

2019

ComboBox SQL Server e C#

ARQUITETURA DAL
RUBEM CÂNDIDO DOS SANTOS

FATEC – SENAI CENTRO | Rua Caetés 741, Belo Horizonte

Sumário

Introdução	2
Criando o Banco.....	2
Criando a conexão com o Banco	2
Criando o Projeto	7
Adicionando Classes	8
Adicionando Class alunoEnt	8
Adicionando classe alunoModel	10
Adicionando Referências.....	11
IDisposable.....	13
No momento vou colar o código, depois preciso passar parte por parte.....	13
Carregando os dados ao selecionar o dado no Combobox.....	20

Introdução

Criação de um CRUD simples para acessar o SQL Server e buscar os dados através de um Combobox.

Criando o Banco

Vamos criar um banco chamado **db_cadastro** e uma tabela chamada **alunos**.

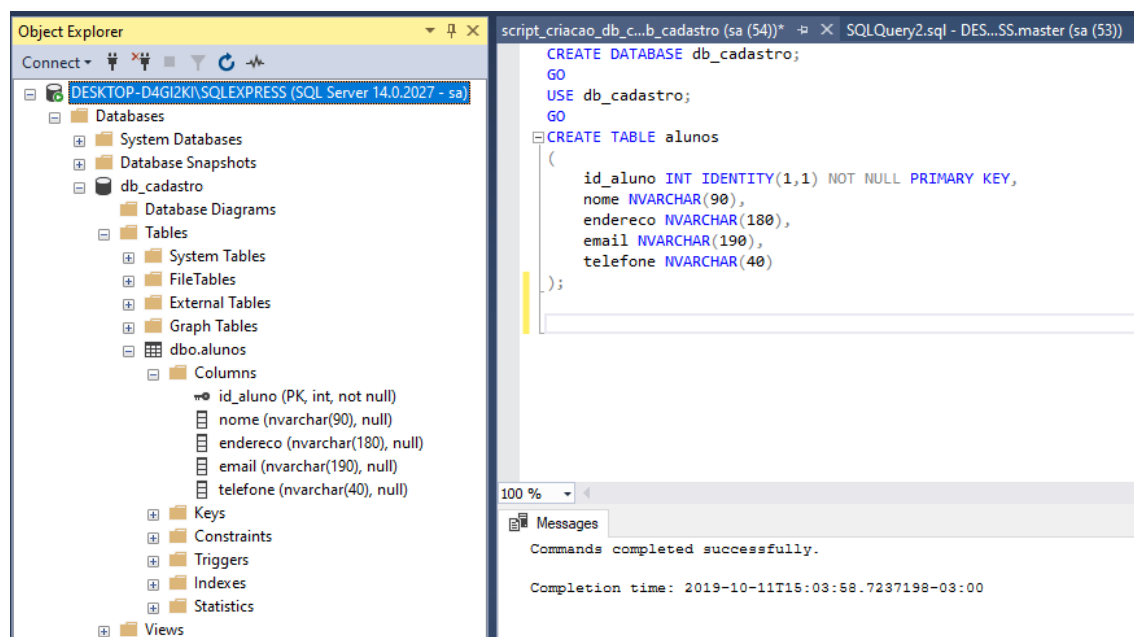


Figura 1. Script de criação do banco e tabela

Criando a conexão com o Banco

Abra o Management do SqlServer e copie o nome do servidor

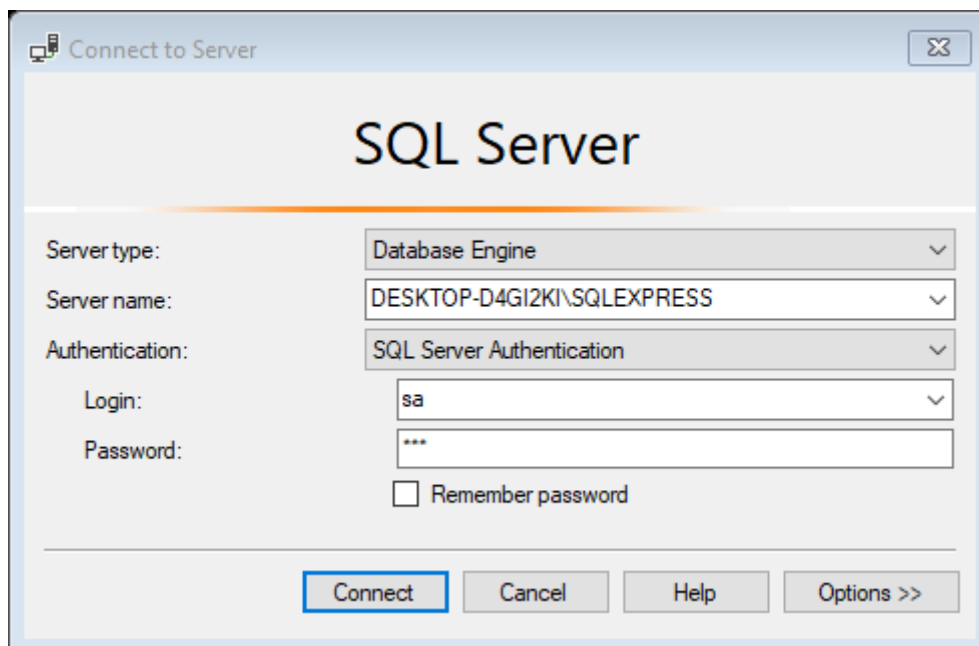


Figura 4. Connet to Server

No Visual Studio clique em Server Explorer > Data Connections com o botão direito e preencha com os dados do seu banco

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:
Microsoft SQL Server (SqlClient) Change...

Server name:
DESKTOP-D4GI2KI\SQLEXPRESS Refresh

Log on to the server

Authentication: SQL Server Authentication

User name: sa

Password: ●●●

☐ Save my password

Connect to a database

☒ Select or enter a database name:
db_cadastro

☐ Attach a database file:
Browse...

Logical name:

Advanced...

Test Connection OK Cancel

Figura 3. Add Connection

Clique com o botão direito do mouse no CRUD_ComboBox e em propriedades

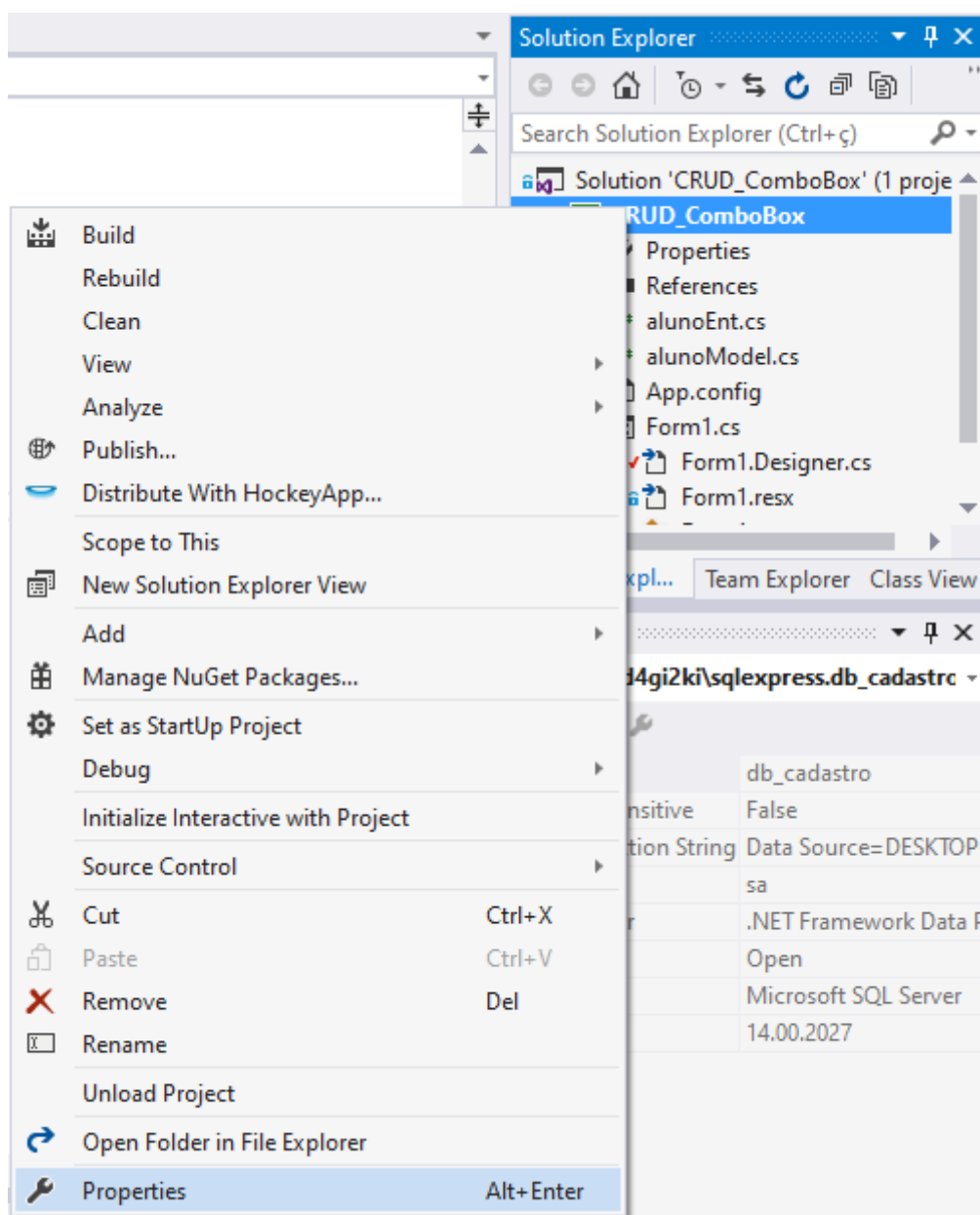


Figura 4. Propriedades do CRUD_ComboBox

Crie uma conexão clicando em Settings preenchendo os campos e clicando nos três pontinhos

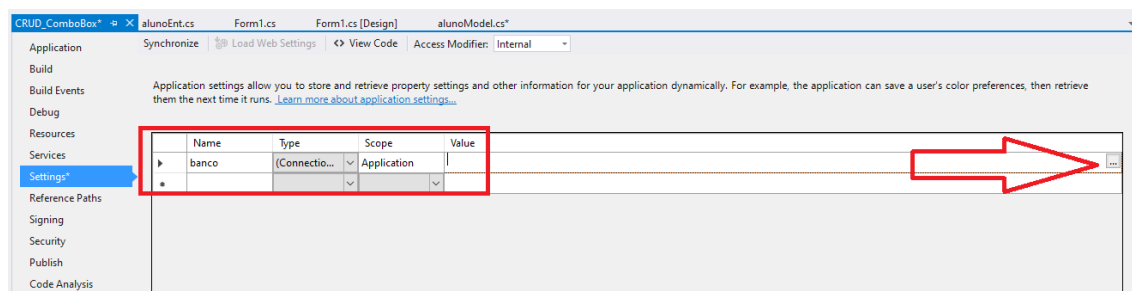


Figura 5. Criando conexão com o banco

Repita a configuração anterior

Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:
Microsoft SQL Server (SqlClient) Change...

Server name:
DESKTOP-D4GI2KI\SQLEXPRESS Refresh

Log on to the server

Authentication: SQL Server Authentication

User name: sa

Password: ●●●

☐ Save my password

Connect to a database

☒ Select or enter a database name:
db_cadastro

☐ Attach a database file:
Browse...

Logical name:

Advanced...

Test Connection OK Cancel

Figura 3. Add Connection

Feche a configuração e salve as alterações clicando em Yes

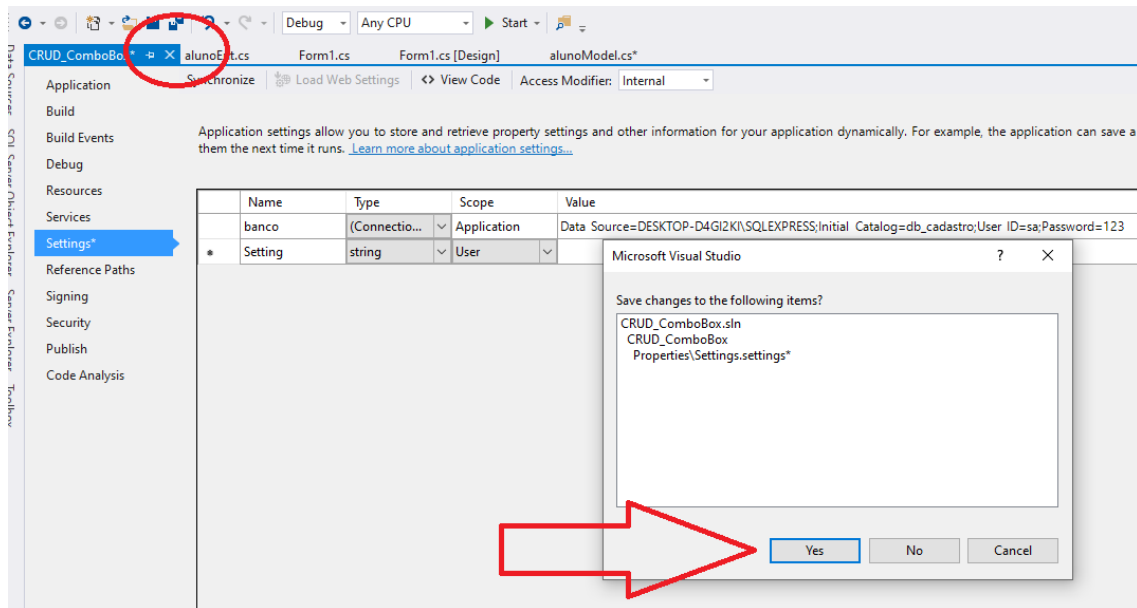


Figura 5. Salvando configurações com o banco

Criando o Projeto

Vamos criar um projeto do tipo Windows Forms com o nome CRUD_ComboBox.

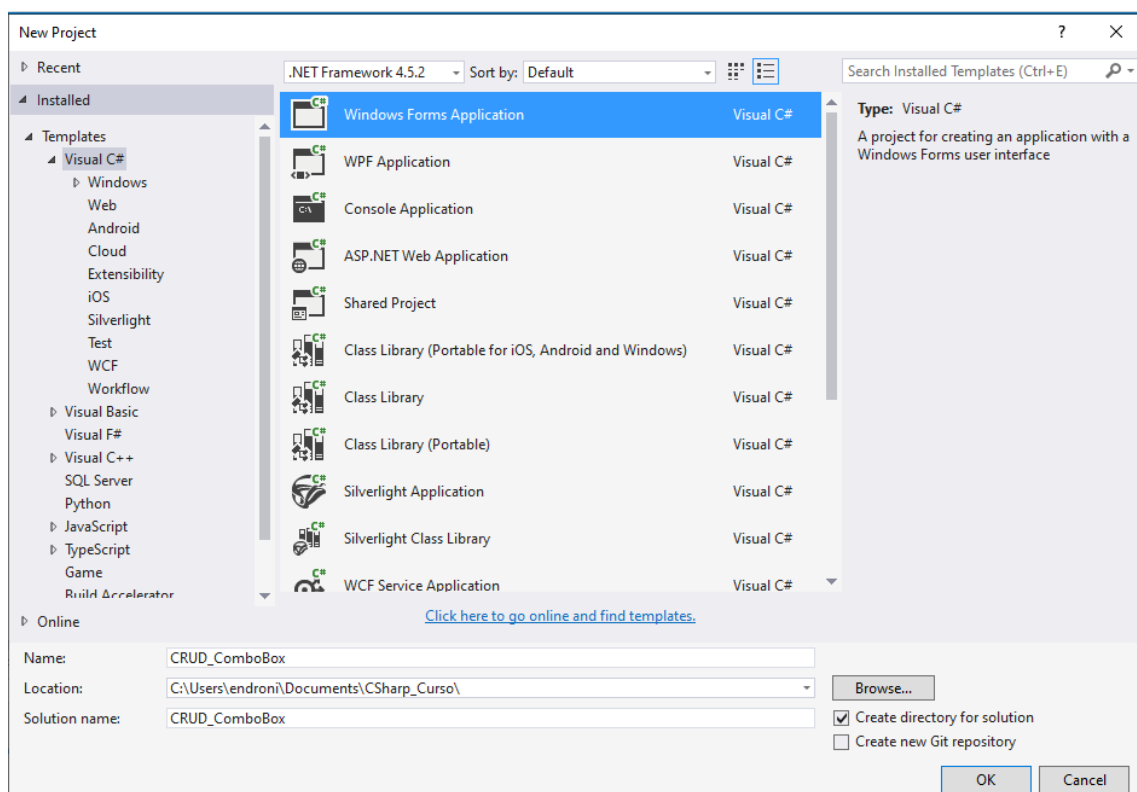


Figura 2. Criando o projeto CRUD_ComboBox

Adicionando Classes

Pensando em arquitetura em camadas, vamos criar duas classes.

Adicionando Class alunoEnt

Precisamos de uma classe que iremos chamar de **alunoEnt**.

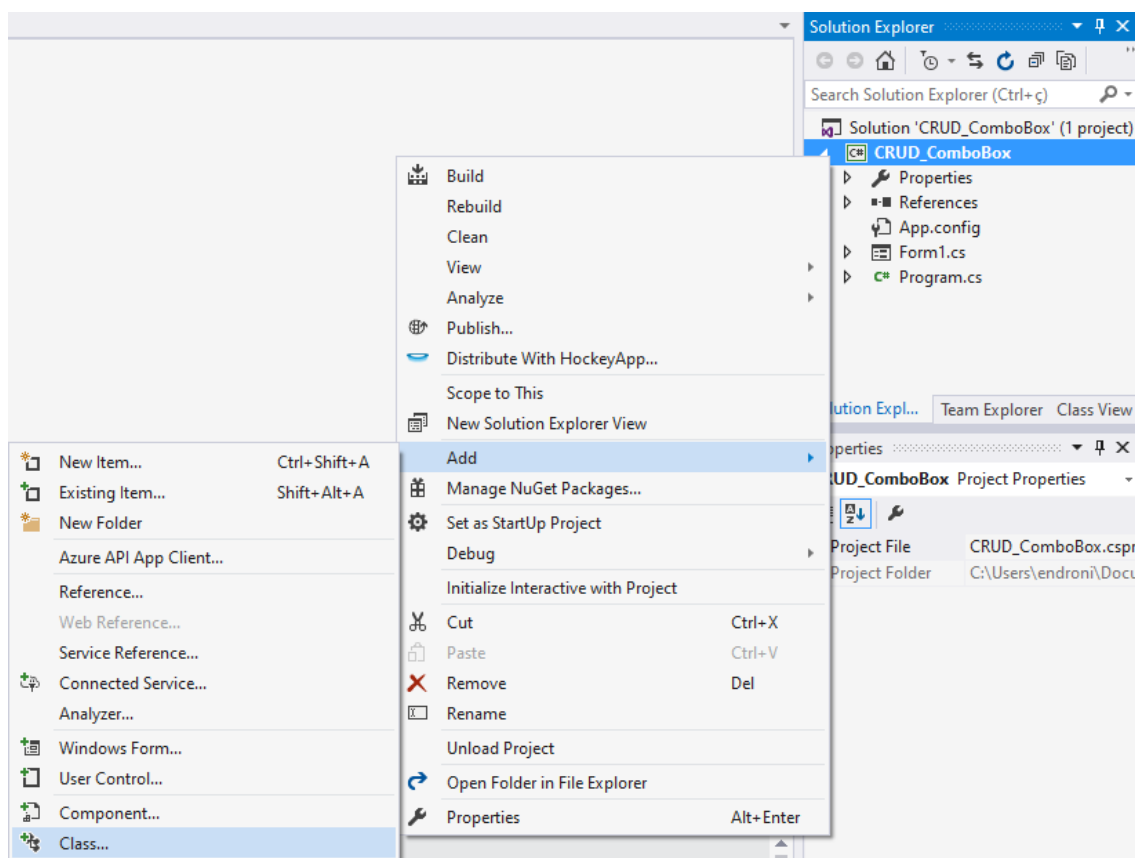


Figura 3. Botão de Add Class

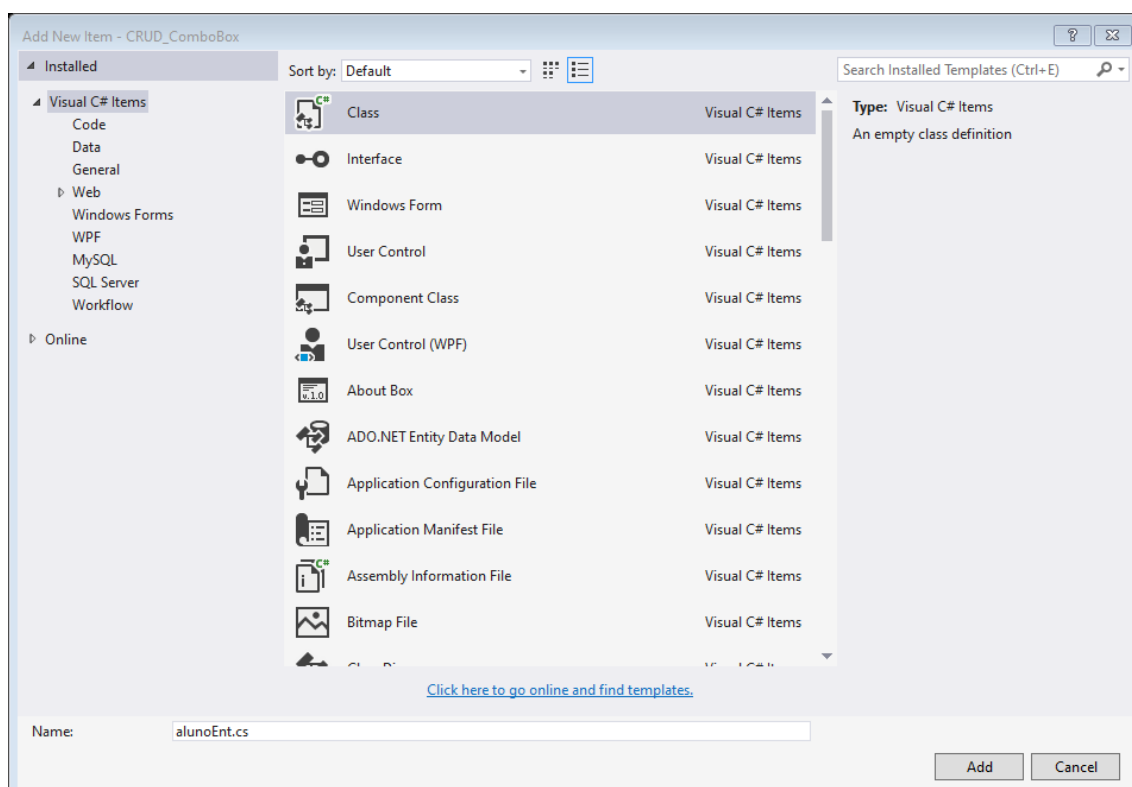


Figura 4. Criando a classe alunoEnt

Insira os get sets na classe alunoEnt

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace CRUD_ComboBox
8  {
9      public class alunoEnt
10     {
11         public int IdAluno { get; set; }
12         public string Nome { get; set; }
13         public string Endereco { get; set; }
14         public string Email { get; set; }
15         public string Telefone { get; set; }
16     }
17 }
18
19

```

Figura 5. get's e set's

Adicionando classe alunoModel

Precisamos de uma classe que irá conter os códigos de acesso e persistencia de dados. Iremos chamala de **alunoModel**

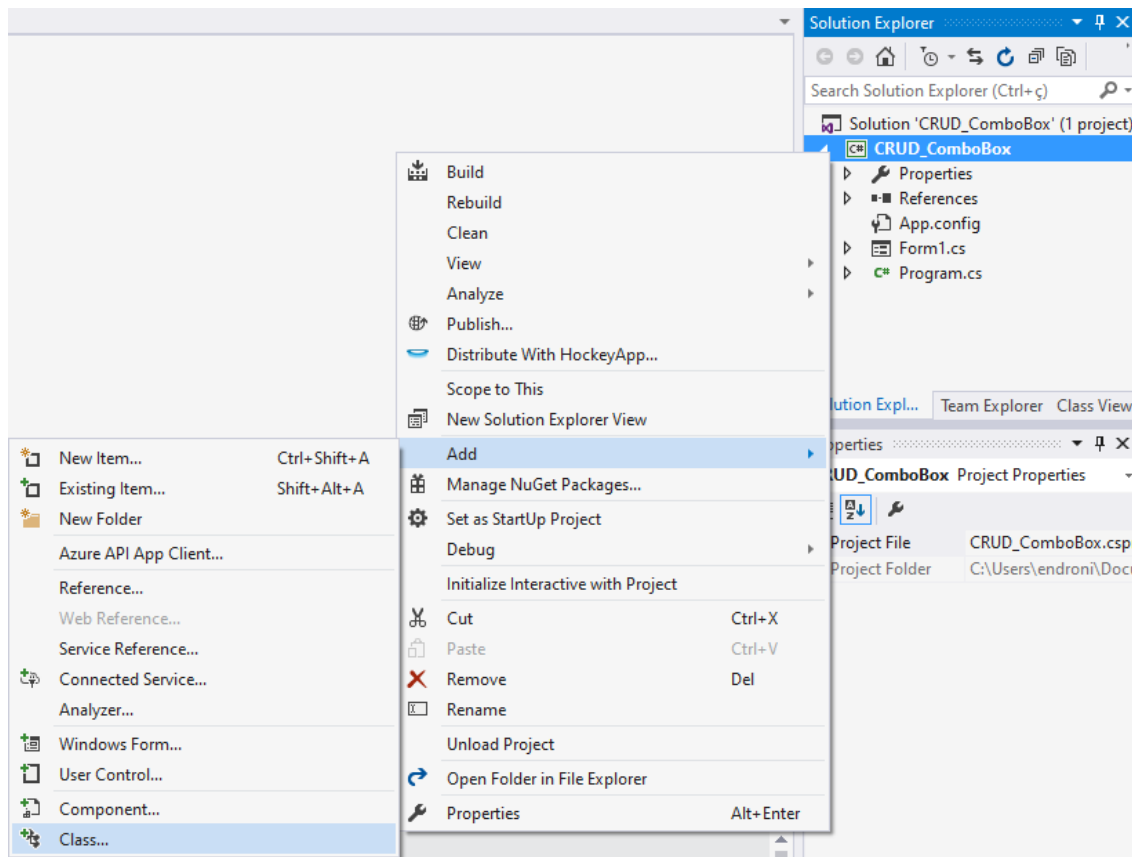


Figura 5. Botão de Add Class

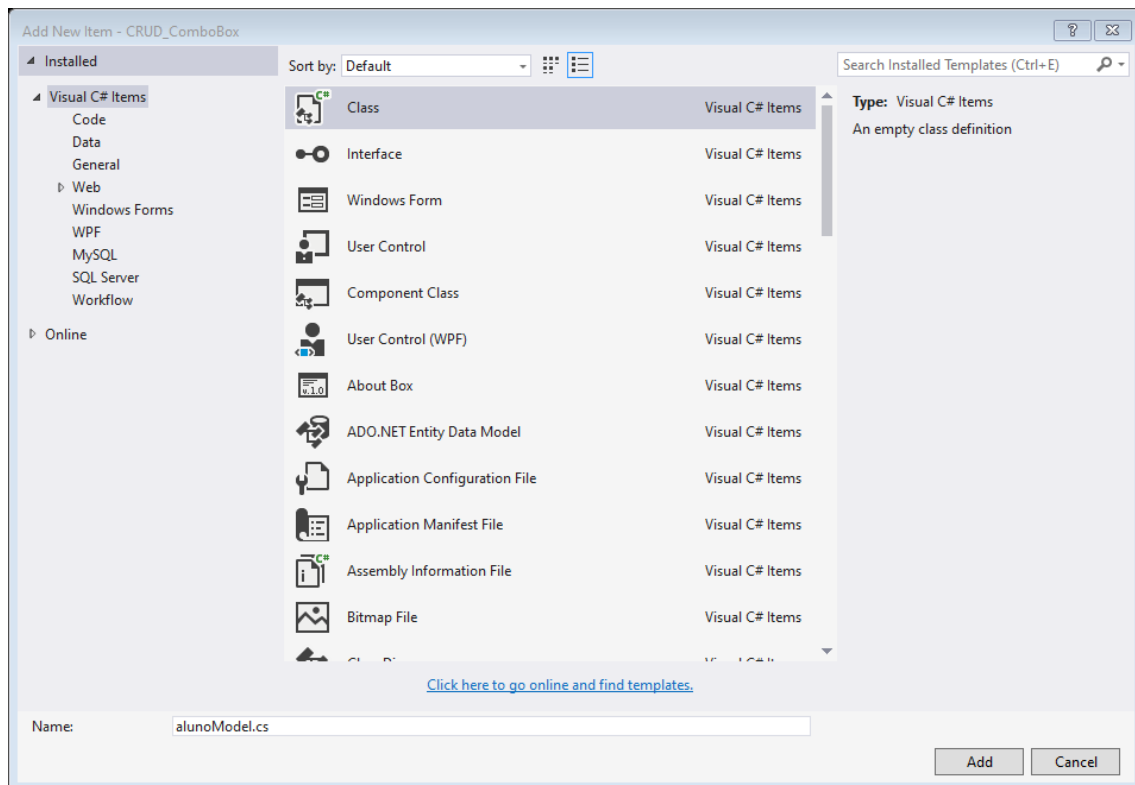


Figura 6. Criando a classe alunoModel

Adicionando Referências

Adicione uma referência à essa classe. Cliquem em Project Add Reference...

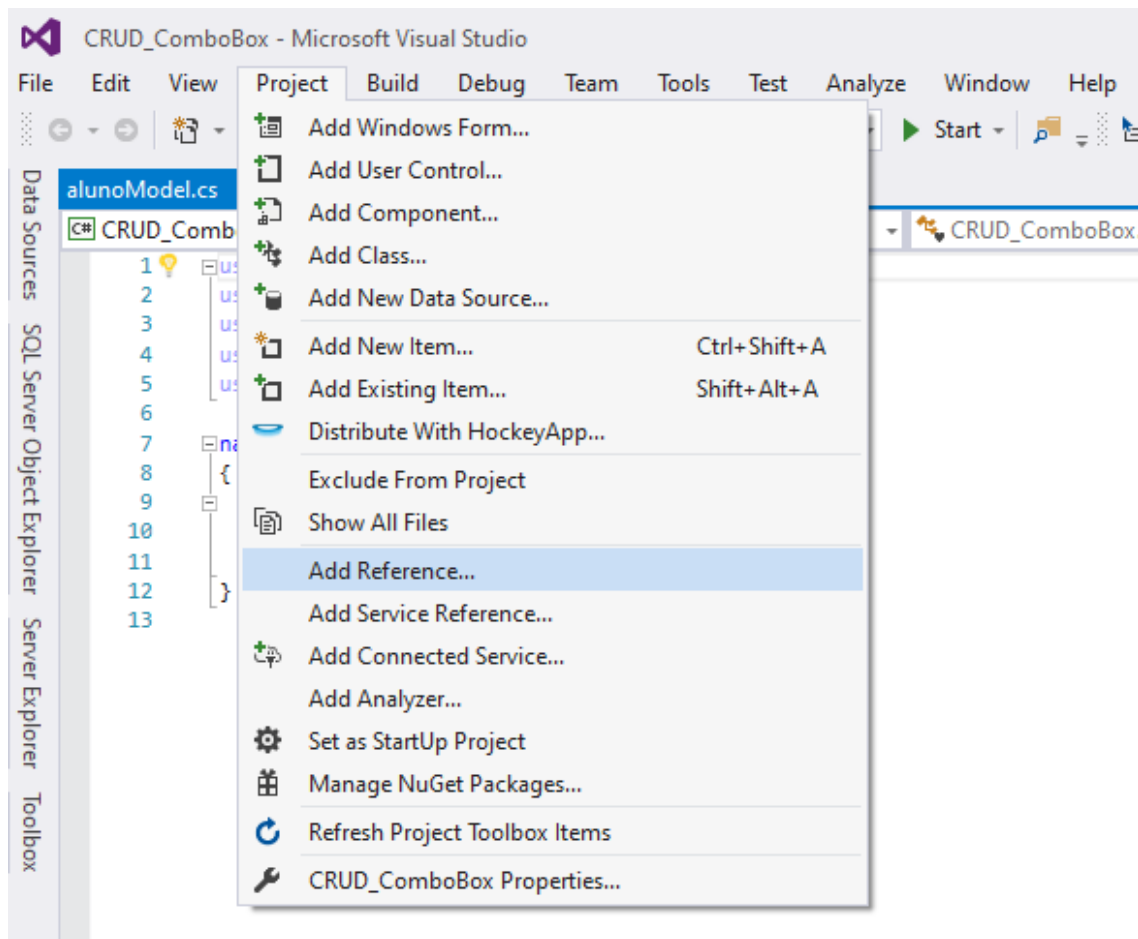


Figura 7. Add Reference

Marque a opção System.Configuration e clique em OK

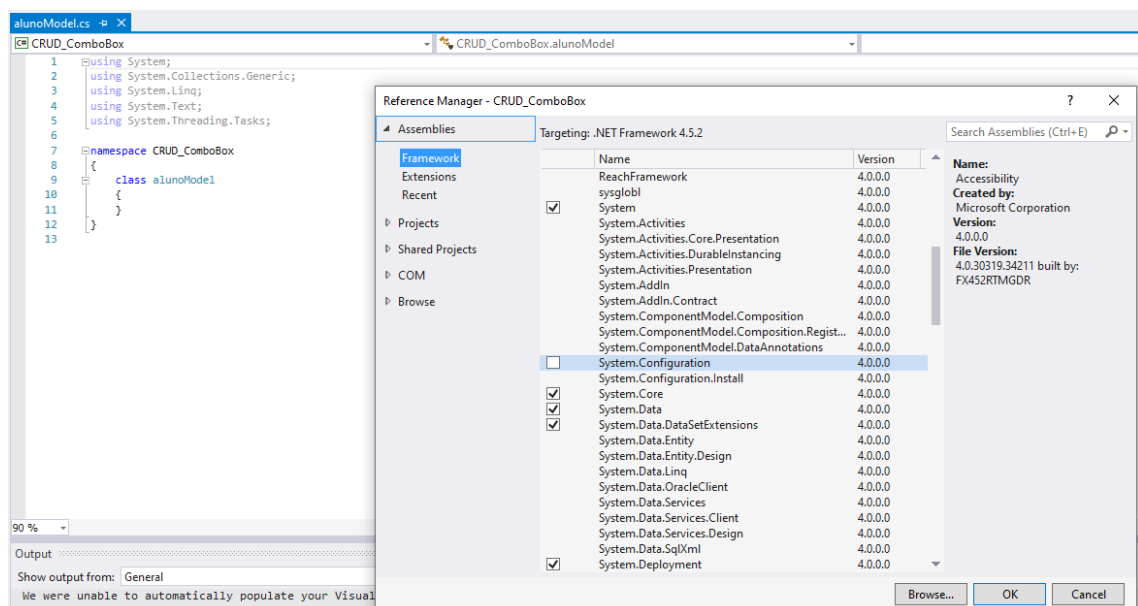


Figura 8. System.Configuration

Agora adicione as três referências

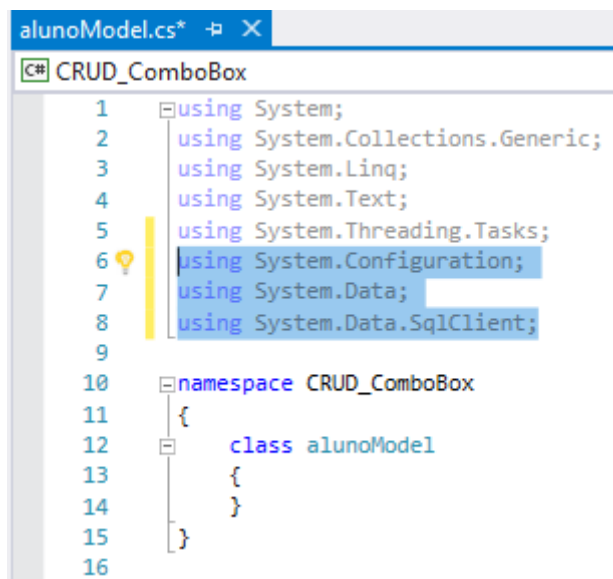


Figura 9. Using System.Configuration

IDisposable

Passa a classe alunoModel para **public** e acrescente o method **IDisposable**

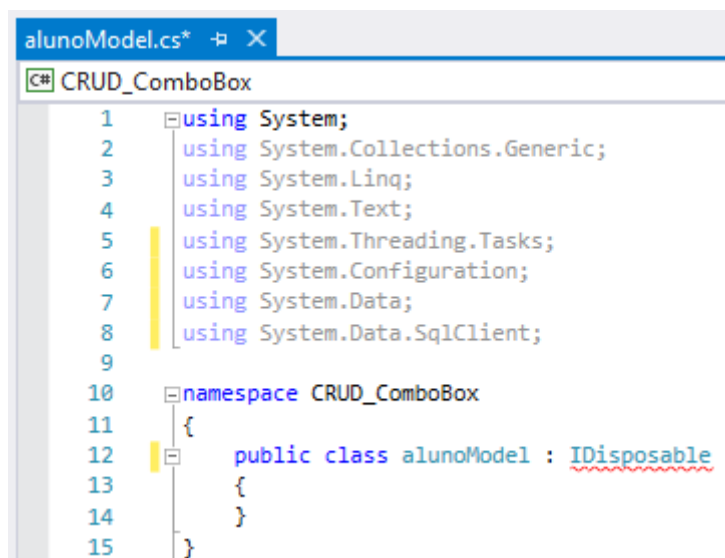


Figura 10. IDisposable

No momento vou colar o código, depois preciso passar parte por parte

O código abaixo serve para persistir dados na tabela alunos do banco db_cadastro

Dentro da classe alunoModel, adicione o código :

```
private static SqlConnection sqlConnection;
private static string sqlConnectionString;
public alunoModel()
{
    private static string DbConnectionString()
    {
        sqlConnectionString =
        ConfigurationManager.ConnectionStrings["CRUD_ComboBox.Properties.Settings.banco"].ConnectionString;
        return sqlConnectionString;
    }
    private static SqlConnection DbConnection()
    {
        sqlConnection = new
        SqlConnection(ConfigurationManager.ConnectionStrings["CRUD_ComboBox.Properties.Settings.banco"].ConnectionString
        );
        sqlConnection.Open();
        return sqlConnection;
    }
    public static DataTable GetAlunos()
    {
        SqlDataAdapter da = null;
        DataTable dt = new DataTable();
        try
        {
            using (var cmd = DbConnection().CreateCommand())
            {
                cmd.CommandText = "SELECT * FROM alunos";
                da = new SqlDataAdapter(cmd.CommandText, DbConnection());
                da.Fill(dt);
                return dt;
            }
        }
        catch (Exception ex)
        {
            throw ex;
        }
    }
    public static DataTable GetAlunoTabela(int id)
    {
        SqlDataAdapter da = null;
        DataTable dt = new DataTable();
        try
        {
            using (var cmd = DbConnection().CreateCommand())
            {
                cmd.CommandText = "SELECT * FROM alunos Where id_aluno=" + id;
                da = new SqlDataAdapter(cmd.CommandText, DbConnection());
                da.Fill(dt);
                return dt;
            }
        }
        catch (Exception ex)
        {
            throw ex;
        }
    }
    public static alunoEnt GetAluno(int id)
    {
        SqlDataAdapter da = null;
        DataTable dt = new DataTable();
        alunoEnt aluno = new alunoEnt();
```

```

try
{
    using (var cmd = DbConnection().CreateCommand())
    {
        cmd.CommandText = "SELECT * FROM alunos Where id_aluno=" + id;
        da = new SqlDataAdapter(cmd.CommandText, DbConnection());
        da.Fill(dt);
        aluno.IdAluno = Convert.ToInt32(dt.Rows[0]["id_aluno"]);
        aluno.Nome = dt.Rows[0]["Nome"].ToString();
        aluno.Endereco = dt.Rows[0]["Endereco"].ToString();
        aluno.Email = dt.Rows[0]["Email"].ToString();
        aluno.Telefone = dt.Rows[0]["Telefone"].ToString();
        return aluno;
    }
}
catch (Exception ex)
{
    throw ex;
}
}

public static void Add(alunoEnt aluno)
{
    try
    {
        using (var cmd = DbConnection().CreateCommand())
        {
            cmd.CommandText = "INSERT INTO alunos(Nome, Endereco,Email,Telefone ) values(@nome, @endereco,
@email, @telefone)";
            cmd.Parameters.AddWithValue("@nome", aluno.Nome);
            cmd.Parameters.AddWithValue("@endereco", aluno.Endereco);
            cmd.Parameters.AddWithValue("@email", aluno.Email);
            cmd.Parameters.AddWithValue("@telefone", aluno.Telefone);
            cmd.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        throw ex;
    }
}

public static void Update (alunoEnt aluno)
{
    try
    {
        using (var cmd = DbConnection().CreateCommand())
        {
            if (aluno != null)
            {
                cmd.CommandText = "UPDATE alunos SET
Nome=@Nome,Email=@Email,Endereco=@Endereco,Telefone = @Telefone WHERE id_aluno = @Id";
                cmd.Parameters.AddWithValue("@Id", aluno.IdAluno);
                cmd.Parameters.AddWithValue("@Nome", aluno.Nome);
                cmd.Parameters.AddWithValue("@Endereco", aluno.Endereco);
                cmd.Parameters.AddWithValue("@Email", aluno.Email);
                cmd.Parameters.AddWithValue("@Telefone", aluno.Telefone);
                cmd.ExecuteNonQuery();
            }
        }
    }
    catch (Exception ex)
    {
        throw ex;
    }
}
}

```



```

public static void Delete(int Id)
{
    try
    {
        using (var cmd = DbConnection().CreateCommand())
        {
            cmd.CommandText = "DELETE FROM alunos Where id_aluno=@Id";
            cmd.Parameters.AddWithValue("@Id", Id);
            cmd.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        throw ex;
    }
}

public void Dispose()
{
    GC.SuppressFinalize(this);
}

```

Formulário

Precisamos inserir:

1 Combobox – cboAlunos

5 TextBox – txtID, txtNome, txtEndereco, txtEmail e txtTelefone

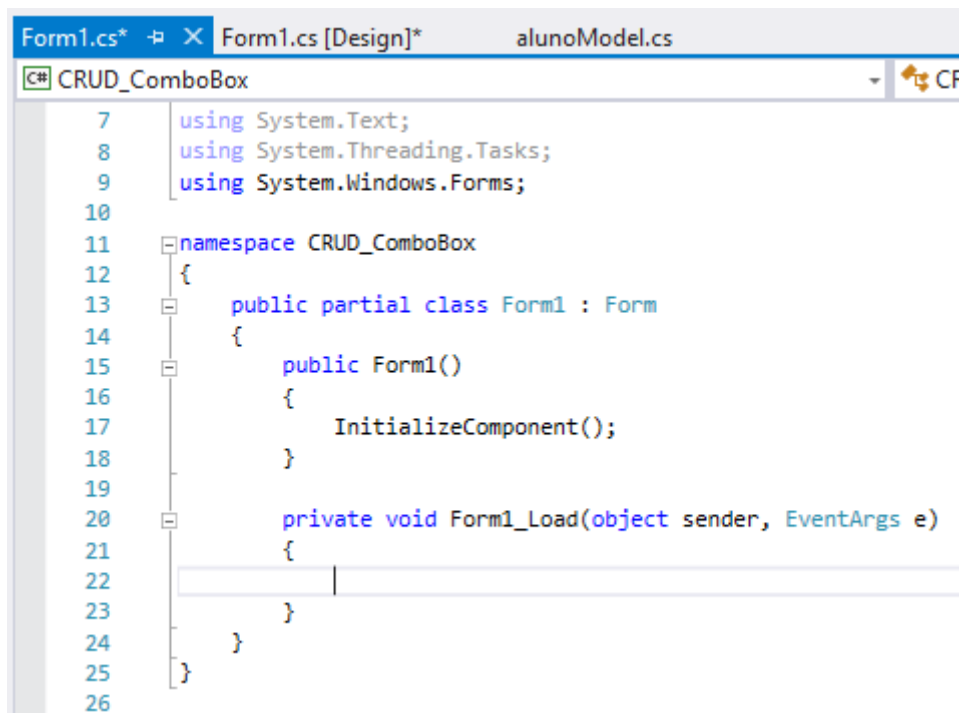
5 Buttons – btnIncluir, btnAlterar, btnExcluir e btnSair

The image shows a Windows Forms application window titled "Alunos". The window has a standard Windows title bar with minimize, maximize, and close buttons. Inside the window, there is a label "Selecione o aluno" above a dropdown menu. Below the dropdown menu, there are five text boxes arranged vertically, each with a label to its left: "ID", "Nome", "Endereço", "E-mail", and "Telefoni". At the bottom of the window, there are four buttons arranged horizontally: "Incluir", "Alterar", "Deletar", and "Sair". The window is currently in the design view, as indicated by the "Form1.cs [Design]*" tab in the background.

Figura 11. Formulário

Codificando Formulário

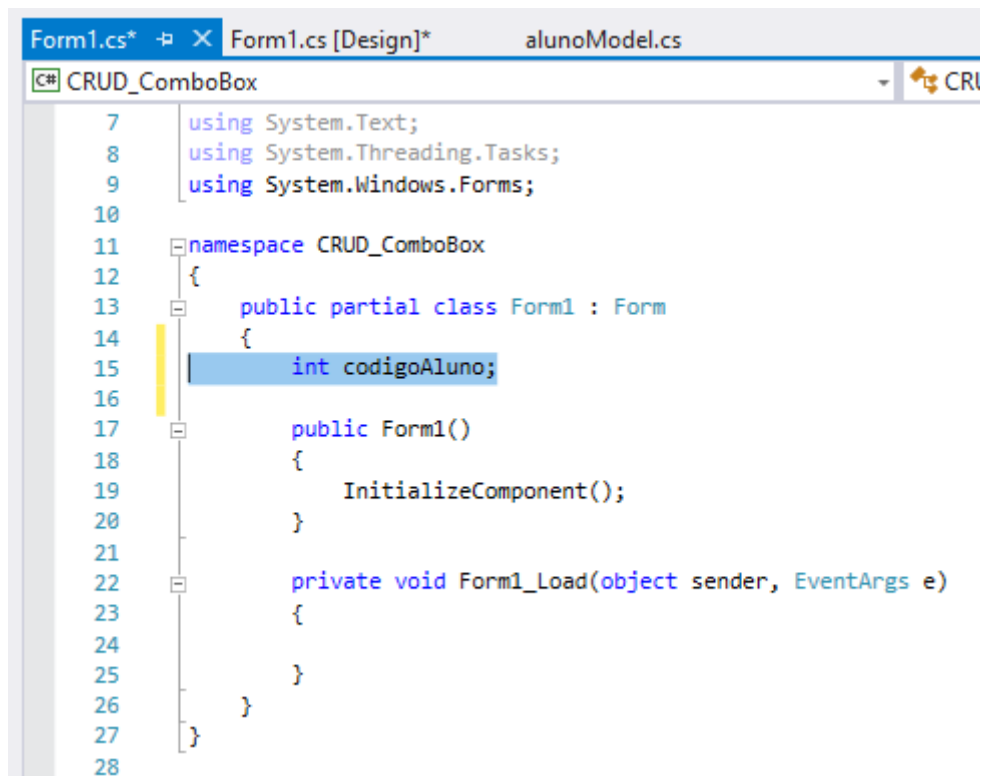
Clique duas vezes no formulário



```
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace CRUD_ComboBox
12 {
13     public partial class Form1 : Form
14     {
15         public Form1()
16         {
17             InitializeComponent();
18         }
19
20         private void Form1_Load(object sender, EventArgs e)
21         {
22
23         }
24     }
25 }
26
```

Figura 12. Código do formulário

Insira uma variável codigoAluno do tipo inteiro



```
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10
11 namespace CRUD_ComboBox
12 {
13     public partial class Form1 : Form
14     {
15         int codigoAluno;
16
17         public Form1()
18         {
19             InitializeComponent();
20         }
21
22         private void Form1_Load(object sender, EventArgs e)
23         {
24         }
25     }
26 }
27
28
```

Figura 13. Variável codigoAluno

Crie o método CarregaDados e faça o evento do formulário chama-lo. O método CarregaDados irá chamar o Método GetAlunos da classe alunoModel e irá definir o IdAluno como filtro.

```
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10
11 namespace CRUD_ComboBox
12 {
13     public partial class Form1 : Form
14     {
15         int codigoAluno;
16
17         public Form1()
18         {
19             InitializeComponent();
20         }
21
22         private void Form1_Load(object sender, EventArgs e)
23         {
24             CarregaDados();
25         }
26
27         private void CarregaDados()
28         {
29             cboAlunos.DataSource = alunoModel.GetAlunos();
30             cboAlunos.ValueMember = "IdAluno";
31             cboAlunos.DisplayMember = "Nome";
32             cboAlunos.Text = "Selecione o aluno";
33             LimpaFormulario();
34         }
35     }
36 }
```

Figura 14. Método CarregaDados

Criando o método LimpaFormulario

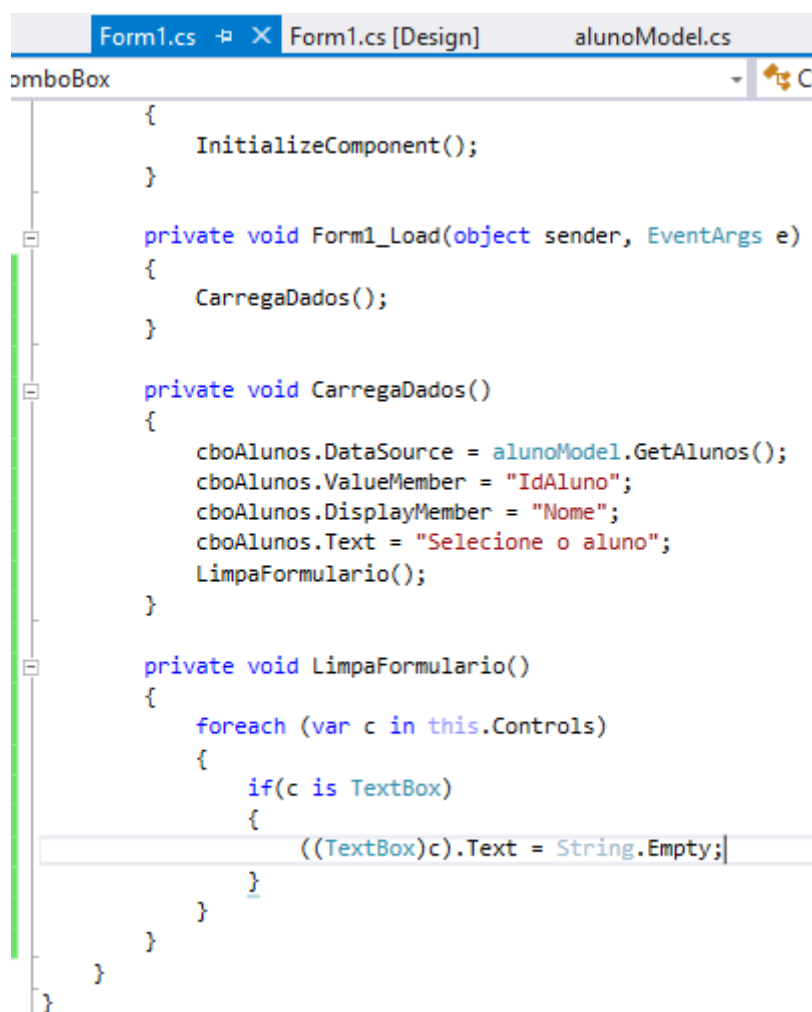


Figura 15. Método LimpaFormulario

Carregando os dados ao selecionar o dado no Combobox

Clique duas vezes no Combobox

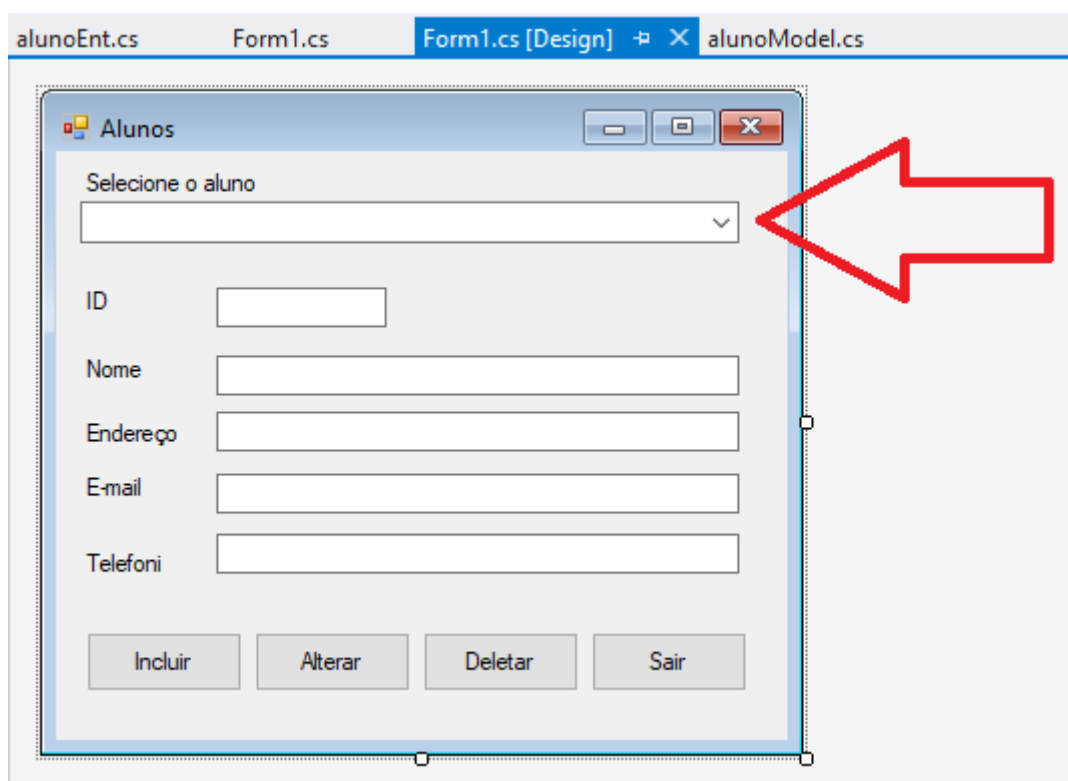


Figura 16. ComboBox

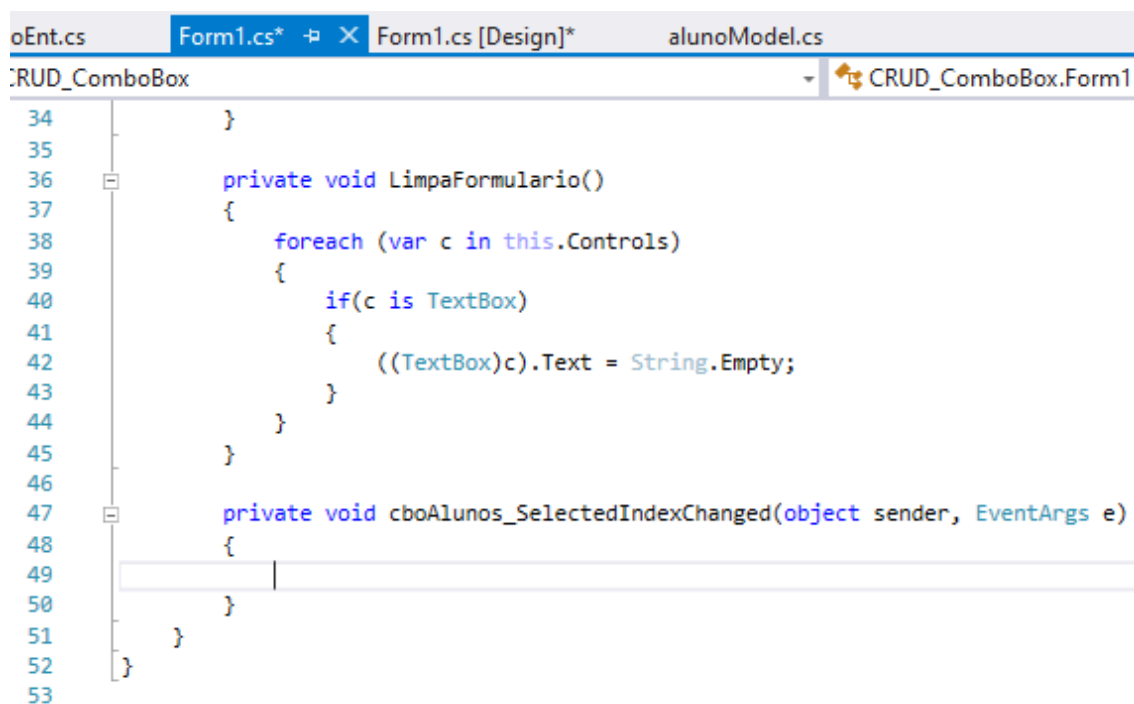


Figura 17. Evento SelectedIndexChanged do ComboBox

Precisamos recuperar o código do aluno através do método `GetAluno` da classe `alunoEnt` para retornar o objeto `alunoEnt`

The screenshot shows a Visual Studio code editor with the following tabs: oEnt.cs, Form1.cs, Form1.cs [Design], and alunoModel.cs. The active file is Form1.cs, and the cursor is at line 66. The code is for a Windows Form application, specifically for a CRUD application. The code is as follows:

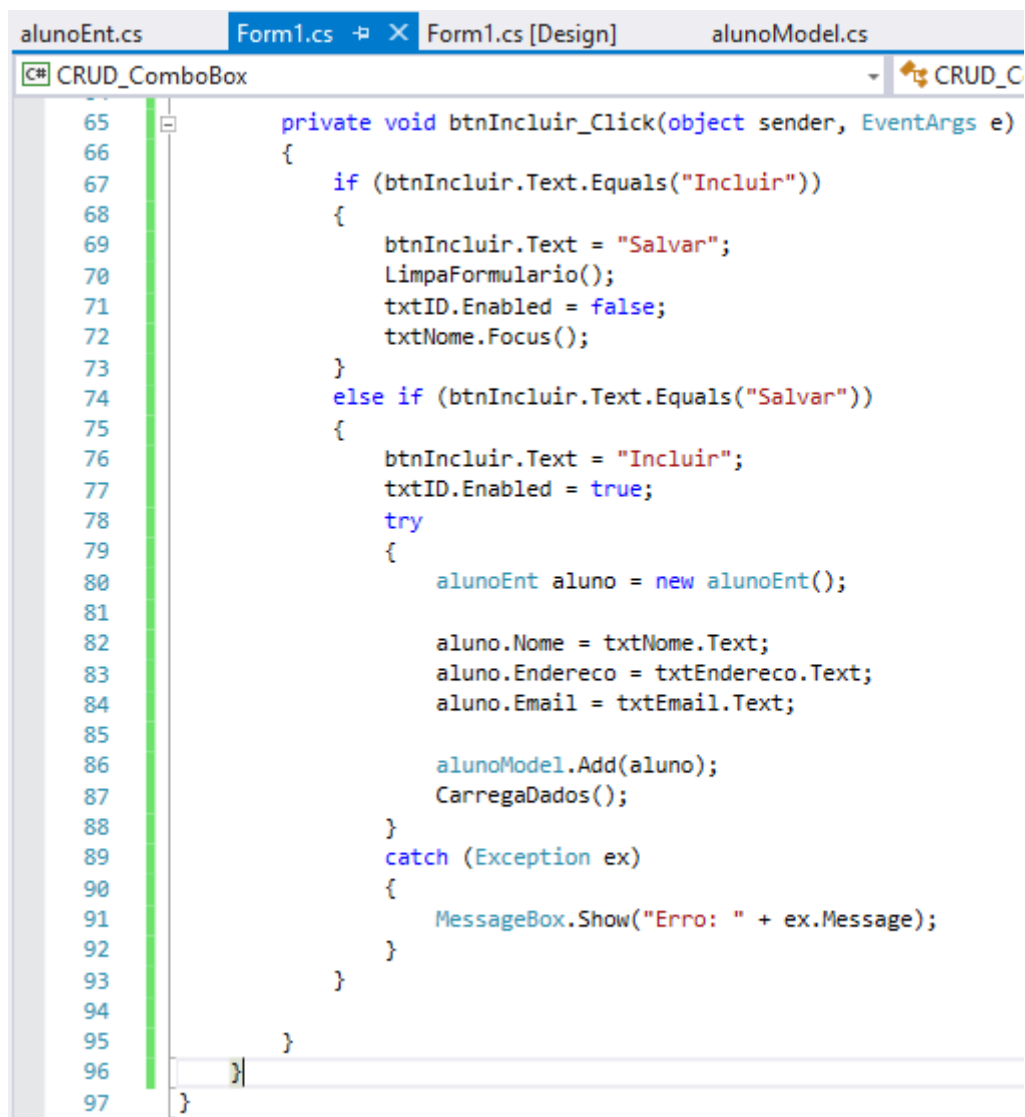
```

46
47     private void cboAlunos_SelectedIndexChanged(object sender, EventArgs e)
48     {
49         alunoEnt aluno = new alunoEnt();
50
51         codigoAluno = Convert.ToInt32(((DataRowView)cboAlunos.SelectedItem)["IdAluno"]);
52
53         aluno = alunoModel.GetAluno(codigoAluno);
54         PreencheDados(aluno);
55     }
56
57     private void PreencheDados(alunoEnt aluno)
58     {
59         txtID.Text = aluno.IdAluno.ToString();
60         txtNome.Text = aluno.Nome;
61         txtEndereco.Text = aluno.Email;
62         txtTelefone.Text = aluno.Telefone;
63     }
64 }
65
66

```

Figura 18. Carregando dados através do combobox

Configurando o botão Incluir. Dê um duplo clique no botão Incluir e insira o código:



```
alunoEnt.cs  Form1.cs  Form1.cs [Design]  alunoModel.cs
C# CRUD_ComboBox  CRUD_C
65 private void btnIncluir_Click(object sender, EventArgs e)
66 {
67     if (btnIncluir.Text.Equals("Incluir"))
68     {
69         btnIncluir.Text = "Salvar";
70         LimpaFormulario();
71         txtID.Enabled = false;
72         txtNome.Focus();
73     }
74     else if (btnIncluir.Text.Equals("Salvar"))
75     {
76         btnIncluir.Text = "Incluir";
77         txtID.Enabled = true;
78         try
79         {
80             alunoEnt aluno = new alunoEnt();
81
82             aluno.Nome = txtNome.Text;
83             aluno.Endereco = txtEndereco.Text;
84             aluno.Email = txtEmail.Text;
85
86             alunoModel.Add(aluno);
87             CarregaDados();
88         }
89         catch (Exception ex)
90         {
91             MessageBox.Show("Erro: " + ex.Message);
92         }
93     }
94 }
95 }
96 }
97 }
```

Figura 19. Configuração do botão incluir


```

Form1.cs*  Form1.cs [Design]*  alunoModel.cs
mbobox
private void btnAlterar_Click(object sender, EventArgs e)
{
    try
    {
        alunoEnt aluno = new alunoEnt();

        aluno.IdAluno = Convert.ToInt32(txtID.Text);
        aluno.Nome = txtNome.Text;
        aluno.Endereco = txtEndereco.Text;
        aluno.Email = txtEmail.Text;
        aluno.Telefone = txtTelefone.Text;

        alunoModel1.Update(aluno);
        CarregaDados();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Erro: " + ex.Message);
    }
}
}

```

Figura 20. Configurando botão alterar

```

Form1.cs*  Form1.cs [Design]*  alunoModel.cs
mbobox
private void btnExcluir_Click(object sender, EventArgs e)
{
    try
    {
        alunoModel1.Delete(codigoAluno);
        CarregaDados();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Erro : " + ex.Message);
    }
}
}

```

Figura 21. Configurando o botão alterar

Diretório no git

<https://github.com/endroni/ComboBoxRefazer.git>

BIBLIOGRAFIA BÁSICA:

PRESSMAN, Roger S. **Engenharia de Software: uma abordagem profissional**. 7. ed. São Paulo: Pearson Makron Books, 2011.

SOMMERVILLE, Ian. **Engenharia de Software**. 8. ed. São Paulo: Pearson Addison Wesley, 2007.

TONSIG, Sergio Luiz. **Engenharia de Software: análise e projeto de sistemas**. 2. ed. Rio de Janeiro: Ciência Moderna, 2008.

BIBLIOGRAFIA COMPLEMENTAR:

BEZERRA, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML**. Rio de Janeiro: Elsevier, 2007.

Macoratti. **C# - CRUD básico no SQL Server usando Combobox/TextBox usando DAL**. Macoratti.net: Tecnologia. Disponível em: <http://www.macoratti.net/18/09/c_crudcbo1.htm>. Acesso em: 11 de outubro de 2019.