* Course Overview
  + Lightweight, high performance ORM
* Intro
* What is Dapper
  + .NET micro ORM
  + Object Relational Mapper(ORM)
    - Mapping between database and .NET objects
  + ORM vs Micro ORM
    - ORM(entity framework)
      * Does mapping
      * Does sql generation
    - Micro ORM
      * Fast mapping
      * Complete control of sql
  + Features of a Micro ORM
    - Lightweight
    - Fast
    - simple/easy to use
* Key Features of Dapper
  + Query and map
  + Query parameters are parameterized meaning they are susceptible to sql injection attacks
  + Performance
  + Simplified API
    - Query method: maps strongly type objects
    - Query method: maps to dynamic objects
    - Execute method: for commands that don’t return results
  + Works with any database
* Show Me Some Code!
  + //entity framework
    - var contacts = context.Contacts.Where(c => c.Id == id);
    - var contacts = context.Contacts.Where(c => c.Id == id)
    - .Select(c => new {
    - FirstName = c.FirstName, LastName = c.LastName
    - })
    - Linq gets converted to sql behind the scene
  + //dapper
    - var contacts = connection.Query<Contact>(
    - “SELECT Id, FirstName, LastName FROM Contacts WHERE Id = @Id”, new { id });
    - Complete control of sql
    - Parameters are parameterized to prevent sql injections
* Database Setup
  + Store procedure example
    - CREATE procedure [dbo].[GetContact]
    - @Id int
    - AS
    - BEGIN
    - SELECT [Id]
    - , [FirstName]
    - , [LastName]
    - , [Company]
    - , [Title]
    - , [Email]
    - FROM [dbo].[Contacts]
    - WHERE Id = @Id;
    - SELECT
    - Id,
    - ContactId,
    - AddressType,
    - StreetAddress,
    - City,
    - StateId,
    - PostalCode
    - FROM [dbo].[Addresses]
    - WHERE ContactID = @Id;
    - END
  + After tables and stored procedures are created, run the Scip.PostDeployment-seed-data.sql
  + Publish the database
  + Use SQL manager studio or azure data studio
  + Establish connection to the database using Windows Authentication and the database server name
* Summary
* Overview
* Installing Dapper
  + Add new project -> class library to hold data layer
  + Model fields must match the column name in the database
  + Install nuget package dapper
  + Add test runner class, .NET core console app
    - Add reference to data layer
* Repository Skeleton
  + Create the definition for the IContactRepository
  + Implement the methods in the ContactRepository
    - public class ContactRepository : IContactRepository
    - {
    - private IDbConnection db;
    - public ContactRepository(string connString)
    - {
    - this.db = new SqlConnection(connString);
    - }
    - …
    - }
  + Add nuget package ‘Microsoft.Extensions.Configuration’ to the Runner project