

# VDM302: ADO.NET Entity Framework Designers

Sanjay Nagamangalam

Microsoft Corporation

[sanagama@microsoft.com](mailto:sanagama@microsoft.com)

# ADO.NET Entity Framework - a brief overview

- Evolution of Microsoft's data platform
- Lets developers work with data at a higher level of abstraction
  - Work with data the way your applications want
  - Focus on your domain problem
- Program in terms of a higher level *conceptual model of your data*
  - Conceptual model is mapped to database schema
    - Conceptual model need not be 1:1 to database model
  - Data classes are generated from Conceptual model
  - Eliminates much of the need for data access code

# ADO.NET Entity Framework - a brief overview (cont'd)

- **Layered architecture**
  - Built on top of ADO.NET 2.0 provider architecture
  - EntityClient returns Entity Data Records
  - ObjectServices
    - **Materializes Data Classes from Entity Data Records**
    - **Provides change tracking**
  - Rich query capabilities
    - **LINQ over Entities or Entity SQL (ESQL)**
    - **Query runs on server, results returned in your application model**
- **This session = Entity Framework Tools**
- **Next session = Entity Framework drill down**

*VDM304: Data Modeling and Application Development with the ADO.NET Entity Framework and LINQ Over Entities*

# Vision for Entity Tools

- Reduce pain points, enhance productivity
  - Great tooling is an important part of the equation
    - Modeling & mapping by hand in XML is difficult
  - Provide the most natural tooling experience
    - Make common Entity Framework tasks *really easy*
    - Align with standard Visual Studio paradigms
    - Principle of *least surprise*

# Tools Target Early Adopters

- **Deep Visual Studio integration**
  - Project system, item template, wizard, tool windows
  - Supports various project types: Console, WinForms, ASP.NET, ...
- **Modeling & Mapping**
  - Boxes & lines designer
  - Generate & validate model via EF runtime & providers
  - Complex Mapping
  - CTP2: Map entities using stored procedures
  - CTP2: Update model from database
- **Command line tools & public APIs**

# Public APIs & Command line

- **Public APIs**
  - Generate model and 1:1 mappings from database
  - Leverages EF provider model
  - Generate C# or VB classes from model
  - Validate model & mappings
- **EdmGen.exe wraps public APIs**
  - (so do the designer & wizard)

# Create model from an existing database

- Designer can create model, 1:1 mappings and classes from database
  - Uses EF provider model & DDEX
  - Uses public APIs to generate model
  - Use generated classes immediately
  - Visually layout diagram
- Great starting point for early adopters
  - Tweak mappings as needed

# Update model when the database changes

- What happens when database changes after model is generated?
  - Typical during iterative development
  - Designer lets you manually “*Update model from database*”
    - (coming in CTP2)
- Shows user new/changed/deleted objects
  - Updates model and mappings as needed
    - (user can choose in GUI)
  - Customizations to the model are preserved

# Designer support for Complex Mappings

- **View and Edit mappings for Entities and Associations**
  - Editing in the designer automatically “fixes” mappings
- **Map an entity hierarchy**
  - To a single table (TPH) with conditions
  - To multiple tables (TPT)
  - Mix TPH and TPT in the same hierarchy
- **Split an Entity across multiple tables**
- **Map an Entity using stored procedures**
  - Map to stored procedures for query, insert, update, delete
  - (coming in CTP2)

# Demo

- **Designer demo**
  - Generate model and mappings
  - GUI elements
  - Generated classes
- **Mapping demo**
  - Reshape entity
  - TPH mapping
  - Entity splitting
- **Simple Test app**

# Plugging in other data sources

- Create an ADO.NET Entity Framework provider
  - Low bar to extend existing 2.0 provider
    - Entity Framework uses existing 2.0 provider for existing APIs
  - Use EDM to map conceptual model of tables/columns to store schema
- Create (or extend) DDEX provider and deploy as needed
- Designer
  - Uses DDEX to discover and work with EF providers
  - Uses public APIs to generate & validate model & mappings
- Designer doesn't require anything special of providers
- Examples & more info at <http://msdn.com/data>

# Demo

- Generate a model with the designer
  - With custom DDEX and EF provider
- Breakpoints
- Simple Test app

# Summary

- We took a quick look at ADO.NET Entity Designer & Tools
  - Targets early adopters, enables core scenarios
  - Makes common ADO.NET Entity Framework tasks easy
  - Leverages ADO.NET provider model to support 3<sup>rd</sup> party providers

# Questions?

- For more information go to <http://msdn.com/data>
- Come see us at booth number 730
- Take a break and come back for

*VDM304: Data Modeling and Application  
Development with the ADO.NET Entity Framework  
and LINQ Over Entities*

at 11:45am today in this room

# Your Feedback is Important

Please fill out a session evaluation form and either put them in the basket near the exit or drop them off at the conference registration desk.

Thank you!