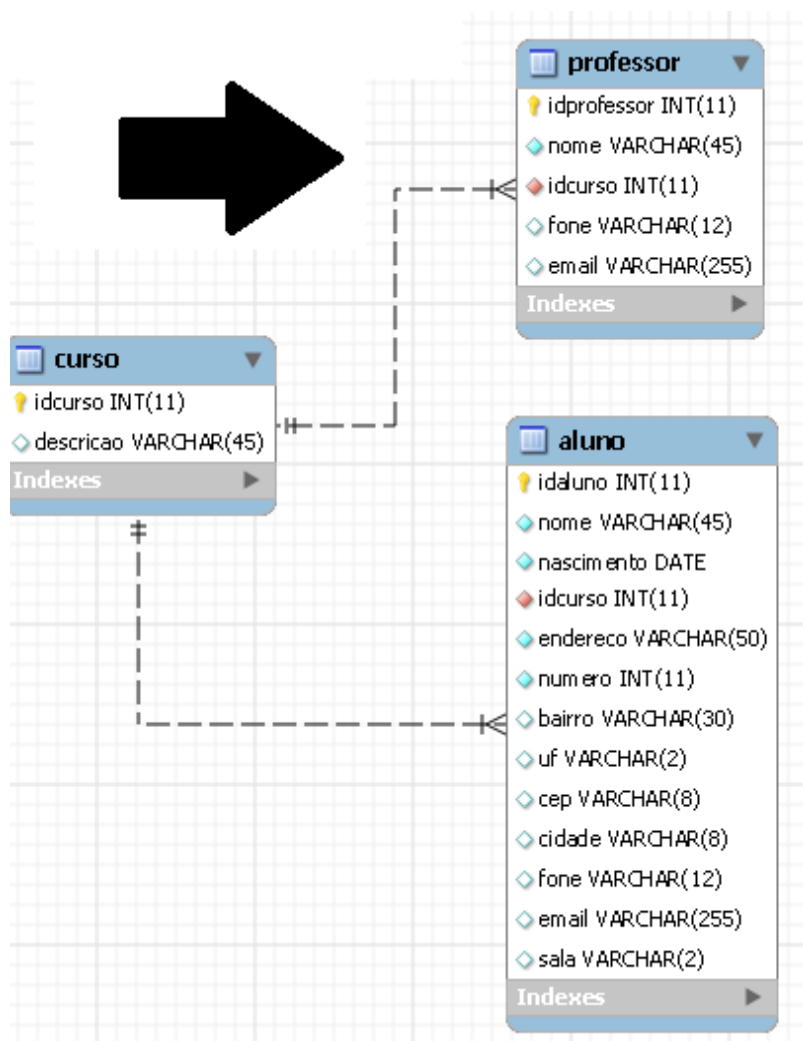
```
public void excluir(modelo.autor reg)
{
    using (var banco = new modelo.bibliotecaEntidades())
    {
        // conecta ao banco
        banco.Database.Connection.ConnectionString = con.open();
        // seleciona o registro a ser deletado usando o método single
        modelo.autor autor =
            banco.autor.Single(qr => qr.idAutor == reg.idAutor);
        // autor o registro selecionado
        banco.autor.Remove(autor);
        // salva a informação no banco de dados
        banco.SaveChanges();
    }
}
```

A codificação da rotina de edição fica no link editar:

```
private void lnkEditarAutor_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)
{
    FormDialogAutor fr = new FormDialogAutor();
    fr.Autor = (modelo.autor)cbAutor.SelectedItem;
    fr.ShowDialog();
    if (fr.Autor != null)
    {
        controle.AutorDB eDb = new controle.AutorDB();
        eDb.listar(cbAutor);
    }
}
```

Desenvolvendo o módulo Professor

O módulo professor será desenvolvido sem o uso de uma barra de navegação. Neste caso, também não será utilizado um formulário de diálogo, ficando toda a lógica de manipulação de registros codificada em apenas um único formulário. Inicie o projeto codificando a tabela de professor no banco de dados biblioteca:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Script:

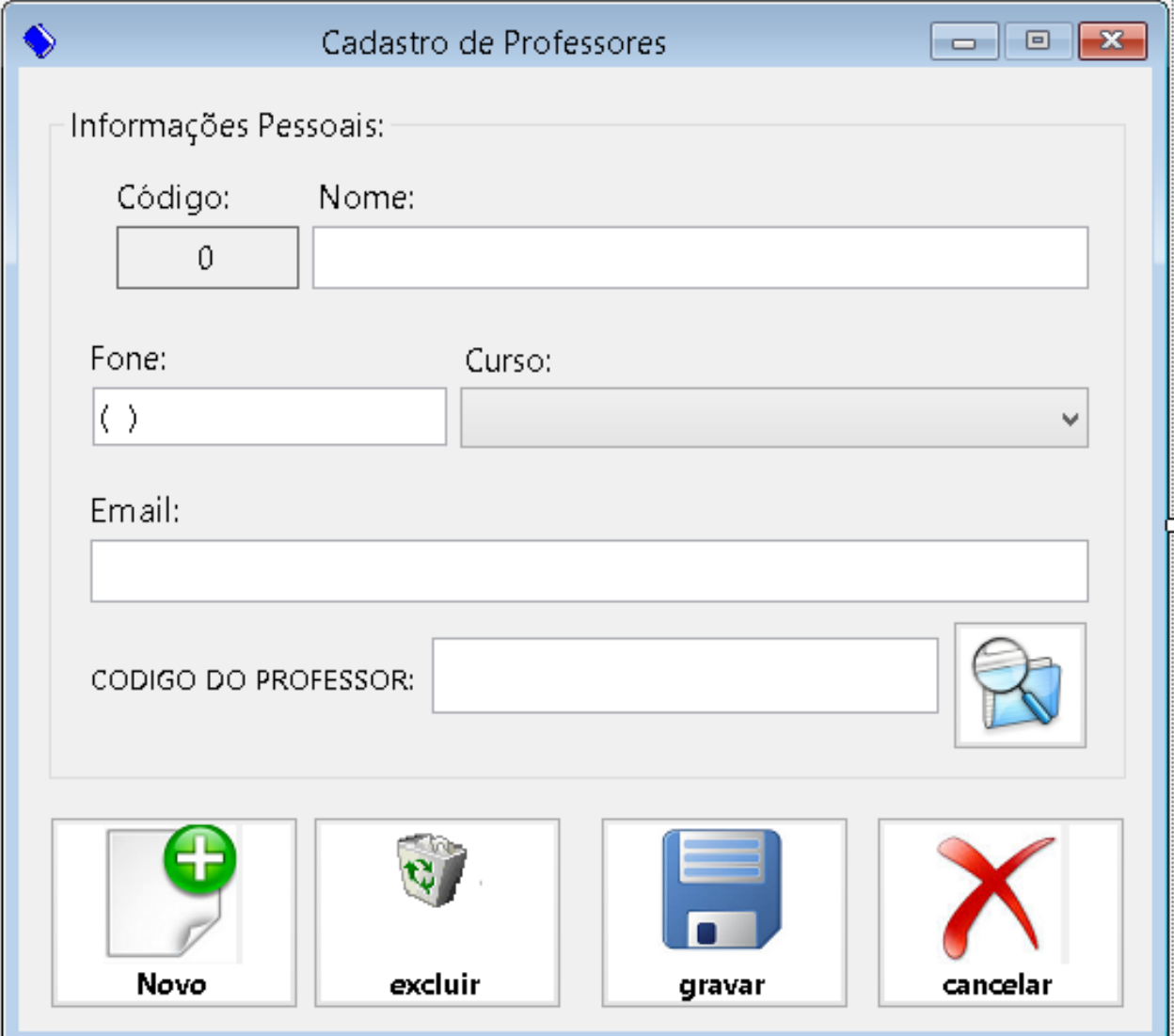
```
CREATE TABLE IF NOT EXISTS `biblioteca`.`professor` (  
  `idprofessor` INT(11) NOT NULL ,  
  `nome` VARCHAR(45) NOT NULL ,  
  `idcurso` INT(11) NOT NULL ,  
  `fone` VARCHAR(12) NULL DEFAULT NULL ,  
  `email` VARCHAR(255) NULL DEFAULT NULL ,  
  PRIMARY KEY (`idprofessor`) ,  
  INDEX `FK_CURSO` (`idcurso` ASC) ,  
  CONSTRAINT `FK_CURSO_PROFESSOR`  
    FOREIGN KEY (`idcurso`)  
      REFERENCES `biblioteca`.`curso` (`idcurso`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB
```

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

DEFAULT CHARACTERSET = latin1

Atualize o arquivo EDMX da pasta modelo, para que o mesmo agora contenha a tabela de professores criada acima.

O próximo passo será desenhar um formulário chamado **FormProfessor** com a seguinte estrutura visual:



Cadastro de Professores

Informações Pessoais:

Código: Nome:

0

Fone: Curso:

()

Email:

CODIGO DO PROFESSOR:

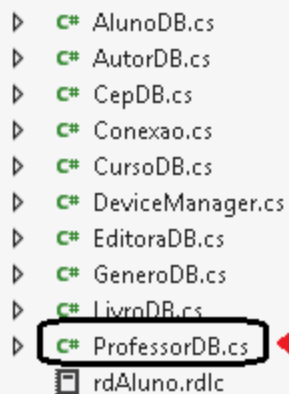
Novo excluir gravar cancelar

Nome do componentes:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
.GroupBox gbInformacoes  
.MaskedTextBox txtFone  
.Label lblFone  
.Label lblNome  
.TextBox txtNome  
.Label lblEmail  
.TextBox txtEmail  
.ComboBox cbCurso  
.Label lblCurso  
.Label lblTitulo  
.TextBox txtCod  
.Button btnPesquisar  
.Button btAdd  
.Button btDel  
.Button btSave  
.Button btCancel  
.Label lblCodigo  
.Label lblCod
```

Crie a classe **ProfessorDB** em controle:


Etec Dr Demétrio Azevedo Junior – Itapeva-SP

- ▶ C# AlunoDB.cs
- ▶ C# AutorDB.cs
- ▶ C# CepDB.cs
- ▶ C# Conexao.cs
- ▶ C# CursoDB.cs
- ▶ C# DeviceManager.cs
- ▶ C# EditoraDB.cs
- ▶ C# GeneroDB.cs
- ▶ C# LivroDB.cs
- ▶ C# ProfessorDB.cs
- ▶ rdAluno.rdlc

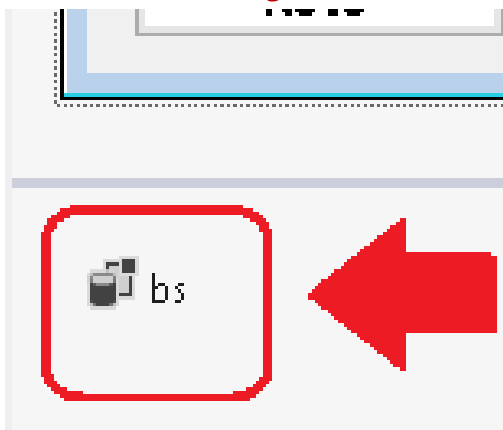
Em ProfessorDB crie a rotina que irá carregar para o **bindingsource**, os campos da tabela:

```
class ProfessorDB
{
    Conexao con = new Conexao("localhost", "biblioteca", "root", "minas");

    public void carregarTabela(System.Windows.Forms.BindingSource bs)
    {
        using (var banco = new modelo.bibliotecaEntidades()) //conecta com o BD
        {
            banco.Database.Connection.ConnectionString = con.open();
            var query = from linha in banco.professor
                        orderby linha.idprofessor
                        select linha;
            bs.DataSource = query.ToList();
        }
    }
}
```



Adicione um **bindingsource** a FormProfessor:



Em FormProfessor codificar o seguinte evento Load:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void FormProfessor_Load(object sender, EventArgs e)
{
    this.Cursor = Cursors.WaitCursor;

    controle.ProfessorDB pDB = new controle.ProfessorDB();
    pDB.carregarTabela(bs);

    // carrega a lista de cursos
    controleCursoDB cDB = new controleCursoDB();
    cDB.listar(cbCurso);

    //move para o primeiro registro se banco de dados não vazio
    if (bs.Count != 0)
    {
        bs.MoveFirst();
        habilitarCampos(true);
        btCancel.Enabled = false;
    }
    else
    {
        btDel.Enabled = false;
        btCancel.Enabled = false;
        btSave.Enabled = false;

        habilitarCampos(false);
    }

    vincularCampos();

    this.Cursor = Cursors.Default;
}
```


Codificar a rotina habilitar campos em **FormProfessor**:

```
private void habilitarCampos(bool status)
{
    txtNome.Enabled = status;
    txtFone.Enabled = status;
    txtEmail.Enabled = status;
    cbCurso.Enabled = status;
    btnPesquisar.Enabled = status;
    txtCod.Enabled = status;
}
```




Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Codificar a rotina vincular campos em **FormProfessor**:



```
private void vincularCampos()
{
    lbCodigo.DataBindings.Add(new Binding("Text", bs, "idprofessor"));
    txtNome.DataBindings.Add(new Binding("Text", bs, "nome"));
    txtFone.DataBindings.Add(new Binding("Text", bs, "fone"));
    txtEmail.DataBindings.Add(new Binding("Text", bs, "email"));
    cbCurso.DataBindings.Add(new Binding("SelectedValue", bs, "idCurso"));
}
```

Codificar o botão novo:



```
private void btAdd_Click(object sender, EventArgs e)
{
    habilitarCampos(true);

    btDel.Enabled = false;
    btAdd.Enabled = false;

    btSave.Enabled = true;
    btCancel.Enabled = true;

    bs.AddNew();

    controle.ProfessorDB pDB = new controle.ProfessorDB();

    lbCodigo.Focus();
    lbCodigo.Text = pDB.proximoCodigo().ToString();

    cbCurso.Focus();
    cbCurso.SelectedIndex = 0;

    txtNome.Focus();
}
```

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Note que a rotina não adiciona nada ainda no banco de dados, ela criar um registro em branco no bindingsource, limpando os campos. A Rotina próximo código calcula o próximo código livre, e posiciona-se no campo nome. Crie em ProfessorDB a rotina de próximo código:

```
public int proximoCodigo()
{
    int t = 0;
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            banco.Database.Connection.ConnectionString = con.open();
            var query = (from linha in banco.professor
                        select linha.idprofessor).Max();
            t = Convert.ToInt16(query.ToString());
        }
        return t + 1;
    }
    catch (Exception err)
    {
        System.Console.WriteLine(err.Message);
        return 1;
    }
}
```

Agora codificar o botão gravar em FormProfessor, que irá gravar os dados no banco de dados, verificando o estado dos demais botões para determinar se será um adição ou edição de registros no banco de dados:



Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void btSave_Click(object sender, EventArgs e)
{
    // se botão cancelar verdadeiro estou adicionando no banco de dados

    if (btCancel.Enabled == true)
    {
        modelo.professor p = (modelo.professor)bs.Current;
        controle.ProfessorDB pDB = new controle.ProfessorDB();
        pDB.inserir(p);
        MessageBox.Show("Registro adicionado", "Alerta",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        // move para o registro adicionado
        bs.MoveLast();
    }
    else {
        // se não é edição de registro
        modelo.professor p = (modelo.professor)bs.Current;
        controle.ProfessorDB pDB = new controle.ProfessorDB();
        pDB.editar(p);
        MessageBox.Show("Registro editado", "Alerta",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}
```

```
btAdd.Enabled = true;
```

```
if (bs.Count != 0)
{
    btDel.Enabled = true;
}
```

```
btCancel.Enabled = false;
btSave.Enabled = true;
}
```

Em **ProfessorDB** crie a rotina de inserir:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public void inserir(modelo.professor professor)
{
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            banco.Database.Connection.ConnectionString = con.open();
            banco.professor.Add(professor);
            banco.SaveChanges(); //salva alterações
        }
    }
    catch (System.Data.Entity.Validation.DbEntityValidationException dbEx)
    {
        string msg = "Lista de Erros ao adicionar registro:\n";
        foreach (var validationErrors in dbEx.EntityValidationErrors)
        {
            // pega os campos que derão problema de gravação
            foreach (var validationError in validationErrors.ValidationErrors)
            {
                msg = String.Format("{0}\n", validationError.ErrorMessage);
            }
        }
        // exibe os campos com erro
        System.Windows.Forms.MessageBox.Show("Erro:" + msg);
    }

    catch (System.Data.EntityException dbEx)
    {
        // erro geral
        System.Windows.Forms.MessageBox.Show("Erro:" + dbEx.Message);
    }
    catch (System.Data.Entity.Infrastructure.DbUpdateException dbEx)
    {
        // erro geral
        System.Windows.Forms.MessageBox.Show("Erro de adição de registro:" + dbEx.Message);
    }
}
```

Em **ProfessorDB** crie a rotina de editar:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public void editar(modelo.professor prof)
{
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            banco.Database.Connection.ConnectionString = con.open();
            modelo.professor reg = (from linha in banco.professor
                                   where linha.idprofessor == prof.idprofessor
                                   select linha).FirstOrDefault<modelo.professor>();

            reg.nome = prof.nome;
            reg.fone = prof.fone;
            reg.idcurso = prof.idcurso;
            reg.email = prof.email;
            banco.SaveChanges();
        }
    }
}

catch (System.Data.Entity.Validation.DbEntityValidationException dbEx)
{
    string msg = "Lista de Erros ao adicionar registro:\n";
    foreach (var validationErrors in dbEx.EntityValidationErrors)
    {
        // pega os campos que derão problema de gravação
        foreach (var validationError in validationErrors.ValidationErrors)
        {
            msg = String.Format("{0}\n", validationError.ErrorMessage);
        }
    }
    // exibe os campos com erro
    System.Windows.Forms.MessageBox.Show("Erro:" + msg);
}
```

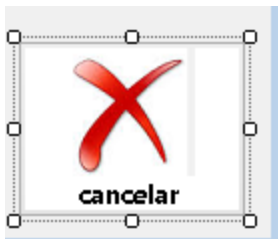
Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
catch (System.Data.EntityException dbEx)
{
    // erro geral
    System.Windows.Forms.MessageBox.Show("Erro:" + dbEx.Message);
}

catch (System.Data.Entity.Infrastructure.DbUpdateException dbEx)
{
    // erro geral
    System.Windows.Forms.MessageBox.Show("Erro de atualização:" + dbEx.Message);
}

}
```

Em **FormProfessor** codificar o botão cancelar:



```
private void btCancel_Click(object sender, EventArgs e)
{
    btAdd.Enabled = true;

    btCancel.Enabled = false;
    btSave.Enabled = false;

    bs.RemoveCurrent();

    if (bs.Count != 0)
    {
        btDel.Enabled = true;
        habilitarCampos(true);
    }
    else {
        btDel.Enabled = false;
        habilitarCampos(false);
    }
}
```

Em **FormProfessor** codificar o botão excluir:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void btDel_Click(object sender, EventArgs e)
{
    if (bs.Count == 0)
    {
        MessageBox.Show("Cadastro Vazio", "Mensagem",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    controle.ProfessorDB lDb = new controle.ProfessorDB();
    int id = Convert.ToInt16(lbCodigo.Text);

    if (MessageBox.Show("Remover " + txtNome.Text,
        "Sistema", MessageBoxButtons.YesNo,
        MessageBoxIcon.Question) == DialogResult.Yes)
    {
        lDb.excluir(id);
        bs.RemoveCurrent();
        bs.MoveFirst();
        if (bs.Count != 0) bs.MoveFirst();
        else
        {
            habilitarCampos(false);
            btDel.Enabled = false;
            btSave.Enabled = false;
            btCancel.Enabled = false;
        }
    }
}
```

Em **ProfessorDB** crie a rotina de exclusão do registro:

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
public void excluir(int cod)
{
    try
    {
        using (var banco = new modelo.bibliotecaEntidades())
        {
            banco.Database.Connection.ConnectionString = con.open();

            var prof = banco.professor.First(a => a.idprofessor == cod);
            banco.professor.Remove(prof);
            banco.SaveChanges();
        }
    }
    catch (System.Data.EntityException dbEx)
    {
        // erro geral
        System.Windows.Forms.MessageBox.Show("Erro:" + dbEx.Message);
    }

    catch (System.Data.Entity.Infrastructure.DbUpdateException dbEx)
    {
        // erro geral
        System.Windows.Forms.MessageBox.Show("Erro de exclusão:" + dbEx.Message);
    }
}
```

Codificar o botão de pesquisar, que vai localizar um registro no bindingsource e retornará a posição do mesmo, preenchendo de forma automática os campos do formulário:



Etec Dr Demétrio Azevedo Junior – Itapeva-SP

```
private void btnPesquisar_Click(object sender, EventArgs e)
{
    int idProfessor = Int16.Parse(txtCod.Text);
    var obj = bs.List.OfType<modelo.professor>().ToList()
        .Find(l => l.idprofessor == idProfessor);

    // passa a posição para o bindingsource bs
    int pos = bs.IndexOf(obj);
    if (pos < 0) MessageBox.Show("professor não encontrado....");
    else bs.Position = pos;
}
```

Programa o evento de pressionamento de tecla do campo txtCod para executar a rotina de pesquisa de forma automatizada:

```
private void txtCod_KeyPress(object sender, KeyPressEventArgs e)
{
    if(e.KeyChar == Convert.ToChar(Keys.Enter)){
        btnPesquisar_Click(sender, e);
    }
}
```

Programa o fechamento do formulário:

```
private void FormProfessor_FormClosing(object sender, FormClosingEventArgs e)
{
    FormMDI fr = (FormMDI)this.MdiParent;
    fr.frProfessor = null;
}
```

Execute o programa e teste a funcionalidade do programa. Você poderá adicionar, excluir e editar os dados pressionando os botão novo, excluir e salvar quando necessário. Para pesquisar use o campo de pesquisa de código do professor.

Etec Dr Demétrio Azevedo Junior – Itapeva-SP

Cadastro de Professores

Informações Pessoais:

Código: 1 Nome: MARCELO CANDIDO

Fone: (11)11111111 Curso: enfermagem

Email: teste@uol.com

CODIGO DO PROFESSOR:

Novo excluir gravar cancelar

Exercício

1. Codifique um formulário de pesquisa contextual por nome para o formulário Professor, nos mesmos moldes do usado para o formulário aluno. Use o componente link Label para abrir o formulário de acordo com a seguinte aparência:

Informações Pessoais:

Código: 0 Nome: [Pesquisar Nome...](#)

Fone: () Curso:

Email:

Propriedades

InkPesquisar System.Windows.Forms.LinkLabel

(ApplicationSettings)

(DataBindings)

(Name) InkPesquisar

AccessibleDescription