Incorporating Live Data using Breeze and the SharePoint REST API

Andrew Connell
http://www.andrewconnell.com
@andrewconnell

pluralsight
hardcore dev and IT training

Outline

- **Why Breeze?**

- **Consuming the SharePoint REST API with Breeze**

- **Adding the Breeze Library to the SPA**

- **Creating a Breeze Data Context**

- **Configuring Breeze to Work with the SharePoint REST API**

- **Defining the SharePoint Metadata for the Breeze Metadata Store**

- **Using Breeze to Create, Read, Update and Delete Items**

- **Addressing SharePoint Relational Lists with Breeze**

- **Adding Client-Side Data Validation with Breeze Validation Directives**

Why Breeze?

- **Think of Breeze as a Client-Side ORM**

    - Like a JavaScript implementation of Entity Framework

- **Value Add:**

    - Client-side cache

    - Query with LINQ-style queries

    - Automatic data validation

    - Automatic hooks into AngularJS

    - Easy navigation to related entities

    - Change tracking + batching up changes

    - Offline change storage

- **Free and open source library with lots of extensions**

Consuming the SharePoint REST API with Breeze

- **Breeze supports**
  - Server-side component
  - Client-side component
  - WebAPI
  - OData

- **Challenges with SharePoint's REST API**
  - Huge metadata description
  - Not a complete implementation of OData
  - Non-standard implementation of relationships

- **To get Breeze to work with SharePoint REST API:**
  - Use the SharePoint Data Service Adapter (from Breeze Labs)
  - Define your own metadata

Summary

- Why Breeze?

- Consuming the SharePoint REST API with Breeze

- Adding the Breeze Library to the SPA

- Creating a Breeze Data Context

- Configuring Breeze to Work with the SharePoint REST API

- Defining the SharePoint Metadata for the Breeze Metadata Store

- Using Breeze to Create, Read, Update and Delete Items

- Addressing SharePoint Relational Lists with Breeze

- Adding Client-Side Data Validation with Breeze Validation Directives