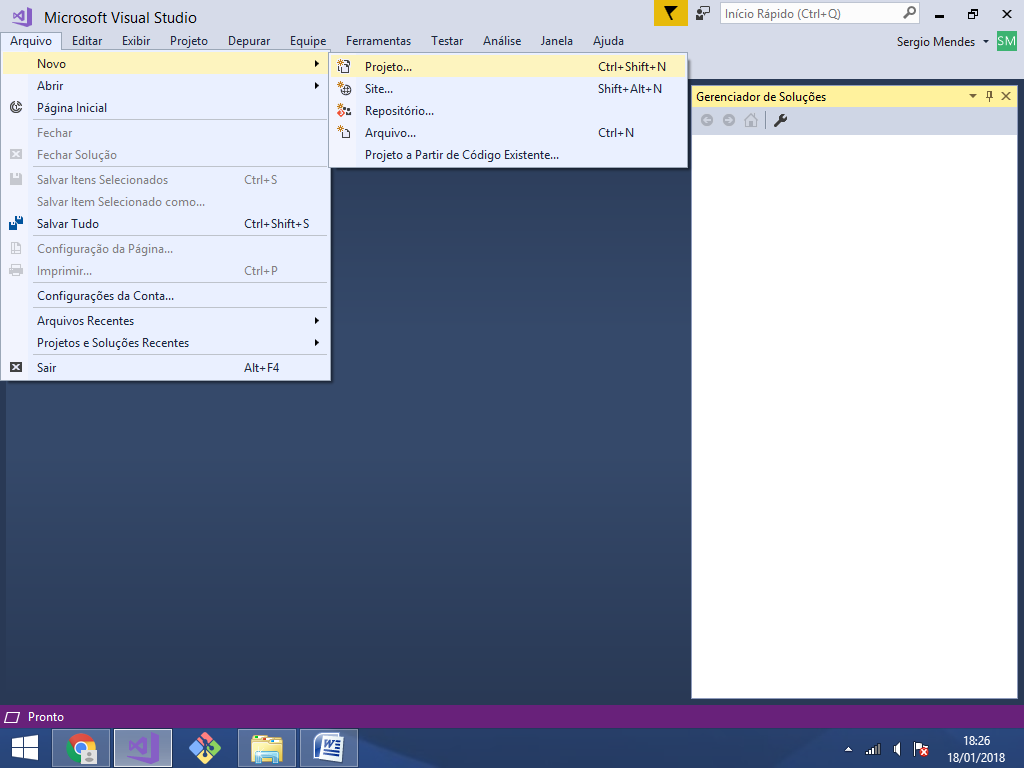
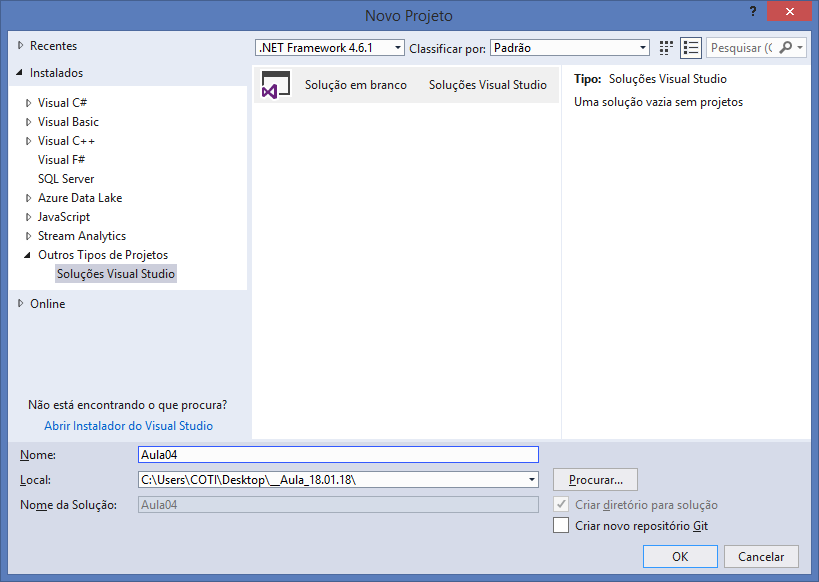
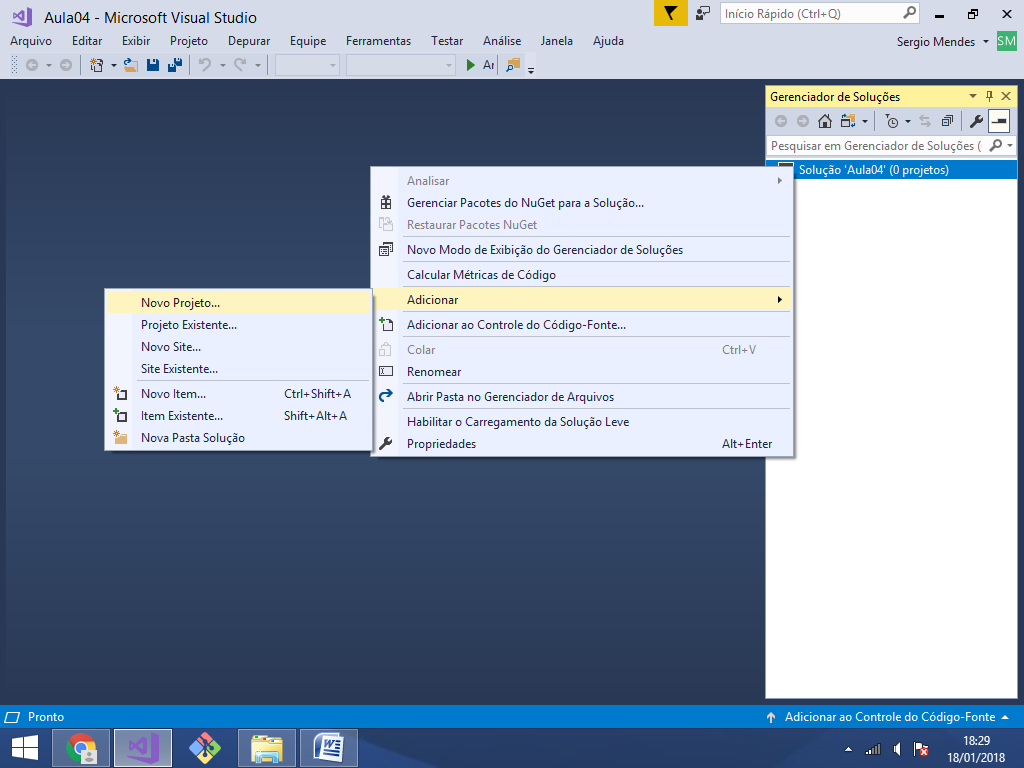
Criando uma nova solution em branco:



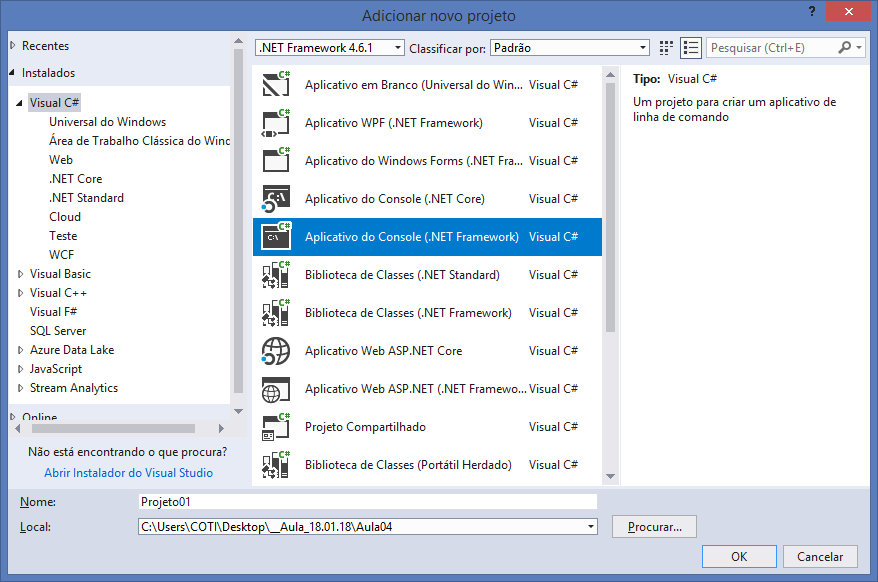


Criando o primeiro projeto **Console Application**

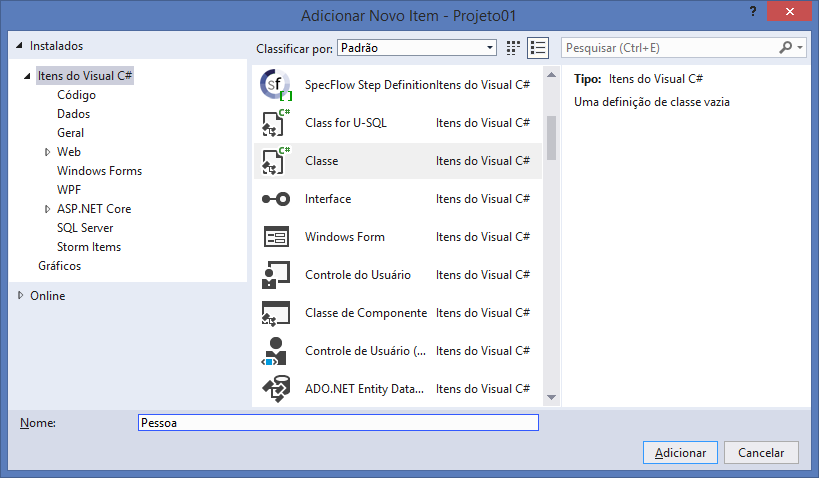


**Aplicativo do Console**

.NET Framework



**Passo 1) Criando uma classe de entidade**



Entidade

* Propriedades implicitamente encapsuladas
* Construtor default
* Construtor com entrada de argumentos (Sobrecarga)
* Sobrescrita de métodos da classe Object

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto01.Entidades

{

public class Pessoa

{

#region Propriedades

public int IdPessoa { get; set; }

public string Nome { get; set; }

public DateTime DataNascimento { get; set; }

#endregion

#region Construtores

//construtor default

public Pessoa()

{

}

//sobrecarga de construtores

public Pessoa(int idPessoa, string nome, DateTime dataNascimento)

{

IdPessoa = idPessoa;

Nome = nome;

DataNascimento = dataNascimento;

}

#endregion

#region Sobrescrita de métodos da classe Object

public override string ToString()

{

return $"Id Pessoa: {IdPessoa}, Nome: {Nome},

Data de Nascimento: {DataNascimento}";

}

#endregion

}

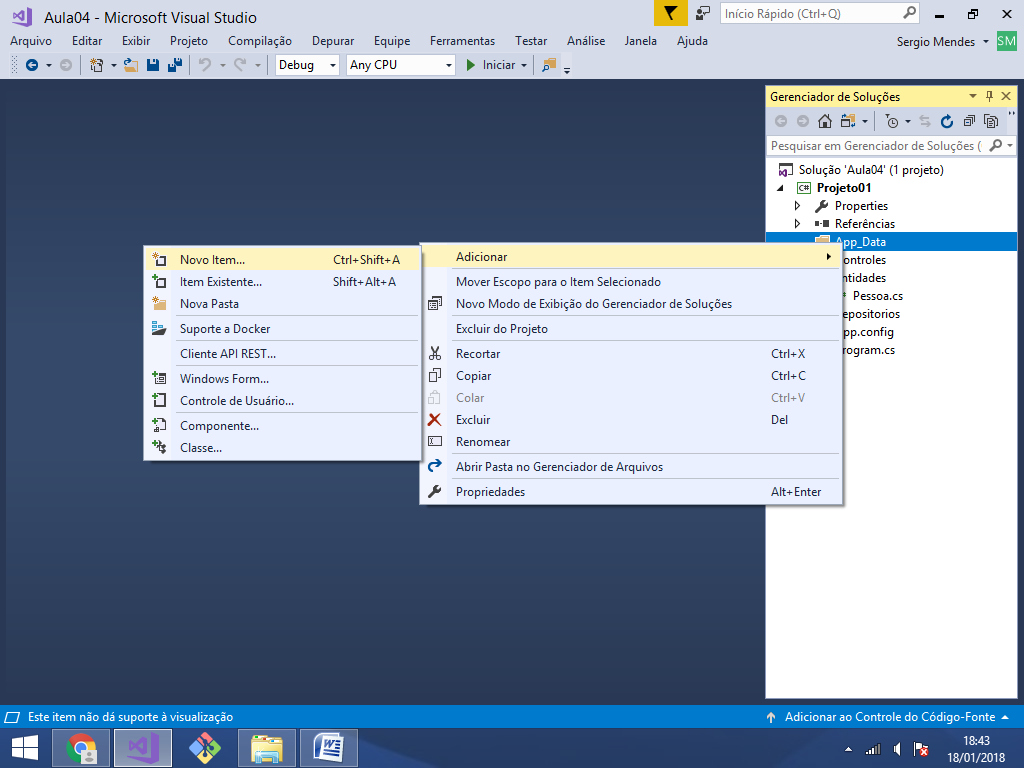
}

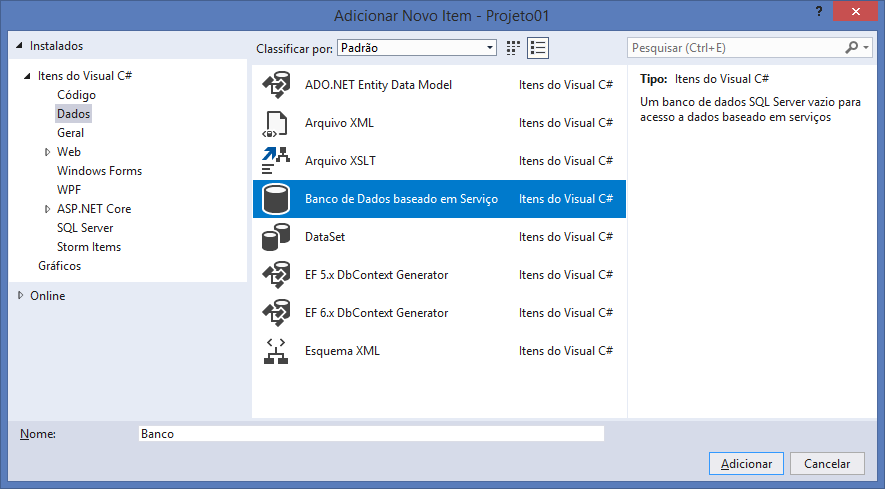
Criando um banco de dados local do SqlServer

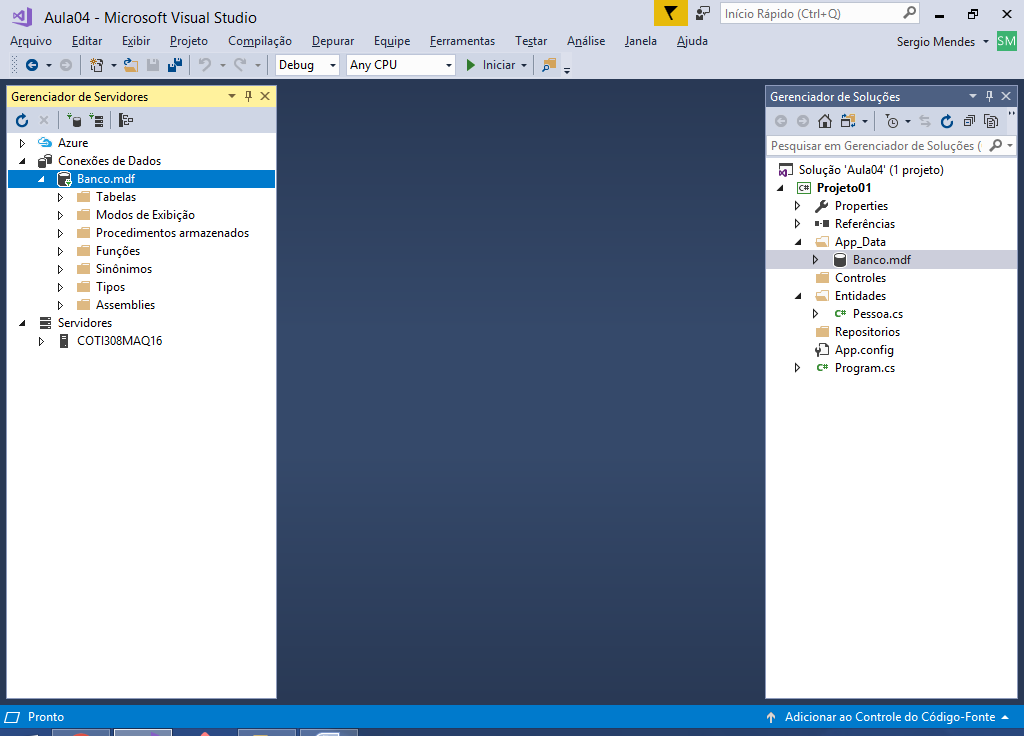
MDF - Master Database File

Tipo de arquivo de banco de dados do SqlServer que pode ser criado de forma local, armazenando a estrutura de um BD dentro de um arquivo.

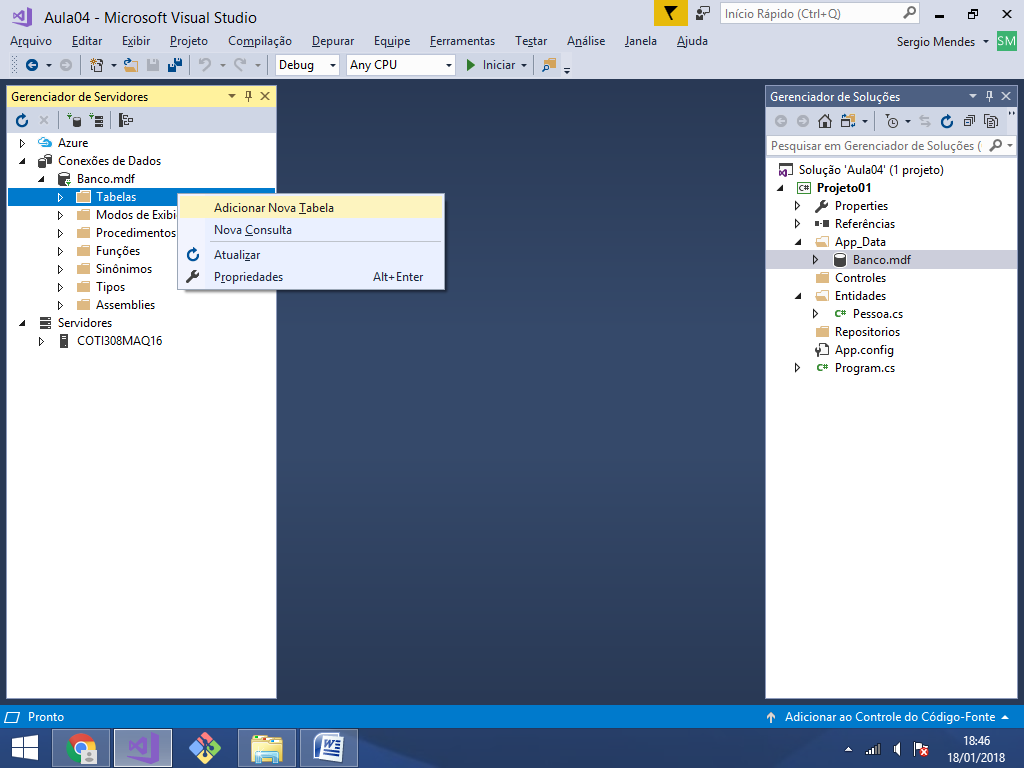
\*\* Por boa prática, este tipo de arquivo é criado no   
projeto dentro de uma pasta denominada **App\_Data**







Criando uma tabela de Pessoa:



create table Pessoa(

IdPessoa integer identity,

Nome nvarchar(50) not null,

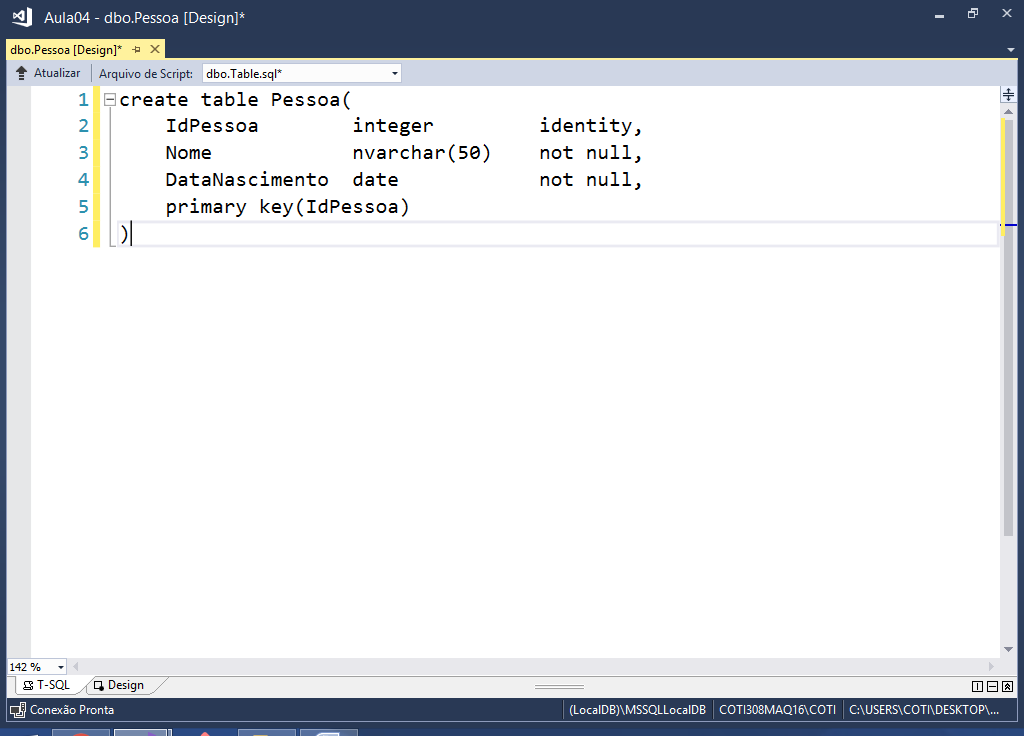
DataNascimento date not null,

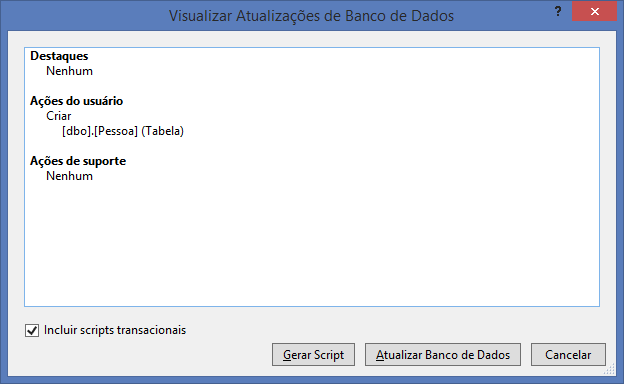
primary key(IdPessoa)

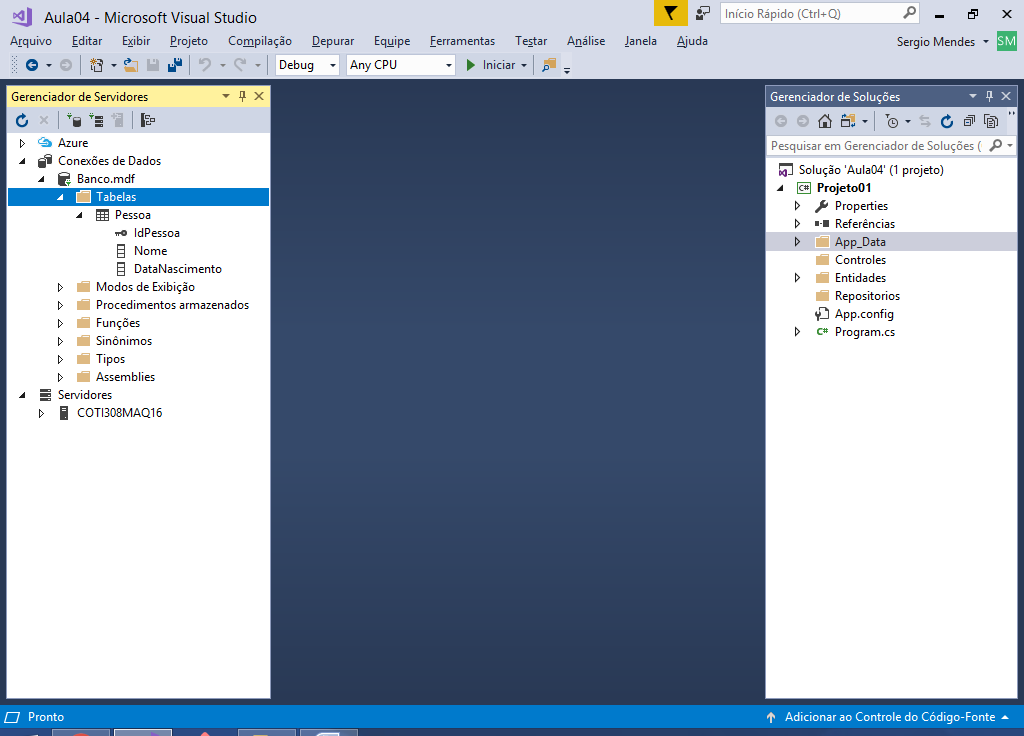
)

---------------------------------------

Para executar:



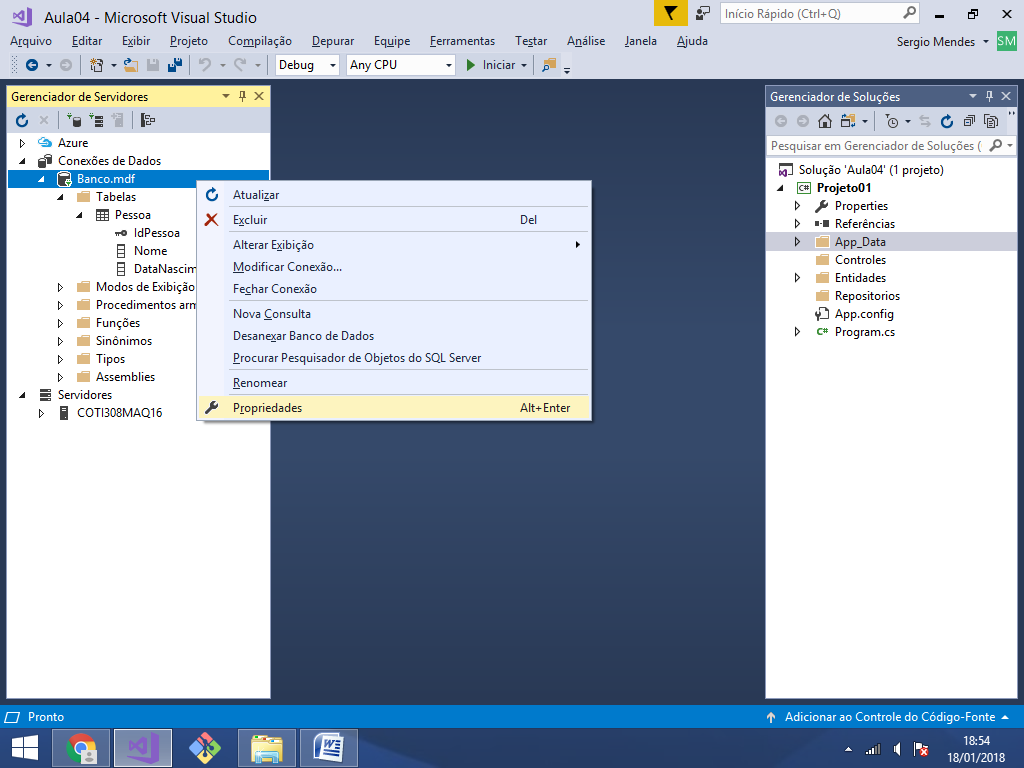


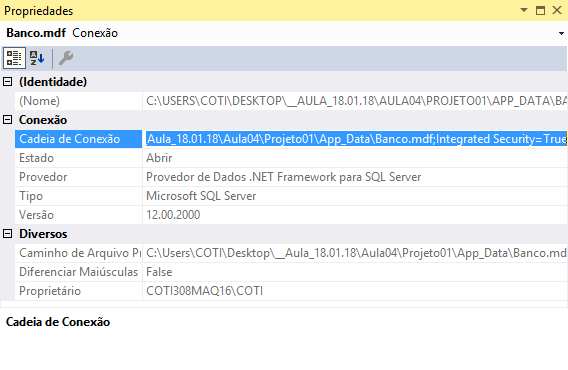


ConnectionString

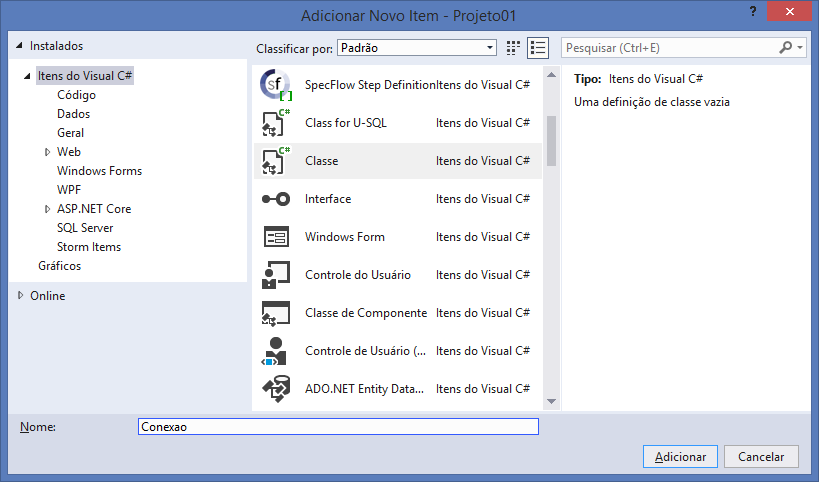
Nome dado para a string que descreve os dados necessarios para   
que uma aplicação NET se conecte a uma base de dados.

Obtendo a connectionstring:





**Criando uma classe para conexão com o banco de dados:**



**System.Data.SqlClient**

Biblioteca do .NET que contem classes para acesso a bases de dados do SqlServer, possui 4 classes principais. São elas:

**SqlConnection**

Classe que permite abrir ou fechar uma conexão com o banco de dados todas as demais dependentes do SqlConnection

**SqlCommand**

Classe que permite executar comandos em linguagem SQL na base de dados, compo exemplo: INSERT, DELETE, UPDATE, SELECT e ate mesmo Stored Procedures.

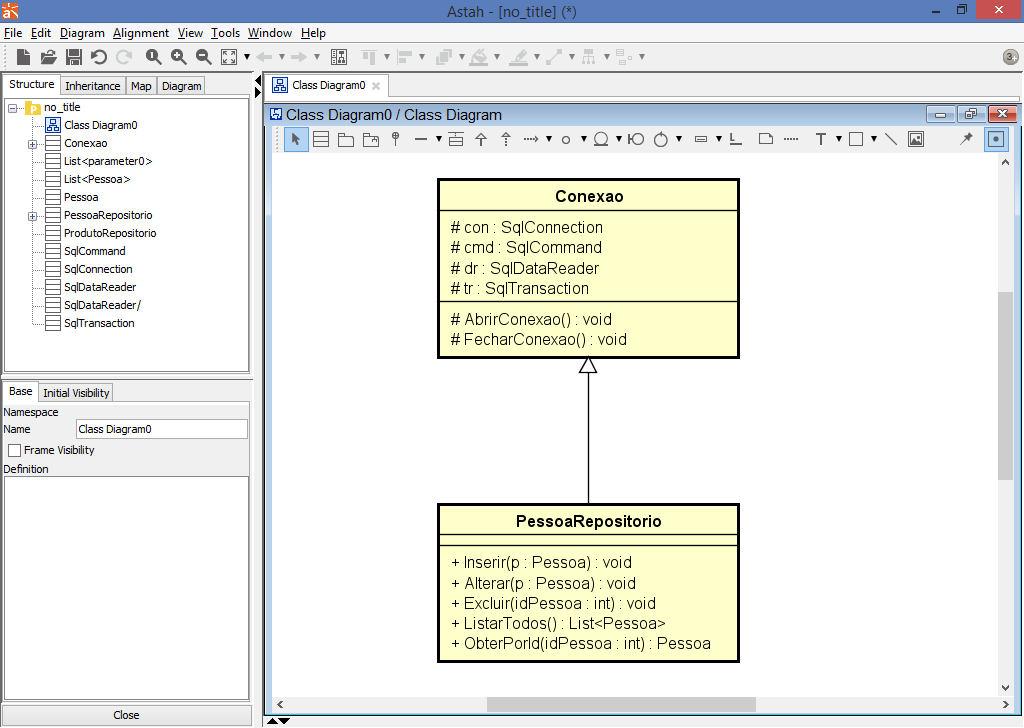
**SqlDataReader**

Classe que permite ler e receber registros provenientes de consultas feitas ao banco de dados. Todo comando executado do tipo SELECT irá precisar de um objeto SqlDataReader para ler suas linhas e colunas de resultados.

**SqlTransaction**

Classe que permite realizar gerenciamento de transações na base de dados, ou seja, executar comandos e ao final fazer o COMMIT (finalizando) ou o ROLLBACK (desfazendo) destes comandos.

**#protected**

Modificador de visibilidade que faz com que um atributo ou método só possa ser acessado por meio de herança. Ou seja, somente as subclasses terão acesso a estes elementos.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

namespace Projeto01.Repositorios

{

public class Conexao

{

#region Atributos

protected SqlConnection con;

protected SqlCommand cmd;

protected SqlDataReader dr;

protected SqlTransaction tr;

#endregion

#region Métodos

protected void AbrirConexao()

{

con = new SqlConnection();

con.Open(); //conectado..

}

protected void FecharConexao()

{

con.Close(); //desconectado..

}

#endregion

}

}

-------------------------------------------

Para que o SqlConnection possa abrir uma conexão com a base de dados é necessário que indiquemos a connectionstring do banco.

O ideal é que o caminho da connectionstring fique mapeado no arquivo App.config.xml do projeto

A Classe de conexão só irá referenciar para o SqlConnection o nome do mapeamento feitono arquivo App.config.xml

\App.config.xml

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />

</startup>

**<connectionStrings>**

**<add**

**name="aula04"**

**connectionString="Data Source=(LocalDB)\MSSQLLocalDB;**

**AttachDbFilename=C:\Users\COTI\Desktop\\_\_Aula\_18.01.18\**

**Aula04\Projeto01\App\_Data\Banco.mdf;Integrated Security=True"**

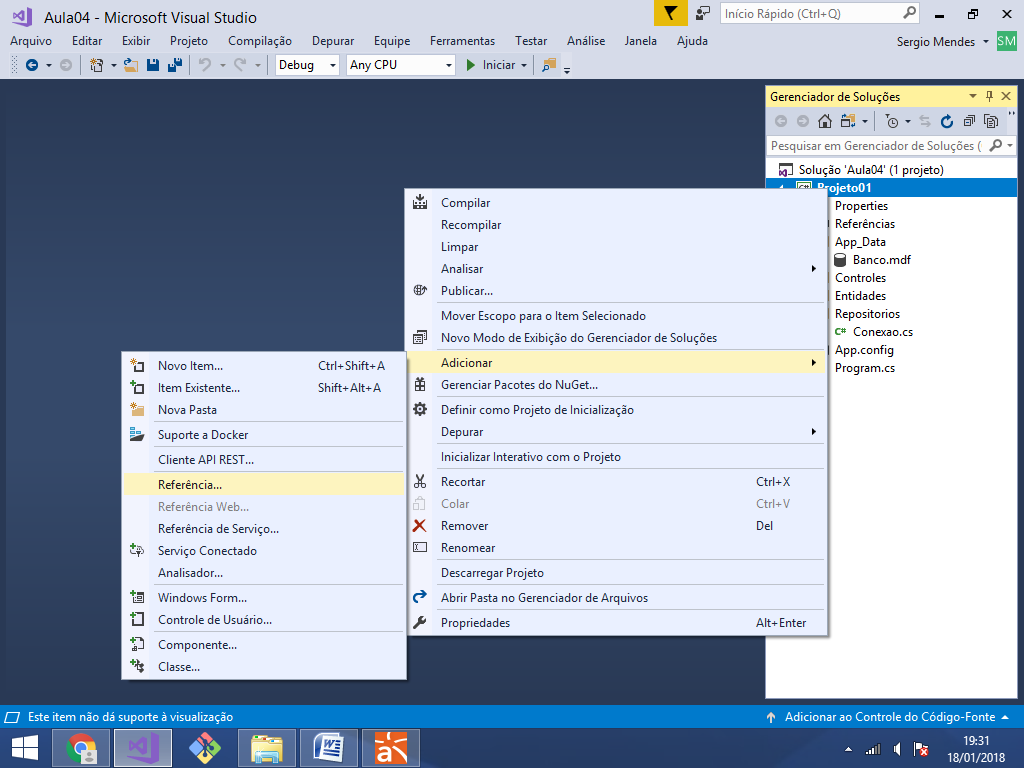
**/>**

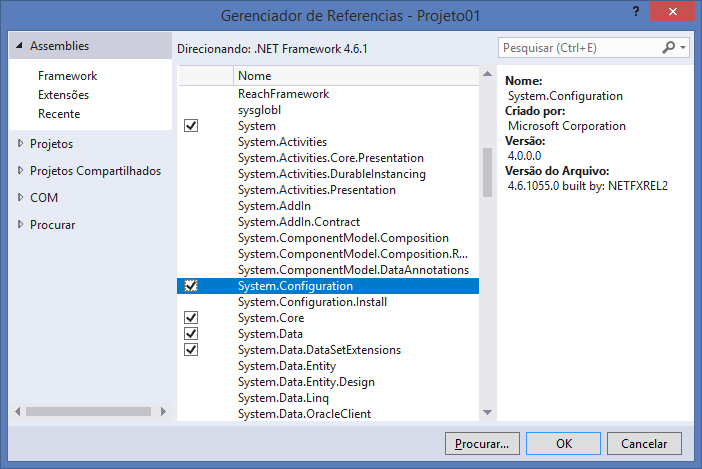
**</connectionStrings>**

</configuration>

-----------------------

Para que possamos referenciar a connectionstring mapeada no App.config.xml é necessário incluir no projeto uma referencia para a biblioteca: System.Configuration





Voltando na classe Conexão:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Configuration;

namespace Projeto01.Repositorios

{

public class Conexao

{

#region Atributos

protected SqlConnection con;

protected SqlCommand cmd;

protected SqlDataReader dr;

protected SqlTransaction tr;

#endregion

#region Métodos

protected void AbrirConexao()

{

con = new SqlConnection(ConfigurationManager.ConnectionStrings

["aula04"].ConnectionString);

con.Open(); //conectado..

}

protected void FecharConexao()

{

con.Close(); //desconectado..

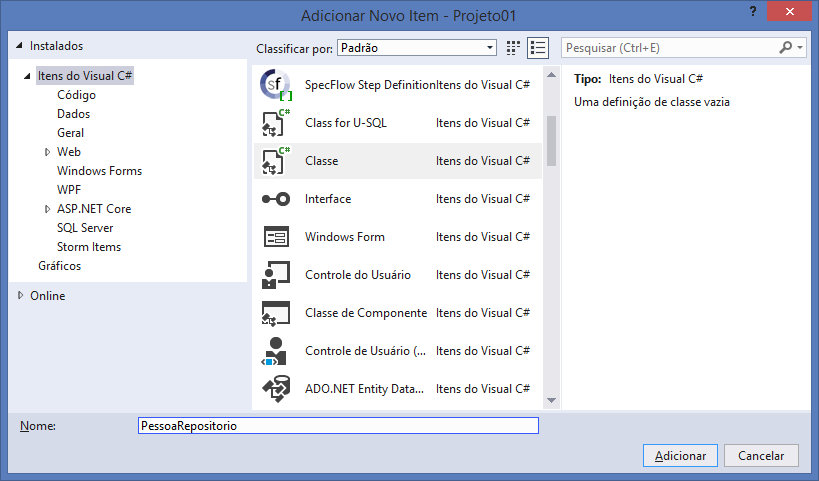
}

#endregion

}

}

Criando uma classe que irá herdar a classe Conexão e realizar as operações no banco de dados com a tabela de Pessoa...



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient; //importando..

using Projeto01.Entidades; //classes de entidade..

namespace Projeto01.Repositorios

{

public class PessoaRepositorio : Conexao

{

//método para inserir registro na tabela de pessoa..

public void Inserir(Pessoa p)

{

AbrirConexao();

//escrever o comando sql (query)..

string query = "insert into Pessoa(Nome, DataNascimento) "

+ "values(@Nome, @DataNascimento)";

//instanciando o atributo SqlCommand da classe Conexao..

cmd = new SqlCommand(query, con);

cmd.Parameters.AddWithValue("@Nome", p.Nome);

cmd.Parameters.AddWithValue("@DataNascimento", p.DataNascimento);

cmd.ExecuteNonQuery(); //executando a query no banco de dados..

FecharConexao();

}

//método para atualizar um registro na tabela de pessoa..

public void Alterar(Pessoa p)

{

AbrirConexao();

//escrever o comando sql (query)..

string query = "update Pessoa set Nome = @Nome,

DataNascimento = @DataNascimento "

+ "where IdPessoa = @IdPessoa";

cmd = new SqlCommand(query, con);

cmd.Parameters.AddWithValue("@IdPessoa", p.IdPessoa);

cmd.Parameters.AddWithValue("@Nome", p.Nome);

cmd.Parameters.AddWithValue("@DataNascimento", p.DataNascimento);

cmd.ExecuteNonQuery(); //executando..

FecharConexao();

}

//método para excluir 1 registro na tabela de pessoa..

public void Excluir(int idPessoa)

{

AbrirConexao();

string query = "delete from Pessoa where IdPessoa = @IdPessoa";

cmd = new SqlCommand(query, con);

cmd.Parameters.AddWithValue("@IdPessoa", idPessoa);

cmd.ExecuteNonQuery(); //executando..

FecharConexao();

}

//método para listar todos os registros da tabela pessoa..

public List<Pessoa> ListarTodos()

{

AbrirConexao();

string query = "select \* from Pessoa";

cmd = new SqlCommand(query, con);

dr = cmd.ExecuteReader();

//declarando uma lista do tipo Pessoa..

List<Pessoa> lista = new List<Pessoa>();

//percorrer cada linha de registro obtido..

while(dr.Read()) //enquanto houver registro..

{

Pessoa p = new Pessoa();

p.IdPessoa = Convert.ToInt32(dr["IdPessoa"]);

p.Nome = Convert.ToString(dr["Nome"]);

p.DataNascimento = Convert.ToDateTime(dr["DataNascimento"]);

lista.Add(p); //adicionar na lista..

}

FecharConexao();

return lista; //retornando a lista..

}

//método para retornar 1 pessoa pelo id..

public Pessoa ObterPorId(int idPessoa)

{

AbrirConexao();

string query = "select \* from Pessoa where IdPessoa = @IdPessoa";

cmd = new SqlCommand(query, con);

cmd.Parameters.AddWithValue("@IdPessoa", idPessoa);

dr = cmd.ExecuteReader();

Pessoa p = null; //sem espaço de memória..

if(dr.Read()) //se algum registro foi encontrado..

{

p = new Pessoa(); //instanciando..

p.IdPessoa = Convert.ToInt32(dr["IdPessoa"]);

p.Nome = Convert.ToString(dr["Nome"]);

p.DataNascimento = Convert.ToDateTime(dr["DataNascimento"]);

}

FecharConexao();

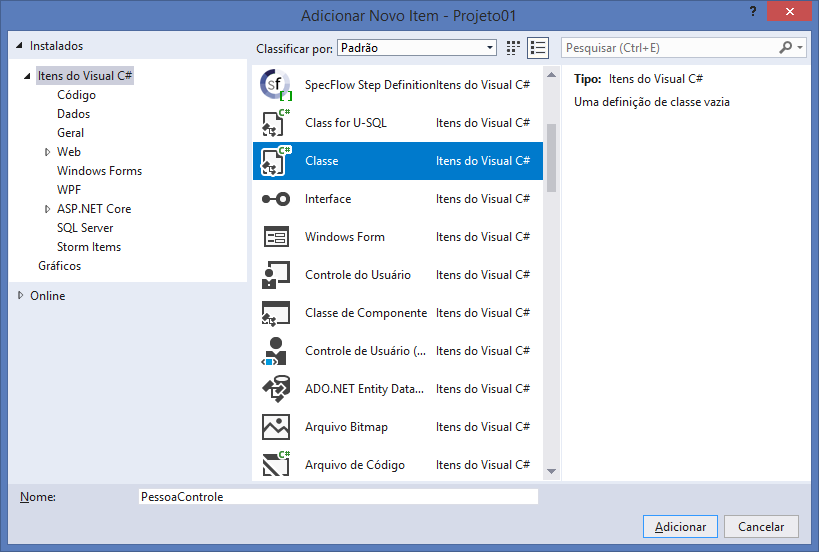
return p; //retornando pessoa..

}

}

}

**Criando uma classe de controle para   
executar os métodos do repositorio:**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto01.Entidades;

using Projeto01.Repositorios;

namespace Projeto01.Controles

{

public class PessoaControle

{

//método para executar o cadastro da pessoa..

public void CadastrarPessoa()

{

try

{

Console.WriteLine("\n - CADASTRO DE PESSOA - \n");

Pessoa p = new Pessoa();

Console.Write("Informe o nome..............: ");

p.Nome = Console.ReadLine();

Console.Write("Informe a data de nascimento.:");

p.DataNascimento = DateTime.Parse(Console.ReadLine());

PessoaRepositorio rep = new PessoaRepositorio();

rep.Inserir(p);

Console.WriteLine("\nPessoa cadastrado com sucesso.");

}

catch(Exception e)

{

Console.WriteLine("Erro: " + e.Message);

}

}

//método para executar a alteração da pessoa..

public void AlterarPessoa()

{

try

{

Console.WriteLine("\n - ALTERAÇÂO DE PESSOA - \n");

Pessoa p = new Pessoa();

Console.Write("Informe o id da pessoa......: ");

p.IdPessoa = int.Parse(Console.ReadLine());

Console.Write("Informe o nome..............: ");

p.Nome = Console.ReadLine();

Console.Write("Informe a data de nascimento.:");

p.DataNascimento = DateTime.Parse(Console.ReadLine());

PessoaRepositorio rep = new PessoaRepositorio();

rep.Alterar(p);

Console.WriteLine("\nPessoa atualizado com sucesso.");

}

catch (Exception e)

{

Console.WriteLine("Erro: " + e.Message);

}

}

//método para excluir a alteração da pessoa..

public void ExcluirPessoa()

{

try

{

Console.WriteLine("\n - EXCLUSÃO DE PESSOA - \n");

Console.Write("Informe o id da pessoa......: ");

int idPessoa = int.Parse(Console.ReadLine());

PessoaRepositorio rep = new PessoaRepositorio();

rep.Excluir(idPessoa);

Console.WriteLine("\nPessoa excluido com sucesso.");

}

catch (Exception e)

{

Console.WriteLine("Erro: " + e.Message);

}

}

//método para exibir a consulta de pessoa..

public void ConsultarPessoas()

{

try

{

Console.WriteLine("\n - CONSULTA DE PESSOAS -\n");

PessoaRepositorio rep = new PessoaRepositorio();

List<Pessoa> lista = rep.ListarTodos();

foreach(Pessoa p in lista)

{

Console.WriteLine("Pessoa: " + p.ToString());

}

}

catch(Exception e)

{

Console.WriteLine("Erro: " + e.Message);

}

}

//método para buscar 1 pessoa pelo id..

public void ObterPessoaPorId()

{

try

{

Console.WriteLine("\n - OBTER PESSOA POR ID - \n");

Console.Write("Informe o id da pessoa......: ");

int idPessoa = int.Parse(Console.ReadLine());

PessoaRepositorio rep = new PessoaRepositorio();

Pessoa p = rep.ObterPorId(idPessoa);

if(p != null) //pessoa encontrado..

{

Console.WriteLine("Pessoa: " + p.ToString());

}

else

{

Console.WriteLine("Pessoa não encontrado");

}

}

catch(Exception e)

{

Console.WriteLine("Erro: " + e.Message);

}

}

}

}

Voltando na classe Program

**Método Main()**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto01.Controles;

namespace Projeto01

{

class Program

{

static void Main(string[] args)

{

try

{

Console.WriteLine("\n - CONTROLE DE CLIENTES - \n");

Console.WriteLine("1) Cadastrar");

Console.WriteLine("2) Alterar");

Console.WriteLine("3) Excluir");

Console.WriteLine("4) Consultar Todos");

Console.WriteLine("5) Consultar por Id");

Console.Write("\nInforme a opção: ");

int opcao = int.Parse(Console.ReadLine());

PessoaControle pc = new PessoaControle();

switch(opcao)

{

case 1: pc.CadastrarPessoa(); break;

case 2: pc.AlterarPessoa(); break;

case 3: pc.ExcluirPessoa(); break;

case 4: pc.ConsultarPessoas(); break;

case 5: pc.ObterPessoaPorId(); break;

}

}

catch(Exception e)

{

Console.WriteLine(e.Message);

}

Console.ReadKey();

}

}

}

