



Ambientes de Desenvolvimento Avançados

<http://www.dei.isep.ipp.pt/~jtavares/ADAV>

Aula Teórico-Prática
Engenharia Informática

2005/2006

José António Tavares
jrt@isep.ipp.pt

Programação com ADO.NET

2005/2006

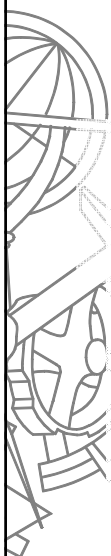
ADAV
Ambientes de Desenvolvimento Avançados

2

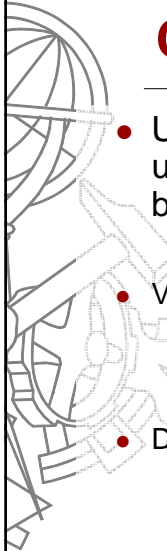


Conteúdo

1. Desenho de aplicações centradas em dados
2. Arquitectura ADO.NET
3. Ligação a Base de Dados
4. Operações ligadas à Base de Dados
5. Construção de *Data Sets*
6. Ler e Escrever XML com ADO.NET
7. Construir *Data Sets* a partir de Bases de Dados




-
1. **Desenho de aplicações centradas em dados**
 2. Arquitectura ADO.NET
 3. Ligação a Base de Dados
 4. Operações ligadas à Base de Dados
 5. Construção de *Data Sets*
 6. Ler e Escrever XML com ADO.NET
 7. Construir *Data Sets* a partir de Bases de Dados



Cenários Conectados

- Um cenário conectado é aquele no qual os utilizadores estão permanentemente ligados à bases de dados
- Vantagens:
 - É mais fácil exercer segurança ao nível do ambiente
 - A concorrência é mais fácil de controlar
 - Os dados estão mais actualizados que nos outros cenários
- Desvantagens
 - É necessário haver uma ligação constante ao servidor
 - Escalabilidade

2005/2006 ADAV 5
Ambientes de Desenvolvimento Avançados



Cenários Desconectados

- Num ambiente desconectado, um sub-conjunto de dados pode ser copiado e modificado independentemente e mais tarde as alterações podem ser introduzidas de novo na base de dados
- Vantagens
 - Pode-se trabalhar a qualquer altura e pode-se efectuar uma ligação à base de dados apenas quando necessário
 - Outros utilizadores podem usar os recursos
 - Este tipo de ambientes aumenta a escalabilidade e desempenho das aplicações
- Desvantagens
 - Os dados nem sempre estão actualizados
 - Podem ocorrer conflitos de dados que têm que ser resolvidos

2005/2006 ADAV 6
Ambientes de Desenvolvimento Avançados



O que é o ADO.NET?

- ADO.NET é a solução de acesso a dados em .NET
 - Faz parte integrante *Framework* .NET
 - Substitui completamente o ADO 'clássico'
- Constitui uma evolução ao 'clássico' ADO
 - Existem similaridades mas mudou muito
- Suporta os modelos conectados e desconectados
 - **DataReaders** para acesso conectados a dados
 - **DataSets** e **DataAdapters** para desconectados
- Baseado em "*pluggable Data Providers*"
 - SQL Server, Oracle, OLE DB & ODBC *standard*

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

7



ADO.NET

A nível do armazenamento de Dados, o ADO.NET suporta vários tipos:

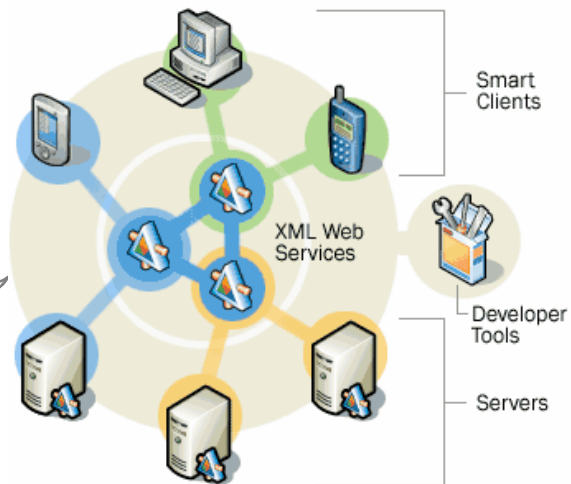
- Não estruturados;
- Estruturados, não-hierárquicos
 - Ficheiros CSV (Comma Separated Value), Folhas Microsoft Excel, Ficheiros Microsoft Exchange, ...
- Hierárquicos
 - Documentos XML e outros
- Bases de Dados Relacionais
 - SQL Server, Oracle, Access, ODBC, ...

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

8

Visão .net

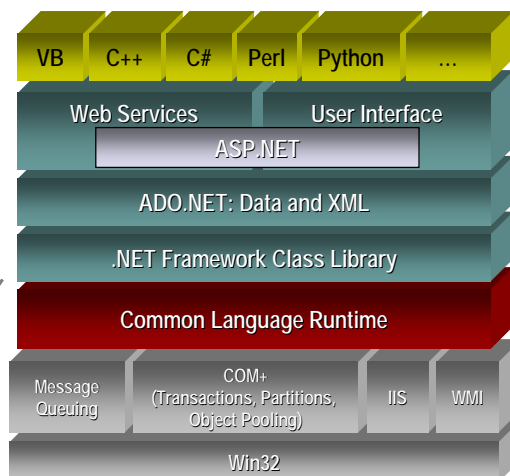


2005/2006

Ambientes de Desenvolvimento Avançados

9

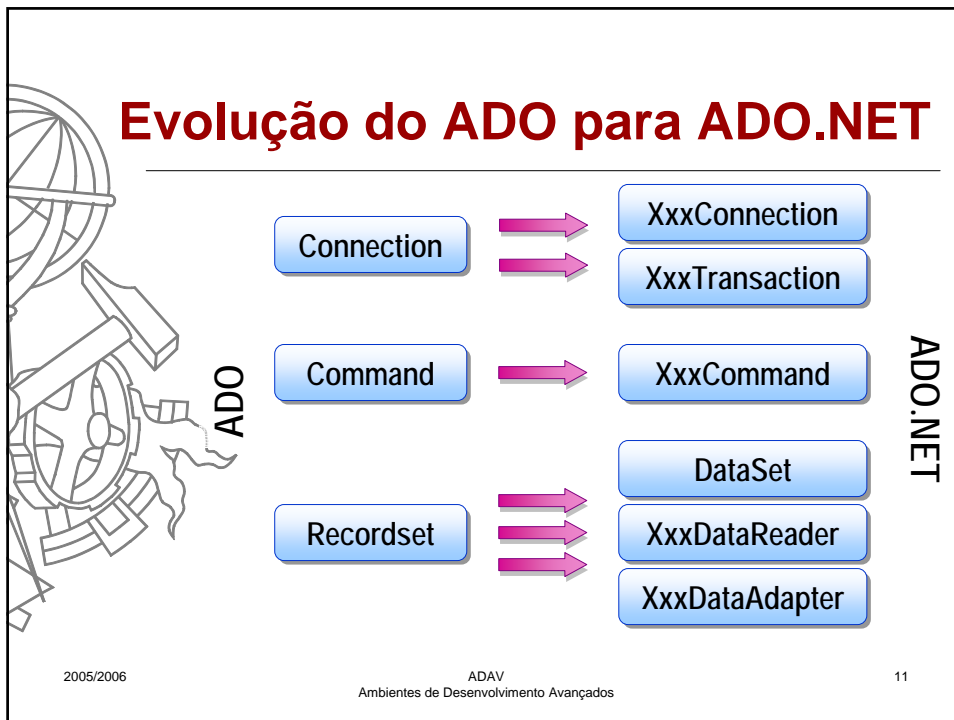
ADO.NET e a *Framework* .NET



2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

10



ADO vs. ADO.NET 1/2

ADO

- Desenhado para acessos conectados
- Amarrado ao modelo físico dos dados
- **RecordSet** é o contendor central de dados
- **RecordSet** é uma (1) tabela que contem todos os dados
 - Obter dados de > 1 tabela requer um comando JOIN
 - Os dados "são planos": perde relações e a navegação é sequencial
- Os tipos de dados estão limitados aos tipos de dados COM/COM+
- A partilha de dados é via COM *marshalling*
- Problemas com *marshalling* através *firewalls* (DCOM, binários)

2005/2006 ADAV 12
Ambientes de Desenvolvimento Avançados



ADO vs. ADO.NET 2/2

ADO.NET

- Desenhado acessos desconectados
- Pode modelar dados logicamente (em memória)!
- **DataSet** substitui o **RecordSet**
- **DataSet** pode conter múltiplas tabelas
 - Obter dados de > 1 tabela não requer um comando JOIN
 - Relações são preservadas e a navegação é relacional
- Os tipos de dados estão apenas limitados por esquemas (*schema*) XML
 - Não é necessário conversão de tipo de dados
 - XML, tal como HTML, é texto: “*Firewall friendly*”

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

13



Benefícios do ADO.NET

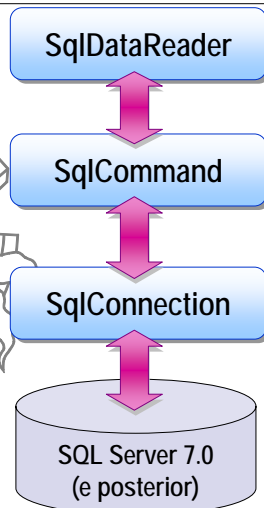
- Interoperacionalidade através do uso de XML
 - *Standard* aberto
 - Texto legível e compreensível pelos humanos
 - Dados descrevem-se a si mesmos (XML)
 - Usado transferir todos os dados em ADO.NET
- Escalabilidade através de **DataSet** desconectados
 - Conexões não são mantidas por longos períodos
 - “*Database locking*” não existe
 - Works the way the Web works: “Hit and Run!”
- “*Maintainability*”
 - Separação entre o tratamento de dados e a “*user interface*”

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

14

Usar as classes ADO.NET num cenário conectado



Num cenário conectado, os recursos são mantidos no servidor até a ligação ser fechada

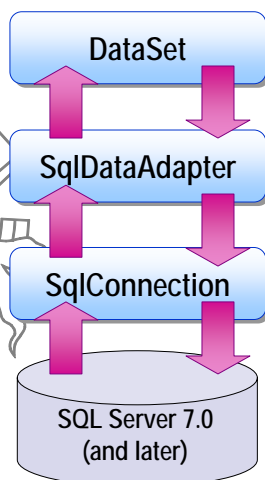
1. Abrir ligação
2. Executar comando
3. Processar linhas no *reader*
4. Fechar *reader*
5. Fechar ligação

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

17

Usar as classes ADO.NET num cenário desconectado



Num cenário desconectado, os recursos não são mantidos no servidor durante o processamento dos dados

1. Abrir a ligação
2. Encher o DataSet
3. Fechar a ligação
4. Processar o DataSet
5. Abrir a ligação
6. Actualizar a fonte de dados
7. Fechar a ligação

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

18

Arquitectura do ADO.NET

Esta animação descreve a arquitectura do ADO.NET considerando ambos os modelos: conectado e desconectado

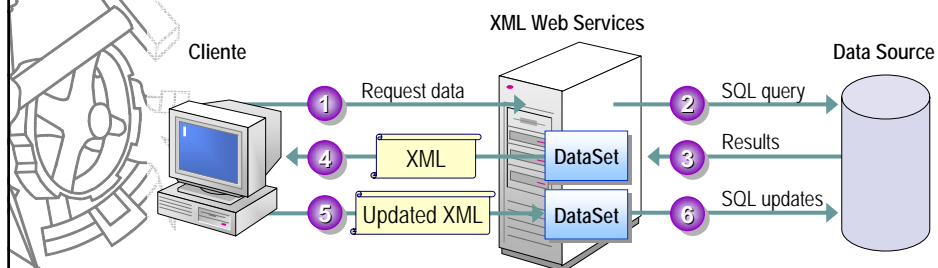


2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

ADO.NET e XML

O ADO.NET é totalmente integrado com XML

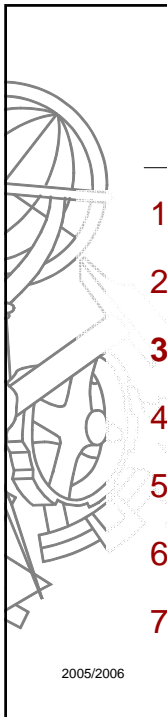


Utilização do XML numa aplicação ADO .NET desconectada

2005/2006

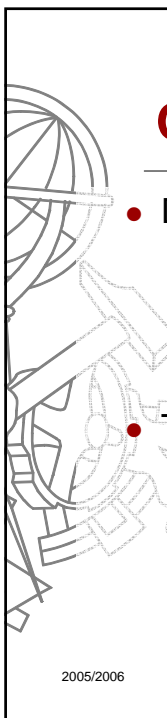
ADAV
Ambientes de Desenvolvimento Avançados

20



1. Desenho de aplicações centradas em dados
2. Arquitectura ADO.NET
3. **Ligação a Base de Dados**
4. Operações ligadas à Base de Dados
5. Construção de *Data Sets*
6. Ler e Escrever XML com ADO.NET
7. Construir *Data Sets* a partir de Bases de Dados

2005/2006 ADAV 21
Ambientes de Desenvolvimento Avançados

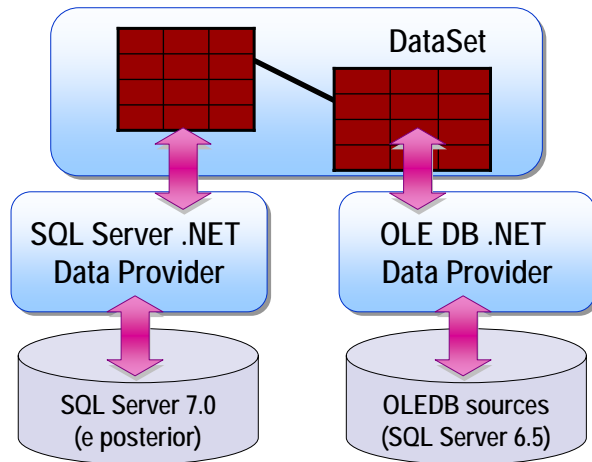
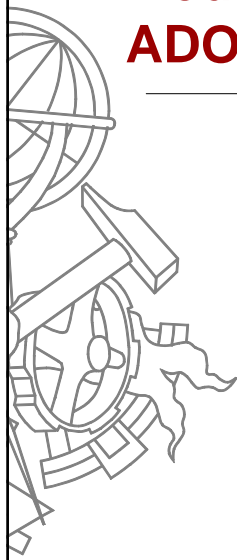


O que são .NET *Data Providers*

- Definição
 - Um .NET "*data provider*" é um conjunto de classes que pode ser usado para efectuar ligações a bases de dados, manipular e actualizar os dados
- Tipos de .NET *data providers*
 - SQL Server .NET Data Provider
 - OLE DB .NET Data Provider
 - ODBC .NET Data Provider
 - Outros (DB2/400, MySQL, ...)

2005/2006 ADAV 22
Ambientes de Desenvolvimento Avançados

Modelo de objectos do ADO.NET

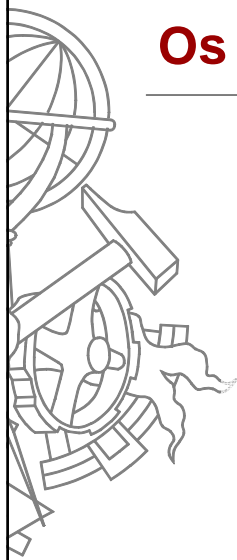


2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

23

Os Namespaces

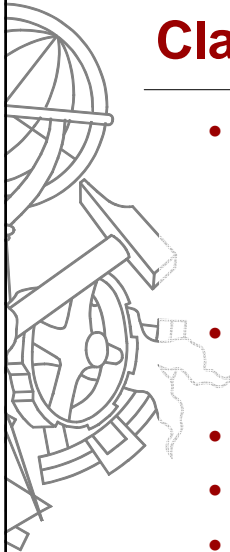


- System.Data
- System.Data.Common
- System.Data.SqlClient
- System.Data.OleDb
- System.Data.SqlTypes
- System.Xml

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

24



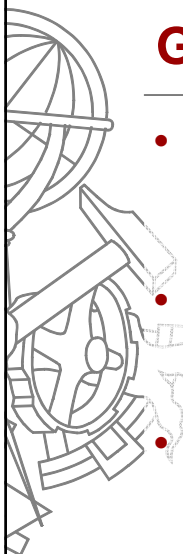
Classes do .NET Data Provider

- XxxConnection – exemplo, SqlConnection
 - XxxTransaction – exemplo, SqlTransaction
 - XxxException – exemplo, SqlException
 - XxxError – exemplo, SqlError
- XxxCommand – exemplo, SqlCommand
 - XxxParameter – exemplo, SqlParameter
- XxxDataReader – exemplo, SqlDataReader
- XxxDataAdapter – exemplo, SqlDataAdapter
- XxxPermission – exemplo, SqlClientPermission

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

25



Gestão de conexões

- Abrir e fechar conexões explicitamente:
 - Open
 - Close
- Abrir e fechar ligações implicitamente:
 - Os “**Data Adapters**” podem abrir e fechar as ligações automaticamente sempre que necessário
- O método Dispose:
 - Remove a conexão da “pool” de conexões

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

26

OleDbConnection e SqlConnection

- Criação, abertura e fecho de uma conexão a uma fonte de dados
- Exemplo **OleDbConnection** :

```
String conStr="Provider=Microsoft.Jet.OLEDB.4.0;" +  
"Data Source=NWIND_RW.MDB";  
OleDbConnection aConn = new OleDbConnection(conStr);  
aConn.Open();  
// acesso, inserção e actualização de dados  
aConn.Close();
```

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

27

Pooling de Conexões

- A abordagem desconectada melhora significativamente a escalabilidade
 - No entanto, requisitar frequentemente conexões degrada o desempenho das aplicações
 - O “*pooling*” de conexões permite uma ‘*pool*’ de conexões idênticas partilhadas por múltiplos utilizadores/aplicações
 - Escalável e rápido
 - Os maiores benefícios são para as aplicações Web / n-tier
- ADO.NET usa “*pooling*” por defeito

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

28

Pooling de Conexões com o SQL Server

Esta animação descreve como é que o *pooling* de conexões funciona com o Microsoft SQL Server 2000



2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

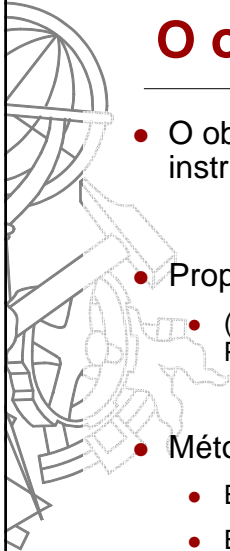


1. Desenho de aplicações centradas em dados
2. Arquitectura ADO.NET
3. Ligação a Base de Dados
4. **Operações ligadas à Base de Dados**
5. Construção de *Data Sets*
6. Ler e Escrever XML com ADO.NET
7. Construir *Data Sets* a partir de Bases de Dados

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

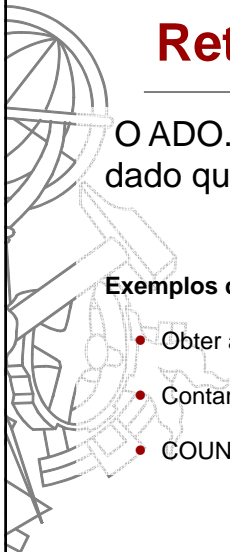
30



O objecto COMMAND

- O objecto *command* é uma referência a uma instrução de SQL ou *Stored Procedure*
- Propriedades
 - (Name), Connection, CommandType, CommandText, Parameters
- Métodos
 - ExecuteScalar, ExecuteReader, ExecuteNonQuery
 - ExecuteXmlReader (só para o SqlCommand)

2005/2006 ADAV 31
Ambientes de Desenvolvimento Avançados



Retornar só um registo

O ADO.NET é mais eficiente que o ADO “clássico”, dado que este retorna um *RecordSet* completo

Exemplos de utilização:

- Obter as unidades em stock para um determinado produto
- Contar os produtos
- COUNT, MAX, MIN, AVERAGE

Exemplo

2005/2006 ADAV 32
Ambientes de Desenvolvimento Avançados



Retornar múltiplos registros

- DataReader
 - Read-only
 - Forward-only
- O método ExecuteReader retorna um **DataReader**

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

33



Exemplo Data Reader

```
Dim cmProducts As New SqlCommand( _  
    "SELECT ProductName, UnitsInStock " & _  
    "FROM Products", cnNorthwind)  
cnNorthwind.Open()  
Dim rdrProducts As SqlDataReader  
rdrProducts = cmProducts.ExecuteReader()  
Do While rdrProducts.Read()  
    ListBox1.Items.Add(rdrProducts.GetString(0))  
Loop  
rdrProducts.Close()
```

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

34

Instruções de Inserção e Actualização de Dados

```
Dim cmd As SqlCommand = New SqlCommand(_
    "dbo.IncreaseProductPrices", cnNorthwind)
cmd.CommandType = CommandType.StoredProcedure
cnNorthwind.Open()

Dim affected As Integer = cmd.ExecuteNonQuery()
cnNorthwind.Close()
MessageBox.Show("Registos Afectados: " & affected)
```

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

35

Transacções

- XxxConnection – por exemplo, SqlConnection
 - BeginTransaction
- XxxTransaction – por exemplo, SqlTransaction
 - Commit
 - Rollback

Exemplo


2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

36

DEMO 01

- Criar uma solução em Visual Studio .NET do tipo *Database*
- Criar uma pequena aplicação que ilustra o modelo conectado



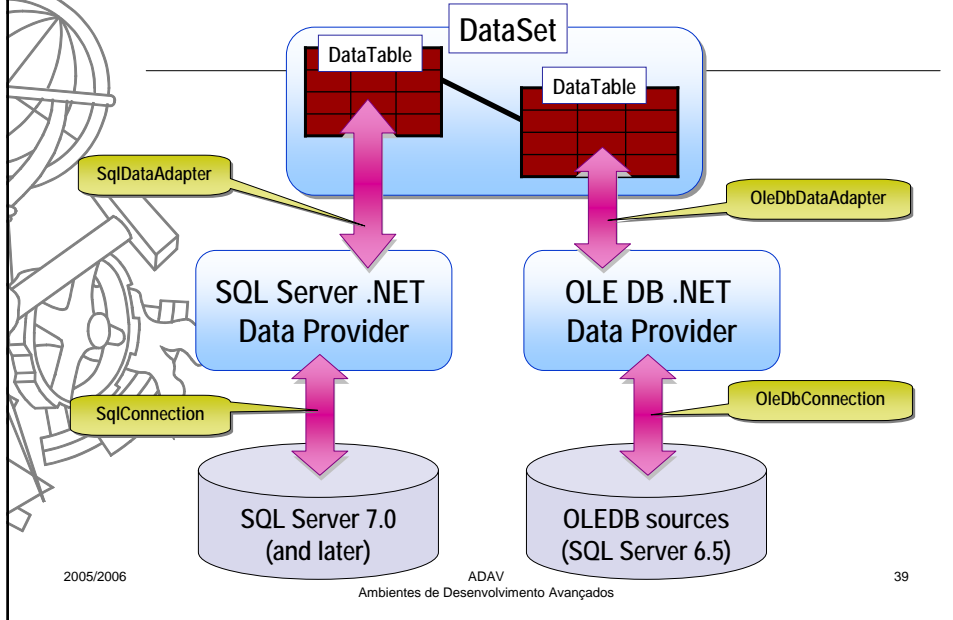
2005/2006 ADAV 37
Ambientes de Desenvolvimento Avançados

Programação com ADO.NET

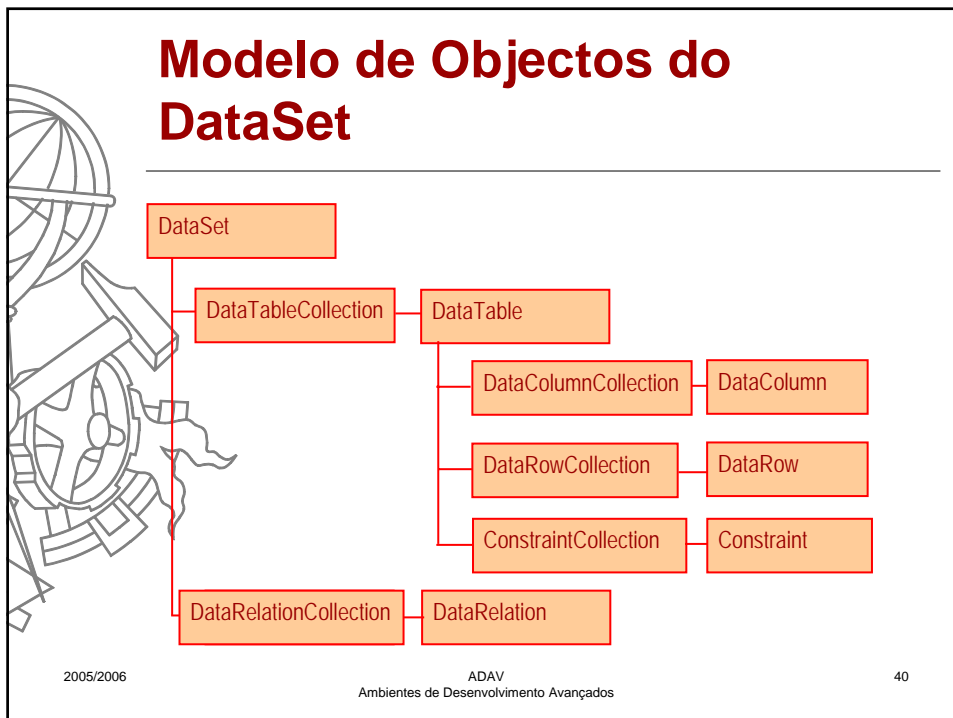
1. Desenho de aplicações centradas em dados
2. Arquitectura ADO.NET
3. Ligação a Base de Dados
4. Operações ligadas à Base de Dados
5. **Construção de *Data Sets***
6. Ler e Escrever XML com ADO.NET
7. Construir *Data Sets* a partir de Bases de Dados

2005/2006 ADAV 38
Ambientes de Desenvolvimento Avançados

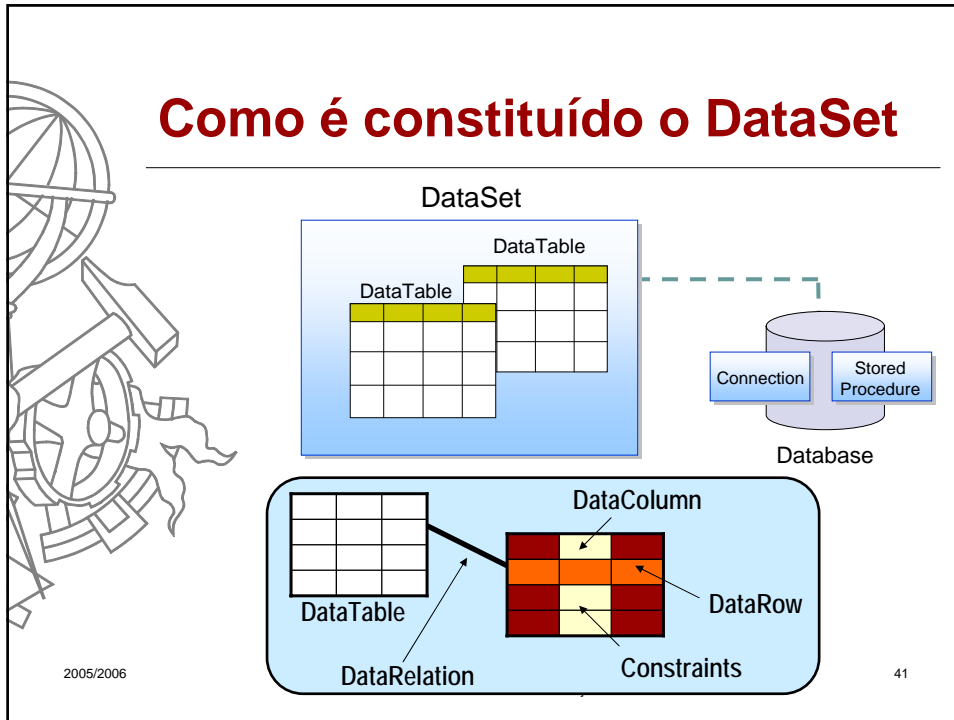
Modelo de Objectos do ADO.NET



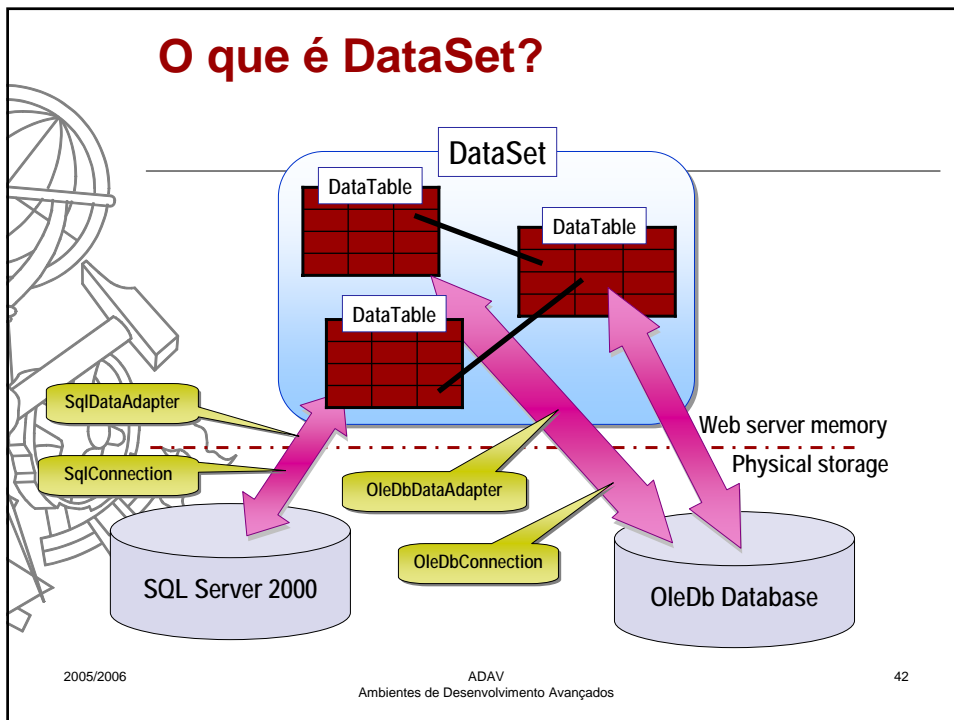
Modelo de Objectos do DataSet



Como é constituído o DataSet



O que é DataSet?





DataSet, DataTable, DataColumn

- **Criar um DataSet**
 - Arrastar o controlo DataSet da Toolbox
- **Criar uma DataTable**
 - Editar a coleção de Tables de um DataSet usando a janela de propriedades
- **Criar uma DataColumn e adicioná-la a uma DataTable**
 - Editar a coleção de Columns de uma DataTable usando a janela de propriedades

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

43



System.Data — DataSet e DataTable

Criar DataTable e adicionar a um DataSet

```
DataSet ds = new DataSet();  
// Create DataTable object: "Customers".  
DataTable dt= new DataTable( "Customers" );  
// Create and add columns to the table  
// 1. Explicitly create and Add a DataColumn  
DataColumn dc = new DataColumn( "CustID", Int16 );  
dt.Columns.Add( dc );  
// 2. Implicitly Create and Add columns (DataColumn).  
dt.Columns.Add( "First_Name", String );  
dt.Columns.Add( "Last_Name", String );  
// Add the DataTable object to the DataSet  
ds.Tables.Add( dt );
```

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

44



Restrição de Chave Primária

- Usar a propriedade `PrimaryKey` da `DataTable`
 - Seleccionar as colunas pela ordem desejada
 - Não permite mudar o nome à restrição
- Editar a colecção de `Constraints` da `DataTable`
 - Adicionar uma `UniqueConstraint`
 - Dar o nome à restrição
 - Seleccionar as colunas
 - Activar a caixa de chave primária



Expressões “Customizadas”

- As expressões “*Custom*” são colunas cujos valores são originados em cálculos e não em valores guardados
- Usar a propriedade `Expression` da `DataColumn`
 - `Sum([Unit Price] * [Quantity])`
- As funções de agregação podem usar relações pai/filho
 - `Avg`, `Count`, `Sum`, `Max`, `Min`

DEMO 02

- Um pequena aplicação que ilustra programaticamente, no modelo desconectado, a criação de *DataSets*



2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

47

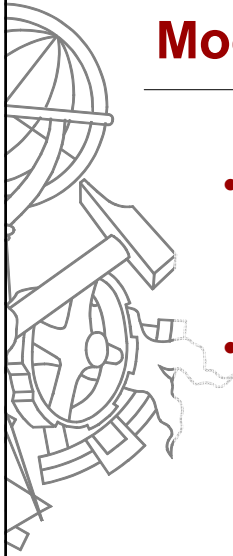
Inserir registos

- Criar uma linha nova
`DataRow drNovoEmpregado = dtEmpregados.NewRow();`
- Preencher a nova linha
`drNovoEmpregado["EmpregadoID"] = 11;`
`drNovoEmpregado["Nome"] = "Nuno";`
- Adicionar a linha a uma DataTable
`dtEmpregados.Rows.Add(drNovoEmpregado);`
- Criar, preencher e adicionar numa só instrução
`dtEmpregados.Rows.Add(new Object[]{11, "Nuno"});`

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

48



Modificar registos

- O método **BeginEdit** da classe **DataRow**
 - Desliga o lançamento de eventos e excepções
- Os métodos **EndEdit** e **CancelEdit** da classe **DataRow**
 - Ligam de novo o lançamento de eventos e excepções



Apagar registos

- O método **Remove** da classe **DataRowCollection** apaga completamente o registo da colecção
`dtEmpregados.Rows.Remove(drEmpregado);`
- O método **Delete** da classe **DataRow** marca o registo como apagado. Este fica escondido, mas acessível, se necessário
`drEmpregado.Delete();`

Alterações de dados no DataSet

Uma **DataRow** pode guardar múltiplas versões de valores para cada coluna:

- **DataRowVersion.Current**
 - O valor actual da coluna
- **DataRowVersion.Original**
 - O valor da coluna antes de qualquer alteração ter ocorrido
- **DataRowVersion.Proposed**
 - O valor da coluna durante o ciclo **BeginEdit / EndEdit**
- **DataRowVersion.Default**
 - O valor por defeito em função do "row's state"

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

51

Processamento de Versões de linha

	CURRENT	ORIGINAL	PROPOSED
	White	White	N/A
<code>dataRow.BeginEdit();</code> <code>dataRow["au_lname"] = "Brown";</code>	White	White	Brown
<code>dataRow.EndEdit();</code>	Brown	White	N/A
<code>dataRow["au_lname", DataRowVersion.Current] // Brown</code> <code>dataRow["au_lname", DataRowVersion.Original] // White</code>			

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

52

Processamento do *Row State*

	CURRENT	ORIGINAL	ROW STATE
	White	White	Unchanged
<code>dr["au_l name"] = "Brown";</code>	Brown	White	Modified
<code>dr.AcceptChanges();</code>	Brown	Brown	Unchanged

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

53

Actualizações com o *DataAdapter*

Actualizações são escritas na base de dados usando o método **Update** do **DataAdapter**

- DataAdapter analisa o "*Row State*" para cada linha
- Executa a acção apropriada (*insert*, *update* or *delete*) de acordo com o seu "*row state*"
 - Usa os objectos **Command** atribuidos às suas propriedades **InsertCommand**, **UpdateCommand** e **DeleteCommand**

DataRows in DataTable

DataAdapter Action

RowState = Unchanged	→ Ignore
RowState = Added	→ Use INSERT command
RowState = Modified	→ Use UPDATE command
RowState = Deleted	→ Use DELETE command

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

54



Optimização de actualizações

- Objectos do tipo **DataSet** e **DataTable** suportam o método **GetChanges**
 - **GetChanges** sem argumentos
 - Extrai todas as linhas cujo **RowState** não é **Unchanged**
 - **GetChanges** com argumento do tipo "**RowState**"
 - Extrai apenas as linhas com um **RowState** especificado
 - Permite controlar a ordem em que os "*inserts*", "*updates*" e "*deletes*" são efectuados à base de dados
- ```
dsChanges = ds.GetChanges();
```
- ```
changes = ds.GetChanges( DataRowState.Added );
```

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

55



Usar o método SELECT

- As *DataTables* têm um método de **Select** que permite filtrar as *DataRows* de modo a cumprir com restrições de ordenação e de estado
- Três parâmetros opcionais
 - Expressões de filtragem, como por exemplo, "City='Porto'"
 - Ordenar, por exemplo, "City ASC"
 - **DataRowState**, por exemplo, **Deleted**

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

56

Criar um DataView

- Criar um DataView através da utilização dos controlos
- Criar um DataView programaticamente

```
DataView dvProdutos = new  
    DataView(dsNorthwind.Tables["Products"]);  
dvProdutos.Sort = "UnitPrice";  
dvProdutos.RowFilter = "CategoryID > 4";  
grdProducts.DataSource = dvProdutos;
```

- Aplicar um DataView a um DataTable

```
dvProdutos.Table = dsNorthwind.Tables("Products");
```

Objectos DataView - Exemplo

```
// Code for myTable "Customers" with  
// "Name" column not shown  
DataView view1 = new DataView( myTable );  
DataView view2 = new DataView( myTable );  
  
// Creates Ascending view of Customers by "Name"  
view1.Sort = "Name ASC";  
  
// Set the view to show only modified (original) rows  
view2.RowStateFilter = DataViewRowState.ModifiedOriginal;  
  
// Bind to UI element(s)...  
DataGrid myGrid = new DataGrid();  
myGrid.SetDataBinding( view1, "Customer" );  
  
//...
```

DEMO 03

- Considerando a aplicação do DEMO 02 e programaticamente:
 - Adicionar e actualizar registos
 - Criar *Data Views*



2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

59

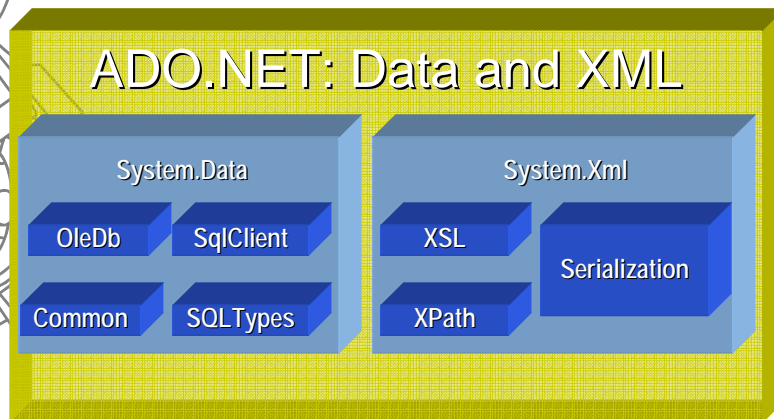
-
1. Desenho de aplicações centradas em dados
 2. Arquitectura ADO.NET
 3. Ligação a Base de Dados
 4. Operações ligadas à Base de Dados
 5. Construção de Data Sets
 6. **Ler e Escrever XML com ADO.NET**
 7. Construir *Data Sets* a partir de Bases de Dados

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

60

ADO.NET: Data e XML



2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

61

XML e ADO .NET

- Método ReadXML

```
dsNorthwind.ReadXml(@"c:\XML\teste.xml",  
XmlReadMode.InferSchema);
```

- Método WriteXML

```
dsNorthwind.WriteXml(@"c:\XML\teste.xml",  
XmlWriteMode.WriteSchema);
```

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

62

DEMO 04

- Utilização de XML
- Um pequena aplicação que ilustra a persistência de *DataSets*



2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

63

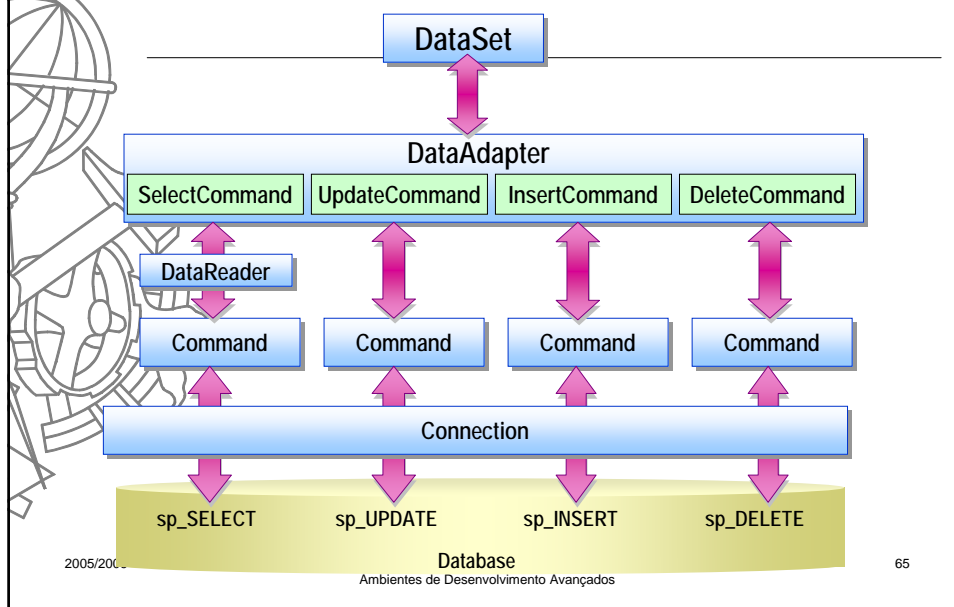
-
1. Desenho de aplicações centradas em dados
 2. Arquitectura ADO.NET
 3. Ligação a Base de Dados
 4. Operações ligadas à Base de Dados
 5. Construção de Data Sets
 6. Ler e Escrever XML com ADO.NET
 7. **Construir *Data Sets* a partir de Bases de Dados**

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

64

Modelo de Objectos do DataAdapter



DataAdapter

- Pode-se preencher um **DataSet** através de um **DataAdapter**
 - Invoca-se o método **Fill** do **DataAdapter**
- O método **Fill** executa o **SelectCommand**
 - Preenche o **DataSet** com a estrutura e conteúdo do resultado do inquérito
- Para otimizar o desempenho
 - aDataSet.EnforceConstraints=False
 - Chamar o método **BeginLoadData** do **DataTable**

DataAdapter - Exemplo

```
// Visual C#  
DataSet dsCustomers = new DataSet();  
dsCustomers.Tables.Add(new DataTable("Customers"));  
  
dsCustomers.Tables[0].BeginLoadData();  
daCustomers.Fill(dsCustomers, "Customers");  
dsCustomers.Tables[0].EndLoadData();  
  
dataGridView1.DataSource =  
    dsCustomers.Tables[0].DefaultView;
```

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

67

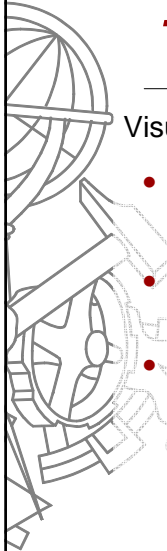
Algumas operações do DataAdapter

- Deve-se usar o método **Merge** para juntar dois DataSets – um original e um que contenha apenas as mudanças ao original
- O método de **Update** de um **DataAdapter** chama o comando apropriado para cada linha alterada (INSERT, UPDATE, DELETE) numa DataTable específica
- O método **AcceptChanges** de um **DataSet** efectua o **Commit** a todas as alterações efectuadas a um **DataSet** específico desde o seu último carregamento ou desde que foi chamado este método pela última vez

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

68

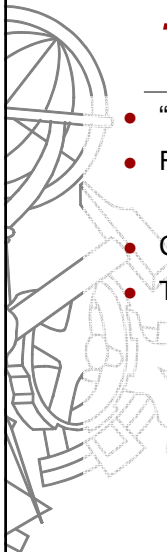


Typed DataSets

Visual Studio .NET pode gerar “*strongly typed* DataSets”

- Classes customizadas derivadas de **DataSet**
 - Proporciona toda a funcionalidade dos DataSet
- Contem classes, métodos, eventos e propriedades específicas relacionadas com os dados em questão
- Por exemplo, um “typed DataSet” pode conter:
 - Uma classe chamada “Customers” derivada de DataTable
 - Uma classe chamada “CustomerRow” derivada de DataRow com propriedades “CustomerID” e “Name”

2005/2006 ADAV 69
Ambientes de Desenvolvimento Avançados



Typed DataSets - Vantagens

- “*Compile-time type checking*”
- Funcionalidades adicionais
 - Por exemplo, o método FindByCustomerID
- Código mais legível e manutenção facilitada
- Total suporte pelo IntelliSense
 - Torna o código mais legível e menos sujeito a erros

```
string s = customerDataSet.Customers[ 1 ].Name;  
comparado com :  
string s = (string) customerDataSet.  
Tables["Customers"].  
Rows[ 1 ].Item[ "Name" ];
```

2005/2006 ADAV 70
Ambientes de Desenvolvimento Avançados

Typed DataSets - Criar

- Visual Studio gera *typed DataSets* a partir definições XML *schema* (XSD)
- Visual Studio oferece varias formas de gerar definições XML *schema*
 - A forma mais fácil é:
 - Seleccionar o "Add New Item" *wizard*
 - Seleccionar a *template* Data Set para adicionar um ficheiro XSD
 - Arrastar uma *table* ou uma "stored procedure" do "Server Explorer" para a área de trabalho
 - E é tudo!

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

71

DEMO 06



2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

72