Implement a Data Access Layer with the Visual Studio 2005 Data Set Designer

Brian Noyes
Chief Architect
IDesign

About Brian

- Chief Architect, IDesign Inc. (www.idesign.net)
- Microsoft Regional Director / MVP
- Writing
  - Data Binding in Windows Forms 2.0, Addison Wesley, January 2006
  - Smart Client Deployment with ClickOnce, Addison Wesley, Fall 2006
  - MSDN Magazine, MSDN Online, CoDe Magazine, The Server Side
  - .NET, asp.netPRO, Visual Studio Magazine
- Speaking
  - Microsoft TechEd US, Europe, Malaysia, Visual Studio Connections,
    DevTeach, INETA Speakers Bureau, MSDN Webcasts
- Participates in Microsoft Design Reviews
- E-mail: brian.noyes@idesign.net
What We Will Cover

- Typed Dataset Overview
- Visual Studio 2005 Capabilities
- Hooking up to tables, views, stored procedures
- Extending the code

Session Prerequisites

- Exposure to ADO.NET data access

Level 300
Typed Dataset Overview

Living with Untyped DataSets

- DataSet is very capable data container
  - Multiple tables
  - Relations
  - Constraints
  - Change tracking
  - Filtering, Sorting, Searching
- Untyped DataSet is bad for maintainability
  - Programming it requires implicit assumptions about contents
**Typed DataSet Overview**

**Programming Untyped DataSets**

- Typical programming model:

  ```csharp
  DataSet ds = GetEmployees();
  DataTable employees = ds.Tables["Employees"]; // Coupling
  DataRow row = employees[0];
  string name = row["LastName"]; // Coupling
  ```

- Problems:
  - How do you know the table/column names?
  - What happens if they change?

**Typed DataSet Overview**

**Defining Typed DataSets**

- Typed DataSet is a type safe wrapper class around the contained schema of a data set

  ```csharp
  public partial class EmployeesDataSet : System.Data.DataSet
  {
    public EmployeesDataTable Employees { get {...}}
    public partial class EmployeesDataTable : System.Data.DataTable, System.Collections.IEnumerable {}
    public partial class EmployeesRow : System.Data.DataRow
    {
      public int EmployeeID { get {...} set {...}}
      public string LastName { get {...} set {...}}
      ...
    }
  }
  ```
Typed DataSet Overview
Using Typed DataSets

Typed DataSets provide a type safe API for accessing the contents of the data set

```csharp
EmployeesDataSet ds = GetEmployees();
EmployeesDataSet.EmployeesDataTable employees = ds.Employees;
EmployeesDataSet.EmployeesRow row = ds.Employees[0];
string name = row.LastName;
```

- Get compile time checking for schema changes
- Requires typed data set regeneration after DB change

Typed DataSet Overview
Using Typed DataSets

Typed DataSets provide IntelliSense for discovering contained schema
Typed DataSet Overview

Using Typed DataSets

Typed DataSets allow you to visualize and maintain the schema in the designer.

Generating Typed DataSets

Visual Studio generates the typed data set class for you.
- Use the DataSet designer to define
- Drag and drop objects from Server Explorer
- Can also generate with xsd.exe command line
- Only get data set classes, no table adapters
**Agenda**

- Typed DataSet Overview
- **TableAdapter Overview**
- Configuring TableAdapters
- Extending DataSets and TableAdapters with Partial Types
- DataSet Designer Limitations

---

**TableAdapter Overview**

**Life Before TableAdapters**

- DataSets are just a container for data
- Need data access code to populate the container
- Need data access code to take changes from the container and persist to database
TableAdapter Overview
Life Before TableAdapters

- DataAdapters bridge between database and DataSet

```csharp
DataSet ds = new DataSet();
SqlConnection conn = new SqlConnection(
    Settings.Default.NorthwindConnectionString);
string query = "SELECT * FROM Employees";
SqlDataAdapter adapter = new SqlDataAdapter(query, conn);
adapter.Fill(ds);
```

```csharp
SqlConnection conn = new SqlConnection(
    Settings.Default.NorthwindConnectionString);
string query = "UPDATE Employees SET ...";
SqlDataAdapter adapter = new SqlDataAdapter(query, conn);
adapter.Update(ds);
```

TableAdapter Overview
Life Before TableAdapters

- DataSet can be untyped or typed
- DataAdapter is type unsafe access to dataset contents
- Still have to write all the surrounding ADO.NET code
- Connection
- Query definition
- Command
- Parameters
**TableAdapter Overview**

**Solution == TableAdapters**

- Type safe data access component for a single typed DataTable
- Encapsulates:
  - DataAdapter
  - Connection
  - Commands
  - Query definitions
  - Parameters
- Automatically generated for you by designer

---

**Demonstration #1**

*demo*

Working with Typed DataSets and Table Adapters
**Agenda**

- Typed DataSet Overview
- TableAdapter Overview
- Configuring TableAdapters
- Extending DataSets and TableAdapters with Partial Types
- DataSet Designer Limitations

**Configuring TableAdapters Capabilities**

- Drag and drop from Server Explorer does initial configuration
- Configuration ➔ Code Generation
- Reconfigure/modify through wizard
  - Modify queries
  - Hook up or define new stored procedures
  - Set data access method names
- Add new queries to table adapter
- Special cases (i.e. GetByCustomerID)
Demonstration #2

Configuring Table Adapters

Agenda

- Typed DataSet Overview
- TableAdapter Overview
- Configuring TableAdapters
- Extending DataSets and TableAdapters with Partial Types
- DataSet Designer Limitations
Extending DataSets/TableAdapters

Overview
- Generated classes are partial types
- Can extend by defining additional partial classes
- Have to satisfy partial type restrictions
  - .NET 2.0 feature

Extending DataSets/TableAdapters

Purposes
- Typed DataSets
  - Validation logic
- TableAdapters
  - Connection management
  - Transaction management
  - Dynamic command modifications
Demonstration #3

Extending DataSets and TableAdapters

Agenda

- Typed DataSet Overview
- TableAdapter Overview
- Configuring TableAdapters
- Extending DataSets and TableAdapters with Partial Types
- DataSet Designer Limitations
DataSet Designer Limitations

- Only really applies to DataSets
  - Could use underlying commands to return data readers
- Limited to Microsoft Data Providers
  - SQL Server
  - OLE DB
  - ODBC
  - Microsoft’s Oracle provider (8i, 9i)

Session Summary

- Visual Studio 2005 DataSet designer is a code generation tool
  - Writes ADO.NET code for you
- When working with DataSets:
  - Always use Typed DataSets unless you cannot know the schema at design time
  - Favor the use of table adapters over writing the data access code yourself
    - Major time saver
- Very flexible and capable
Next Steps

- Evaluate your designs – are you using untyped DataSets?
  - Fix it
- Favor stored procedures for data access
  - Define parameters consistently across CRUD procs for same logical data entities
- Use TableAdapters whenever possible with Typed DataSets

For More Information

- Data Binding with Windows Forms 2.0, Brian Noyes, Addison Wesley, January 2006.
Questions and Answers

- Submit text questions using the “Ask” button.
- Don’t forget to fill out the survey.
- For upcoming and previously live webcasts: www.microsoft.com/webcasts
- Got webcast content ideas? E-mail us at: http://go.microsoft.com/fwlink/?LinkId=41781
- Today's webcast was presented using Microsoft Office Live Meeting. Get a free 14 day trial http://www.microsoft.com/presentlive

Microsoft