CEUB

EDUCAÇÃO SUPERIOR

Programação Orientada a Objetos

ceub.br

CEUB
EDUCAÇÃO SUPERIOR

Programação Orientada a Objetos

Aula 08 – Persistencia de Dados com C#

Agenda

- **Persistência de Dados com C#**
- **ADO**
- **Entity Framework Core**

CEUB
EDUCAÇÃO SUPERIOR

Introdução a ADO.NET (Banco de Dados)

Introdução a ADO.NET

O ADO.NET é um conjunto de classes que expõem serviços de acesso a dados para desenvolvedores do .NET Framework.

O ADO.NET fornece um conjunto rico de componentes para criar aplicativos distribuídos e de compartilhamento de dados. Faz parte do .NET Framework, fornecendo acesso a dados de aplicativo relacionais e XML.

Introdução a ADO.NET

O ADO.NET oferece suporte a uma variedade de necessidades de desenvolvimento, incluindo a criação de clientes front-end de banco de dados e objetos comerciais de camada intermediária usados por aplicativos, ferramentas, linguagens ou navegadores da Internet.

O ADO.NET fornece acesso consistente a fontes de dados como o SQL Server e o XML, e a fontes de dados expostas através do OLE DB e do ODBC.

Introdução a ADO.NET

Os aplicativos do consumidor de compartilhamento de dados podem usar o ADO.NET para se conectar a essas fontes de dados, e para recuperar, manipular e atualizar os dados nelas contidos.

O ADO.NET separa o acesso a dados da manipulação de dados em componentes discretos que podem ser usados separadamente ou em conjunto.

Introdução a ADO.NET

O ADO.NET inclui os provedores de dados do .NET Framework para se conectar a um banco de dados, executar comandos e recuperar resultados.

Os resultados são processados diretamente, colocados em um objeto DataSet do ADO.NET para serem expostos para o usuário ad hoc, combinados com dados de várias fontes ou passados entre as camadas.

Introdução a ADO.NET

O objeto DataSet também pode ser usado independentemente de um provedor de dados .NET Framework para gerenciar o local dos dados para o aplicativo ou originado no XML.

As classes do ADO.NET estão no System.Data.dll e são integradas às classes XML encontradas no System.Xml.dll.

ADO .NET oferece suporte a uma variedade de opções para desenvolvimento de soluções com acesso a dados que permitem a comunicação com qualquer fonte de dados, desde os já conhecidos gerenciadores de banco de dados relacionais (SGBD) como : SQL Server, MySQL, FireBird, Oracle, Sybase, Access, XML, arquivos textos, etc.

Introdução a ADO.NET

ADO .NET oferece suporte a uma variedade de opções para desenvolvimento de soluções com acesso a dados que permitem a comunicação com qualquer fonte de dados, desde os já conhecidos gerenciadores de banco de dados relacionais (SGBD) como :

- **SQL Server;**
- **MySQL;**
- **FireBird;**
- **Oracle;**
- **Sybase;**
- **Access;**
- **XML;**
- **arquivos textos, etc.**

Introdução a ADO.NET - Qual a situação atual

Acompanhar a evolução do .NET Framework fez com que o ADO.NET evoluísse junto.

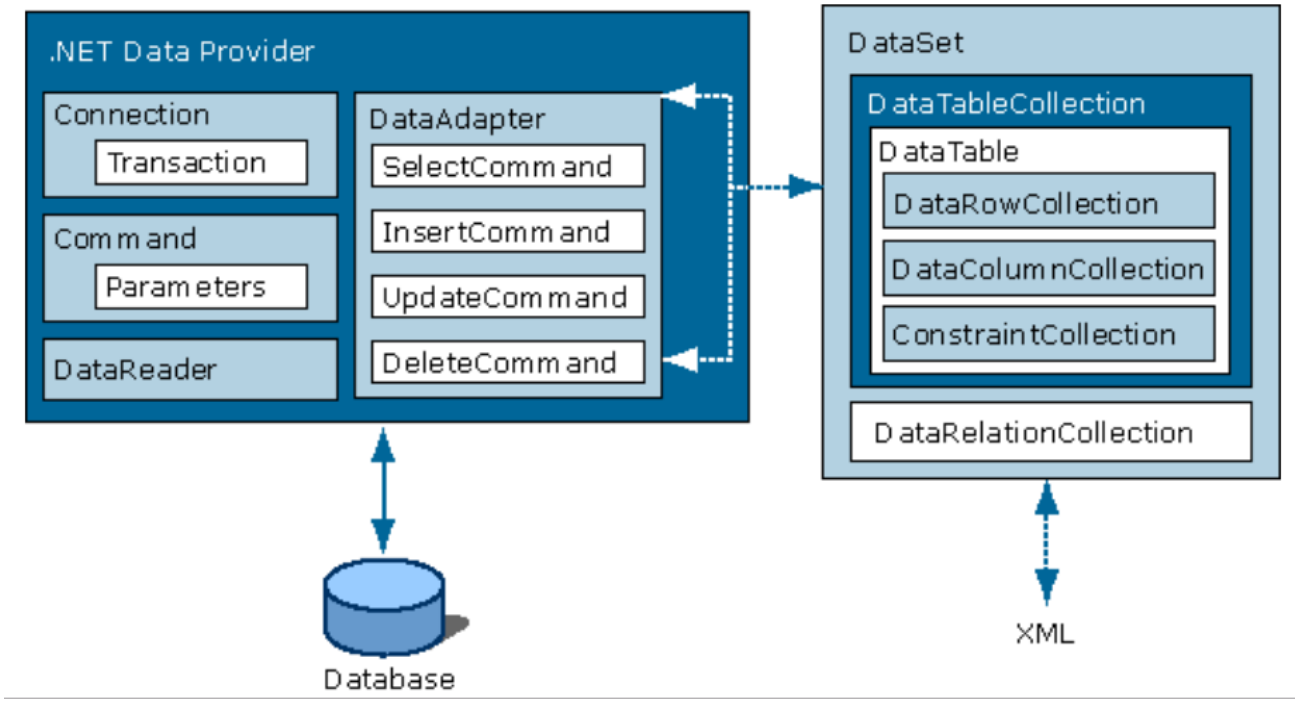
Hoje, não contamos somente com o ADO.NET como tecnologia de acesso e manipulação de dados presente na plataforma

Entre outras tecnologias de acesso e manipulação de dados, podemos citar o LINQ to SQL e o ADO.NET Entity Framework.

Ambas são tecnologias de mapeameto objeto-relacional, sendo que a primeira funciona somente com o SGBD SQL Server.

Arquitetura do ADO.NET

A arquitetura do ADO.NET está dividida em dois grupos: Manager Providers (provedores gerenciados) e Content Components (componentes de conteúdo)



Data Provider

Os providers do ADO.NET são pacotes de classes que possibilitam a interação com uma fonte de dados específica, como por exemplo, com uma base SQL Server;

Interfaces principais: IDbConnection, IDbCommand e IDbDataReader, IDbDataTransaction e IDbDataAdapter;

A partir destas interfaces qualquer fabricante de banco de dados pode criar componentes (um pacote) que implementam as mesmas para permitir o acesso ao seu SGBD;

Cada pacote possui um prefixo que indica qual a fonte de dados que este pacote suporta;

CEUB
EDUCAÇÃO SUPERIOR

Data Provider

Para utilizar o Data Providers devem ser importados os seguintes Namespaces do .Net Framework

Data Provider	Namespace
SQL Server	System.Data.**Sql**Client
OLE DB	System.Data.**OleDb**
ODBC	System.Data.**Odbc**
Oracle	System.Data.**Oracle**Client

Data Providers

- **Data Provider SQL Server**

- **Constituem a seção núcleo da arquitetura ADO.NET, permitindo comunicação entre as aplicações e fontes de dados.**

- **Um Data Provider permite que você se conecte com uma fonte de dados, capture e manipule dados e,por fim, atualize a fonte de dados.**

CEUB
EDUCAÇÃO SUPERIOR

Data Providers

Importando a classe System.Data.SqlClient

Exemplo : using System.Data.SqlClient;

Os quatro principais objetos que compõem um Data Provider ADO.NET são:

- **Connection – Estabelece uma conexão para uma fonte de dados específica**
- **Command – Executa um comando na fonte de dados**
- **DataReader – Provê acesso rápido, forward-only, read-only a dados**
- **DataAdapter – Preenche um DataSet e resolve atualizações com a fonte de dados**

Suporte nativo a SQL Server™, OLEDB, Oracle e ODBC

CEUB
EDUCAÇÃO SUPERIOR

IDbConnection

- Abstrai conexão ao banco de dados físico como SQL Server
- Principal propriedade: string de conexão
  - Série de pares chave/valor (key/value) delimitados por ponto-e-vírgula.
    - Data Source-or- Server - O nome ou endereço de rede da instância do SQL server para o qual conectar
    - AttachDbFilename – Caminho do arquivo de dados, caso não seja feita uma conexão com um servidor.
    - Integrated Security -or- Trusted_Connection - Quando false, User ID e Password são especificados na conexão. Quando true, a conta atual do Windows é utilizada para autenticação

IDbConnection

- **User ID - A conta de login do SQL Server**
- **Password -or- Pwd - A senha para o SQL Server logar**
- **Initial Catalog -or- Database - O nome do banco de dados**

- **Server=myServerAddress;Database=myDataBase;User ID= myUsername;Password=myPassword;Trusted_Connection=False;**

- **Principais métodos: Open() e Close()**
- **Controle de transação**

IDbCommand

- Usada para executar consultas e chamar Stored Procedures
- Principais métodos:

- ExecuteReader – Utilizado para comandos que retornam informações com várias linhas e colunas, executa e retorna um IDataReader

- ExecuteNonQuery – Utilizado para comandos que não retornam informações (Insert, Update, Delete)

- ExecuteScalar – Utilizado para comandos que retornam apenas 1 linha e 1 coluna, ou seja, executa e retorna um "escalar" (um valor simples como string ou número)

IDataReader

- **Representa um conjunto de resultado SQL**
- **Permite a leitura apenas para frente de cursores**
- **Não permite a modificação dos dados lidos**

IDbDataAdapter

- Sabe como carregar tabelas de bancos de dados e como atualizá-las
- Principais propriedades: comandos para consulta e atualização
- Principais métodos:
  - Fill (DataSet e DataTable)
  - Update (DataSet e DataTable)
- Programador pode fornecer comandos para insert/update/delete
- Permite mapeamento entre colunas das tabelas

CEUB
EDUCAÇÃO SUPERIOR

Data Classes

- Objetos que podem armazenar e manipular dados, mas não sabem sua origem
  - DataSet
    - DataTable
    - DataColumn
    - DataRow
    - DataRelation
    - Constraint
  - DataView

CEUB
EDUCAÇÃO SUPERIOR

DataSet

- Funciona como um BD em memória
- É independente da fonte de dados
- Pode conter várias fontes
- Não sabe a origem dos dados
- Tabelas acessadas como array
- Opcionalmente pode ser “tipado”
- Suporta atualização em lotes
- Conecta-se aos dados físicos com a classe DataAdapter

Entity Framework Core

O Entity Framework Core é um framework de mapeamento objeto-relacional (ORM) desenvolvido pela Microsoft.

Ele permite que os desenvolvedores trabalhem com bancos de dados relacionais usando objetos e classes em vez de escrever consultas SQL diretamente.

O Entity Framework Core é uma evolução do Entity Framework original, sendo uma versão mais leve, modular e multiplataforma. Ele foi projetado para ser usado em aplicativos .NET Core e suporte abordagens Code-First e DataBase-First

Entity Framework Core

O Entity Framework Core suporta recursos avançados, como controle de transações, otimização de consultas, carregamento preguiçoso (lazy loading) e migrações de banco de dados.

Ele também oferece suporte a diferentes provedores de banco de dados, permitindo que você trabalhe com uma variedade de sistemas de gerenciamento de banco de dados, como SQL Server, MySQL, PostgreSQL, SQLite e outros.

Entity Framework Core

- **Ele é instalado via pacote Nugets;**
- **Pode ser usado para criar aplicações para web, mobile desktop e cloud**
- **Em síntese ele simplifica o acesso e a manipulação de bancos de dados relacionais em aplicativos .NET, fornecendo uma abstração de alto nível que permite aos desenvolvedores se concentrarem na lógica de negócios, em vez de lidar diretamente com detalhes de banco de dados**

Entity Framework Core

DbContext e o DbSet

O DbContext é uma classe central no Entity Framework Core. Ele representa uma sessão com o banco de dados e é responsável por fornecer funcionalidades para consulta, inserção, atualização e exclusão de objetos no banco de dados.

O DbContext é uma classe que você deve criar no seu aplicativo como uma derivada da classe DbContext fornecida pelo Entity Framework Core. Essa classe derivada do DbContext contém propriedades DbSet para representar as entidades (tabelas) do seu banco de dados e fornece métodos para interagir com essas entidades.

Entity Framework Core - Funcionamento

DbSet

O DbSet é uma classe genérica do Entity Framework Core que representa uma entidade (tabela) no banco de dados.

Cada propriedade DbSet em uma classe derivada do DbContext representa uma tabela ou uma coleção de entidades no banco de dados.

O DbSet fornece métodos para realizar operações comuns, como adicionar, atualizar e excluir entidades, bem como para consultar e filtrar dados.

Entity Framework Core - Funcionamento

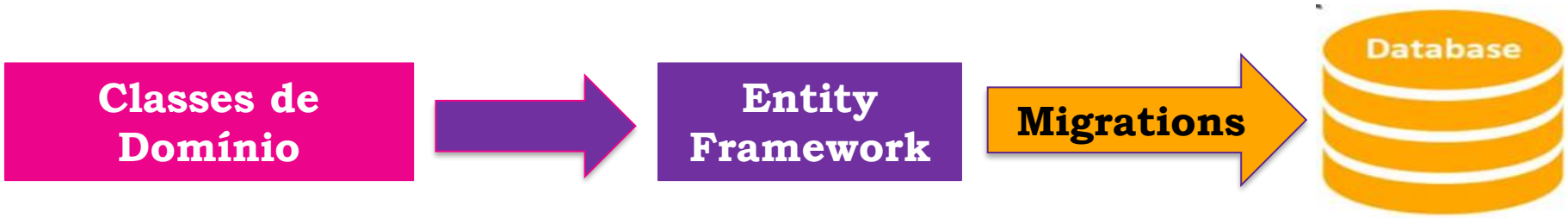
- DbContext é uma classe que representa uma sessão com o banco de dados e fornece funcionalidades gerais para trabalhar com dados.
- Conexão com Banco de Dados
- Operações com dados
- Consulta e Persistência
- Gestão de Transações,
- Etc...

- DbSet é uma classe que representa uma entidade específica (tabela) no banco de dados e fornece métodos para manipular dados dessa entidade.
- Os DbSets trabalham com coleções em memória;
- Para concluir a persistência temos que usar o SaveChanges.

Entity Framework Core – Code First

Code First é uma abordagem de desenvolvimento no Entity Framework Core, na qual você define o modelo de dados usando classes e relacionamentos no código-fonte do seu aplicativo, e o Entity Framework Core gera o banco de dados correspondente automaticamente com base nesse modelo.

Isso é realizado usando Migrations



Entity Framework Core – Code First

Vantagens do Code First no Entity Framework Core:

1. **Desenvolvimento Orientado a Objetos**: O Code First permite que os desenvolvedores trabalhem com objetos e classes em seu código-fonte, em vez de se preocupar diretamente com o banco de dados. Isso facilita o desenvolvimento orientado a objetos e a modelagem do domínio do seu aplicativo.

2. **Produtividade:** Com o Code First, você pode se concentrar na lógica de negócios do seu aplicativo, em vez de escrever scripts SQL manualmente para criar o banco de dados. O Entity Framework Core cuida de criar e atualizar o esquema do banco de dados com base nas classes de entidade definidas no seu código-fonte, o que pode aumentar a produtividade do desenvolvimento.

Entity Framework Core – Code First

Vantagens do Code First no Entity Framework Core:

3. **Flexibilidade**: O Code First oferece uma ampla gama de opções de personalização para modelar o seu esquema de banco de dados. Você pode usar atributos e fluent API para definir propriedades, chaves primárias, chaves estrangeiras, restrições, índices e outros detalhes do modelo de dados.

4. **Migrações de Banco de Dados:** O Entity Framework Core suporta migrações de banco de dados, que permitem que você faça alterações no esquema do banco de dados ao longo do tempo de forma controlada.

5. **Suporte a Diferentes Provedores de Banco de Dados:** O Code First é independente de provedor e suporta uma variedade de sistemas de gerenciamento de banco de dados, como SQL Server, MySQL, PostgreSQL, SQLite e outros.

Entity Framework Core – Code First

Vantagens do Code First no Entity Framework Core:

3. **Flexibilidade**: O Code First oferece uma ampla gama de opções de personalização para modelar o seu esquema de banco de dados. Você pode usar atributos e fluent API para definir propriedades, chaves primárias, chaves estrangeiras, restrições, índices e outros detalhes do modelo de dados.

4. **Migrações de Banco de Dados:** O Entity Framework Core suporta migrações de banco de dados, que permitem que você faça alterações no esquema do banco de dados ao longo do tempo de forma controlada.

1.Suporte a Diferentes Provedores de Banco de Dados: O Code First é independente de provedor e suporta uma variedade de sistemas de gerenciamento de banco de dados, como SQL Server, MySQL, PostgreSQL, SQLite e outros. Você pode alternar facilmente entre diferentes provedores de banco de

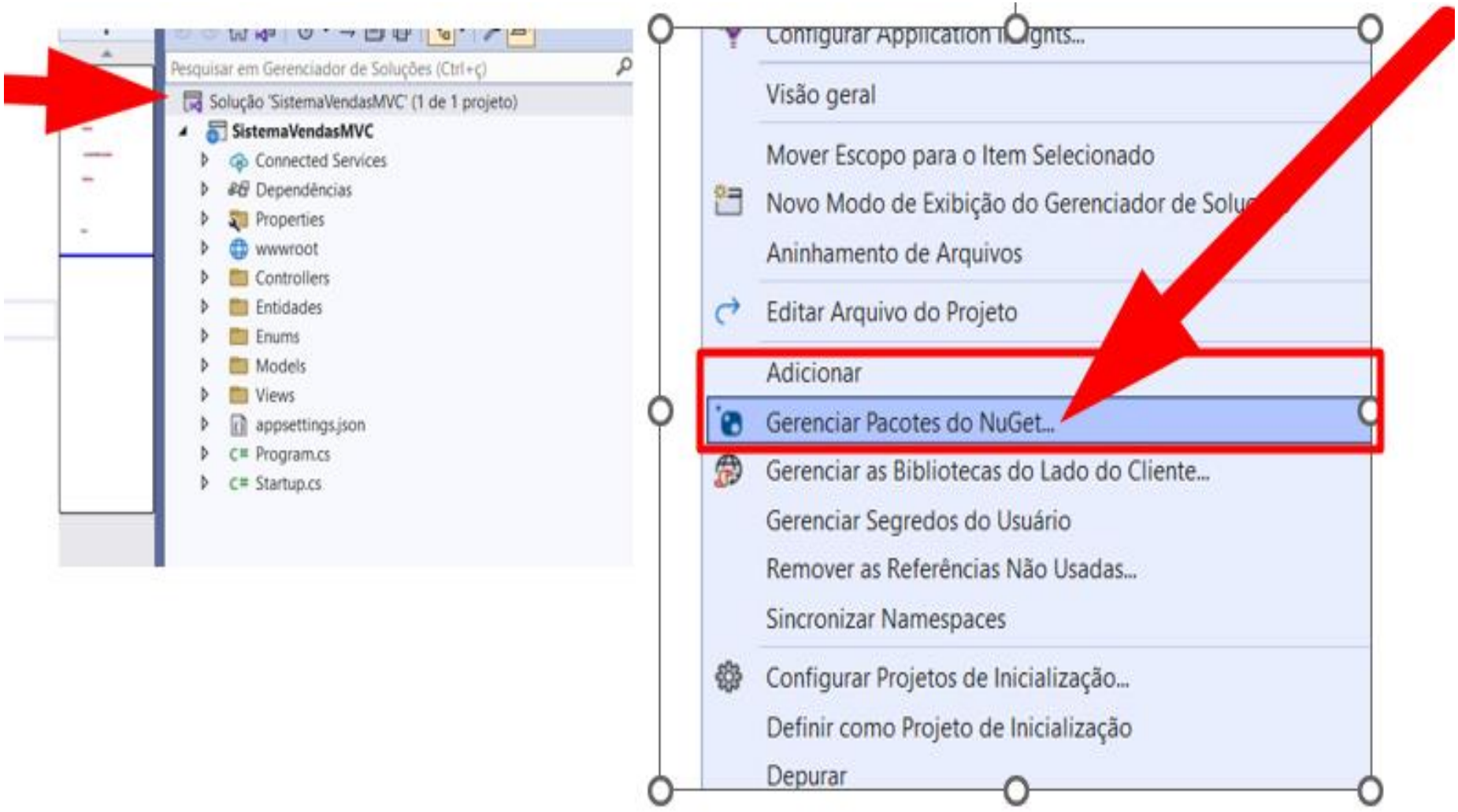
Entity Framework Core – DataBase First

- A abordagem **Database First** é outra abordagem de desenvolvimento disponível no Entity Framework Core.
- Ao contrário do Code First, que começa com a definição do modelo de dados no código-fonte, o Database First começa com um banco de dados existente e gera automaticamente as classes de entidade correspondentes no código-fonte;
- Isso pode ser realizado usando Scaffolding.

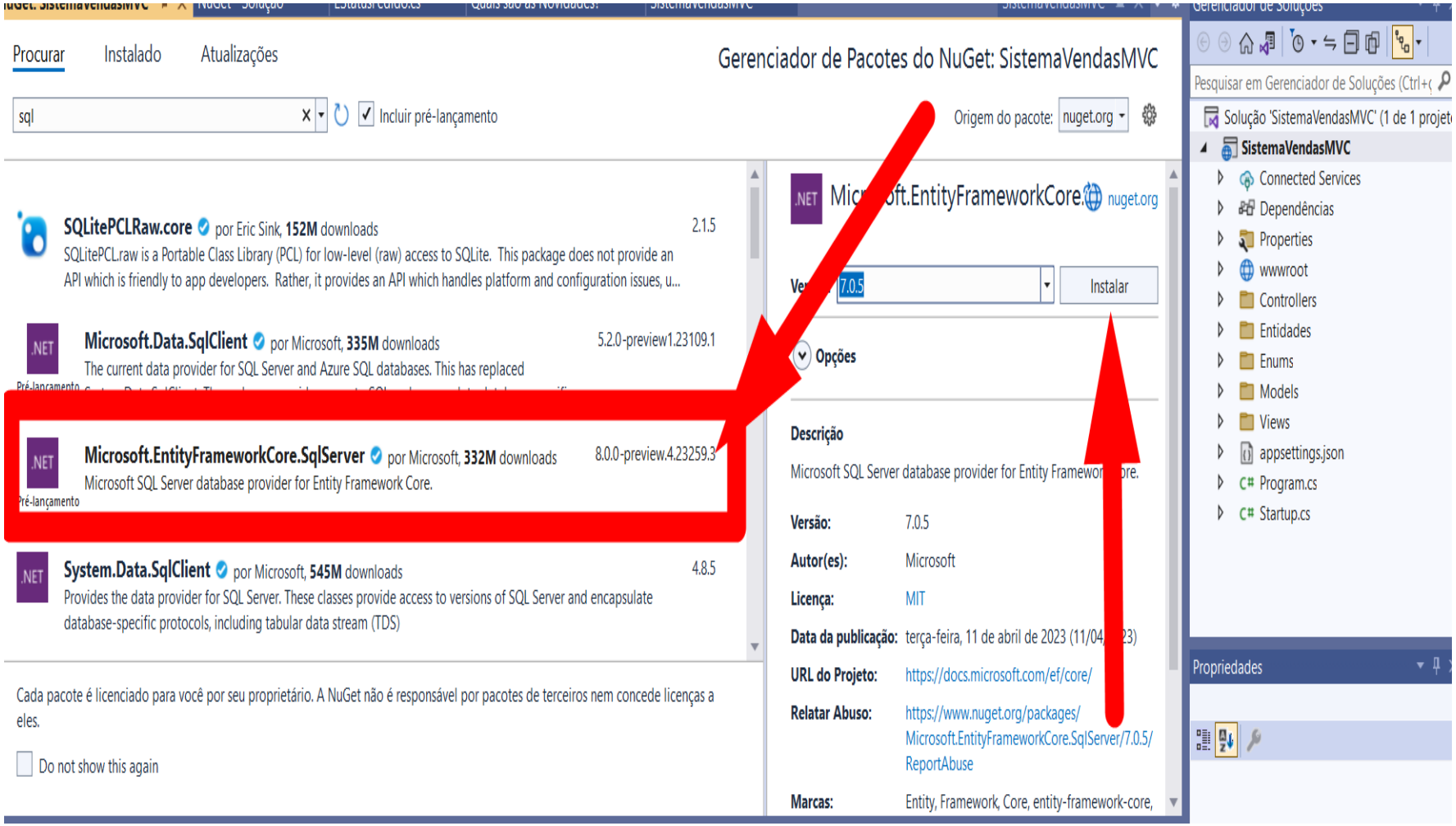
Entity Framework Core – DataBase First

- A abordagem Database First é útil quando você já possui um banco de dados existente e deseja gerar o modelo de dados correspondente no seu projeto.
- Ela permite que você trabalhe com o banco de dados existente sem precisar definir manualmente todas as classes de entidade.
- No entanto, observe que qualquer alteração posterior no banco de dados exigirá uma atualização do modelo gerado usando ferramentas de scaffolding novamente.

Entity Framework Core – Configuração



Entity Framework Core – Configuração



Arquitetura do Projeto

Arquivo appsettings.json

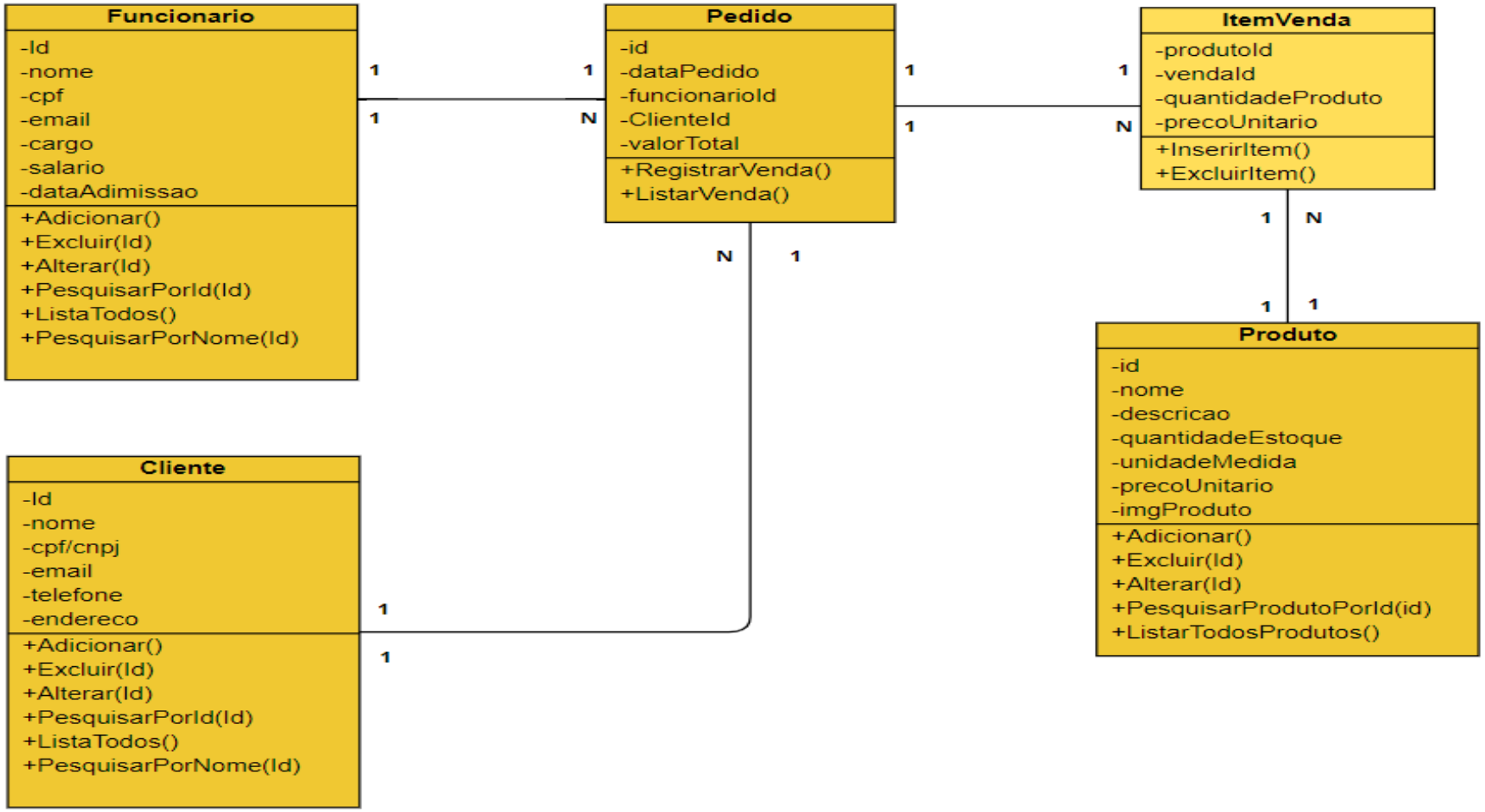
Com o ASP.NET CORE, o arquivo WEB.CONFIG foi praticamente abolido. Sua utilização ainda é recomendada que tivermos casos em que seja necessário hospedar uma aplicação .NET CORE no IIS.

No lugar do WEB.CONFIG foi criado o arquivo AppSettings.JSON, um arquivo de configuração mais moderno, que é suportado no desenvolvimento de novas aplicações .NET.

O appsettings.json substitui as configurações localizadas anteriormente no web.config. O .NET Core suporta arquivos JSON, XML e ini.

1 - EXERCÍCIO

Crie um projeto e implemente as suas classes usando os princípios da abstração e encapsulamento, de acordo com o diagrama abaixo



REFERÊNCIAS

http://www.hardware.com.br/artigos/programacao-orientada-objetos/

http://www.fontes.pro.br/educacional/materialpaginas/C#/arquivos/jdbc/jdbc.php

http://www.dm.ufscar.br/~waldeck/curso/C#

PORTAL EDUCAÇÃO - Cursos Online : Mais de 900 cursos online com certificado http://www.portaleducacao.com.br/informatica/artigos/7852/moderadores-de- acesso#ixzz2AAmxO3JD

http://www.slideshare.net/regispires/C#-08- modificadores-acesso-e-membros-de-classe- presentation

https://www.devmedia.com.br/abstracao-encapsulamento-e-heranca-pilares-da-poo-em-C#/26366