Programação Windows Avançada

ADO.Net

Faculdade Vianna Júnior

Professor: Camillo Falcão

Introdução

- ADO.Net é um conjunto de bibliotecas que permitem a interação com fontes de dados.

- Fontes de dados podem ser:
  - Bases de dados;
  - Arquivos XML;
  - Arquivos texto;
  - Etc.

Introdução

- Existem provedores de dados ADO.Net para diversas bases de dados, tais como:
  - SQL Server;
  - Oracle;
  - DB2;
  - Postgree;
  - MySQL;
- Neste curso nós utilizaremos ADO.Net para interagir com uma base de dados MySQL.

Data providers

- A maioria dos SGBD's existentes possuem provedores de dados (*data providers*) nativos para ADO.Net, ou seja, para acessar uma base de dados MySQL, ao invés de utilizarmos ODBC ou OLEDB, nós podemos utilizar um data provider nativo para MySQL. Ao codificar um data provider para uma determinada base de dados, a equipe de desenvolvimento podem explorar as especificidades de cada base de forma a obter um melhor desempenho.

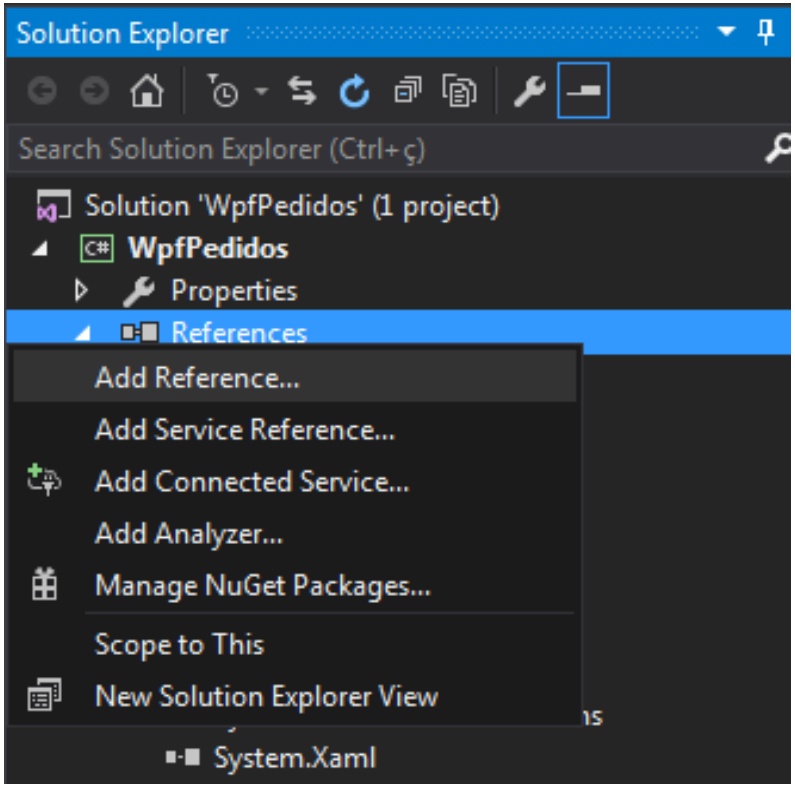
Data provider para MySQL

- Para acessar a base de dados MySql pelo C#, você precisará instalar o *data provider* Connector (http://dev.mysql.com/downloads/connector/net/) se o mesmo já não estiver instalado em seu computador.

- Para essa aula, crie um projeto do tipo “Console Application” no Visual Studio.

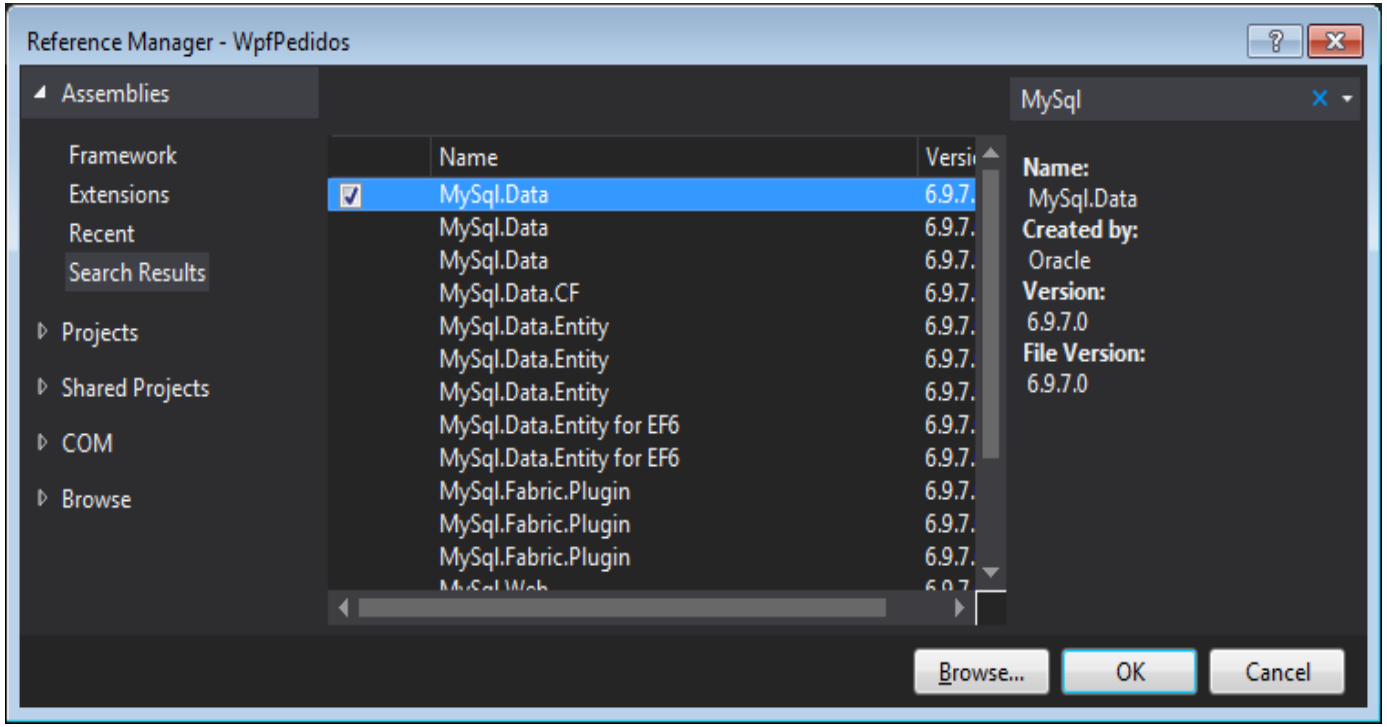
Data provider para MySQL

- É necessário adicionar a referência ao MySql em seu projeto. Para isso, clique com o botão direito do mouse em "*References*" e selecione "*Add Reference*".



Data provider para MySQL

- Na tela apresentada, digite MySql no campo de pesquisa e selecione (como na figura abaixo) a primeira ocorrência de MySql.Data e clique no botão “Ok”.



String de Conexão

- Para se conectar a uma base de dados MySQL, a sua string de conexão deve ser formada pelas chaves:
  - SERVER: endereço do servidor;
  - DATABASE: nome da base de dados a ser acessada;
  - UID: nome do usuário que acessará o MySQL;
  - PASSWORD: senha do usuário.
- Exemplo de *string* de conexão para acesso à base de dados local:

```
string connectionString = "SERVER=localhost; DATABASE=NomeDaBaseDeDados;
UID=NomeDoUsuario; PASSWORD=SenhaDoUsuario;";
```

String de Conexão


```
private string RetornarConnectionString()
{
    #region Essas informações deveriam ser lidas em um arquivo de configuração.
     string server = "localhost";
     string database = "pedidos";
     string uid = "root";
     string password = "aluno";
    #endregion
     string connectionString = "SERVER=" + server + ";" +
            "DATABASE=" + database + ";" + "UID=" + uid + ";" +
            "PASSWORD=" + password + ";";
     return connectionString;
}
```

SQL Connection

- Para se conectar a uma base de dados MySQL, nosso sistema utilizará um objeto MySqlConnection.

- Um objeto MySqlConnection é utilizado para estabelecer a conexão com uma base de dados MySQL.

SQL Connection

- É possível passar para o construtor do objeto MySqlConnection a string de conexão que contém dados como o caminho do servidor, o nome da base de dados, o usuário e a senha de acesso.

```
MySqlConnection conexao = new MySqlConnection(RetornarConnectionString());
```

SQL Connection

- Principais propriedades de um objeto MySqlConnection:

- Principais métodos de um objeto MySqlConnection:
  - O método Open do objeto MySqlConection abre a conexão com a base de dados.
  - O método Close fecha a conexão aberta.

SQL Command

- Um comando em ADO.Net é representado por um objeto.
- No MySql, a classe para instanciação de nossos objetos que representam comandos é chamada de MySqlCommand.
- Um objeto MySqlCommand permite especificar que tipo de interação será feita em uma base de dados.

SQL Command

- Um objeto MySqlCommand necessita ao menos de duas informações:
  - O comando a ser executado (string)
  - Uma conexão com a base de dados (MySqlConnection)

- Um comando pode ser utilizado para alterar dados, obter dados ou mesmo para chamar uma *stored procedure*.

SQL Command

- Para executar um comando que altera a base de dados, basta utilizar o método ExecuteNonQuery() do objeto MySqlCommand, conforme o exemplo abaixo:

```
MySqlConnection conexao = new MySqlConnection(RetornarConnectionString());

string sqlCmd = "insert into Cliente (codigo, nome) values (1, 'Ana')";

MySqlCommand cmd1 = new MySqlCommand(sqlCmd, conexao);
```

SQL DataReader

- Um objeto MySqlDataReader é retornado quando executa-se um comando para leitura sequencial em uma base de dados.

- A classe MySqlDataReader define uma coleção de registros que utiliza a função *Read* para ler o registro atual e apontar para o próximo registro. Existirá ao menos um registro a ser lido enquanto a função *Read* retornar *true*.

SQL DataReader

- Para executar um comando para leitura da base de dados, o método ExecuteReader() do objeto MySqlCommand pode ser chamado:

```
MySqlConnection conexao = new MySqlConnection(RetornarConnectionString());
string sqlCmd = "select codigo, nome from Clientes";
MySqlCommand cmd1 = new MySqlCommand(sqlCmd, conexao);
MySqlDataReader dr = cmd1.ExecuteReader();
while(dr.Read())
{
    Console.WriteLine("Código: {0}, Nome: {1}", dr["codigo"], dr["nome"]);
}
dr.Close();
conexao.Close();
```

SQL DataReader

- Cada campo retornado pela sua query poderá ser lido utilizando-se o seu índice (iniciado por zero) ou o nome de seu campo.

```
MySqlConnection conexao = new MySqlConnection(RetornarConnectionString());
string sqlCmd = "select codigo, nome from Clientes";
MySqlCommand cmd1 = new MySqlCommand(sqlCmd, conexao);
MySqlDataReader dr = cmd1.ExecuteReader();
while(dr.Read())
{
    Console.WriteLine("Código: {0}, Nome: {1}", dr[0], dr[1]);
}
dr.Close();
conexao.Close();
```

DataTable

- Um objeto DataTable representa uma tabela ou visualização de dados, sendo que cada tupla (registro) é representada por uma linha (Row). O acesso aos campos de cada registro (Row) existente no DataTable é feito através da passagem de uma *string* contendo o nome do campo entre colchetes.

MySqlDataAdapter

- Um objeto MySqlDataAdapter é utilizado para controlar a interação de dados entre um DataTable e a fonte de dados. O método Fill do objeto DataAdapter é utilizado para preencher um DataTable com o resultado do MySqlCommand passado no construtor do objeto MySqlDataAdapter.

Exemplo DataTable


```
MySqlConnection conexao = new MySqlConnection(RetornarConnectionString());
string sqlCmd = "select codigo, nome from Clientes";
MySqlCommand cmd1 = new MySqlCommand(sqlCmd, conexao);
conexao.Open();
DataTable tabela = new DataTable();
MySqlDataAdapter da = new MySqlDataAdapter(cmd);
da.Fill(tabela);
conexao.Close();
foreach (DataRow row in tabela.Rows)
{
  Console.WriteLine("Código: {0}, Nome: {1}", row["codigo"], row["nome"]);
}
```

DataSet

- Um objeto DataSet representa um conjunto completo de dados, incluindo:
  - Tabelas;
  - Restrições;
  - Relações entre tabelas.

Exemplo DataSet


```
MySqlConnection conexao = new MySqlConnection(RetornarConnectionString());
string sqlCmd = "select codigo, nome from Clientes";
MySqlCommand cmd1 = new MySqlCommand(sqlCmd, conexao);
conexao.Open();
DataSet ds = new DataSet();
da.Fill(tabela, "Clientes");
conexao.Close();
foreach (DataRow row in ds.Tables["Clientes"].Rows)
{
  Console.WriteLine("Código: {0}, Nome: {1}", row["codigo"], row["nome"]);
}
```