Supporting OData Formatting and Entity Data Models

Brian Noyes
CTO, Solliance (www.solliance.net)
@briannoyes

pluralsight
hardcore developer training

Outline

- OData formats overview
- Entity Data Models (EDMs)
- Implementing an OData Service with Web API

OData Formats

- **OData formatting defines a several standard representations for data from any domain to accommodate CRUD manipulation of that data via REST service calls**
  - Metadata about data types and feeds available
  - Data properties for each entity instance
  - Relation links between entities
  - Action links for actions that can be taken on an entity
- **OData formats include:**
  - ATOM Publishing Protocol
  - JSON-Verbose
  - JSON-Light

ATOM Publishing Protocol

- **XML-based format**
- **Defines a “feed” of data**
  - Often used to expose blog content, news feeds, podcast feeds, etc.
  - Alternative to RSS
- **Also supports representing the CUD (update) operations through the publishing side of the protocol**
- **Each feed is a collection of items**
- **Each item can contain certain metadata tags about the item**
- **Each item contains a collection of properties**
- **Each item can contain hypermedia links for related resources and available actions on the resource**

OData JSON Format

- **Versions 1.0 and 2.0 of OData just had one JSON format**
  - Somewhat more compact representation of the same information in the ATOM format (~40-60%)
  - Accomodated client platforms that could more easily consume JSON than XML
- **Version 3.0 added a lighter weight JSON format**
  - Not as much metadata nor hypermedia links
  - Just focused on conveying the data of the request
  - Referred to as JSON-Light
  - Older format now referred to as JSON-Verbose

Entity Data Models

- **Defines the type system, relationships, and actions that can be expressed in the OData formats**
    - Similar to the type system of .NET, but not 1:1
    - Based on the Conceptual Schema Definition Language (CSDL) defined for Entity Framework models
- **Allows you to define:**
    - EntitySets – resource collections
    - AssociationSets – relationship links
    - EntityContainer – aggregation of EntitySets and their AssociationSets
    - EntityTypes – types of objects in the collections and of the properties on the objects
    - Actions available on a resource type

Defining Entity Data Models

- **Implicitly – ODataConventionModelBuilder**
  - Define what EntitySets you want exposed
  - The builder will reflect on the types for those sets and define the EntityTypes of the EDM through convention based on the collection type and the properties on the objects
  - The builder will follow any navigation properties to add additional types to the model and define the association links
  - Add action links if appropriate
  - Takes very little code
- **Explicitly**
  - Can define exactly what you want to be seen at the wire level
  - Define each EntitySet, EntityType, property, association link, etc.
  - Takes a lot of code
  - Gives you complete control over what is exposed
  - Your wire level model can be different from the .NET model objects that feed it

Implementing an OData Service with Web API

- **Add OData routes to your WebApiConfig.Register method**
  - MapODataRoute extension methods
- **Derive from EntitySetController<T,K> for full CRUD OData services**
  - T is your model type that you are exposing a collection of
  - K is the type of the key property on the entity
  - Derives from ODataController
  - Has abstract and override methods for standard CRUD patterns
  - Makes getting an OData service following simple conventions very easy

Summary

- **OData formatting standardizes the representations of data for any domain for CRUD scenarios**
- **OData and Web API support XML formatting with the ATOM Publishing Protocol**
- **OData and Web API support JSON verbose or light formats**
- **Entity Data Models define the type system, model objects, and relationships as they will be expressed on the wire in Odata format**
- **EDMs have their own type system, as well as support for relationships and actions**
- **The easiest way to define an EDM is with the ODataConventionModelBuilder – implicitly**
- **To implement an OData service with ASP.NET Web API:**
  - Setting up OData routes based on an EDM
  - Using the EntitySetController base class for CRUD operations